

X-Sim: An Accurate and Scalable Simulator for Memristive Computing-in-Memory Accelerators

Konstantinos Stavrakakis, Bas Smeele, Emmanouil Arapidis, Theofilos Spyrou,
Anteneh Gebregiorgis, Stephan Wong, Georgi Gaydadjiev, Said Hamdioui
Computer Engineering Lab, Delft University of Technology, Delft, The Netherlands

Abstract—Computing-in-Memory (CIM) architectures using memristive crossbar arrays enable energy-efficient AI acceleration. Analog non-idealities, such as IR drop and nonlinearity, impose design constraints that existing simulators cannot capture and thus explore effectively. Current approaches sacrifice either modeling accuracy or simulation speed, preventing systematic design space exploration. In this paper we propose X-Sim, a crossbar simulator that resolves this trade-off through a modular architecture. Our approach decouples device physics from circuit analysis using a fixed-point scheme, avoiding expensive Jacobian computations while preserving device fidelity. X-Sim delivers SPICE-level accuracy ($< 1\%$ error) with up to $200\times$ speedup over physics-based simulators. This enables quick and systematic design space exploration across thousands of configurations, guiding reliable system design. X-Sim will be released as open source.

Index Terms—Memristor, Computing In Memory, Simulation

I. INTRODUCTION

Artificial intelligence (AI) applications create computational demands that challenge von Neumann architectures. Modern AI workloads require massive Vector-Matrix Multiplications (VMMs), yet conventional processors suffer from the memory wall, where data movement between memory and compute units is far more energy-intensive than the operations. [1]–[3]. Computing-in-Memory (CIM) architectures address this bottleneck by co-locating computation and storage. [4]–[6]. Implementations based on RRAM, PCM, and STT-MRAM include crossbar arrays that leverage Ohm’s and Kirchhoff’s laws to perform analog accumulation in a single step, with peripheral digitization completing the Multiply-Accumulate (MAC) [4], [5]. These technologies provide high density, low leakage power, low energy and latency per MAC, making them ideal to power- and area-constrained edge AI systems [4]–[7].

However, realizing CIM’s energy and latency benefits requires managing inherent non-idealities such as IR drop, sneak currents, device variability, and nonlinear I-V characteristics, all of which degrade computational accuracy. Moreover, these non-idealities do not act independently but influence each other through the electrical state of the system. Specifically, each device’s conductance affects the voltage distribution across the array, which in turn affects device behavior, creating a feedback loop. This electrical coupling produces a spatially dependent and input-dependent error landscape that cannot be captured by simple analytical crossbar models [7]. Therefore, accurate

simulation requires a framework that integrates detailed device physics with explicit circuit parasitics, balancing physical accuracy and computational efficiency.

Existing simulators addressing these requirements fall into four categories. First, behavioral simulators such as MNSIM [8] and NeuroSim [9] achieve efficiency through closed-form approximations, but sacrifice accurate device modeling and circuit parasitics. This abstraction yields 5–10% computational errors [10] that invalidate reliability analysis for large crossbar arrays. Second, Machine Learning (ML) focused frameworks such as AIHWKit [11] and MemTorch [12] partially address device variability through stochastic models and approximate circuit interactions via piecewise-linear conductances. Such simplifications miss voltage dependent parasitics, which are essential for accurate large-scale simulation. They also lack transient analysis capabilities, mandated for temporal drift modeling. Third, physics-based simulators such as XBarSim [13] maintain high circuit-level accuracy, but achieve only modest speedup ($10 - 45\times$) over SPICE. This limitation prevents the extensive Monte Carlo analysis needed for Design Space Exploration (DSE). Fourth, hybrid approaches such as CrossSim [10] and RxNN [14] attempt to strike a balance. However, they either rely on parametric device approximations, which limit accuracy, or sacrifice generality through precomputed matrices. This restricts their applicability across diverse operating conditions. Hence, there is a clear need for a fast, accurate, and scalable behavioral simulator of memristor-based CIM accelerators.

We present X-Sim, a crossbar simulator achieving SPICE-level accuracy with practical execution times for design space exploration through a Jacobian-free fixed-point scheme. Our key contributions demonstrate three capabilities:

- **Accuracy:** A physics-based device core implementing the JART-VCM model [15] achieves $< 1\%$ SPICE error and reproduces experimental RRAM measurements;
- **Speed:** A device-circuit coupling algorithm that replaces cubic complexity Jacobian steps with sparse linear system solutions, delivering up to $200\times$ speedup over SPICE;
- **Scalability:** Efficient runtime enabling 10^4 -sample Monte Carlo analysis that identifies critical design boundaries.

This paper is organized as follows: Section II covers RRAM device physics and crossbar modeling challenges. Section III presents our three-module architecture. Section IV details validation against SPICE at device and circuit levels. Section V demonstrates DSE capabilities. Section VI discusses impact and limitations, while Section VII concludes this work.

This work was funded by the EU HORIZON-JU-RIA under grant agreement No. 101112268 for the project NEUROKIT2E and by the Dutch Organization for Scientific Research (NWO) under grant agreement No. KICH1.ST04.22.021 for the project Self-Healing Neuromorphic Systems (SNS).

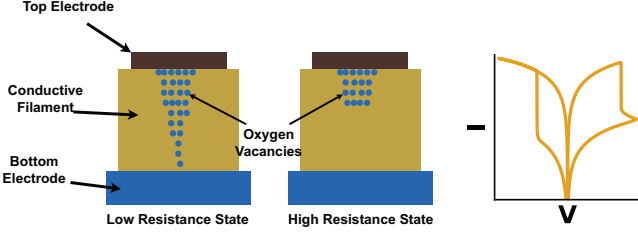


Fig. 1. RRAM switching mechanism showing oxygen vacancy migration-driven conductive filament formation/dissolution between LRS and HRS, with the resulting pinched hysteresis current-voltage (I-V) characteristic

II. BACKGROUND

A. Resistive Random Access Memory (RRAM)

RRAM devices exploit reversible changes in electrical resistance to achieve non-volatile data storage and computation. As illustrated in Figure 1, these devices consist of a metal-oxide layer sandwiched between two electrodes. Data is encoded through the formation and dissolution of conductive filaments within the oxide matrix. When a positive voltage bias is applied, oxygen vacancies migrate toward the cathode, forming a low-resistance conductive path that represents logic ‘1’ state, known as Low Resistance State (LRS). In contrast, a negative bias causes oxygen to return to the filament structure, disrupting the conductive path and resulting in a logic ‘0’ state, or High Resistance State (HRS). This switching mechanism is responsible for the pinched hysteresis loop in the current-voltage relationship. That essentially indicates that device resistance is determined by both the applied voltage and the switching history. The resistance ratio between HRS and LRS, which typically spans one to three orders of magnitude [16], [17] enables reliable data storage. Additionally, the continuous switching process allows for operations with intermediate resistance states [16].

B. Crossbar Vector-Matrix Multiplication

RRAM crossbars execute vector matrix multiplication by encoding each matrix element as a device conductance G_{ij} at the intersection of wordline i and bitline j . Applying the input vector on wordlines as voltages V_i (or via bit-sliced/pulse-width codes) produces device currents $I_{ij} = G_{ij}V_i$. These currents sum along each bitline by Kirchhoff’s law to form $I_{out,j} = \sum_i G_{ij}V_i$, which is then converted to a voltage with a transimpedance or resistive load and digitized by an analog-to-digital converter (ADC) [18], [19]. Figure 2 depicts the conductance mapping, wordline drive and bitline accumulation that together implement the dot product.

C. Crossbar Non-Idealities

Performing VMMs in practical crossbar-based CIM implementations deviates from the ideal VMM abstraction due to circuit and device effects (Figure 2). Interconnect resistances introduce position dependent IR drop. This compresses dynamic range and distorts the effective weights [20]. In passive arrays, parasitic sneak paths also conduct through unselected devices, an effect that grows with array size and the fraction of low-resistance cells [21]. This issue is commonly mitigated by including a select transistor in a one-transistor-one-resistor

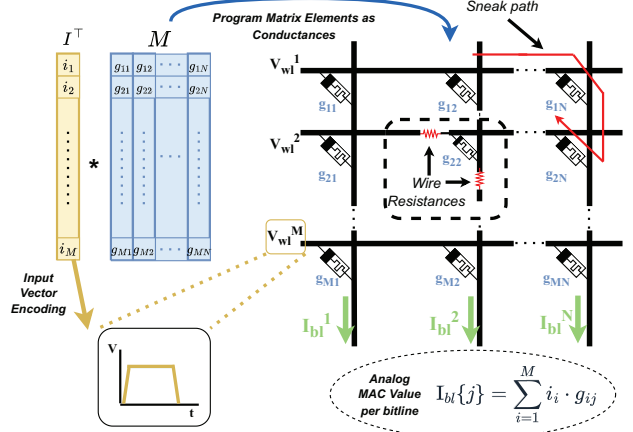


Fig. 2. Crossbar analog vector matrix multiplication in passive (1R) configuration showing conductance-encoded weights and parasitic effects including sneak path currents and wire resistances.

(1T1R) cell. At the device level, nonlinear I-V behavior, cycle-to-cycle programming variability, device-to-device variation and time-dependent drift perturb conductances over space and time [22], [23]. Together these mechanisms yield spatially structured, input-dependent errors that degrade VMM accuracy and bound scalable array dimensions.

D. Simulation Challenges

To capture these effects it is necessary to treat the array as a coupled nonlinear resistive network [24]. Each device current depends on its local voltage. Those voltages are set by the global network of device conductances and line resistances. The result is a circular dependency that rules out closed-form solutions [11]. Therefore, the simulator must preserve calibrated device physics and explicit parasitics while scaling to realistic array sizes.

III. PROPOSED SIMULATION APPROACH

A. Simulator Architecture and Design Rationale

We resolve the device-circuit circular dependency with a fixed-point scheme that alternates device evaluation and linear voltage solution steps. We assume a 1T1R structure for its consistency with common practice in fabricated CIM macros [25]–[27], and we treat the access transistor as an ideal switch. The proposed three-module architecture:

- 1) **Physics-Based Memristor Core:** Computes accurate device current-voltage characteristics by implementing complete JART-VCM [15] device physics, handling internal state dynamics and nonlinear switching behavior independently of circuit-level effects.
- 2) **Linear Crossbar Solver:** Efficiently determines circuit-level voltage distributions for a given set of device conductances using Chen’s matrix formulation [28], exploiting the favorable numerical properties of crossbar networks.
- 3) **Nonlinear Coupling Engine:** Resolves the device-circuit circular dependency through fixed-point iteration, enabling integration of device physics with efficient circuit analysis.

The modules interact through an iterative protocol, as illustrated in Figure 3: Physics-based Memristor core computes

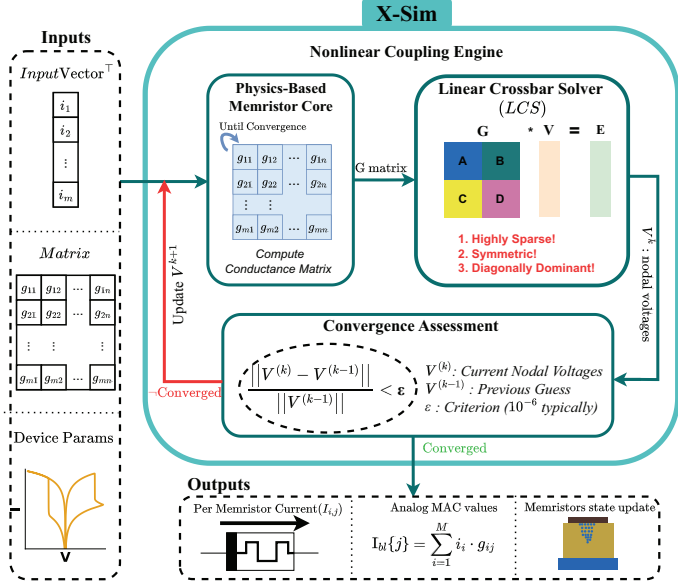


Fig. 3. X-Sim’s three-module architecture enabling fixed-point device-circuit coupling that exploits conductance matrix structure to avoid $O((MN)^3)$ Jacobian computations.

$I(V)$ and state; Linear Crossbar Solver computes V on the crossbar; Nonlinear Coupling Engine updates conductances and checks convergence.

B. Physics-Based Memristor Core

Model Selection and Rationale. Accurate crossbar simulation demands a physics-based device model that captures switching dynamics for circuit-level analysis. While behavioral models like VTEAM [29] offer simplicity, they omit physical mechanisms needed for accuracy across operating conditions. Physics-based alternatives include Stanford-PKU [30] and JART-VCM [15] models. We select JART-VCM for its well-validated implementation of vacancy drift, field-dependent mobility, and Schottky contact physics. Its continuous differential equations provide stable convergence properties, which are essential for our iterative coupling approach.

JART-VCM Model Implementation and Challenges. The model represents the device through four series elements: Schottky contact, disc resistance, plug resistance, and series resistance, as illustrated in Figure 4a. The disc resistance governs primary conduction through $R_{\text{disc}} = l_{\text{disc}} / (z v_0 A \mu_n N_{\text{disc}})$, where N_{disc} evolves according to ion current flow. An implicit Schottky voltage relation creates a circular dependency between current and voltage partitioning. We write Eq. 1 and Eq 2, solve for V_S iteratively, and then compute $I(V_{\text{applied}})$. Accurate current calculation is necessary, since conductance updates set the circuit voltages in subsequent coupling steps.

$$V_S = V_{\text{applied}} - V_{\text{disc}} - V_{\text{plug}} - V_{\text{series}} \quad (1)$$

$$V_{\text{applied}} = V_S + I \cdot R_{\text{total}} \quad (2)$$

Algorithmic Implementation Strategies. Standard fixed point updates can diverge under strong nonlinearity. We use a damped residual update with Eq. 3 and adapt the damping magnitude during iteration.

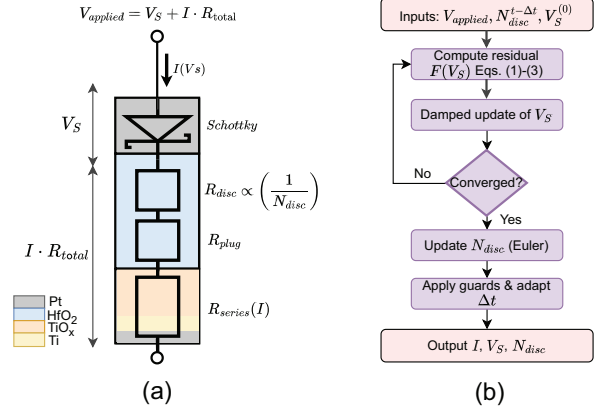


Fig. 4. (a) JART-VCM model [15] architecture; (b) Adaptive damped iteration algorithm for resolving the implicit Schottky voltage dependency.

$$F(V_S) = V_{\text{applied}} - V_S - I(V_S) \cdot R_{\text{total}} \quad (3)$$

This stabilizes convergence under typical bias conditions with low overhead. We use adaptive time steps for state integration and reduce Δt when $|\Delta N_{\text{disc}} / N_{\text{disc}}| > \epsilon$ to capture fast transients. We bound physical parameters and state variables and reset the step when constraints are violated to prevent overflow from exponential terms. Figure 4b depicts the residual-based update and guards used to maintain stability.

C. Device-Circuit Integration

Complete crossbar simulation requires accurate determination of two interdependent quantities: the voltage across each memristive device, and the current contribution from each device to the network. The physics-based memristor model requires local device voltages as inputs, yet these voltages are determined by current flow through the crossbar’s resistive network, creating a circular dependency.

Linear Crossbar Solver. We employ Chen’s conductance-matrix method [28] to write the crossbar nodal equations as $GV = E$. For an $M \times N$ array, with M wordlines and N bitlines, $G \in \mathbb{R}^{2MN \times 2MN}$ comes from applying Kirchoff’s current law at every internal node, with one wordline node and one bitline node per cell. Adjacent nodes are connected by wordline and bitline resistances, and the device conductance links the corresponding wordline and bitline node at (i, j) . The vector V stacks all nodal voltages, and E encodes the boundary conditions at the driven wordline and bitline. This yields a sparse, symmetric, and diagonally dominant G enabling stable and efficient linear system solutions.

Nonlinear Coupling Engine. Traditional approaches solve the complete nonlinear system simultaneously, requiring expensive Jacobian matrix computation. To avoid this, we employ a fixed-point iteration illustrated in Figure 3 that decouples the evaluation of the device physics from the circuit voltage solution. More specifically, the algorithm alternates between the following computational phases:

- 1) **Device Resistance Update:** Given the current voltage estimate $V^{(k)}$, each memristive device computes its local conductance $G_i^{(k)} = I_i(V_i^{(k)}) / V_i^{(k)}$ using the JART-VCM model implementation (see Section III-B);

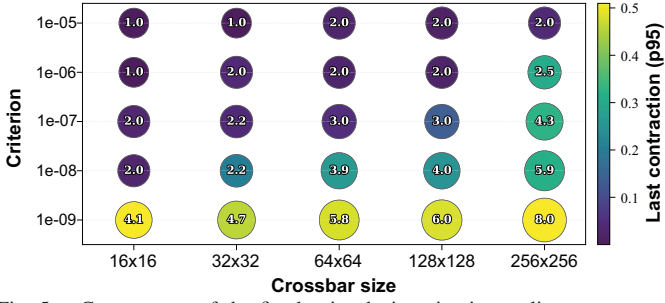


Fig. 5. Convergence of the fixed-point device–circuit coupling across array sizes and stopping criteria; Bubble area and numeric label both encode the mean iteration count; color encodes the 95th percentile of the last-iteration contraction factor.

- 2) *Linear circuit solution*: These conductance values populate $\mathbf{G}^{(k+1)}$; the sparse conjugate-gradient solver then obtains $\mathbf{V}^{(k+1)}$ from $\mathbf{G}^{(k+1)}\mathbf{V}^{(k+1)} = \mathbf{E}$;
- 3) *Convergence check*: Iteration stops when $\|\mathbf{V}^{(k+1)} - \mathbf{V}^{(k)}\|_\infty \leq \varepsilon$ with $\varepsilon = 10^{-6}$ V; otherwise the loop repeats.

Complexity Analysis. For an $M \times N$ crossbar, the nodal matrix has $\text{nnz} = \Theta(MN)$ nonzeros. Each outer fixed-point sweep performs: i) device-state updates across all cells in $O(MN)$ and ii) a Conjugate Gradient (CG)-based solution of the sparse nodal system. Let k be the number of outer sweeps to convergence, and let t be the CG iterations (per sweep) to reach tolerance ε . Each CG iteration costs $O(\text{nnz}) = \Theta(MN)$. Thus, the per-input runtime is $T = \Theta(kMNt)$ and the memory footprint is $\text{Mem} = \Theta(\text{nnz}) = \Theta(MN)$.

Convergence Analysis. We assess the convergence of fixed-point device-circuit coupling and bound required outer sweeps across operating conditions. For each array size and criterion ε , we compute the reduction factor between successive node-voltage iterates at termination and average outer sweeps to obtain k . Figure 5 shows mean iterations and last-sweep contraction rise with array size and tighter criteria, yet remain modest. Across all operating conditions, the number of outer sweeps falls between 1 and 8, ($1 \leq k \leq 8$). Additionally, the fact that the maximal 95th percentile of last-sweep contraction is well below 1, indicates significant per-iteration shrinkage.

IV. VALIDATION

All experiments ran on a RHEL system with an AMD EPYC 7543 32-core Processor and 247 GB of physical memory. To demonstrate both accuracy and practicality, the validation proceeds in three steps. We begin at the device level, showing that X-Sim’s JART-VCM implementation reproduces custom-fitted SPICE models and measured RRAM I-V curves. We then scale to complete crossbars (32×32 , 64×64 , 128×128) where the simulator matches SPICE within sub-percent error while delivering significant speedup. Finally, we compare the execution time and accuracy against MemTorch, the primary framework that combines memristor device modeling with full crossbar circuit simulation. Collectively, these results verify the correctness of both kernels and establish X-Sim as a fast, SPICE-accurate platform for CIM design.

A. Single Device Physics Validation

We validate our JART-VCM model implementation against:

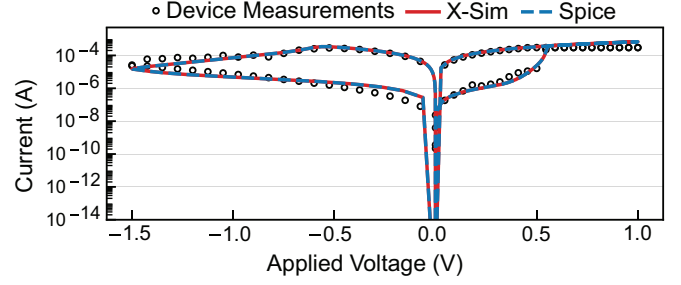
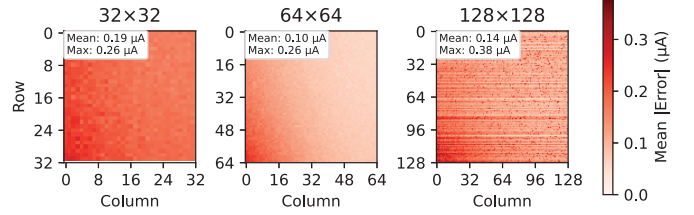
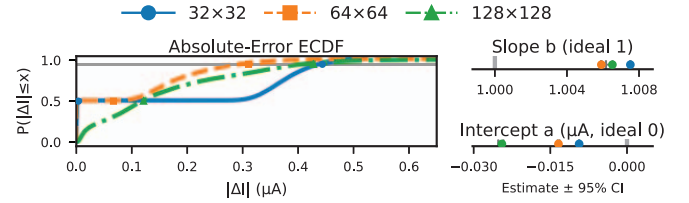


Fig. 6. Device-level validation demonstrating exact correspondence between X-Sim, SPICE simulation, and experimental RRAM measurements across the complete hysteresis loop.



(a) Mean error heatmaps across all positions (i, j) in a crossbar for different crossbar dimensions (32×32 , 64×64 , 128×128).



(b) Cumulative distribution of absolute errors, and linear correlation analysis with slope and intercept values.

Fig. 7. Circuit level behavioral validation.

- 1) Direct I-V measurements from an oxide-based RRAM cell, providing ground-truth device response.
- 2) A SPICE simulation of the same fitted JART-VCM netlist, using identical parameters and ± 1.5 V, 100mV step triangular sweep.

We disable variability effects to isolate numerical solver accuracy. Figure 6 shows that X-Sim and SPICE traces overlap across the entire hysteresis loop and both coincide with the experimental curve, reproducing threshold voltages, loop shape, and resistance levels. This correspondence demonstrates that X-Sim delivers SPICE-level precision while reproducing calibrated device’s behavior.

B. Circuit-Level Validation

To establish crossbar-level fidelity, we validate X-Sim against SPICE across 32×32 , 64×64 and 128×128 configurations under standard read pulses that mirror practical VMMs. These array sizes represent realistic CIM implementations while enabling comprehensive SPICE validation. Larger configurations exhibit similar error patterns but require prohibitive simulation times. Each crossbar validation encompasses randomly initialized device conductances spanning the full resistance range with diverse input vector patterns. Device conductances are randomly sampled across the full HRS-to-LRS range (10 k Ω to 10 M Ω) to capture statistical variations encountered in deployed

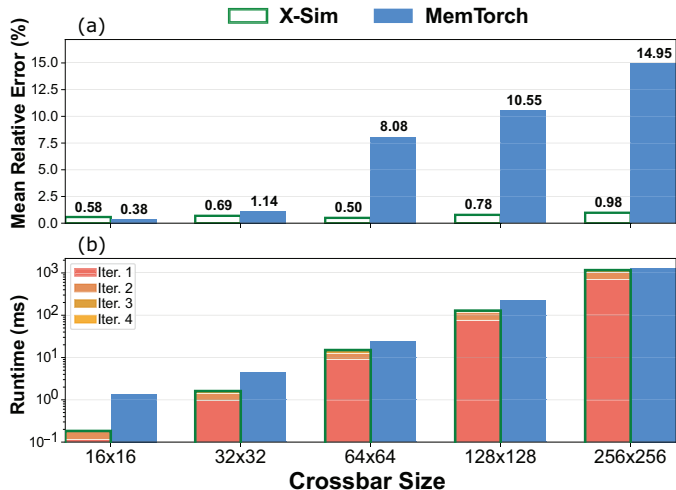


Fig. 8. Accuracy relative to SPICE (a) and runtime with per-iteration breakdown (b) for X-Sim and MemTorch across crossbar sizes.

systems. Input patterns include structured vectors representative of neural network activations and random distributions to identify potential computational edge cases.

Behavioral Accuracy. Figure 7a depicts the spatial error uniformity with mean absolute errors of $0.19 \mu\text{A}$, $0.10 \mu\text{A}$, and $0.14 \mu\text{A}$ for 32×32 , 64×64 , and 128×128 arrays. The uniform spatial distribution confirms there were no systematic modeling artifacts or edge effects. Mean relative errors of 0.44%, 0.30% and 0.69% for 32×32 , 64×64 and 128×128 arrays, respectively, demonstrate consistent accuracy across array sizes. Correlation analysis presented in Figure 7b shows $R^2 > 0.999$ between X-Sim and SPICE with near-unity slope and minimal intercept across all array sizes, confirming the absence of scaling bias and accurate zero-current prediction. Cumulative distribution (Figure 7b) confirms that 95% of errors remain below $0.35 \mu\text{A}$ establishing bounded error characteristics.

Runtime Comparison. We measure the runtime for simulating the full read pulse used in the behavioral accuracy tests. As shown in Table I, X-Sim significantly reduces SPICE runtimes, yielding mean speedup of 139 \times , 174 \times , 208 \times for the 32×32 , 64×64 , and 128×128 arrays, respectively. Speedup increases with array size, as fixed-point coupling with sparse linear solves scales more favorably than general nonlinear steps in SPICE.

C. Scalability Analysis

We compare X-Sim against MemTorch, a widely adopted memristive simulation platform that achieves high computational speed through linear conductance approximations. This isolates trade-offs between physics-based nonlinear and simplified linear approaches across realistic array dimensions.

We executed single read operations across array sizes from 16×16 to 256×256 , applying identical input vectors and conductance matrices to both simulators. Each read operation requires only one linear system solution in MemTorch and one nonlinear iteration in X-Sim. We validated both outputs against SPICE reference simulations using 5000 random samples per array size, measuring the mean relative error of accumulated bitline currents that represent VMM computational results.

TABLE I
FULL-PULSE SIMULATION: SPICE VS. X-SIM RUNTIME

Crossbar Size	SPICE (s)	X-Sim (s)	Speedup
32×32	79.38	0.57	139.26x
64×64	660	3.79	174.14x
128×128	3858.00	18.52	208.31x

TABLE II
DESIGN SPACE EXPLORATION PARAMETERS

Parameter	Values
Array Size	16×16 , 32×32 , 64×64 , 128×128
Wire Resistance	1Ω , 5Ω , 10Ω
Conductance Range	$0.2 - 600 \mu\text{S}$, $16 - 600 \mu\text{S}$, $16 - 40 \mu\text{S}$
Input Sparsity	25%, 50%, 75%

Figure 8b demonstrates that X-Sim matches or exceeds MemTorch’s runtime performance despite requiring iterative nonlinear convergence. The first iteration dominates computational cost, with subsequent iterations adding minimal overhead through selective matrix updates. Accuracy analysis (Figure 8a) reveals mean relative errors ranging from 0.38% for 16×16 arrays to 14.95% for 256×256 configurations in MemTorch, while X-Sim maintains sub-percent accuracy across all array sizes. MemTorch’s linear approximations introduce deviations that increase exponentially with array size.

Despite the narrowing runtime gap, only X-Sim maintains accuracy below one percent across all tested sizes. This positions X-Sim as a fast and reliable tool for design space exploration.

V. PARAMETER SENSITIVITY ANALYSIS

We demonstrate X-Sim’s capabilities through systematic parameter sweeps across key crossbar design variables affecting VMM computational accuracy. This analysis quantifies design trade-offs and reveals critical operating boundaries that guide practical CIM system design.

Methodology and Metrics. We sweep key design parameters as detailed in Table II, performing Monte Carlo simulation with 10,000 random $(1 \times 128) * (128 \times 128)$ VMM operations per parameter combination. VMM tiling enables computation evaluation for the smaller crossbar configurations. We quantify VMM accuracy using Mean Absolute Error (MAE) between ideal mathematical results and analog crossbar outputs. MAE captures the cumulative impact of IR drop and device nonlinearities on computational correctness. Based on observed error magnitudes we establish three accuracy zones:

- 1) *High Accuracy* ($MAE < 1$): Analog errors remain below unit magnitude relative to ideal computations;
- 2) *Medium Accuracy* ($1 \leq MAE < 10$): Single-order-of-magnitude deviations acceptable for error-tolerant applications;
- 3) *Low Accuracy* ($MAE > 10$): Computational errors that likely compromise system utility.

These zones enable systematic parameter comparison across the design space. VMM accuracy serves as a practical proxy for crossbar computational reliability. However, specific tolerance requirements depend on target application constraints.

Key Parameter Interactions. Our analysis reveals three critical design parameters that significantly affect VMM accuracy:

Wire Resistance and Array Size: Figure 9a reveals a sharp accuracy transition that depends critically on both parameters.

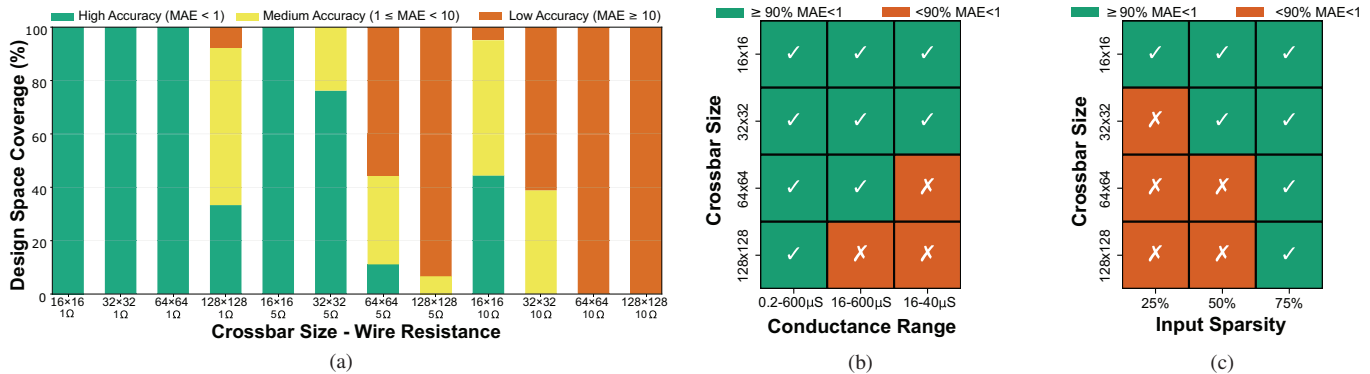


Fig. 9. Parameter Sensitivity Analysis: (a) Wire resistance impact on VMM accuracy; (b) Operating zones classified using $MAE < 1$, where 90% $MAE < 1$ means 90% of VMM samples achieved $MAE < 1$; (c) Accuracy mapping vs input sparsity with $MAE < 1$ metric.

Small arrays (16×16 , 32×32) demonstrate remarkable robustness, maintaining predominantly high accuracy across all wire resistance conditions. However, the system exhibits a critical performance cliff at 64×64 , where wire resistance transforms from a minor perturbation to a dominant error source. Large arrays (128×128) show fundamental limitations, achieving high accuracy in less than 40% of samples even under low wire resistance conditions, indicating that scaling challenges extend beyond simple wire resistance optimization.

Device Conductance Range: This emerges as a critical determinant of VMM accuracy (Figure 9b). Higher LRS conductance yields lower MAE across samples and maintains it below one across more array sizes. The HRS-to-LRS ratio strongly affects accuracy, with 0.2 to 600 μS consistently outperforming 16 to 40 μS across sizes. These results indicate that device targets must meet conductance and dynamic range criteria and that practical usability is application- and workload-dependent.

Input Sparsity: Figure 9c reveals sparsity’s dramatic impact on computational accuracy. Accuracy improves substantially from 25% to 75% input sparsity across larger crossbars. Dense inputs (25% sparsity) produce low accuracy even for 32×32 arrays. On the contrary, sparse inputs (75% sparsity) restore high accuracy for 128×128 configurations. This demonstrates that structured sparsity becomes essential for maintaining computational accuracy in large crossbar arrays, extending beyond its traditional role as an efficiency optimization.

Design Implications. These results demonstrate X-Sim’s capability for rapid design space exploration, revealing how parameter interactions affect VMM computational accuracy. With its ability to explore design options systematically and at practical speed, X-Sim provides reliable guidance for early architectural design choices.

VI. DISCUSSION

Comparison to Prior Work. Table III highlights a persistent accuracy–performance trade-off; behavioral tools and ML frameworks [8], [9], [11], [12] prioritize runtime but sacrifice physics fidelity, circuit-accurate simulators [13] achieve sub-1% SPICE error at prohibitive computational cost, and hybrid approaches [10], [14] compromise generality for speed. X-Sim uniquely combines physics-based device modeling, full parasitics, nonlinearity, device-level transient modeling, and sub-1% SPICE error while maintaining high runtime efficiency.

TABLE III

COMPARISON BETWEEN THE STATE-OF-THE-ART CIM SIMULATORS

	[8]	[9]	[10]	[11]	[12]	[13]	[14]	X-Sim
Physics Device Model	×	×	○	○	○	×	×	✓
Nonlinearity	×	×	✓	○	×	×	×	✓
Circuit Parasitic Modeling	×	×	✓	○	○	✓	○	✓
Temporal device dynamics	×	×	×	×	×	×	×	✓
Sub-1% SPICE Error	×	×	×	×	×	✓	×	✓
Runtime Efficiency	✓	✓	✓	○	○	×	✓	✓

✓ = Full Support, ○ = Partial, × = No Support

Practical Impact. X-Sim’s combination of speed and fidelity enables systematic exploration across the entire hardware–software design stack. Device researchers can evaluate novel materials against realistic workloads, quantifying how material properties translate to application-level performance. System architects can optimize array configurations for specific computational requirements rather than relying on simplified models. Algorithm designers can incorporate actual circuit constraints into their optimization strategies, moving beyond idealized assumptions that ignore parasitic effects and device nonlinearities.

Current limitations. This work targets RRAM and implements the JART-VCM model; the modular device core can extend to FeFET and PCM through parameterization and validation without new solvers. The present scope is crossbar level analysis for accuracy and speed in design exploration; integration with synthesis and mapping flows is a direct next step. These extensions expand coverage while preserving the fixed point coupling and the sparse linear solver.

VII. CONCLUSION

In this work, we present X-Sim, a solution to the speed–accuracy trade-off in crossbar simulation. It uses a modular architecture that decouples device physics from circuit analysis through fixed-point iteration. The result is SPICE-level accuracy ($< 1\%$ error) with fast runtime (up to $200\times$ speedup over SPICE) suitable for large-scale design space exploration. X-Sim enables systematic behavioral analysis across realistic parameter spaces that were previously too time-consuming. It serves as a practical, high-fidelity tool for designing reliable Computing-in-Memory systems.

REFERENCES

- [1] W. A. Wulf and S. A. McKee, "Hitting the memory wall: implications of the obvious," *SIGARCH Comput. Archit. News*, vol. 23, no. 1, p. 20–24, Mar. 1995.
- [2] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, 2014, pp. 10–14.
- [3] V. Sze, Y.-H. Chen, T.-J. Yang, and J. Emer, "Efficient processing of deep neural networks: A tutorial and survey," 2017. [Online]. Available: <https://arxiv.org/abs/1703.09039>
- [4] D. Ielmini and H.-S. P. Wong, "In-memory computing with resistive switching devices," *Nature electronics*, vol. 1, no. 6, pp. 333–343, 2018.
- [5] Q. Xia and J. J. Yang, "Memristive crossbar arrays for brain-inspired computing," *Nature materials*, vol. 18, no. 4, pp. 309–323, 2019.
- [6] B. Rajendran and F. Alibart, "Neuromorphic computing based on emerging memory technologies," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 6, no. 2, pp. 198–211, 2016.
- [7] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, "Memory devices and applications for in-memory computing," *Nature nanotechnology*, vol. 15, no. 7, pp. 529–544, 2020.
- [8] L. Xia, B. Li, T. Tang, P. Gu, P.-Y. Chen, S. Yu, Y. Cao, Y. Wang, Y. Xie, and H. Yang, "MNSim: Simulation platform for memristor-based neuromorphic computing system," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 5, pp. 1009–1022, 2018.
- [9] P.-Y. Chen, X. Peng, and S. Yu, "NeuroSim: A circuit-level macro model for benchmarking neuro-inspired architectures in online learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 12, pp. 3067–3080, 2018.
- [10] B. Feinberg, T. P. Xiao, C. J. Brinker, C. H. Bennett, M. J. Marinella, and S. Agarwal, "CrossSim: accuracy simulation of analog in-memory computing," *GitHub*, v3.1, 2025. [Online]. Available: <https://github.com/sandialabs/cross-sim>
- [11] M. J. Rasch, C. Mackin, M. Le Gallo, A. Chen, A. Fasoli, F. Odermatt, N. Li, S. Nandakumar, P. Narayanan, H. Tsai *et al.*, "Hardware-aware training for large-scale and diverse deep learning inference workloads using in-memory computing-based accelerators," *Nature communications*, vol. 14, no. 1, p. 5282, 2023.
- [12] C. Lammie, W. Xiang, B. Linares-Barranco, and M. Rahimi Azghadi, "MemTorch: An open-source simulation framework for memristive deep learning systems," *Neurocomputing*, vol. 485, p. 124–133, May 2022. [Online]. Available: <http://dx.doi.org/10.1016/j.neucom.2022.02.043>
- [13] A. Kolinko, M. H. Amin, R. Zand, and J. Bakos, "XbarSim: A decomposition-based memristive crossbar simulator," 2024. [Online]. Available: <https://arxiv.org/abs/2410.19993>
- [14] S. Jain, A. Sengupta, K. Roy, and A. Raghunathan, "RxNN: A framework for evaluating deep neural networks on resistive crossbars," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 2, pp. 326–338, 2021.
- [15] C. Bengel, A. Siemon, F. Cüppers, S. Hoffmann-Eifert, A. Hardtdegen, M. von Witzleben, L. Hellmich, R. Waser, and S. Menzel, "Variability-aware modeling of filamentary oxide-based bipolar resistive switching cells using spice level compact models," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 12, pp. 4618–4630, 2020.
- [16] R. Islam, S. Qin, S. Deshmukh, Z. Yu, C. Köroğlu, A. I. Khan, K. Schauble, K. C. Saraswat, E. Pop, and H.-S. Wong, "Improved gradual resistive switching range and 1000× on/off ratio in hfox rram achieved with a ge2sb2te5 thermal barrier," *Applied Physics Letters*, vol. 121, no. 8, 2022.
- [17] M. Lanza, R. Waser, D. Ielmini, J. J. Yang, L. Goux, J. Suñe, A. J. Kenyon, A. Mehonic, S. Spiga, V. Rana *et al.*, "Standards for the characterization of endurance in resistive switching devices," *ACS nano*, vol. 15, no. 11, pp. 17214–17231, 2021.
- [18] S. Lee, M. E. Fouda, J. Lee, A. Eltawil, and F. Kurdahi, "Accurate prediction of rram crossbar performance under iv nonlinearity and ir drop," in *2022 IEEE 40th International Conference on Computer Design (ICCD)*. IEEE, 2022, pp. 9–16.
- [19] J. Chen, J. Li, Y. Li, and X. Miao, "Multiply accumulate operations in memristor crossbar arrays for analog computing," *Journal of Semiconductors*, vol. 42, no. 1, p. 013104, 2021.
- [20] H. Zhang, N. Xiao, F. Liu, and Z. Chen, "Leader: Accelerating rram-based main memory by leveraging access latency discrepancy in crossbar arrays," in *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2016, pp. 756–761.
- [21] L. Shi, G. Zheng, B. Tian, B. Dkhil, and C. Duan, "Research progress on solutions to the sneak path issue in memristor crossbar arrays," *Nanoscale Advances*, vol. 2, no. 5, pp. 1811–1827, 2020.
- [22] A. P. James and L. O. Chua, "Variability-aware memristive crossbars—a tutorial," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 6, pp. 2570–2574, 2022.
- [23] M. Le Gallo, C. Lammie, J. Büchel, F. Carta, O. Fagbohunbe, C. Mackin, H. Tsai, V. Narayanan, A. Sebastian, K. El Maghraoui *et al.*, "Using the ibm analog in-memory hardware acceleration kit for neural network training and inference," *APL Machine Learning*, vol. 1, no. 4, 2023.
- [24] F. Aguirre, A. Sebastian, M. Le Gallo, W. Song, T. Wang, J. J. Yang, W. Lu, M.-F. Chang, D. Ielmini, Y. Yang *et al.*, "Hardware implementation of memristor-based artificial neural networks," *Nature communications*, vol. 15, no. 1, p. 1974, 2024.
- [25] W. Wan, R. Kubendran, C. Schaefer, S. B. Eryilmaz, W. Zhang, D. Wu, S. Deiss, P. Raina, H. Qian, B. Gao *et al.*, "A compute-in-memory chip based on resistive random-access memory," *Nature*, vol. 608, no. 7923, pp. 504–512, 2022.
- [26] K. Humood, P. Foster, S. Wang, A. Serb, and T. Prodromakis, "Spika: 200-tops/w rram-based neural network accelerator chip," in *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2025, pp. 1–5.
- [27] D. F. Falcone, V. Clerico, W. Choi, T. Stecconi, F. Horst, L. Bégon-Lours, M. Galetta, A. La Porta, N. Garg, F. Alibart *et al.*, "All-in-one analog ai hardware: On-chip training and inference with conductive-metal-oxide/hfox rram devices," *Advanced Functional Materials*, p. 2504688, 2025.
- [28] A. Chen, "A comprehensive crossbar array model with solutions for line resistance and nonlinear device characteristics," *IEEE Transactions on Electron Devices*, vol. 60, no. 4, pp. 1318–1326, 2013.
- [29] S. Kvatinisky, M. Ramadan, E. G. Friedman, and A. Kolodny, "VTEAM: A general model for voltage-controlled memristors," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 8, pp. 786–790, 2015.
- [30] Z. Jiang, Y. Wu, S. Yu, L. Yang, K. Song, Z. Karim, and H.-S. P. Wong, "A compact model for metal–oxide resistive random access memory with experiment verification," *IEEE Transactions on Electron Devices*, vol. 63, no. 5, pp. 1884–1892, 2016.