frontiers | Frontiers in Neuroscience

TYPE Original Research
PUBLISHED 18 August 2022
DOI 10.3389/fnins.2022.941753

Check for updates

OPEN ACCESS

EDITED BY
Abhronil Sengupta,
The Pennsylvania State University (PSU), United States

REVIEWED BY
Jie Jiang,
Central South University, China
Yuhan Shi,
University of California, San Diego,
United States

*CORRESPONDENCE
Daniel Bedau
daniel.bedau@wdc.com

SPECIALTY SECTION
This article was submitted to Neuromorphic Engineering, a section of the journal Frontiers in Neuroscience

RECEIVED 11 May 2022
ACCEPTED 04 August 2022
PUBLISHED 18 August 2022

CITATION
Hennen T, Elias A, Nodin J-F, Molas G, Waser R, Wouters DJ and Bedau D (2022) A high throughput generative vector autoregression model for stochastic synapses. *Front. Neurosci.* 16:941753. doi: 10.3389/fnins.2022.941753

A high throughput generative vector autoregression model for stochastic synapses

Tyler Hennen[1], Alexander Elias[2], Jean-François Nodin[3], Gabriel Molas[3,4], Rainer Waser[1], Dirk J. Wouters[1] and Daniel Bedau[2]*

[1]Institut für Werkstoffe der Elektrotechnik 2 (IWE II), RWTH Aachen University, Aachen, Germany, [2]Western Digital San Jose Research Center, San Jose, CA, United States, [3]CEA, LETI, Grenoble, France, [4]Weebit Nano Ltd., Grenoble, France

By imitating the synaptic connectivity and plasticity of the brain, emerging electronic nanodevices offer new opportunities as the building blocks of neuromorphic systems. One challenge for large-scale simulations of computational architectures based on emerging devices is to accurately capture device response, hysteresis, noise, and the covariance structure in the temporal domain as well as between the different device parameters. We address this challenge with a high throughput generative model for synaptic arrays that is based on a recently available type of electrical measurement data for resistive memory cells. We map this real-world data onto a vector autoregressive stochastic process to accurately reproduce the device parameters and their cross-correlation structure. While closely matching the measured data, our model is still very fast; we provide parallelized implementations for both CPUs and GPUs and demonstrate array sizes above one billion cells and throughputs exceeding one hundred million weight updates per second, above the pixel rate of a 30 frames/s 4K video stream.

KEYWORDS

neuromorphic computing, machine learning, time series, emerging technologies, stochastic model, ReRAM, neural networks, nanotechnology

Introduction

Recent trends in computing hardware have placed increasing emphasis on neuromorphic architectures implementing machine learning (ML) algorithms directly in hardware. Such bio-inspired approaches, through in-memory computation and massive parallelism, excel in new classes of computational problems and offer promising advantages with respect to power consumption and error resiliency. While CMOS-based neuromorphic computing (NC) implementations have made substantial progress recently, new materials and physical mechanisms may ultimately provide better opportunities for energy efficiency and scaling (Burr et al., 2017; Milo et al., 2020; Sangwan and Hersam, 2020).

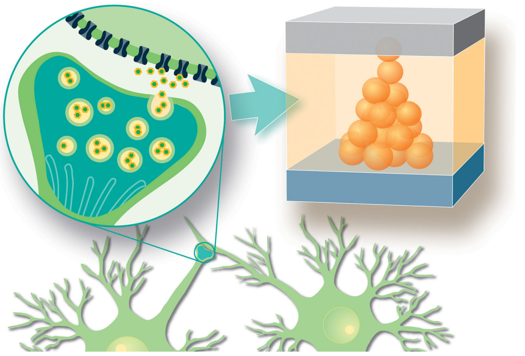

FIGURE 1
In analogy to biological synapses, two terminal solid state nanodevices such as ReRAM can store synaptic weights as electrical resistance states. The devices, consisting simply of patterned metal-insulator-metal material stacks, have an adjustable resistance level determined by the ionic configuration inside the insulating layer. This nano-ionic mechanism also exhibits non-ideal properties such as stochasticity and noise.

A specific functionality required in NC applications is the ability to mimic synaptic connections and plasticity by allowing the storage of large numbers of interconnected and continuously adaptable resistance values. Several candidate memory technologies such as MRAM, ReRAM, PCM, CeRAM, are emerging to cover this behavior using different physical mechanisms (Chen et al., 2014; Liu et al., 2014; You Zhou and Ramanathan, 2015; Yu and Chen, 2016). Among these, ReRAM is attractive for its simplicity of materials and device structure, providing the necessary CMOS compatibility and scalability (Waser et al., 2009). ReRAM is essentially a two terminal nanoscale electrochemical cell, whose variable resistance state is based on manipulation of the point defect configuration in the oxide material (depicted in Figure 1). This redox-based switching mechanism is intrinsically analog, allowing stable resistance levels to be stored and adjusted through application of bipolar voltage stimuli. However, non-idealities such as stochasticity, nonlinearity, and noise are prominent features of these devices that critically impact the performance of neuromorphic systems composed of them (Kim et al., 2018).

Modern ML models have reached an astonishingly large and ever-increasing size, with recent examples exceeding a hundred billion weights (Brown et al., 2020). Before comparable neuromorphic hardware using artificial solid-state synapses can become a reality, large-scale network designs need to first be implemented and evaluated in computer simulations. Training, validation, and optimization of such networks is a process that involves a huge number of simulated devices, voltage pulses, and current readouts. Within this process, it is important to accurately consider the constraints of the underlying hardware in detail. Therefore, lightweight, fast, and accurate stochastic simulations of the individual synaptic devices is a key requirement.

Traditionally, device modeling begins with a physical description of the materials and processes involved. In the case of ReRAM, the physical situation is immensely complicated with many degrees of freedom, and accurate modeling is a wide-scale and ongoing research undertaking. Efforts in this direction are motivated by advancing an understanding of physical and chemical dependencies that can in principle inform design choices on physically justified grounds. In the past decade, many different computational techniques have been employed to furnish device models, from ab initio density-functional theory (DFT), molecular dynamics (MD), kinetic Monte Carlo (KMC), finite element method (FEM), as well as ordinary differential equation (ODE) and differential algebraic equation (DAE) solvers (Ascoli et al., 2015; Jiang and Stewart, 2017; Messaris et al., 2018; Stewart, 2019; Kopperberg et al., 2021). The resulting models exist on a spectrum of physical abstraction, such that the cost of increasing computational speed is generally a trade-off in physical accuracy/detail (Ielmini and Milo, 2017).

Device models that naturally encompass stochasticity do so at the cost of complexity needed to compute the physical scenario in high detail. For example, atomistic KMC simulates switching processes with atomic precision and is inherently stochastic but requires hours of computation per cycle even for small individual cell volumes (e.g., 125 nm^2 Abbaspour et al., 2020). At the other end of the spectrum, dynamic models based on numerical solutions of ODE systems are designed to run significantly faster while sometimes aiming to remain physically realistic. However, their higher speed invariably comes at the cost of approximations, simplifications, and omissions of physical reality. Typically, device operation is distilled to a dynamical description of one or two state variables, such as a conducting filament length, radius, or a defect concentration.

Due in part to ambiguity in their high dimensional parameter space, a given ODE model encompasses a diverse range of possible cell behaviors and has the flexibility to approximately match measurement data (Mayer et al., 2010; Reuben et al., 2019). However, fitting the model to data is commonly an *ad-hoc*, manual, and/or unspecified procedure. Having dispensed with the atomistic sources of variability, ODE models are fully deterministic by default. Where stochasticity is required, it is accounted for by injecting noise into the state variables or parameters of the model (Maria Puglisi et al., 2015; Li et al., 2017; Bengel et al., 2020). Due to the unique experimental challenges posed by electrical measurement of ReRAM, the data used for fitting is not necessarily statistically sufficient nor measured under relevant electrical conditions and timescales. While models can be tuned by hand to roughly match the dispersion observed in a measurement (Chen and Yu, 2015; Jiang et al., 2016), they generally fail to accurately reproduce the complex statistical properties of actual devices.

The main purpose of ODE device models is to be computationally efficient enough to support circuit simulation. Still, nonlinear ODE solvers require many finely spaced timesteps and a considerable amount of total time to compute dynamical trajectories. Although they have been successfully used to demonstrate small scale circuitry such as logic elements and small crossbar arrays (Bocquet et al., 2014; Huang et al., 2017; Siemon et al., 2019; Wald and Kvatinsky, 2019), benchmarks or indications of run time for ODE-based simulations have so far not been supplied. Except for extremely small ML model sizes on the order of 10^3 weights or below, demonstrations of network performance are expected to remain computationally intractable *via* conventional circuit simulation.

In this article, we address these device modeling challenges with a new type of generative model for arrays of artificial synapses. The main objective of the model is to accurately reproduce the statistical properties of fabricated devices while remaining computationally lightweight. Starting with newly available electrical measurement data as an input, this phenomenological model is systematically fit using a well defined statistical regression analysis. The exclusive use of easily computable analytical expressions provides close quantitative agreement with relevant experimental observation. Taking advantage of parallel resources on a modern CPU and GPU, we demonstrate the ability to simulate hundreds of millions of synaptic connections with over 10^8 weight updates per second. With its high throughput and low memory footprint, the model can be usefully employed to simulate large arrays of solid-state synapses for investigation of emerging NC concepts on a large scale.

Methods

The basic requirement for an electronic device serving as an artificial synapse is to moderate the flow of electrical signals through connections in a network. Left undisturbed, the device ideally maintains a fixed weight, or dependence between the voltage across the two device terminals, U, and the resulting current through the device, I. Further, for learning there must be some means of affecting the weight in a durable way. ReRAMs are bipolar devices that have an adjustable (potentially nonlinear) non-volatile resistance state, which is based on the size and shape of a conducting filament that partially or fully bridges the insulating gap of the oxide material. Simplistically, when U exceeds certain threshold levels, the resistance state begins to transition toward lower or higher values depending on the voltage polarity, which corresponds to growth and shrinkage of the conducting filament. When the filament only partially bridges the insulating gap, conduction may be limited for example by tunneling through a Schottky barrier of a material interface, leading to a relatively high resistance levels (Yang et al., 2008; Waser et al., 2009). As the filament grows and gradually bridges the gap, the resistance decreases as conduction transitions into the ohmic type.

In designing our model, we place high priority on speed and fitting accuracy. One of the beginning assumptions is that in every possible device state, the current can be represented by a linear mixture of two fixed polynomials in U. These two polynomials, which are each estimated from a fit to measurement data, can be thought of as limiting cases for the highest possible high resistance state, $I_{\mathrm{HHRS}}(U)$, and lowest possible low resistance state, $I_{\mathrm{LLRS}}(U)$. The device current in all possible resistance states is then given by

$$I(r, U) = rI_{\mathrm{HHRS}}(U) + (1 - r)I_{\mathrm{LLRS}}(U), \qquad (1)$$

conveniently reducing the description of the conduction in the material to a single state variable $0 < r < 1$. This set of functions can be efficiently evaluated by Horner's algorithm and serve as a close enough approximation to the true non-linear conduction behavior for our purposes.

In ReRAM, the overall resistance state as well as the transition behavior is affected by a vast number of different possible configurations of ionic defects in the material, giving rise to the observed stochastic behavior and history dependence (Figure 2). Rather than attempting to describe the ionic transport physically, we turn instead to measurement data to directly provide the necessary statistical information. A discrete multivariate stochastic process based on a Structural Vector Autoregression (SVAR) model is fit to the data and used to generate latent variables that guide the state evolution of simulated memory cells. As a cell is exposed to voltage signals, new terms of the SVAR model are realized by a sum of easily computable linear transformations of past states and pseudorandom vectors.

As an overview, the experimental and simulation approach that will be elaborated in this section can be shortly summarized as follows:

1. A fabricated ReRAM cell is experimentally driven through a large number of resistance cycles by applying a continuous periodic voltage signal while measuring the resulting current.
2. A time series of feature vectors, $\boldsymbol{x}_n$, composed of resistance values and switching threshold voltages, is extracted from each of the measured cycles.
3. A discrete stochastic process, $\boldsymbol{x}_n^*$, is constructed to enable generation of simulated feature vectors that reproduce the measured distributions as well as the long-range correlation structure of $\boldsymbol{x}_n$.
4. An array of simulated cells are instantiated according to independent realizations of $\boldsymbol{x}_n^*$ to represent cycle-to-cycle variations, together with a random scaling vector $\boldsymbol{s}_m$ to represent device-to-device variations.
5. Two programming methods are exposed for each cell; one to apply voltages and another to make realistic current readouts. Applied voltages above the generated thresholds alter the

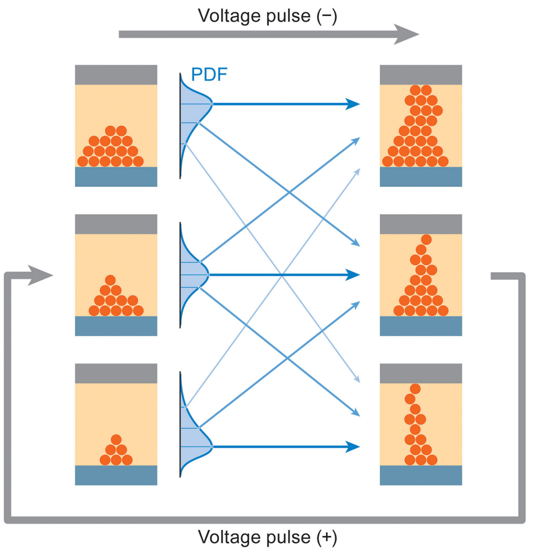

FIGURE 2
Resistance states reached in a synaptic ReRAM device through application of voltage pulses exhibit a probabilistic dependence on past states, leading to long-range correlations that also involve other parameters such as the voltage thresholds required for switching. Starting with effectively infinite state possibilities, represented by the three cells on the left, an applied voltage pulse brings about a set of transition probabilities to many possible future states (right).

device state, following an empirical structure which encodes the resistance transition behavior and allows access to a range of resistance states. Each voltage driven resistance cycle triggers the generation of new stochastic terms from x_n^*, which govern the progression to future states.

Data collection

For the purposes of stochastic modeling, electrical measurement data is needed that capture relevant information about the internal state of a memory cell and its variation cycle-to-cycle (CtC) and device-to-device (DtD). However, ReRAM measurements performed at operational speed typically make exclusive use of rectangular voltage pulse sequences, which yield very little useful state information. On the other hand, measurements applying continuously swept voltage signals while sampling the resulting current are more suitable because much more information is collected each cycle, such as switching threshold voltages, current-voltage nonlinearity, resistance states, and transition behavior.

Conventionally, measurements employing voltage sweeps are carried out using the source measure units (SMUs) of commercial semiconductor parameter analyzers (SPAs). However, SMUs make heavy use of averaging to measure noisy signals at high resolution and thus sample too slowly to collect cycling data in a meaningful quantity. Furthermore, because two-terminal switching devices are prone to electrical instability and runaway transitions, voltage sweeping measurements usually require integrated current limiting transistors to avoid destruction or rapid degradation of the cell. This presents a significant fabrication overhead and limits the materials available for study. In light of these challenges, the input data for the present stochastic model was acquired using a custom measurement technique, introduced in detail in a recent publication (Hennen et al., 2021). The setup uses an external current-limiting amplifier circuit to allow for collection of sweeping measurements at over six orders of magnitude higher speeds than SMUs, while also eliminating the cumbersome requirement of on-chip current limiting.

The ReRAM cell used for measurement of cycling statistics was integrated in the back end of line of a 130 nm CMOS process, between M4 and M5 aluminum metal lines (Figure 3). On M4, a damascene TiN *via* followed by a patterned TiN bottom electrode were processed, forming the inert electrode of the device. The memory stack was then deposited. First, 10 nm HfO_2 deposited by atomic layer deposition (using $HfCl_4$ and H_2O precursors) acts as the resistive switching layer (Nail et al., 2016). Then, a 20 nm Ti scavenging layer was deposited by physical vapor deposition, allowing creation of oxygen vacancies within the HfO_2 during the memory operation. A 100 nm TiN top layer was used to cap the device. Deep ultraviolet photolithography and dry etching were used to pattern the memory dot, defining the active area. A SiN capping layer was used to isolate the memory from adjacent cells. Top vias were then opened by photolithography and dry etching in order to contact the memory dots. Finally, aluminum M5 was deposited and patterned to complete the process flow.

The measured device was electrically isolated with contact pads leading directly to the top and bottom device electrodes, with no access transistor or added series resistance. Using a fixed 100 μA current limit in the SET polarity, the pristine cell was electroformed by application of 100 μs duration triangular pulses with incrementally increasing amplitude until a current jump was recorded near 3 V. For all subsequent cycling, a 1.5 V amplitude 10 kHz triangular waveform was applied. The cell was first exercised for 2.4×10^6 cycles before 10^6 additional cycles were collected for analysis. Current (I) and voltage (U) waveforms were simultaneously recorded with 8-bit resolution and with a sample rate of 1,042 samples per cycle. The measured current array was smoothed with a moving average filter to improve the quality of the raw data before further analysis. An adaptive rectangular window size was used to preserve current steps in the signal, with the maximum window size of 25 samples gradually reducing to a minimum of 3 samples at the pre-detected locations of SET transitions of each cycle. After smoothing, the contiguous I and U waveforms were split into

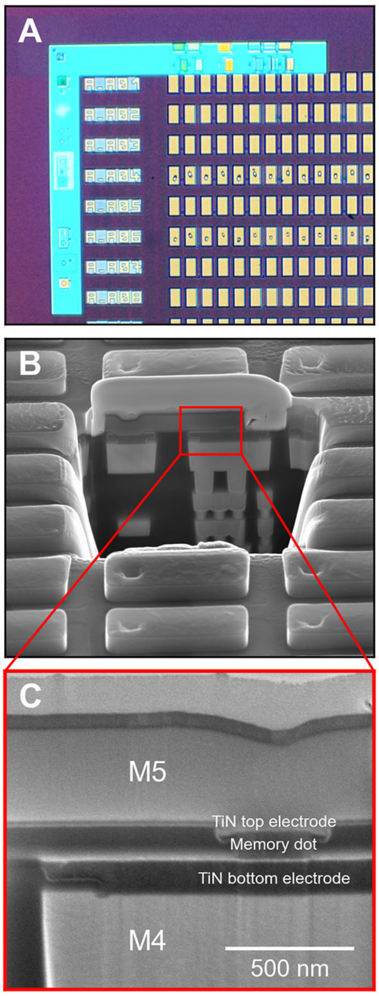

FIGURE 3
Scanning electron micrographs of the ReRAM cell design used for electrical measurement. **(A)** shows an optical image of the array of contact pads, **(B)** shows a cross-section of a cell, and **(C)** shows a zoom-in of the resistive memory between metalization layers M4 and M5.

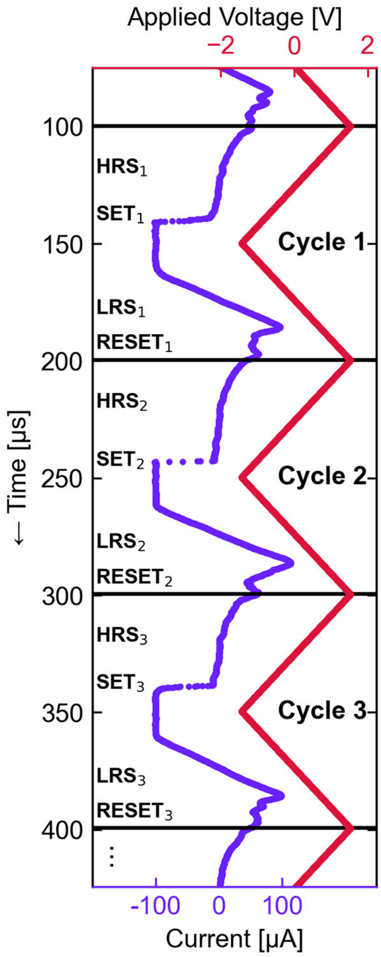

FIGURE 4
Measured time dependence of *I* and *U* waveforms resulting from the ReRAM cycling experiment. The waveforms are divided into 10^6 indexed cycles, the first three of which are shown. From this dataset, the periodic temporal sequence of the states and events of each cycle (HRS_n, SET_n, LRS_n, RESET_n) is extracted and subject to statistical modeling.

indexable cycles at most positive value of the periodic applied voltage (see Figure 4).

Each cycle exhibits the following temporal sequence of states and events: a high resistance state (HRS), a transition (SET) out of the HRS into the following low resistance state (LRS), and finally another transition (RESET) into the next HRS. Current vs. voltage (I, U) plots for a subset of the collected cycles are shown in Figure 5, which highlights the significant stochastic CtC variations. The observed characteristics are typical for ReRAM devices subjected to voltage-controlled sweeps — on average, there is relatively higher voltage non-linearity in the HRS than in the LRS, and a large proportion of the SET transitions are abrupt with respect to the applied voltage. The SET transition times as defined by the time spent between −30 µA and −90 µA is connected to the voltage sweep rate, and was distributed between 100 ns and 5 µs in this case. The RESET transitions, in contrast, proceed relatively gradually over a voltage range of approximately 700 mV, following a concave transition curve with N-type negative differential resistance.

Feature extraction

The full I, U cycling measurement just described consists of over 16 GB of numerical data and would not be practical to model on a point-by-point basis. Therefore, we aim to compress the dataset while retaining enough information such that the full

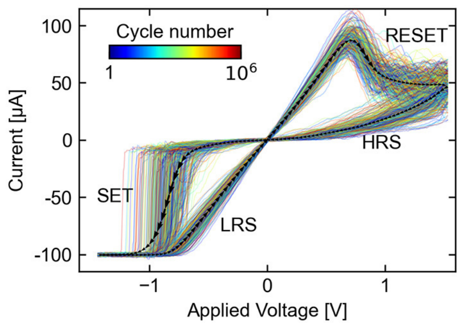

FIGURE 5
A subset of the 10^6 measured (I, U) cycles used as input to the stochastic model. The black arrowed path shows the average (I, U) curve and its temporal direction. Different cycle indices are represented by colored paths, which show significant statistical variation.

(I, U) characteristics can be approximately reconstructed from the compressed representation. Accordingly, the full dataset is reduced to a vector time series of distinguishing features of each cycle. Four scalar features were chosen for extraction: the value of the HRS, $R_H[\Omega]$, the SET threshold voltage, $U_S[V]$, the value of the LRS, $R_L[\Omega]$, and the RESET voltage, $U_R[V]$. We denote the series as

$$\boldsymbol{x}_n = \begin{bmatrix} R_{H,n} \\ U_{S,n} \\ R_{L,n} \\ U_{R,n} \end{bmatrix} = \begin{bmatrix} R_H \\ U_S \\ R_L \\ U_R \end{bmatrix}_n , \tag{2}$$

where $n = \{1, 2, \ldots, 10^6\}$ is the set of cycle indices. The feature vector elements, whose precise definition follows, are chronologically ordered from top to bottom as they occur in the measurement dataset.

The SET voltage U_S, or the voltage where the cell resistance abruptly decreases, is extracted from each cycle as the absolute value of the linearly interpolated U corresponding to the first level crossing of $I = -50\ \mu\text{A}$. The RESET voltage U_R, defined as the voltage where the reset process begins, is determined from the I datapoints by peak detection using simple comparison of neighboring samples. Here, only the increasing section of the voltage sweep with $U > 0$ is considered. The voltage corresponding to the first encountered peak with prominence $\geq 5\ \mu\text{A}$ is taken as the RESET voltage. If no peak satisfies this criterion, the peak with maximum prominence is taken instead.

The device current for any static state is approximated in our model as a polynomial function of the applied voltage. The values of R_H and R_L are likewise extracted from least squares polynomial fits to appropriate subsets of the measured (I, U) data of each cycle. The HRS is fit with a 5th degree polynomial on the decreasing U sweep in the variable range $U_S + 0.1\ \text{V} \leq U \leq 1.5\ \text{V}$ and $-25\ \mu\text{A} \leq I \leq 80\ \mu\text{A}$, and the LRS is fit with a 3rd degree polynomial on the increasing part of the V sweep in the range $-0.7\ \text{V} \leq U \leq U_R - 0.05\ \text{V}$ and $-80\ \mu\text{A} \leq I \leq 120\ \mu\text{A}$. The fits are constrained such that the 0th order coefficient equals 0 A, and the 1st order coefficient is ≥ 1 nA/V. The values of R_H and R_L are then defined as the static resistance of the respective polynomials at a fixed voltage $U_0 = 200$ mV.

An overview of the result of this feature extraction is given in Figure 6. The 10^6 cycles proceeded without significant long-term drift from the overall mean value,

$$\bar{\boldsymbol{x}}_n = \begin{bmatrix} 166.5\ \text{k}\Omega \\ 0.85\ \text{V} \\ 8.2\ \text{k}\Omega \\ 0.72\ \text{V} \end{bmatrix} , \tag{3}$$

but with significant variations in each feature between cycles. A prominent characteristic of this data is that it is strongly correlated over long cycle ranges, as quantified in Figure 14. The asymmetric marginal distributions for each of the features were very well resolved due to the large number of samples, and they did not accurately converge to any analytical probability density function (PDF) in common use, including the normal and log-normal.

Stochastic modeling

This section will introduce the statistical methods used to model the internal states of an array of synaptic ReRAM devices, including CtC and DtD variability effects. The handling of voltages applied to the cells as well as the simulation of realistic readouts of the resistance states will also be established. To help orient the reader, the overall structure of the generative model that will be described is provided in advance in Figure 7.

Cycle-to-cycle variations

In seeking to represent the input time series $\boldsymbol{x}_n$ with a stochastic process, the main goals are to recreate the marginal distributions as well as the correlation structure of its vector components. To achieve the first goal with high generality, we use an approach based on transformation of the measured densities to and from the standard normal distribution $\mathcal{N}(0, 1)$. This way, a single process can be used to achieve any set of marginals presented by the input data, with the relatively unrestrictive requirement that this base process generates normal marginals. Notationally, we define and apply an invertible, smooth mapping $\boldsymbol{\Gamma} : \mathbb{R}^4 \rightarrow \mathbb{R}^4$ that normalizes the marginal distributions of the vector components,

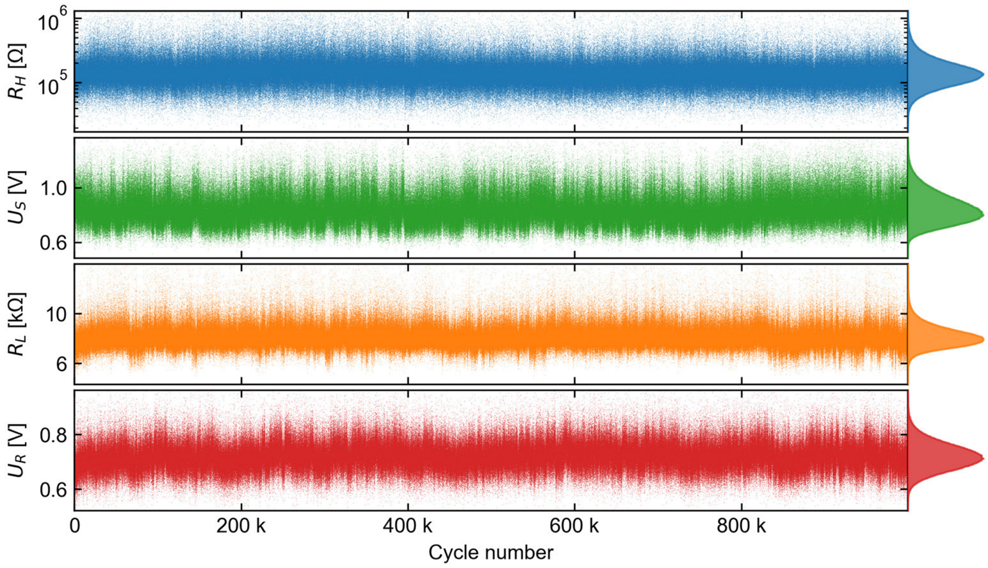

FIGURE 6
A view of the feature vector time series extracted from each of 10^6 measured (I, U) cycles. Each feature, which represents either a resistance state or a switching voltage, has its marginal histogram shown on the right.

$$\boldsymbol{x}_n = \begin{bmatrix} R_H \\ U_S \\ R_L \\ U_R \end{bmatrix}_n \xrightarrow{\boldsymbol{\Gamma}} \begin{bmatrix} \widehat{R}_H \\ \widehat{U}_S \\ \widehat{R}_L \\ \widehat{U}_R \end{bmatrix}_n = \widehat{\boldsymbol{x}}_n, \tag{4}$$

where a hatted variable signifies that it is distributed as $\mathcal{N}(0, 1)$. We then construct a base process $\widehat{\boldsymbol{x}}^*_n$ whose marginals are normal, and finally transform its output back to the original data distributions *via* the inverse map $\boldsymbol{\Gamma}^{-1}$. The overall process $\boldsymbol{x}^*_n$ is thus defined,

$$\widehat{\boldsymbol{x}}^*_n = \begin{bmatrix} \widehat{R}^*_H \\ \widehat{U}^*_S \\ \widehat{R}^*_L \\ \widehat{U}^*_R \end{bmatrix}_n \xrightarrow{\boldsymbol{\Gamma}^{-1}} \begin{bmatrix} R^*_H \\ U^*_S \\ R^*_L \\ U^*_R \end{bmatrix}_n = \boldsymbol{x}^*_n, \tag{5}$$

where a star indicates a generated random variable to distinguish from variables originating from measurement data.

This type of density transformation procedure is a widely used technique for working with arbitrary distributions, which finds application in a variety of fields and can be constructed in many different ways (Cario and Nelson, 1996; Rezende and Mohamed, 2015). While the transformation is trivially constructed in the case where the target quantile function and its inverse are each analytically defined, we do not make this assumption in the present scenario. A simple numerical method in this case is a so-called quantile transform, where the input and output quantile functions are each discretely sampled and the transformation is defined through a direct map between bins or through interpolation. The main requirement for $\boldsymbol{\Gamma}$ in our model, however, is that its inverse (Equation 5) is easy to evaluate without causing cache misses due to memory access, thus it is preferable to avoid referencing and interpolation of large look-up tables. The forward transformation (Equation 4), on the other hand, only needs to be computed once for model fitting and is not used for the generating process. We therefore define $\boldsymbol{\Gamma}^{-1}$ as essentially a quantile transform, operating on each feature independently, that is evaluated from a fit of the quantiles to a specific analytic function. Namely,

$$\boldsymbol{\Gamma}^{-1}(\widehat{\boldsymbol{x}}_n) = \exp \begin{bmatrix} \gamma_1(\widehat{R}_{H,n}) \\ \gamma_2(\widehat{U}_{S,n}) \\ \gamma_3(\widehat{R}_{L,n}) \\ \gamma_4(\widehat{U}_{R,n}) \end{bmatrix} = \boldsymbol{x}_n, \tag{6}$$

where γ_1–γ_4 are each 5th degree polynomials, and the exponential function is applied element-wise. The coefficients of the polynomials are fit to standard normal quantiles vs. those of the respective (log) features, sampled at 500 equally spaced values between 0.01 and 0.99. The fitted polynomials are checked for monotonicity within four standard deviations above and below zero, and the forward transformation,

$$\boldsymbol{\Gamma}(\boldsymbol{x}_n) = \begin{bmatrix} \gamma_1^{-1}(\log R_{H,n}) \\ \gamma_2^{-1}(\log U_{S,n}) \\ \gamma_3^{-1}(\log R_{L,n}) \\ \gamma_4^{-1}(\log U_{R,n}) \end{bmatrix} = \widehat{\boldsymbol{x}}_n, \tag{7}$$

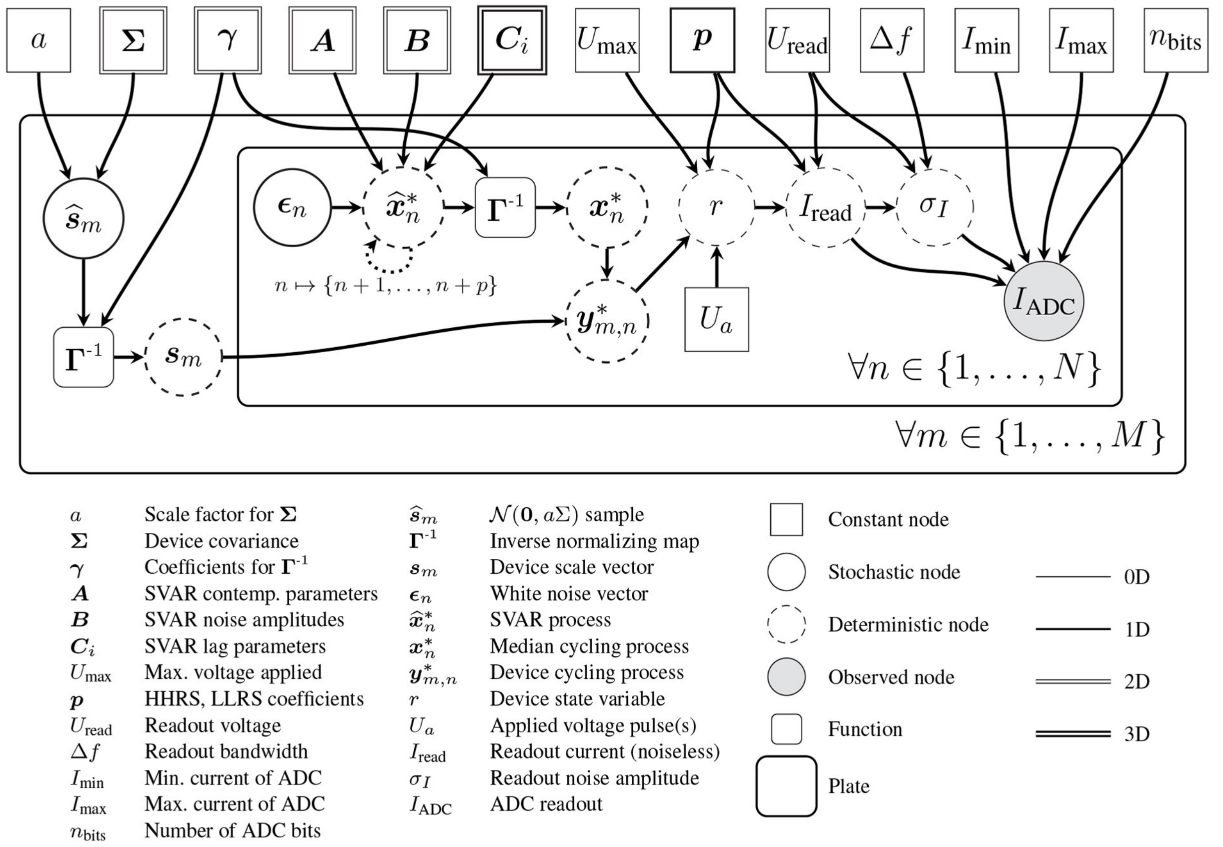
a
$\boldsymbol{\Sigma}$
γ
$\boldsymbol{A}$
$\boldsymbol{B}$
$\boldsymbol{C}_i$
U_{max}
$\boldsymbol{p}$
U_{read}
Δf
I_{min}
I_{max}
n_{bits}
$\widehat{\boldsymbol{s}}_m$
$\boldsymbol{\epsilon}_n$
$\widehat{\boldsymbol{x}}_n^*$
$\boldsymbol{\Gamma}^{-1}$
$\boldsymbol{x}_n^*$
r
I_{read}
σ_I
$n \mapsto \{n+1, \dots, n+p\}$
$\boldsymbol{y}_{m,n}^*$
U_a
I_{ADC}
$\boldsymbol{\Gamma}^{-1}$
$\boldsymbol{s}_m$
$\forall n \in \{1, \dots, N\}$
$\forall m \in \{1, \dots, M\}$
a Scale factor for $\boldsymbol{\Sigma}$
$\boldsymbol{\Sigma}$ Device covariance
γ Coefficients for $\boldsymbol{\Gamma}^{-1}$
$\boldsymbol{A}$ SVAR contemp. parameters
$\boldsymbol{B}$ SVAR noise amplitudes
$\boldsymbol{C}_i$ SVAR lag parameters
U_{max} Max. voltage applied
$\boldsymbol{p}$ HHRS, LLRS coefficients
U_{read} Readout voltage
Δf Readout bandwidth
I_{min} Min. current of ADC
I_{max} Max. current of ADC
n_{bits} Number of ADC bits
$\widehat{\boldsymbol{s}}_m$ $\mathcal{N}(\boldsymbol{0}, a\Sigma)$ sample
$\boldsymbol{\Gamma}^{-1}$ Inverse normalizing map
$\boldsymbol{s}_m$ Device scale vector
$\boldsymbol{\epsilon}_n$ White noise vector
$\widehat{\boldsymbol{x}}_n^*$ SVAR process
$\boldsymbol{x}_n^*$ Median cycling process
$\boldsymbol{y}_{m,n}^*$ Device cycling process
r Device state variable
U_a Applied voltage pulse(s)
I_{read} Readout current (noiseless)
σ_I Readout noise amplitude
I_{ADC} ADC readout
Constant node
Stochastic node
Deterministic node
Observed node
Function
Plate
0D
1D
2D
3D

FIGURE 7
Graphical model depicting the relationships between all parameters and latent variables involved in the stochastic synapse model. Plate notation is used to represent N switching cycles of M devices, each yielding an observed readout current. The dotted recurrent arrow denotes a connection to each of the p following frames, as needed by the history dependent stochastic process.

is computed using numerical inverse of the γ polynomials. A visualization of the function $\boldsymbol{\Gamma}$ as well as the marginal histograms corresponding to input series $\boldsymbol{x}_n$ and output series $\widehat{\boldsymbol{x}}_n$, are shown in Figure 8.

Now that we have transformed the input measurement data into a normalized vector time series $\widehat{\boldsymbol{x}}_n$, a suitable stochastic process will be chosen for fitting. This process should serve as a useful approximation to the true physical mechanisms that generated the data, capturing the long-range correlation structure of the observed features. Time series analysis is broadly used across scientific and engineering domains, but despite its applicability to the rich statistical behavior displayed by resistive switching devices, device models have not yet widely employed dependent stochastic processes. Many models and analyses assume for convenience that features are independently and identically distributed according to a normal or lognormal PDF (Chen, 2015; Li et al., 2017). However, there is not a strong theoretical basis for this assumption in a highly nonlinear and path-dependent system based on continuous evolution of conducting filaments. Dependent stochastic processes, on the other hand, more appropriately allow for a description of the dependence of future states on past states.

Simple models in the category of Markov chains have been considered as generating processes for memory cells. A rudimentary example is a 1-dimensional random walk process, where each future state is computed as a random additive perturbation on the previous state (Bengel et al., 2020). While random walk represents a reasonable short-range approximation, it has the well known property that the expected absolute distance between the initial value and the Nth value is proportional to $\sqrt{N}$ for large N, causing the process to eventually drift to unphysical values without the use of artificial constraints.

Autoregressive (AR) models are simple univariate processes sharing some characteristics of random walk, but based additionally on a deterministic linear dependence on past observations. Each new term of an AR(p) (AR of order p) model is computed by linear combinations of p previous (lagged) values together with a noise term, producing processes that are wide-sense stationary and mean-reverting within suitable

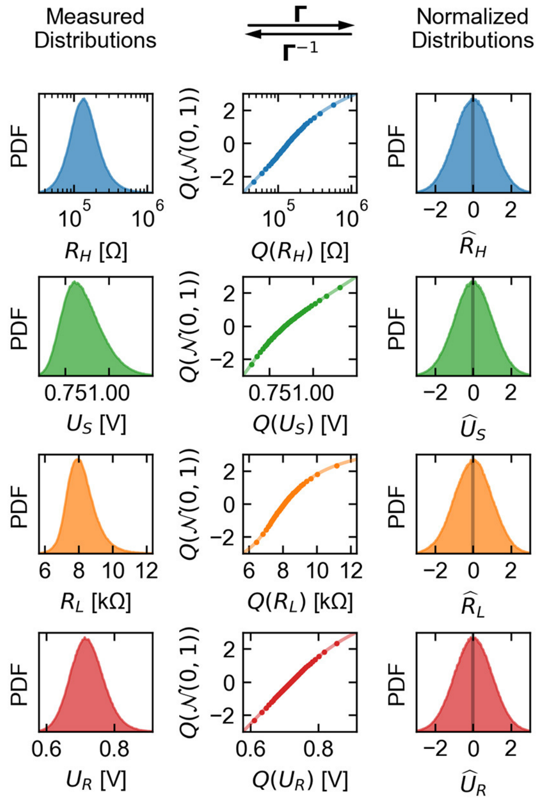

FIGURE 8
Visualization of the invertible normalizing transformation $\boldsymbol{\Gamma}$ that is applied to the measured feature vectors before fitting with a base stochastic process. The left column shows the marginal PDFs of the vector time series $\boldsymbol{x}_n$ extracted from measurement. The center column shows the input and output quantile-quantile plots with the fitted log-polynomial function used to transform the distributions (here, Q denotes the quantile function of its argument). The right column is the result of applying $\boldsymbol{\Gamma}$ to the input data, producing $\widehat{\boldsymbol{x}}_n$ whose elements are normally distributed.

parameter ranges (Hamilton, 1994; Lütkepohl, 2005). The few times they have appeared in the literature, low order models like AR(1) and AR(2) were used to describe state variables independently (e.g., a sequence of high and/or low resistance states) (Fantini et al., 2015; Roldán et al., 2019). Here we pursue a more comprehensive statistical description of the interrelations between the different variables contained in the vectors $\widehat{\boldsymbol{x}}_n$ which takes into account long-range correlations $p \gg 1$. This is enabled by using a VAR(p) model (vector AR of order p), which is the multivariate counterpart of the AR model applicable to discrete vector time series (Hamilton, 1994; Lütkepohl, 2005).

We adopt in particular a Structural VAR (SVAR) formulation of the model, which is a factorization that makes the relationships between the contemporaneous (same index) variables explicit. The model has the form

$$\boldsymbol{A}\widehat{\boldsymbol{x}}_n^* = \sum_{i=1}^{p} \boldsymbol{C}_i \widehat{\boldsymbol{x}}_{n-i}^* + \boldsymbol{B}\boldsymbol{\epsilon}_n, \tag{8}$$

where $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}_i$ are 4×4 matrices of model parameters, and $\boldsymbol{\epsilon}_n$ is a 4-dimensional standard white noise process. With this formulation we impose a general structure of causal ordering for the generated random variables consistent with the chronological chain of measurement events. Within this structure, each variable may have a causal and deterministic effect on all future variables within range p, as visualized by the graph of Figure 9. The size of these effects are all subject to fitting *via* the coefficients of the model. Constraints on the structural parameters,

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ A_{21} & 1 & 0 & 0 \\ A_{31} & A_{32} & 1 & 0 \\ A_{41} & A_{42} & A_{43} & 1 \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} B_{11} & 0 & 0 & 0 \\ 0 & B_{22} & 0 & 0 \\ 0 & 0 & B_{33} & 0 \\ 0 & 0 & 0 & B_{44} \end{bmatrix} \tag{9}$$

enforce the desired causal structure while assuming an uncorrelated noise driving process. Model fitting was performed using the Python statsmodels package (Seabold and Perktold, 2010), wherein a VAR(p) model is first fit by ordinary least squares regression, and a maximum likelihood estimate is then used to determine the structural decomposition.

Device-to-device variations

So far, we have only considered the statistical modeling of the cycling process of a single memory cell. However, the purpose of the presented model is to simultaneously simulate a large number of cells in a network. Individual memory devices on a wafer generally show statistical variations, mainly arising due to defects and non-uniformities in fabrication (Fantini et al., 2013; Dalgaty et al., 2021). These DtD variations depend strongly on the lithography processes and materials used. They can also originate from intrinsic factors and are influenced by conditions during the electroforming of each cell (Butcher et al., 2012; Zhao et al., 2014). Because of the potential positive or negative impact on network performance, it is important for the model to account for the DtD variability (Moon et al., 2019; Dalgaty et al., 2021).

The electrical effect of device variability is modeled with each cell using a modification of the same underlying SVAR cycling process. Device-specific processes are defined as members of a parametric family of processes, all based on element-wise scaling of $\boldsymbol{x}_n^*$, where the scaling factors are themselves random vectors. The specific process is denoted

$$\boldsymbol{y}_{m,n}^* = \boldsymbol{s}_m \odot \boldsymbol{x}_n^*, \tag{10}$$

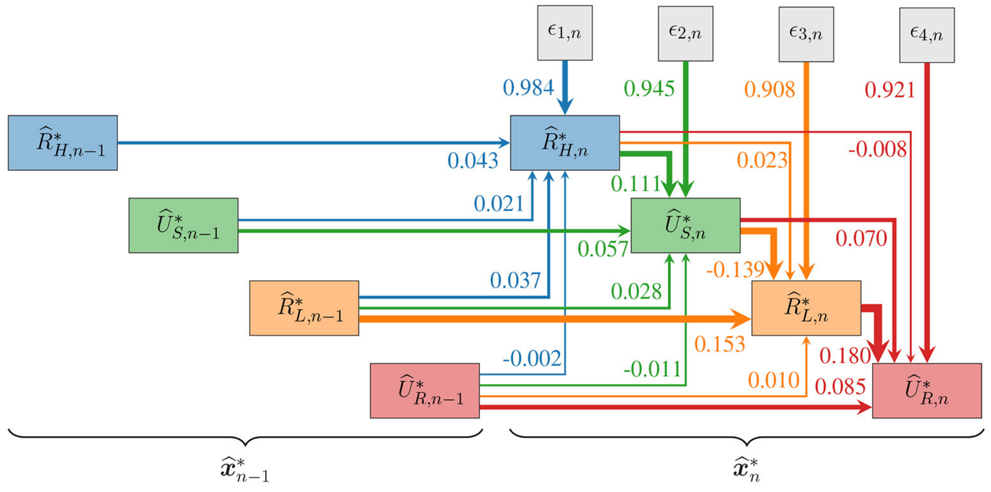

FIGURE 9
A weighted graph displaying the causal structure of the utilized SVAR(p) process, showing the nearest temporal contributions to realizations of the random vector $\widehat{\boldsymbol{x}}^*_n$. Arrow weights show the model parameters contained in $\boldsymbol{A}$, $\boldsymbol{B}$ and the upper triangular part of $\boldsymbol{C}_1$ when fit with $p = 100$. The actual SVAR(p) model uses many more connections than shown ($16p + 10$), so that each variable is impacted by all past values of all other variables within cycle range p.

where $m = \{1, 2, \ldots, M\}$ is the device index, $\odot$ is the Hadamard (element-wise) product, and $\boldsymbol{s}_m$ are 4×1 random vectors drawn from a fixed distribution at cell initialization.

The distribution of $\boldsymbol{s}_m$ is chosen so that the features of the median cycles of different devices are distributed and correlated in the same way as the measured cycling data $\boldsymbol{x}_n$. This choice reflects that the covariations of switching features DtD arise in the same physical system with causes and effects that are comparable to those of the CtC variations. To this end, random vectors $\widehat{\boldsymbol{s}}_m$ are drawn from a multivariate normal (MVN) distribution and Γ^{-1} is then reused to map them to the measured CtC distribution,

$$\boldsymbol{s}_m = \boldsymbol{\Gamma}^{-1}(\widehat{\boldsymbol{s}}_m) \oslash \boldsymbol{\Gamma}^{-1}(\boldsymbol{0}), \text{ where } \widehat{\boldsymbol{s}}_m \sim \mathcal{N}(\boldsymbol{0}, a\boldsymbol{\Sigma}). \tag{11}$$

Here, the denominator of the Hadamard division ($\oslash$) sets the median scale vector to the identity, $\boldsymbol{\Sigma} = \text{cov}(\widehat{\boldsymbol{x}}_n)$ is the sample covariance of the normalized measurement data, and a is a free scalar parameter providing adaptability to different DtD covariance levels. A robust determination of a requires measurement of many switching cycles across a large number of devices of interest. Values in the range $a \in [1, 1.5]$ approximately correspond to published DtD measurement samples (Fantini et al., 2013; Dalgaty et al., 2021), but improved processing and electroforming procedures may justify the use of $a < 1$.

Control logic

As components of a network, each simulated cell possesses a resistance state that encodes the weight of a connection. Voltage pulses directly applied to the cells are used to produce resistance state transitions to update the weights. In this model, applied voltage pulses are distinguished only by a scalar amplitude U_a, whether they are in fact square waveforms or they have a more complex shape of an action potential. Although ReRAMs are known to be highly time-dependent devices (Menzel et al., 2015), we assume here that the duration of the pulses are appropriately matched to the experimental timescale, such that a simulated voltage pulse of a given amplitude produces an effect comparable to the experimental voltage sweep at the instant it reaches that same amplitude. Possible state modifications in response to an input pulse is computed with respect to I, U sweeps that are reconstructed from each stochastic feature vector generated for each cycle as illustrated in Figure 10.

As previously specified in Equation (1), every possible electrical state of a device is assumed to correspond to a polynomial $I(U)$ dependence parameterized by a state variable r. It is straightforward to calculate that the state variable for a curve passing through an arbitrary (I, U) point is uniquely given by the function

$$r(I, U) = \frac{I_{\text{LLRS}}(U) - I}{I_{\text{LLRS}}(U) - I_{\text{HHRS}}(U)}. \tag{12}$$

Therefore, the state variable corresponding to any static resistance level R (evaluated at U_0) can be calculated using

$$r(R) = \frac{I_{\text{LLRS}}(U_0) - U_0 R^{-1}}{I_{\text{LLRS}}(U_0) - I_{\text{HHRS}}(U_0)}. \tag{13}$$

The $I(U)$ curves for the electrical states corresponding to the HRS and LRS of each cycle, hereafter called $I_{\text{HRS,n}}(U)$ and $I_{\text{LRS,n}}(U)$, are defined according to equations (1) and (13)

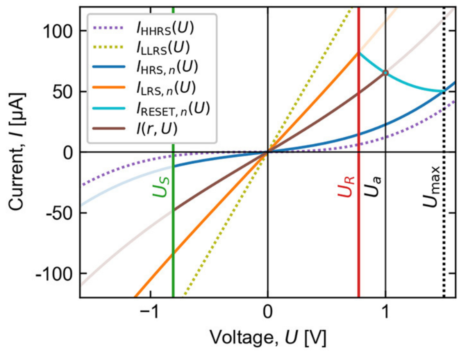

FIGURE 10
Conduction polynomials and threshold voltages allow reconstruction of (I, U) cycles from generated feature vectors. Simulated resistance switching is such that the conduction state $I(r, U)$ induced by an applied voltage U_a intersects the reconstructed cycle at $U = U_a$. For visual simplicity, the cycle shown begins and ends in the same HRS ($R_{H,n} = R_{H,n+1}$).

such that their static resistance equals the respective value of $R^*_{H,n}$ and $R^*_{L,n}$.

Transitions between the HRS, LRS, and intermediate resistance states (IRS) in response to an applied pulse amplitude U_a follow an empirically motivated structure, represented by the flow chart of Figure 11. The SET transition for the nth cycle HRS$_n$ → LRS$_n$ may occur for negative voltage polarities and follows a simple threshold behavior, fully and instantaneously transitioning the first time a voltage pulse with amplitude $U_a \leq U^*_{S,n}$ is applied. In contrast, the RESET transition LRS$_n$ → HRS$_{n+1}$ occurs gradually in the positive polarity with increasing U_a in the range $U^*_{R,n} < U_a \leq U_{\mathrm{max}}$, where $U_{\mathrm{max}} = 1.5$ V is the maximum voltage applied in the voltage sweeping measurement. A transition curve $I_{\mathrm{RESET,n}}(U)$ is defined to connect the (I, U) points of the two limiting states where the RESET transition begins and ends. The functional form of the transition curve is chosen to be the parabola with boundary conditions

$$I_{\mathrm{RESET,n}}(U^*_{R,n}) = I_{\mathrm{LRS,n}}(U^*_{R,n}) \tag{14}$$

$$I_{\mathrm{RESET,n}}(U_{\mathrm{max}}) = I_{\mathrm{HRS,n+1}}(U_{\mathrm{max}}) \tag{15}$$

$$\left.\frac{dI_{\mathrm{RESET,n}}}{dU}\right|_{U=U_{\mathrm{max}}} = 0. \tag{16}$$

When a voltage pulse in the RESET range is applied, an IRS results which is calculated with reference to the transition curve such that $I(r, U_a) = I_{\mathrm{RESET,n}}(U_a)$. Additional RESET pulses with larger amplitudes may be applied to incrementally increase the cell resistance, with HRS$_{n+1}$ being reached only if $U_a \geq U_{\mathrm{max}}$, after which no further RESET switching is possible for the nth cycle. After either partial or full RESET, the resistance may only decrease again by entering the following LRS$_{n+1}$ with a voltage pulse meeting the SET criterion $U_a \leq U^*_{S,n+1}$.

Readout

Simulated current measurements (readouts) for each individual cell can be generated given an arbitrary readout voltage input U_{read}. The noise-free current level simply corresponds to evaluation of $I(r, U_{\mathrm{read}})$ for each cell. In any real system, however, current readouts are accompanied by measurement noise, which may impact system performance and even present a fundamental bottleneck. Furthermore, in digital systems current readouts are converted to finite resolution by analog to digital converters (ADCs). Due to constraints of power consumption and chip area, ADC resolution is often limited such that digitization is the dominant contributor to the total noise (Ma et al., 2019). Many additional noise sources can be considered, such as $1/f$ noise (Wiefels et al., 2020), but at minimum the Johnson-Nyquist noise and the shot noise should be included because they represent a lower bound of noise amplitude impacting all systems.

To account for measurement noise, each individual current readout includes an additive noise contribution drawn from a normal distribution. The noise amplitude is approximated from the Nyquist and Schottky formulas,

$$\sigma_I = \sqrt{\frac{4k_B T I_{\mathrm{read}} \Delta f}{U_{\mathrm{read}}} + 2qI_{\mathrm{read}}\Delta f}, \tag{17}$$

where Δf is the noise equivalent bandwidth, k_B is the Boltzmann constant, $T = 300$ K is the temperature, q is the electron charge, I_{read} is the noiseless current readout, and U_{read} is the voltage used for readout. The total current is then ideally digitized with an adjustable resolution n_{bits} between adjustable minimum I_{min} and maximum I_{max} current levels.

Program implementation

To facilitate investigations of neuromorphic systems, model implementations designed to simulate arrays of devices were developed in the Julia programming language. Julia is a modern high-level language that is focused on performance and that provides an advanced ML and scientific computing ecosystem. Julia programs compile to efficient native code for many platforms *via* the LLVM compiler infrastructure, and a cursory analysis indicated that single threaded CPU performance of a Julia implementation is up to 5,000 times faster than a Python implementation. Furthermore, as modern computational resources are highly parallel, Julia's support for CPU multi-threading and GPU programming through CUDA.jl (Besard et al., 2019) is an important advantage.

All model parameters corresponding to the device characterized in this article, including different possible SVAR model orders, $p \in [1, 200]$, are stored in a binary file which is read in by the program at startup. Each instantiated cell stores state information and p cycles of history using

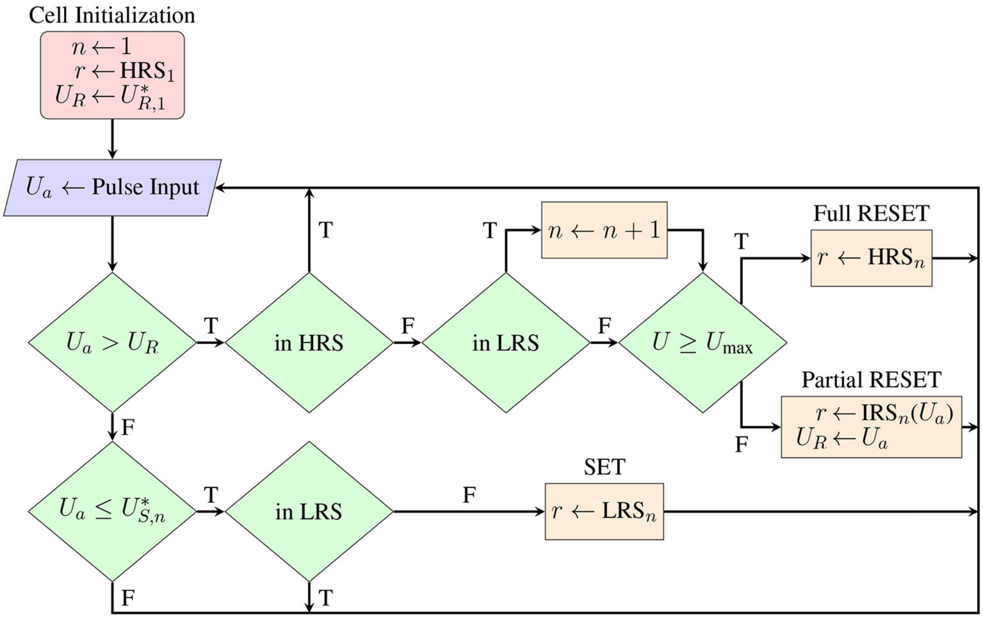
Cell Initialization
$n \leftarrow 1$
$r \leftarrow \mathrm{HRS}_1$
$U_R \leftarrow U^*_{R,1}$
$U_a \leftarrow$ Pulse Input
$U_a > U_R$
in HRS
in LRS
$U \geq U_{\max}$
$n \leftarrow n+1$
Full RESET
$r \leftarrow \mathrm{HRS}_n$
Partial RESET
$r \leftarrow \mathrm{IRS}_n(U_a)$
$U_R \leftarrow U_a$
$U_a \leq U^*_{S,n}$
in LRS
SET
$r \leftarrow \mathrm{LRS}_n$
T
F

FIGURE 11
Logical flow chart showing how applied voltage pulses affect the state of each cell during simulation. Following the experimental observations, SET processes always occur abruptly below a threshold voltage, while partial switching is induced for a range of RESET voltages, with intermediate states bounded for cycle n by resistance values between $R_{L,n}$ and $R_{H,n+1}$. As resistance cycling progresses, later terms of the stochastic driving process are used for limiting resistance states and threshold voltages. Pulse amplitudes not producing a state change are efficiently disregarded.

primarily 32-bit floating point numbers. The total memory footprint grows linearly with the chosen model order and is approximately $16p + 56$ bytes per cell. A reduced form VAR process is used to compute realizations of $\boldsymbol{x}^*_n$, which are lazily evaluated along with the parabolic transition polynomials if and when they are needed. The majority of the necessary runtime computations are formulated as matrix multiplications, which are heavily optimized operations across many different contexts.

The present release contains two model implementations to suit a wide variety of computing platforms and use cases (Hennen, 2022). The first is a CPU optimized version wherein the cells of an array are individually addressable for read/write operations. These operations are naturally parallelized for multi-core processors by partitioning the cells and assigning each partition to independent threads of execution. The second implementation is a GPU accelerated version compatible with CUDA capable GPUs. This version uses a vectorized data structure and parallel array abstractions to take advantage of the implicit parallelism programming model of CUDA.jl. Here, all defined cells are always accessed simultaneously, with each read/write operation employing optimized linear algebra GPU kernels. While the GPU implementation integrates well with other ML components residing in GPU shared memory and achieves higher throughput per cell for large parallel operations, the CPU implementation obtains higher update rates for sparse operations commonly encountered in large-scale models (Pedroni et al., 2019, 2020).

Results

As shown visually in the scatterplot of Figure 12, the stochastic process $\boldsymbol{x}^*_n$ generates data that closely resemble the measurement data $\boldsymbol{x}_n$. The generated distributions match the empirical distributions so closely that it is difficult to visualize their difference. The Wasserstein metric is a distance function defined between probability distributions that can be used to quantify a small discrepancy (Kantorovich, 1960). The first Wasserstein distance was calculated element-wise and averaged across 100 realizations of $\boldsymbol{x}^*_n$ with length 10^6. The result,

$$\overline{\boldsymbol{W}}_1(\boldsymbol{x}_n, \boldsymbol{x}^*_n) = \begin{bmatrix} 5,146\ \Omega \\ 937\ \mu\mathrm{V} \\ 20\ \Omega \\ 356\ \mu\mathrm{V} \end{bmatrix}, \tag{18}$$

is much smaller than the mean feature vector, $\bar{\boldsymbol{x}}_n$ (Equation 3), and independent of the chosen model order. This shows that the goal of reproducing the measurement distributions is well

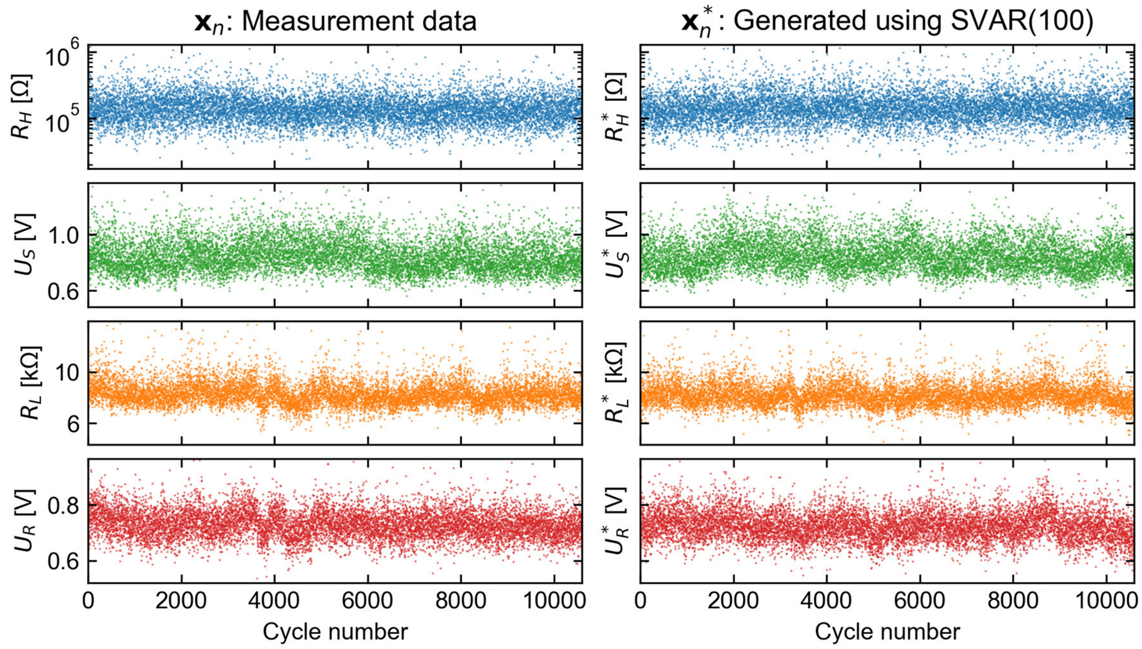

FIGURE 12
Comparison of feature time series extracted from measurement data and those generated by the SVAR-based model. The compared features converge to effectively equivalent distributions and the short-range behavior is qualitatively similar across thousands of cycles.

achieved for the input dataset by using the described method of probability density transformation.

Simulations of full (I, U) cycling measurements (Figure 13A) show close similarity with the measurement data of Figure 5. Multi-resistance-level capability is also demonstrated by a similar simulation involving partial RESET operations by changing the maximum voltage applied (Figure 13B). The dependence of the resulting HRS value on the applied voltage reproduces a non-linear characteristic comparable to experimental findings (Park et al., 2013; Ambrogio et al., 2016).

While a full structural analysis of the fitted SVAR(p) model parameters $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}_i)$ will not be presented here, a few aspects are worthy of note. For the fit corresponding to the particular device and measurement described in this work, the white noise terms are by far the dominant contributors to all four modeled features. The contemporaneous terms ($\boldsymbol{A}$) and first order ($\boldsymbol{C}_1$) terms are the next most significant, which indicates that the most recent cell history is most relevant for generating the proceeding states. Nevertheless, input data correlations persist for many cycles, and the generating process $\boldsymbol{x}_n^*$ successfully reproduces the overall correlation structure of the data up to at least p cycle lags, as shown in detail in Figure 14.

Although no physical effects were explicitly put into the model definition, it is important to recognize that the effects are quantitatively captured and put into a useful statistical context by the SVAR model fitting procedure. The model weights contained in $\boldsymbol{A}, \boldsymbol{B}$, and $\boldsymbol{C}_i$ quantify deterministic relationships between past and future variables even in the presence of large random fluctuations. As seen in the graph of Figure 9, the four strongest coefficients in the fitted model correspond to the relationships

$$\widehat{R}^*_{H,n} \xrightarrow{0.111} \widehat{U}^*_{S,n}, \tag{19}$$

$$\widehat{U}^*_{S,n} \xrightarrow{-0.139} \widehat{R}^*_{L,n}, \tag{20}$$

$$\widehat{R}^*_{L,n-1} \xrightarrow{0.153} \widehat{R}^*_{L,n}, \tag{21}$$

$$\widehat{R}^*_{L,n} \xrightarrow{0.180} \widehat{U}^*_{R,n}, \tag{22}$$

where weight of each relationship is printed above the arrows.

Comparable relationships between switching variables have been identified and discussed in physics-based models and simulations as well as in experimental studies involving various materials (Ielmini, 2011; Nardi et al., 2011, 2012; Nishi et al., 2015; Kim et al., 2016a,b; La Torre et al., 2016). According to relation 19, larger starting HRS values tend to contribute to a higher SET voltage, which is a well known effect due to a reduced driving force for ionic motion at a given applied voltage, as a larger HRS gives both reduced power dissipation as well as a reduced electric field in a thicker insulating gap. The subsequent LRS is strongly affected by the SET voltage (relation 20). This can be attributed to the runaway nature of the SET transition and a higher voltage initial condition, and is also connected

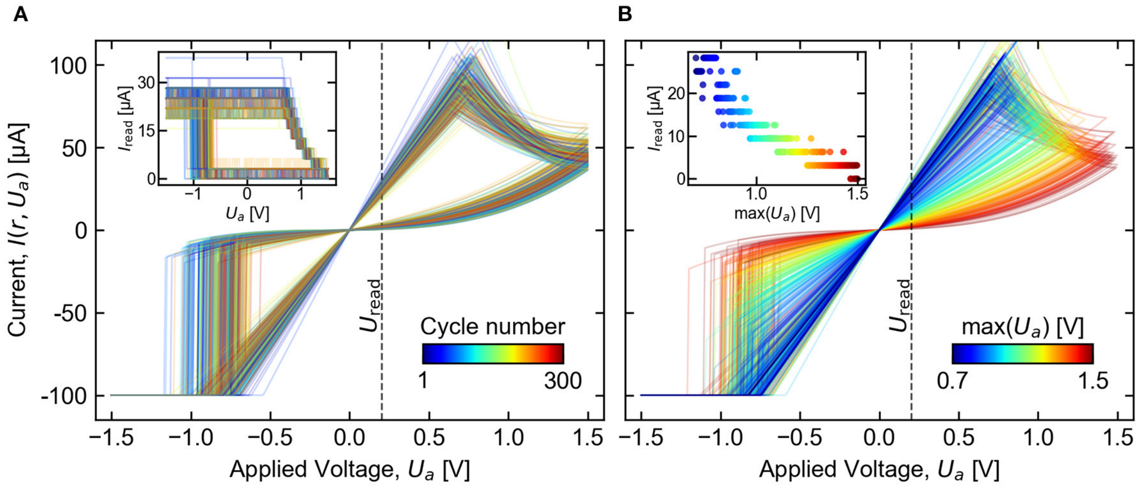

FIGURE 13
Two example simulations involving repeated cycling of a single device. Voltage pulse sequences were applied with varying amplitude following a triangular envelope, and the (I, U) characteristic of each cycle is plotted in a different color. Subplot **(A)** shows 300 consecutive cycles between the full voltage range ±1.5 V, with a readout performed after every pulse (inset). Subplot **(B)** demonstrates multilevel capability with 300 cycles between −1.5 V and maximum voltage that increases each cycle, from 0.7 V to 1.5 V. Readouts following each cycle are shown in the inset. In each case, readouts were simulated using a fixed $U_{read} = 200$ mV, including noise and 4-bit quantization between $I_{min} = 0$ µA and $I_{max} = 40$ µA.

with the dynamics of the current limiting circuitry (Hennen et al., 2021). The LRS value is also strongly correlated with the value of the previous LRS (relation 21), because of the influence of the residual filamentary structure from the previous cycle (Piccolboni et al., 2015). Lastly, relation 22 indicates that higher LRS values tend to have larger reset voltages, which has to do with a balance of factors influencing filament dissolution, including temperature and drift. This balance depends on the cell materials, operating regime, and internal series resistance (Ielmini et al., 2011).

Benchmarks

As a benchmark of the throughput of write operations, repeated resistance cycling was induced on arrays of simulated cells under varying conditions. In each case, voltage pulse sequences to be applied to all defined cells were generated prior to the benchmarks, consisting of amplitudes ±1.5 V with alternating polarity. Defined as such, every pulse drives each cell through a transition into its next HRS or LRS. The read operation was benchmarked separately under equivalent conditions, reading out the entire array using a fixed readout voltage of $U_{read} = 0.2\ V$.

The CPU benchmark was performed using an Intel Xeon Silver 4116 CPU, varying the cell array size M, the order of the VAR process p, as well as the number of threads used to perform the operations in parallel. The resulting read/write throughputs are summarized in Figure 15. Write throughputs up to 2×10^8 operations per second (OPS) were obtained, which is equivalent to 5 ns per individual write operation. Read operations were nearly an order of magnitude faster than writes, with up to 10^9 OPS or 1 ns per read operation. Due to the size of necessary matrix multiplications, increasing the VAR order p incurs a cost of write throughput, with a $p = 100$ model running approximately 4× slower than one with $p = 10$. The read operation, in contrast, shows a negligible dependence on the VAR order.

The GPU accelerated version was benchmarked in an analogous way, using the same host machine with an NVIDIA TITAN RTX GPU device. The results are shown in dependence of the cell array size M in Figures 15B,D. The GPU implementation overtakes the CPU above $M = 10^6$ parallel operations where the entire array is updated, and achieves 2× faster updates and 5× faster readouts for large arrays with $M > 10^7$. However, CPU throughput is applicable to subsets of the array, and may retain an advantage for sparse operations.

Discussion

In order to assess the potential of emerging synaptic devices, new lightweight and accurate device models are needed to constitute the millions/billions of weights used in modern machine learning (ML) models. Candidate memory cells such as ReRAM are highly non-linear stochastic devices with complex internal states and history dependence, all of which needs to be explicitly taken into account. In this article we introduced an efficient generative model for large synaptic arrays, which closely reproduces the statistical behavior of real devices.

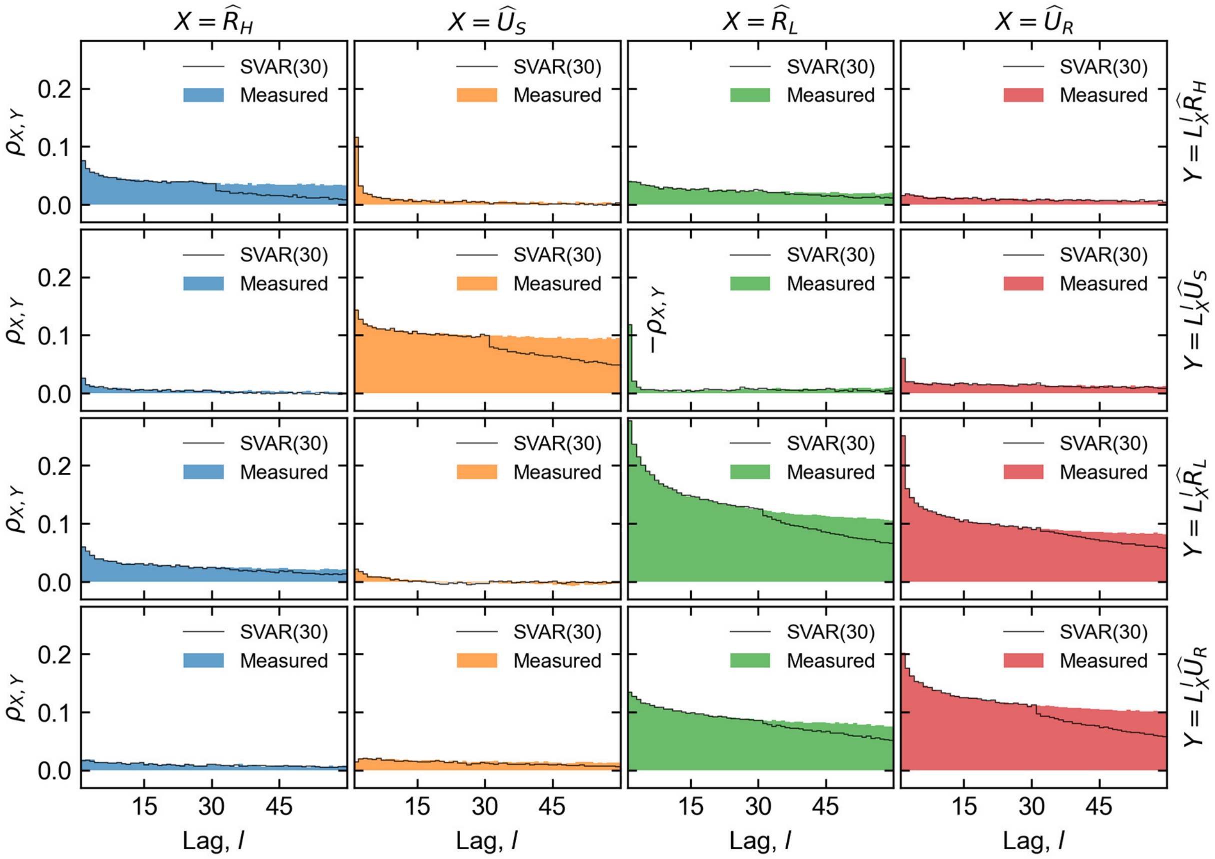

FIGURE 14
Auto- and cross-correlations of the normalized feature vector components, showing the Pearson coefficients $\rho_{X,Y}$ of the variables specified in the subplot columns X and rows Y as a function of lag l. Row variables are lagged with respect to column variables, as denoted by the lag operators L_X. A comparison between measurement data and data generated from SVAR(30) shows extremely close agreement up to cycle range 30. For lags larger than the chosen model order, some of the correlations of $\widehat{x}^*$ decay more quickly than $\widehat{x}$.

Taking advantage of a recently developed electrical measurement technique (Hennen et al., 2021), we systematically fit the model to a dataset that is dense in relevant information about the device state evolution. Together with this new kind of measurement, our modeling approach helps complete a neuromorphic design feedback loop by defining a programmatic connection from the measured behavior of a fabricated device under the intended operating conditions directly to fitted model parameters. Probability density transformation of the underlying SVAR stochastic process gives the model the power to accurately reproduce nearly arbitrary distribution shapes and covariance structures across the switching cycles and across the separate devices. These features enable evaluation of network performance while automatically adapting to a wide variety of possible future device designs.

We provide parallelized implementations for both CPU and GPU, where up to 15 million cells per GB of available memory can be simulated at once. Benchmarks show throughputs above three hundred million weight updates per second, which exceeds the pixel rate of a 30 frames per second video stream at 4K resolution (3,840 × 2,160 pixels). Realistic current readouts including digitization and noise were also benchmarked, and are approximately an order of magnitude faster than weight updates. While speeds can be expected to improve with future optimizations, these benchmarks give a basis for estimating the scope of applicability of the model to ML tasks.

The implementation and the general concept of this model are naturally extendable. Although model parameters were adapted here to a specific HfO_2-based ReRAM device, the method is applicable to a variety of other types of stochastic memory cells such as PCM, MRAM, etc. Four specific switching features were chosen in this demonstration to reconstruct (I, U) cycling behavior, but additional switching parameters can also be extracted from measurements and accommodated within this framework. Ideally informed by statistical measurement data, different functional forms, transition behaviors, time dependence, and underlying stochastic processes can each be

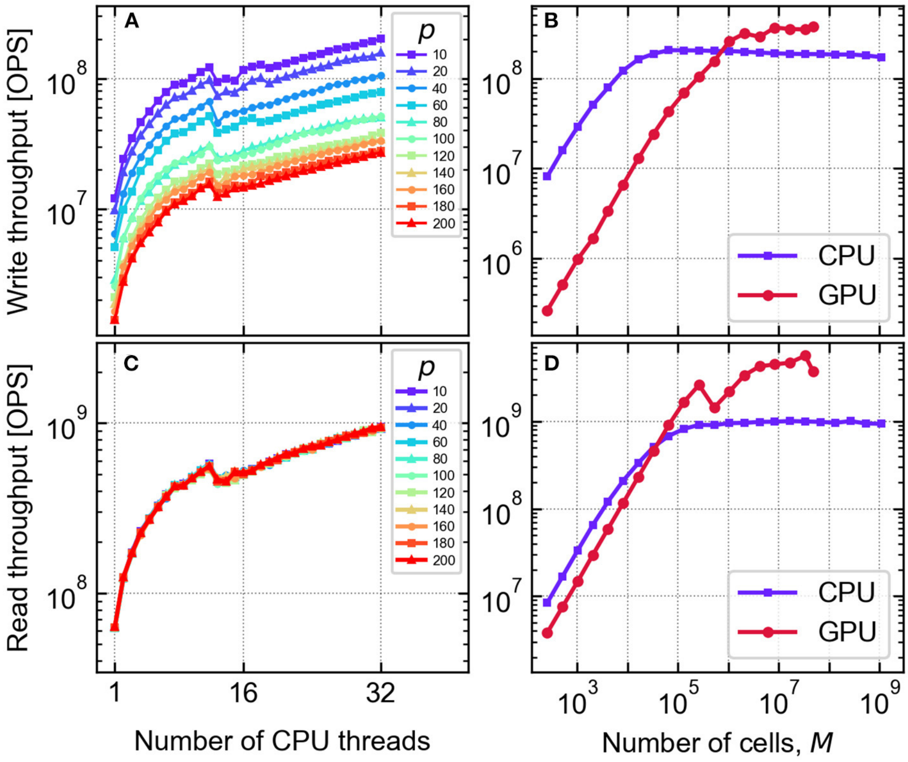

FIGURE 15
Benchmarks of the read/write operation throughput per cell of the Julia model implementations. In **(A,C)**, an array of 2^{20} ($\approx 10^6$) cells are simulated on the CPU as a function of number of parallel threads spawned, and the VAR model order p. In **(B,D)**, the CPU (32 threads) and GPU implementations are benchmarked vs. the cell array size M, with $p = 10$.

substituted. Fitting may also be performed with respect to the output of physics-based simulations, thereby establishing an indirect link to physical parameters while achieving much higher computational speed. With these considerations, the model represents a flexible foundation for implementing large-scale neuromorphic simulations that incorporate realistic device behavior.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author. A Julia implementation of the model is available on GitHub (https://github.com/thennen/StochasticSynapses.jl) and archived in Zenodo (https://doi.org/10.5281/zenodo.6535411).

Author contributions

TH performed the data analysis, implemented the model, and wrote the manuscript. AE carried out the (I, U) measurement. JN and GM fabricated the ReRAM devices. RW and DW co-advised the project. DB conceived of the concept and was in charge of the project. All authors contributed to the article and approved the submitted version.

Acknowledgments

The authors thank Thomas Pössinger of RWTH Aachen for illustrating Figures 1, 2.

Conflict of interest

Author GM is employed by Weebit Nano Ltd.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Abbaspour, E., Menzel, S., and Jungemann, C. (2020). Studying the switching variability in redox-based resistive switching devices. *J. Comput. Electron*. 19, 1426–1432. doi: 10.1007/s10825-020-01537-y

Ambrogio, S., Balatti, S., Milo, V., Carboni, R., Wang, Z.-Q., Calderoni, A., et al. (2016). Neuromorphic learning and recognition with one-transistor-one-resistor synapses and bistable metal oxide RRAM. *IEEE Trans. Electron Devices* 63, 1508–1515. doi: 10.1109/TED.2016.2526647

Ascoli, A., Tetzlaff, R., Biolek, Z., Kolka, Z., Biolkova, V., and Biolek, D. (2015). The art of finding accurate memristor model solutions. *IEEE J. Emerg. Sel. Top. Circ. Syst*. 5, 133–142. doi: 10.1109/JETCAS.2015.2426493

Bengel, C., Siemon, A., Cuppers, F., Hoffmann-Eifert, S., Hardtdegen, A., von Witzleben, M., et al. (2020). Variability-aware modeling of filamentary oxide-based bipolar resistive switching cells using SPICE level compact models. *IEEE Trans. Circuits Syst. Regul. Pap*. 67, 4616–4630. doi: 10.1109/TCSI.2020.3018502

Besard, T., Foket, C., and De Sutter, B. (2019). Effective extensible programming: unleashing julia on GPUs. *IEEE Trans. Parallel Distrib. Syst*. 30, 827–841. doi: 10.1109/TPDS.2018.2872064

Bocquet, M., Aziza, H., Zhao, W., Zhang, Y., Onkaraiah, S., Muller, C., et al. (2014). Compact modeling solutions for oxide-based resistive switching memories (OxRAM). *J. Low Power Electron. Appl*. 4, 1–14. doi: 10.3390/jlpea4010001

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., et al. (2020). "Language models are few-shot learners," in *Advances in Neural Information Processing Systems, Vol. 33*, eds H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Curran Associates, Inc.), 1877–1901.

Burr, G. W., Shelby, R. M., Sebastian, A., Kim, S., Kim, S., Sidler, S., et al. (2017). Neuromorphic computing using non-volatile memory. *Adv. Phys. X* 2, 89–124. doi: 10.1080/23746149.2016.1259585

Butcher, B., Bersuker, G., Young-Fisher, K. G., Gilmer, D. C., Kalantarian, A., Nishi, Y., et al. (2012). "Hot forming to improve memory window and uniformity of low-power HfOx-based RRAMs," in *2012 4th IEEE International Memory Workshop* (Milan: IEEE), 1–4.

Cario, M. C., and Nelson, B. L. (1996). Autoregressive to anything: time-series input processes for simulation. *Operat. Res. Lett*. 19, 51–58. doi: 10.1016/0167-6377(96)00017-X

Chen, A. (2015). Utilizing the variability of resistive random access memory to implement reconfigurable physical unclonable functions. *IEEE Electron. Device Lett*. 36, 138–140. doi: 10.1109/LED.2014.2385870

Chen, A., Hutchby, J., Zhirnov, V. V., and Bourianoff, G. (Eds.). (2014). *Emerging Nanoelectronic Devices*. Chichester: John Wiley & Sons Inc.

Chen, P.-Y., and Yu, S. (2015). Compact modeling of RRAM devices and its applications in 1T1R and 1S1R array design. *IEEE Trans. Electron. Devices* 62, 4022–4028. doi: 10.1109/TED.2015.2492421

Dalgaty, T., Castellani, N., Turck, C., Harabi, K.-E., Querlioz, D., and Vianello, E. (2021). *In situ* learning using intrinsic memristor variability *via* markov chain monte carlo sampling. *Nat. Electron*. 4, 151–161. doi: 10.1038/s41928-020-00523-3

Fantini, A., Gorine, G., Degraeve, R., Goux, L., Chen, C. Y., Redolfi, A., et al. (2015). "Intrinsic program instability in HfO_2 RRAM and consequences on program algorithms," in *2015 IEEE International Electron Devices Meeting (IEDM)* (Washington, DC: IEEE), 7.5.1–7.5.4.

Fantini, A., Goux, L., Degraeve, R., Wouters, D., Raghavan, N., Kar, G., et al. (2013). "Intrinsic switching variability in HfO_2 RRAM," in *2013 5th IEEE International Memory Workshop* (Monterey, CA: IEEE), 30–33.

Hamilton, J. D. (1994). *Time Series Analysis*. Princeton, NJ: Princeton University Press.

Hennen, T. (2022). *StochasticSynapses.jl*. Zenodo. doi: 10.5281/zenodo.6535411

Hennen, T., Wichmann, E., Elias, A., Lille, J., Mosendz, O., Waser, R., et al. (2021). Current-limiting amplifier for high speed measurement of resistive switching data. *Rev. Sci. Instrum*. 92, 054701. doi: 10.1063/5.0047571

Huang, P., Zhu, D., Chen, S., Zhou, Z., Chen, Z., Gao, B., et al. (2017). Compact model of HfO_x-based electronic synaptic devices for neuromorphic computing. *IEEE Trans. Electron. Devices* 64, 614–621. doi: 10.1109/TED.2016.2643162

Ielmini, D. (2011). Modeling the universal set/reset characteristics of bipolar RRAM by field- and temperature-driven filament growth. *IEEE Trans. Electron. Devices* 58, 4309–4317. doi: 10.1109/TED.2011.2167513

Ielmini, D., and Milo, V. (2017). Physics-based modeling approaches of resistive switching devices for memory and in-memory computing applications. *J. Comput. Electron*. 16, 1121–1143. doi: 10.1007/s10825-017-1101-9

Ielmini, D., Nardi, F., and Cagli, C. (2011). Universal reset characteristics of unipolar and bipolar metal-oxide RRAM. *IEEE Trans. Electron. Devices* 58, 3246–3253. doi: 10.1109/TED.2011.2161088

Jiang, H., and Stewart, D. A. (2017). Using dopants to tune oxygen vacancy formation in transition metal oxide resistive memory. *ACS Appl. Mater. Interfaces* 9, 16296–16304. doi: 10.1021/acsami.7b00139

Jiang, Z., Wu, Y., Yu, S., Yang, L., Song, K., Karim, Z., et al. (2016). A compact model for metal-oxide resistive random access memory with experiment verification. *IEEE Trans. Electron. Devices* 63, 1884–1892. doi: 10.1109/TED.2016.2545412

Kantorovich, L. V. (1960). Mathematical methods of organizing and planning production. *Manag. Sci*. 6, 366–422. doi: 10.1287/mnsc.6.4.366

Kim, K. M., Yang, J. J., Strachan, J. P., Grafals, E. M., Ge, N., Melendez, N. D., et al. (2016a). Voltage divider effect for the improvement of variability and endurance of TaO_x memristor. *Sci. Rep* 6, 20085. doi: 10.1038/srep20085

Kim, S., Lim, M., Kim, Y., Kim, H.-D., and Choi, S.-J. (2018). Impact of synaptic device variations on pattern recognition accuracy in a hardware neural network. *Sci. Rep*. 8, 2638. doi: 10.1038/s41598-018-21057-x

Kim, W., Menzel, S., Wouters, D. J., Guo, Y., Robertson, J., Roesgen, B., et al. (2016b). Impact of oxygen exchange reaction at the ohmic interface in Ta_2O_5-based ReRAM devices. *Nanoscale* 8, 17774–17781. doi: 10.1039/C6NR03810G

Kopperberg, N., Wiefels, S., Liberda, S., Waser, R., and Menzel, S. (2021). A consistent model for short-term instability and long-term retention in filamentary oxide-based memristive devices. *ACS Appl. Mater. Interfaces* 13, 58066–58075. doi: 10.1021/acsami.1c14667

La Torre, C., Fleck, K., Starschich, S., Linn, E., Waser, R., and Menzel, S. (2016). Dependence of the SET switching variability on the initial state in HfO_x-based ReRAM. *Phys. Status Solidi A* 213, 316–319. doi: 10.1002/pssa.201532375

Li, H., Huang, P., Gao, B., Liu, X., Kang, J., and Philip Wong, H.-S. (2017). Device and circuit interaction analysis of stochastic behaviors in cross-point RRAM arrays. *IEEE Trans. Electron. Devices* 64, 4928–4936. doi: 10.1109/TED.2017.2766046

Liu, H., Bedau, D., Sun, J., Mangin, S., Fullerton, E., Katine, J., et al. (2014). Dynamics of spin torque switching in all-perpendicular spin valve nanopillars. *J. Magn. Magn. Mater*. 358–359, 233–258. doi: 10.1016/j.jmmm.2014.01.061

Lütkepohl, H. (2005). *New Introduction to Multiple Time Series Analysis*. New York, NY; Berlin: Springer.

Ma, W., Chiu, P.-F., Choi, W. H., Qin, M., Bedau, D., and Lueker-Boden, M. (2019). "Non-volatile memory array based quantization- and noise-resilient LSTM neural networks," in *2019 IEEE International Conference on Rebooting Computing (ICRC)* (San Mateo, CA: IEEE), 1–9.

Maria Puglisi, F., Larcher, L., Padovani, A., and Pavan, P. (2015). Bipolar resistive RAM based on HfO_2: physics, compact modeling, and variability control. *IEEE J. Emerg. Sel. Top. Circ. Syst*. 6, 171–184. doi: 10.1109/JETCAS.2016.2547703

Mayer, J., Khairy, K., and Howard, J. (2010). Drawing an elephant with four complex parameters. *Am. J. Phys*. 78, 648–649. doi: 10.1119/1.3254017

Menzel, S., Böttger, U., Wimmer, M., and Salinga, M. (2015). Physics of the switching kinetics in resistive memories. *Adv. Funct. Mater*. 25, 6306–6325. doi: 10.1002/adfm.201500825

Messaris, I., Serb, A., Stathopoulos, S., Khiat, A., Nikolaidis, S., and Prodromakis, T. (2018). A data-driven verilog-A ReRAM model. *IEEE Trans. Comput. Aided Des. Integr. Circ. Syst*. 37, 3151–3162. doi: 10.1109/TCAD.2018.2791468

Milo, V., Malavena, G., Monzio Compagnoni, C., and Ielmini, D. (2020). Memristive and CMOS devices for neuromorphic computing. *Materials* 13, 166. doi: 10.3390/ma13010166

Moon, J., Ma, W., Shin, J. H., Cai, F., Du, C., Lee, S. H., et al. (2019). Temporal data classification and forecasting using a memristor-based reservoir computing system. *Nat. Electron*. 2, 480–487. doi: 10.1038/s41928-019-0313-3

Nail, C., Molas, G., Blaise, P., Piccolboni, G., Sklenard, B., Cagli, C., et al. (2016). "Understanding RRAM endurance, retention and window margin trade-off using experimental results and simulations," in *2016 IEEE International Electron Devices Meeting (IEDM)* (San Francisco, CA: IEEE), 4.5.1–4.5.4.

Nardi, F., Ielmini, D., Cagli, C., Spiga, S., Fanciulli, M., Goux, L., et al. (2011). Control of filament size and reduction of reset current below 10μA in NiO resistance switching memories. *Solid State Electron*. 58, 42–47. doi: 10.1016/j.sse.2010.11.031

Nardi, F., Larentis, S., Balatti, S., Gilmer, D. C., and Ielmini, D. (2012). Resistive switching by voltage-driven ion migration in bipolar RRAM-Part I: experimental study. *IEEE Trans. Electron. Devices* 59, 2461–2467. doi: 10.1109/TED.2012.2202319

Nishi, Y., Fleck, K., Böttger, U., Waser, R., and Menzel, S. (2015). Effect of RESET voltage on distribution of SET switching time of bipolar resistive switching in a tantalum oxide thin film. *IEEE Trans. Electron. Devices* 62, 1561–1567. doi: 10.1109/TED.2015.2411748

Park, S., Noh, J., Choo, M.-,l., Sheri, A. M., Chang, M., Kim, Y.-B., et al. (2013). Nanoscale RRAM-based synaptic electronics: toward a neuromorphic computing device. *Nanotechnology* 24, 384009. doi: 10.1088/0957-4484/24/38/384009

Pedroni, B. U., Deiss, S. R., Mysore, N., and Cauwenberghs, G. (2020). "Design principles of large-scale neuromorphic systems centered on high bandwidth memory," in *2020 International Conference on Rebooting Computing (ICRC)* (Atlanta, GA: IEEE), 90–94.

Pedroni, B. U., Joshi, S., Deiss, S. R., Sheik, S., Detorakis, G., Paul, S., et al. (2019). Memory-efficient synaptic connectivity for spike-timing- dependent plasticity. *Front. Neurosci*. 13, 357. doi: 10.3389/fnins.2019.00357

Piccolboni, G., Molas, G., Portal, J. M., Coquand, R., Bocquet, M., Garbin, D., et al. (2015). "Investigation of the potentialities of vertical resistive RAM (VRRAM) for neuromorphic applications," in *2015 IEEE International Electron Devices Meeting (IEDM)* (Washington, DC: IEEE), 17.2.1–17.2.4.

Reuben, J., Fey, D., and Wenger, C. (2019). A modeling methodology for resistive RAM based on stanford-PKU model with extended multilevel capability. *IEEE Trans. Nanotechnol*. 18, 647–656. doi: 10.1109/TNANO.2019.2922838

Rezende, D. J., and Mohamed, S. (2015). "Variational inference with normalizing flows," in *Proceedings of the 32nd International Conference on Machine Learning (PMLR)* (Lille), *Vol*. 37, 1530–1538.

Roldán, J. B., Alonso, F. J., Aguilera, A. M., Maldonado, D., and Lanza, M. (2019). Time series statistical analysis: a powerful tool to evaluate the variability of resistive switching memories. *J. Appl. Phys*. 125, 174504. doi: 10.1063/1.5079409

Sangwan, V. K., and Hersam, M. C. (2020). Neuromorphic nanoelectronic materials. *Nat. Nanotechnol*. 15, 517–528. doi: 10.1038/s41565-020-0647-z

Seabold, S., and Perktold, J. (2010). "Statsmodels: econometric and statistical modeling with Python," in *Python in Science Conference* (Austin, TX), 92–96.

Siemon, A., Wouters, D., Hamdioui, S., and Menzel, S. (2019). "Memristive device modeling and circuit design exploration for computation-in-memory," in *2019 IEEE International Symposium on Circuits and Systems (ISCAS)* (Sapporo: IEEE), 1–5.

Stewart, D. A. (2019). Diffusion of oxygen in amorphous tantalum oxide. *Phys. Rev. Mater*. 3, 055605. doi: 10.1103/PhysRevMaterials.3.055605

Wald, N., and Kvatinsky, S. (2019). Understanding the influence of device, circuit and environmental variations on real processing in memristive memory using Memristor Aided Logic. *Microelectron. J*. 86:22–33. doi: 10.1016/j.mejo.2019.02.013

Waser, R., Dittmann, R., Staikov, G., and Szot, K. (2009). Redox-based resistive switching memories - nanoionic mechanisms, prospects, and challenges. *Adv. Mater*. 21, 2632–2663. doi: 10.1002/adma.200900375

Wiefels, S., Bengel, C., Kopperberg, N., Zhang, K., Waser, R., and Menzel, S. (2020). HRS instability in oxide-based bipolar resistive switching cells. *IEEE Trans. Electron. Devices* 67, 4208–4215. doi: 10.1109/TED.2020.3018096

Yang, J. J., Pickett, M. D., Li, X., Ohlberg, D. A. A., Stewart, D. R., and Williams, R. S. (2008). Memristive switching mechanism for metal/oxide/metal nanodevices. *Nat. Nanotech* 3, 429–433. doi: 10.1038/nnano.2008.160

You, Z., and Ramanathan, S. (2015). Mott memory and neuromorphic devices. *Proc. IEEE* 103, 1289–1310. doi: 10.1109/JPROC.2015.2431914

Yu, S., and Chen, P.-Y. (2016). Emerging memory technologies: recent trends and prospects. *IEEE Solid State Circ. Mag*. 8, 43–56. doi: 10.1109/MSSC.2016.2546199

Zhao, L., Chen, H.-Y., Wu, S.-C., Jiang, Z., Yu, S., Hou, T.-H., et al. (2014). Multi-level control of conductive nano-filament evolution in HfO_2 ReRAM by pulse-train operations. *Nanoscale* 6, 5698–5702. doi: 10.1039/C4NR00500G