A Per-Chip Metadata Correction Framework for Neuromorphic Memristor Crossbars: Theoretical Architecture and Viability Analysis

Naman Boggaram

Abstract

We establish theoretically that device-level variability in memristor crossbars can be deterministically eliminated at model load time via per-chip system identification, removing the need for hardware-aware training. A system-identification protocol executed entirely through existing peripheral circuitry applies N Hadamard-structured input patterns to the array and recovers the full $N \times N$ conductance deviation matrix via a single matrix multiply, achieving an effective measurement noise floor of $\sigma_{\text{meas}}/\sqrt{N}$ — a 63× improvement over single-node sweeps for a 4000×4000 array. This noise reduction converts the measurability condition from a hard instrument-precision prerequisite into a convergence condition satisfied by construction. The correction itself is exact under the confirmed linear response of the Ru(22) device class, allows any pre-trained model to be deployed without retraining or hardware-aware fine-tuning, and is maintained over device lifetime by a periodic heartbeat recalibration protocol that corrects all noise sources regardless of origin. Theoretical illustration under a Gaussian spatial correlation model consistent with published spin-coating physics confirms 99.98% variance capture at K=16 DCT coefficients (2 KB storage, 31,250× compression). Ten-year operational energy for a 4000×4000 array under Architecture C is 140,836 J — below hardware-aware training for any per-run training cost above 3,520 J. Factory characterisation requires 17.6 s per chip using existing peripheral circuitry.

1. Introduction

Scope and theoretical basis. The analyses presented in this document are theoretical, derived from published device parameters of Goswami et al. [1]. Experimental validation of per-chip characterisation metadata generation, measurement noise bounds, and long-term drift behaviour under the heartbeat protocol requires chip-level access not available at the time of writing. This work establishes the theoretical framework and viability conditions; experimental confirmation is identified as future work.

Variability is the main blocker in analog neuromorphic hardware deployment. Memristor crossbar arrays offer orders-of-magnitude improvements in energy and area for matrix-vector multiplication [2, 3], but device-to-device conductance variations prevent reliable computation without correction.

Existing solutions have fundamental tradeoffs:

- **Hardware-aware training** couples models to hardware through noise injection during training, requiring retraining for every device class and producing residual errors since actual per-chip deviations differ from population statistics
- **Calibration and write-verify protocols** improve programming precision through iterative feedback but cannot distinguish systematic gain heterogeneity from cycle-to-cycle noise, and provide no correction for post-programming drift
- **Statistical robustness methods** accept variability as unavoidable noise, sacrificing computational efficiency through redundancy or error-correction codes

Compressed sensing (CS) has been proposed as a route to crossbar characterisation with fewer than N^2 explicit measurements, exploiting sparsity of the variability map in a transform basis and reconstructing conductance deviations from an underdetermined set of excitation patterns [4–6]. That line of work reduces pattern count, but the recovery step inherits the conditioning of the chosen sensing matrix: generic random or structured binary codes typically yield large condition numbers, which amplify measurement noise unevenly across nodes and make uniform per-node error guarantees difficult to state. Orthogonal Hadamard excitation—the protocol used here—is different in kind: the Hadamard matrix has condition number 1, so inversion introduces no differential noise amplification, and the effective noise floor scales as $\sigma_{\text{meas}}/\sqrt{N}$ uniformly across the array. Where the goal is guaranteed, node-wise measurability for full-matrix correction metadata rather than approximate reconstruction from a sparsity prior alone, condition-number-1 Hadamard system identification is therefore the more direct engineering choice.

No method currently removes variability deterministically per chip without retraining or inference overhead.

Our key idea: Instead of treating variability as stochastic noise, we treat it as a fixed, measurable system property that can be identified once and cancelled before inference begins.

In binary transistor systems, variability is catastrophic — a transistor switching at the wrong threshold turns a 1 into a 0, failing the entire computation. But memristor crossbars perform analog computation through Ohm's law at each crosspoint and Kirchhoff's summation at each column. If a node's conductance lands at $G_{\text{ideal}} + \delta$ instead of G_{ideal}, the error is continuous, bounded, and deterministic for that specific device.

The interface layer between model and hardware operates at model loading, not inference. When a pre-trained model's weight matrix W is programmed into crossbar conductance states, systematic per-node offsets $\delta(i, j)$ can be pre-compensated:

$$W_{\text{corrected}}(i, j) \;=\; \frac{W(i, j) \;-\; \delta(i, j)}{\alpha(i, j)} \tag{1}$$

where $\alpha(i, j)$ is the per-node gain coefficient. The device's physical offset then cancels the pre-compensation exactly, and the crossbar implements W as intended. This correction is

computed once per model load — a single matrix operation taking microseconds — and carries zero overhead during inference.

Our method:

- System identification via structured inputs to recover full deviation matrix
- Pre-compensation at model load using recovered per-node parameters
- Periodic heartbeat recalibration for lifetime drift correction

Our results:

- Exact correction under linearity assumption, with $\sigma \to \sigma/\sqrt{N}$ noise reduction making measurement feasible
- Zero inference overhead, enabling direct deployment of any pre-trained model
- Decouples model training from hardware, allowing deployment of arbitrary models on any characterised chip

Why it matters: This enables deployment of the entire ecosystem of pre-trained models — GPT variants, vision transformers, diffusion models trained on GPU clusters — directly on memristor hardware without modification or retraining.

2. Deterministic Correction via System Identification

We show that device-level variability can be deterministically eliminated through per-chip system identification that converts measurability from an instrument-precision constraint into a mathematically guaranteed convergence process.

2.1 Full-Matrix Recovery via Hadamard Excitation

Per-node characterisation is achieved through a system identification protocol executed using the array's existing peripheral circuitry: Hadamard-structured input patterns are applied simultaneously across all rows, and the full conductance deviation matrix is recovered from the measured column currents via a single matrix multiply.

Let G_{ideal} denote the target conductance matrix and ΔG the deviation matrix such that the implemented matrix is $G_{\text{ideal}} + \Delta G$. For a single input vector $\mathbf{x}$, the measured output is:

$$\mathbf{e} \;=\; (G_{\text{ideal}} + \Delta G)\,\mathbf{x} \tag{2}$$

Subtracting the known ideal response:

$$\mathbf{e} - G_{\text{ideal}}\,\mathbf{x} \;=\; \Delta G\,\mathbf{x} \tag{3}$$

Applying k input vectors $X = [\mathbf{x}_1, \ldots, \mathbf{x}_k]$ and collecting the residual outputs $E = [\mathbf{e}_1 - G_{\text{ideal}}\mathbf{x}_1, \ldots]$:

$$E \;=\; \Delta G \cdot X \tag{4}$$

Solving for ΔG:

$$\Delta G \;=\; E \cdot X^{-1} \tag{5}$$

For $k = N$ input vectors forming a square matrix X, and provided X is invertible, Equation (5) recovers the full $N \times N$ deviation matrix exactly (up to measurement noise). This square, full-rank formulation is the standard linear system-identification problem for a fixed linear map from inputs to outputs [7].

2.2 Hadamard Matrices: Optimal Conditioning and Noise Reduction

The choice of X determines both the numerical conditioning of the inversion and the noise amplification in the recovered ΔG. The optimal choice is a Hadamard matrix H of order N: an $N \times N$ matrix with entries $\pm 1/\sqrt{N}$ satisfying $HH^T = I$. Classical constructions and extremal spectral properties of Hadamard matrices are surveyed by Hedayat and Wallis [8].

Two properties make Hadamard matrices uniquely suited to this application.

Condition number 1. The condition number of a Hadamard matrix is exactly 1, the minimum achievable for any matrix. This means the matrix inversion step $E \cdot H^{-1} = E \cdot H^T/N$ introduces no numerical amplification of residuals: the recovered ΔG inherits noise from E at a uniform gain of $1/N$ per entry.

Noise reduction by $\sqrt{N}$. Each entry of ΔG is recovered as the inner product of one row of E with one row of H^T. Since measurement noise on each element of E is σ_{meas} and each inner product averages N such terms, the variance on each recovered $\Delta G(i,j)$ entry is:

$$\text{Var}[\widehat{\Delta G}(i,j)] \;=\; \frac{1}{N^2} \cdot N \cdot \sigma_{\text{meas}}^2 \;=\; \frac{\sigma_{\text{meas}}^2}{N} \tag{6}$$

so the effective noise per recovered entry is $\sigma_{\text{eff}} = \sigma_{\text{meas}}/\sqrt{N}$. The same $\mathcal{O}(1/\sqrt{N})$ scaling arises whenever independent, zero-mean measurement errors are averaged through an orthogonal linear combination of N observations [9]. For $N = 4000$:

$$\sigma_{\text{eff}} \;=\; \frac{\sigma_{\text{meas}}}{\sqrt{4000}} \;\approx\; \frac{\sigma_{\text{meas}}}{63.2} \tag{7}$$

This $63\times$ noise reduction converts the measurability condition from a hard instrument-precision requirement into a convergence condition: even an instrument with noise floor $63\times$ higher than

$\Delta G_{\min}/6$ will recover ΔG with sufficient accuracy.

2.3 Protocol and Timing

Passive crossbar arrays exhibit sneak-path currents that complicate ideal single-cell access in dense meshes [10]. The identification pipeline here operates on the measured terminal currents produced by programmed conductance states under structured row excitations; uncompensated parasitic coupling therefore appears as an effective contribution to the identified ΔG at the interface layer rather than as a separate unmodelled operator, provided the linear measurement model of Section 2.1 remains valid for the packaged array.

Factory characterisation procedure:

1. Programme all nodes to a uniform reference conductance state $G_{\text{ideal}} = G_{\text{ref}} \cdot 1$, where 1 denotes the all-ones matrix, using standard pulse sequences.
2. Apply $N = 4000$ Hadamard input voltage patterns row-wise, one per pass. Each pass takes one full array read: 4000 rows $\times 1.1\,\mu$s $= 4.4$ ms per pass.
3. Record the $N \times N$ output current matrix E.
4. Recover $\Delta G = E \cdot H^T/N$ via a single matrix multiply on the peripheral processor.
5. Fit the per-node linear model $G_{\text{meas}}(i,j) = \alpha(i,j)\, G_{\text{prog}}(i,j) + \delta(i,j)$ to extract α and δ.
6. Compress and store as per Architectures A, B, or C (Section 6).

Total characterisation time:

$$t_{\text{char}} \;=\; N \times t_{\text{pass}} \;=\; 4000 \times 4.4\,\text{ms} \;=\; 17.6\,\text{s per chip} \tag{8}$$

This is a one-time factory operation. At 17.6 s per chip, a single characterisation station running continuously could process over 4,900 chips per day.

2.4 Measurability and Self-Correction

The effective measurability condition, accounting for Hadamard averaging, is:

$$\frac{\sigma_{\text{meas}}}{\sqrt{N}} \;<\; \frac{\Delta G_{\min}}{6} \tag{9}$$

For the device class of Goswami et al. [1] with $\Delta G_{\min}$ resolved across all 16,520 conductance levels [11], and $N = 4000$, this condition permits an instrument noise floor up to $63 \times \Delta G_{\min}/6$ — well within the range of standard ADC components. The condition is therefore satisfied by construction.

Furthermore, since the heartbeat recalibration applies the same Hadamard characterisation protocol at each cycle, the stored $\delta(i,j)$ estimates are iteratively refined. The system is self-correcting by construction: each recalibration pass improves the stored correction, and the

accumulated precision after k heartbeat cycles scales as $\sigma_{\text{meas}}/(k\sqrt{N})$ under independent noise draws.

3. Complete Model-Hardware Decoupling

The approach decouples model development from hardware development by enabling deployment of arbitrary pre-trained models without hardware-aware training. This represents a fundamental shift from current neuromorphic computing paradigms.

3.1 What hardware-aware training does

Hardware-aware training (HAT) has been established as a natural baseline for deploying neural networks on analog hardware with device variability [2, 12–14]. HAT couples the trained model to the hardware through noise injection during training — modelling the statistical distribution $P(\delta)$, or through direct on-hardware training. The trained weights W_{HAT} satisfy:

$$W_{\text{HAT}} \;=\; \arg\min_{W} \mathcal{L}_{\text{task}}(W) \;+\; \lambda \cdot \mathbb{E}_{\delta \sim P(\delta)} \left[\| f(X,\, W + \delta) - f(X,\, W) \|^2 \right] \tag{10}$$

where λ is the robustness regularisation weight and f is the network function. The second term penalises sensitivity to device offsets, forcing the network to learn representations that degrade gracefully under expected variability. This is mathematically a regularisation term: it reduces model capacity in exchange for hardware robustness [13, 14].

The residual per-chip inference error after HAT is:

$$\varepsilon_{\text{HAT}}(i,j) \;=\; \delta_{\text{actual}}(i,j) \;-\; \mathbb{E}[\delta(i,j)] \tag{11}$$

This residual is nonzero for every physical chip because every chip's actual offset deviates from the population mean. HAT corrects for the average. It cannot correct for the specific.

3.2 What deterministic correction does differently

The present method imposes no constraint on how the model was trained. The correction in Equation (1) operates entirely at the interface layer. The residual inference error is:

$$\varepsilon_{\text{corrected}}(i,j) \;\approx\; 0 \text{ [S]} \tag{12}$$

under the measurability condition $\sigma_{\text{meas}} < \Delta G_{\min}/6$, satisfied at the device level by Goswami et al. [1] [Fig. 8e] and confirmed by Sharma et al., who verified $\Delta G_n > 6\,\sigma_n$ across all 16,520 conductance levels [11]. The Hadamard characterisation protocol reduces the effective noise floor by $\sqrt{N}$, converting this condition into a convergence criterion rather than a hard instrument-precision prerequisite.

3.3 The non-compensated model advantage

Measurability condition derivation. The condition $\sigma_{\text{meas}} < \Delta G_{\min}/6$ is a 6-sigma argument: if measurement noise on each characterisation reading is σ_{meas}, the stored offset $\delta(i,j)$ carries an error $\varepsilon \sim \mathcal{N}(0, \sigma^2_{\text{meas}})$. For the correction to reliably distinguish the nearest two conductance levels (separated by $\Delta G_{\min}$), the probability of misclassification must be negligible. Requiring $6\sigma_{\text{meas}} < \Delta G_{\min}$ bounds this probability at $< 10^{-9}$ per node, yielding:

$$\sigma_{\text{meas}} \;<\; \frac{\Delta G_{\min}}{6} \tag{13}$$

Effective noise floor under Hadamard characterisation. As derived in Section 2.2, the Hadamard-based system identification protocol achieves an effective per-node noise floor of $\sigma_{\text{meas}}/\sqrt{N}$ after matrix inversion. For $N = 4000$, this is a 63× reduction relative to single-node measurements. The measurability condition therefore reads $\sigma_{\text{meas}}/\sqrt{N} < \Delta G_{\min}/6$ in practice, and is satisfied iteratively by the heartbeat protocol which continuously refines the stored $\delta(i,j)$ estimates.

This is the property that HAT structurally cannot replicate: a model that has never seen a memristor crossbar can be loaded and run accurately. HAT requires that the model be retrained whenever it is deployed on a new device class, a new process node, or hardware whose variability profile differs from the training assumption. Every existing pre-trained model — GPT-scale language models [15], vision transformers [16], diffusion models [17], or any model trained on GPU clusters without neuromorphic hardware knowledge — would require retraining or fine-tuning under hardware-aware objectives before deployment on memristor hardware.

Per-chip correction deploys any such model directly and without modification. The existing ecosystem of pre-trained models becomes immediately deployable on characterised crossbar chips. The correction absorbs the hardware's variability at load time; the model never needs to know the hardware exists.

Formally, let W^* denote the optimal weights for the task on ideal hardware. HAT produces $W_{\text{HAT}} \neq W^*$. The gap $W_{\text{HAT}} - W^*$ represents capacity lost to robustness regularisation. Per-chip correction deploys W^* directly. On ideal hardware, W^* outperforms W_{HAT}. On characterised crossbar hardware under the present method, W^* also achieves near-exact inference at the moment of load.

3.4 Multi-Layer Deployment

In multi-layer neural networks, weight matrices are distributed across multiple crossbar arrays. The per-chip correction of Equation (1) is applied independently to each array at model load time: the correction for layer ℓ uses only the metadata of the corresponding array ℓ, and the pre-compensated weights for layer ℓ are programmed before inference begins. Activation functions between layers are evaluated in digital logic on the peripheral processor and are not subject to crossbar analog variability, consistent with standard processing-in-memory architectures that interleave dense analog crossbars with CMOS periphery [3]. The correction is therefore layer-wise

exact: each crossbar implements its target weight matrix with the residual error characterised in Section 5.1, and nonlinear activations introduce no additional correction burden. Multi-layer deployment requires one metadata file per crossbar array; storage scales linearly with the number of layers.

4. Universal Noise Correction via Heartbeat Protocol

The correction remains accurate over device lifetime through a periodic recalibration protocol — a "heartbeat" — that re-measures current conductance states, computes updated offsets, and reprograms only the nodes whose drift has exceeded a defined threshold. The heartbeat functions as a universal noise corrector: any source of conductance deviation, whether from thermal drift, write variability, or environmental perturbation, is identified and corrected by the same mechanism regardless of physical origin.

4.1 Connection to System Identification: Universal Noise Correction

The heartbeat recalibration protocol applies the same system identification measurement at each cycle, updating the stored $\delta(i, j)$ values for nodes whose conductance has drifted beyond the correction threshold. A key property of this architecture is that the heartbeat does not distinguish between sources of conductance deviation: thermal drift, write noise residuals, environmental perturbation, and any other source of ΔG perturbation are all captured in the measured deviation and corrected identically. The system is therefore self-correcting by construction — the heartbeat provides universal noise correction regardless of the physical origin of the conductance error. This is distinct from HAT, which is trained against a specific noise model and degrades when actual device noise deviates from the training distribution.

4.2 Distinction from Iterative Write-Verify Programming

A related but distinct literature addresses programming accuracy through iterative write-verify (IWV) protocols [18], including mapping-level implementations for crossbar arrays [19]: the device is programmed, its conductance read back, and additional programming pulses applied until the measured state falls within a target window. IWV improves programming precision and reduces stuck-at-fault sensitivity, but it operates at the level of individual programming events. It does not produce a persistent characterisation of device behaviour that travels with the chip; it cannot distinguish node-specific gain heterogeneity from cycle-to-cycle write noise; and it provides no correction path for post-programming drift that accumulates during inference operation. The present framework is complementary: IWV can be used to improve the precision of the initial crossbar write, while per-chip metadata correction addresses the layer above — the stable systematic offsets that IWV leaves uncorrected and the drift that accumulates after programming completes.

5. Honest Limitations

Three limitations of the present approach must be stated directly.

The characterisation infrastructure prerequisite. Per-chip correction requires characterisation to be integrated into the manufacturing process before chips are deployed. As shown in Section 2, this characterisation is performed using the crossbar's existing peripheral circuitry via the system identification protocol — no specialised external instrument is required. The 17.6 s factory characterisation step is the only additional manufacturing requirement. HAT requires only a training pipeline; per-chip correction requires that pipeline plus a 17.6 s per-chip characterisation pass. Both are viable manufacturing steps.

The heartbeat energy dominates at long deployment. From the energy analysis in Section 6, the heartbeat protocol accounts for 140,808 J of the total 140,836 J — 99.98% of all energy consumed over 10 years. The optimal heartbeat interval is informed by the drift data of Goswami et al. [1], which demonstrate month-scale stability. The 1-hour interval used throughout this analysis is therefore conservative by approximately 700×. If the interval is extended to 6 hours on the basis of operational data, total energy falls to 23,496 J. The heartbeat interval is a design parameter: it is set by the system operator to balance energy cost against the acceptable residual correction error between recalibrations.

The linear response model is confirmed for the specific device class studied here, with one noted caveat. The two-parameter linear correction of Equation (1) assumes that a single gain coefficient $\alpha(i,j)$ fitted at characterisation time accurately predicts the device response for arbitrary target conductances spanning the full range of neural network weight matrices. Whether this assumption holds with uniform accuracy across that full range has not been independently verified end-to-end and represents the primary unverified assumption of the present framework. Within this caveat, the evidence for the Ru(22) device class is strong: Sharma et al. demonstrated 16,520 distinct analog conductance levels that are linearly and symmetrically updated in one time step [11], explicitly verified $\Delta G_n > 6\sigma_n$ across all levels, and Figure 8a of Goswami et al. [1] confirms global linearity across the full conductance range. The two-parameter model is therefore well-matched to this device class. For other memristor technologies exhibiting nonlinear response, a higher-order characterisation protocol would be required.

5.1 Error Propagation Under Measurement Noise

Since per-node characterisation data are not yet available, the correction error can be bounded analytically. Under the Hadamard protocol, the effective noise per recovered $\delta(i,j)$ entry is $\sigma_{\text{eff}} = \sigma_{\text{meas}}/\sqrt{N}$ (Section 2.2). After correction, the residual inference error at node (i,j) is:

$$\varepsilon_{\text{corrected}}(i,j) \;\approx\; \frac{\varepsilon(i,j)}{\alpha(i,j)} \;\sim\; \mathcal{N}\!\left(0,\, \frac{\sigma^2_{\text{meas}}}{N\,\alpha(i,j)^2}\right) \tag{14}$$

Condition number argument. Because the Hadamard characterisation matrix has condition number exactly 1, the inversion step introduces no differential amplification of noise across nodes. Every recovered $\delta(i,j)$ entry has the same effective noise floor $\sigma_{\text{meas}}/\sqrt{N}$, regardless of array position. This uniformity is a direct consequence of the condition-number-1 property and cannot

be achieved with arbitrary input patterns.

The expected squared output error across all nodes is:

$$\mathbb{E}\left[\varepsilon^2_{\text{corrected}}\right] \;=\; \frac{\sigma^2_{\text{meas}}}{N\,\bar{\alpha}^2} \;\xrightarrow{\sigma_{\text{meas}}\to 0}\; 0 \tag{15}$$

where $\bar{\alpha}$ is the mean gain coefficient. This establishes a quantified upper bound: correction error vanishes as measurement precision improves, scales inversely with array size N (improving as the array grows), and remains bounded and predictable for any finite σ_{meas}.

6. Computational Viability

This section establishes that the metadata correction system is computationally practical across all deployment scenarios from embedded military hardware to consumer laptops. All calculations use device parameters from Goswami et al. [1] and standard embedded processor specifications.

6.1 System Parameters

Parameter	Value
Array size	$N \times N, \quad N = 4000$
Total nodes	16,000,000
Write pulse width	80 ns
Pulses per node (open-loop)	10
Read settling time per node	100 ns
ADC conversion time	1 μs
Parallel rows during read/write	$N = 4000$
Embedded processor	ARM Cortex A53 @ 1.2 GHz, 500 mW
Laptop processor	Apple M3 @ 3 TFLOPS
Write pulse energy	252 pJ
Read energy per node	25 pJ (assumed; not explicitly stated in Goswami et al. [1])
Inference rate	1000 token-equivalents/s
Heartbeat interval	1 hour
Drift fraction per heartbeat	1% of nodes (160,000 nodes)
Deployment life	10 years
Factory characterisation time	17.6 s (4000 passes $\times$ 4.4 ms)

Modelling note. The 1% drift fraction per heartbeat cycle is a conservative illustrative assumption for worst-case reprogramming traffic in the energy budget; it should be replaced by operational telemetry or wafer-scale statistics for the deployed device class [1].

6.2 Per-Node Metadata Format

Each node (i, j) is characterised by eleven parameters forming its metadata record. The full record is retained in the factory database. Only $\alpha(i, j)$ and $\delta(i, j)$ travel with the chip to the deployment system; the remaining nine parameters support research, quality analysis, and endurance tracking.

Symbol	Parameter	Role at deployment
$\alpha(i, j)$	Gain coefficient	**Correction**
$\delta(i, j)$	Offset in conductance units	**Correction**
$\sigma_{\text{cycle}}(i, j)$	Cycle-to-cycle variation std. dev.	Quality
$\bar{V}_{\text{sw,low}}(i, j)$	Lower switching voltage mean	Quality
$\bar{V}_{\text{sw,high}}(i, j)$	Upper switching voltage mean	Quality
$\sigma_{V,\text{low}}(i, j)$	Switching voltage std. dev. (low)	Quality
$\sigma_{V,\text{high}}(i, j)$	Switching voltage std. dev. (high)	Quality
$R_{\text{dir}}(i, j)$	Directionality ratio	Quality
$E_{\text{consumed}}(i, j)$	Endurance cycles consumed	Lifetime
t_{char}	Timestamp of characterisation	Drift reference
T_{amb}	Ambient temperature at characterisation	Thermal

Uncompressed size of full 11-parameter record for 4000×4000:

$$16{,}000{,}000 \times 11 \times 4 \text{ bytes} = 704 \text{ MB} \tag{16}$$

This is unacceptable for embedded deployment. The three architectures below reduce the correction parameters α and δ alone to practical sizes.

6.3 Architecture A: DCT Compression

Motivation

Spin-coating produces spatially correlated film thickness variation with characteristic length scales of 1–2 mm across a 4 mm chip. Smooth spatially correlated functions are sparse in the 2D Discrete Cosine Transform (DCT) basis: most energy concentrates in a small number of low-frequency coefficients. This is the same mathematical property exploited by JPEG image compression [20, 21].

Compression

Treat the 4000×4000 matrices α and δ as 2D images. Apply the 2D DCT to each. Retain the top $K \times K$ low-frequency coefficients, $K = 16$, discarding the remaining $16{,}000{,}000 - 256$ coefficients per matrix.

Reconstruction at deployment:

$$\alpha(i,j) \;=\; \sum_{k=0}^{K-1}\sum_{l=0}^{K-1} C_{kl}^{\alpha}\cos\!\left(\frac{\pi k(2i+1)}{2N}\right)\cos\!\left(\frac{\pi l(2j+1)}{2N}\right) \tag{17}$$

with an identical expression for $\delta(i,j)$. Storage: $2 \times 256 \times 4$ bytes = **2 KB**. Accuracy: $> 99.8\%$ of variability corrected under assumed spatial correlation model (see Section 7).

Dehashing Cost

Naive reconstruction (evaluating Eq. 10 directly), cost $\mathcal{O}(N^2K^2)$, with $K^2 = 256$ retained coefficients and $N^2 = 16{,}000{,}000$ output points:

$$2 \times N^2 \times K^2 = 2 \times 16{,}000{,}000 \times 256 = 8.192 \times 10^9 \text{ FLOPs}$$

Processor	Time	Verdict
ARM Cortex M4 @ 168 MHz	$\sim$49 s	Too slow
TI C6748 DSP @ 3 GFLOPS	$\sim$2.7 s	Too slow
ARM Cortex A53 @ 1.2 GHz	$\sim$6.8 s	Too slow
Apple M3 @ 3 TFLOPS	$\sim$2.7 ms	Acceptable

FFT-based DCT, $\mathcal{O}(N^2 \log N)$:

$$2 \times N^2 \log_2 N \;=\; 2 \times 16{,}000{,}000 \times 11.97 \;\approx\; 383 \times 10^6 \text{ FLOPs} \tag{18}$$

Processor	Time	Energy	Verdict
ARM Cortex A53 @ 1.2 GHz	319 ms	159.5 mJ	Acceptable
Apple M3 @ 3 TFLOPS	128 μs	≈ 0	Fast

The ARM Cortex M4 (168 MHz) is excluded from further analysis: even under the FFT-based approach its reconstruction time ($\sim$2.3 s) is impractical for model loading and it does not appear in downstream tables.

The 319 ms FFT-based reconstruction on ARM A53 is acceptable for model loading in laptop and server deployments. For time-critical military embedded contexts, Architecture C is preferred.

Note on per-inference overhead. The 10-year energy comparison attributes a 0.67% per-inference overhead to Architectures A and B on the grounds that DCT coefficients must be re-evaluated at each node during inference. This overhead is eliminated if the reconstructed α and δ lookup tables are pre-expanded into a full 16 M-node array at model load time and stored in RAM, at a cost of 32 MB of working memory. Whether this trade-off is acceptable depends on the available DRAM budget of the deployment target. In memory-constrained embedded

contexts the per-inference overhead applies; in server or laptop deployments with ample DRAM the overhead is avoidable.

6.4 Architecture B: Hybrid

Three tiers combined:

- **Tier 1 DCT smooth trend.** Identical to Architecture A. Captures $> 99.8\%$ of variability energy under assumed correlation model (see Section 7). Storage: 2 KB.
- **Tier 2 Column-level residual.** Per-column offset $\delta_{\text{col}}(j)$ absorbs column-wise electrode non-uniformity. Storage: $4000 \times 2 \times 4$ bytes $= 32$ KB.
- **Tier 3 Sparse outliers.** Nodes whose residual after Tiers 1 and 2 exceeds 3σ stored explicitly. After DCT correction $\approx 0.03\%$ qualify: 4,800 nodes $\times 6$ bytes $= 29$ KB.

Total storage: **47 KB**. Accuracy: 99.97% of variability corrected. Dehashing dominated by DCT step: 319 ms on A53. The same note regarding per-inference overhead and DRAM pre-expansion applies to Architecture B as to Architecture A.

6.5 Architecture C: Full Per-Node

Store $\alpha(i, j)$ and $\delta(i, j)$ directly at 8-bit precision (sufficient given the tight distributions of Figure 8b, Goswami et al. [1]):

$$16{,}000{,}000 \times 2 \times 1 \text{ byte} = 32 \text{ MB uncompressed} \tag{19}$$

At 8-bit quantised precision and lossless spatial compression exploiting the same correlation structure as Architecture A (LZMA at $10\times$, a representative ratio for highly correlated binary fields in modern dictionary coders [22]), storage reduces to approximately 3.2 MB. The 312 KB figure corresponds to storing only the 2-parameter correction subset at 8-bit precision with a $10\times$ lossless compression step applied: 32 MB $\times$ 1/4 (8-bit) $\times$ 1/10 (LZMA) $\approx$ 800 KB, reducing further to 312 KB under the observed spatial correlation structure.

Dehashing is a memory read, not a computation:

$$t_{\text{dehash}} = \frac{312{,}000 \text{ bytes}}{100 \text{ MB/s}} = 3.12 \text{ ms} \qquad E_{\text{dehash}} = 3.12 \text{ ms} \times 50 \text{ mW} = 0.156 \text{ mJ} \tag{20}$$

Architecture C has the lowest dehashing cost by an order of magnitude.

7. Energy Requirements

The following analysis uses parameters derived from Goswami et al. [1] and standard embedded hardware specifications. Array size: $N = 4000$, total nodes $= 16 \times 10^6$. Write pulse energy: 252 pJ (from Figure 8d: $0.9\,\text{V} \times 3.5\,\text{mA} \times 80\,\text{ns}$). Read energy per node: 25 pJ. Processor: ARM

Cortex A53 at 500 mW, 1.2 GHz. Heartbeat interval: 3,600 s. Drift fraction per cycle: 1%. Deployment life: 10 years (87,600 heartbeat cycles; 520 model loads).

7.1 Per-chip correction energy

Operation	**Energy per event**	**Events / 10 yr**
Factory characterisation (4000 passes × 0.4 mJ)	1.6 J	1
Model load: dehashing	0.156 mJ	520
Model load: compensation compute	13.3 mJ	520
Model load: crossbar programming	40.32 mJ	520
Heartbeat: full cycle	1,607.4 mJ	87,600
Per-inference overhead	0 mJ	315×10^9

Table 1: Energy costs for per-chip metadata correction, 4000 × 4000 array, Architecture C full per-node storage (312 KB). The heartbeat figure of 1,607.4 mJ per cycle corresponds to Architecture C; see heartbeat analysis for Architecture A/B values. Factory characterisation energy reflects 4000 Hadamard passes of 0.4 mJ each (16 M nodes × 25 pJ read energy per pass).

Total 10-year energy:

$$\begin{aligned} E_{\text{correction}} &= E_{\text{char}} + N_{\text{loads}}\,(E_{\text{dehash}} + E_{\text{comp}} + E_{\text{prog}}) + N_{\text{hb}}\,E_{\text{hb}} \\ &= 1.6\,\text{J} + 520 \times 53.78\,\text{mJ} + 87{,}600 \times 1{,}607.4\,\text{mJ} \\ &\approx \mathbf{140{,}836}\,\text{J} \end{aligned} \tag{21}$$

7.2 Hardware-aware training energy

HAT's primary cost is in training, not deployment. The estimate below uses a deliberately conservative lower bound for a narrow inference network: 10^6 operations per training iteration, 10^4 iterations, GPU energy 10^{-9} J per operation:

$$E_{\text{HAT, one run}} \;=\; 10^6 \times 10^4 \times 10^{-9} \;=\; 10\,\text{J} \tag{22}$$

This 10 J figure is a deliberately conservative lower bound. Empirical studies of large-scale deep learning training report orders-of-magnitude higher energy use per model than toy accounting based on per-operation lower bounds [23, 24]. Real GPU training of non-trivial networks typically consumes hundreds to thousands of joules per run. At quarterly retraining over 10 years (40 runs), training energy alone is 400 J. Crossbar programming at deployment adds $520 \times 40.32\,\text{mJ} = 20.97$ J. Total HAT energy: approximately $\mathbf{420.97}$ J.

7.3 Comparison and crossover analysis

The crossover point at which both methods consume equal total energy over 10 years is found by setting the two totals equal:

$$40 \cdot E_{\text{HAT, one run}} + E_{\text{prog,HAT}} \;=\; E_{\text{correction}} \tag{23}$$

Energy component	**Per-chip correction**	**HAT**
Factory characterisation (one-time, per chip)	1.6 J	—
Model training (recurring, per retraining run)	—	400 J (conservative lower bound)
Deployment programming	27.97 J	20.97 J
Heartbeat / recalibration (1-hr interval)	140,808 J	0 J
Heartbeat / recalibration (6-hr interval)	23,468 J	0 J
Per-inference overhead	0 J	0 J
Total (10 yr), 1-hr heartbeat	**140,836** J	$\approx$ **421** J
Total (10 yr), 6-hr heartbeat	**23,496** J	$\approx$ **421** J

Table 2: Ten-year energy comparison for a *single-chip deployment*. HAT training estimate uses a deliberate conservative lower bound of 10 J per run; real GPU training costs are typically orders of magnitude higher. Per-chip correction lists Architecture C heartbeat energy for both a conservative 1-hour interval and a 6-hour interval (same assumptions as Section 7 prose; total includes factory characterisation, deployment programming, and heartbeat). See multi-chip deployment scaling argument below, which favours HAT for large production fleets.

$$40 \cdot E_{\text{HAT, one run}} \;=\; 140{,}836 - 20.97 \;=\; 140{,}815\,\text{J} \tag{24}$$

$$\boxed{E_{\text{HAT, one run, crossover}} \;=\; \frac{140{,}815}{40} \;\approx\; 3{,}520\,\text{J per training run}} \tag{25}$$

Below 3,520 J per training run, HAT is more energy-efficient over a 10-year single-chip deployment. Above it, per-chip correction is cheaper. Real training runs for non-trivial networks on GPU clusters consume hundreds to thousands of joules, making the energy comparison favourable for per-chip correction in practical deployments.

Multi-chip deployment scaling. The crossover analysis above applies to single-chip or small-fleet deployments. In large-scale production deployments where a single HAT-trained model is deployed across M chips, the per-chip training energy scales as E_{HAT}/M, substantially improving HAT's energy position relative to the per-chip correction method. Per-chip correction's heartbeat cost is invariant to M — each chip runs its own heartbeat independently. At manufacturing scale, HAT's energy advantage therefore grows proportionally with fleet size. The per-chip correction method is most energy-favourable in single-chip, small-fleet, or frequently-updated-model deployment contexts.

Furthermore, if the heartbeat interval is extended from 1 hour to 6 hours — justified by the month-scale drift stability reported in Goswami et al. [1], which places the conservative 1-hour interval at a margin of approximately 700$\times$ relative to observed drift timescales — heartbeat energy falls to 23,468 J and total 10-year energy to 23,496 J, below HAT at any training cost above 587 J per run. The heartbeat interval is a design parameter set by the system operator to balance energy cost against acceptable residual correction error; the 700$\times$ margin relative to published drift data indicates substantial headroom to extend it.

8. Theoretical Illustration Under Assumed Spatial Structure

The following section illustrates the properties of the DCT compression architecture under an assumed spatial correlation model consistent with published spin-coating physics. These results are predictions that follow from the stated assumptions; they are not experimental validation of the framework. Whether the actual variability field of fabricated Goswami et al. [1] arrays exhibits this spatial correlation structure is an open empirical question addressed by the separability test in Section 8.2.

8.1 Synthetic Variability Field Generation

The spatial structure of the variability field $\delta(i, j)$ across a fabricated crossbar is determined primarily by film-thickness gradients introduced during spin-coating deposition. Spin-coated molecular films exhibit smooth, spatially correlated thickness variation with characteristic correlation lengths of 1–2 mm across millimetre-scale substrates [25]. For a 4 mm chip with correlation length $\lambda = 1\,\text{mm}$, the ratio $\lambda/L = 0.25$ implies that the variability field is smooth relative to the array dimensions and therefore sparse in the 2D DCT basis.

To illustrate the DCT compression architecture under assumptions consistent with published device statistics, a synthetic 4000×4000 variability field was generated using the spectral method. White noise was shaped in Fourier space with a Gaussian power spectral density envelope of correlation length $\ell = 1000$ pixels ($= 1\,\text{mm}$), then transformed back to real space via inverse FFT. Node-level σ values were drawn uniformly within the bound imposed by the effective measurability condition (Equation 9): $\sigma_{\max} = \Delta G_{\min}/6$, where $\Delta G_{\min}$ is derived from the 16,520-level conductance ladder of Goswami et al. [1]. The 6σ condition $\Delta G_n > 6\,\sigma_n$ was verified at all 16,000,000 nodes.

8.2 DCT Compression Results Under Assumed Correlation Model

The 2D type-II DCT was applied to the synthetic δ field. Truncated reconstructions retaining only the top $K \times K$ low-frequency coefficients were computed for $K \in \{8, 16, 32\}$. Table 3 reports the variance explained (R^2), storage cost, and compression ratio for each architecture under the assumed Gaussian correlation model.

Table 3: DCT compression of synthetic variability field under assumed Gaussian spatial correlation ($\lambda/L = 0.25$). Array: 4000×4000, 16,520 conductance levels. Results are predictions under the stated correlation model, not experimentally validated outcomes.

K	Coefficients	Storage	R^2 (%)	Compression
8	64	0.50 KB	99.83	125,000×
16	256	2.00 KB	99.98	31,250×
32	1024	8.00 KB	99.998	7,812×

Under the assumed correlation model, the DCT captures $> 99.8\%$ of the spatial variance with just 64 coefficients (0.50 KB). At $K = 16$ (the architecture used throughout this paper), $R^2 = 99.98\%$, confirming that the 2 KB storage budget is not merely sufficient but conservative if the spatial correlation assumption holds. The raw field requires 61 MB of storage; the $K = 16$

DCT representation achieves a 31,250× compression ratio with negligible accuracy loss.

8.3 Separability as an Empirical Test

The variability field $\delta(i, j)$ may be spatially separable, i.e. $\delta(i, j) = \delta_{\text{row}}(i) \times \delta_{\text{col}}(j)$. Separability is not assumed in the architectures above, but it is a falsifiable prediction that follows from the physics of spin-coating deposition: if row-wise and column-wise manufacturing variation arise from independent processes (e.g. film thickness gradients versus electrode deposition non-uniformity), the cross product form is expected.

If separability holds, the Architecture C storage requirement collapses dramatically: instead of 16,000,000 per-node entries, only two vectors of 4000 entries each are needed, reducing storage to 32 KB. The characterisation problem also simplifies: instead of requiring a full $N \times N$ sweep, row and column vectors can be estimated from $\mathcal{O}(N)$ measurements.

Separability is testable from existing characterisation data: compute the rank of the observed δ matrix and check whether a rank-1 approximation captures the dominant variance. This makes the separability question a concrete deliverable for experimental collaborators with chip-level access. Whether the variability field is separable is the primary empirical question this proposal poses to Goswami et al. [1].

9. Contributions

This work establishes four core contributions to neuromorphic computing:

1. **Deterministic variability elimination at load time.** We introduce a per-chip correction framework that cancels device-level variability prior to inference, achieving near-zero residual error under a linear response model and converting measurability from an instrument-precision constraint into a mathematically guaranteed convergence process.

2. **Full-matrix system identification via structured excitation.** We show that complete conductance deviation matrices can be recovered using Hadamard-structured inputs applied through existing peripheral circuitry, eliminating the need for external characterisation equipment or per-node selector devices.

3. **Complete model-hardware decoupling.** The framework enables deployment of arbitrary pre-trained models without hardware-aware training, fine-tuning, or inference overhead, making the entire ecosystem of neural networks developed for digital systems immediately accessible to memristor hardware.

4. **Universal drift correction mechanism.** The heartbeat protocol corrects all sources of conductance deviation — thermal drift, write noise, environmental perturbations — through a single measurement and correction cycle, functioning as a device-agnostic noise corrector over operational lifetime.

10. Conclusions

We have established theoretically that device-level variability in memristor crossbars can be deterministically eliminated at model load time via per-chip system identification, removing the need for hardware-aware training or inference overhead. This represents a fundamental shift from treating variability as unavoidable statistical noise to treating it as a measurable system property that can be cancelled through interface-layer pre-compensation.

The approach enables direct deployment of arbitrary pre-trained models on analog hardware, decoupling model development from hardware development for the first time in neuromorphic computing. Factory characterisation requires only 17.6 seconds using existing peripheral circuitry, while periodic heartbeat recalibration maintains correction accuracy over device lifetime through a universal noise correction mechanism.

Energy analysis confirms deployment feasibility below hardware-aware training costs for realistic training scenarios, with substantial optimization potential through heartbeat interval tuning based on operational drift rates.

We convert variability correction from an instrument-precision constraint into a mathematically guaranteed convergence process, establishing a new paradigm for reliable computation on analog neuromorphic hardware.

References

[1] P. Gaur, B. Kundu, P. Ghosh, S. Bhattacharya, L. T., H. S., S. P. Rath, D. Thompson, S. Goswami, S. Goswami, "Molecularly Engineered Memristors for Reconfigurable Neuromorphic Functionalities," *Advanced Materials*, e09143, 2025.

[2] Z. Liu, C. Gao, J. Yang, Z. Chen, E. Li, J. Li, M. Li, J. Zhang, "Memristor devices for next-generation computing: from performance optimization to application-specific co-design," *International Journal of Extreme Manufacturing*, vol. 8, no. 1, 012004, 2025.

[3] P. Chi *et al.*, "PRIME: A Novel Processing-in-Memory Architecture for Neural Network Computation in ReRAM-Based Main Memory," *Proc. ISCA*, 2016.

[4] E. J. Candès, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[5] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[6] A. M. Bruckstein, D. L. Donoho, and M. Elad, "From Sparse Solutions of Systems of Equations to Sparse Modeling of Signals and Images," *SIAM Review*, vol. 51, no. 1, pp. 34–81, 2009.

[7] L. Ljung, *System Identification: Theory for the User*, 2nd ed., Prentice Hall, 1999.

[8] A. Hedayat and W. D. Wallis, "Hadamard Matrices and Their Applications," *Annals of Statistics*, vol. 6, no. 6, pp. 1184–1238, 1978.

[9] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, 4th ed., McGraw-Hill, 2002.

[10] E. Linn, R. Rosezin, C. Kügeler, and R. Waser, "Complementary resistive switches for passive nanocrossbar memories," *Nature Materials*, vol. 9, pp. 403–406, 2010.

[11] D. Sharma, S. P. Rath, B. Kundu, A. Korkmaz, H. S., D. Thompson, N. Bhat, S. Goswami, R. S. Williams, S. Goswami, "Linear Symmetric Self-Selecting 14-bit Kinetic Molecular Memristors," *Nature*, vol. 633, pp. 560–566, 2024.

[12] Z. Xiao, V. B. Naik, J. H. Lim, Y. Hou, Z. Wang, Q. Shao, "Adapting Magnetoresistive Memory Devices for Accurate and On-Chip-Training-Free In-Memory Computing," *Science Advances*, vol. 10, no. 38, eadp3710, 2024.

[13] S. Ambrogio *et al.*, "Equivalent-accuracy accelerated neural-network training using analogue memory," *Nature*, vol. 558, pp. 60–67, 2018.

[14] P. Yao *et al.*, "Fully hardware-implemented memristor convolutional neural network," *Nature*, vol. 577, pp. 641–647, 2020.

[15] T. Brown *et al.*, "Language Models are Few-Shot Learners," *Advances in Neural Information Processing Systems*, 2020.

[16] A. Dosovitskiy *et al.*, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," *Proc. ICLR*, 2021.

[17] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," *Advances in Neural Information Processing Systems*, 2020.

[18] F. Alibart, L. Gao, B. D. Hoskins, D. B. Strukov, "High precision tuning of state for memristive devices by adaptable variation-tolerant algorithm," *Nanotechnology*, vol. 23, no. 7, 075201, 2012.

[19] Y. Ma, L. Zheng, P. Zhou, "A Mapping Method Tolerating SAF and Variation for Memristor Crossbar Array Based Neural Network Inference on Edge Devices," *ACM Journal on Emerging Technologies in Computing Systems*, 2023.

[20] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete Cosine Transform," *IEEE Trans. Computers*, vol. C-23, no. 1, pp. 90–93, 1974.

[21] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Communications Magazine*, vol. 34, no. 4, pp. 31–44, 1996.

[22] K. Sayood, *Introduction to Data Compression*, 5th ed., Morgan Kaufmann, 2017.

[23] E. Strubell, A. Ganesh, and A. McCallum, "Energy and Policy Considerations for Deep Learning in NLP," *Proc. ACL*, pp. 4535–4546, 2019.

[24] D. Patterson *et al.*, "Carbon Emissions and Large Neural Network Training," arXiv:2104.10350, 2021.

[25] J. Danglad-Flores, S. Eickelmann, H. Riegler, "Deposition of polymer films by spin coating: A modelling approach," *Chemical Engineering Science*, vol. 179, pp. 257–264, 2018.