

Don't Read Every Cell: Few-Read Fault Localization and Drift Calibration in Analog Crossbars

Prateek Yadav

Abstract—Deployed analog in-memory chips silently lose cells, and the default fix, reading every cell, costs 256 reads for a 256×256 tile. We show that a few random probe reads are enough. A sparse-recovery decoder over random ± 1 probes localizes up to 64 stuck cells in a 256×256 tile with 4 to 12 counted reads at 1% read noise, on a simulated tile that is otherwise ideal. For $k \geq 4$ faults it needs 4.3 to $6.4\times$ fewer reads than a hybrid that we built from a published multi-outlier code (1.9 to $3.2\times$ if its 8 noise-calibration reads are charged), and at least $2\times$ fewer than tuned X-ABFT, which in this regime needs more reads than a full readout. The sharper finding is where the gain comes from: the decoder, not the second read direction. Forward reads alone already need 1.6 to $4.1\times$ fewer reads than the hybrid; joint decoding with transposed reads needs a further 1.3 to $4\times$ fewer at 1% noise, and gains less at higher noise. For drift, a 16-parameter field fitted per tile from a few probe reads restores programmed accuracy. As a digital correction that needs the stored weights, it gains +2.1 to +13.3 pt over the best cheap digital baseline on CIFAR-10 from 1 h to 1 yr (+1.7 to +13.1 pt with the 17 MVMs actually run); applied in the array, it keeps +0.6 to +3.7 pt. Every claim is graded against criteria committed before the counted runs, and all code, data and pre-specified criteria are public.

Index Terms—analog in-memory computing, fault localization, sparse recovery, phase-change memory, conductance drift, calibration

I. INTRODUCTION

Analog in-memory chips compute matrix-vector products (MVMs) inside resistive crossbars. Their cells drift, and some get stuck. A repair step can fix a stuck cell only if it knows where the cell is, and a deployed chip cannot stop to read every cell: a full readout of a 256×256 tile costs 256 reads. A 2026 repair paper calls explicit localization “impractical to perform continuously on-chip” and repairs without a map [1]. We ask the plain question behind that judgment: how many counted reads does it take to find k stuck cells?

The answer is few. A sparse-recovery decoder over random ± 1 probes localizes up to 64 stuck cells in a 256×256 tile with 4 to 12 counted reads at 1% read noise (Table V). For $k \geq 4$ it needs 4.3 to $6.4\times$ fewer reads than a hybrid built from a published multi-outlier code (5.1 to $5.4\times$ at 2% noise). It needs at least $2\times$ fewer reads than tuned X-ABFT, which in this regime needs more reads than a full readout and reaches F1 0.9 in only 2 of 12 (fault count, noise) cells (Sec. V-A).

Prior localizers read in one direction, assume a few faults per block, or give guarantees for bounded noise and output lines only [2]–[5]. Row-plus-column checksums and two-dimensional group testing are classical [6]. Section II positions each work against ours.

A forward read drives the input lines and senses the output lines. A faulty cell shows in one output, and that output names one coordinate. The other coordinate hides in the probe pattern, and the decoder must tell n candidate cells apart. For one fault and no noise, with ± 1 probes, this takes at least 9 reads with the best design and about 16 with random probes (Sec. III-B). With read noise we measured 12 to 16 forward-only reads at 1% noise for every k from 1 to 64 (Table V). In our model this floor comes from cross-talk between candidate cells and not from noise. We support that with a simulation, not a theorem.

A transposed read drives the output lines and senses the input lines, so the second coordinate is also read from an output index. For one fault and no noise, two reads suffice. With noise and up to 64 faults, a joint decoder over both directions needs 4 to 12 reads at 1% noise, 1.3 to $4\times$ fewer than forward-only.

The decoder, not the second direction, delivers most of the gain. The forward-only form of the decoder already needs 1.6 to $4.1\times$ fewer reads than the hybrid, and it passes its own pre-specified bar. We then tested where the second direction stops paying. Two of our pre-specified claims, each asking for a 0.7 reads ratio from the second direction, failed their bars at 2% noise on 256-cell tiles. We report both, with the regime map that explains them. At 2% the joint decoder needs 16, 8, 12 and 12 reads against 16, 12, 16 and 24 for forward-only; at 4% and $k = 1$, forward-only needs fewer reads (48 against 64).

Every claim is graded against its pre-specified test. We committed criteria files before the counted runs and let a script judge them (Sec. IV-D). We re-ran the competitors on their own terms, X-ABFT tuned per row and Wei and Roth with bounded noise. Tables I and IV show the outcome of every claim, including the kills.

Scope. All results are simulation of PCM tiles in one architecture (ResNet-9), with three checkpoint seeds per dataset. Localization uses stuck-at faults on an otherwise ideal tile, with the stored digital weights as reference (Sec. III-A). Section VI-C lists every limit.

Contributions.

- 1) **Few-read localization.** A sparse-recovery decoder over random ± 1 probes, forward-only or joint with transposed reads, localizes up to 64 stuck cells in a 256×256 tile with 4 to 12 reads at 1% noise (Sec. III-A, V-A).
- 2) **A decisive comparison.** It beats a hybrid built from a published multi-outlier code by 4.3 to $6.4\times$ in reads for $k \geq 4$, and beats tuned X-ABFT by at least $2\times$ or X-ABFT misses F1 0.9. Both registered comparisons pass. The Wei–Roth decoder run on its own terms

is INCONCLUSIVE by design, with a stated regime mismatch (Sec. V-A).

- 3) **An attribution.** We separate decoder from read direction: the decoder carries most of the gain, and the second direction helps at low noise only. Two registered claims about it were killed, and a regime map shows where it pays (Sec. V-A).
- 4) **A recovery guarantee.** We prove a sufficient condition for exact recovery: cells that share no row or column are orthogonal in the joint operator (Sec. III-B, App. A). For one fault and no noise, two reads suffice, whereas a forward-only code with ± 1 probes needs at least 9. This is an identifiability argument, not a lower bound over all probe types, and the condition does not prove an advantage over forward-only probing.
- 5) **Drift calibration from 16 parameters.** The field-fit method M4 fits a 16-parameter drift field per tile from 15 probe MVMs plus one bias read (16 MVMs per tile, 17 as run). As a digital correction it restores accuracy; the in-array forms recover less (Sec. III-D, V-C, V-D).
- 6) **Concurrent regression from live inputs.** It localizes faulty input lines from ordinary inference traffic over thousands of inferences; its registered test is INCONCLUSIVE (Sec. V-B).
- 7) **A ledger of what failed.** Limits L1–L21 list each idea that did not survive its test, and why (Sec. VI-C). Code, data and criteria are public.

The rest of the paper is organized as follows. Section II reviews background and related work, Section III describes the methods, Section IV the experimental setup, Section V the results, Section VI discussion and limitations, and Section VII concludes.

II. BACKGROUND AND RELATED WORK

Background. A resistive crossbar stores a weight matrix as cell conductances and computes an MVM in one analog step. A *forward* read drives the input lines and senses the output lines. A *transposed* read does the reverse on the same array. A *stuck* cell no longer follows its programmed conductance. A *counted read* is one probe vector applied once; Sec. IV gives the cost model. We score localization by the cell-level F1 of the decoded fault set.

Drift compensation. GDC [7], [8] and AdaBS [8] are the field’s incumbents; the HERMES chip paper [9] documents the per-weight drift-rate spread that limits GDC. Differential mapping with linear compensation was reported at IEDM 2025 [10] (abstract only; it may remove the common-mode part of drift). Temperature-induced degradation has been shown to be systematic and calibratable on a CMOS neuromorphic ASIC [11]; we do not use that as evidence for 2-D patterns in PCM. Open-loop low-rank branches (VeRA+ [12], HaLoRA [13]) are trained offline; VeRA+ selects its parameter set by a timer. A DoRA-style method uses 10 calibration *samples* to train a digital correction [14]. IBM patents describe in-hardware per-output-line affine correction with reference columns [15] and re-reading fixed inputs at two times to fit a “function” that is applied to later outputs [16].

Fault localization. FAMCroNA reads one row at a time after a global set or reset ($2n$ reads and two writes, noise-free, offline, forward only) [17]. Li et al. demonstrate analog error-correcting codes with redundancy columns, threshold detection on live inputs and single-error column location [18]. Compensation schemes such as SEC-CiM consume an offline fault map and do not localize [19], [20]. HPE patents locate a single erroneous column from an analog syndrome [4], or isolate a cell by successive structured test vectors with binary search [21]; Khezeli et al. compare the stored low-order bits of the output of a test vector against a golden signature to find faulty columns (soft errors, exact digital comparison) [22]; Xia et al. detect stuck-at faults by quiescent-voltage comparison within fault-tolerant training [23]; a recent fault-free-computing scheme assumes the fault positions are known (obtained at programming time; read as the arXiv preprint) and re-decomposes the matrix around them, without estimating the map [24]; Wei and Roth give multiple-outlier output-column codes (restricted isometry, disjunct matrices) in one read direction, as theory with bounded noise [3]; Neural Checksums locate (PE, column) faults by exact digital comparison of two checksums per MVM and correct only a single faulty column or PE [5]; geometric analog codes extend multi-outlier output-entry location with m -height guarantees under bounded noise [25], and weighted checksums with syndrome patterns correct one or two adjacent columns [26]. In digital, noise-free systolic arrays, coprime algebraic test vectors localize one faulty PE per column (row within the column) in one or two passes [27]. STAR [1] repairs without any localization; our closed loop (Sec. VI) is compared only against an oracle map, not against such a no-map baseline.

Closest checksum prior art (X-ABFT). Liu, Xia, Wang and Chakrabarty store two checksum columns per test block (plain and index-weighted), apply dedicated forward test vectors every $\tau = 10,000$ cycles, and recover (row, column, deviation) of up to two faults per block from the weighted-to-plain ratio, at 5% time redundancy in their example (1% faults, 4×16 blocks) [2]. They argue that checksums on the ordinary computation cycles cannot determine the row address, and use live-input mismatches only to correct an output once the cell is known. It reads forward only, assumes at most two faults per block, and models write variation but not per-read noise. The two weighted checksum lines and the ratio decoder are therefore prior art [2], [26], as is forward-only cell-level localization of ≤ 2 faults per block from checksums plus test vectors.

Scope of our localization claims. *Joint decoding:* the forward plus transposed probe pair on the same array, decoded jointly to cells under read noise, for $k \leq 64$ faults per tile; we do not claim compressed sensing or group testing for crossbar faults in general (Wei–Roth). *Concurrent regression:* ordinary inference inputs and no dedicated test vectors, with whitened checksum syndromes regressed across many inferences and a false-alarm-bounded stopping rule. It adds no dedicated test cycles (0% test-time overhead, against the 5% time redundancy X-ABFT reports for its test rounds, a design comparison and not a measured benchmark), at the price of thousands of inferences before it localizes.

TABLE I

EVERY CLAIM, GRADED AGAINST ITS PRE-SPECIFIED TEST. STRENGTH: *REGISTERED PASS* = A CRITERIA FILE COMMITTED BEFORE THE COUNTED RUN, MECHANICAL VERDICT *PASS*; *KILL* = A REGISTERED KILL CLAUSE FIRED; *INCONCLUSIVE* = REGISTERED, NOT ALL CLAUSES MET; *EXPLORATORY* = REPORT-LEVEL NUMBERS, NO REGISTERED CLAUSE. DETAILS ARE IN THE SECTION GIVEN.

Claim	Strength	Section	What bounds it
Joint decoder needs $\geq 2\times / \geq 4\times$ fewer reads than our hybrid built from a published code (1–2% noise)	registered PASS	V-A	Mostly the decoder, not the second direction; 8 calibration reads uncharged
Forward-only decoder needs $\geq 2\times$ fewer reads than the hybrid Decoder vs tuned X-ABFT (outside its design regime)	registered PASS registered PASS	V-A V-A	Same baseline Rests mostly on X-ABFT missing F1 0.9; no positive control
Wei–Roth decoder on its own terms reaches line F1 0.9	NOT_MET; overall inconclusive by design	V-A	Regime mismatch, not a failure of their method
Second direction gives a 0.7 reads ratio (two claims)	KILL (tile 256, 2% noise)	V-A	Advantage at 1% noise is exploratory
At 4% noise, $k \geq 4$, decoder reaches F1 0.93–0.95 where baselines do not	exploratory	V-A	Not a registered clause
Concurrent regression localizes input lines over thousands of inferences	inconclusive (registered); curves exploratory	V-B	Needs thousands of inferences
M4 (ideal form) beats B8 under 16 MVMs per tile	registered PASS (strict count); inconclusive as run (17)	V-C	Needs stored $\mathbf{W}_0$ at inference; CIFAR-100 1h CI includes 0
Deployable M4 recovers accuracy in the array	KILL (free rank-1 form); inconclusive (full form)	V-D	Beaten by 32-image digital recalibration
M4 health estimate improves refresh allocation	inconclusive (bar 15% not met)	V-D	Cost-aware allocation alone gives most of the saving
Estimated map matches oracle map in a closed repair loop	registered PASS (extension: KILL at 4% noise)	VI	No-difference-detected, not equivalence
Sufficient condition for exact recovery	proof sketch	III-B	Not tight; not independently reviewed

Concurrent checking. Row/column checksums of matrix products (ABFT) [6] are the root of the HPE syndrome circuits, Neural Checksums and X-ABFT, which decide per MVM or per dedicated test round; Liu et al. monitor network-level health with a handful of offline-synthesized test patterns (detection only) [28]; an NVIDIA patent monitors crossbar power and estimate a fault *percentage* at changepoints [29]. We found no method that regresses the checksum mismatch on the live inference inputs across many inferences.

Row/column checksum and 2-D group-testing precedents. The mechanism of Fig. 1 is close to classical row-plus-column checksum ABFT of Huang and Abraham [6], which locates an erroneous element at the intersection of a failing row checksum and a failing column checksum, and to two-dimensional pooled group testing [30], where a positive row pool and a positive column pool name a candidate at their crossing and ghosts are resolved by further tests. Neither is new to us. What differs here is the setting and the decoder: dense random probes on an analog array under counted-read accounting and per-read Gaussian noise, a joint noisy sparse-recovery decode (orthogonal matching pursuit [31]; see also [32], [33]) that scores row and column evidence together, and a measured regime map. We claim the regime map and the decoder comparison, not the idea of reading two directions.

Eigen/SVD circuits. Sun and Ielmini’s closed-loop eigen-vector circuit [34] and the later closed-loop circuits of Man-nocci et al. [35], [36] are enabling hardware (full texts read; neither uses a circuit to test or diagnose its own array, and we claim no circuit novelty); pointing the eigen loop at an array’s own deviation matrix as a self-test is a new use of these circuits, and a negative one (L18); “Memristive SVD”

[37] is digitally orchestrated power iteration on stored data with no self-diagnosis.

Table II states explicitly how this work differs. Absence statements mean “not found in the sources we could read” (our literature review read the full claims or full text for most items; where only an abstract was available we say so where the item is cited).

III. METHOD

A. Bidirectional probing and the joint decoder

Terms. A *read* is one probe vector applied once to the array, forward or transposed. A *tile* is an $m \times n$ block of the array ($m = n = 256$ here). *Forward* means: drive the n input lines, sense the m output lines. *Transposed* means: drive the output lines, sense the input lines.

Model. We hold a noise-free digital copy W_{ref} of the programmed weights, the intended values that an accelerator’s digital weight memory keeps. For a probe x_p the residual is the analog read minus $W_{\text{ref}}x_p$. Let $E = W_{\text{now}} - W_{\text{ref}}$; in our model the tile is otherwise ideal (no programming error, no drift), so E has exactly k nonzero cells when k cells are stuck. This is the same digital resource that makes the ideal M4 form an upper reference (Sec. III-D). A forward probe x_p gives $r_p = Ex_p + \epsilon$. A transposed probe y_q gives $r'_q = E^T y_q + \epsilon'$. Here ϵ, ϵ' are white read noise of standard deviation $s = \sigma_c \max |w|$, fresh on every read. We use p_f forward and p_b transposed probes with random ± 1 entries, scaled to unit norm. The cost is $P = p_f + p_b$ reads.

Idealizations. Four assumptions go beyond the probe model. (1) The reference is the stored digital W_{ref} above; we

TABLE II
POSITIONING AGAINST THE CLOSEST PRIOR ART.

Work	What it does	Difference to this paper
IBM US 12,272,422 [16]	Claim 1: input the same activation vectors at two times, read outputs, determine “a function” mapping later to earlier outputs, apply it to later outputs. Dependents: per-column polynomial.	Closest prior art. M4 also re-reads fixed probes against stored reads, but fits a 2-D multiplicative field model that uses the stored weights $\mathbf{W}_0$, probes in both directions, and a counted budget; it is evaluated at network level against GDC/per-row/per-column/bi-scale.
IBM US 11,663,458 [15]	Per-output-line affine correction (γ, β) from reference columns; hardware.	Run as the reference-column baseline (Sec. V-D): deployable M4 is non-inferior to it at every time on both datasets (+2.4 / +4.4 pt at 1 yr, C10 / C100).
VeRA+ / HaLoRA / DoRA-cal. [12]–[14]	Offline-trained low-rank digital branches; VeRA+ switches parameter sets by a timer; no estimate from the aged array.	We estimate the correction from probes of the aged array (sensing) and train nothing. Run as the timer-adaptor baseline (Sec. V-D): deployable M4 is non-inferior at every time; a registered headroom check shows an open-loop timer loses up to 57.6 pt to GDC when the true drift rate deviates $\geq 1.25\times$.
Hadamard-probe preprint [38]	N Hadamard passes recover the full $N \times N$ deviation map; DCT compression; heartbeat reprogramming. Theory only.	We use 15 probes per tile plus one bias read (not N), a digital correction rather than reprogramming, noisy real trained weights, and network accuracy against baselines. The concept “structured probes $\rightarrow$ compressed deviation map $\rightarrow$ periodic refresh” is theirs.
HPE US 11,804,859 / 11,532,356 [4], [21]	Syndrome comparators; one erroneous column per syndrome; test-vector binary search, one fault at a time.	Reimplemented as the forward-only HPE-style baseline for localization and as the per-inference decoder in Sec. V-B. Not pre-empted: one read direction, column level, per-MVM decision.
Neural Checksums [5], NVIDIA [29]	Concurrent checksum per MVM $\rightarrow$ (PE, column), exact digital comparison; concurrent power monitor $\rightarrow$ fault percentage.	Our concurrent comparator CK is <i>our adaptation</i> of this decision rule to analog read noise (Sec. V-B); the originals were not evaluated under analog noise. Concurrent regression differs by regressing the mismatch on the ordinary inference inputs over many MVMs and naming input lines.
X-ABFT [2], Pinto–Touba [26]	Two weighted checksum columns plus dedicated forward test vectors every τ cycles (5% time redundancy example); (row, col) of ≤ 2 faults per block; weighted checksums correct 1–2 adjacent columns.	Prior art for the two checksum lines and the ratio decoder. Concurrent regression uses ordinary inputs and no test vectors, with whitened syndromes regressed over many inferences; X-ABFT judged live-input localization infeasible. Our joint decoder reads a transposed direction, which neither does. We ran X-ABFT as a baseline, with its design and repeat factor tuned per row. It needs at least $2\times$ more reads than the joint decoder on every k at 1% and 2% read noise, or it misses F1 0.9 while the joint decoder reaches it; it also needs more reads than a full readout, and reaches F1 0.9 in only 2 of 12 cells. At 4% it stays below F1 0.6 (registered, PASS, 240 instances per row; regime caveats in Sec. V-A).
Liu et al. [28]	A few offline-synthesized test patterns; network-level health from output confidence; detection only.	Same genre as our dedicated-probe detection (Table XV, Sec. V-B); no localization, no crossbar-level signal.
Wei & Roth [3]	Multiple-outlier column localization with RIP/group-testing codes, forward reads only.	Pre-empts “sparse recovery for multiple faults” (and Zhu et al. [25] extend it). We study cell-level localization with forward <i>and</i> transposed reads under noisy trained weights, against a forward-only family that includes this code class (our implementation, not the authors’ code; the paper stops at output lines, so its cell-level stage in our baseline is our own extension). A registered run executed their one-MVM decoder on its own terms: it does not reach line F1 0.9 on our moderate stuck faults, a regime mismatch (Sec. V-A).
Sun & Ielmini [34]	Analog closed-loop eigenvector circuit.	Building block. We test it as a self-diagnosis primitive; the answer is negative (L18, which simulates the Sun-type leading-eigenvector loop, not the λ -sweep circuit). Mannocci et al. [35], [36] (full text read) present their circuits as their own contribution and never use them to test an array: no circuit-novelty claim ; the self-test use is new and negative.

did not test faults on a drifted or reprogrammed array, and we did not combine the localizer with the drift fit. (2) The decoder knows the read-noise level σ , which it estimates from 8 repeated forward reads on a fault-free tile (2 reads for each of 4 dense probes); these 8 reads are not counted in the tables or in any ratio, and the closed loop (Sec. VI) does charge them. (3) The orthogonal matching pursuit stops at a cap of $2k + 4$ cells, so the decoder is given an upper bound on the true fault count k ; probe designs are also selected per (k, σ_c) cell on a calibration split, for the baselines as well. (4) Success is a mean cell-level F1 of at least 0.9 over 240 instances. A mean F1 of 0.9 is compatible with missed cells on individual tiles, and we report no per-chip detection, escape or false-alarm rate.

Decoder. We stack all reads into one linear system in the unknown E . Cell (i, j) appears in two places: in the forward

reads of output line i , with pattern $a_j = X_{j,:}$, and in the transposed reads of input line j , with pattern $b_i = Y_{i,:}$. Orthogonal matching pursuit adds the best-matching cell one at a time. It stops when the best match falls below a threshold set for a fixed false-alarm rate, and a final step drops weak coefficients. We freeze every setting on a calibration split before the counted run.

Figure 1 shows the mechanism. One direction names one line per fault. With several faults, the named rows and columns cross at ghost cells. The two directions give independent evidence on the row and on the column of each cell, and the joint decoder keeps only the cells that both directions support.

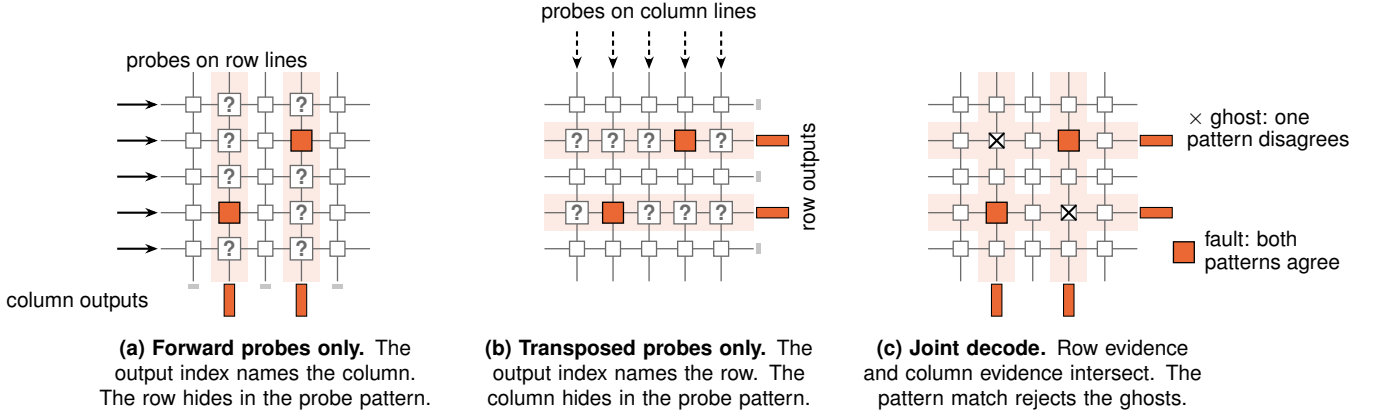


Fig. 1. How two directions resolve ghost cells (an illustration of the mechanism, not a statement about read counts: the joint decoder lowers the ± 1 -probe read floor at 1% noise only, Sec. V-A). The tile is drawn transposed relative to W : input lines are rows and output lines are columns. Two cells are faulty. (a) A forward read flags the two output columns but not their rows. (b) A transposed read flags the two rows but not their columns. (c) The joint decode intersects both sets. Four cells are candidates. Two are ghosts: one direction's pattern does not match.

B. What we can prove

We prove a sufficient condition (App. A) and argue one identifiability fact. We do not prove that two directions need fewer reads than one.

Lemma 1 (Cells in different rows and columns are orthogonal): Let φ_{ij} be the joint-operator atom of cell (i, j) . Then $\langle \varphi_{ij}, \varphi_{i'j'} \rangle = [i = i'] \langle a_j, a_{j'} \rangle + [j = j'] \langle b_i, b_{i'} \rangle$. Two cells that share no row and no column are exactly orthogonal.

Only same-row pairs (through the forward reads) and same-column pairs (through the transposed reads) interact. Let S be the fault set and ψ a normalized atom. Define $a = \max_{\psi \notin S} \sum_{\varphi \in S} |\langle \psi, \varphi \rangle|$ and $a' = \max_{\varphi \in S} \sum_{\psi \in S \setminus \varphi} |\langle \varphi, \psi' \rangle|$. Lemma 1 bounds both by the number of faults that share a row or column with a cell.

Proposition 1 (Sufficient condition): Assume: white Gaussian noise with known s ; an exact linear model with no quantization and no drift during the probes; probes drawn independent of E ; $a' < 1$ and $a + a' < 1$; and $|E_{ij}| \geq \delta$ on S . Let $\nu = p_f/n + p_b/m$, $b = \sqrt{2 \ln(2mn/\eta)}$ and $\kappa = 1 + \max(a, a')/(1 - a')$. If

$$\frac{\delta \sqrt{\nu}}{s} > \frac{2\kappa b}{1 - a - a'}, \quad (1)$$

then orthogonal matching pursuit with stopping threshold κb recovers S exactly with probability at least $1 - \eta$.

The proof adapts the exact-recovery condition of Tropp [39] and the noise analysis of Cai and Wang [40] (App. A). We state its limits.

- It is a sufficient condition and it is not tight. At the registered minimum visible amplitude $\delta = 0.1 \max |w|$, it needs more than the 256 reads of a full readout at every noise level, so it gives no guarantee there.
- It is about exact support recovery. It gives no guarantee on $F1 \geq 0.9$, which allows missed cells.
- It does not describe the shipped decoder. The shipped decoder uses a smaller threshold ($\kappa = 1$) and a final prune. The proposition covers a more conservative variant.

- It does not cover non-white or drifting noise, ADC/DAC quantization, a wrong noise estimate, or an inexact reference.
- It is not a like-for-like comparison with Wei and Roth. Their guarantee is deterministic, for bounded noise and output-level localization in one MVM. Ours is probabilistic, for Gaussian noise and cell-level localization.

As a consistency check, take $k = 1$ and the idealized constants $a = a' = 0$, $\kappa = 1$, $\delta = 0.88 \max |w|$ (the median visible deviation) and $\eta = 0.1$. Condition (1) then needs 3.7, 14.9 and 59.6 reads at 1%, 2% and 4% noise. We measured 4, 16 and 64 reads (Table V). The σ^2 scaling of 1:4:16 is built into condition (1), and the measured counts sit on a power-of-two read grid, so the measured values are consistent with the bound; this is a consistency check, not a validation.

One fault, no noise. With forward reads only, the output index names one coordinate. The other coordinate must come from the vector $E_{ij}a_j \in \mathbb{R}^{p_f}$, which has to tell n candidates apart. With ± 1 probes, a counting argument over sign patterns gives $p_f \geq \log_2 n + 1 = 9$ reads with the best ± 1 design, and about 16 with random probes; real-valued (multilevel-DAC) probes are not covered by this argument. With both directions, one forward read names the output line and one transposed read names the input line, so two reads suffice. The two-read statement is an achievability illustration for $k = 1$ without noise, not a lower bound, and App. A does not prove it.

Why this is not an advantage theorem. Condition (1) is a worst-case incoherence bound, and we do not derive that it is smaller for the joint design than for forward-only decoding. The measured advantage at low noise (1.5 to 3.7 $\times$ fewer reads at 1% noise, Sec. V-A) is therefore empirical: a worst-case incoherence bound cannot show it. An average-case bound on ghost cells would turn it into a theorem. We did not prove one.

C. Concurrent regression from live inputs

Setup (registered before the full run). Each 256×256 tile gets two analog checksum lines programmed with the same noise as the weights: $c_1 = \sum_i W_{0,i}$ and $c_w = \sum_i \frac{i+1}{m} W_{0,i}$,

each scaled to the full conductance range and read back with its gain. Every inference MVM then yields two syndromes $s_1 = \sum_i y_i - g_1 y_{c_1}$ and s_w . A commissioning step right after programming (128 passes over 2048 fault-free inputs) fits the checksum lines' programming mismatch and the noise scale. All methods see the whitened residuals. We store no digital outputs. Live inputs are the cached im2col vectors of each layer. Faults are stuck cells injected after programming. Read noise is 2%. We use 3 seeds $\times$ 8 conv layers $\times$ 4 tiles. Methods: **CK** (our adaptation of the Neural Checksums / HPE detection rule, whose originals compare exactly in the digital domain): energy of the s_1 residuals, threshold at 1% false alarms. **HPE decoder** (the weighted/plain ratio decoder of HPE and X-ABFT [2]): the syndrome ratio over flagged samples names the faulty output row. **Regression** (ours; the two checksum lines and the ratio decoder are prior art, the combination below is what we study): orthogonal matching pursuit of the s_1 residuals on the whitened inputs across T inferences, Bonferroni stop at 1% false alarms per tile. It returns faulty *input* lines.

D. Few-probe field-fit calibration (M4)

Model. Let $\mathbf{W}_0 \in \mathbb{R}^{m \times n}$ be the stored digital reference of a tile's programmed weights and $A(t)$ the aged array. Patterned drift is multiplicative, $A(t) - A(t_c) \approx \mathbf{W}_0 \circ F$ (not low-rank even when F is smooth, which is why a low-rank fit of the error does not work), with $F = \sum_{a,b < 4} c_{ab} \varphi_a(i) \varphi_b(j)$ and $\varphi_a(i) = \cos(\pi a(i + \frac{1}{2})/m)$ (*cosine basis, order 4*: 16 coefficients per tile).

Probes. At calibration time t_c (right after programming) read $Y_0 = A(t_c)X$ with $X \in \{\pm 1\}^{n \times 8}$ and $Z_0 = A(t_c)^T Z$ with $Z \in \{\pm 1\}^{m \times 7}$, and store Y_0, Z_0 digitally. At recalibration time t , re-read the same probes, form $\Delta Y = A(t)X - Y_0$ and $\Delta Z = A(t)^T Z - Z_0$. Both are linear in c : $(\mathbf{W}_0 \circ \varphi_{ab})X$ and $(\mathbf{W}_0 \circ \varphi_{ab})^T Z$. A ridge least-squares solve (ridge 10^{-8} of the mean Gram diagonal) uses $8m + 7n = 3840$ scalars to fit 16 unknowns. The corrected effective operator is $A(t) - \mathbf{W}_0 \circ \hat{F}$ (*ideal form*). **Ideal versus deployable.** The ideal form needs the stored digital $\mathbf{W}_0$ at every inference: it is a digital MVM per tile, so it serves only as an upper reference for what the field *estimate* is worth. The *deployable* form multiplies the analog array by $G = 1/(1 + \hat{F})$ projected on the 4×4 cosine span and truncated to rank r by SVD, $\sum_{k \leq r} \text{diag}(u_k) A(t) \text{diag}(v_k)$: each term is one analog pass with digital input/output scaling, so rank 1 is free and the full form (rank ≤ 4) costs up to 3 extra passes per inference (Sec. V-D). One further read of the stored layer-mean input vector gives a digital bias correction (the same bias step is given to the B8 baselines). **Cost.** 8 forward + 7 backward probes + 1 bias read = 16 MVMs per tile per recalibration (the *strict* budget); the variant run in Exp. 2 and first run in a_bnfold uses 8 + 8 probes plus the bias read = 17. Zero calibration images are needed.

Incumbent B8. The strongest cheap digital baseline: per tile, the best of {global GDC, per-row gain, per-column gain, bi-scale $\text{diag}(a)\mathbf{W}_0\text{diag}(b)$ } (each fitted from 8 probe reads) plus the same mean-bias correction, *selected per tile by its output error against the programmed weights on held-out activations*. This oracle-style selection is optimistic for the incumbent (Sec. VI-D).

TABLE III
PROVENANCE OF SIMULATOR CONSTANTS (FULL REGISTRY:
SRC/NEUROFAULT/SPECTRAL/PARAMS.PY, DUMPED TO PARAMS.JSON IN
EVERY RUN DIR).

Constant	Status	Value / source
PCM $\mu(g), \sigma(g)$ of ν	sourced	$\mu = \text{clip}(-0.0155 \ln g_r + 0.0244, 0.049, 0.1)$; $\sigma = \text{clip}(-0.0125 \ln g_r - 0.0059, 0.008, 0.045)$; AIHWKit
t_0	sourced	20 s (AIHWKit)
1/f read noise	sourced	$q_s = \min(0.0088/g_r^{0.65}, 0.2) \sqrt{\ln \frac{t+t_r}{2t_r}}$, $t_r = 250$ ns
Programming noise	sourced	AIHWKit coefficients [0.26348, 1.9650, -1.1731]
Pattern strength ρ	assumed	sweep {0, .25, .5, 1, 2, 4}; no measured spatial correlation of ν found
Field shapes, hotspot width, rank r	assumed	width $0.2 \max(m, n)$, $r = 2$
Temperature (D4): $E_a, T, \beta, \Delta T$	assumed	0.5 eV, 330 K, 10^{-3} /K, 20 K
Energy/latency constants	assumed	not used for any headline claim

IV. EXPERIMENTAL SETUP

A. Simulator

Tiles are 256×256 blocks of the flattened conv weights of a trained ResNet-9 (109 tiles per network; blocks with a side < 16 dropped). Weights are encoded as differential pairs: $|w|$ on one device (G^+ for positive, G^- for negative weights), the other at $g_{\min} = 0.01 g_{\max}$, with tile-abs-max scaling and $g_{\max} = 25 \mu\text{S}$. Programming noise, drift and read noise use constants *read from the installed AIHWKit 1.1.0 source* [41], [42] (“sourced” in Table III); the spatial structure of drift and the temperature model have no measured source and are “assumed” and swept. Device (i, j) evolves as $G(t) = G_0((t + t_0)/t_0)^{-\nu_{ij}}$ with $t_0 = 20$ s; D2 draws $\nu \sim \mathcal{N}(\mu(g), \sigma(g))$ per device from the AIHWKit PCM model; D3 adds a position-dependent field, $\nu_{ij} = |\mu(g) + \sigma(g)(z_{ij} + \rho f_{ij})|$ with f unit-variance (row gradient, rank- r cosine, or Gaussian hotspot), so ρ is the structured-to-iid ratio and $\rho = 0$ reproduces D2 exactly; D4 adds an Arrhenius time-acceleration and a read-gain temperature field (all assumed). Probe reads are white per-cell noise of $\sigma_c = 2\%$ of the tile's $\max |w|$, fresh on every MVM. This is the AIHWKit 1/f model's value for trained ResNet-9 tiles (2.0% rms, 2.6% p90, 4.5% at $g_{\max}$ at $t = 20$ s; 2.4% rms at 1 d), treated as uncorrelated across reads. Network accuracy is evaluated in *weight space*: the decoded, drifted, corrected effective weights replace the conv weights in PyTorch (BN folded, digital bias). No ADC/DAC quantization is applied to the probe reads in the headline runs.

B. Counted-read model

Every method touches a tile only through a reader that counts reads: a forward MVM of p vectors adds p to `mvm_fwd` (and `mp` ADC conversions, `np` DAC drives), a transposed MVM adds p to `mvm_bwd`, a full cell readout adds mn cell

reads (equivalently $n = 256$ one-hot MVMs for an $m \times n$ tile; the tables count MVM reads, so a full readout is 256). In this paper a backward MVM costs one MVM (backward-read cost is hardware-unverified). AdaBS cost is images $\times$ per-layer patch positions, in MVM-equivalents. A full-network recalibration of M4 costs $109 \times 16 = 1744$ MVMs (strict) or $109 \times 17 = 1853$ (as run) versus 2,622,312 for AdaBS-256 and 26,624,872 for AdaBS-2600 (Exp. 2 results). *The inference-time cost of applying the correction is not counted; see Sec. VI-D.* **What the localization read counts include.** Counted reads in Table V cover the `localize()` call only. The 8 forward reads that estimate the read-noise level are excluded for every method and from every ratio in this paper, and the decoder is given the cap $2k + 4$ (Sec. III-A); whether the baselines would need the same estimate is not specified, and we charge them nothing. Charging the 8 reads to the joint decoder only raises its 6, 8 and 12 reads at $k = 4, 16, 64$ and 1% noise to 14, 16 and 20, and lowers the factor against the hybrid (27, 51 and 51 reads) from 4.3–6.4 $\times$ to 1.9–3.2 $\times$. We did not re-run the registered clauses with that charge. Counts come from a coarse read grid (4, 6, 8, 12, 16, 24, 32, 48, 64), so ratios are quantized: 6 against 8 reads is a 33% step.

C. Pre-specification protocol

We call a claim *registered* when its criteria file was committed to the author’s timestamped development repository before the counted run; this is pre-specification, not independent registration (no third-party timestamp). Each claim is a YAML file of clauses. A clause has a metric, a filter, an aggregation, a comparator and a threshold, and the file has a frozen date. A script (`verdict.py`) computes each status: PASS, NOT_MET, KILL, CLEAR, INCONCLUSIVE or REPORTED. Each `verdict.json` stores the SHA-256 of the criteria file it used. A threshold may change only in a new commit with a logged reason. The log has two entries: a YAML parsing bug (1.0e4 read as a string) and a check we had transcribed against the wrong circuit. Neither entry changed a threshold on the data. A *registered follow-up* is allowed only if the failed claim stays on record. `a_bnfold` (after Exp. 2) and `b_c1b` (after the first kill) were registered *after* the result that suggested them. `a_harden` was registered before its run. Follow-ups use fresh random streams (seed offsets +1000, +2000, +4000, +5000) and are listed next to the failure that motivated them (Table IV). Fresh randomness is not a fresh checkpoint. `a_bnfold` reuses the CIFAR-10 checkpoints (seeds 42/101/102); only the CIFAR-100 checkpoints are new. Smoke runs never count.

D. Methodology and rigor

We tried to break our own results. Five practices did the work.

1. We pre-specified first and let a script judge. The deployability runs, the closed loop, its extension, and the X-ABFT and Wei–Roth comparisons had their criteria committed before their counted runs. Commits: 6ade20d (stronger baselines), 37af453 (deployable cost), a9a1928 (refresh allocation, deployable baselines), d6bc34d (concurrent regression),

35a83d3 (comparison with prior-art localizers) and 018276a (X-ABFT and Wei–Roth). For the last two the counted runs began 11 and 12 minutes after the commit. For every registered follow-up and every deployability file, the commit that adds the criteria file precedes the start time of the run it judges. We checked this with `git log`. Commit times are the only independent timestamps.

2. We disclose the late first wave. The criteria for the validity gates, the first drift gate, the AdaBS comparison and the first transposed-read claims were committed in 8c2ab18, after nine full runs. The commit message says so. Their thresholds were fixed in plans committed earlier (8cb7364). We did not diff the YAML against the plan text. So the repository cannot prove that these thresholds predate the data.

3. We re-ran the competitors on their terms. We tuned X-ABFT per row and ran Wei and Roth’s decoder under bounded noise. We froze these criteria after a smoke run of 2 instances on 1 checkpoint. That run had shown that the one-MVM Wei–Roth decoder flags no line at 1 to 4% noise, and that X-ABFT reaches F1 0.0 to 0.66 at 4% even with 16 repeats. So two outcomes (Wei–Roth own-terms NOT_MET and X-ABFT noise degradation PASS) were foreseeable. We set the thresholds by what would change a paper claim, and we print the outcomes as they came.

4. We report what failed. The 21 criteria files end as follows: 4 PASS, 7 KILL, 10 INCONCLUSIVE (overall verdict of the final counted run of each file; `a_bnfold` is counted under its strict 16-MVM reading, and under the as-run 17-MVM reading it is INCONCLUSIVE, giving 3 PASS, 7 KILL and 11 INCONCLUSIVE). The kills include two of our own registered claims about transposed reads, the σ_1 watchdog, the analog eigen loop, the free in-array form of M4, and the closed-loop extension at 4% noise. Table IV lists 19 of the 21 files with their mechanical outcomes, and Table XV lists 21 limits.

5. We audited the raw data. We recomputed all 52 verdict files from the stored metrics. 46 reproduce exactly. The other six are smoke runs (four) or killed variants without a stored metrics file (two), and no counted verdict differs. For 40 of 44 run directories, the code commit does not reproduce the run tree, because the scripts were first committed in 2ac6e07, after those runs. The stored configuration hashes and checkpoint SHA-256s identify the inputs, but the code state of those runs is not archived.

V. RESULTS

A. Fault localization: when two directions help

We start with the registered comparison against prior-art localizers, then the two fair re-runs (X-ABFT, Wei–Roth), and then the regime map where the second direction did not help (the two killed claims).

1) Against prior-art localizers (registered, PASS): Table V compares the HPE-style binary search, our Roth-style hybrid built from the Wei–Roth disjunct code (not the authors’ code, and not the own-terms decoder below), and our joint compressed-sensing decoder with forward-only and with forward plus transposed probes. Every family’s configuration is

TABLE IV

LEDGER OF EVERY REGISTERED CLAIM THAT THIS PAPER TOUCHES, WITH ITS MECHANICAL OUTCOME. THE CLAIM IDS (F0.x, G0-x, BF, H-NUMBERS AND SO ON) AND CRITERIA-FILE NAMES ARE INTERNAL RUN LABELS FROM THE AUTHOR'S REPOSITORY; THEY APPEAR ONLY IN THIS TABLE. FROZEN DATES ARE 2026 (MONTH-DAY). FOLLOW-UPS ARE LISTED DIRECTLY BELOW THE CLAIM THAT MOTIVATED THEM.

Claim (criteria file)	Frozen	Outcome	Run dir
Validity gates F0.0a-e	10-08	all PASS: AIHWKit D2 distribution match (KS $p = 0.023 > 0.01$, mean-ratio error 0.003), legacy-drift parity (max diff 0), invariants.	f0_0/ 20261008T204618_e55b3002_s0
H1 bridge F0.8a-c (f0_validity)	10-08	F0.8c PASS (no-drift accuracy within the binomial CI, 0.53 half-widths); F0.8a NOT_MET (twin vs CrossSim max 5.66 pt, bar 0.5; mostly B8 in BN-folded/hotspot cells); F0.8b NOT_MET (twin vs AIHWKit native 1.34 pt, bar 1.0).	f0_8/ 20261011T151445_d931ec73_s0
Gate 0' (a_gate0)	10-08	G0-a PASS (B8 excess error 4.0 pp), G0-d PASS (oracle-16 vs B8 3.0 pt); G0-b NOT_MET (specificity 0.095 < 0.15), G0-c NOT_MET, G0-e NOT_MET (M1 partial AUC 0.60), G0-f NOT_MET (Jacobian rank deficit 1). Frozen rules: GK-drop-C2 KILL, GK-relabel REDIRECT.	a_gate0/ 20261009T020655_ae67b976_s0
F0.1 stuck cells vs σ_1	10-08	H1.1 PASS (AUC < 0.6 in 99.4% of cells); H1.2 NOT_MET (exponents -1.40, +0.81); H1.3-line NOT_MET (AUC 0.69).	f0_1/ 20261009T023921_a8f8370e_s0
F0.3 headroom, watchdog	10-08	H3.2 PASS, H3.3 PASS, H3.5 CLEAR; H3.4 KILL (σ_1 watchdog); H3.1 NOT_MET (strict per-seed ordering; seed 101 within evaluation noise).	f0_3/ 20261009T181003_aca2558e_s0
Exp. 2 (a_success)	10-08	S1 PASS; S2 not met (stored verdict INCONCLUSIVE, see text); K1/K2/K4 CLEAR; overall INCONCLUSIVE.	a_exp2/ 20261009T052819_52891c48_s0
↔ a_bifold (BF1-3)	10-09	Strict 16 MVM: PASS on all five. As run (17 MVM): BF3 NOT_MET ⇒ INCONCLUSIVE.	a_bifold_v2/ 20261010T022159_fca882d8_s0
B claim C1 (b_claims)	10-08	C1 NOT_MET (0/5 k qualify) ⇒ C1-kill KILL .	b_c1_preg/ 20261009T035140_0fdb78ed_s0
↔ b_c1b	10-09	C1b-128 PASS (5/5 k); C1b-256 NOT_MET (0/5) ⇒ C1b-kill KILL ; tile 512 reported 1/5.	b_c1b/ 20261009T115059_609a698a_s0
C' analog loop (c_claims)	10-08	S1 NOT_MET, K1a KILL, K1b KILL (median SNR ratio 0.24), D1 KILL.	c_x1/ 20261010T023044_1e4807e2_s0
H2 stronger baselines, ideal	10-10	NI vs HB1 ₈ , HB1 ₃₂ , HB2, HB3 all PASS; kill not fired. Upper reference only (ideal form needs digital W_0).	a_hardening/ 20261010T220159_8434d92b_s0
M4 (a_hardening)			
H5 deployable cost (a_deploy)	10-10	H5-deploy NOT_MET; free rank-1 variant KILL (paired-CI lower bound < 0 at every CIFAR-100 time).	a_deploy/ 20261010T223102_71522b22_s0
↔ H2b deployable M4 (a_hardening_deploy)	10-10	NI vs HB2, HB3 PASS; NI vs HB1 ₈ , HB1 ₃₂ NOT_MET; kill and pairing gate CLEAR ⇒ INCONCLUSIVE.	a_hardening_deploy/ 20261011T041351_794c47e8_s0
H8 CS vs prior-art (b_vs_prior)	10-11	CS-2x, CS-4x, CS-forward-2x PASS; kill CLEAR ⇒ PASS .	b_vs_prior/ 20261011T210412_0ec5ea78_s0
X closed loop, estimated map (x_loop)	10-12	dense, edge, sparse, gap PASS; kill CLEAR ⇒ PASS .	x_loop_box/results
X ext. (5 seeds, σ 4%, ViT-tiny) (x_loop_ext)	10-12	σ 4, ViT PASS; R9 5-seed NOT MET (reads 0.2525 vs 0.25); kill FIRES (recall 0.85–0.89 vs 0.90 at σ 4% sparse) ⇒ KILL ; accuracy gap ≥ -0.07 pt every group.	x_loop_ext/results
H6 refresh allocation (a_refresh)	10-10	saves-writes PASS; sensing-adds-value NOT_MET; kill CLEAR (0.991 / 0.993 < 1) ⇒ INCONCLUSIVE.	a_refresh/ 20261011T185337_449adbef_s0
H7 concurrent (b_concurrent)	10-11	C5-detect, C5-locate, beats-HPE NOT_MET; both kills CLEAR ⇒ INCONCLUSIVE.	b_concurrent/ 20261011T152741_d4b34e56_s0
E4 X-ABFT baseline (e4_xabft)	10-14	E4-CS-2x PASS (4 of 4 k at 1% and at 2%); E4-ratio-decode-degrades PASS (X-ABFT F1 < 0.9 at 4%, max 0.594); both kills CLEAR ⇒ PASS (report scope INCONCLUSIVE by design).	e4_xabft/ 20261014T124345_4b2720f4_s0
E5 Wei-Roth on its own terms (e5_weiroth_fair)	10-14	E5-own-terms NOT_MET (line F1 0); E5-CS-2x-bounded PASS; E5-CS-2x-gauss PASS; both kills CLEAR ⇒ INCONCLUSIVE as designed.	e5_weiroth_fair/ 20261014T124416_9b59bf83_s0

frozen on a calibration split (fault seeds 3000–3019, tile origin 1; selection rule as in the follow-up claim below) and evaluated on 240 fresh instances per cell (3 checkpoints $\times$ 4 layers $\times$ fault seeds 60–79). Registered bars: $\geq 2\times$ and $\geq 4\times$ fewer reads than Wei–Roth (paired-CI upper bound $\leq 0.6 / \leq 0.33$, or Wei–Roth misses F1 0.9 while CS reaches it) on ≥ 3 of 4 k at each of 1% and 2% noise; both PASS, and so does the forward-only $2\times$ bar; kill CLEAR (median ratio 0.19). Read counts exclude the 8 noise-calibration reads and lie on a coarse grid (Sec. IV). The forward-only decoder passes its own registered bar too, so the comparison with the hybrid mostly measures the decoder and not the second direction (see the attribution below). With both directions the decoder needs 4.3–6.4 $\times$ fewer reads at 1%

and 5.1–5.4 $\times$ at 2% for $k \geq 4$; a single fault at 2% is the exception (16 vs 32 reads, 2.0 $\times$, CI upper 0.52). At 4% and $k \geq 4$ our reimplementations of both prior-art schemes fall short of F1 0.9 (HPE-style 0.02–0.40, Wei–Roth-style 0.28–0.39) while the decoder reaches F1 0.93–0.95 with 32 reads (report-level; this is not a registered clause). We do not report the F1 of a full readout at 4% noise; it was not measured.

How much is the decoder and how much is the second direction. Table V separates the two. The hybrid needs 26/27/51/51 reads at 1% for $k = 1/4/16/64$; forward-only compressed sensing needs 16/12/16/16, and the joint decoder 4/6/8/12. So the forward-only decoder alone already needs 1.6 to 4.1 $\times$ fewer reads than the hybrid across 1–2% noise, and

TABLE V

REGISTERED COMPARISONS (PRIOR-ART LOCALIZERS, X-ABFT, WEI-ROTH):
 MEAN COUNTED READS OF THE FROZEN CONFIGURATION ON FRESH FAULTS (TILE
 256, 240 INSTANCES PER CELL, 8 NOISE-CALIBRATION READS NOT INCLUDED); “–”
 = HELD-OUT CELL F1 < 0.9, FULL CELL READOUT = 256 ONE-HOT MVM READS
 ($m n = 65,536$ CELL READS); ITS F1 IS NOT REPORTED.

Method	1%	2%	4%
	reads at $k=1/4/16/64$		
HPE binary search	10/–/–/–	39/–/–/–	157/–/–/–
Roth-style hybrid (our reimpl.)	26/27/51/51	32/41/65/–*	–/–/–/–
Joint CS, forward only	16/12/16/16	16/12/16/24	48/32/32/32
Joint CS, both directions	4/6/8/12	16/8/12/12	64/32/32/32
Full cell readout (reference)	256/256/256/256	256/256/256/256	256/256/256/256
X-ABFT, tuned	–/–4096/2048	–/–/–/–	–/–/–/–
Wei–Roth line, one MVM	–/–/–/–	–/–/–/–	–/–/–/–
Wei–Roth cell, best	–/–/–/–	–/–/–/–	–/–/–/–

*65 reads at F1 0.895. X-ABFT reaches F1 0.9 at 1% noise only for $k = 16$ and 64 (F1 0.915, 0.936). Both Wei–Roth rows hold under bounded and under Gaussian noise; the line decoder’s line F1 is 0.

the registered forward-only bar passes. The second direction adds the rest at 1% noise; at 2% it adds nothing at $k = 1$ (16 vs 16) and 1.3 to 2.0 $\times$ in the other cells, and at 4% it adds nothing for $k \geq 4$ (32 vs 32) and costs reads at $k = 1$ (64 vs 48).

2) *X-ABFT, tuned per row (registered, PASS)*: X-ABFT stores two checksum columns per block and decodes (row, column, deviation) from their ratio [2]. We built a runnable version. We tuned its design and its repeat-and-average factor separately for every (k, σ_c) row, on the same 240 instances per row as the comparison above. The registered 2 $\times$ clause passes. On 4 of 4 k at both 1% and 2% noise, the joint decoder needs at least 2 $\times$ fewer reads than X-ABFT, or X-ABFT misses F1 0.9 while the decoder reaches it. We print the F1 values next to the read counts, because the pass rests mostly on the second branch. Over the whole grid of 12 (fault count, noise) cells X-ABFT reaches F1 0.9 in only 2, both at 1% noise: $k = 16$ and $k = 64$ (F1 0.915 and 0.936), with 4096 and 2048 reads against 8 and 12 for the decoder. For $k = 1$ and $k = 4$ its F1 is 0.667 and 0.822. At 2% noise X-ABFT never reaches 0.9 (F1 0.484 to 0.858). Every tuned X-ABFT row at 1% and 2% noise uses 1024 or more reads, so it needs more reads than a full 256-read readout of the tile, including the two rows where it reaches F1 0.9 (4096 and 2048 reads). At these fault counts the practical comparator is therefore a full readout and not X-ABFT. We ran no positive control: X-ABFT inside its own design regime (at most two faults per block, negligible noise) was checked only by unit tests with noise-free reads, not in the registered harness, so this run cannot separate a stress test outside the design regime from a weak implementation. In particular, a blocked configuration (blocks of 4 $\times$ 16 cells, where a density of 0.1% puts about 0.06 faults in a block, inside its regime) was not run, and how reads should be counted for blocks read in parallel is undefined here. The strict rule, a measured 2 $\times$ on every row, holds on 2 of 4 k at 1% and on 0 of 4 at 2%. The second registered clause also passes. At 4% noise X-ABFT stays below F1 0.9 for every k (best 0.594), while the decoder reaches 0.93 to 0.95 for $k \geq 4$. This turns “the ratio decoder is noise-fragile” from an expectation into a measurement. Neither kill fired: X-ABFT never reached F1 0.9 with fewer reads than

the decoder.

Regime caveat. X-ABFT is designed for at most two faults per block at 1 to 10% fault density, with no per-read noise. Our regime is $k \leq 64$ faults in a 256 $\times$ 256 tile (about 0.1% density) with Gaussian read noise. This run is a stress test outside its design regime. It is not a like-for-like benchmark, and we do not write “X-ABFT fails”. Three choices favor X-ABFT: its checksum lines come from the noise-free digital reference, its detection is noise-aware, and we gave it a repeat-and-average knob that its paper lacks.

3) *Wei and Roth on their own terms (registered, INCONCLUSIVE as designed)*: Our comparison baseline (“Roth-style hybrid” in Table V) is a hybrid (see the list below), so we also ran the Wei–Roth decoder as its paper defines it [3]: one MVM with r redundant columns and a threshold on the syndrome. We ran it under bounded noise (rms-matched to the Gaussian noise) and under Gaussian noise, at 1, 2 and 4% (240 instances per row). We froze the criteria in 018276a before the counted run.

Own terms: NOT_MET. The one-MVM line decoder never reaches line F1 0.9. It flags no line: line F1 is 0 on every row (at most 1.3×10^{-4}), with 25, 49, 121 and 121 redundant columns for $k = 1, 4, 16, 64$. Their guarantee needs outliers larger than $2\rho\delta\|x\|$. With our noise model the threshold grows with the probe norm, and the parity checks sum about 52 noisy lines, so it lies above our fault amplitudes (median visible deviation about $0.88 \max|w|$, visibility floor $0.1 \max|w|$). This explanation is our judgment from a smoke note, not a registered result: no clause evaluated the guarantee’s constants. We call it a regime mismatch and not a failure of their method; their decoder behaves as specified in our unit tests (an outlier above $2\rho\delta$ is located, one below it is missed).

Comparison clauses: PASS. Forward-only decoding needs at least 2 $\times$ fewer reads than the best Wei–Roth cell-level arm (PASS under our own pass rule), or the arm misses F1 0.9 while the decoder reaches it. This holds on 4 of 4 k under bounded noise and on 3 of 4 k under Gaussian noise, at both 1% and 2%. Under Gaussian noise at $k = 4$ and 2%, our decoder reaches F1 0.896, so that row fails. All passes come from the second branch. No Wei–Roth cell arm reaches F1 0.9 on any 1–2% row. The best arm reaches F1 0.38, 0.26, 0.11 and 0.17 at 1% for $k = 1, 4, 16, 64$, and at most 0.016 at 2%. So we did not measure a read ratio. We measured that these arms do not reach F1 0.9 on our faults. Neither kill fired. The overall verdict is INCONCLUSIVE by design: the registered rule needs all clauses to pass, and a pass of the own-terms clause would have meant that their method works on our faults.

Fairness of the Wei–Roth comparison. Four points bound it, and the own-terms run removes none of (i) and (ii). (i) The Wei–Roth paper stops at output lines. The cell-level stage of our baseline is our own forward-only extension. So the baseline is a hybrid of their output-level stage and our input-side probe stage. The headline is cell-level localization against a hybrid group-testing baseline, not “ $\times$ fewer reads than Wei and Roth”. (ii) Their cost unit is r redundant columns inside one MVM. The reads in Table V are the cost of our reimplementation. The cost axes differ. (iii) Their guarantees are for bounded noise.

TABLE VI

FOLLOW-UP CLAIM: READS RATIO (BIDIRECTIONAL / BEST FORWARD-ONLY) AT $F1 \geq 0.9$, FRESH FAULT SEEDS. ENTRIES MARKED † FAIL THE REGISTERED ROW TEST ($F1 \geq 0.9$, RATIO ≤ 0.7 , CI UPPER ≤ 0.8). BAR: 0.7.

Tile	σ_c	$k=1$	2	4	8	16
128	1%	0.27	0.33	0.49	0.50	0.67
128	2%	0.50	0.50	0.50	0.50	0.67
256	1%	0.42	0.50	0.38	0.38	0.50
256	2%	1.00 †	0.75 †	0.75 †	0.75 †	0.75 †
512	1%	0.38	0.38	0.38	0.50	0.50
512	2%	1.00 †	0.67 †	1.00 †	1.00 †	0.67

Under Gaussian read noise at 1 to 4% our reimplementation needs thresholds and repeats, a regime the authors did not design for. (iv) Our registered pass rule credits a Wei–Roth failure to reach F1 0.9, which counts failures outside their noise model as wins. The result is this. Under bounded and under Gaussian noise, no Wei–Roth arm we built reaches F1 0.9 in our fault regime, and the one-MVM line decoder detects nothing because the outliers sit below its guarantee.

4) *When the second direction does not help (two registered claims, both killed): Setup.* Localize $k \in \{1, 2, 4, 8, 16\}$ stuck cells in a tile with *counted* reads at cell-level $F1 \geq 0.9$; compare the best bidirectional design (forward Ax plus transposed $A^T y$ reads) against the best forward-only design (seven forward-only designs, including HPE-style binary search and a Wei–Roth-style code; our reimplementations), with identical tuning effort. Configs are frozen on a calibration split and evaluated on held-out faults (3 checkpoints $\times$ all conv layers, 20 fault seeds in the first claim; 20 fresh seeds in the follow-up). No ADC or DAC quantization (`adc_bits=None`); read noise $\sigma_c \in \{1\%, 2\%\}$ of $\max |w|$ (1% is optimistic-PCM/pessimistic-ReRAM, 2% the AIHWKit PCM central value; Sec. IV). **First claim (registered): KILL.** Reads ratio ≤ 0.7 with CI upper bound ≤ 0.8 and $F1 \geq 0.9$ for ≥ 3 of 5 k . The reads ratio held out of sample on 29 of 30 rows (0.28–0.75), but the frozen bidirectional configuration’s held-out F1 fell just below 0.9 (0.857–0.899) on at least one row for every k : 0/5 qualify. Supplementary per-tile-size evaluation of the same criterion: tile 128 and 256 PASS, tile 512 NOT_MET (`verdict_tile*.json` of the first claim). **Follow-up claim (registered after the kill): KILL** (Table VI, Fig. 2). Margin-calibrated selection (calibration $F1 \geq 0.93$), tiles 128 and 256 only (scope chosen *after* tile 512 failed; disclosed), fresh fault seeds. Tile 128 PASS 5/5; tile 256 0/5 (reads, not accuracy: at $\sigma_c = 2\%$ the bidirectional design needs 12 reads vs 16 for forward-only, ratio 0.75 $>$ 0.7). The margin fixed generalization (held-out $F1 \geq 0.9$ on 29/30 rows).

What survives as a finding. At $\sigma_c = 1\%$ bidirectional reads need 1.5–3.7 $\times$ fewer reads on every tile size (ratios 0.27–0.67, typically 2–3 $\times$); at the PCM-typical 2% the saving is 1.5–2 $\times$ on tile 128 and marginal or absent on 256 and 512 (ratios 0.67–1.0). *Selection optimism is visible in our own history:* an exploratory run (best-of selection on the evaluation instances, 2 layers $\times$ 5 seeds) reported a 2.0 $\times$ saving (12 vs 6 reads) at $k=1$, tile 256, 2% noise; the same cell on fresh seeds shows 16 vs 16 reads (ratio 1.0). Two further regime facts: with random ± 1 probes and a generic joint decoder

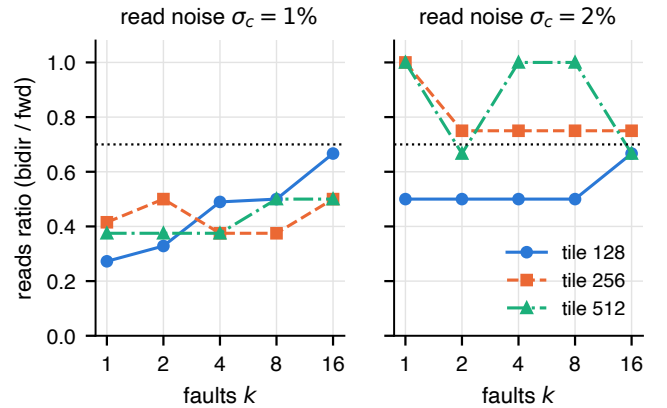


Fig. 2. Reads ratio versus fault count at 1% (left) and 2% (right) read noise. Dashed: the registered 0.7 bar. The advantage is real at 1% on every tile size and disappears at 2% on tiles ≥ 256 .

the two-direction advantage vanishes by about 16 reads (L9; exploratory numbers, not a registered result), and probing beats a full 256-read readout of a 256 $\times$ 256 tile only up to a fault density of 0.3% (`probing_win_max_density=0.003`; at $\geq 1\%$ full readout wins), far below the 10% per-wave density of repair campaigns, where localization by probing is not worth its reads. The claim that an estimated map drives repair is out of scope here.

5) *Ways to localize stuck cells, and what each costs:* Several of these methods localize stuck cells, and they differ in what hardware and decoding they need, not only in reads. Table VII collects the numbers above for one representative fault count, $k = 16$ in a 256 $\times$ 256 tile, and adds the requirements and the regime where each method is the better choice.

Which method to choose depends on the noise, the fault density and the array. Forward-only sparse decoding is the robust default: it needs no transposed read path, and it reached or nearly reached F1 0.9 on every row of our grid with 12 to 48 reads (F1 0.900 at 2% and $k = 4$). Transposed reads pay only at low noise: at 1% the joint decoder needs 8 reads against 16 at $k = 16$, and at 2% 12 against 16, and at 4% with $k \geq 4$ it needs no fewer. Above about 0.3% fault density a full readout of the tile is cheaper than probing, so it wins in the dense regime (an exploratory finding at tile 256). The prior-art arms are not shown to be poor methods: each is outside its design regime here (see the X-ABFT and Wei–Roth subsections above), and the table is a statement about our fault model and read-noise grid.

B. Concurrent localization from live inputs

The setup is in Sec. III-C. **Results** (Table VIII). The planned budgets (detect within 64 MVMs, localize within 256) are out of reach for every concurrent method: summing 256 cells’ read noise on a checksum line leaves a per-inference SNR of about 0.05 (L21). The registered verdict is INCONCLUSIVE: the C5-detect, C5-locate and beats-HPE clauses were all NOT_MET. Read from the report-level curves (Table VIII),

TABLE VII

WAYS TO LOCALIZE STUCK CELLS, AND WHAT EACH COSTS. READS ARE MEANS AT $k = 16$ STUCK CELLS (TILE 256) FROM TABLE V; “–” MEANS HELD-OUT CELL F1 < 0.9 (HPE-STYLE F1 0.11–0.12; X-ABFT AT 2% REACHES F1 0.832 WITH 4096 READS). EVERY ARM HERE USES THE STORED DIGITAL WEIGHTS AS REFERENCE. DECODER COMPUTE IS A QUALITATIVE JUDGMENT, NOT A MEASUREMENT. “BEST CHOICE” STATES A REGIME, NOT A RANKING, AND IS LIMITED TO THE TILE SIZE, NOISE GRID AND STUCK-AT FAULT MODEL WE TESTED.

Method	Reads, 1% noise	Reads, 2% noise	Transposed reads?	Stored digital weights?	Decoder compute	Regime where it is the best choice
Full readout	256	256	No	Yes	None (subtract)	Fault density above about 0.3% (exploratory, tile 256), or when a robust answer is needed whatever the noise
HPE-style binary search	–	–	No	Yes	Very low	Not the best in our grid; it reaches F1 0.9 only at $k = 1$ (10 reads at 1%, 39 at 2%), where a forward-only decoder is not available
Roth-style hybrid (our reimpl.)	51	65	No	Yes	Low (threshold and group test)	Not the best in our grid; its published guarantee is for bounded noise and outliers above its threshold, and our fault amplitudes sit below it (E5)
X-ABFT, tuned	4096	–	No	Yes	Low (ratio decode)	Its design regime, at most two faults per block at 1–10% density with negligible read noise, which we did not test in the registered harness
Forward-only sparse decode	16	16	No	Yes	Moderate (sparse recovery)	Default when the array cannot be read transposed, or at 2–4% noise; it reaches or nearly reaches F1 0.9 on every row of the grid (0.900 at 2%, $k = 4$)
Joint (forward + transposed) sparse decode	8	12	Yes	Yes	Moderate to high (larger operator)	Low noise (1%) on a transposable array, where it needs half the reads of forward-only; at 4% and $k \geq 4$ it gives no gain

TABLE VIII

CONCURRENT METHODS VS NUMBER OF LIVE INFERENCES T (MIN OVER SEEDS FOR DETECTION; MEAN F1 OVER INSTANCES).

T	64	256	1024	4096	16384
Detect 1 big fault, CK	0.04	0.06	0.09	0.11	0.13
Detect 1 big fault, regression	0.05	0.15	0.60	0.92	0.99
Line F1, HPE decoder ($k=4$)	0.00	0.01	0.01	0.01	0.01
Line F1, regression ($k=4$)	0.03	0.11	0.35	0.70	0.87
Line F1, regression ($k=1 / 16$)	.03/.03	.10/.09	.39/.27	.71/.60	.86/.83

with thousands of inferences the regression does better than our adaptations of the per-inference decoders: detection 0.99 vs 0.13 for CK at 16k inferences, and line F1 0.83–0.87 vs about 0.01 for the HPE decoder, whose row estimate needs ratio precision $1/(2m)$. False alarms on fault-free tiles were 0.4–1.2% (regression). Almost no input line is dead (median dead fraction 0), so faults are observable from live data. 16k inferences of one tile are 64 images at the median layer (16 to 1024 by layer). Dedicated tests that pause inference are cheaper for detection: 8 stored natural test inputs or 8 random ± 1 probes detect 16 stuck cells with power 0.98–1.00 (1 cell: 0.38–0.46).

C. Drift compensation

1) *The pre-run gate: what failed:* The registered gate asked four questions before any network-level work. We report each answer, including the failures. (i) *Headroom* exists: the best cheap incumbent leaves 4.0 pp more output error than at $\rho = 0$ (PASS) and an oracle rank-16 correction would add 3.0 pt accuracy (PASS). (ii) **Structure-specificity failed.** The original claim was that gains come from tracking drift *structure*; the

registered test required the capture gap between structured and iid drift to be ≥ 0.15 , and measured 0.095 (NOT_MET). Figure 3 shows why: M4 already beats B8 by 2.4 pp output error at $\rho = 0.25$ (sigma 2%) and the gap grows to 12–14 pp at $\rho = 4$ for hotspot and rank- r fields, i.e. it is a better calibrator at every structure level, not a structure tracker. Under the frozen rule the work continues only as *calibration-cost reduction*. (iii) The *watchdog* idea (dominant-mode statistics as drift classifier/trigger) failed: M1 partial AUC given the free GDC factor is 0.60 (< 0.7 , NOT_MET) and temperature, drift and aging are not separable (Jacobian rank deficit 1, NOT_MET); the frozen drop rule fired (KILL). (iv) Realizable-over-oracle capture at 2% probe noise was negative (NOT_MET, -2.2 vs bar 0.6); noisy-probe M4 still beat B8 on tile errors, and we did not investigate the metric further; the registered clause is NOT_MET. Caveats: reduced grid (1–2 tiles per layer), 1000-image evaluations, best-of per tile on both sides.

2) *Exp. 2: M4 versus AdaBS:* Table IX and Fig. 4 give the network-level comparison (3 seeds, 2500 eval images, 2% probe noise, PCM, 1 h/1 d/30 d/1 yr; pooled over the scenarios on which every listed method was run: D2 and D3-hotspot $\rho=1$). *With BN kept (W1)*, M4 (87.29%) beats B8 (86.12%, +1.17 pt, CI [0.67, 1.87]; registered clause PASS) but is 0.61 pt below AdaBS-2600 (87.90%; CI [−1.21, +0.30]), and AdaBS-256 alone (87.58%) is already within 0.25–0.39 pt of AdaBS-2600 at one tenth of the cost, so there is no “10× cheaper than AdaBS” story: that saturation belongs to AdaBS. Stacking M4 on AdaBS-256 adds +0.32 pt over AdaBS-2600 (CI [0.04, 0.70]) at 2.6M MVM-eq. **The registered AdaBS-parity clause (match AdaBS-2600 within 0.3 pt at $\leq 1/10$ cost on ≥ 2 devices) was not met (L16):** the stored mechanical

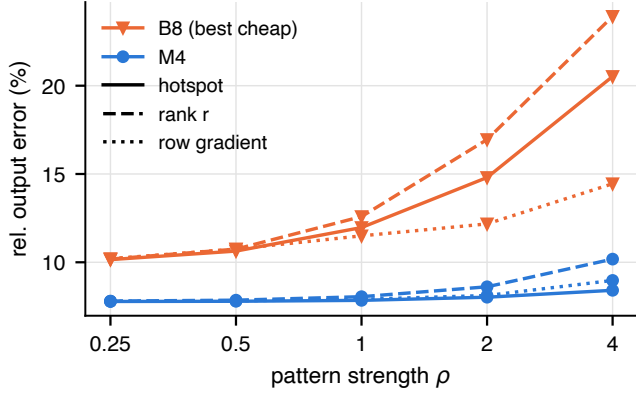


Fig. 3. Pre-run gate: relative output error of the best cheap incumbent (B8, orange) and field-fit M4 (blue) versus pattern strength ρ for three field shapes (read noise 2%). M4 is ahead already at $\rho = 0.25$ and the gap grows with ρ .

TABLE IX

EXP. 2: ACCURACY AND CALIBRATION COST (MVM-EQUIVALENTS PER WHOLE NETWORK), MEAN OVER 3 SEEDS, 4 TIMES AND 2 SCENARIOS. CI: PAIRED BOOTSTRAP OF THE DIFFERENCE TO B8.

Method	Cost	W1 (BN kept)	W2 (BN folded)
GDC	872	85.32	
B8	985 / 982	86.12	
M4 (8+8 probes + bias, 17 MVM/tile)	1853	87.29	
AdaBS-256	2.62M	87.58	
AdaBS-2600	26.6M	87.90	
M4+AdaBS-256	2.62M	88.22	
programmed (oracle)	—	87.73	

verdict reads INCONCLUSIVE because that clause found no rows (the script did not emit the device column and only PCM was run); read by hand from the per-scenario table, no M4 variant alone is within 0.3 pt of AdaBS-2600 (gaps 0.53–0.88 pt); the stacked M4+AdaBS-256 (one tenth of the cost) is within 0.3 pt on 2 of 4 scenarios; and only one device was evaluated. *With BN folded* (W2), AdaBS is not applicable, GDC reaches 80.58% and B8 81.42%, while M4 reaches 86.83% (+5.41 pt over B8, CI [3.75, 7.08]), equal to the programmed network (86.64%). That workload was the motivation for the BN-folded follow-up (a_bnfold), which was registered *after* seeing this number and is reported next to the parity miss.

3) a_bnfold: BN-folded deployment, registered follow-up:

Registration. Platform: ResNet-9 with BN folded; D2, D3 {hotspot, rank-r, row-gradient} at $\rho \in \{1, 2\}$ (6 scenarios) and D4: 8 scenarios; CIFAR-10 and CIFAR-100, 3 checkpoint seeds each; $t \in \{1h, 1d, 30d, 1yr\}$; $\sigma_c = 2\%$; 10k test images; streams offset +1000. Claims: (gap) median |gap to programmed| ≤ 0.5 pt; (gain) gain over B8 ≥ 2.0 pt with paired-CI lower bound > 0 at ≥ 3 of 4 times, per dataset; (budget) ≤ 16 calibration MVMs per tile; kills: no-harm (mean gain on iid drift < -0.3 pt) and no-gain (CI lower bound ≤ 0 at every time). **Budget readings.** Three readings exist and are reported together: (a) *strict 16*: 8 fwd + 7 bwd + 1 bias = 16 MVMs total; (b) *as run, registered count*: 16 field-fit probes counted, with the bias read as in the baselines

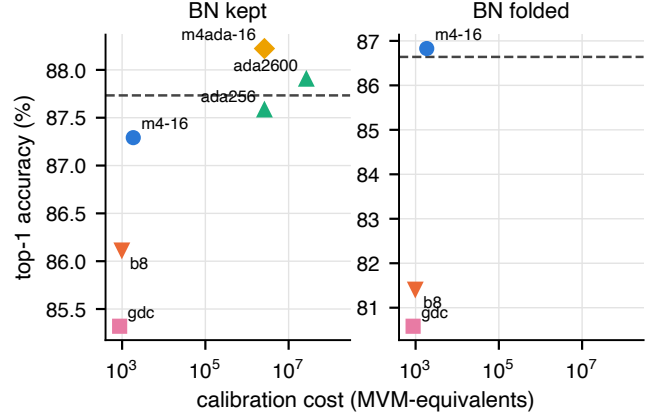


Fig. 4. Exp. 2 accuracy versus calibration cost. Dashed: programmed network. With BN kept, AdaBS buys its last 0.6 pt with 10^3 – $10^4\times$ more MVMs than M4; with BN folded only M4 (and cheap baselines) apply.

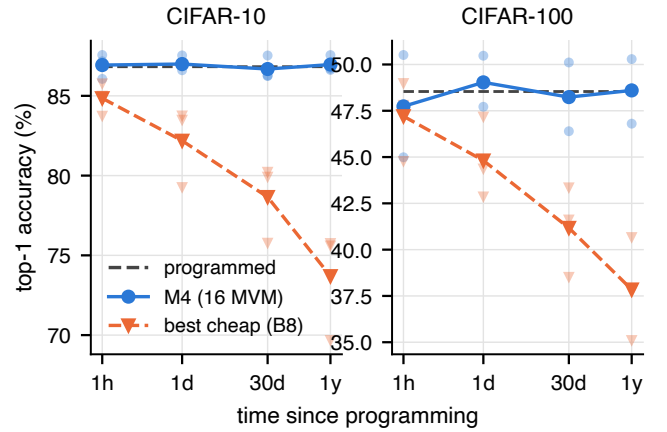


Fig. 5. BN-folded ResNet-9 under simulated PCM drift (mean of 8 scenarios $\times$ 3 checkpoints). M4 tracks the programmed network (dashed) from 1h to 1yr; the best cheap compensation falls 13.1 (CIFAR-10) and 10.7 (CIFAR-100) points below programmed accuracy at 1yr.

(script convention); (c) *as run, counting the bias read*: 17 total. Table X and Fig. 5 show the outcome per time point, Table XII per scenario, and Table XI the mechanical verdicts.

Reading. The headline passes only under the strict 16-MVM budget; if one insists on counting the bias read for the as-run variant the registered budget clause is not met and the verdict is INCONCLUSIVE. The gain clause passes with exactly the allowed fraction (0.75) on CIFAR-100: the 1h point misses 2 pt on both datasets under the as-run variant, and under the strict variant CIFAR-100 at 1h has a gain of +0.53 pt whose CI includes zero. From 1d to 1yr the effect is large and unambiguous; at 1h it is small and, for CIFAR-100, not distinguishable from zero. M4 also helps on iid drift (+3.2 to +3.8 pt), which is consistent with the pre-run gate: the gain is not specific to structure. An ablation (CIFAR-10, 1d and 1yr, all scenarios and seeds, report only) isolates the mechanism: in the ideal form even an order-1 field (one coefficient) from forward probes only reaches 85.2% / 85.0% (programmed 84.9%, B8 83.0% / 76.4%), and order 4 or transposed probes

TABLE X

A_BNFOLD FINAL (VALUES DIFFER FROM THE LATER BASELINE TABLE, TABLE XIII, FOR THE SAME QUANTITIES, E.G. B8 AT 1 YR ON CIFAR-10 73.70 vs 74.80, BECAUSE THAT RUN USED FRESH RANDOM STREAMS AND ACTIVATION STATISTICS FROM AUGMENTED TRAINING IMAGES; SEE SEC. V-D); TOP-1 ACCURACY (%), MEAN OVER 8 SCENARIOS $\times$ 3 CHECKPOINT SEEDS, 10K IMAGES. GAIN = M4 – B8; CI LO = LOWER END OF THE 95% PAIRED CLUSTER BOOTSTRAP CI (CLUSTERS = 3 CHECKPOINTS). STRICT = 8+7+1 MVMs; AS RUN = 8+8 PROBES + BIAS READ.

Dataset	t	Prog.	Strict (16 MVM)			As run (17 MVM)	
			M4	B8	Gain [CI lo]	M4	Gain [CI lo]
CIFAR-10	1 h	86.83	86.93	84.85	+2.08 [1.08]	86.58	+1.73 [1.01]
CIFAR-10	1 d	86.83	86.99	82.19	+4.81 [3.03]	86.46	+4.28 [2.50]
CIFAR-10	30 d	86.83	86.68	78.66	+8.03 [5.84]	86.83	+8.18 [6.03]
CIFAR-10	1 yr	86.83	86.96	73.70	+13.27 [9.93]	86.76	+13.07 [9.81]
CIFAR-100	1 h	48.54	47.74	47.20	+0.53 [–0.26]	48.55	+1.35 [0.81]
CIFAR-100	1 d	48.54	49.04	44.81	+4.22 [3.18]	48.73	+3.92 [2.43]
CIFAR-100	30 d	48.54	48.24	41.18	+7.06 [5.53]	49.06	+7.89 [6.09]
CIFAR-100	1 yr	48.54	48.60	37.85	+10.75 [8.76]	48.61	+10.76 [8.68]

TABLE XI

MECHANICAL VERDICTS OF A_BNFOLD.YAML UNDER THE THREE BUDGET READINGS (VALUES: C10 / C100).

Value		Strict 16	As run, count 16	As run, count 17
Gap	median gap to programmed (pt)	PASS (–0.09/–0.20)	PASS (0.13/–0.14)	PASS
Gain	time pts (of 4) clearing gain ≥ 2 , CI > 0	PASS (C10 4,4; C100 3,3)	PASS (C10 3,4; C100 3,4)	PASS
Budget	MVMs per tile	PASS (16)	PASS (16)	NOT_MET (17)
noharm	mean gain on iid D2 (pt)	CLEAR (+3.81/+3.23)	CLEAR (+3.57/+3.45)	CLEAR
kill	max CI lo over times (pt)	CLEAR (9.93/8.76)	CLEAR (9.81/8.68)	CLEAR
Overall		PASS	PASS	INCONCLUSIVE

TABLE XII

STRICT-VARIANT GAIN OVER B8 (PT) PER SCENARIO, MEAN OVER 3 SEEDS AND 4 TIMES.

Scenario	CIFAR-10	CIFAR-100
D2 (iid, sourced PCM)	3.81	3.23
D3 hotspot $\rho=1 / 2$	7.31 / 10.88	4.70 / 7.22
D3 rank- r $\rho=1 / 2$	5.78 / 11.66	4.89 / 9.97
D3 row-gradient $\rho=1 / 2$	5.78 / 6.22	5.12 / 5.51
D4 (temperature, assumed)	4.92	4.50

add at most 1.1 pt. The ideal correction’s strength therefore comes from using the stored digital $\mathbf{W}_0$ as the reference, not from the field’s richness or the read direction; in the free rank-1 deployable form all orders stay at 75.6–78.1% at 1 yr.

D. Deployability, stronger baselines and refresh

Four registered runs (Table IV) test what the M4 estimate is worth in hardware. All use the a_bnfold platform with fresh streams and, from the deployability run on, activation statistics from augmented *training* images only (this removes the test-image overlap of B8’s selection, Sec. VI-D).

Baselines (registered before any run). Digital recalibration: per-output-channel gain and bias from $N \in \{8, 32\}$ augmented training images (the closest analogue of AdaBS for a folded network); reference columns: per-column affine from reference columns [15]; timer adapters: offline-trained rank-4 adapters switched by a timer [12]. Non-inferiority: scenario-mean difference ≥ -0.3 pt at ≥ 3 of 4 times per dataset. The *ideal* M4 is non-inferior to all four (registered: all non-inferiority tests PASS), but that comparison only says the estimate is good.

The deployable form recovers much less. The free rank-1 form gains +0.1 to +1.5 pt over B8 (C10: +0.89, +0.10, +1.46, +0.65; C100: +0.41, +0.88, +1.37, +0.25 at 1 h/1 d/30 d/1 yr) with a paired-CI lower bound below zero at every CIFAR-100 time: the registered free-variant kill fired (L19). Reprogramming every tile recovers everything (86.2% C10, 48.9% C100 at 1 yr; one write per device per refresh).

Deployable M4 against the baselines (paired with the ideal-form run: same streams, baseline rows reused; pairing gate 0.00 pt). Table XIII. The full deployable form is non-inferior to the reference-column and timer baselines at every time on both datasets and beats B8 by 0.6–3.7 pt, but a digital recalibration from 32 images is 1.0–4.9 pt better from 1 d on (8 images: up to 3.2 pt better, tied at CIFAR-100 30 d): M4 is *competitive among image-free, weight-free methods*, not best.

M4 as a refresh scheduler. Since reprogramming recovers everything, the remaining use of the estimate is deciding *which* tile to reprogram and *when*. Registered model: each tile is refreshed at its own period from a 12-point age grid, chosen by a cost-weighted allocation from a health estimate (M4: predicted output error from 16 MVMs; GDC: drift-gain magnitude, 8 MVMs; STATIC: one nominal age curve, no sensing; ORACLE: the true error). Writes are per cell per year at equal worst-case network accuracy (Table XIV). Any per-tile cost-aware allocation saves 98.8 to 99.6% of the writes of a uniform timer at the 1-day target (from the medians in Table XIV, e.g. $1 - 4.44/365$ to $1 - 1.52/365$) (91.6% at the 30-day target, where every policy needs about one write per cell per year), because small tiles are cheap to refresh; M4’s sensing saves a further 9–13% of writes over STATIC at a 1-day target (median per-scenario ratio 0.87 and 0.91; bar 15%, NOT_MET) and nothing at 30 days, where every policy

TABLE XIII

BASELINES AND DEPLOYABLE M4: TOP-1 ACCURACY (%), MEAN OVER 8 SCENARIOS $\times$ 3 CHECKPOINTS, 10K IMAGES. VALUES FOR B8 AND IDEAL M4 DIFFER FROM TABLE X (E.G. B8 74.80 VS 73.70 AND IDEAL M4 86.29 VS 86.96 AT 1 YR ON CIFAR-10) BECAUSE THIS RUN USED FRESH RANDOM STREAMS AND ACTIVATION STATISTICS FROM AUGMENTED TRAINING IMAGES INSTEAD OF TEST IMAGES. M4-D = DEPLOYABLE M4 (RANK-1 FREE; FULL = UP TO 3 EXTRA ANALOG PASSES PER INFERENCE). IDEAL M4 NEEDS DIGITAL $\mathbf{W}_0$ AT INFERENCE (UPPER REFERENCE).

Method	Cost per recalibration	CIFAR-10				CIFAR-100			
		1 h	1 d	30 d	1 yr	1 h	1 d	30 d	1 yr
No correction	0	49.31	24.29	14.48	12.56	16.25	3.98	1.58	1.17
Timer adapters	0 reads, trained offline	84.74	82.23	77.28	72.31	46.61	42.14	34.93	29.13
Reference columns	reference columns per tile	85.21	82.87	78.66	74.04	45.89	43.24	38.31	33.31
B8 (optimistic)	8 MVM/tile	84.54	82.47	78.52	74.80	45.85	43.70	39.80	34.06
M4-d rank 1	16 MVM/tile, 0 extra passes	84.94	82.48	77.49	72.46	46.69	43.63	38.79	33.65
M4-d full	16 MVM/tile, ≤ 3 extra passes	85.14	83.21	79.87	76.48	46.96	44.55	41.70	37.75
Digital recal., 8 images	8 images	85.49	84.19	81.97	79.72	47.59	45.42	41.68	38.27
Digital recal., 32 images	32 images	85.94	84.88	83.15	81.40	47.77	45.80	42.70	39.80
M4 ideal (needs digital $\mathbf{W}_0$)	16 MVM/tile + digital MVM	86.58	86.31	86.41	86.29	48.83	48.73	49.20	49.21

TABLE XIV

REFRESH SCHEDULING: WRITES PER CELL PER YEAR TO MATCH A UNIFORM TIMER'S WORST-CASE ACCURACY (MEDIAN OVER 3 SEEDS $\times$ 8 SCENARIOS).

Policy	CIFAR-10		CIFAR-100	
	1-day	30-day	1-day	30-day
Timer	365	12.2	365	12.2
GDC allocation	2.44	1.03	3.46	1.01
STATIC (no sensor)	2.07	1.01	4.44	1.01
M4 allocation	1.61	1.02	2.91	1.01
Oracle allocation	1.52	1.02	2.25	1.01

needs about one write per cell per year (L20). The medians in Table XIV are medians of writes, not of per-scenario ratios, and so imply larger savings (for example 1.61 against 2.07 on CIFAR-10).

VI. DISCUSSION AND LIMITATIONS

A. What holds and what failed its test

A few counted probe reads tell a controller a lot about an aging crossbar.

What holds. (i) The joint decoder localizes tens of stuck cells in about a dozen reads at 1–2% noise, given the stored digital weights and an otherwise ideal tile. For $k \geq 4$ it needs 4.3 to 6.4 $\times$ fewer reads than our hybrid built from a published multi-outlier code at 1% noise (registered PASS under our own pass rule, before charging the 8 noise-calibration reads, Sec. IV), most of it owed to the decoder, since the forward-only form already needs 1.6 to 4.1 $\times$ fewer. It needs at least 2 $\times$ fewer reads than tuned X-ABFT, or X-ABFT misses F1 0.9 (registered PASS, with regime caveats); tuned X-ABFT needs more reads than a full readout. At 4% noise and $k \geq 4$ it reaches F1 0.93 to 0.95 where our reimplementations of the baselines do not. This last comparison is report-level, not a registered clause, and it does not hold at $k = 1$. (ii) The drift estimate is accurate: The ideal (digital) correction, an upper reference that needs the stored weights at inference, restores programmed accuracy, and the health estimate correlates 0.98

with the true tile error. (iii) Concurrent regression on ordinary inputs reaches detection 0.99 and line F1 0.83 to 0.87 at 16k inferences, where our adaptations of per-inference checksum decoders do not. This is report-level; the registered verdict is INCONCLUSIVE with all three pass clauses NOT_MET.

What failed its pre-specified test, and why. The extra benefit of transposed reads over forward-only decoding is limited to small tiles or low noise, and two registered claims about it were killed. We have no theorem for it. In the array, without stored digital weights, the drift correction is competitive among image-free methods but loses to a few-image digital recalibration. As a refresh scheduler, sensing saves a further 9 to 13% of writes over a sensorless cost-aware schedule (median per-scenario ratio 0.87 and 0.91 at a 1-day target; bar 15%, NOT_MET).

Open. A proof of the advantage at low noise (an average-case bound on ghost cells). A decoder that works within 64 to 256 live inferences. Any result on real hardware or on faults other than stuck-at. Table IV lists 19 of the 21 criteria files with their mechanical verdicts, including those that were killed or not met. The two not listed (an invariants check and an eigen-loop simulator check) include no kill; the latter is INCONCLUSIVE.

B. Hand-off to the closed loop

The tools that survive (the field estimate, the sparse-recovery decoder in its forward-only form, concurrent regression) are the inputs to the closed-loop repair work. A first registered closed-loop test (PASS) supports that hand-off: a remap-and-retrain controller acting on a fault map estimated from counted probe reads (the forward-only compressed-sensing decoder with 24 dense Rademacher probes and an OMP cap of 64, a full-readout fallback and 8 calibration reads charged per tile, 2% read noise; the bidirectional decoder is not used in this loop) matched its own oracle-map accuracy within 0.25 pt on ResNet-9 (3 seeds; 27% stuck cells, and a sparse $\approx 0.3\%$ regime where the decoder used 0.23–0.25 $\times$ the reads of full readout). The registered accuracy bars were -1.0 pt (dense) and -0.5 pt

(sparse) mean gap. The observed gap (≤ 0.25 pt) is the size of the run-to-run noise of the oracle itself (0.24 pt), so this is a no-difference-detected result, not an equivalence test. In the dense regime the “estimated” map is largely the full-readout fallback (all 111 tiles in 4 of 5 waves), so only the sparse regime tests the efficient localizer. The dense oracle runs for seeds 42, 101 and 102 come from an earlier campaign on another machine, and the 0.25 pt range does not extend to five seeds, where the dense gap ranges from -0.42 to $+0.24$ pt. A registered extension (5 seeds, 4% read noise, ViT-tiny) kept the *group-mean* accuracy gap within 0.3 pt in every group (per-seed gaps from -0.88 to $+0.90$ pt for ViT dense). It fired its registered kill clause: at 4% noise the worst wave of the sparse regime found only 85 to 89% of the newly visible faults (bar 90%), and one ResNet-9 seed used $0.2525\times$ the reads (bar 0.25). Its details belong to a companion study of repair controllers (in preparation).

C. Limits (L1–L21)

The original motivation was an analog eigen loop pointed at the array’s own matrix. The widely quoted “about 10% error at 10% device variation” for the Sun–Ielmini circuits belongs to the linear-system circuit [43], not the eigenvector one, so our loop simulator was checked against first-order eigenvector perturbation theory instead (registered perturbation check, median relative difference 5.6×10^{-4} ; that run is INCONCLUSIVE overall). Table XV lists what we measured and why we do not build on it. Evidence tiers: **R** = registered, mechanical verdict; **A** = analysis or proof; **S** = exploratory numbers from an unregistered probe (re-implementation pending, treat as indicative). Three measured scaling laws stand out: the shift of σ_1 from one stuck cell goes as $\text{tile size}^{-1.40}$ and $k^{+0.81}$ (theory -1 and 0.5 ; NOT_MET, theory to be revised); the eigenvector error under device variation scales as $1/\text{gap}$ (slope -0.90 on real tiles in the registered perturbation run, band $[-1.3, -0.7]$; an earlier unregistered probe gave -1.00); and the trained-weight spectral gap σ_1/σ_2 is only 1.02–1.11, which makes settling slow and the eigenvector ill-conditioned. Figure 6 gives the main empirical picture: at 2% noise the σ_1 shift is at chance for one fault on every tile size and only reaches AUC 0.60–0.92 at $k=64$ simultaneous faults (tile 512 down to tile 64), while eight ± 1 probe reads already give AUC 0.72–0.89 for *one* fault.

D. Threats to Validity

Simulation only and the digital-twin bridge. Network accuracy is computed in weight space from our own device model; no silicon is used. The registered bridge checks did not meet their agreement bars. No-drift accuracy matches the stored checkpoints (PASS). Our drift + GDC against AIHWKit’s native drift + GDC differs by up to 1.34 pt (bar 1.0, NOT_MET). Pushing the *same* corrected matrices through the repository’s CrossSim system (8-bit DAC, calibrated 8-bit ADC) differs from the twin by up to 5.66 pt (bar 0.5, NOT_MET); the large gaps are almost all on B8 in BN-folded and hotspot cells (CrossSim is harsher on it), while M4 agrees within about 2 pt. On CrossSim the

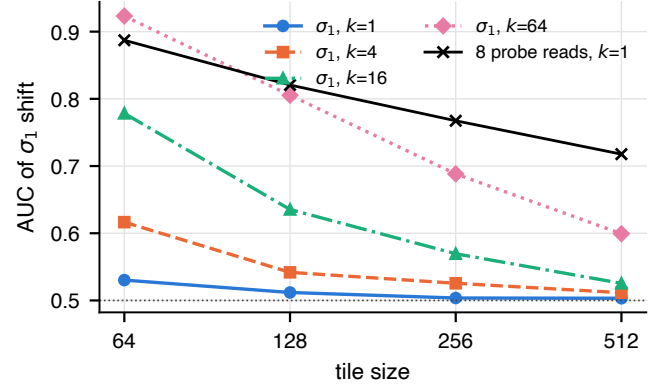


Fig. 6. AUC of the dominant-singular-value shift as a stuck-cell detector versus tile size and fault count k (2% read noise, signed encoding; mean over layers/seeds), against eight ± 1 probe reads for $k=1$ (dashed).

M4–B8 gain of the `a_bnfold` headline is *larger* than in the twin ($+3.3/+6.8/+11.2/+16.7$ pt vs $+2.1/+4.7/+8.3/+13.2$ at 1 h/1 d/30 d/1 yr), so relative comparisons are conservative, but absolute accuracies are not validated to 0.5 pt. A post-hoc diagnosis (report only; `f0_8_diag.py`) re-ran the 16 worst cells (BN-folded, seeds 101/102, D2 and hotspot, 1 d and 1 yr) on CrossSim at other converter resolutions: at 16-bit ADC and DAC CrossSim matches the twin to within 0.02 pt in every cell, so the twin computes the same network and the whole gap is converter quantization. Most of it comes from the 8-bit DAC (16-bit ADC with 8-bit DAC leaves 0.3–3.8 pt; 8-bit ADC with 16-bit DAC leaves ≤ 0.7 pt), and it hits B8 (2.5–4.8 pt) far harder than M4 (0.6–1.6 pt). **Localization idealizations.** The localizer uses the stored digital copy of the programmed weights, the test tile has no drift or programming noise, the noise level is estimated from 8 uncounted reads (charged in the closed loop), the decoder is given the cap $2k + 4$, probes and decoder settings are tuned per (k, σ_c) cell, read noise is white and independent across the forward and transposed reads of a cell, there is no ADC/DAC quantization, success is a mean F1 of 0.9 with no per-chip escape rate, and a transposed read assumes bidirectional periphery whose cost is unverified. Cell-persistent noise would correlate the two directions and weaken the second one. Localization was evaluated only on an otherwise ideal tile: the fault matrix E is exactly k -sparse, with no drift residual or programming error on top, so localization under a dense small background (the regime in which sparse recovery degrades) was not tested, nor was the combination of the localizer with the drift fit. Probes are ± 1 only; real-valued or index-weighted probes are not covered. A blocked X-ABFT positive control, inside its design regime, was not run. Statistics treat the 240 instances per cell as independent although they are clustered (3 checkpoints $\times$ 4 layers $\times$ 20 fault draws), so the confidence bounds and paired-CI upper bounds on read ratios are too tight. **Inference-time cost of the correction.** Measured in the deployability runs (Sec. V-D): the ideal form costs a digital MVM per tile, the deployable full form up to 3 extra analog passes per inference, rank 1 nothing. The headline `a_bnfold` numbers use the ideal

TABLE XV
LIMITS L1–L21. NUMBERS ARE FROM THE CITED RUN DIRECTORIES; TIERS AS DEFINED IN THE TEXT.

#	Investigated	Finding	Tier	Source
L1	σ_1 as a fault signature	One stuck cell moves σ_1 by $\approx \delta/\sqrt{mn}$ (relative $\sim 10^{-4}$), below read noise. AUC < 0.60 in 99.4% of (layer, tile, encoding, seed) cells at $k=1$, 2% noise, max 0.59; identical for signed and G^+/G^- . Shift $\propto \text{tile}^{-1.40}, k^{0.81}$.	R	f0_1/ 20261009T023921_a8f8370e_s0
L2	Self-check with no stored reference	Impossible: $\langle y, Ax \rangle = \langle A^T y, x \rangle$ uses the same cells, so forward and backward reads always agree.	A	proof
L3	Gain near the stability edge, shift-invert, deflation	Amplification is paid back in settling time or noise; shift-invert only guaranteed stable for positive-definite matrices.	A	analysis
L4	Rank-1 pilot to widen the gap	Fast settling, but the eigenvector shift is then a probe with $x = v_0$, i.e. a weighted checksum; scalar σ_1 stays blind (0% at every β).	S	exploratory
L5	Spectral gap of trained weights	$\sigma_1/\sigma_2 = 1.02$ – 1.11 : slow loop settling, ill-conditioned eigenvector, noise-dominated “mode-shape” localization.	S	exploratory
L6	Loop with hard-clipping amplifiers	All outputs latch to the rails at $\beta \geq 3$; localization 0%. Needs amplitude control.	S	exploratory
L7	“Analog-only, no ADC” as novelty	Already patented (HPE comparators on a syndrome; NVIDIA ADC-free power monitoring) [4], [21], [29].	A	literature
L8	Uniform drift as evidence	Our first test (all cells +5% toward a rail) is an additive rank-1 change, near tautological; uniform multiplicative drift is fixed exactly by GDC.	A	analysis
L9	Random ± 1 probes, both directions	No two-direction advantage once reads reach about 16: information per read is symmetric. Gain needs structured designs.	S	exploratory
L10	Low-rank capture weighted by real inputs	Equal for random and patterned drift (inputs sit in few principal components); not evidence of drift structure.	S	exploratory
L11	Spectral/low-rank WATCH classifier	M1 partial AUC given GDC factor 0.60 (< 0.7); temperature/drift/aging not separable (Jacobian rank deficit 1). Dropped by a frozen rule.	R	a_gate0/ 20261009T020655_ae67b976_s0
L12	Drift-structure-specific compensation	M4 beats the best cheap incumbent at every structure level: specificity 0.095 < 0.15. Gain is generic calibration.	R	a_gate0
L13	Spectral detection of stuck lines	σ_1 AUC for a fully stuck line on tiles ≥ 128 : 0.69 (NOT_MET); a 1-row checksum matches within 0.006 AUC at 0.8% of the cost.	R	f0_1
L14	σ_1 ratio as drift watchdog	AUC 0.92 but only +0.008 over the free GDC-factor statistic (bar 0.05): KILL.	R	f0_3/ 20261009T181003_aca2558e_s0
L15	Bidirectional localization, first claim	Reads ratio held out (0.28–0.75 on 29/30 rows) but F1 0.857–0.899: 0/5 k qualify, KILL.	R	b_c1_prereg/ 20261009T035140_0fdb78ed_s0
L16	AdaBS accuracy at $\geq 10\times$ fewer MVMs	M4 0.61 pt below AdaBS-2600 (CI $[-1.21, 0.30]$); AdaBS-256 already near AdaBS-2600: S2 not met.	R	a_exp2/ 20261009T052819_52891c48_s0
L17	Follow-up claim	Tile 128 PASS 5/5, tile 256 0/5, KILL. Bidirectional advantage real at 1% noise (2–3 $\times$), fades at 2%.	R	b_c1b/ 20261009T115059_609a698a_s0
L18	Analog eigen loop as localizer	Registered: median SNR ratio 0.24 < 1/3 against a ± 1 probe at equal output noise (KILL), and a second clause KILL on 18 cells / 3 seeds. Unregistered exploratory probes (indicative only): settles in 0.05–0.19 $\times$ one MVM when latch-free; signal 4–14 $\times$ weaker than a probe; complex dominant eigenvalues on un-dilated tiles (50–100% of tiles); never reaches 90% localization at the probe’s noise point; at probe-matched rail amplitude SNR ratio 0.45–0.57, localization 25–51%, and the speed advantage disappears. Simulates the Sun-type leading-eigenvector loop, not the Mannocci λ -sweep circuit.	R	c_x1/ 20261010T023044_1e4807e2_s0
L19	M4 applied in the array without digital W_0	Free rank-1 form: +0.1 to +1.5 pt over B8, CI lower bound < 0 at every CIFAR-100 time (registered kill). Full form (≤ 3 extra passes) beats the reference-column and timer baselines but loses to 8–32-image recalibration.	R	a_deploy/ 20261010T223102; a_harden_deploy/ 20261011T041351
L20	M4 health estimate as refresh allocator	Cost-aware allocation alone saves 98.8 to 99.6% of timer writes at a 1-day target (91.6% at 30 days); M4’s sensing saves a further 9–13% of writes over sensorless allocation at a 1-day target (median per-scenario ratio 0.87 and 0.91; bar 15%, NOT_MET), nothing at 30 days.	R	a_refresh/ 20261011T185337
L21	Concurrent detection/localization within 64–256 inferences	Per-inference checksum SNR ≈ 0.05 at 2% noise; regression detection 0.05 at 64 MVMs, line F1 0.06 at 256 (bars 0.99/0.9). Works at thousands of inferences (Sec. V-B).	R	b_concurrent/ 20261011T152741

form and are an upper reference. **Counting of the probe budget.** 16 vs 17 MVMs (Table XI); backward reads are assumed to cost one MVM; reference reads Y_0, Z_0 are stored digitally (15 vectors per tile) and their capture at programming time is not charged to recalibration; a transposable array is required. **Statistics.** Three checkpoint seeds per dataset, so the paired bootstrap resamples 3 clusters (its lower bound is a coarse guide); headline binomial standard error at 10k images is about 0.35 pt at 87% accuracy. **Incumbent selection and test-set reuse.** B8 picks its best member per tile using the programmed weights as ground truth on held-out im2col activations built from the first 5000 CIFAR test images (the same images overlap the 10k evaluation set); the layer-mean input for the bias step comes from the same cache. This favors the incumbent (a deployable B8 would be worse), so it makes the reported gain conservative, but absolute accuracies of both arms may be slightly optimistic. From the deployability run on, every statistic comes from augmented training images.

Absolute accuracy. The as-programmed networks sit 1.3 pt (CIFAR-10) and 3.3 pt (CIFAR-100) below float (88.15% and 51.87% mean); M4 restores programmed, not float, accuracy; programming error is out of scope. **Scope.** Probing beats a full readout only below about 0.3% fault density (exploratory, tile 256). One architecture (ResNet-9), 256 $\times$ 256 tiles, PCM only for the headline (RRAM/CMO and FeFET paths exist in the simulator but have no network result); structured drift fields and temperature parameters are assumed; no ADC/DAC quantization (except the CrossSim tier of the bridge check) and white (not 1/f-correlated) read noise. The localization results (Sec. V-A, V-B) use stuck-at faults only. **Pre-specification.** Criteria were committed in the author’s development repository, not to a third-party registry. a_bnfold and b_c1b were registered after seeing the results that motivated them; they used fresh random streams but not fresh checkpoints (CIFAR-10) and, for the follow-up claim, a tile scope chosen after tile 512 failed. The thin CIFAR-100 1 h point and the two budget

readings are reported rather than resolved.

VII. CONCLUSION

A deployed analog chip does not have to read every cell to find what broke. A sparse-recovery decoder over random ± 1 probes localizes up to 64 stuck cells in a 256×256 tile with 4 to 12 counted reads at 1% read noise. For $k \geq 4$ it needs 4.3 to $6.4\times$ fewer reads than a hybrid that we built from a published multi-outlier code, and at least $2\times$ fewer than tuned X-ABFT, which here needs more reads than a full readout. Most of that gain belongs to the decoder, which helps with forward reads alone. Reading from both sides adds 1.3 to $4\times$ at 1% noise, against a ± 1 -probe floor of about 9 reads for one fault in one direction. For drift, 16 parameters fitted from a few probe reads per tile restore programmed accuracy as a digital correction. Every claim was graded by a script against criteria fixed before the counted run, and the claims that failed are reported next to the ones that passed. What this enables: a controller that spends a handful of reads per tile can keep a map of failed cells current, which is the input a remap-and-repair loop needs; the closed-loop test in Sec. VI ran on such a map. Next come real-device faults, localization under a drift residual, and an average-case proof of the low-noise advantage (Sec. VI-C).

REPRODUCIBILITY

Code, configurations, raw run records and analysis scripts are in the public repository <https://github.com/prateek1656/NeuroFault>: library code under `src/neurofault/spectral/`, experiments under `experiments/spectral/` with frozen criteria in `experiments/spectral/criteria/*.yaml`, and the figure scripts (including the TikZ source of Fig. 1) in the repository's figure directory. Every run directory holds `config.json`, `params.json`, `run_metadata.json` (config hash, checkpoint SHA-256s, library versions) and `verdict.json`. Commit identifiers cited in this paper refer to the author's development repository history, which the one-commit public release does not contain; they are available on request.

STATEMENTS AND DECLARATIONS

Funding. No funding was received for conducting this study.

Competing interests. The author has no financial interests to declare other than the following: provisional patent applications related to the methods described in this paper have been filed by the author. The author has no non-financial competing interests.

Author contributions. The author is the sole contributor: conception, software, experiments, analysis and writing.

Data and code availability. All code, configurations, raw run records, pre-specified criteria and analysis scripts are available at <https://github.com/prateek1656/NeuroFault>. A preprint of this paper is archived at <https://doi.org/10.5281/zenodo.23472381>.

Ethics approval and consent. Not applicable: the study is a simulation and involves no human or animal participants.

USE OF AI TOOLS

The author used generative AI tools (Claude) to assist with code and writing; the author takes full responsibility for the content.

APPENDIX

PROOF SKETCH OF LEMMA 1 AND PROPOSITION 1

Lemma 1. The atom of cell (i, j) is $\varphi_{ij} = [e_i \otimes a_j; e_j \otimes b_i]$. Its forward block lives on the forward measurements of output line i with pattern a_j . Its transposed block lives on the transposed measurements of input line j with pattern b_i . The inner product of two atoms is the sum of a forward term and a transposed term. The forward term is zero unless $i = i'$, and then it is $\langle a_j, a_{j'} \rangle$. The transposed term is zero unless $j = j'$, and then it is $\langle b_i, b_{i'} \rangle$. $\square$

Proposition 1 (sketch). Normalize the atoms, $\psi = \varphi/\|\varphi\|$, and whiten the reads by s . Then $y = \Psi_S c + \varepsilon$ with $\varepsilon \sim \mathcal{N}(0, I)$ and $|c_{ij}| \geq c_{\min} = \delta\sqrt{v}/s$, because $\|\varphi_{ij}\|^2 = v$. The event $G = \{\max_{\psi} |\langle \psi, \varepsilon \rangle| \leq b\}$ has probability at least $1 - \eta$ (Gaussian tail and a union over the mn atoms). On G we show by induction that matching pursuit picks only atoms of S . Suppose the chosen set $T \subset S$ and $T \neq S$. The residual is $u + \tilde{n}$ with $u = (I - P_T)\Psi_{S \setminus T} c$ and $\tilde{n} = (I - P_T)\varepsilon$. (i) For every atom, $|\langle \psi, \tilde{n} \rangle| \leq \kappa b$, because $\|G_T^{-1}\|_{\infty \rightarrow \infty} \leq 1/(1 - a')$. (ii) On S : $\|\Psi_S^T u\|_{\infty} \geq (1 - a')c_{\min}$. (iii) Off S : $|\langle \psi, u \rangle| \leq \frac{a}{1-a'} \|\Psi_S^T u\|_{\infty}$. The best atom of $S \setminus T$ then beats every atom outside S when $(1 - a - a')c_{\min} > 2\kappa b$. This is condition (1). It also exceeds the threshold κb . When $T = S$, every correlation is at most κb and the procedure stops. The steps follow Tropp [39] and Cai and Wang [40]. We checked Lemma 1 by hand. The proof of the proposition has not been independently reviewed.

REFERENCES

- [1] Y. A. Khan, Z. Han, and A. Sengupta, "STAR: Astrocyte-inspired state-augmented repair for supervised memristive AI hardware systems," arXiv preprint arXiv:2607.15415, 2026.
- [2] M. Liu, L. Xia, Y. Wang, and K. Chakrabarty, "Algorithmic fault detection for RRAM-based matrix operations," *ACM Transactions on Design Automation of Electronic Systems*, vol. 25, no. 3, pp. 1–31, May 2020.
- [3] H. Wei and R. M. Roth, "Multiple-error-correcting codes for analog computing on resistive crossbars," arXiv preprint arXiv:2402.13503, 2024.
- [4] J. P. Strachan, C. Li, and C. Graves, "Analog error detection and correction in analog in-memory crossbars," US Patent 11,804,859 B2, Oct., 2023, assignee: Hewlett Packard Enterprise Development LP. Continuation: US 12,063,052 B2 (2024).
- [5] L. Parrini, T. Soliman, B. Hettwer, J. M. Borrmann, S. Singh, A. Bende, V. Rana, F. Merchant, and N. Wehn, "Error detection and correction codes for safe in-memory computations," in *Proc. IEEE European Test Symposium (ETS)*, 2024, pp. 1–4.
- [6] K.-H. Huang and J. A. Abraham, "Algorithm-based fault tolerance for matrix operations," *IEEE Transactions on Computers*, vol. C-33, no. 6, pp. 518–528, 1984.
- [7] M. Le Gallo, A. Sebastian, G. Cherubini, H. Giefers, and E. Eleftheriou, "Compressed sensing with approximate message passing using in-memory computing," *IEEE Transactions on Electron Devices*, vol. 65, no. 10, pp. 4304–4312, 2018.
- [8] V. Joshi, M. Le Gallo, S. Haefeli, I. Boybat, S. R. Nandakumar, C. Piveteau, M. Dazzi, B. Rajendran, A. Sebastian, and E. Eleftheriou, "Accurate deep neural network inference using computational phase-change memory," *Nature Communications*, vol. 11, p. 2473, 2020.

- [9] M. Le Gallo, R. Khaddam-Aljameh, M. Stanisavljevic, A. Vasilopoulos, B. Kersting, M. Dazzi, G. Karunaratne, M. Brändli, A. Singh, S. M. Müller, J. Büchel, X. Timoneda, V. Joshi, M. J. Rasch, U. Egger, A. Garofalo, A. Petropoulos, T. Antonakopoulos, K. Brew, S. Choi, I. Ok, T. Philip, V. Chan, C. Silvestre, I. Ahsan, N. Saulnier, V. Narayanan, P. A. Francese, E. Eleftheriou, and A. Sebastian, "A 64-core mixed-signal in-memory compute chip based on phase-change memory for deep neural network inference," *Nature Electronics*, vol. 6, no. 9, pp. 680–693, 2023.
- [10] D. Bondi, S. Giangrosso, A. Salvatore, G. Panettieri, A. Al Mukdash, L. Pistolesi, A. Glukhov, L. Cattaneo, N. Lepri, P. Mannocci, M. Baldo, A. Bonfanti, and D. Ielmini, "Linear compensation of drift and mitigation of drift variations in PCM arrays for highly accurate, energy efficient in-memory computing," in *Proc. IEEE Int. Electron Devices Meeting (IEDM)*, 2025, pp. 1–4.
- [11] N. Summ, X. Wang, H. Borrás, B. Klein, and H. Fröning, "Beyond noise: Understanding and overcoming temperature effects in analog DNN inference," arXiv preprint arXiv:2609.15527, 2026.
- [12] W. Dong, K. Zhou, Z. Kong, Z. Yang, Q. Cheng, H. Li, J. Huang, J. Lan, Y. Li, M. Hashimoto, and L. Lin, "VeRA+: Vector-based lightweight digital compensation for drift-resilient RRAM in-memory computing," arXiv preprint arXiv:2603.26016, 2026.
- [13] T. Wu, C. Ding, W. Zhou, Y. Cheng, X. Feng, S. Wang, W. Xu, C. Shi, Z. Liu, and N. Wong, "HaLoRA: Hardware-aware low-rank adaptation for large language models based on hybrid compute-in-memory architecture," arXiv preprint arXiv:2502.19747, 2025.
- [14] W. Dong, K. Zhou, Z. Kong, Q. Cheng, J. Huang, Z. Yang, M. Hashimoto, and L. Lin, "Efficient calibration for RRAM-based in-memory computing using DoRA," arXiv preprint arXiv:2504.03763, 2025.
- [15] V. M. Joshi, S. Haefeli, M. Le Gallo-Bourdeau, I. Boybat Kara, and A. Sebastian, "Conductance drift corrections in neuromorphic systems based on crossbar array structures," US Patent 11,663,458 B2, May, 2023, assignee: International Business Machines Corp.
- [16] C. Mackin, "Compensation for conductance drift in analog memory in crossbar array," US Patent 12,272,422 B2, Apr., 2025, assignee: International Business Machines Corp.
- [17] D. N. Yadav, P. L. Thangkhiew, K. Datta, S. Chakraborty, R. Drechsler, and I. Sengupta, "FAMCrNA: Fault analysis in memristive crossbars for neuromorphic applications," *Journal of Electronic Testing*, vol. 38, no. 2, pp. 145–163, Apr. 2022.
- [18] C. Li, R. M. Roth, C. Graves, X. Sheng, and J. P. Strachan, "Analog error correcting codes for defect tolerant matrix multiplication in crossbars," in *Proc. IEEE Int. Electron Devices Meeting (IEDM)*, 2020, pp. 36.6.1–36.6.4.
- [19] A. R. Bommana, F. Firouzi, C. Ogbogu, B. K. Joardar, J. R. Doppa, P. P. Pande, and K. Chakrabarty, "SEC-CiM: Selective error compensation for ReRAM-based compute-in-memory," in *Proc. IEEE Int. Test Conf. (ITC)*, 2024, pp. 177–186.
- [20] A. R. Bommana, F. Firouzi, C. Ogbogu, B. Joardar, J. Doppa, P. P. Pande, and K. Chakrabarty, "SEC-CiM+: Selective and reconfigurable error compensation for ReRAM-based compute-in-memory," *ACM Transactions on Design Automation of Electronic Systems*, 2026.
- [21] A. S. Sharma, J. P. Strachan, C. Graves, S. Kumar, C. Warner, and M. Foltin, "Self-healing dot-product engine," US Patent 11,532,356 B2, Dec., 2022, assignee: Hewlett Packard Enterprise Development LP. Parent: US 10,984,860 B2 (2021).
- [22] B. Khezeli, H. R. Zarandi, and E. Cheshmikhani, "Online soft error tolerance in ReRAM crossbars for deep learning accelerators," arXiv preprint arXiv:2412.03089, 2024.
- [23] L. Xia, M. Liu, X. Ning, K. Chakrabarty, and Y. Wang, "Fault-tolerant training with on-line fault detection for RRAM-based neural computing systems," in *Proc. ACM/IEEE Design Automation Conference (DAC)*, 2017, pp. 1–6.
- [24] Z. Xu, J. Liu, S. Huang, Z. Li, S. Wang, B. Wen, R. Mao, M. Jiang, G. Pedretti, J. Ignowski, K. Huang, and C. Li, "Fault-free analogue computing with imperfect hardware," *Nature Electronics*, vol. 9, no. 7, pp. 811–824, 2026.
- [25] Z. Zhu, C. Yuan, R. M. Roth, P. H. Siegel, and A. Jiang, "A new class of geometric analog error correction codes for crossbar based in-memory computing," arXiv preprint arXiv:2603.03723, 2026.
- [26] K. X. Pinto, K. Kodali, A. Das, and N. A. Toubia, "Double adjacent error correction in RRAM matrix multiplication using weighted checksums," in *Proc. IEEE Int. Symp. on On-Line Testing and Robust System Design (IOLTS)*, 2024, pp. 1–5.
- [27] L. Venkatasubramanian, Z. Wan, and V. Cadambe, "FLARE: One-shot PE-level fault localization in systolic arrays via algebraic test vectors," arXiv preprint arXiv:2605.08594, 2026.
- [28] Q. Liu, T. Liu, Z. Liu, W. Wen, and C. Yang, "Monitoring the health of emerging neural network accelerators with cost-effective concurrent test," in *Proc. ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.
- [29] K. Chakrabarty and M. Liu, "Online fault detection in ReRAM-based AI/ML," US Patent 11,971,790 B2, Apr., 2024, assignee: Nvidia Corp.
- [30] M. Aldridge, O. Johnson, and J. Scarlett, *Group Testing: An Information Theory Perspective*, ser. Foundations and Trends in Communications and Information Theory. Now Publishers, 2019, vol. 15, no. 3–4.
- [31] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.
- [32] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.
- [33] E. J. Candès, R. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489–509, 2006.
- [34] Z. Sun, G. Pedretti, E. Ambrosi, A. Bricalli, and D. Ielmini, "In-memory eigenvector computation in time O(1)," *Advanced Intelligent Systems*, vol. 2, no. 8, p. 2000042, 2020.
- [35] P. Mannocci, E. Giannone, and D. Ielmini, "In-memory principal component analysis by analogue closed-loop eigendecomposition," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 71, no. 4, pp. 1839–1843, Apr. 2024.
- [36] P. Mannocci and D. Ielmini, "A generalized block-matrix circuit for closed-loop analog in-memory computing," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 9, no. 1, pp. 47–55, Jun. 2023.
- [37] C. Ding, Z. Liu, Y. Zhang, C. Li, J. Tang, B. Gao, H. Yu, H. Wu, and N. Wong, "Memristive singular value decomposition," *Nature Communications*, vol. 17, no. 1, p. 8006, 2026.
- [38] N. Boggaram, "A per-chip metadata correction framework for neuromorphic memristor crossbars: Theoretical architecture and viability analysis," EngrXiv preprint, doi:10.31224/6812, 2026.
- [39] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2231–2242, Oct. 2004.
- [40] T. T. Cai and L. Wang, "Orthogonal matching pursuit for sparse signal recovery with noise," *IEEE Transactions on Information Theory*, vol. 57, no. 7, pp. 4680–4688, Jul. 2011.
- [41] M. J. Rasch, D. Moreda, T. Gokmen, M. Le Gallo, F. Carta, C. Goldberg, K. El Maghraoui, A. Sebastian, and V. Narayanan, "A flexible and fast PyTorch toolkit for simulating training and inference on analog crossbar arrays," in *Proc. IEEE Int. Conf. on Artificial Intelligence Circuits and Systems (AICAS)*, 2021, pp. 1–4.
- [42] M. J. Rasch, C. Mackin, M. Le Gallo, A. Chen, A. Fasoli, F. Odermatt, N. Li, S. R. Nandakumar, P. Narayanan, H. Tsai, G. W. Burr, A. Sebastian, and V. Narayanan, "Hardware-aware training for large-scale and diverse deep learning inference workloads using in-memory computing-based accelerators," *Nature Communications*, vol. 14, p. 5282, 2023.
- [43] Z. Sun, G. Pedretti, E. Ambrosi, A. Bricalli, W. Wang, and D. Ielmini, "Solving matrix equations in one step with cross-point resistive arrays," *Proceedings of the National Academy of Sciences*, vol. 116, no. 10, pp. 4123–4128, 2019.