Modeling SDC Memristors Using Artificial Neural Networks

Karol Bednarz

September 1, 2025

Contents

1 Introduction

Numerous memristor models have been proposed in the scientific literature following the discovery of memristive behavior at the nanoscale. The original model, introduced in [8], conceptualizes the memristor as a series connection of two variable resistances corresponding to the conductive and insulating regions of the thin film. To more accurately model Self-Directed Channel (SDC) memristors, M. Nugent and T. Molter proposed the generalized *Mean Metastable Switch* (MMS) model, which is a semi-empirical formulation. In this approach, the time derivative of the internal state variable is defined as a function of both the transition probabilities between metastable states and the current value of the state variable [6]. In [5], the authors attempted to model memristive behavior using the *Physics-Informed Neural Networks* (PINNs) framework. Although the reported results indicate the potential of this method, the study is not grounded in experimental measurements of physical memristor devices. Instead, the analysis is limited to comparisons with the outcomes of existing simulation models. Moreover, the training data employed during the neural network learning process were generated based on previously developed theoretical models, thereby limiting the ability to assess the method's accuracy in the context of real-world physical systems.

In [3], the authors introduce the concept of deep neural models in which the dynamics of the hidden state are governed by an *ordinary differential equation* (ODE). The training process is performed in an end-to-end manner, meaning that all parameters are optimized simultaneously within a single training routine. A key innovation of the proposed approach is a novel backpropagation technique, which relies on solving the corresponding adjoint ODE backward in time using *adjoint*

sensitivity methods. This formulation enables efficient gradient computation and facilitates the application of neural ODEs in various architectures, including continuous-depth residual networks and generative flow-based models.

In [2], the authors extend the classical Neural ODE framework to enable the modeling of discontinuous events in continuous time—without requiring prior knowledge of the number or timing of such events. The proposed differentiable event functions allow for efficient simulation of hybrid systems, such as systems with collisions, state-switching mechanisms, or point processes. This development opens up new possibilities for modeling and training systems with discrete control inputs and non-smooth dynamics.

TODO: Add more information about the referenced work, including applications and results.

The present study aims to investigate the feasibility of modeling Self-Directed Channel (SDC) memristors using artificial neural networks, and to compare the effectiveness of this approach with that of existing theoretical models. Specifically, we focus on the application of Neural Ordinary Differential Equation (Neural ODE) models to simulate the behavior of SDC memristors, utilizing experimental data obtained from real physical systems.

Our objective is to assess whether neural network-based models can accurately reproduce the dynamic characteristics of SDC memristors, and to determine whether they offer any advantages over traditional theoretical approaches—particularly in terms of modeling flexibility, accuracy, and applicability to real-world, measurement-driven scenarios.

2 Materials and Methods

2.1 Memristors and Measurement Procedure

This study utilizes experimental data obtained from physical Self-Directed Channel (SDC) memristors doped with tungsten. The structure and properties of these devices have been described in detail in the literature, e.g., in [4, 1].

The experimental setup, illustrated in Fig. 1, consisted of an SDC memristor connected in series with a resistor. The sinusoidal input voltage was supplied by an arbitrary waveform generator, while the voltage signals were measured using a data acquisition (DAQ) system.

Measurements were carried out for various combinations of supply voltage amplitudes and frequencies. Specifically, the amplitude of the applied voltage was varied as $V_s \in \{0.5,\ 1.0,\ 1.5\}$ [V], and the frequency was selected from the set $f \in \{1,\ 5,\ 10,\ 20,\ 50,\ 100\}$ [Hz].

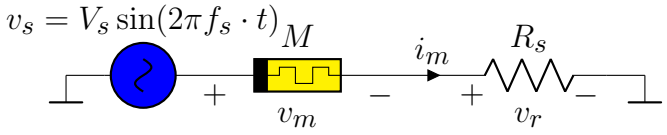


Figure 1: Schematic diagram of the measurement setup used for the SDC memristor. The device is connected in series with a resistor, and the input voltage is applied via an arbitrary waveform generator. Voltage measurements are acquired using a data acquisition (DAQ) system.

2.2 Modeling with Neural Networks

The analyzed model is based on a general description of a memristive device, formulated using the equations (1) and (2).

$$v(t) = \mathcal{M}\Big(x(t), v(t)\Big) i(t), \tag{1}$$

$$\frac{\mathrm{d}x}{\mathrm{d}t} = f\Big(x(t), v(t)\Big) \tag{2}$$

where $x \in \langle 0, 1\rangle$ denotes the internal state variable, whose evolution follows a specified dynamic function f, and $\mathcal{M}$ represents the memristance of the device. A central challenge in memristor modeling is the proper identification of the functions $\mathcal{M}(x, v)$ and $f(x, v)$.

As the resistance of self-directed channel (SDC) memristors varies due to the evolution of Ag^+ ion channels and the dynamic formation of conductive pathways, the device conductance exhibits a continuous transition between distinct resistance states. The memristor conductance can be mathematically described as a weighted combination of the extreme conductance values, governed by the internal state variable. The memristor conductance $G_m(x) = \mathcal{M}(x)^{-1}$ is defined as:

$$\mathcal{G}_m(x) = \frac{x}{R_{\mathrm{ON}}} + \frac{1 - x}{R_{\mathrm{OFF}}} \tag{3}$$

where R_{ON} and R_{OFF} correspond to the resistance values in the low-conductance and high-conductance states, respectively.

The function $f(x(t), v(t))$ was implemented using an artificial neural network, whose conceptual architecture is illustrated in Fig. 2. The neural network consists of interconnected nodes (neurons) that process information through weighted connections. Each neuron in the network computes its output according to:

$$y = \sigma\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{4}$$

where w_i are the weights that determine the strength and importance of each input connection x_i, b is the bias term that provides an adjustable threshold for neuron activation, and σ is the activation function (e.g., sigmoid, ReLU, or tanh) that introduces non-linearity into the model.

The weights represent the learnable parameters that encode the relationship between inputs and outputs, while biases allow neurons to shift their activation threshold, enabling the network to better fit the training data even when all inputs are zero. During the optimization process, the network parameters (weights and biases) are tuned jointly with the values of R_{ON} and R_{OFF}.

The learning process involves iteratively adjusting the network parameters to minimize a loss function that quantifies the difference between predicted and target outputs. This is achieved through backpropagation algorithm combined with gradient descent optimization, where gradients of the loss function with respect to each parameter are computed and used to update the weights and biases in the direction that reduces the overall error.

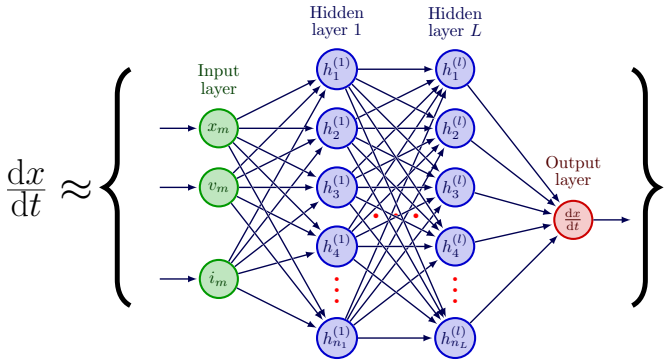


Figure 2: Conceptual diagram illustrating the use of a neural network to simulate the dynamics of the memristor.

The neural network model development and training pipeline was implemented using the `PyTorch` deep learning framework within the `Python` programming environment, leveraging its automatic differentiation capabilities and GPU acceleration for efficient computational processing.

To enable seamless integration of ordinary differential equations (ODEs) within the neural network training paradigm, the specialized `torchdiffeq` library was employed. This library provides differentiable ODE solvers that facilitate efficient gradient computation with respect to solution trajectories through the adjoint sensitivity method, well described in [3]. This approach enables end-to-end training of neural differential equation models by maintaining gradient flow through the numerical integration process, which is essential for learning dynamic systems governed by differential equations.

Specifically, the `dopri5` (Dormand-Prince 5th order) adaptive Runge-Kutta solver was utilized for its superior balance between computational efficiency and numerical accuracy. This solver employs embedded error estimation for adaptive step-size control, ensuring stable integration of the memristor dynamics while maintaining computational tractability during training.

The neural network architecture is illustrated in Figure 3 and incorporates a strategically designed bottleneck configuration with dimensional two time reduction in the initial and final hidden layers. This architectural choice addresses several critical aspects of neural ODE training and memristor system modeling, like:

- Dimensional reduction at the start and end of the network significantly decreases the total parameter count,
- Parameter count reduction contributes to more stable optimization dynamics by limiting the search space complexity
- Dimensional reduction helps mitigate gradient vanishing and explosion problems that commonly arise in deep networks processing long temporal sequences
- The bottleneck structure implicitly enforces a low-dimensional manifold assumption consistent with the underlying physics of memristor dynamics

The network weights were optimized using the `Adam` (Adaptive Moment Estimation) optimizer, which represents one of the most robust and widely adopted optimization algorithms in deep learning applications. Adam combines the advantages of momentum-based methods with adaptive learning rate adjustment, featuring:

- Efficient handling of sparse gradients
- Fast convergence properties, making it suitable for training deep neural networks with complex architectures and large parameter spaces.

- Minimal hyperparameter tuning requirements

To achieve adaptive learning rate control throughout the training process, the `ReduceLROnPlateau` scheduling strategy was implemented. This performance-based approach dynamically modulates the learning rate in response to validation metric plateaus, specifically monitoring training loss convergence behavior. The scheduler operates according to the following principles:

This implementation strategy ensures robust and efficient training of neural differential equation models for memristor circuit simulation, combining state-of-the-art deep learning methodologies with specialized techniques for dynamic system modeling.

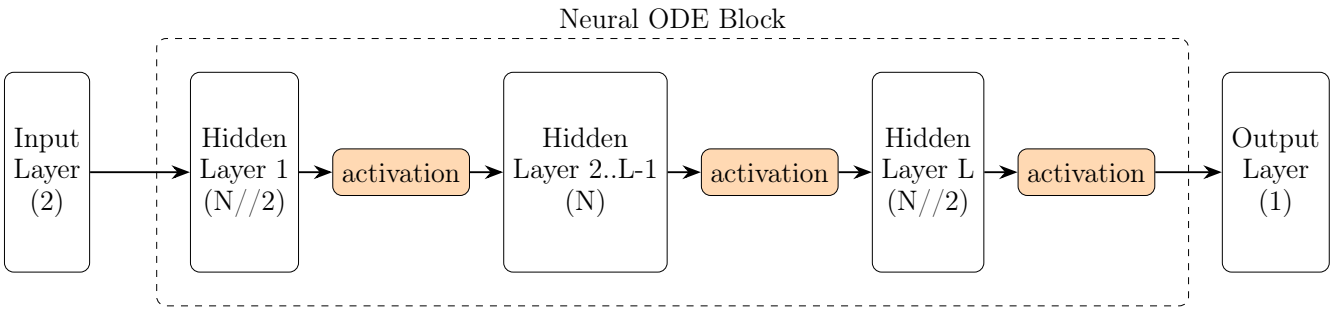


Figure 3: Architecture of the feedforward neural network used within the Neural ODE framework for memristor dynamics modeling. The network consists of an input layer receiving two inputs, multiple hidden layers with decreasing-increasing-decreasing neuron counts (N//2, N, N//2), activation functions between layers, and a single output. The Neural ODE block encompasses the core computational layers that approximate the derivative function $f(x(t), v(t))$ in the differential equation system.

2.3 MMS Model — Mean Metastable Switch

To evaluate the effectiveness of the proposed model, it was compared with the deterministic Mean Metastable Switch (MMS) model introduced in [6, 7]. This model describes the memristor dynamics using Eq. (5).

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \frac{1}{\tau}\left(\frac{1}{1+e^{-\beta\left(v(t)-V_{\mathrm{ON}}\right)}}(1-x) - \left(1 - \frac{1}{1+e^{-\beta\left(v(t)-V_{\mathrm{OFF}}\right)}}\right)x\right) \tag{5}$$

where $\beta = \frac{q}{kT}$ is a temperature-related parameter, V_{ON} denotes the switching voltage to the low-resistance state (LRS), V_{OFF} denotes the switching voltage to the high-resistance state (HRS), k is the Boltzmann constant, T is the absolute temperature, and q represents the elementary charge. The instantaneous conductance of the memristor is defined as in Eq. (3).

Other conductance functions reported in the literature include, among others:

- $\mathcal{G}_m(x) = \left(xR_{\mathrm{ON}} + (1-x)R_{\mathrm{OFF}}\right)^{-1}$
- $\mathcal{G}_m(x) = \frac{1}{R_{\mathrm{ON}}}\left(\frac{R_{\mathrm{OFF}}}{R_{\mathrm{ON}}}\right)^{x-1}$

The influence of these functions on the memristor resistance is illustrated in Fig. 4.

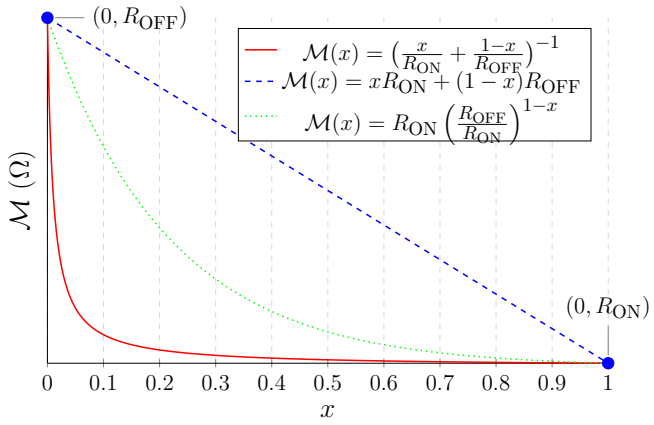


Figure 4: Comparison of the memristor resistance functions $R_m(x)$ for different memristor models.

In [6], the authors also proposed the Generalized Mean Metastable Switch (GMMS) model, which represents the memristor as a parallel connection of two memristors along with a Schottky diode. This model is more complex and accounts for the nonlinear current-voltage characteristics, allowing for a more accurate representation of the real memristor behavior under various operating conditions.

However, the introduction of additional parameters and nonlinearities makes this model more complicated and challenging to optimize. The derivative of the state variable corresponds to Eq. (5), while the current-voltage relationship is described by the following equations:

$$i(t) = \phi i_m(v,t) + (1-\phi) i_d(v) \tag{6}$$

$$i_d(v) = \alpha_f e^{\beta_f v} - \alpha_r e^{-\beta_r v} \tag{7}$$

$$i_m(v,t) = \mathcal{G}_m(x) v \tag{8}$$

where $\phi \in [0,1]$ is a parameter representing the fraction of the total current flowing through the memristor relative to the entire current through the device. The parameters α_f, β_f, α_r, and β_r are positive constants characterizing the forward and reverse current behavior along the Schottky barrier.

In the case of the GMMS model, an additional challenge arises due to the Schottky effect, which introduces nonlinearity into the current–voltage characteristic. When the memristor is connected in series with a resistor, this requires solving a nonlinear equation of the form Eq. (9) numerically at each time step. As a result, the system of ordinary differential equations is transformed into a differential-algebraic equation (DAE).

$$v_s(t) = v_r(t) + v_m(t) \tag{9}$$

where $v_s(t)$ denotes the supply voltage, $v_r(t) = R_s i(t)$ is the voltage across the resistor, and $v_m(t)$ is the voltage across the memristor.

This formulation renders the system stiff and prevents the straightforward application of automatic differentiation with respect to model parameters in the objective function. An alternative approach to address this is the use of implicit differentiation. Let F be a function defined by the equation:

$$F(v_m, t) = v_s(t) - v_r(v_m, t) - v_m(t) = 0. \tag{10}$$

Applying the chain rule yields:

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \frac{\partial F}{\partial v_m}\frac{\mathrm{d}v_m}{\mathrm{d}t} + \frac{\partial F}{\partial t} = 0, \tag{11}$$

which implies:

$$\frac{\mathrm{d}v_m}{\mathrm{d}t} = -\frac{\partial F/\partial t}{\partial F/\partial v_m}. \tag{12}$$

Substituting the expressions gives:

$$\frac{\mathrm{d}v_m}{\mathrm{d}t} = -\frac{\mathrm{d}v_s}{\mathrm{d}t} \cdot \frac{1}{-R_s\left(\mathcal{G}_m\phi + (1-\phi)\left(\alpha_f\beta_f e^{\beta_f v_m} + \alpha_r\beta_r e^{-\beta_r v_m}\right)\right) - 1}. \tag{13}$$

Thus, the system requires an additional state variable; however, it is not necessary to solve a nonlinear system of equations at each time step. Consequently, the system of differential equations describing the memristor-resistor setup using the GMMS model is as follows:

$$\begin{cases} \dfrac{\mathrm{d}x}{\mathrm{d}t} = \dfrac{1}{\tau}\left(\dfrac{1}{1+e^{-\beta(v_m(t)-v_{\mathrm{on}})}}(1-x) - \left(1 - \dfrac{1}{1+e^{-\beta(v_m(t)-v_{\mathrm{off}})}}\right)x\right), \\ \dfrac{\mathrm{d}v_m}{\mathrm{d}t} = \dfrac{\mathrm{d}v_s}{\mathrm{d}t} \cdot \dfrac{1}{R_s\left(\mathcal{G}_m\phi + (1-\phi)\left(\alpha_f\beta_f e^{\beta_f v_m} + \alpha_r\beta_r e^{-\beta_r v_m}\right)\right) - 1}. \end{cases} \tag{14}$$

2.4 Objective Function

The loss function based on the phase portrait is designed to simultaneously account for both the absolute signal magnitudes and its dynamic structure. To this end, it is defined as the sum of four components, each serving a distinct and complementary role in the optimization process. The total loss function is given by:

$$\mathcal{L}_{\mathrm{total}} = \lambda_1 \cdot \mathcal{L}_{\mathrm{base}} + \lambda_2 \cdot \mathcal{L}_v + \lambda_3 \cdot \mathcal{L}_c + \lambda_4 \cdot \mathcal{L}_{\mathrm{iv}} + \sum_{i=1}^{N_c} \lambda_{\mathrm{con}_i} \cdot \mathcal{L}_{\mathrm{con}_i}(x), \tag{15}$$

Here, $\mathcal{L}_{\mathrm{base}}$ denotes the primary error term, measuring the difference between predicted and reference values, in that case the mean squared error (MSE). The term $\mathcal{L}_v$ enforces accurate modeling of the velocity characteristics by considering the derivatives of trajectories in phase space. The component $\mathcal{L}_c$ promotes consistency of the trajectories in terms of their curvature, for example by analyzing changes in trajectory direction or curvature. Finally, $\mathcal{L}_{\mathrm{con}}(x)$ introduces additional physical or structural constraints that the model is required to satisfy, including conditions of stability, nonlinearities, or conformity to the underlying physical model. The weighting coefficients λ_1, λ_2, λ_3, and λ_4 balance the influence of each term within the overall loss function according to the priorities of the modeling task, and each of the values were calculated during optimization using the Geometric Loss Strategy (GLS) with the following algorithm presented in Algorithm 1.

Algorithm 1 Adaptive Loss Balancing with Clamping

Require: Loss terms $L_{\text{base}}, L_{\text{vel}}, L_{\text{curv}}, L_{\text{iv}}$, small constant $\varepsilon > 0$, clamp bounds $[w_{\min}, w_{\max}]$
Ensure: Adaptive weights $w_{\text{base}}, w_{\text{vel}}, w_{\text{curv}}, w_{\text{iv}}$

1: Collect losses:

$$\mathbf{L} \leftarrow [L_{\text{base}}, L_{\text{vel}}, L_{\text{curv}}, L_{\text{iv}}]$$

2: Compute geometric mean:

$$\overline{L}_{\text{geo}} \leftarrow \exp\left(\frac{1}{|\mathbf{L}|}\sum_{i=1}^{|\mathbf{L}|}\ln(L_i + \varepsilon)\right)$$

3: **for** each loss $L_i \in \mathbf{L}$ **do**
4: Compute raw weight:

$$w_i \leftarrow \frac{\overline{L}_{\text{geo}}}{L_i + \varepsilon}$$

5: Clamp weight:

$$w_i \leftarrow \min\big(\max(w_i,\, w_{\min}),\, w_{\max}\big)$$

6: **end for**
7: Return weights $(w_{\text{base}}, w_{\text{vel}}, w_{\text{curv}}, w_{\text{iv}})$

For circuit trajectory datasets indexed by $k = 1, \ldots, N_{\text{traj}}$, a trajectory-specific standardization procedure is implemented to normalize each electrical variable $z \in \{v, i\}$ (representing voltage and current measurements, respectively). This normalization process utilizes the standard deviation computed independently for each individual trajectory sequence, ensuring statistical consistency within each temporal dataset.

The trajectory-specific standard deviation is calculated using the unbiased sample estimator:

$$\sigma_{z,k} = \sqrt{\frac{1}{T-1}\sum_{t=1}^{T}\left(z_{k,t}^{\text{true}} - \bar{z}_k^{\text{true}}\right)^2} \tag{16}$$

where the trajectory-specific temporal mean is defined as:

$$\bar{z}_k^{\text{true}} = \frac{1}{T}\sum_{t=1}^{T} z_{k,t}^{\text{true}} \tag{17}$$

The parameter T represents the temporal resolution of the trajectory, corresponding to the total number of discrete time-series sampling points within each measurement sequence.

This trajectory-wise normalization methodology serves several critical analytical objectives in circuit analysis and machine learning applications. Primarily, it establishes scale invariance across electrical signals exhibiting disparate amplitude characteristics, thereby facilitating quantitative comparison of dynamic waveform morphologies independent of absolute magnitude variations. Furthermore, this standardization approach enhances the numerical stability and convergence properties of optimization algorithms employed in neural network training procedures.

By implementing trajectory-specific normalization rather than global standardization, the methodology preserves the inherent temporal dynamics and statistical properties unique to each circuit configuration while simultaneously ensuring that amplitude disparities do not introduce systematic bias during pattern recognition, feature extraction, or predictive modeling procedures. This approach

is particularly advantageous in scenarios involving heterogeneous circuit topologies or varying operational conditions, where maintaining the relative temporal structure of electrical phenomena is paramount for accurate system identification and dynamic analysis. The neural network loss function comprises several specialized components designed to optimize memristor circuit modeling performance across multiple dynamical characteristics. Each component addresses specific aspects of the temporal and nonlinear behavior exhibited by memristive systems.

The fundamental component of the loss function is responsible for model fitting to experimental voltage-current data, incorporating trajectory-specific normalization procedures. This component minimizes the mean squared error (MSE) between predicted and ground-truth voltage and current waveforms, normalized by the standard deviation of each individual trajectory. This normalization ensures scale invariance across datasets with varying amplitude characteristics.The mean squared error for a sequence of n data points is defined in Eq (18):

$$\mathrm{MSE}(\hat{y}, y) = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{18}$$

where $\hat{y}_i$ represents the predicted values and y_i represents the true values.

The base loss is formally expressed as it shown in Eq. (19):

$$\mathcal{L}_{\text{base}} = \frac{1}{N_{\text{traj}}}\sum_{k=1}^{N_{\text{traj}}}\left[\mathrm{MSE}\left(\frac{v_{\text{pred},k}}{\sigma_{v,k}}, \frac{v_{\text{true},k}}{\sigma_{v,k}}\right) + \mathrm{MSE}\left(\frac{i_{\text{pred},k}}{\sigma_{i,k}}, \frac{i_{\text{true},k}}{\sigma_{i,k}}\right)\right] \tag{19}$$

where $v_{\text{pred},k}$ and $i_{\text{pred},k}$ represent the predicted voltage and current for trajectory k, while $v_{\text{true},k}$ and $i_{\text{true},k}$ denote the corresponding experimental measurements.

The first-order dynamics comparison enables evaluation of temporal derivative agreement between model predictions and experimental data, particularly critical for capturing rapid memristor state transitions. This component ensures that the neural network learns not only absolute signal values but also their instantaneous rates of change, which is fundamental for accurate modeling of dynamic systems exhibiting switching phenomena. The first-order loss component is defined as:

$$\mathcal{L}_v = \mathrm{MSE}\left(\frac{\mathrm{d}v}{\mathrm{d}t}_{\text{pred}}, \frac{\mathrm{d}v}{\mathrm{d}t}_{\text{true}}\right) + \mathrm{MSE}\left(\frac{\mathrm{d}i}{\mathrm{d}t}_{\text{pred}}, \frac{\mathrm{d}i}{\mathrm{d}t}_{\text{true}}\right) \tag{20}$$

This formulation captures the velocity characteristics of electrical variables, ensuring that transient behaviors and switching dynamics are accurately reproduced by the model.

The second-order comparison term addresses trajectory curvature characteristics, which are essential for capturing subtle variations in system dynamics that manifest as nonlinear memristor behaviors. This component facilitates superior representation of curvature properties within the phase-space trajectory, particularly important for modeling systems exhibiting complex nonlinear and highly dynamic responses. The curvature loss is formulated as:

$$\mathcal{L}_c = \mathrm{MSE}\left(\frac{\mathrm{d}^2v}{\mathrm{d}t^2}_{\text{pred}}, \frac{\mathrm{d}^2v}{\mathrm{d}t^2}_{\text{true}}\right) \tag{21}$$

By incorporating second-order temporal derivatives, this component ensures that acceleration characteristics and higher-order dynamical features are preserved during the learning process.

To enforce parameter bounds within specified intervals, a sophisticated constraint loss function utilizing smooth sigmoid transitions is implemented. This component prevents parameter drift beyond physically meaningful ranges while maintaining differentiability for gradient-based optimization algorithms. The constraint loss is mathematically expressed as:

$$\mathcal{L}_{\text{con}}(\mathbf{x}) = \mathbb{E}\left[\left(\sigma\left(-\frac{\mathbf{x}-(a+\delta)}{\delta/4}\right)\cdot((a+\delta)-\mathbf{x}) + \sigma\left(\frac{\mathbf{x}-(b-\delta)}{\delta/4}\right)\cdot(\mathbf{x}-(b-\delta))\right)^2\right] \tag{22}$$

where the constituent parameters are defined as follows:

- $\sigma(z) = \frac{1}{1+e^{-z}}$ represents the sigmoid activation function
- a and b denote the lower and upper bounds of the admissible parameter range, respectively
- δ specifies the transition margin (width of the transitional zone between admissible and penalized parameter regions)
- λ_{con} represents the weighting coefficient controlling the constraint enforcement strength
- $\mathbb{E}$ denotes the expectation operator (ensemble average over the parameter set or trajectory collection)

This constraint formulation provides smooth penalty gradients that guide parameters toward the feasible region while avoiding discontinuities that could disrupt the optimization process. The sigmoid-based transitions ensure that the constraint becomes progressively more restrictive as parameters approach the boundary limits, promoting stable convergence behavior.

During neural network training, specific interval constraints were implemented to ensure physically meaningful and numerically stable parameter values that align with experimental memristor characteristics and theoretical boundaries. These constraints prevent parameter drift into unphysical regimes while maintaining the differentiability required for gradient-based optimization algorithms.

The following constraint intervals were applied to critical memristor model parameters:

- $R_{\text{ON}} \in (0, R_{\text{OFF}})\,\text{k}\Omega$ and $R_{\text{OFF}} \in (R_{\text{ON}}, 200)\,\text{k}\Omega$ – These constraints establish a physically meaningful hierarchy where the ON-state resistance is strictly smaller than the OFF-state resistance ($R_{\text{ON}} < R_{\text{OFF}}$), consistent with typical memristor switching behavior. The lower bound for R_{ON} prevents zero resistance values that would cause numerical singularities in current calculations, while the upper bound of 200 kΩ for R_{OFF} reflects experimentally observed resistance ranges in memristive devices and ensures computational stability during optimization.
- $x_0 \in (0, 1)$ – This constraint restricts the initial internal state variable to the normalized domain, ensuring that the memristor begins operation within physically admissible boundaries. The open interval prevents boundary singularities that could occur at $x_0 = 0$ or $x_0 = 1$.
- $x(t) \in (0, 1)$ – This temporal constraint maintains the internal state variable within the normalized domain throughout the entire simulation duration, preserving the physical interpretation of $x(t)$ as a fractional parameter representing the extent of ionic migration or structural modification within the memristive element.

These constraints are particularly crucial for memristor modeling because they preserve the fundamental physical relationships governing resistive switching phenomena while preventing the emergence of artifacts that could arise from parameter values outside the feasible operating regime.

The operational characteristics of the constraint function $\mathcal{L}_{\text{con}}(\mathbf{x})$ are illustrated in Figure 5 for the parameter configuration: $a = 0$, $b = 1$, $\delta = 0.01$ over the evaluation domain $\mathbf{x} \in (-0.5, 1.5)$. This visualization demonstrates the smooth sigmoid-based transition behavior that provides continuous

penalty gradients while maintaining differentiability across the parameter space. The constraint function architecture ensures that parameters within the admissible range $(0, 1)$.

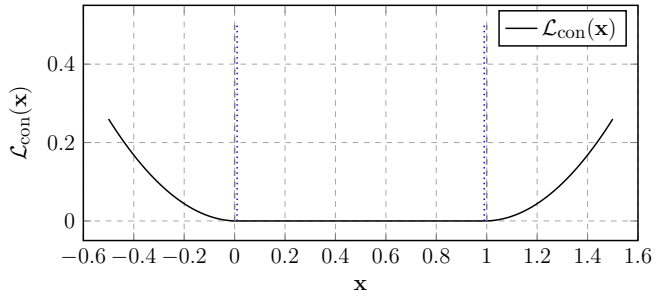


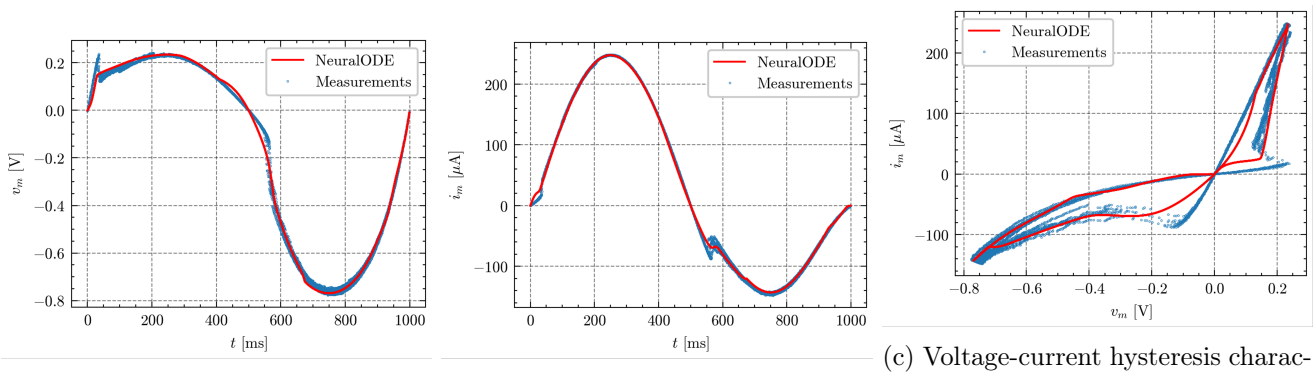
Figure 5: Operational characteristics of the constraint function $\mathcal{L}_{\text{con}}(\mathbf{x})$ demonstrating smooth penalty transitions for parameter bounds $a = 0$, $b = 1$ with transition margin $\delta = 0.01$ evaluated over the domain $\mathbf{x} \in (-0.5, 1.5)$. The function exhibits minimal penalty within the feasible region $(0, 1)$ and progressively increasing penalties as parameters approach or exceed the boundary limits.

3 Results

Representative simulation results demonstrating the neural network model's predictive capabilities are presented in Figure 6 through comparative analysis of a tungsten-doped memristor device subjected to sinusoidal voltage excitation with an amplitude of 1V and frequency of 1Hz. These specific excitation parameters were selected to capture the characteristic switching dynamics while maintaining operation within the device's linear regime, enabling comprehensive evaluation of the model's ability to reproduce fundamental memristive behaviors.

The analysis encompasses three critical dynamic characteristics that collectively characterize memristor behavior: temporal current evolution, applied voltage profiles, and voltage-current hysteresis relationships. The temporal current response reveals the device's instantaneous electrical behavior and switching kinetics, while the applied voltage profile confirms the fidelity of the input stimulus. Most significantly, the voltage-current hysteresis loops $(v_m - i_m)$ demonstrate the nonlinear relationship between these electrical quantities and illustrate the fundamental memory properties that define memristive behavior.

The hysteresis loops are particularly diagnostic of memristor performance, as their shape, area, and switching thresholds directly reflect the underlying ionic transport mechanisms and structural modifications responsible for resistive switching. The pinched hysteresis characteristic observed at the origin serves as a definitive signature of memristive behavior, while the loop area quantifies the energy dissipation associated with switching events. These features enable comprehensive validation of the neural network model's ability to capture both the static and dynamic aspects of memristor operation.



(a) Applied voltage waveform (b) Memristor current response (c) Voltage-current hysteresis characteristic

Figure 6: Representative simulation results for tungsten-doped memristor dynamics under sinusoidal voltage excitation (amplitude: 1V, frequency: 1Hz). (a) Applied voltage waveform demonstrating the sinusoidal input stimulus, (b) Predicted memristor current response showing nonlinear temporal evolution and switching characteristics, (c) Voltage-current hysteresis loop ($v_m - i_m$) illustrating the characteristic pinched behavior and memory properties fundamental to memristive operation. The neural network model successfully captures the essential features of memristor dynamics including switching thresholds, nonlinear conductance modulation, and hysteretic memory effects.

3.1 Aggregated Results for Neural Network Architecture Comparison

To systematically evaluate the impact of neural network architectural choices on memristor modeling accuracy, a comprehensive comparison was conducted across multiple network configurations. This analysis employed period-averaged waveforms to eliminate temporal phase variations and focus on the fundamental characteristics of memristive behavior reproduction. The period-averaging methodology involves temporal alignment of multiple excitation cycles followed by ensemble averaging, which effectively suppresses transient artifacts and highlights the steady-state dynamical properties captured by each architecture.

The architectural comparison encompasses variations in both network depth (number of hidden layers) and width (neurons per layer), enabling systematic investigation of the representational capacity required for accurate memristor dynamics modeling. This analysis is crucial for determining optimal network complexity that balances modeling accuracy with computational efficiency and training stability.

The systematic architectural comparison reveals several key insights into the representational requirements for accurate memristor modeling. Network depth influences the ability to capture complex nonlinear transformations associated with ionic transport and structural modifications, while network width affects the dimensionality of the internal representation space available for encoding dynamic state information. The loss function evolution patterns provide quantitative metrics for assessing convergence stability and final model accuracy across the architectural spectrum.

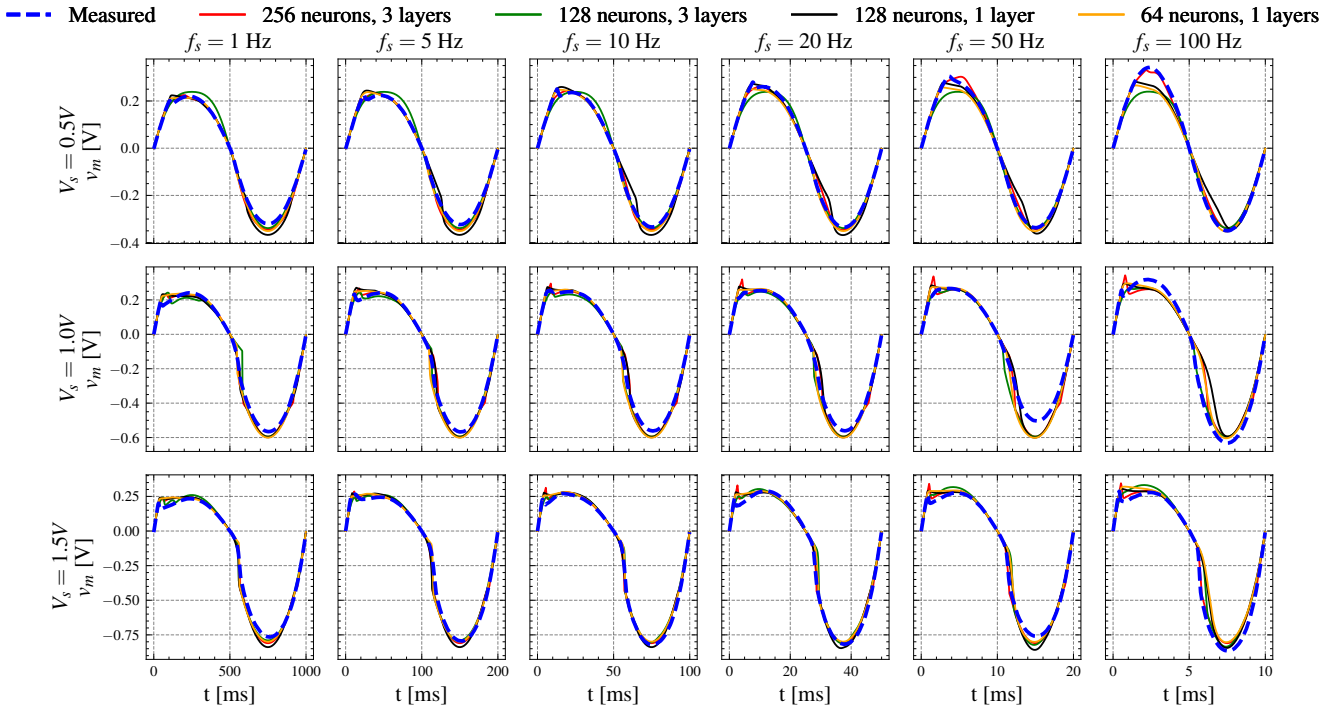


Figure 7: Comparative analysis of period-averaged memristor voltage waveforms (v_m) across different neural network architectures. The period-averaging process eliminates cycle-to-cycle variations and reveals the fundamental voltage characteristics reproduced by each network configuration. Variations between architectures indicate differences in the models' ability to capture the nonlinear voltage-dependent switching dynamics and internal state evolution of the memristive device.

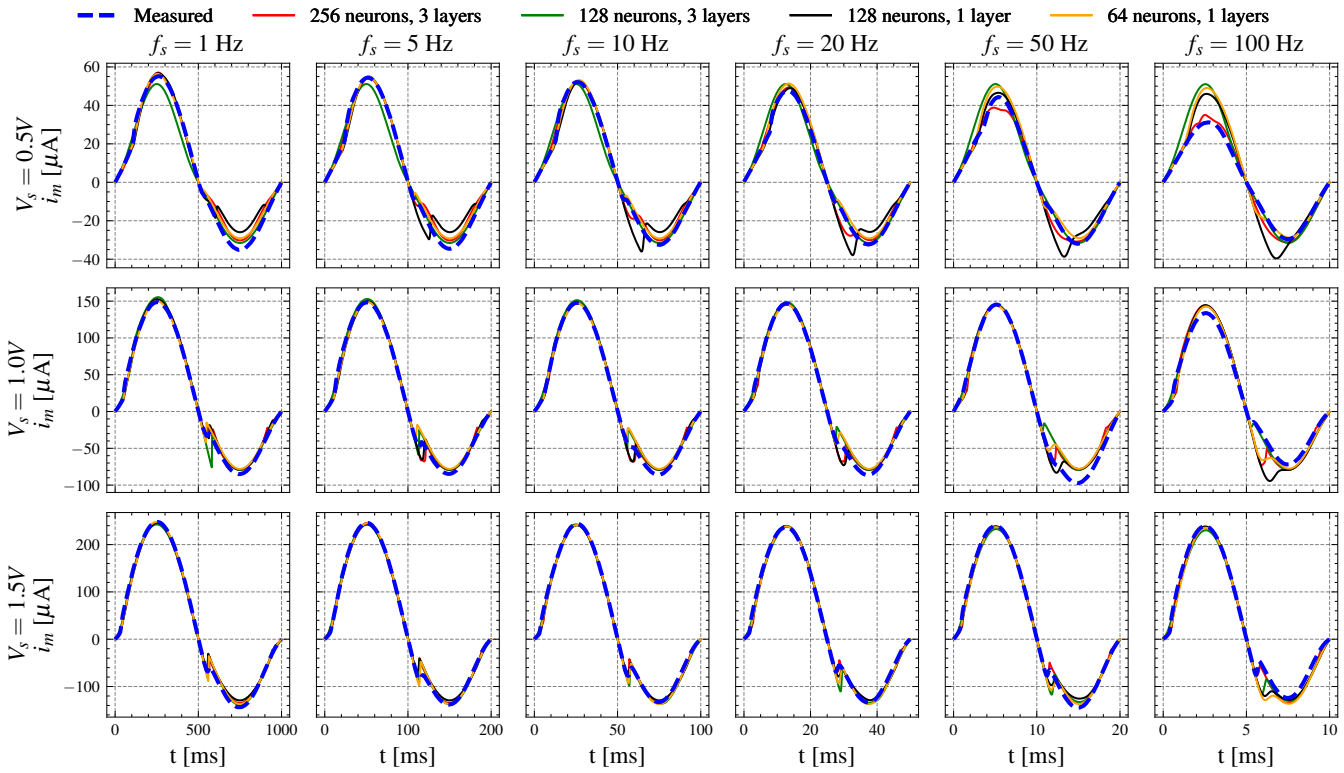


Figure 8: Comparative analysis of period-averaged memristor current responses (i_m) for various neural network architectures. The current waveforms directly reflect the conductance modulation characteristics and switching kinetics captured by each model. Architectural differences manifest as variations in current amplitude, switching sharpness, and temporal evolution patterns, providing insights into each network's capacity to represent the underlying ionic transport mechanisms governing memristor operation.

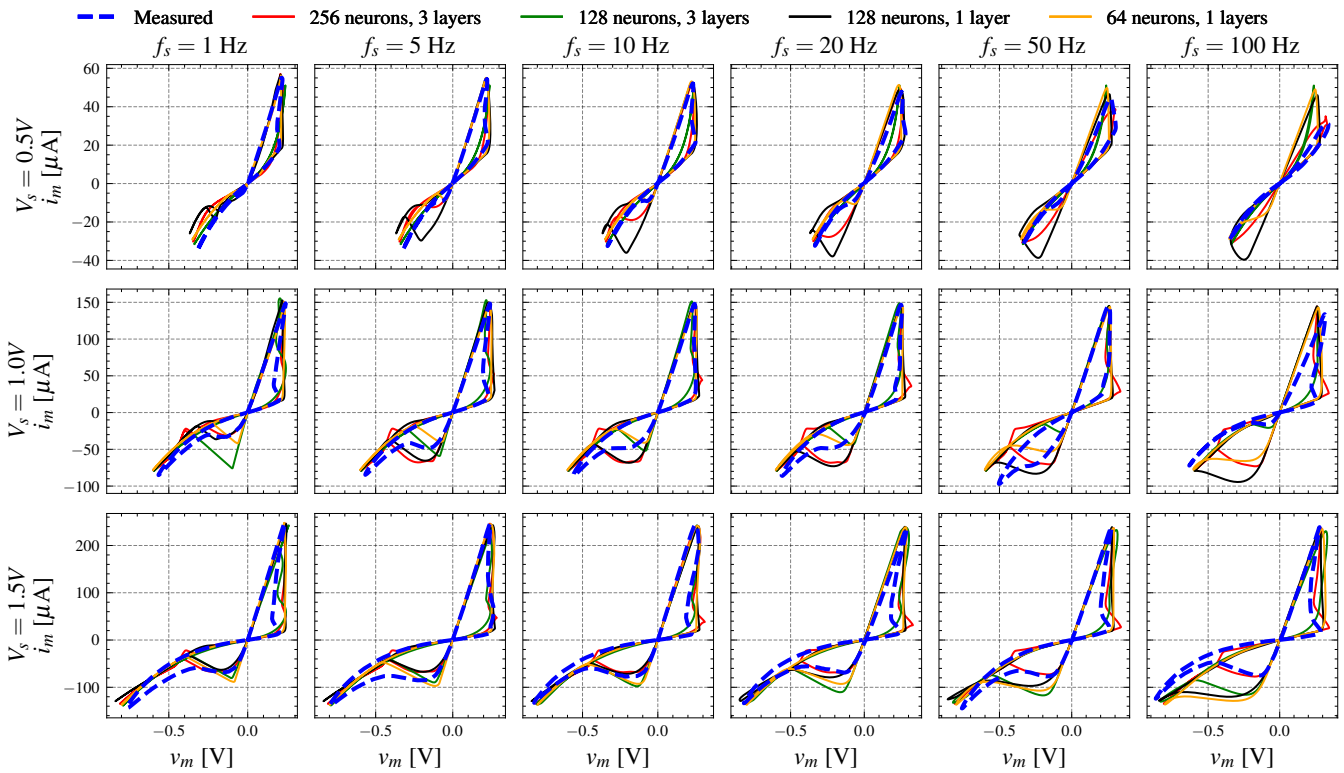
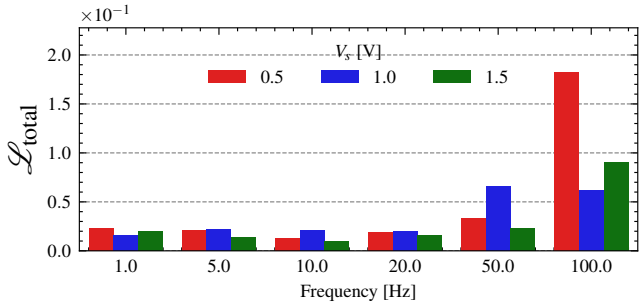
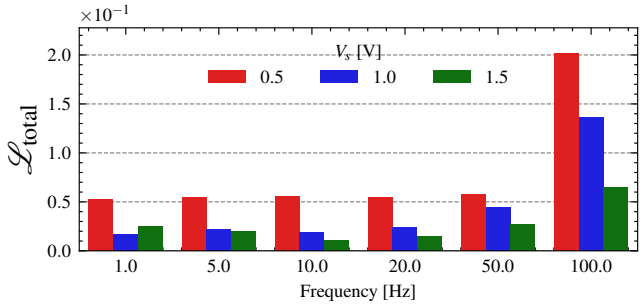


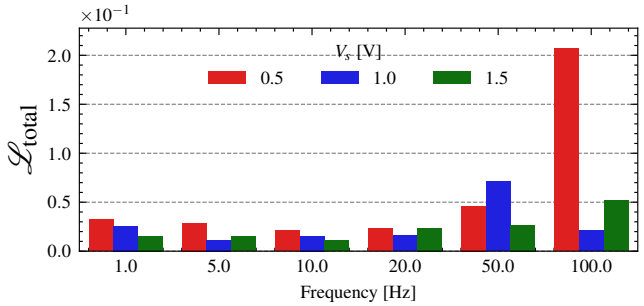
Figure 9: Comparative analysis of period-averaged voltage-current hysteresis characteristics ($v_m - i_m$) across different neural network architectures. The hysteresis loops represent the most diagnostic feature of memristive behavior, with loop shape, switching thresholds, and pinching characteristics directly reflecting the quality of the neural network's representation of memory-dependent conductance modulation. Architectural variations affect the fidelity of hysteretic reproduction, particularly in the switching transition regions and asymmetric behavior between positive and negative voltage excursions.



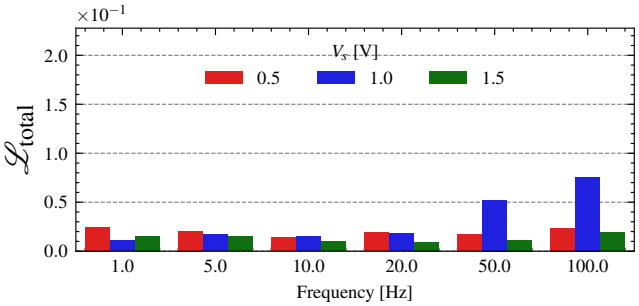
(a) 64 neurons, 1 hidden layer



(b) 128 neurons, 1 hidden layer



(c) 128 neurons, 3 hidden layers



(d) 256 neurons, 3 hidden layers

Figure 10: Training loss evolution comparison across different neural network architectures demonstrating the relationship between network complexity and learning performance.

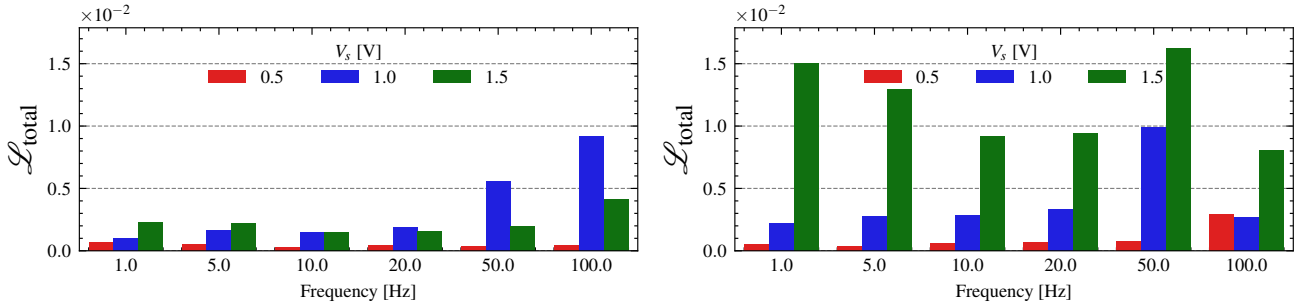
3.2 Comparative Analysis with Mean Metastable Switch (MMS) Model

To establish the performance advantages of the neural differential equation approach, a comprehensive comparative analysis was conducted between the developed neural network model and the established Mean Metastable Switch (MMS) phenomenological model. The MMS model represents a widely accepted analytical framework for memristor behavior that employs metastable state dynamics to describe resistive switching phenomena through probabilistic state transitions and thermal activation mechanisms.

This comparison is particularly significant as the MMS model has been extensively validated against experimental data and serves as a benchmark for memristor circuit simulation in the scientific literature. The comparative evaluation encompasses temporal voltage and current evolution, hysteretic characteristics, and quantitative loss function metrics to provide comprehensive assessment of modeling fidelity across multiple performance dimensions.

Quantitative analysis reveals substantial performance improvements achieved by the neural network approach. The average loss function value for the neural network model was 0.002052, representing a significant improvement compared to the MMS model's average loss of 0.005589. This $\approx$ 63% reduction in loss indicates markedly superior agreement between neural network predictions and experimental data, demonstrating the enhanced modeling capability achieved through the neural differential equation framework.

The superior performance of the neural network model can be attributed to its capacity for learning complex nonlinear mappings directly from experimental data, whereas the MMS model relies on predetermined phenomenological relationships that may not fully capture the device-specific dynamics and material-dependent switching characteristics inherent in experimental memristor devices.



(a) Neural network (256 neurons, 3 hidden layers)

(b) MMS phenomenological model

Figure 11: Comparative analysis of loss function evolution for (a) the optimal neural network architecture (3 hidden layers, 256 neurons per layer) and (b) the Mean Metastable Switch (MMS) model. The neural network demonstrates superior convergence characteristics and achieves significantly lower final loss values, indicating enhanced modeling accuracy and better agreement with experimental memristor dynamics.

Representative waveform comparisons between the neural network and MMS models are presented in Figures 12, 13, and 14, providing visual assessment of the modeling differences across key electrical characteristics. These comparisons reveal the neural network's superior ability to capture subtle features of memristor dynamics that are not adequately represented by the phenomenological MMS framework.

The comprehensive comparative analysis establishes the neural differential equation approach as a superior methodology for memristor circuit simulation, achieving significant improvements in modeling accuracy while maintaining computational tractability. These results validate the effectiveness of data-driven neural network approaches for capturing complex nonlinear device physics that may be inadequately represented by traditional phenomenological models.

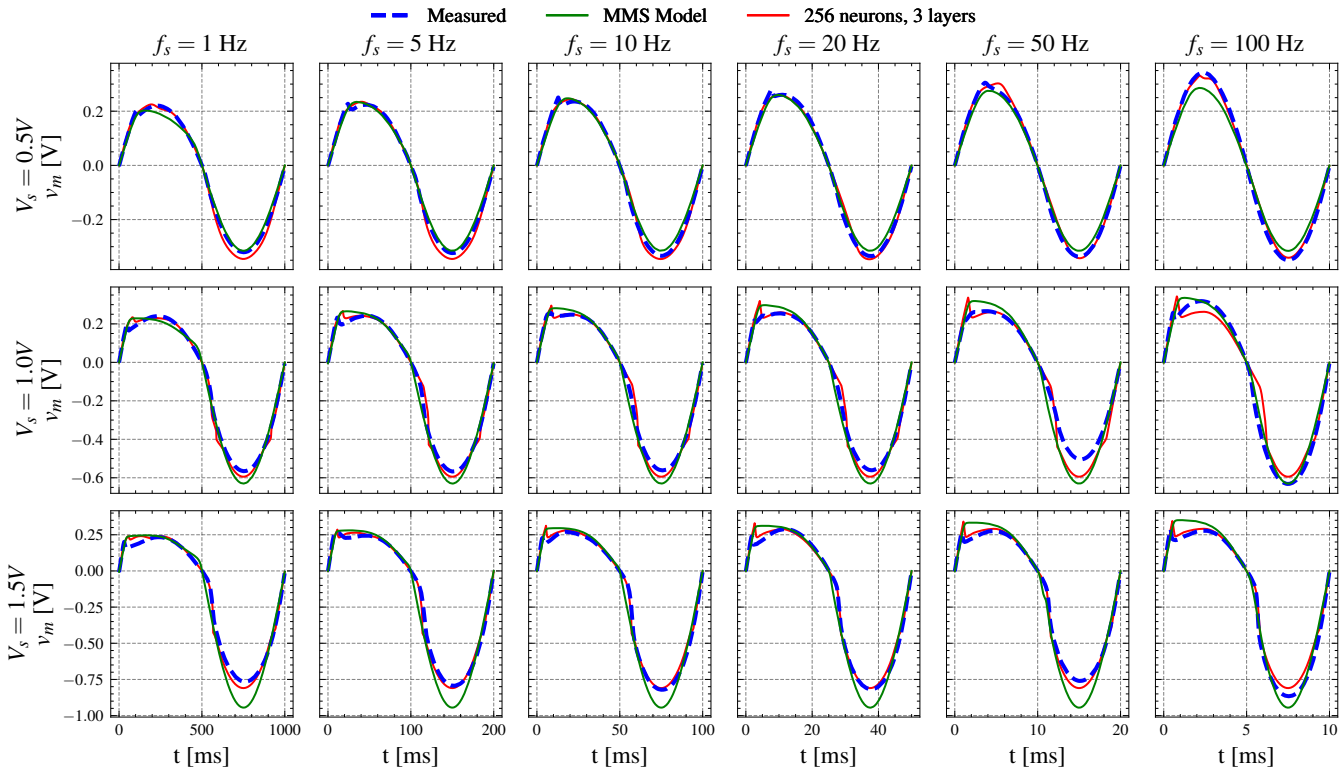


Figure 12: Comparative analysis of period-averaged memristor voltage waveforms (v_m) between the neural network model and the MMS phenomenological model. The neural network demonstrates superior fidelity in reproducing experimental voltage characteristics, particularly in the transition regions where nonlinear switching dynamics dominate. Differences between the models highlight the enhanced representational capacity of the neural differential equation approach for capturing complex voltage-dependent switching phenomena.

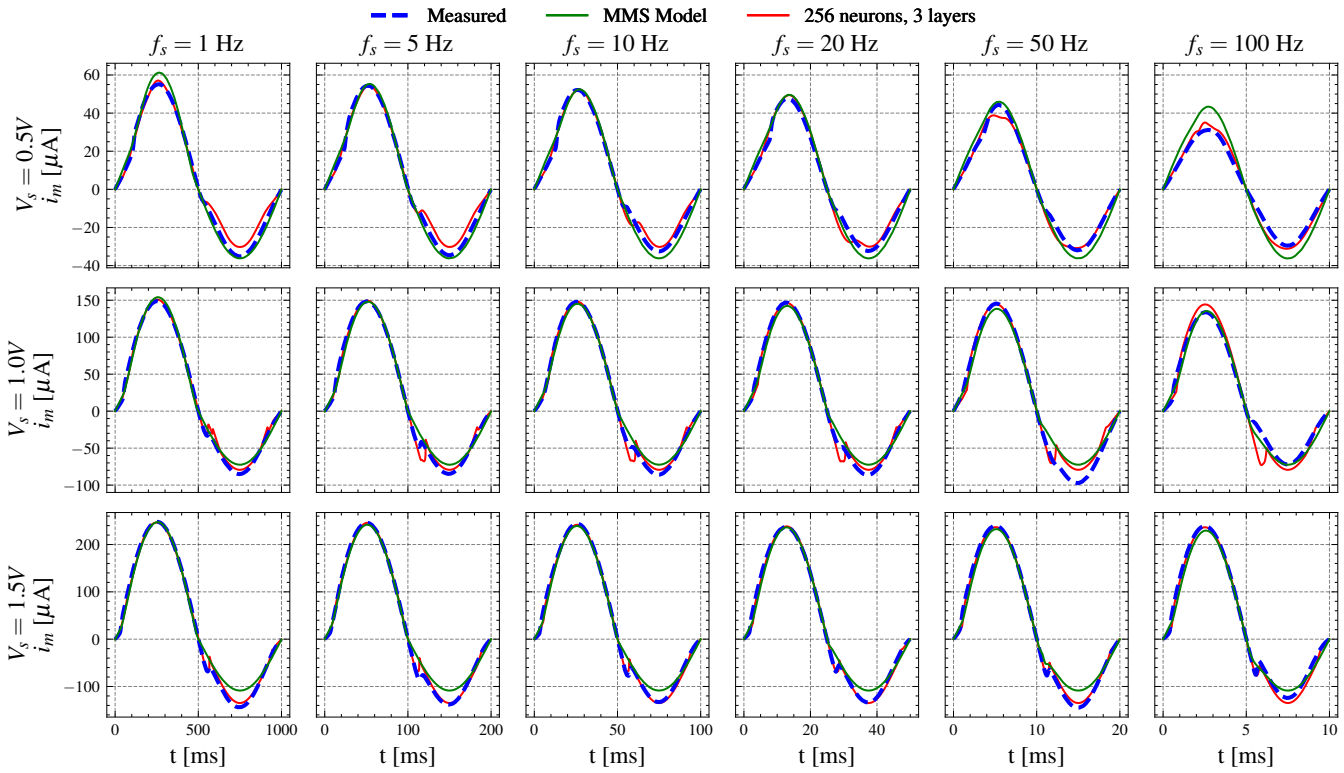


Figure 13: Comparative analysis of period-averaged memristor current responses (i_m) for the neural network model versus the MMS model. The neural network exhibits superior accuracy in reproducing current switching characteristics, amplitude modulation, and temporal evolution patterns. The enhanced current prediction capability reflects the neural network's ability to learn device-specific conductance modulation mechanisms that extend beyond the assumptions embedded in the MMS phenomenological framework.

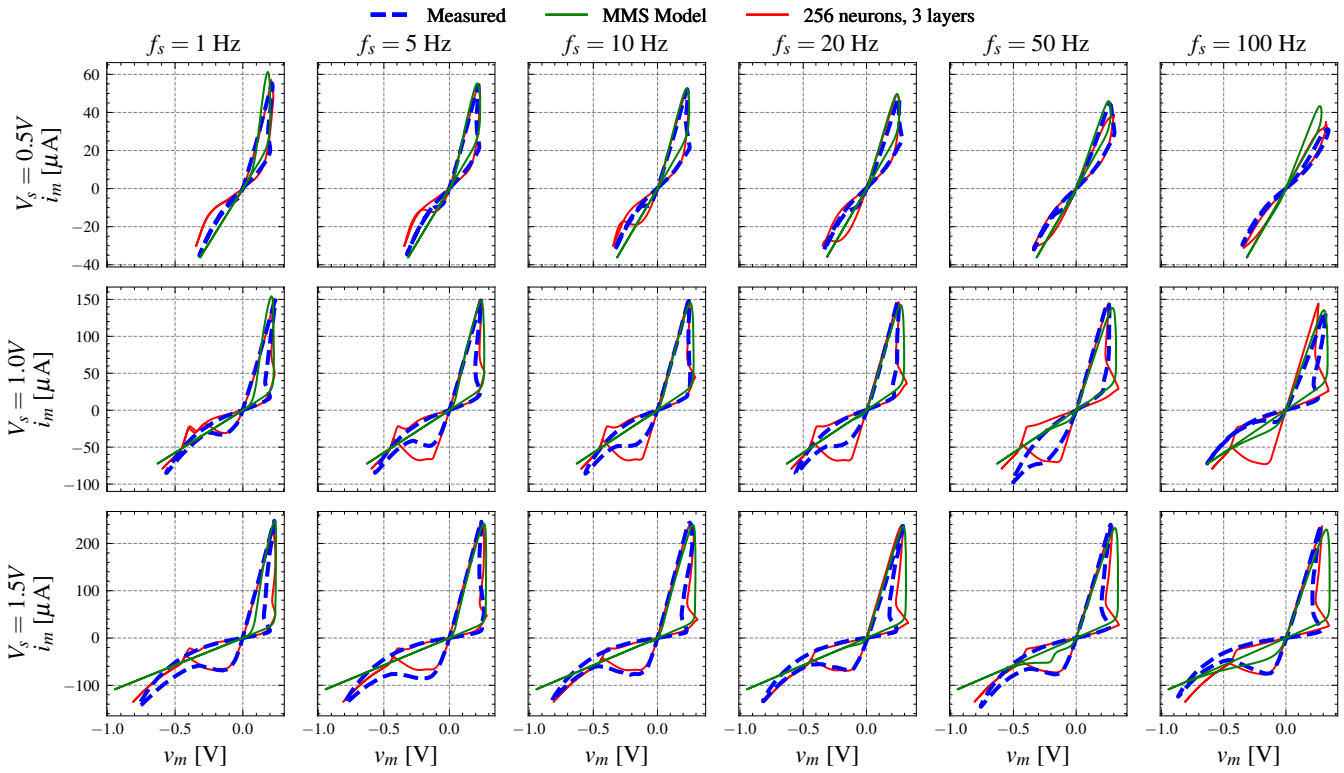


Figure 14: Comparative analysis of period-averaged voltage-current hysteresis characteristics ($v_m - i_m$) between the neural network and MMS models. The hysteresis loop comparison provides the most diagnostic assessment of modeling quality, as these characteristics directly reflect the memory-dependent conductance properties fundamental to memristive behavior. The neural network demonstrates superior reproduction of loop shape, switching thresholds, and asymmetric behavior, indicating enhanced capture of the underlying physical mechanisms governing resistive switching dynamics.

3.3 Hyperparameter Sensitivity Analysis

TODO: The hyperparameter sensitivity analysis will be conducted after the neural network training process is completed.

References

[1] Kristy A. Campbell. "Self-directed channel memristor for high temperature operation". In: *Microelectronics Journal* 59 (Jan. 2017), pp. 10–14. ISSN: 1879-2391. DOI: 10.1016/j.mejo.2016.11.006. URL: http://dx.doi.org/10.1016/j.mejo.2016.11.006.

[2] Ricky T. Q. Chen, Brandon Amos, and Maximilian Nickel. "Learning Neural Event Functions for Ordinary Differential Equations". In: *International Conference on Learning Representations* (2021).

[3] Ricky T. Q. Chen et al. "Neural Ordinary Differential Equations". In: 2018.

[4] Bartłomiej Garda and Karol Bednarz. "Comprehensive Study of SDC Memristors for Resistive RAM Applications". In: *Energies* 17.2 (Jan. 2024), p. 467. ISSN: 1996-1073. DOI: 10.3390/en17020467. URL: http://dx.doi.org/10.3390/en17020467.

[5] Younghyun Lee, Kyeongmin Kim, and Jonghwan Lee. "A Compact Memristor Model Based on Physics-Informed Neural Networks". In: *Micromachines* 15.2 (2024). ISSN: 2072-666X. DOI: 10.3390/mi15020253. URL: https://www.mdpi.com/2072-666X/15/2/253.

[6] Timothy W. Molter and M. Alexander Nugent. "The Generalized Metastable Switch Memristor Model". In: *CNNA 2016; 15th International Workshop on Cellular Nanoscale Networks and their Applications*. 2016, pp. 1–2.

[7] Valerii Ostrovskii et al. "Structural and Parametric Identification of Knowm Memristors". In: *Nanomaterials* 12.1 (Dec. 2021), p. 63. ISSN: 2079-4991. DOI: 10.3390/nano12010063. URL: http://dx.doi.org/10.3390/nano12010063.

[8] Dmitri B. Strukov et al. "The missing memristor found". In: *Nature* 453.7191 (May 2008), pp. 80–83. ISSN: 1476-4687. DOI: 10.1038/nature06932. URL: http://dx.doi.org/10.1038/nature06932.