Learning Chaotic Dynamics with Neuromorphic Network Dynamics

Y. Xu,[1] G.A. Gottwald,[2] and Z. Kuncic[3,4]

[1)]*School of Physics, University of Sydney, Sydney, NSW 2006, Australia.*

[2)]*School of Mathematics and Statistics, University of Sydney, Sydney, NSW 2006, Australia*

[3)]*School of Physics, University of Sydney, Sydney, NSW 2006, Australia.*

[4)]*Centre for Complex Systems, University of Sydney, Sydney, NSW 2006, Australia.*

(*zdenka.kuncic@sydney.edu.au)

(Dated: 13 June 2025)

This study investigates how dynamical systems may be learned and modelled with a neuromorphic network which is itself a dynamical system. The neuromorphic network used in this study is based on a complex electrical circuit comprised of memristive elements that produce neuro–synaptic nonlinear responses to input electrical signals. To determine how computation may be performed using the physics of the underlying system, the neuromorphic network was simulated and evaluated on autonomous prediction of a multivariate chaotic time series, implemented with a reservoir computing framework. Through manipulating only input electrodes and voltages, optimal nonlinear dynamical responses were found when input voltages maximise the number of memristive components whose internal dynamics explore the entire dynamical range of the memristor model. Increasing the network coverage with the input electrodes was found to suppress other nonlinear responses that are less conducive to learning. These results provide valuable insights into how a practical neuromorphic network device can be optimised for learning complex dynamical systems using only external control parameters.

I. INTRODUCTION

Neuromorphic computing aims to achieve efficient computation by emulating the brain's powerful information processing abilities[1–4]. Two broad approaches that have been successfully demonstrated are: (i) electrical circuits that emulate point neuron spike models and/or spike–based learning models[5,6]; and (ii) nano–electronic materials with inherent neuro–synaptic dynamics[1,7–9]. The latter includes resistive switching memory (memristive) materials[10–13]. Neuromorphic computing with memristors has been demonstrated with crossbar arrays, which can be used to perform physical matrix–vector operations to accelerate inference of artificial neural network models[14,15]. Alternately, memristor crossbar arrays have also been used to implement reservoir computing (RC)[16]. Neuromorphic RC with memristive systems is a particularly promising approach given that electronic reservoirs with neuromorphic dynamics closely resemble the brain's physical reservoir[17,18]. However, a drawback of memristor crossbars for neuromorphic RC is their regular structure and limited scale.

This study focuses on neuromorphic RC using memristive nanowire networks. These are complex, self-organised electrical circuits comprised of axon-like nanowires and synapse-like memristive junctions between overlapping nanowires, where conductance changes — the physical realisation of synaptic weights — are triggered by migration of atoms similar to the transmission of neurotransmitter molecules, constrained by physical equations of state and conservation laws[1]. Memristive nanowire networks exhibit emergent brain–like dynamics, such as dynamical phase transitions (continuous and discontinuous) and synchronisation. Previous studies have shown that different dynamical regimes may be exploited for learning[8,19–21]. As a consequence of their self–organisation (i.e. their ability to acquire non–trivial structure–function properties), a single voltage pulse may have cascading effects that evolve all memristive elements in the neuromorphic network, in contrast to digital implementations of artificial neural networks, where large numbers of transistors must be individually addressed to update a similar number of network weights[2].

Nanowire networks naturally self–organise into a Recurrent Neural Network (RNN) structure, with a functional connectivity that is highly suitable for RC. In standard algorithmic RC, the reservoir is typically a fixed random RNN of which each node evolves according to a mathematical activation function, and only a single linear output layer is trained with a computationally efficient linear regressor or classifier[22]. In this way, RC approaches can achieve excellent performance in various learning tasks involving temporal data[23–25], with training times that are orders of mag-

nitude shorter than standard machine learning approaches. Physical RC, however, may operate differently because of the dynamics of the physical reservoir. In the case of neuromorphic RC with memristive nanowire networks, all the network dynamics are driven by the memristive edge junctions and are constrained by physics[26].

To further investigate how memristive nanowire networks operate as physical reservoirs for neuromorphic RC, we present a simulation study focusing on a specific application: autonomous prediction of a multivariate chaotic time series, the well-known Lorenz63 system[27]. Prediction of the Lorenz63 system has been previously demonstrated with several different RC and related approaches[28–33]. To the best of our knowledge, this study is the first to demonstrate autonomous prediction of the Lorenz63 attractor with neuromorphic RC. Understanding and predicting the behaviour of complex nonlinear dynamical systems is crucial for many scientific disciplines, including physics, climatology, neuroscience and biology, and for real–world applications such as financial markets and social networks[34–36]. Chaotic dynamical systems, in particular, are notoriously difficult to forecast due to their high sensitivity to initial conditions and highly nonlinear nature, resulting in aperiodic and categorically unpredictable dynamics. As such, the task of forecasting chaotic dynamics provides a challenging testing ground for exploring the limits of the capabilities of neuromorphic networks, and for testing the hypothesis that neuromorphic physical reservoirs can predict dynamical systems using the inherent physics of the neuromorphic system itself.

This study primarily aims to gain deeper insights into the inner workings of memristive nanowire networks for learning via neuromorphic RC. No attempt is made to achieve state-of-the-art prediction results, such as those already achieved with random feature models[33], Long-Short Term Memory (LSTM) networks[37] or algorithmic RC models[38]. Rather our study provides the first proof of concept that analog devices consisting of a physical neuromorphic network is capable of performing learning tasks, through controlling the network via only the external input parameters. The forecast skill we obtain in our study for a network with $2,000$ nodes is far below those achieved by digital reservoir computers of the same size. However, as we will see performance increases with network size, and physical neuromorphic networks can easily be scaled up to millions of nodes[8,39]; such network sizes are out of reach for digital reservoir computers. Also considering optimisations of the computational framework and the network itself (which is beyond the scope of this study), physical neuromorphic devices have great potential to outperform digital computers, with significantly faster and more energy efficient performance.

The remainder of this article is organised as follows. Sect. II describes the methods used in the simulation study, including the RC setup and memristive network model. The results in Sec. III start with analysis of the chaotic time series prediction task (in Sec. III A), then presents and discusses all sources of nonlinear dynamical features, on the global network level in Sec. III B, then local memristive edge level and their interactions in Sec. III C and Sec. III D. This study concludes with Sec. IV, which shines light on the future works needed to fully optimise this neuromorphic network for modelling dynamical systems.

II. METHODS

A neuromorphic reservoir was constructed using a physically–motivated model of a complex electrical circuit comprised of nonlinear resistive switching memory circuit elements known as memristors[1,19,20,40,41]. Given some input signal $\mathbf{u}(t) \in \mathbb{R}^{N_\mathrm{u}}$, the neuromorphic network is used to autonomously predict a dynamical time series $\mathbf{y}(t) \in \mathbb{R}^{N_\mathrm{y}}$, by computing the estimate $\hat{\mathbf{y}}(t) \in \mathbb{R}^{N_\mathrm{y}}$ of the teacher signal $\mathbf{y}(t)$. This is performed using a reservoir computing (RC) framework in which reservoir input values $\mathbf{r}_\mathrm{in}(t)$ are coupled to the input signal $\mathbf{u}(t)$ via a linear input layer with weight matrix $W_\mathrm{in} \in \mathbb{R}^{N_\mathrm{in} \times N_\mathrm{u}}$ and an input bias vector $\mathbf{b}_\mathrm{in} \in \mathbb{R}^{N_\mathrm{in}}$, such that

$$\mathbf{r}_\mathrm{in}(t) = W_\mathrm{in}\, \mathbf{u}(t) + \mathbf{b}_\mathrm{in}. \tag{1}$$

with W_in and $\mathbf{b}_\mathrm{in}$ entries uniformly randomly sampled on intervals $[-w, w]$ and $[-b, -b/2] \cup [b/2, b]$, respectively (the choice of the bias sampling is explained in Sec. III D). We choose $w = \alpha$, $b = 5\alpha$ with $\alpha = 0.2\,\mathrm{V}$. The α variable effectively serves as the input voltage scaling parameter which converts the input signals into appropriate voltages.

A graph representation of the neuromorphic network $\mathscr{N}$ consisting of N nodes is abstracted as the reservoir (see Fig. 1), a high dimensional dynamical system which nonlinearly transforms a temporal signal $\mathbf{r}_\mathrm{in}(t) \in \mathbb{R}^{N_\mathrm{in}}$ into readout values $\mathbf{r}_\mathrm{out}(t) \in \mathbb{R}^{N_\mathrm{out}}$, with $N_\mathrm{out} \geq N_\mathrm{in}$. Only a subset $N_\mathrm{in} < N$ of all available nodes are used as input nodes, and $N_\mathrm{out} < N$ is the number of voltage readout nodes. In a machine learning context, these readouts $\mathbf{r}_\mathrm{out}(t)$ serve as high-dimensional dynamical feature embeddings that can be used to learn the nonlinear dynamics of the input data. The neuromorphic network model is described further in Sec. II A.

The dynamical features $\mathbf{r}_\mathrm{out}(t)$ generated by $\mathscr{N}$ are linearly coupled to an external output layer with a weight matrix $W_\mathrm{out} \in \mathbb{R}^{N_\mathrm{y} \times N_\mathrm{out}}$ that is trained to learn the dynamics of $\mathbf{u}(t)$ by optimising

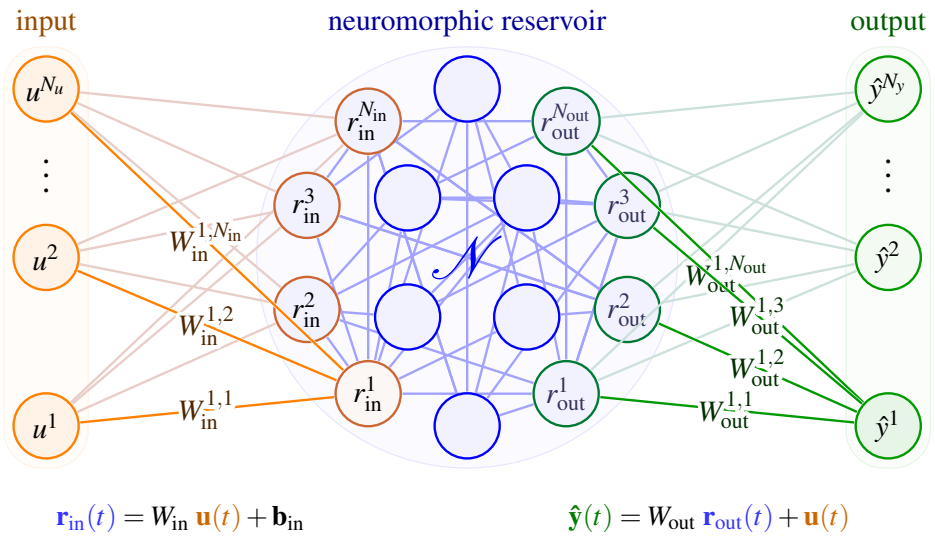

FIG. 1. Schematic of the dynamic neuromorphic reservoir computer. A signal vector $\mathbf{u}(t)$, weighted by a fixed random W_{in} and scaled by a constant voltage α, is input into the neuromorphic reservoir $\mathscr{N}$ via selected input nodes. Voltage signals $\mathbf{r}_{\text{in}}(t)$ (which also include random bias values $\mathbf{b}_{\text{in}}$) are mapped into a higher-dimensional dynamical feature space which is sampled from other nodes $\mathbf{r}_{\text{out}}(t)$. Only the output weight matrix W_{out} is trained to learn estimates $\hat{\mathbf{y}}(\mathbf{t})$.

output estimates $\hat{\mathbf{y}}(t)$ according to

$$\hat{\mathbf{y}}(t) = W_{\text{out}}\,\mathbf{r}_{\text{out}}(t) + \mathbf{u}(t), \tag{2}$$

which includes components $\mathbf{u}(t)$ that serve a similar role to skip connections in residual networks[42].

During training, the input signal is $\mathbf{u}(t) = \mathbf{y}(t - \Delta t)$, and W_{out} is trained using ridge regression, given by

$$W_{\text{out}} = \underset{W_{\text{out}}}{\arg\min} \left(||W_{\text{out}}\mathbf{r}_{\text{out}}(t) - \Delta\mathbf{y}(t)||_F^2 + \gamma ||W_{\text{out}}||_F^2 \right), \tag{3}$$

where $\Delta\mathbf{y}(t) = \mathbf{y}(t) - \mathbf{y}(t - \Delta t)$, with a Tikhonov regularisation parameter value of $\gamma = 10^{-6}$, and $||\cdot||_F$ denotes the Frobenius norm.

To perform dynamical system forecasting, the current estimate $\hat{\mathbf{y}}(t)$ at one time step ahead is predicted using the previous estimate $\hat{\mathbf{y}}(t - \Delta t)$ as input (i.e. $\mathbf{u}(t) = \hat{\mathbf{y}}(t - \Delta t)$). Substituting into Eq. (2), it can be seen that during prediction

$$\frac{d\hat{\mathbf{y}}}{dt} \approx \frac{\hat{\mathbf{y}}(t) - \hat{\mathbf{y}}(t - \Delta t)}{\Delta t} = \frac{1}{\Delta t} W_{\text{out}}\,\mathbf{r}_{\text{out}}(t). \tag{4}$$

Hence, with skip connections, the neuromorphic reservoir learns an Euler discretisation with fixed sample time Δt of the dynamical equation.

Here we use the neuromorphic RC framework to perform autonomous prediction of the Lorenz63 system[27], given by

$$\frac{d\tilde{y}_1}{dt} = \rho_1(\tilde{y}_2 - \tilde{y}_1), \quad \frac{d\tilde{y}_2}{dt} = \tilde{y}_1(\rho_2 - \tilde{y}_3) - \tilde{y}_2, \quad \frac{d\tilde{y}_3}{dt} = \tilde{y}_1\tilde{y}_2 - \rho_3\tilde{y}_3. \tag{5}$$

The constants were set to the standard values of $\rho_1 = 10$, $\rho_2 = 28$ and $\rho_3 = 8/3$ that generate chaotic dynamics. The normalised Lorenz values is recorded at equidistant sampling times $\Delta t = 0.005$, and a training length of 135 Lyapunov times was chosen to ensure a low training error (less than 0.01 normalised root mean squared error). The sampling time Δt is used as the discretisation time step of our network simulations. All three discretised variables of the Lorenz63 system in Eq. (5) were provided as input after being normalised to a mean of 0 and standard deviation of 1, i.e. the ith element of $\mathbf{y}(t)$ is

$$y_i(t) = \frac{\tilde{y}_i(t) - \langle \tilde{y}_i(t) \rangle}{\sigma_{\tilde{y}_i}}, \tag{6}$$

where $\langle \cdot \rangle$ denotes the time average, and $\sigma_{\tilde{y}_i}$ is the standard deviation (across all training time) of the ith component of the Lorenz system.

For quantifying prediction accuracy, the forecast time t_f was determined as the longest time such that the relative forecast error $\mathscr{E}_f(t_f)$ is less than a threshold θ:

$$t_f = \max_t \left(\lambda_{\max} t | \mathscr{E}_f(t) \leq \theta\right) \quad \text{with} \quad \mathscr{E}_f(t) = \frac{||\mathbf{y}(t) - \hat{\mathbf{y}}(t)||^2}{\langle ||\mathbf{y} - \langle \mathbf{y} \rangle||^2 \rangle}, \tag{7}$$

where $||\cdot||$ denotes the Euclidean norm. A threshold of $\theta = 0.4$ was chosen to compare results with related studies[28,29]. The forecast time t_f is measured in units of the Lyapunov time $\lambda_{\max}^{-1}$, where the largest Lyapunov exponent of the Lorenz63 system is $\lambda_{\max} = 0.91$.

A. Neuromorphic Network Model

The neuromorphic network $\mathscr{N}$ is comprised of a complex memristive circuit as a model of self–organised nanowire networks, which exhibit neuro–synaptic dynamics under electrical stimulation[1]. Due to its unique network topology, the neuromorphic networks were generated by modelling the bottom-up self-assembly process of the physical nanowire network; see Ref.[43] for detailed network generation and network topology analysis.

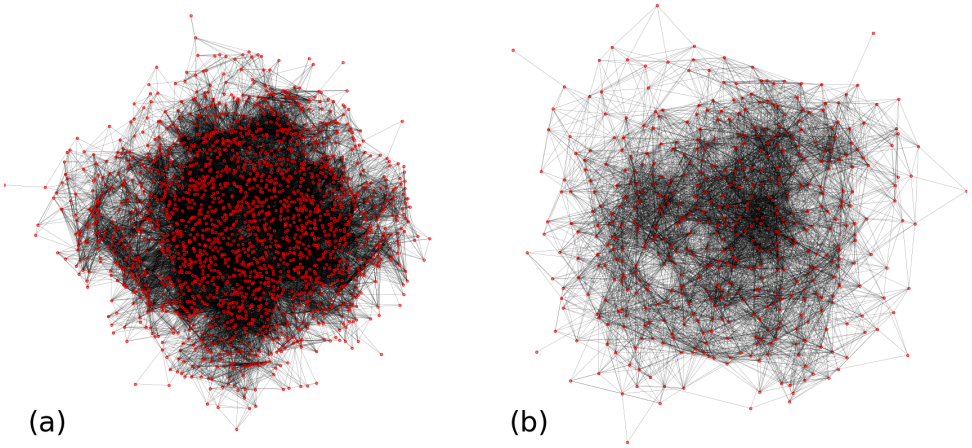

FIG. 2. Graph representations of simulated neuromorphic networks used in this study: (a) network with 2,000 nodes and 47,946 edges; (b) network with 500-nodes and 9,905 edges.

The neuromorphic network $\mathscr{N}$ is abstracted as a graph representation, with nodes representing nanowires and edges representing memristors. A network of 2,000 nodes and 47,946 edges, alongside another network of 500 nodes and 9,905 edges were used, with 5% of all nodes serving as inputs and up to 90% of all other nodes serving as readout nodes. Their graph representations are illustrated in Fig. 2.

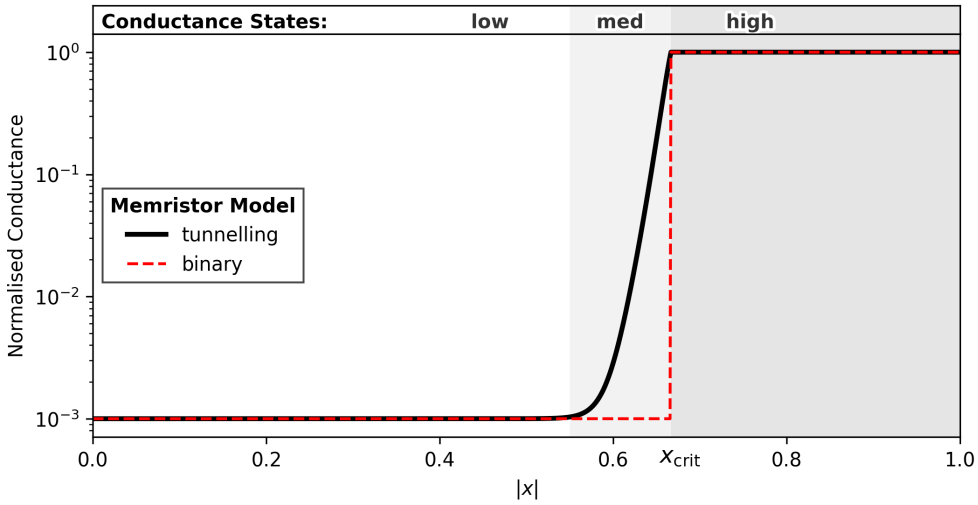

FIG. 3. Memristive edge normalised conductance $(g(x)/\max_x[g(x)])$ as a function of the internal state parameter x, for both the tunnelling memristor model (black) and the binary model (dotted red). The low, medium and high conductance states are indicated.

The graph Laplacian of $\mathscr{N}$ is used to define the linear system of equations corresponding to Kirchhoff's and Ohm's laws for electrical circuits. For an arbitrary, non–trivial network with N nanowires and N_{in} contact electrodes, for every specified time-step, the node voltages $\mathbf{r}$ and electrode currents $\mathbf{i}_{\text{in}}$ are calculated by solving the following system of equations in response to the external input signals $\mathbf{r}_{\text{in}}$

$$L \begin{bmatrix} \mathbf{r} \\ \mathbf{i}_{\text{in}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{r}_{\text{in}} \end{bmatrix}, \tag{8}$$

where L in its block matrix representation is

$$L = \left[\begin{array}{c|c} \mathscr{L} & C^T \\ \hline C & 0_{N_{\text{in}}} \end{array}\right]. \tag{9}$$

Here, $\mathscr{L} = D - W$ is the graph Laplacian of size $N \times N$, W is the weighted adjacency matrix of the network with each of its element given by $W_{ij} = g_{ij}A_{ij}$, where A is the adjacency matrix of the network, and g_{ij} is the conductance of the edge (junction) connecting the ith and jth nodes (nanowires), which evolve dynamically with respect to past voltage values (described formally in Sec. II B). The $N \times N$ matrix D is the weighted degree matrix given by $D = \text{diag}(d_i)$, $d_i = \sum_{l=1}^{N} W_{il}$. C is a $N_{\text{in}} \times N$ matrix, with $C_{ij} = 1$ for electrodes with index i that are assigned on the node with index j, and $C_{ij} = 0$ otherwise. $0_{N_{\text{in}}}$ is a zero matrix of size $N_{\text{in}} \times N_{\text{in}}$, and $\mathbf{0}$ is a $1 \times N$ zero vector. $\mathbf{r}$ is a $1 \times N$ vector encoding the voltage of all N nanowire nodes.

The voltage v_{ij} across the edge connecting the ith and jth node is $v_{ij} = r_i - r_j$, where r_i and r_j are the node voltages at the ith and jth index, respectively. Once the node voltages are solved, an internal state parameter x_{ij} is updated for each junction with a memristor model. The model is then used to calculate the conductance g_{ij} at each memristive junction, which can be used to update $\mathscr{L}$ for the next timestep.

B. Memristor Model

We use a threshold memristor model[19], given by the following expression for a single memristive junction with voltage $v(t)$ (indices omitted for clarity),

$$\frac{dx}{dt} = \begin{cases} \eta(|v(t)| - V_{\text{set}})\ \text{sgn}(v(t)) & |v(t)| > V_{\text{set}} \\ 0 & V_{\text{reset}} \leq |v(t)| \leq V_{\text{set}} \\ \eta q(|v(t)| - V_{\text{reset}})\ \text{sgn}(x(t)) & V_{\text{reset}} > |v(t)| \\ 0 & |x| \geq 1 \end{cases} \tag{10}$$

The dimensionless state variable $x(t)$ describes the physico–chemical processes responsible for memristive switching, including electro–chemical metallisation and electron tunnelling. Here q controls the decay of memory of the memristor, and η controls the time scale of the memristor relative to the input signal.

The memristive junction conductance $g(x)$ only depends on $x(t)$ and is expressed as

$$g(x) = \frac{1}{R_t(x) + R_{\mathrm{on}} + \frac{R_{\mathrm{on}}^2}{R_{\mathrm{off}} - R_{\mathrm{on}}}} + \frac{1}{R_{\mathrm{off}}}, \tag{11}$$

where the tunnelling conductance $G_t(x) = R_t(x)^{-1}$ assumes the low voltage Simmon's formula[44]:

$$G_t(x) = \frac{\zeta_0}{s} \exp(\zeta_1 s), \tag{12}$$

with

$$s = \max\left[(x_{\mathrm{crit}} - |x|)\frac{s_{\max}}{x_{\mathrm{crit}}}, 0\right]. \tag{13}$$

All parameter values not explicitly stated here are listed in Appendix A.

We further consider a simpler binary memristor model where the junction conductance g is either R_{on}^{-1} or R_{off}^{-1}, which is effectively equivalent to the above memristor model without electron tunnelling transport:

$$g(x) = \begin{cases} R_{\mathrm{on}}^{-1} & |x| > x_{\mathrm{crit}}, \\ R_{\mathrm{off}}^{-1} & |x| \leq x_{\mathrm{crit}}. \end{cases} \tag{14}$$

The conductance profiles of both memristor models are illustrated in Fig. 3.

We remark that the nontrivial temporal evolution of the internal state variable in Eq. (10) is the only nonlinearity explicitly introduced in the neuromorphic network. If the state variables do not evolve (i.e. $\dot{x}_{ij} = 0$ for all memristive junctions), the neuromorphic network reduces effectively to a network of linear resistors.

Further details of the neuromorphic network and memristor models can be found in Refs.[19–21].

C. Network Dynamics

Due to the memristive dynamics described above, the neuromorphic network exhibits coupled node–edge dynamics. An example of a network's dynamical behaviour is presented in Fig. 4, which shows snapshots of conductance changes for different times as a voltage bias is applied to one node. As voltage is distributed to nodes according to their connectivity, edge conductances

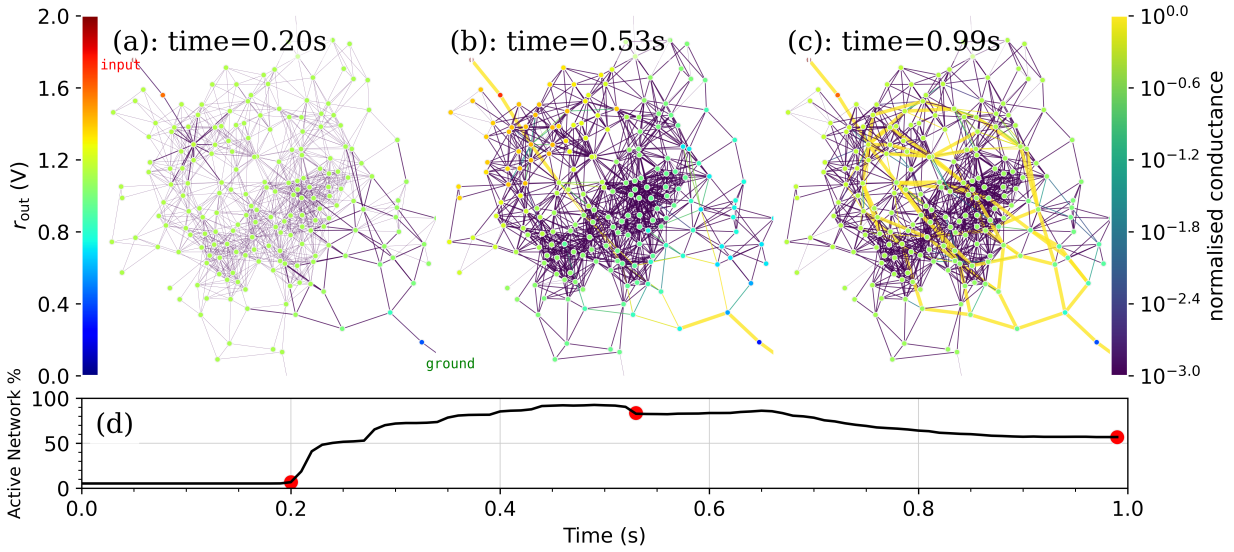

FIG. 4. Graph visualisations of a simulated 200-node, 1,213-edge neuromorphic network showing dynamic connectivity in response to a constant DC voltage bias of 2 V. Snapshots are shown at times (a) 0.2 s, (b) 0.5 s and (c) 1.0 s, with the normalised conductance on edges, $g(x)/\max_x[g(x)]$, indicated by the right colourbar and the thickness of the respective edges, the node voltages $\mathbf{r}_{\text{out}}$ indicated by the colourbar on the left; (d) corresponding percentage of the network that is active with edge voltage $v_{ij} \geq V_{\text{set}}$.

begin to evolve forming localised branches (Fig. 4(a)) that drive further node voltage redistribution, with the formation of the first high conductance path (Fig. 4(b), yellow) between the input and the ground node in the electrical circuit. As the input voltage persists, more parallel conductance paths form (Fig. 4(c)). These paths reset under reverse voltage polarity and/or via stochastic thermodynamic fluctuations (not modelled here to preserve interpretability of results).

For arbitrary input voltage signals, the network is only dynamically active in certain parts at different times due to its connectivity. As shown in Fig. 4(d), network activity (defined by edge conductance changes when voltage difference are above a threshold set value V_{set}) gradually increases, reaching a maximum of around 90% before decreasing towards 50%. After 0.5 s, conductance changes slow down in tandem with reduced voltage differences between nodes. With multiple input nodes and more varied, bipolar input signals, the network's behaviour follows similar dynamic patterns, but with richer complexity in its response. Recurrent edge–node interactions enable more nonlinear coupling between inputs and readouts, which may be advantageous for tasks involving multivariate time series with complex dynamics, such as the Lorenz system.

D. Dynamic Reservoir Computing

Due to the presence of memristive edge dynamics which induces complex network dynamics (cf. Fig. 4), the neuromorphic network behaves very differently when compared with classical reservoirs used in standard RC approaches. In conventional RC, edges are random and fixed, and dynamics are introduced solely on the nodes with a prescribed nonlinear activation function σ (such as tanh or ReLU) according to

$$\mathbf{r}(t+\Delta t) = \sigma\left(W\mathbf{r}(t) + W_{\text{in}}\mathbf{u}(t)\right), \tag{15}$$

where W is the static reservoir network weight matrix. In contrast, for neuromorphic reservoirs Eq. (15) is replaced by

$$\mathbf{r}(t+\Delta t) = MW_{\text{in}}\mathbf{u}(t), \tag{16}$$

where $M(t;\mathbf{r}_(t),\mathbf{r}(t-\Delta t),...\mathbf{r}(0);\mathbf{u}(t),\mathbf{u}(t-\Delta t),...,\mathbf{u}(0))$ is a submatrix of L^{-1}, where L is the block matrix in Eq. (9). M has size $N \times N_{\text{in}}$, such that its elements $M_{ij} = L^{-1}_{ik}$, where $k = j+N$ for $i \in [1,N]$, $j \in [1,N_{\text{in}}]$. M is updated at every timestep from the previous $M(t-\Delta t)$ and current input $\mathbf{u}(t)$. Hence, M is dependent on all previous input signals and internal voltage values and encodes all conductance weight changes, via network connectivity and Kirchhoff's laws.

In conventional reservoirs, all reservoir nodes are linearly combined to generate the output signal. In neuromorphic reservoirs, however, due to the limited number of electrodes used in a physical device, only a subset of all reservoir node voltages, the so called readout nodes, is used to generate the output signal. Another key difference when compared to conventional RCs is the presence of adaptive weights. Although the dynamically evolving conductance values on edges cannot be interpreted as neural network weights, since the neuromorphic network circuitry induces coupled edge–node dynamics instead of a matrix multiplication used in abstract neural networks, they serve a similar role as network weights. Indeed, the most important distinction is the different sources of nonlinearity: nonlinear effects in a conventional reservoir stem from the activation function imposed on its nodes, whereas neuromorphic network nonlinear effects derive from its internal physical dynamics (i.e. electro–ionic transport across memristive edges) and, as we show below, its connectivity.

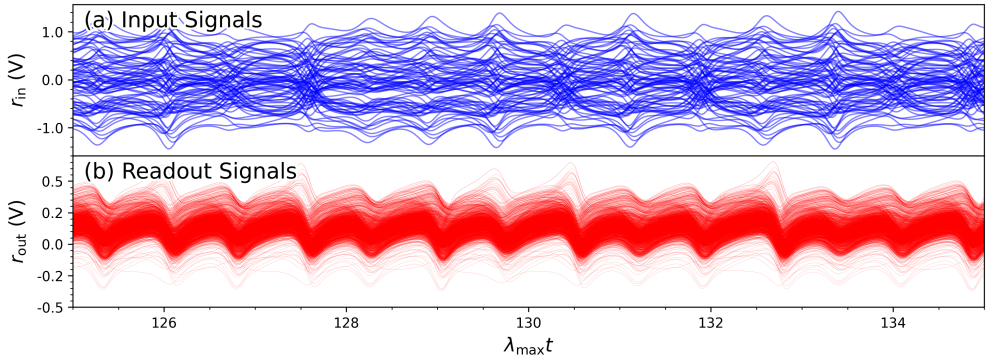

FIG. 5. Input–output mapping of a neuromorphic network with $2,000$ nodes and $47,946$ edges. (a) Input signals $\mathbf{r}_{\text{in}}(t)$ (blue) constructed from a linear combination of the Lorenz system (cf. Eq. (1)) and delivered to 80 randomly selected nodes; (b) readout signals $\mathbf{r}_{\text{out}}(t)$ (red) from $1,919$ randomly selected nodes (distinct from the input nodes). Time duration is the last 10 Lyapunov times of training.

III. RESULTS & DISCUSSION

The results presented below aim to demonstrate how a neuromorphic network works as a physical reservoir and how it is implemented for dynamic RC, using the specific example of autonomous prediction of the Lorenz63 attractor. Both short–term and long–term predictions are presented and analysed, and the influence of external global parameters (input voltage scaling and bias) on forecasting performance is investigated using input–output mappings and a dynamics measure. Nonlinear dynamical features produced by the neuromorphic network are closely examined to determine which features are learned by the auto–regressor used for prediction. How the desirable nonlinear features can be promoted by the set-up of the neuromorphic reservoir is investigated by studying electrode node connectivity and the nanoscale transport dynamics within the memristive edges.

A. Lorenz System Prediction

Fig. 5 shows the input–output mapping of a $2,000$–node, $47,946$–edge neuromorphic network used as a reservoir for dynamic RC for Lorenz63 forecasting. Input and output signals are respectively delivered to and read out from 80 and $1,919$ randomly selected nodes. This network and setup was used to autonomously forecast the Lorenz63 signal, with the best example shown in Fig. 6. The forecast time is $t_f = 9.0$ Lyapunov times, after which the predicted signal deviates but

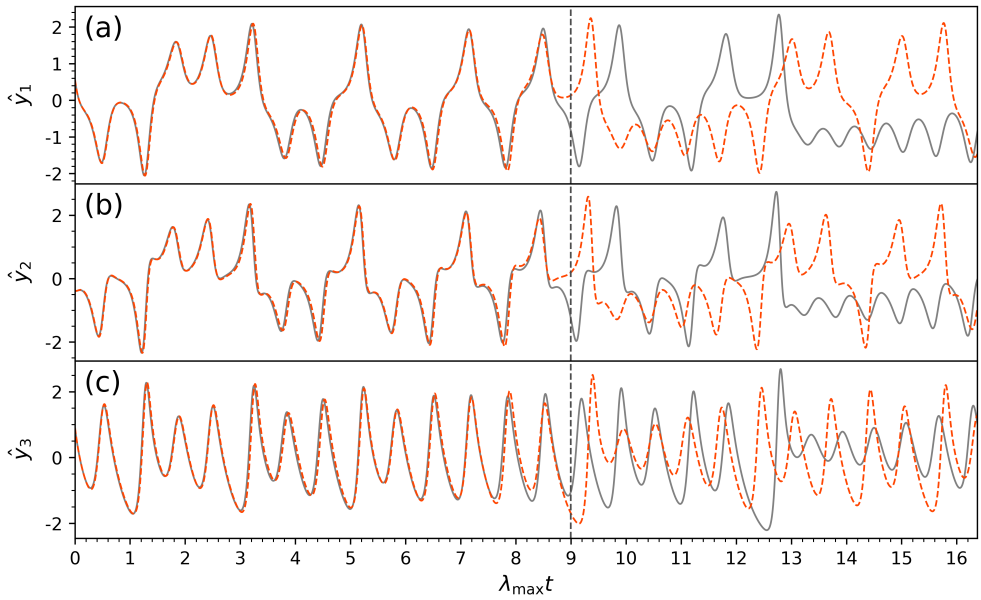

FIG. 6. Example autonomous prediction of the Lorenz system (5) using a neuromorphic network. (a), (b) and (c) respectively show the individual $\hat{y}_1(t)$, $\hat{y}_2(t)$ and $\hat{y}_3(t)$ values for the predicted signals (orange dashes) and the normalised true Lorenz signals (grey curves). The vertical grey dashed lines indicate the short-term prediction length, defined by Eq. (7), as around 9.0 Lyapunov times. Data extracted from the same simulation as in Fig. 5.

still follows the overall dynamics of the Lorenz attractor. An average forecast time of 2.9 ± 1.2 Lyapunov times was found over $1,250$ trials, shown in Fig. 7, indicating the network's general performance in predicting the short-term dynamics of the Lorenz63 system.

With a positive Lyapunov exponent, the difference between the predicted and actual trajectory will always diverge as an unavoidable consequence of chaotic dynamics. Besides forecasting individual trajectories it is desirable to also reproduce the statistical long–term behaviour of the dynamical system. The simulation from Fig. 6 was extended to 100 Lyapunov times to produce the chaotic attractor in Fig. 8; over this long–term timescale, the prediction appears to qualitatively replicate the Lorenz attractor dynamics, and in particular remains stable. For a quantitative assessment of long–term forecasting, the normalised power spectral density (PSD) of the y_3-variable of the Lorenz system is compared in Fig. 9, where a high degree of concordance between the PSDs of the predicted and true Lorenz signals is evident across the majority of the frequency spectrum.

This study utilised a network of $2,000$ nodes, which is relatively large compared with other RC studies[28,29]. However, physical neuromorphic reservoirs typically have network sizes an order

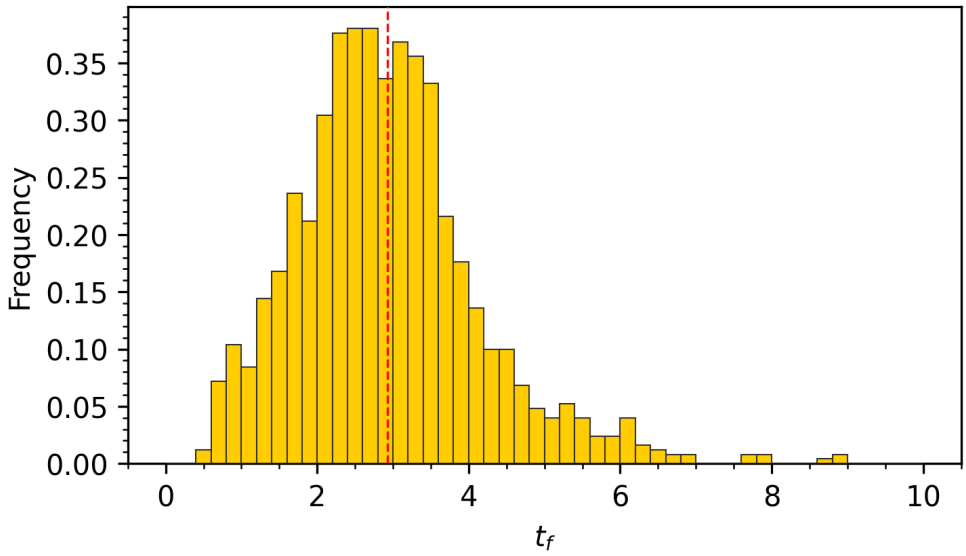

FIG. 7. Histogram of forecast times t_f, defined by Eq. (7), in units of Lyapunov time for a total of $1,250$ trials using the $2,000$ node neuromorphic network. An average of 2.93 Lyapunov times is achieved (indicated by the dotted red line), with a standard deviation of 1.21 Lyapunov times.

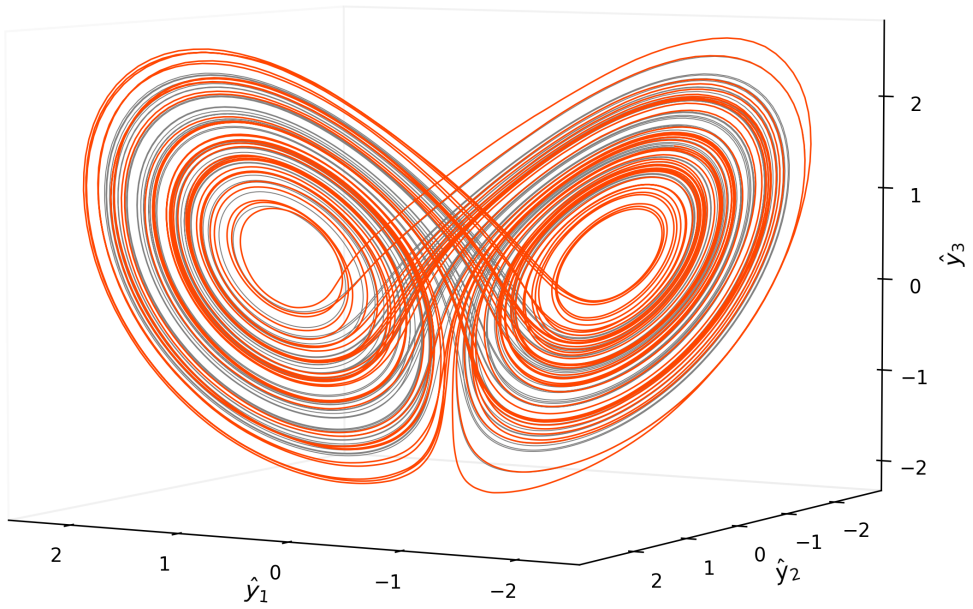

FIG. 8. Predicted (orange) and true (grey) Lorenz attractor trajectory over a forecast length of 50 Lyapunov times. Data extracted from the same simulation as in Fig. 5.

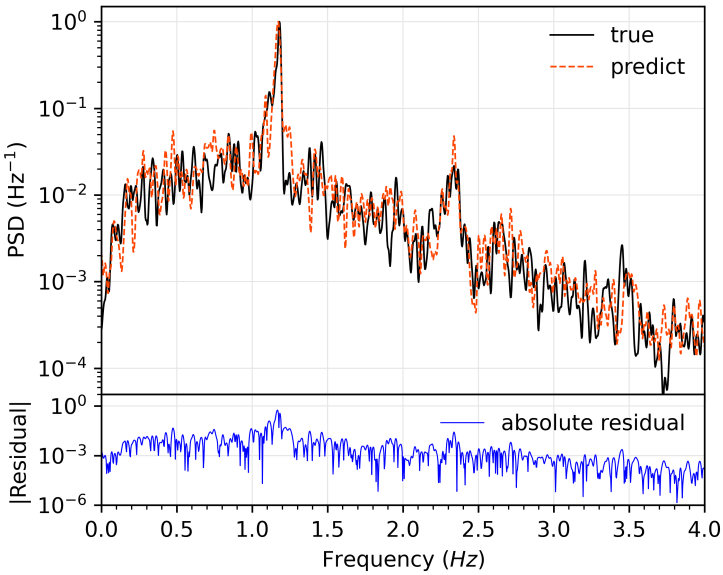

FIG. 9. Normalised smoothed power spectral density (PSD) of the predicted (red) and true signal (black) of the y_3 component of the Lorenz system, and the corresponding absolute residual (blue), over a forecast length of 100 Lyapunov times. Data extracted from the same simulation as in and Fig. 6, during the prediction stage, and smoothed via a Gaussian filter (with a standard deviation of 2).

of magnitude larger, up to millions of physical nodes[7–9,19,45,46], and they are vastly more energy efficient than their software simulator counterparts. Due to the network size and the necessity to model each dynamical interaction within $\mathscr{N}$, the simulations can be more compute intensive when compared with standard RC methods. However, when viewed as a simulation of a physical system, the results are valuable for optimising the design of physical neuromorphic RC systems for complex tasks such as multivariate time series forecasting. For example, a meta–learning scheme can be implemented for meta–parameter optimisation (e.g. W_{in}, $\mathbf{b}_{\text{in}}$) using methods such as simulated annealing[46]. Note that in real world applications, all nonlinear information processing would be completed implicitly by the physics of the neuromorphic device itself; the only computation necessary would be linear regression on the single output layer.

B. Induced Nonlinear Dynamics

From Sec. II D and Eq. (16), it is evident that the neuromorphic network's nonlinearity and thus functionality stems from its memristive edge dynamics; hence, network performance is expected to be influenced by dynamical activity.

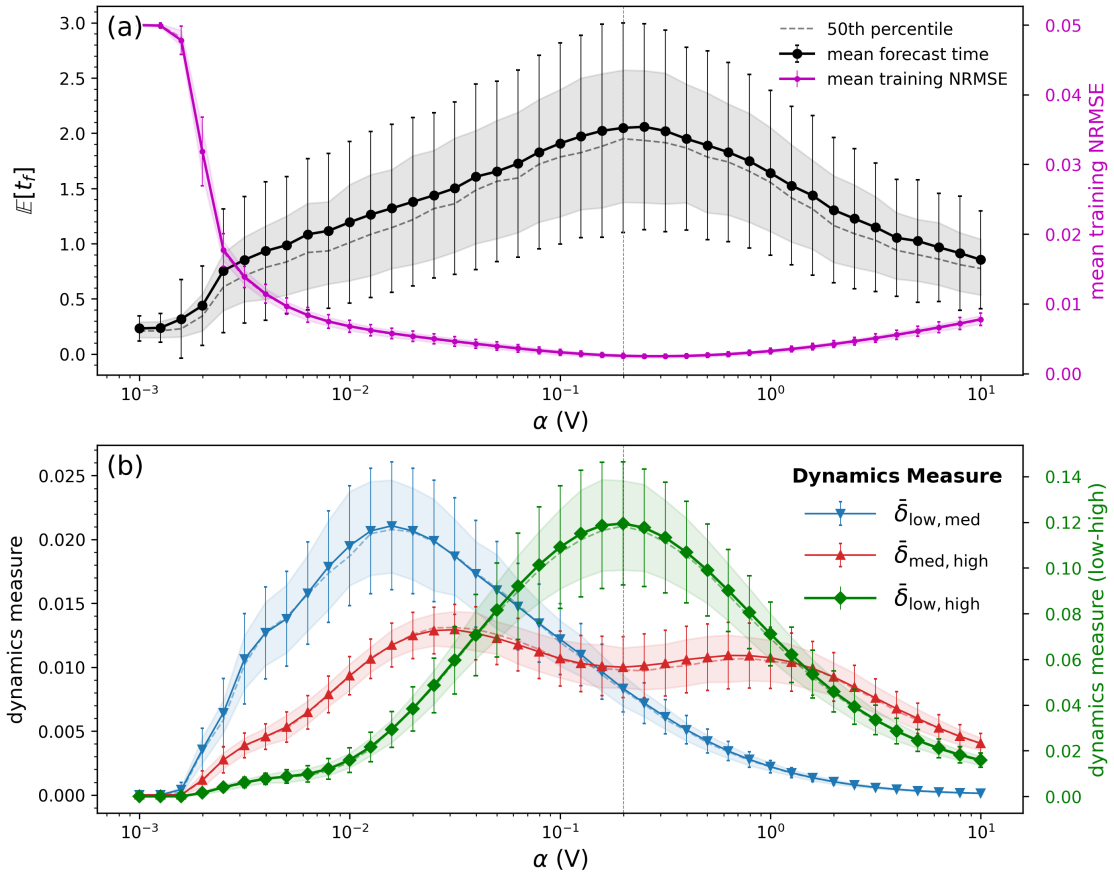

FIG. 10. Influence of input scaling parameter α on forecasting performance and reservoir dynamics. (a) Mean forecast times (black) and mean training NRMSE (pink) as a function of α; (b) corresponding dynamics measure (as defined in Eq. (18)). As the low-to-high conductance state changes are one order of magnitude more frequent, the corresponding dynamics measure (green) is shown on a different scale. The dotted line at $\alpha = 0.2\,\text{V}$ indicates the default α used in previous figures. Simulations were performed using a neuromorphic network with 500 nodes and 9,905 edges. Each data point in (a) was obtained from 1,500 trials, and 100 trials for (b).

From Eqs. (10) and (11), conductance evolution is driven by the voltage difference across memristive edges. On a network level, this implies that nonlinear dynamics can be directly controlled (and hence optimised) by the external voltage amplitude α of the input signal. Fig. 10(a) shows the mean of the forecast times, $\mathbb{E}[t_f]$, as a function of α. The mean forecast time exhibits a clear peak at $\alpha \approx 0.2\,\text{V}$. In Fig. 10(b) it is seen that the training error achieves its minimal Normalised Root Mean Square Error (NRMSE) at the same value of $\alpha \approx 0.2\,\text{V}$. This supports our choice for the default value $\alpha = 0.2\text{V}$ used in the simulations.

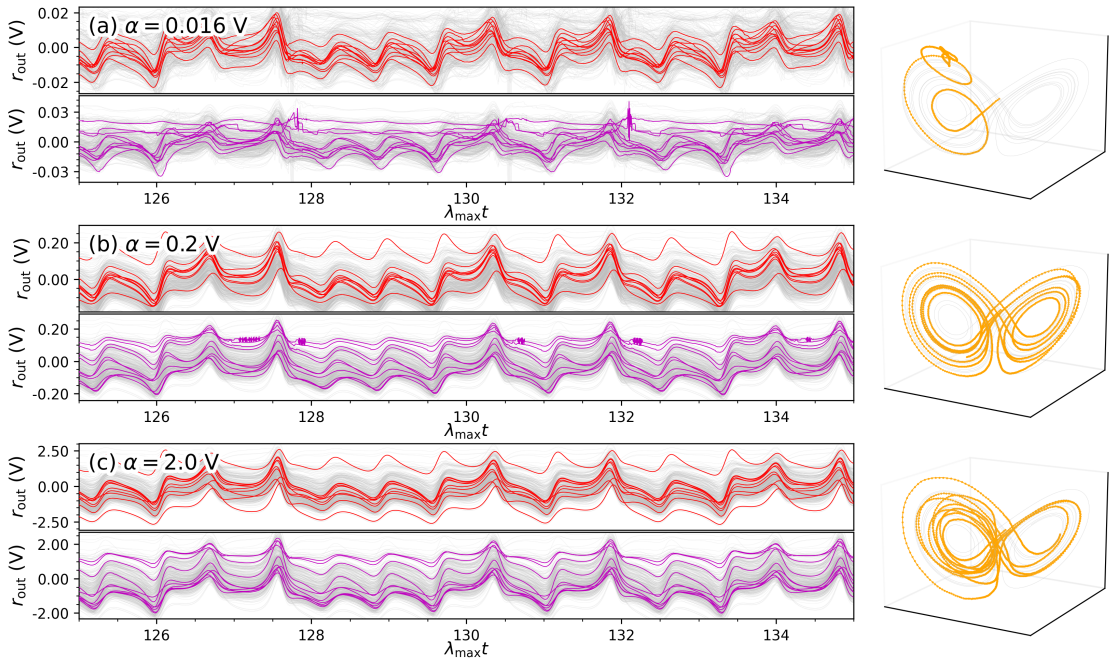

FIG. 11. Representative behaviour of readout signals with high (red) and low (magenta) ω values (W_{out} weightings), and its corresponding predicted attractor (orange) with respect to input scaling parameters α, at values of (a) 0.016 V, (b) 0.2 V, (c) 0.4 V and (d) 2.0 V. The forecast times are $t_f = 1.30$, $t_f = 1.64$ and $t_f = 1.69$ Lyapunov times, respectively. The corresponding attractors evolve for 10 Lyapunov times.

As shown in Sec. II D, nonlinearity within this system is a consequence of the memristive edge dynamics. To quantify this, a *dynamics measure* is introduced, based on changes in the conductance states of each memristive edge over a dynamical timescale of one Lyapunov time; high, medium and low conductance state changes were counted over time (cf. Fig. 3). Formally, we define the dynamics measure $\delta_{a,b}^{i,j}$ for counting the number of transitions between conductance states X_a and X_b across the memristive edge between node i and j as

$$\delta_{a,b}^{i,j} = \frac{1}{T}\sum_{\ell}\left([|x_{ij}(t_\ell)| \in X_a][|x_{ij}(t_{\ell+1})| \in X_b] + [|x_{ij}(t_\ell)| \in X_b][|x_{ij}(t_{\ell+1})| \in X_a]\right), \tag{17}$$

across all time t_ℓ during training through increments of Lyapunov times (i.e. $t_{\ell+1} - t_\ell = \lambda_{\max}^{-1}$), normalised by the total number of training timesteps T. Here $[\cdot]$ denote Iverson brackets, where $[P]$ is defined as 1 if statement P is true, and 0 otherwise. The three conductance states are determined by their corresponding x values, with intervals $X_{\text{low}} = [0, 0.55)$, $X_{\text{med}} = [0.55, 2/3]$ and $X_{\text{high}} = (2/3, 1]$ for the low, medium and high conductance states, respectively. Averaging across all K

edges in the network, the overall network dynamic measure $\bar{\delta}_{a,b}$ is thus

$$\bar{\delta}_{a,b} = \frac{1}{K}\sum_{i,j}\delta_{a,b}^{i,j}. \tag{18}$$

The network dynamics measure is shown in Fig. 10(b). The low–med and med–high conductance changes represented by $\bar{\delta}_{\text{low,med}}$ and $\bar{\delta}_{\text{med,high}}$ capture dynamic activity between intermediate states, and are maximal at around $\alpha = 10^{-2}\,\text{V}$ (with med–high state changes also persistent at $\alpha \approx 1\,\text{V}$). This is in fact near the onset of nonlinear dynamics, as memristive edges only evolve at a voltage greater than $V_{\text{set}} = 10^{-2}\,\text{V}$. The low–high conductance changes represented by $\bar{\delta}_{\text{low,high}}$ capture persistent network–scale dynamic states, corresponding to continuous formation and decay of high conductance pathways; its maximal value at $\alpha \approx 0.2\,\text{V}$ also corresponds to the highest mean forecast time shown in Fig. 10(a). This suggests that optimal task performance corresponds to large dynamical network activity that sweeps across many internal edge states. Next, we consider what this means on a practical level for performing dynamic RC.

The output layer weights W_{out} learn nonlinear dynamical features that are most useful for predicting the Lorenz system. Each ith column of W_{out} includes the output weighting attributed to the features produced by readout node i. Denoting the absolute sum over the ith column of W_{out} as ω_i, where $\omega_i = \sum_{j=1}^{N_{\text{out}}} |W_{\text{out}}^{j,i}|$, a high ω_i implies node i produces more useful features, while low ω_i means that the ith feature is suppressed. Fig. 11 shows examples of readout voltages during the last 10 Lyapunov times of training for three different values of α, with readout signals extracted from nodes i with the highest ten ω_i values shown in the top panel (red), and readout signals from the smallest (near zero) ten ω_i values below (magenta), with their corresponding autonomously predicted Lorenz attractor (orange) pictured on the right. It is clear that all cases in Fig. 11 suppress readout signals with high frequency fluctuations, suggesting that such nonlinear features are not useful for learning the Lorenz63 system.

The nonlinear features across different α also show qualitative differences. In Fig. 11(a), the value of α was chosen to match the onset of nonlinearity, at the maxima of the low–med dynamics measure with $\alpha = 0.016\text{V}$; this is also the case with the largest amount of high frequency fluctuations overall. These fluctuations are not as evident in the two cases (Fig. 11(b) and (c)), with $\alpha = 0.2\,\text{V}$ chosen to match the optimal forecasting time, and with a high value of $\alpha = 2.0\,\text{V}$ which exhibits relatively smoother readouts. These are produced from the persistent dynamics in the formation and decay of conductance paths within the network, shifting the weights within the M matrix gradually at every timestep. The memristive edges evolve across all available conduc-

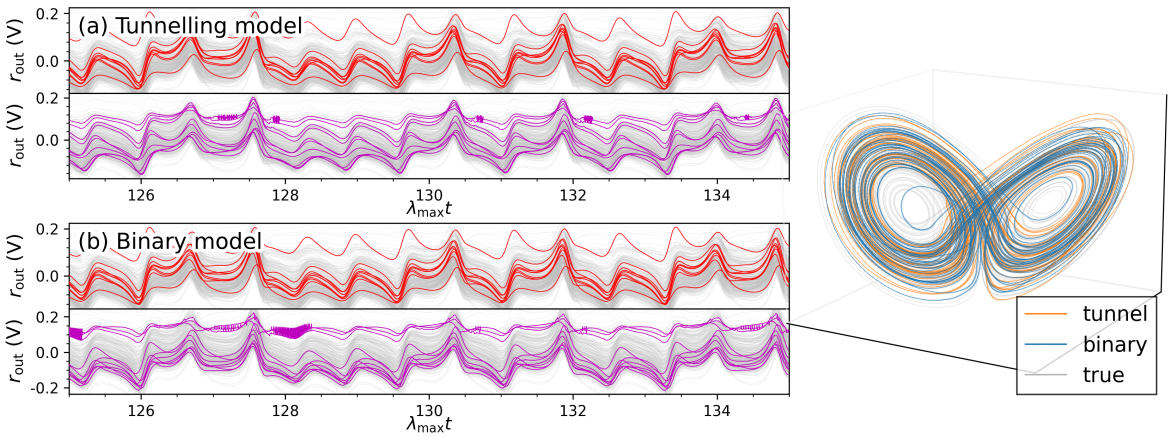

FIG. 12. Representative readout signals with high (red) and low (magenta) ω weightings, for two memristive models: (a) tunnelling (corresponding to the same plot for $\alpha = 0.2\,\mathrm{V}$ in Fig. 11) and (b) binary. Forecast times are $t_f = 1.64$ and $t_f = 1.66$ Lyapunov times, respectively. The corresponding attractors (right) evolve for 50 Lyapunov times.

tance states (low through medium to high conductance state, and back), thereby maximising the use of available internal degrees of freedom. This favourable nonlinearity is conducive for good network performance, whereas the high frequency fluctuations represent less useful nonlinearity that is suppressed by W_{out}. The corresponding predicted attractor in each case shows that the most useful nonlinear dynamical features are produced away from very low and very high values of α, where the predicted attractor becomes unstable.

C. Memristor Transport Dynamics

The memristor model, Eq. (11), utilised for this study includes electron tunnelling transport in addition to ballistic transport. As shown in Fig. 3, tunnelling introduces intermediate conductance states that do not exist in the simpler binary model. Knowing that the high frequency fluctuations arise from low to medium conductance state changes (as previously seen in Fig. 11(a)), it is reasonable to question whether they are caused by tunnelling, that is, if the neuromorphic network's capabilities are limited by the inherent physics of the system.

Fig. 12 reveals, however, that the high frequency fluctuations are even more pronounced in the binary model. The shown examples of both models (simulated with the same parameters) share similar below–average forecast times of $t_f \approx 1.6$ Lyapunov times, and their predicted attractors

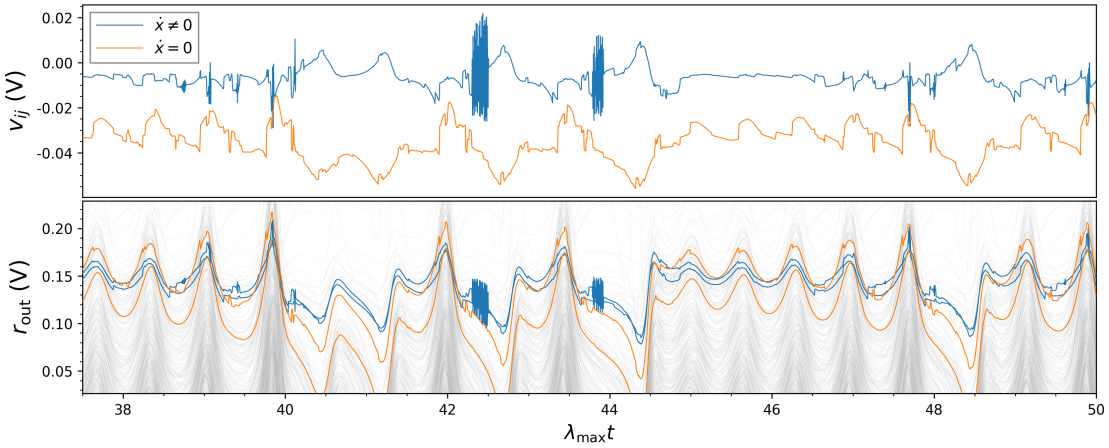

FIG. 13. Examples of (top) memristive edge voltages with and without dynamic activity ($\dot{x} \neq 0$ and $\dot{x} = 0$, respectively) and (bottom) their corresponding node voltage readouts, showing that high-frequency nonlinear features may be present even in nodes connected to edges that are not evolving (orange), as compared to a typical nonlinear voltage signal (blue). All simulation parameters are the same as in Fig. 11 with $\alpha = 0.2\,\mathrm{V}$.

exhibit similar long–term dynamics (up to to 50 Lyapunov times); both successfully reproduce the Lorenz dynamics, only deviating in regions with relatively little training data (at the edges of the attractor). Thus, the high-frequency nonlinear features do not originate from tunnelling dynamics *per se*, but rather appear to arise as a result of rapid switching of conductance states, fluctuating around the voltage threshold V_{set} for the onset of memristive dynamics. Moreover, the observation that the rapid fluctuations are more prominent in the binary model suggests that the tunnelling model is able to naturally suppress these features, and this may be attributed to the many more degrees of freedom afforded by the intermediate states of tunnelling dynamics (see Fig. A.3 in Appendix). Nevertheless, both models still yield similar average short–term prediction accuracy (see Fig. A.6 in Appendix) due to the suppression of these features via ridge regression in the output layer. See Appendix B for further details on how these high-frequency fluctuations arise from memristive edges in both models.

The intermediate conductance state changes (low–med, med–high) do not alone explain all fluctuating dynamics in the network; by studying dynamics at the individual memristive edge level, these high frequency fluctuations can also be observed in nodes connected to edges which have no dynamical activity. This is shown in Fig. 13, where high frequency fluctuations can also be observed in edges without any dynamics (i.e. $\dot{x}_{ij} = 0$ at edge between node i and node

j). For a more detailed analysis, see Appendix B. In summary, these high frequency dynamical features are caused in part by the network connectivity distributing nonlinear dynamics amongst all connected parts, effectively acting as a multiplier of nonlinearity, hence allowing network nodes to exhibit nonlinear dynamics even when corresponding edges have no internal dynamical activity themselves.

Overall, nonlinear dynamical features are induced by the complex interactions between memristive edge evolution and network connectivity, resulting in the formation and destruction of conductance paths and feedback loops which cannot be easily inferred unless fully numerically simulated (since a closed-form solution does not exist). From these results, it is expected that proximity of non-active network components to dynamically active components should affect how dynamical features are generated and distributed throughout the network; this is explored next.

D. Node Interactions

Local node interactions are influenced by the input electrode node placements. This section investigates how proximity of readouts to input electrodes affects nonlinear dynamics, and how it may be possible to infer input node allocations that promote good forecasting performance.

To aid analysis and visualisation of node interactions, a homogeneous square 2D lattice network was constructed. This network consists of $25 \times 25 = 625$ nodes, with each node connected to 4 other nodes (implying $1,250$ edges), with doubly periodic boundary conditions.

We introduce a *nonlinearity measure*, $m^{i,j}_{\text{nonlin}}$, defined as the mean squared difference between the node readouts $r^{i,j}_{\text{out}}$ of the nonlinear model and the readouts $r^{L;i,j}_{\text{out}}$ of the closest linear model, normalised by the mean squared readout signals,

$$m^{i,j}_{\text{nonlin}} = \frac{\left\langle \left| r^{i,j}_{\text{out}} - r^{L;i,j}_{\text{out}} \right|^2 \right\rangle}{\langle ||\mathbf{r}_{\text{out}}||^2 \rangle}. \tag{19}$$

On the overall network level, this nonlinearity measure $\bar{m}_{\text{nonlin}}$ would be

$$\bar{m}_{\text{nonlin}} = \frac{\langle ||\mathbf{r}_{\text{out}} - \mathbf{r}^L_{\text{out}}||^2 \rangle}{\langle ||\mathbf{r}_{\text{out}}||^2 \rangle}. \tag{20}$$

The readout $\mathbf{r}^L_{\text{out}}$ is obtained from a linear model for which the internal state variables are static with $\dot{\mathbf{x}} = 0$ for all memristive junctions, effectively rendering them linear resistors. Consequentially for very small values of $\alpha < 0.001\,\text{V}$, $\bar{m}_{\text{nonlin}} = 0$ as voltage signals is too small to form any

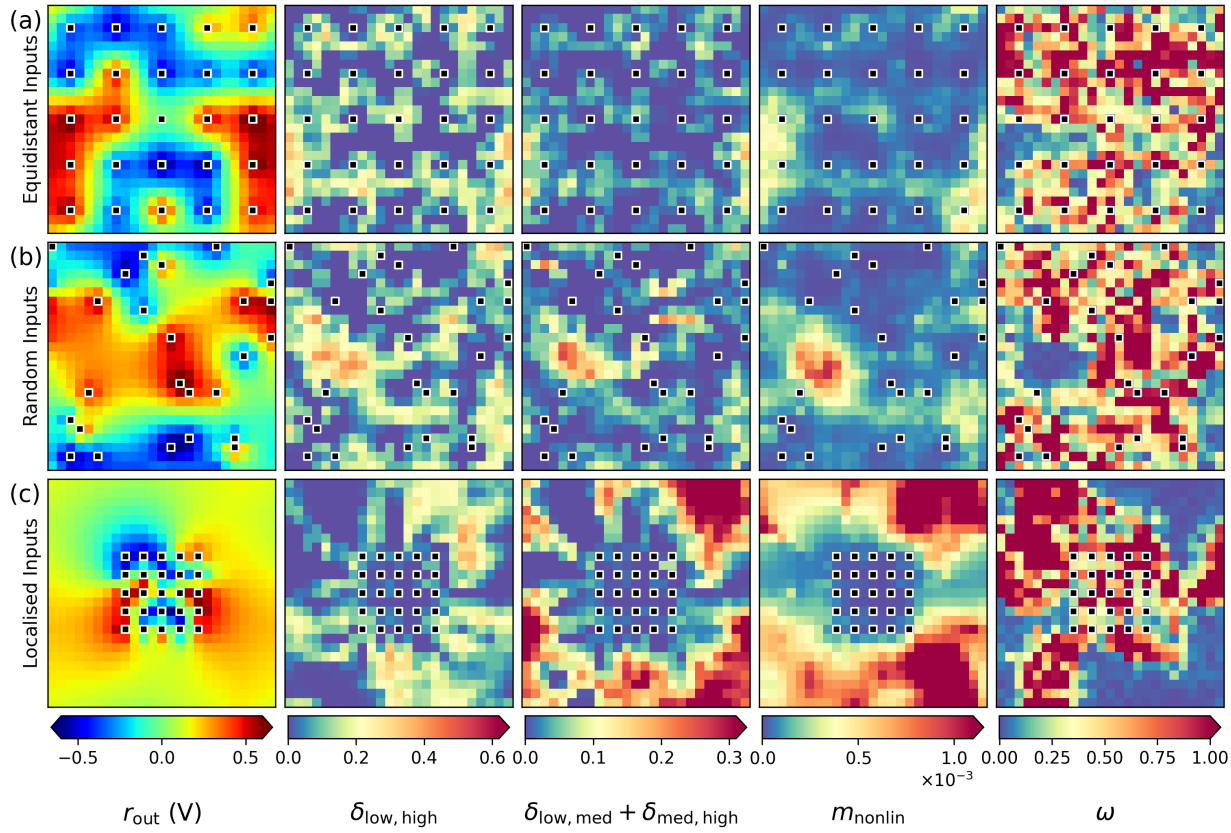

FIG. 14. Heat map of selected time-averaged measures on a 2D 25×25 doubly periodic lattice network for different input electrode node placements: (a) uniformly equidistant inputs, (b) randomly selected inputs, and (c) localised inputs. The first column shows the readout voltages r_{out} averaged over time. The second column shows the dynamics measures $\delta_{\text{low,high}}$ for conductance state transitions between low and high conductance states, the third column shows dynamics measures $\delta_{\text{low,med}} + \delta_{\text{med,high}}$ for low-med and mid-high conductance states changes (defined in Eq. (17)), the fourth column shows the nonlinearity measure m_{nonlin} (defined in Eq. (19)), and the fifth column shows the corresponding ω weights. Indices omitted in all labels for clarity.

conductance paths (i.e. $\dot{\mathbf{x}} = 0$). Likewise for large values of $\alpha > 10\,\text{V}$, the corresponding $\bar{m}_{\text{nonlin}}$ are also approximately zero, as all memristive edges are oversaturated with voltage, effectively also resulting in a near linear network with $\dot{\mathbf{x}} \approx 0$ (see Fig. A.4 in Appendix).

Fig. 14 shows heat maps of the time–averaged readout voltages (column 1), along with the low–high dynamics measure (column 2), the sum of the low-med and med-high dynamics measures (column 3), the nonlinearity measure defined in Eq. (19) (column 4), and weight matrix ω values (column 5), with each pixel value representing a node that respects its relative location

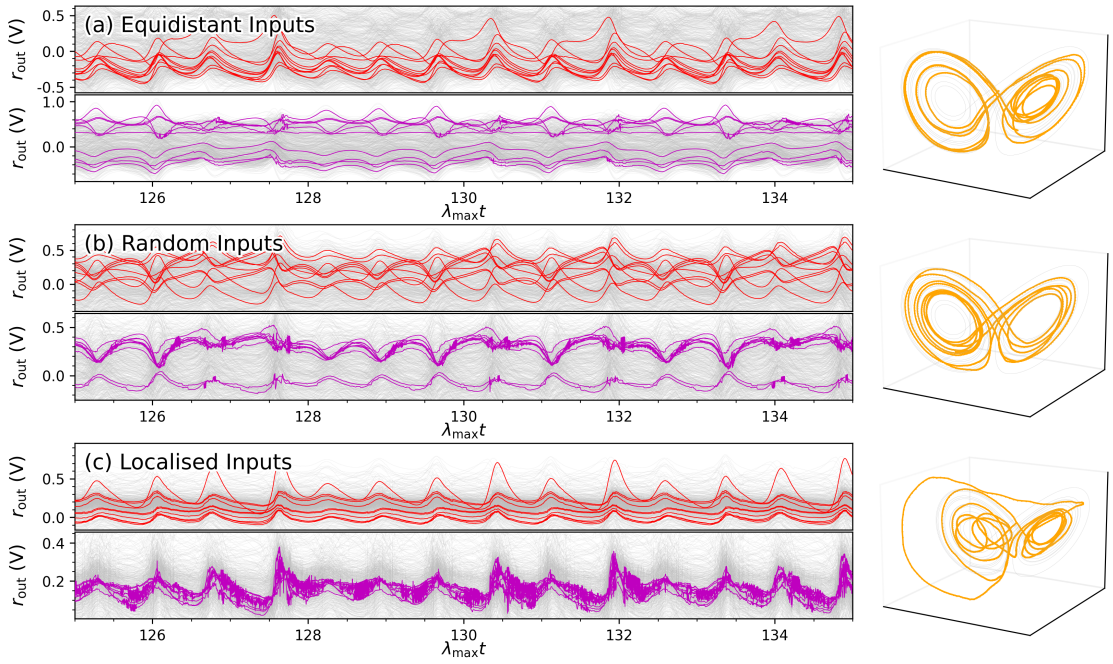

FIG. 15. Representative readout signals with high (red) and low (magenta) ω weightings, and their corresponding attractors (orange), for different input node placements, for the experiments on the doubly periodic lattice shown in Fig. 14. (a) uniformly spaced inputs, (b) randomly selected inputs and (c) localised inputs. The respective forecast times are $t_f = 3.41$, $t_f = 1.68$ and $t_f = 1.58$ Lyapunov times. The corresponding attractors evolve for 10 Lyapunov times.

on the lattice network. Fig. 14(a) uses uniformly distributed, equidistant input electrode nodes, while Fig. 14(b) and (c) use randomly placed and localised input nodes, respectively. Comparison of these heat maps suggests that the nonlinearity measure is largely correlated with the dynamics measures (especially the low-med and med-high dynamics), and that nonlinearity is anti-correlated with ω_i. This anti-correlation implies suppression of highly nonlinear and certain highly dynamical readout nodes by the output layer, which means on a local level some highly nonlinear regions are not useful for the prediction task.

Fig. 14 reveals that nonlinear regions appear mostly in regions which are far away from input electrode nodes. This is more obvious in Fig. 14(c), where by clustering all input nodes together in one region, it becomes clear that the nonlinear parts are the areas away from input nodes. The anti-correlation between ω_i and dynamics measures is also clear in this example, where the highest ω_i weights are mostly in areas near the input nodes. Every input electrode node has its own area of influence, and although this can be affected by other input nodes, on average the further away a

readout node is to this input, the less influence this input has on the readout. These far-away nodes dynamically behave in a manner increasingly more different from those of the input electrode nodes.

Note, however, that this anti-correlation is not exact, nor is the correlation between the nonlinearity and dynamics. This is due to the aforementioned network connectivity effects; it is most obvious in the localised input example, where there exists regions with relatively little dynamics but high nonlinearity, and these regions do appear smooth and connected, tending to distribute amongst close neighbouring network components.

The suppressed highly nonlinear nodes also exhibit high frequency fluctuations in their readout signals, which is the unfavourable nonlinearity encountered previously in Fig. 11. Fig. 15 shows the corresponding readout signals of the above three examples in Fig. 14, with the readout signals corresponding with the largest (red) and smallest (magenta) ten ω_i weights both emphasised. As seen previously in Fig. 11, high frequency fluctuations can be observed in Fig. 15 in signals with small ω_i values. As expected, these suppressed nonlinear features are also much more prominent in Fig. 15(c) with the clustered inputs. This implies the unwanted nonlinear readouts arise from nodes distant from the inputs, restricting them to only evolve between the low and medium conductance states as indicated by the low-med dynamic measure, which correlates with the appearance of nonlinear signal that exhibit high frequency fluctuations.

Due to lower presence of unwanted nonlinear features, the examples with random and equidistant inputs both perform better than the clustered input example, as seen from the Lorenz attractor predictions in Fig. 15. This impact of input node placement remains true for the heterogeneous neuromorphic network (see Fig. A.5 in Appendix). Note that with a high network density, randomly selecting nodes leads to an input electrode node placement in which nodes are on average separated by the same number of edges. Even in the sparse lattice network structure there are only minor differences between equidistant and the randomly selected input nodes (cf. Fig. 14(a) and Fig. 14(b)).

To maximise the area of influence of each readout node over time, and hence minimising the occurrence of less useful nonlinear features, the input voltage should remain sufficiently large on average. This can be achieved with the input bias voltages. Instead of sampling values in $\mathbf{b}_{\text{in}}$ uniformly from $[-b, b]$, sampling from $[-b, -b/2] \cup [b/2, b]$ ensures that no input node will have low voltages. On average this does have a noticeable effect in the Lorenz prediction task, where for the 500 node network an average forecast time of 2.12 Lyapunov times can be observed with

the new bias sampling, as opposed to 1.95 when the bias is sampled form the whole range $[-b, b]$ (cf. Fig. A.7 in Appendix).

In summary, proper input electrode node number and placement allows maximal usage of the given network, by ensuring all readouts are within the area of influence of the inputs such that less useful nonlinear features are minimised. Practically, in a sufficiently dense network, random allocation of input electrode nodes is very similar to evenly distributed input nodes. This motivates our choice to choose the number of input nodes as around 5% of all available nodes - with roughly evenly spaced input nodes, 5% is sufficient to activate and continuously sustain network dynamics.

IV. CONCLUSIONS

These results validate the hypothesis that the internal and network dynamics of memristive networks can be used in a reservoir computing approach to learn complex chaotic dynamics. It was found that the network dynamics can be optimised for predicting the Lorenz attractor by appropriate scaling of the external input voltage to values that are not too low or too high. Optimal intermediate values of input voltage scaling were found to drive internal memristive states that traversed the entire dynamical range, from low to high conductance states, thereby maximising use of available internal degrees of freedom. Interestingly, this study also revealed that some nonlinear dynamical features, such as high frequency fluctuations, were effectively filtered out by ridge regression in the external output layer. This suggests that physical reservoir computing may be robust against noise and variations due to the specific internal dynamics of the memristive material system. On the other hand, our results also indicate that it is possible to suppress such features by ensuring electrodes provide sufficient coverage of network nodes to sustain dynamic activity across the memristive edges.

Overall, this study revealed how the nonlinear dynamics of memristive neuromorphic networks can be controlled and optimised for reservoir computing solely via adjusting the external input layer parameters. This is promising for practical implementations of physical reservoir computing with memristive networks, which have orders of magnitude more physical nodes and memristive edges[8,39]. Ongoing work is also exploring the optimal network structure[26], output layer and computational frameworks beyond standard reservoir computing, to be eventually applied to real world applications on noisy, unstructured data, thereby unlocking the full potential of neuromorphic networks for modelling dynamical systems via the dynamics of the physical network.

ACKNOWLEDGMENTS

The authors acknowledge use of the Artemis high performance computing resource at the Sydney Informatics Hub, a Core Research Facility of the University of Sydney. Y.X. is supported by an Australian Government Research Training Program (RTP) Scholarship. G.A.G. acknowledges support from the Australian Research Council under Grant No. DP220100931.

DATA AVAILABILITY

The data that support the findings of this study may be generated with the code available at `https://github.com/xvyh/dynamicRC_NWN`.

Appendix A: Parameters for the Memristor Model

The parameters ζ_0 and ζ_1 in Eq. (12) are given by

$$\zeta_0 = A\frac{3(2m_*)^{1/2}e^{5/2}(\phi/e)^{1/2}}{2h^2}, \tag{A1}$$

$$\zeta_1 = -\frac{4\pi(2m_*e)^{1/2}}{h}\left(\frac{\phi}{e}\right)^{1/2}, \tag{A2}$$

and are comprised entirely of physical parameters of the nanowires as described in Ref.[19]. The largest and smallest resistances of the memristors are respectively assigned the values of $R_{\text{off}} = 12.9\times10^6\,\Omega$, and $R_{\text{on}} = 12.9\times10^3\,\Omega$. The memristor voltage needed to start the evolution of x is $V_{\text{set}} = 0.01\,\text{V}$, and the voltage to reset the memristor state is $V_{\text{reset}} = 0.005\,\text{V}$. The largest flux is $\xi_{\text{max}} = 0.015\,\text{Vs}$, the normalised critical flux value is $x_{\text{crit}} = 2/3$. The thickness of the electrically insulating, but ionically conducting polymer layer between nanowires is $s_{\text{max}} = 5\,\text{nm}$. The boost parameter in Eq. (10) is $q = 10$, and the global $\dot{x}$ scaling parameter $\eta = \eta_0/\xi_{\text{max}}$ is determined with $\eta_0 = 10$.

Appendix B: Nonlinear Responses in Memristive Edges

From Eq. (10) it is evident that x will only increase in amplitude when the voltage amplitude at the corresponding memristive edge crosses V_{set}, and decrease in amplitude when it crosses V_{reset}. This will occur in two scenarios: if the maximum edge voltage signal (across time) of the

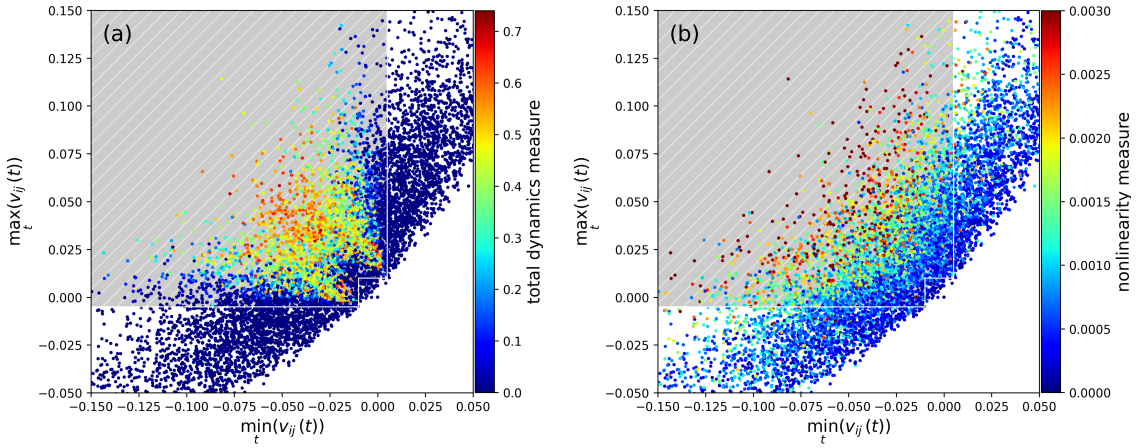

FIG. A.1. (a) Total dynamics measure (i.e. $\delta^{i,j}_{\text{low,med}} + \delta^{i,j}_{\text{med,high}} + \delta^{i,j}_{\text{low,high}}$) and (b) nonlinearity measure $m^{i,j}_{\text{nonlin}}$ with respect to the maximum and minimum of each corresponding memristive edge voltage signal across time. The shaded regions are where edge voltage signal amplitudes can cross V_{set} and V_{reset}, i.e. where $\dot{x}_{ij}$ may be nonzero. All data points are obtained from the same experiment for $\alpha = 0.2\,\text{V}$ in Fig. 11. Edges connected to the input nodes are omitted, and the figures are zoomed in for clarity.

edge between node i and node j satisfies $\max_t(v_{ij}(t)) > V_{\text{set}}$, while the minimum edge voltage satisfies $\min_t(v_{ij}(t)) < V_{\text{reset}}$, or $\max_t(v_{ij}(t)) > -V_{\text{reset}}$ and $\min_t(v_{ij}(t)) < -V_{\text{set}}$. This results in two overlapping regions where dynamical activity can be observed. Indeed as seen in Fig. A.1(a), the dynamic measure is only nonzero within this specified (shaded) region, and it is clear that larger ranges (i.e. larger $\max_t(v_{ij}(t))$ and smaller $\min_t(v_{ij}(t))$) of edge voltages which centre near 0 tend to produce higher dynamics, while those near the boundary of dynamic activity (i.e. boundary of the shaded region) produce less.

However note that in Fig. A.1(b), although nonlinearity is maximal and most frequent within the specified region, there exists numerous highly nonlinear edges outside this region. The fact that nonlinearity can exist in edges which do not dynamically evolve implies the network connectivity has a significant effect on the overall nonlinear dynamics, and memristive edge dynamics is only a fraction of the cause and perpetuation of nonlinear effects. In fact, the network has the largest nonlinearity measure near α values with the most network induced nonlinearity (see Fig. A.4). Since only a small portion of memristive edges are evolving actively in Fig. A.1, the network appears to be underutilised; future studies (beyond simply changing the input layer) should optimise parameters which maximise the number of edges within this dynamically active region.

It has been identified that both network and memristive dynamics causes nonlinearity. To in-

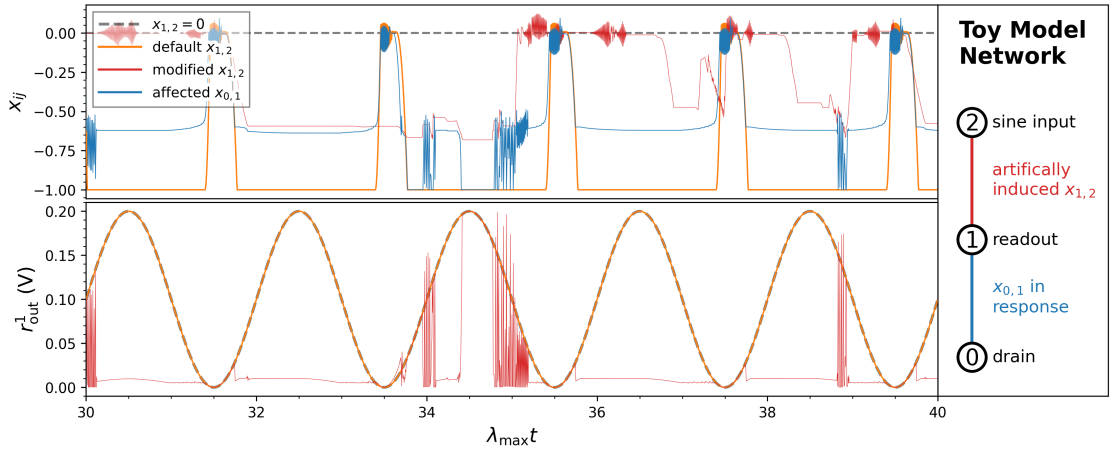

FIG. A.2. By artificially inducing a highly nonlinear (top, red) x signal into the edge between node 2 and 1 of this simple 3-node toy model, unwanted nonlinear features (bottom, red) can be observed in the readouts of node 1, which are vastly different from the original (orange) signal. This also affect the x (top, blue) signals of the edge between node 1 and 0. These memristive edges uses the tunnelling model.

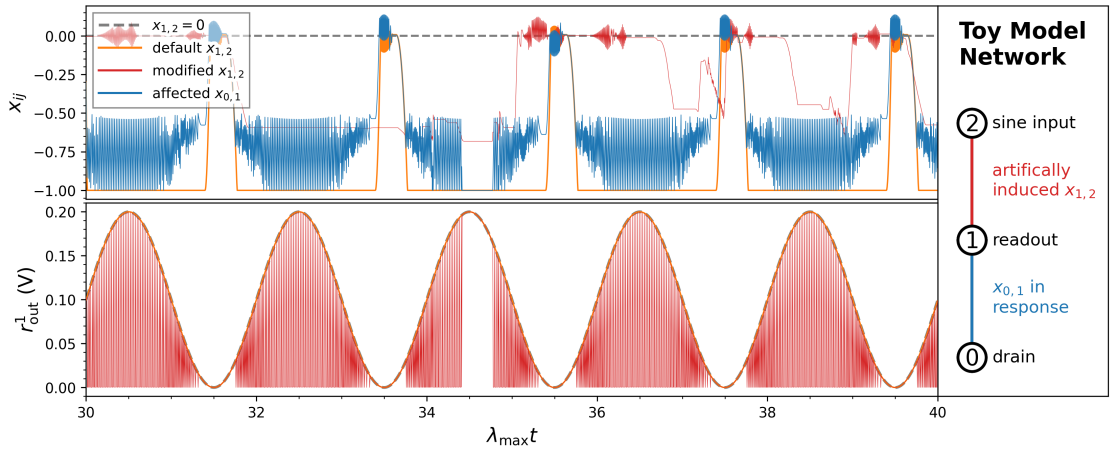

FIG. A.3. By artificially inducing a highly nonlinear (top, red) x signal into the edge between node 2 and 1 of this simple 3-node toy model, unwanted nonlinear features (bottom, red) can be observed in the readouts of node 1, which are vastly different from the original (orange) signal. This also affect the x (top, blue) signals of the edge between node 1 and 0. Unlike Fig. A.2, these memristive edges uses the binary model.

vestigate its exact origin, a simple 3-node toy model was constructed, with node 0 serving as the drain (ground) node, node 1 as readout and node 2 as the input electrode node (see schematic in Fig. A.2), which takes in a sine wave (shifted away from zero) as input. A highly nonlinear x signal (extracted from the edge with the highest nonlinearity measure from Fig. A.1) is artificially

injected into the $1-2$ edge. Both the modified and original $x_{1,2}$ values can be seen in Fig. A.2, where the modified $x_{1,2}$ display the classic behaviour of a memristive with high low-med dynamics measure as it never reaches the high conductance state. High frequency fluctuations can only be observed in the readouts with the modified x instance, confirming these nonlinear features are directly caused by these types of memristive edge dynamics.

Performing the same analysis on the 3-node network with a binary memristor model yields vastly different results. As seen in Fig. A.3, despite having the same modified $x_{1,2}$ signal as Fig. A.2, the high frequency fluctuations in the readouts are much more frequent, suggesting that the more physically realistic tunnelling memristor model may be more robust, and produce fewer high-frequency fluctuations when compared with the simpler binary model.

Interestingly, the observed fluctuations are bounded between two values: the original sine wave readout and a constant zero signal (from the drain node). This is due to conductance paths (specifically $g_{0,1}$ and $g_{1,2}$) rapidly switching in amplitude across time, favouring one signal over the other at different times; in the actual network the high frequency fluctuations would be bounded by numerous other readouts in close proximity, as opposed to just two signals in this simple example. The gap at 34.5 Lyapunov times in Fig. A.3 reveals how these fluctuations disappear when $|x_{1,2}| > x_{\text{crit}}$, which explains the absence of high frequency fluctuations with large low-high dynamics measures. Both Fig. A.2 and A.3 shows examples of how nonlinear responses in $x_{1,2}$ can cause similar behaviours in the neighbouring $x_{0,1}$ values, illustrating the spread of similar memristive dynamics within the network which has potential to cause observed nonlinear effects in the readouts.

Appendix C: Supporting Figures

Figs. A.4–A.7 are referred to in the main text.

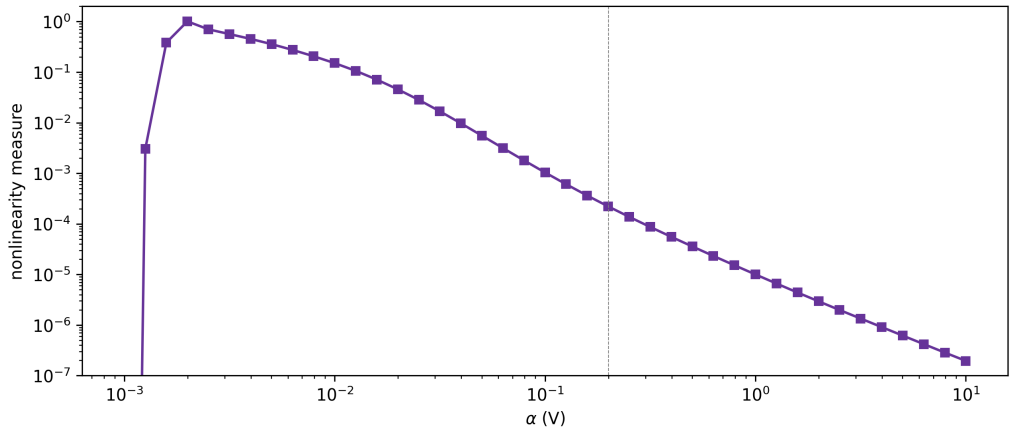

FIG. A.4. The average nonlinearity measure $\bar{m}_{\text{nonlin}}$ (defined in Eq. (20)) as a function of the scaling parameter α. The simulations were performed on a neuromorphic network with 500 nodes and $9,905$ edges. The dotted line at $\alpha = 0.2$ volts indicate the default α used. Each data point was obtained from 100 trials of the neuromorphic network simulation.

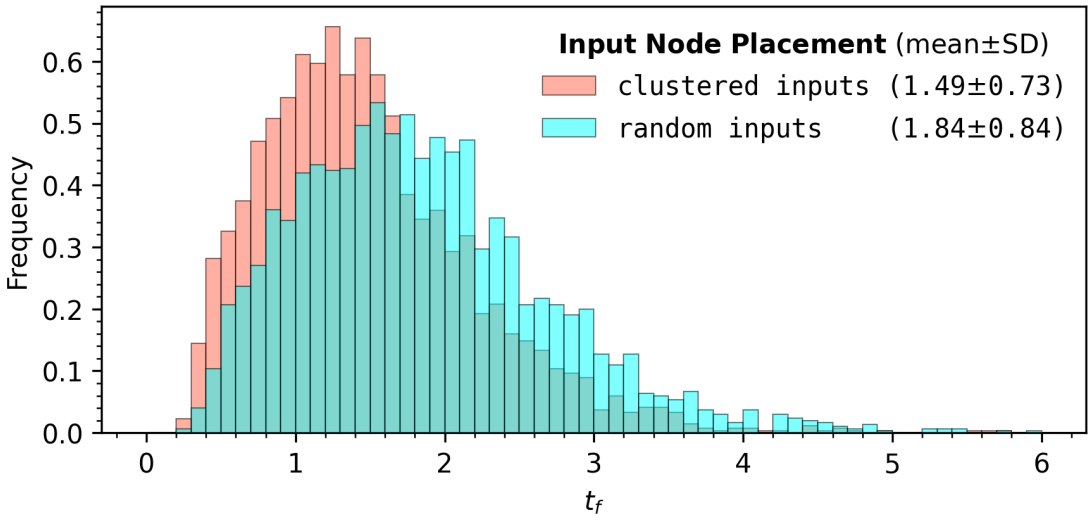

FIG. A.5. Effect of node-node distances for input node selection on forecast times. Displayed are histograms of forecast times, with clustered inputs (each is within 2 distance of each other), and randomly selected inputs. Simulation performed on a neuromorphic network with 500 nodes and 9905 edges.

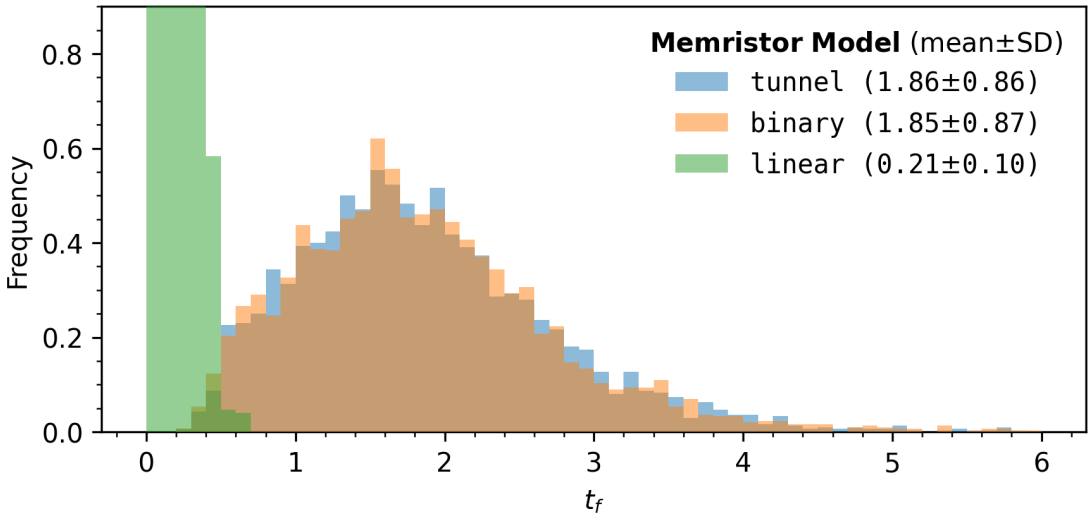

FIG. A.6. Effect of memristor model on forecast times. Displayed are histograms of forecast times, with the (default) tunnelling, binary, and linear (i.e. resistor) models. Simulation performed on a neuromorphic network with 500 nodes and 9905 edges, with $3,000$ data points per histogram.

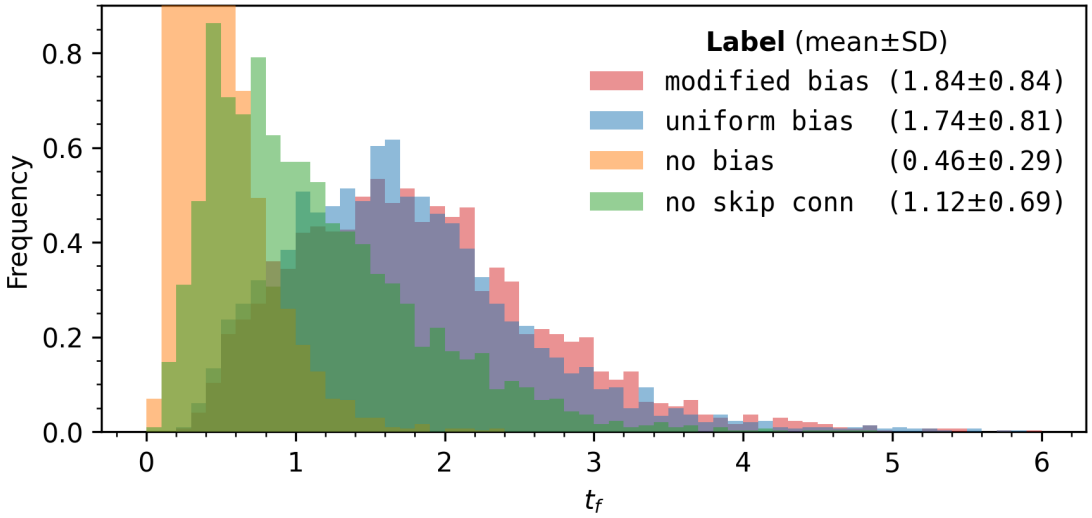

FIG. A.7. Effect of selected techniques (or lack thereof) used in this study on forecast times, displayed as histograms. "Modified bias" corresponds to the same parameters outlined in methods, with bias sampled from the interval $[-b,-b/2]\cup[b/2,b]$. "Uniform bias" has its values sampled from $[-b,b]$ (i.e. a uniform distribution $U(-b,b)$). "No bias" is equivalent to $\mathbf{b}_{\text{in}}=0$. The skip connection is the additional $\mathbf{u}(t)$ term in Eq. (2). Simulation performed on a neuromorphic network with 500 nodes and $9,905$ edges, with $3,000$ data points per histogram.

REFERENCES

[1]Zdenka Kuncic and Tomonobu Nakayama. Neuromorphic nanowire networks: Principles, progress and future prospects for neuro-inspired information processing. *Advances in Physics: X*, 6(1):1894234, 2021.

[2]Herbert Jaeger, Beatriz Noheda, and Wilfred G Van Der Wiel. Toward a formal theory for computing machines made out of whatever physics offers. *Nature communications*, 14(1):4911, 2023.

[3]Dhireesha Kudithipudi, Catherine Schuman, Craig M Vineyard, Tej Pandit, Cory Merkel, Rajkumar Kubendran, James B Aimone, Garrick Orchard, Christian Mayr, Ryad Benosman, et al. Neuromorphic computing at scale. *Nature*, 637(8047):801–812, 2025.

[4]Danijela Marković, Alice Mizrahi, Damien Querlioz, and Julie Grollier. Physics for neuromorphic computing. *Nature Reviews Physics*, 2(9):499–510, 2020.

[5]Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural networks*, 10(9):1659–1671, 1997.

[6]Jason K Eshraghian, Max Ward, Emre O Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D Lu. Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*, 111(9):1016–1054, 2023.

[7]Gianluca Milano, Giacomo Pedretti, Kevin Montano, Saverio Ricci, Shahin Hashemkhani, Luca Boarino, Daniele Ielmini, and Carlo Ricciardi. In materia reservoir computing with a fully memristive architecture based on self-organizing nanowire networks. *Nature materials*, 21(2):195–202, 2022.

[8]Adrian Diaz-Alvarez, Rintaro Higuchi, Paula Sanz-Leon, Ido Marcus, Yoshitaka Shingaya, Adam Z. Stieg, James K. Gimzewski, Zdenka Kuncic, and Tomonobu Nakayama. Emergent dynamics of neuromorphic nanowire networks. *Scientific Reports*, 9(1):14920, 2019.

[9]Alexander Vahl, Gianluca Milano, Zdenka Kuncic, Simon A Brown, and Paolo Milani. Brain-inspired computing with self-assembled networks of nano-objects. *Journal of Physics D: Applied Physics*, 2024.

[10]Leon Chua. Memristor-the missing circuit element. *IEEE Transactions on circuit theory*, 18(5):507–519, 2003.

[11]Dmitri B Strukov, Gregory S Snider, Duncan R Stewart, and R Stanley Williams. The missing memristor found. *Nature*, 453(7191):80–83, 2008.

[12]Massimiliano Di Ventra, Yuriy V Pershin, and Leon O Chua. Circuit elements with memory: memristors, memcapacitors, and meminductors. *Proceedings of the IEEE*, 97(10):1717–1724, 2009.

[13]Francesco Caravelli and Juan Pablo Carbajal. Memristors for the curious outsiders. *Technologies*, 6(4):118, 2018.

[14]Kuk-Hwan Kim, Siddharth Gaba, Dana Wheeler, Jose M Cruz-Albrecht, Tahir Hussain, Narayan Srinivasa, and Wei Lu. A functional hybrid memristor crossbar-array/cmos system for data storage and neuromorphic applications. *Nano letters*, 12(1):389–395, 2012.

[15]Can Li, Zhongrui Wang, Mingyi Rao, Daniel Belkin, Wenhao Song, Hao Jiang, Peng Yan, Yunning Li, Peng Lin, Miao Hu, et al. Long short-term memory networks in memristor crossbar arrays. *Nature Machine Intelligence*, 1(1):49–57, 2019.

[16]John Moon, Wen Ma, Jong Hoon Shin, Fuxi Cai, Chao Du, Seung Hwan Lee, and Wei D. Lu. Temporal data classification and forecasting using a memristor-based reservoir computing system. *Nature Electronics*, 2(10):480–487, 2019.

[17]Xiangpeng Liang, Jianshi Tang, Yanan Zhong, Bin Gao, He Qian, and Huaqiang Wu. Physical reservoir computing with emerging electronics. *Nature Electronics*, 7(3):193–206, 2024.

[18]Pierre Enel, Emmanuel Procyk, René Quilodran, and Peter Ford Dominey. Reservoir computing properties of neural dynamics in prefrontal cortex. *PLoS Computational Biology*, 12(6):e1004967, 2016.

[19]Joel Hochstetter, Ruomin Zhu, Alon Loeffler, Adrian Diaz-Alvarez, Tomonobu Nakayama, and Zdenka Kuncic. Avalanches and edge-of-chaos learning in neuromorphic nanowire networks. *Nature Communications*, 12(1):4008, 2021.

[20]Ruomin Zhu, Joel Hochstetter, Alon Loeffler, Adrian Diaz-Alvarez, Tomonobu Nakayama, Joseph T. Lizier, and Zdenka Kuncic. Information dynamics in neuromorphic nanowire networks. *Scientific Reports*, 11(1):13047, 2021.

[21]Valentina Baccetti, Ruomin Zhu, Zdenka Kuncic, and Francesco Caravelli. Ergodicity, lack thereof, and the performance of reservoir computing with memristive networks. *Nano Express*, 5(1):015021, 2024.

[22]Mantas Lukoševičius and Herbert Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009.

[23]Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Physical*

Review Letters, 120(2):024102, 2018.

[24]Herbert Jaeger. Adaptive nonlinear system identification with echo state networks. In *Advances in Neural Information Processing Systems*, volume 15. MIT Press, 2002.

[25]Chrisantha Fernando and Sampsa Sojakka. Pattern recognition in a bucket. In Wolfgang Banzhaf, Jens Ziegler, Thomas Christaller, Peter Dittrich, and Jan T. Kim, editors, *Advances in Artificial Life*, pages 588–597, Berlin, Heidelberg, 2003. Springer.

[26]Yinhao Xu, Georg A Gottwald, and Zdenka Kuncic. Dynamic reservoir computing with physical neuromorphic networks. In *2025 International Joint Conference on Neural Networks (IJCNN)*, 2025. accepted.

[27]Edward N. Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20(2):130–141, 1963.

[28]Felix Köster, Dhruvit Patel, Alexander Wikner, Lina Jaurigue, and Kathy Lüdge. Data-informed reservoir computing for efficient time-series prediction. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 33(7):073109, 2023.

[29]Takanori Akiyama and Gouhei Tanaka. Computational efficiency of multi-step learning echo state networks for nonlinear time series prediction. *IEEE Access*, 10:28535–28544, 2022.

[30]Sanghyeon Choi, Jaeho Shin, Gwanyeong Park, Jung Sun Eo, Jingon Jang, J Joshua Yang, and Gunuk Wang. 3D-integrated multilayered physical reservoir array for learning and forecasting time-series information. *Nature communications*, 15(1):2044, 2024.

[31]Shahrokh Shahi, Flavio H Fenton, and Elizabeth M Cherry. Prediction of chaotic time series using recurrent neural networks and reservoir computing techniques: A comparative study. *Machine learning with applications*, 8:100300, 2022.

[32]William Gilpin. Model scale versus domain knowledge in statistical forecasting of chaotic systems. *Physical Review Research*, 5(4):043252, 2023.

[33]Pinak Mandal and Georg A Gottwald. Learning dynamical systems with hit-and-run random feature maps. *arXiv:2501.06661*, 2025.

[34]Francisco Alves dos Santos, Reneé Rodrigues Lima, Jerson Leite Alves, Davi Wanderley Misturini, and Joao B Florindo. Reservoir computing and non-linear dynamics for time series analysis: An application in the financial market. *Physica D: Nonlinear Phenomena*, page 134698, 2025.

[35]Steven H Strogartz. Nonlinear dynamics and chaos: With applications to physics, biology. *Chemistry and Engineering*, 441, 1994.

[36]Steven H Strogatz. Exploring complex networks. *Nature*, 410(6825):268–276, 2001.

[37]Pantelis-Rafail Vlachas, Jaideep Pathak, Brian R Hunt, Themistoklis P Sapsis, Michelle Girvan, Edward Ott, and Petros Koumoutsakos. Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics. *Neural Networks*, 126:191–217, 2020.

[38]Jason A Platt, Stephen G Penny, Timothy A Smith, Tse-Chun Chen, and Henry DI Abarbanel. A systematic exploration of reservoir computing for forecasting complex spatiotemporal dynamics. *Neural Networks*, 153:530–552, 2022.

[39]Henry O Sillin, Renato Aguilera, Hsien-Hang Shieh, Audrius V Avizienis, Masakazu Aono, Adam Z Stieg, and James K Gimzewski. A theoretical and experimental study of neuromorphic atomic switch networks for reservoir computing. *Nanotechnology*, 24(38):384004, 2013.

[40]Zdenka Kuncic, Omid Kavehei, Ruomin Zhu, Alon Loeffler, Kaiwei Fu, Joel Hochstetter, Mike Li, James M. Shine, Adrian Diaz-Alvarez, Adam Stieg, James Gimzewski, and Tomonobu Nakayama. Neuromorphic information processing with nanowire networks. In *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5, 2020.

[41]Kaiwei Fu, Ruomin Zhu, Alon Loeffler, Joel Hochstetter, Adrian Diaz-Alvarez, Adam Stieg, James Gimzewski, Tomonobu Nakayama, and Zdenka Kuncic. Reservoir computing with neuromemristive nanowire networks. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2020.

[42]Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[43]Alon Loeffler, Ruomin Zhu, Joel Hochstetter, Mike Li, Kaiwei Fu, Adrian Diaz-Alvarez, Tomonobu Nakayama, James M Shine, and Zdenka Kuncic. Topological properties of neuromorphic nanowire networks. *Frontiers in Neuroscience*, 14:184, 2020.

[44]J. G. Simmons. Generalized formula for the electric tunnel effect between similar electrodes separated by a thin insulating film. *Journal of Applied Physics*, 34:1793–1803, 1963.

[45]Adam Z. Stieg, Audrius V. Avizienis, Henry O. Sillin, Cristina Martin-Olmos, Masakazu Aono, and James K. Gimzewski. Emergent criticality in complex turing b-type atomic switch networks. *Advanced Materials*, 24(2):286–293, 2011.

[46]Ruomin Zhu, Jason Eshraghian, and Zdenka Kuncic. Memristive reservoirs learn to learn. In *Proceedings of the 2023 International Conference on Neuromorphic Systems*, ICONS '23, pages 1–7, New York, NY, USA, 2023. Association for Computing Machinery.

[47]Ahmad Alsharef, Karan Aggarwal, Sonia, Manoj Kumar, and Ashutosh Mishra. Review of ML and AutoML solutions to forecast time-series data. *Archives of Computational Methods in Engineering*, 29(7):5297–5311, 2022.

[48]Erik Bollt. On explaining the surprising success of reservoir computing forecaster of chaos? The universal machine learning dynamical system with contrast to VAR and DMD. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(1):013108, 2021.

[49]Daniel Canaday, Aaron Griffith, and Daniel J. Gauthier. Rapid time series prediction with a hardware-based reservoir computer. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(12):123119, 2018.

[50]Changsong Gao, Di Liu, Chenhui Xu, Weidong Xie, Xianghong Zhang, Junhua Bai, Zhixian Lin, Cheng Zhang, Yuanyuan Hu, Tailiang Guo, and Huipeng Chen. Toward grouped-reservoir computing: Organic neuromorphic vertical transistor with distributed reservoir states for efficient recognition and prediction. *Nature Communications*, 15(1):740, 2024.

[51]Daniel J. Gauthier, Erik Bollt, Aaron Griffith, and Wendson A. S. Barbosa. Next generation reservoir computing. *Nature Communications*, 12(1):5564, 2021.

[52]Herbert Jaeger and Harald Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004.

[53]Herbert Jaeger. A tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach.

[54]Zdenka Kuncic, Tomonobu Nakayama, and James Gimzewski. Editorial: Focus on disordered, self-assembled neuromorphic systems. *Neuromorphic Computing and Engineering*, 2(4):040201, 2022.

[55]Decai Li, Min Han, and Jun Wang. Chaotic time series prediction based on a novel robust echo state network. *IEEE Transactions on Neural Networks and Learning Systems*, 23(5):787–799, 2012.

[56]Zhixin Lu, Brian R. Hunt, and Edward Ott. Attractor reconstruction by machine learning. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(6):061104, 2018.

[57]Liqun Luo. Architectures of neuronal circuits. *Science*, 373(6559):eabg7285, 2021.

[58]Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

[59]Jaideep Pathak, Zhixin Lu, Brian R. Hunt, Michelle Girvan, and Edward Ott. Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(12):121102, 2017.

[60]Junfei Qiao, Gongming Wang, Wenjing Li, and Xiaoli Li. A deep belief network with PLSR for nonlinear system modeling. *Neural Networks*, 104:68–79, 2018.

[61]Shahrokh Shahi, Flavio H. Fenton, and Elizabeth M. Cherry. Prediction of chaotic time series using recurrent neural networks and reservoir computing techniques: A comparative study. *Machine Learning with Applications*, 8:100300, 2022.

[62]Gouhei Tanaka, Toshiyuki Yamane, Jean Benoit Héroux, Ryosho Nakane, Naoki Kanazawa, Seiji Takeda, Hidetoshi Numata, Daiju Nakano, and Akira Hirose. Recent advances in physical reservoir computing: A review. *Neural Networks*, 115:100–123, 2019.

[63]F. Wyffels and B. Schrauwen. A comparative study of reservoir computing strategies for monthly time series prediction. *Neurocomputing*, 73(10):1958–1964, 2010.