Reservoir Computing with Neuro-memristive Nanowire Networks

Kaiwei Fu*, Ruomin Zhu*, Alon Loeffler*, Joel Hochstetter*, Adrian Diaz-Alvarez†,
Adam Stieg†‡, James Gimzewski†‡, Tomonobu Nakayama‡* and Zdenka Kuncic*‡,
*School of Physics and Sydney Nano Institute, University of Sydney, Sydney, NSW 2006, Australia
Email: zdenka.kuncic@sydney.edu.au
†International Centre for Materials Nanoarchitectonics, National Institute for Materials Science, Tsukuba, Japan
‡California NanoSystems Institute, University of California at Los Angeles, California USA

***Abstract*—We present simulations based on a model of self–assembled nanowire networks with memristive junctions and neural network–like topology. We analyze the dynamical voltage distribution in response to an applied bias and explain the network conductance fluctuations observed in our previous experimental studies. We then demonstrate the potential of neuromorphic nanowire networks as a physical reservoir by performing benchmark reservoir computing tasks. The tasks include sine wave nonlinear transformation, sine wave auto–generation and forecasting the Mackey–Glass chaotic time series.**

***Index Terms*—Neuromorphic systems, Memristive systems, Nanowire networks, Reservoir computing**

I. Introduction

Artificial neural networks are computational models using algorithms that are loosely based on biological neural networks [1]. Learning from data is the most time and power consuming part of training such models and memory bandwidth is also an important practical consideration. These challenges have motivated further development of alternate approaches based on Reservoir Computing (RC). Originating from echo state networks and liquid state machines, which were proposed as a means to reduce the training cost of recurrent neural network models [2], RC leverages nonlinear dynamical properties of a high-dimensional reservoir to process information [3]. Importantly, the reservoir itself is not trained, but rather the readout is trained using relatively straightforward methods such as linear regression or classification.

RC has been most successfully demonstrated for computational tasks in the temporal domain, such as wave generation and time series prediction, including the well-known benchmarking task Mackey–Glass chaotic time series prediction [4]–[6].

Physical reservoir computing is based on the concept that any physical dynamical system has the potential to serve as a reservoir if it meets several requirements. Tanaka *et al.* [7] list a number of such requirements, including high dimensionality, non-linearity and fading memory. A number of different physical reservoir models fulfil these requirements, including in particular ones based on analog circuits [8] and memristors [9]. Memristor-based physical RC is attractive because of the synapse-like dynamical electrical switching properties of memristive devices [9]–[13]. This has led to the development of a class of neuromorphic systems and circuits in which memristive RC has been successfully implemented, as evidenced by the ability to perform benchmark learning tasks such as hand-written digit and speech pattern recognition, as well as the Mackey-Glass forecasting task [14]–[16].

Nanowire networks [17]–[24] represent another class of neuro-memristive systems with an additional neuromorphic property: their neural network-like topology, which arises from their self–assembly, analogous to biological neural networks [25]. Inorganic nanowires comprised of silver readily self–assemble into a complex network, forming two-terminal memristive junctions where nanowires intersect. Memristive switching occurs at the junctions as a result of the formation of a conductive Ag filament above a voltage threshold [19].

Previous experimental and simulation studies based on Ag-Ag_2S-Ag nanowire networks developed by Stieg, Gimzewski and colleagues demonstrated their potential for physical RC through properties including higher harmonic generation, recurrent dynamics and waveform transformation [17], [19]–[21]. While the synthesis of those particular nanowire networks was aided by a pre-patterned substrate, the resulting network topology was sufficiently complex to observe emergent nonlinear (i.e. power-law) dynamics at the network level.

More recently, we showed that self-assembled polymer (PVP) coated Ag nanowire networks exhibit similar emergent dynamical properties as Ag-Ag_2S-Ag networks, even though their memristive junctions differ [24]. Training in hardware demonstrated these Ag-PVP-Ag nanowire networks can associatively learn spatial patterns by recalling previously established current pathways [26]. In addition to meeting the requirements of high dimensionality, non-linearity and fading memory, this suggests that physical RC may also be implemented on our Ag-PVP-Ag nanowire networks. In this study, we present simulations based on a model of our experimental self-assembled Ag-PVP-Ag nanowire networks. Simulations of these networks allow us to modify various parameters at the junction level, which is impossible to do experimentally. We first explore the switch junction and network dynamics under an applied bias. Following this, we implement several reservoir computing tasks, including nonlinear wave transformation, sine wave generation and Mackey–Glass signal prediction.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

II. Method

A. Modeling switch and network dynamics

Self-assembled nanowire networks were modeled in Matlab 2019a, as described in our previous study [23]. Nanowire–nanowire junctions were modeled as voltage–controlled memristive junctions described by a state-dependent Ohm's law, $I = G(\lambda)\,V$, where I is current, G is conductance, V is voltage and $\lambda = \lambda(t)$ is a state variable representing flux in memristor theory [17], [24]. At each junction, λ represents the conducting filament which parameterizes the conductance. All junctions are initially in a high resistance "off" state. After a voltage bias is established, an individual junction switches to a low resistance "on" state when $\lambda \geq \lambda_{\text{crit}}$, where λ_{crit} is a set threshold. The ratio of these resistance states is $R_{\text{off}}/R_{\text{on}} = 10^3$, with $R_{\text{on}} = G_0^{-1}$, where $G_0 = (13\,\text{k}\Omega)^{-1}$ is the conductance quanta.

A junction's state $\lambda(t)$ depends on its past history of voltage input. The evolution of $\lambda(t)$ is prescribed by the following model [23]:

$$\frac{d\lambda}{dt} = \begin{cases} (|V(t)| - V_{\text{set}})sgn[V(t)]\,, & |V(t)| > V_{\text{set}} \\ 0\,, & V_{\text{reset}} < |V(t)| < V_{\text{set}} \\ b(|V(t)| - V_{\text{reset}})sgn[\lambda(t)]\,, & |V(t)| < V_{\text{reset}} \end{cases} \tag{1}$$

where V_{set} is the on-threshold and V_{reset} is the off-threshold, and b is a positive constant relating the relative rates of decay and formation of the conductive filament, such that when a junction switches off, the filament state can immediately return to the state $\lambda = \lambda_{crit}/b$ to suppress fluctuations around λ_{crit}. The following default parameter values were used for all the simulation results presented here: $V_{\text{set}} = 10^{-2}\,\text{V}$, $V_{\text{reset}} = 10^{-3}\,\text{V}$ and $b = 10$. These values were found to produce simulation results that most closely matched experimental measurements [23], [24].

A self–assembled nanowire network was modelled using equation (1) and a modified nodal analysis [27] to solve Kirchoff's circuit law equations at each time point. As shown in Fig. 1, the model simulates wires scattered on a plane with uniformly random distributions of orientations and positions and with lengths sampled from a gamma distribution [23]. At the intersection of overlapping nanowires, memristive junctions that have switched on are shown as large circles, while those that are off are small squares. The wire and junction color reflects the network voltage distribution (with wires as equipotentials) and white arrows indicate the current flow across the network, from source (green star) to drain (red star). The voltage distribution and hence current flow continuously adapt to the dynamically changing memristive junctions and to the network's structural connectivity [25].

B. Reservoir computing implementation

Reservoir Computing (RC) was implemented using the nanowire network as a reservoir by allocating specific input and readout nodes and training the readout using either linear regression or classification. Three temporal information processing tasks were performed: (i) nonlinear waveform transformation; (ii) wave generation; and (iii) prediction of the Mackey–Glass chaotic time series.

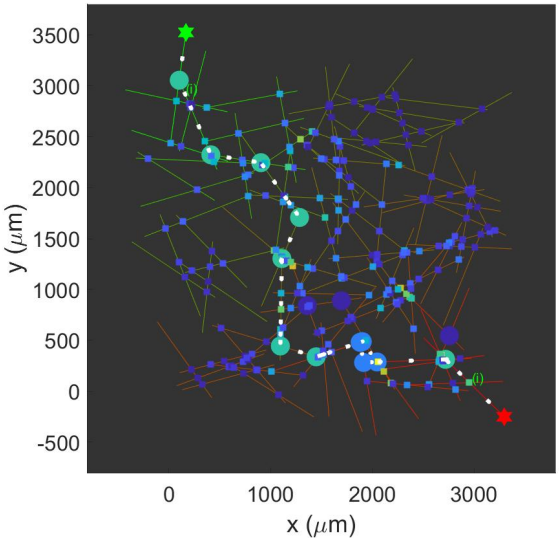


Fig. 1. Simulation snapshot of a neuromorphic nanowire network: self-assembled nanowires form a neural-like network, with each intersection of overlapping nanowires forming a memristive junction. For visual clarity, a network with only 100-nanowires and 262-junctions is shown. A voltage bias applied across the network induces memristive switching from a low conductance state (small squares) to a high conductance state (large circles). Wire and junction color reflects voltage distribution and white arrows denote current flow across the network.

For all tasks, the output signal error was calculated using the mean square error (MSE):

$$\text{MSE} = \frac{1}{M}\sum_{m=1}^{M}[T(t_m) - y(t_m)]^2 \tag{2}$$

where $T(t_m)$ is the target signal, $y(t_m)$ is the trained readout result, and M is number of time points. Performance accuracy is quantified by $1 -$ RNMSE, where RNMSE is the root–normalized MSE:

$$\text{RNMSE} = \sqrt{\frac{\sum_{m=1}^{M}[T(t_m) - y(t_m)]^2}{\sum_{m=1}^{M}[T(t_m)]^2}} \tag{3}$$

1) Nonlinear transformation: For this task, RC was implemented by setting two of N total nanowires in the network as the source and drain. The voltage of all other nanowire nodes was used to record the reservoir state at each time point. In an experimental setting, this readout scheme may be implemented using a multi–electrode array. A sine wave was used as the input voltage signal and the target signal $T(t)$ was one of either four waveforms: sawtooth, square, doubled–frequency sine, and cosine. The voltage readout was trained by linear regression to nonlinearly transform the input by solving the system of linear equations

$$X(t)\theta^T = T(t) \tag{4}$$

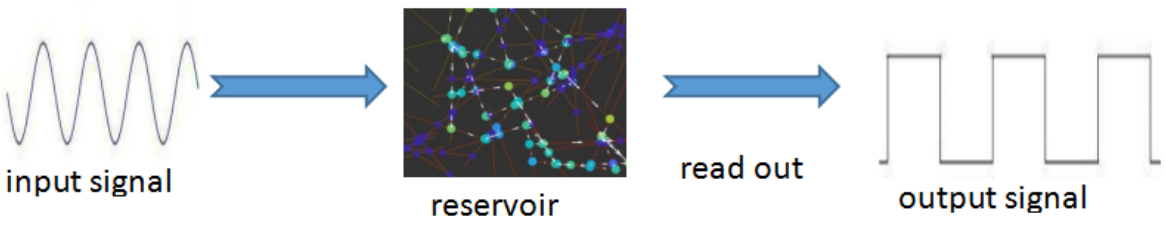


Fig. 2. Schematic depicting nonlinear transformation of an input signal (e.g. sine wave) using a nanowire network reservoir. In our implementation, one reservoir node (nanowire) receives the input signal and others serve as readout nodes which are trained by linear regression to produce the desired output signal (e.g. square wave).

where

$$X(t) = [v_1(t), v_2(t), v_3(t), \cdots, v_n(t)] \tag{5}$$

is the reservoir state, represented by a vector containing n elements (the voltage values of the N nanowires in the network reservoir), and θ is the output weight. The procedure is shown schematically in Fig. 2.

To train the readout and solve (4) for θ, the following cost function was used

$$J(\theta) = \frac{1}{M}\sum_{m=1}^{M}[T(t_m) - y(t_m)]^2 \tag{6}$$

with $y(t) = X(t)\theta^T$, and minimized using the gradient descent method:

$$\frac{\partial J(\theta)}{\partial \theta_n} = \frac{1}{M}\sum_{m=1}^{M}[T(t_m) - y(t_m)]v_n(t_m) \tag{7}$$

Where the θ_n is the weight for each readout node (N totally readouts). The training of weights was achieved in Matlab 2019a using the *regress* command.

To increase the accuracy of regression, piecewise linear regression was used [28]. Using 1000 time points in each period, the input and target signals were segmented into several equal (i) intervals that were linearly regressed independently, i.e.

$$X(t_i)\theta_i^T = T(t_i) \tag{8}$$

where $X(t_i)$ is the voltage readout of every nanowire in time interval t_i and θ_i are the corresponding output weights. Each segment is a matrix, with rows representing the nodes (n) and columns representing the time points t_i of the current segment.

2) Wave auto–generation: This task is performed differently from nonlinear waveform generation. The goal is to generate the desired wave without any external input after a training period. The signal was divided into eight equal intervals. During the first 1/8 interval, the network was primed with the input signal (same sine wave as used in the previous task) delivered to one source node and no output is read out. In this period the network gets pre-actived. Training was applied during the 2/8 – 4/8 period by delivering a teacher sine signal $u(t)$ to the source node. The voltage of every node was read out and the output weights calculated using linear regression.

The predicted next time point value of the target signal $u(t)$ is a linear combination of $u(t-1)$ and the readout signal values at t and $t-1$ according to the following auto–regression equation:

$$u(t) = \theta_{2N+2}v_0 + \theta_{2N+1}u(t-1) + \vec{\theta}\cdot[\vec{v}(t-1), \vec{v}(t-2)] \tag{9}$$

where $v_0 = 1\,\mathrm{V}$, $\vec{v}(i)$ is the N-element vector containing absolute values of voltage readout from each node at time i, and $\vec{\theta}$ is the corresponding $2N$-element weight vector, so that the output weights are given by

$$\Theta = [\theta_1, \theta_2, \theta_3, ..., \theta_{2N}, \theta_{2N+1}, \theta_{2N+2}] \tag{10}$$

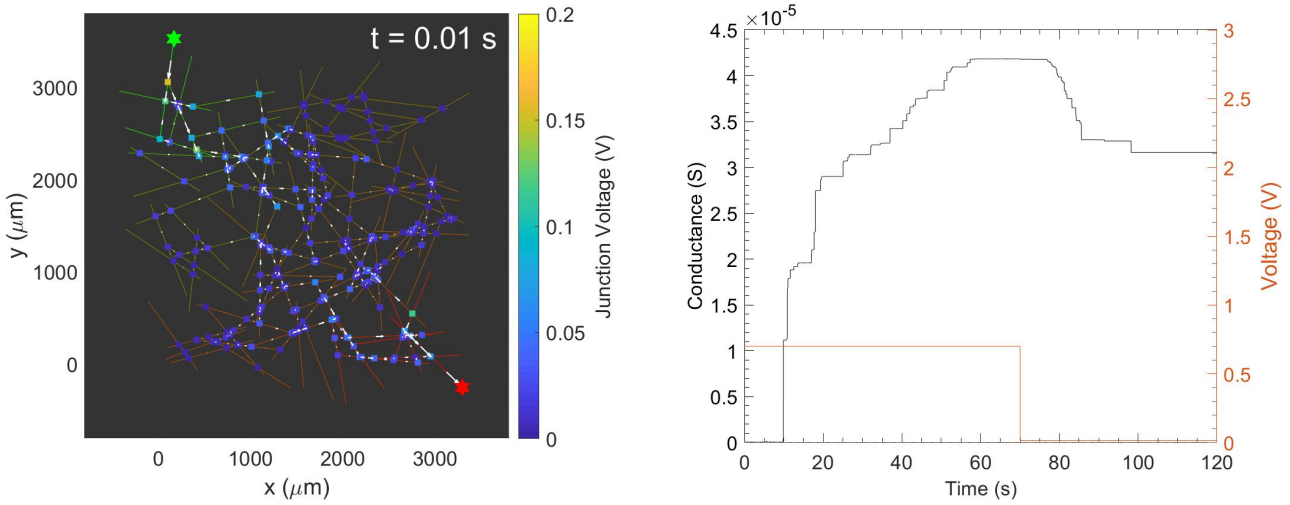
This recording of history states is also referred to as the virtual nodes method when the network is a delayed feedback system [15]. Two virtual nodes are set in the sine wave generation task. After the training period, the input was set to zero. The last 4 intervals (from 5/8 to the end) is the wave generating period, when $u(t)$ is used as input for the next time point. In this way, the network generates a wave without any external input.

3) Mackey–Glass prediction: This task was performed in a similar manner to the wave auto–generation task by explicitly using the virtual nodes method [29] for delayed feedback systems. Two external nodes, one serving as a source, the other as a drain were connected to the network reservoir. The source node receiving the input signal $u(t)$ was connected to 20 nanowire nodes on the left of the network, while the drain was connected to 20 nodes on the right. To maximize the network's effective reservoir capacity, the initial Mackey–Glass time-series signal was transformed by a proper temporal mask [30]. The voltage drop across the 20 network nodes to the drain was used as the readouts of the reservoir. For each of the 20 reservoir voltage readouts, 50 virtual nodes were used to track readout history, with the next step predicted as

$$u(t) = \vec{w}\cdot[v_0, u(t-1), \vec{S}(t-1), \vec{S}(t-2), .., \vec{S}(t-50)]\ , \tag{11}$$

where $v_0 = 1\,\mathrm{V}$, $u(i)$ is the input signal at time i, $\vec{w}$ is a 1002-element weight vector and $\vec{S}(i)$ is a 20-element vector representing the network's readout at time i.

A total of 10,000 time points were used to generate 100 s of the discretized Mackey–Glass signal. The first 50 s was



(a) (b)

Fig. 3. Nanowire network simulation geometry and activation. (a) Snapshot visualization of nanowire network at t = 0.01 s after a square pulse is input at (1) with voltage 0.7 V (network is grounded at (2)), showing the distribution of nanowires (lines) and junctions (squares). Nanowire colour denotes absolute voltage and junction colour denotes voltage drop from low (blue) to high (yellow). The green and red stars indicate the locations of source (1) and drain (2). The white arrow marks the current with direction and the value (arrow's length). (b) Voltage bias as a function of time (red line) and the resulting network conductance (blue curve) between source and drain. The minimum voltage after the square pulse is 0.015 V.

the training period using the masked Mackey–Glass signal as input. After training, for each subsequent 90 time point period, the network generated 60 time points of predicted signal using the auto–regression equation 11. For the remaining 30 time points, the original signal was input to renormalize the system's state and thereby mitigate error amplification. Different combinations of prediction and update periods were also tested.

III. Results and Discussion

A. Switch dynamics

A 100-nanowire network, with 261 junctions, was simulated. Fig. 3 (a) shows a snapshot of the network structure upon initialization, where all junctions (squares) are in a high-resistance state (R_{off} = 10 kΩ and $\lambda = 0$). Fig. 3 (b) shows the one source and one drain network's conductance response (blue) to a square input voltage pulse (red) of amplitude 0.7 V and duration 70 s, after which the voltage drops to 0.015 V. This stimulation protocol was used to analyze the effect of the network's circuit loops on local voltage redistribution and its fading memory property as the response persists with a relatively slow decay after the input signal drop.

The conductance time series reflects the voltage redistribution dynamics in the network, which depends on the local circuitry (i.e. series vs. parallel). Thus, junctions at the edge of the network where there are fewer loops and close to the source (drain), have higher (lower) voltages, while junctions near the center of the network receive relatively lower absolute voltage. Fig. 4 shows the network and junction states at two neighboring time points. The white circle highlights a memristive junction that switches from off to on, with its voltage drop decreasing accordingly by several orders of magnitude to a value lower than V_{reset}. Subsequently, this junction will turn off again and regain a high voltage, switching on again once $|V(t)| > V_{\text{reset}}$ and $\lambda > \lambda_{\text{crit}}$ (cf. eqn. (1)). Thus, voltage redistribution and switch dynamics are inextricably linked via the network's complex circuitry.

The voltage fluctuations at the single junction level lead to fluctuations in network conductance, as can be seen in Fig. 5, which shows a zoom-in of Fig. 3 in the interval 59 s - 60 s. Notice that every time the junction turns off, the filament state immediately returns to its initial state (λ drops by a factor of 10). The just-off junctions need a period of time to turn on which prolong the fluctuation and make it available to be observed. Avizienis *et al.* [17] observed similar fluctuations in current in their experimental Ag-Ag_2S-Ag nanowire network. They attributed this behavior to voltage redistribution by recurrent loops. As their network was activated by a DC bias, this produced an overall decrease in network conductivity with time [17]. Here, the network is activated with a square pulse (Fig. 3), after which a residual DC bias of 0.015 V is applied. The network conductance exhibits fading memory immediately after the pulse ends, but still maintains a high value, which suggests capacity for long–term memory. The short–term, fading memory results from the change in filament state from $\lambda_{\max}$ to λ_{crit}. The rate at which this occurs depends on the voltage distribution dynamics in the post-activation stage. The fading memory property is essential for reservoir computing applications [7].

B. Application to Reservoir Computing

1) Nonlinear transformation: Nonlinear transformation tasks were performed with our network as a reservoir using a 0.1 Hz sine wave input signal delivered to one nanowire

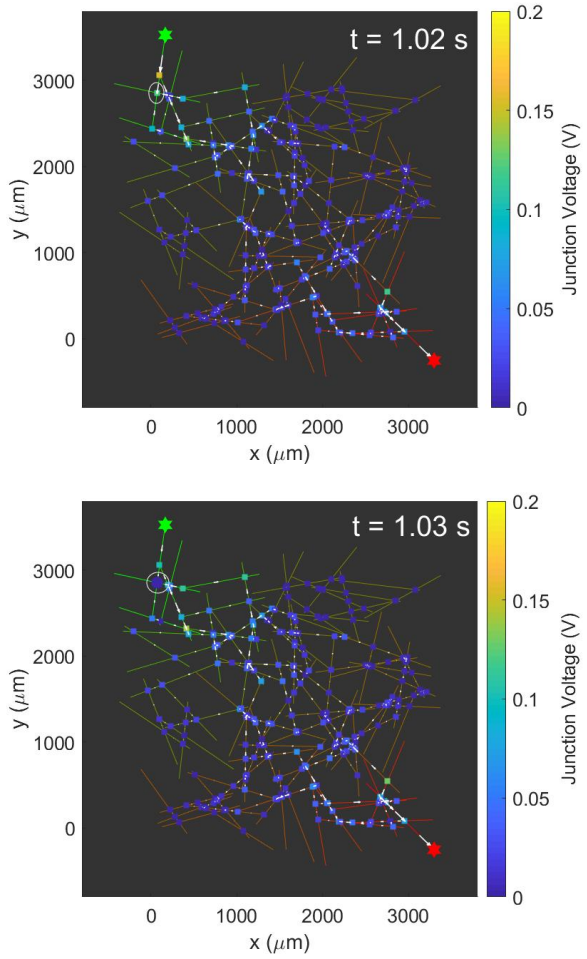


Fig. 4. Network and junction states at two neighbouring time points: t = 1.02 s (top); and t = 1.03 s (bottom). The colorbar shows voltage absolute values. The white circle marks a junction whose voltage decreases significantly once it turns on. The voltage absolute value of this junction is 0.12 V at 1.02 s, and 0.0002 V at 1.03 s when this junction switches on.

allocated as the source node, with another allocated as the drain. Simulations were performed for networks of two different sizes: 100 and 700 nanowires. Fig. 6 confirms the ability of the nanowire networks to generate higher harmonics and thus its potential to perform reservoir computing (RC).

In these simulations, the targets are: sawtooth, square, cosine and doubled–frequency sine wave. Fig. 7 shows the results for the 700-node network. Table I lists the accuracy results for both network sizes and for single vs. piecewise linear regression with 4 segments. The 700-node network, which is a larger reservoir with more degrees of freedom, performs consistently better than the 100-node network for all transformation tasks. For each network, multi–segment regression improves accuracy significantly in all cases, most significantly for the $2f$ sine and cosine transformations, which show the lowest accuracies for single–segment regression.

Using the $2f$ sine wave transformation as an example, Fig. 8 plots the accuracy achieved for this task as a function of

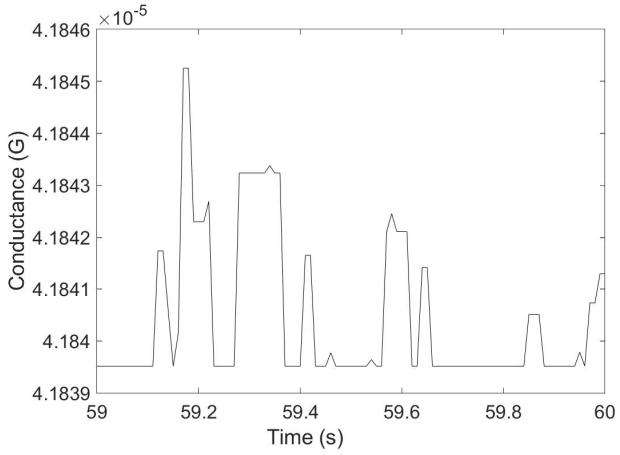


Fig. 5. Zoom–in of network conductance in Fig. 3(b) in the interval 59 s - 60 s, showing fluctuations.

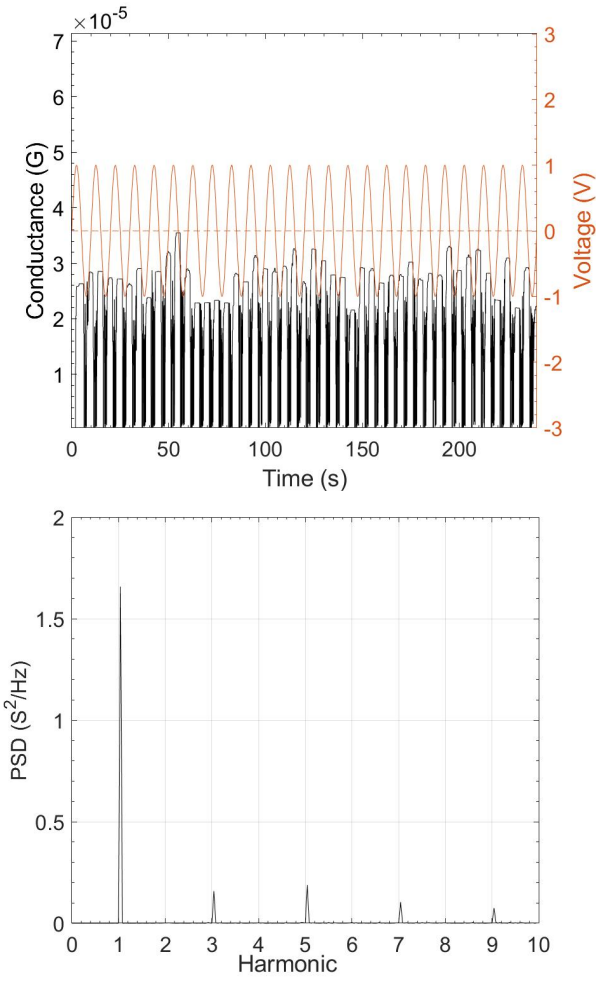


Fig. 6. Sine wave input and higher harmonics generation in a 100-node network. Top panel: Sine wave (0.1 Hz) voltage input used for nonlinear transformation tasks and corresponding network conductance; Bottom panel: Power Spectral Density (PSD) of one nanowire node, showing odd harmonics at 3,5,7,9.

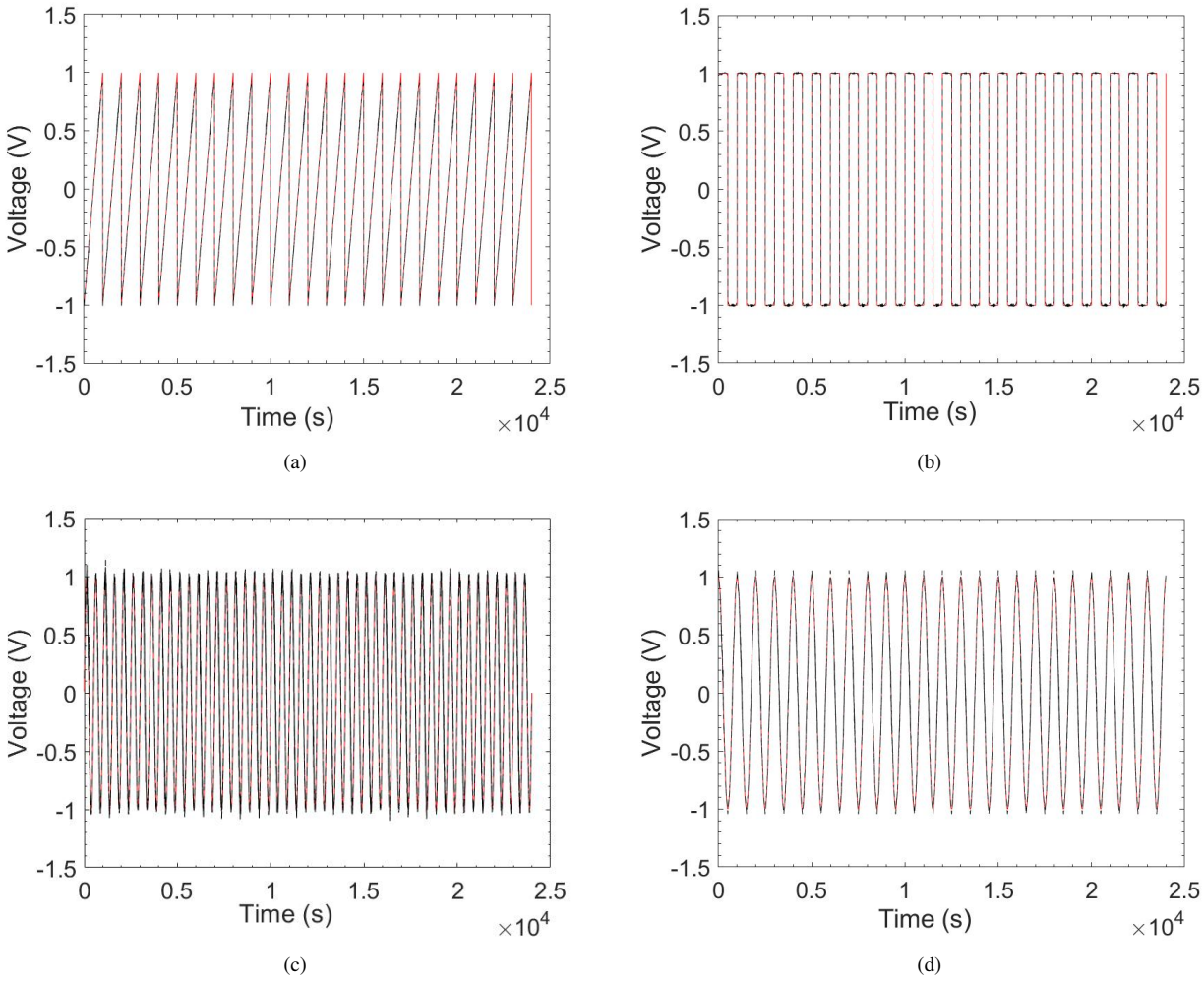


Fig. 7. Nonlinear transformation of an input sine wave by a 700-node nanowire network using piecewise linear regression (target in red, result in black): (a) sawtooth target with 4-times regression, accuracy=89.3%; (b) square wave target with 2-times regression, accuracy=92%; (c) doubled–frequency sine wave target with 4-time regression, accuracy=93.7%; (d) cosine wave target with 4-times regression, accuracy=97.2%.

TABLE I
ACCURACY OF NONLINEAR TRANSFORMATION TASKS FOR NETWORKS OF DIFFERENT SIZES (100 AND 700 NODES) AND FOR SINGLE (1) VS. MULTIPLE–SEGMENT (4) LINEAR REGRESSION.

Accuracy	100-node (1)	700-node (1)	100-node (4)	700-node (4)
Square	58.1 %	64.3 %	92.0 %	92.0 %
Sawtooth	39.2 %	44.9 %	88.5 %	89.3 %
$2f$-sine	2.4 %	31.6 %	71.3 %	93.7 %
Cosine	2.6 %	16.9 %	88.2 %	97.2 %

the regression time segmentation. For both the 100-node and 700-node networks, the accuracy increases most rapidly up to 5-times regression. Saturation is reached by the 700-node network (i.e. improvement is marginal) beyond this number of temporal regression segments. The noticeable decrease in accuracy for the 100-node network (Fig. 8 (a)) at 10-times regression may be due to how the time series of the input signal (sine wave) is segmented around the monotonically increasing/decreasing intervals. A similar effect was also observed for the 700-node network for other targets.

These results corroborate previous experimental studies by Demis *et al.* [31] using their Ag-Ag_2S-Ag physical nanowire reservoir for nonlinear transformation tasks. They applied one input electrode and read out the voltage of other measurement electrodes in contact with their network (the size of which is difficult to determine experimentally). The accuracy of their experimental measurement are $< 90\,\%$ (using standard single regression readout). Here, piecewise regression improves the accuracy remarkably and thus could be an effective way to process periodic signals in RC.

2) Wave auto-generation: In this task, the sine wave used previously as the input signal is now generated by the network without any external input after a training period. The result is shown in Fig. 9 for a 700-node network. The MSE is negligible, giving an accuracy of $\approx 100\,\%$. Notice that the length of the generated sine wave (120 s) is longer than the

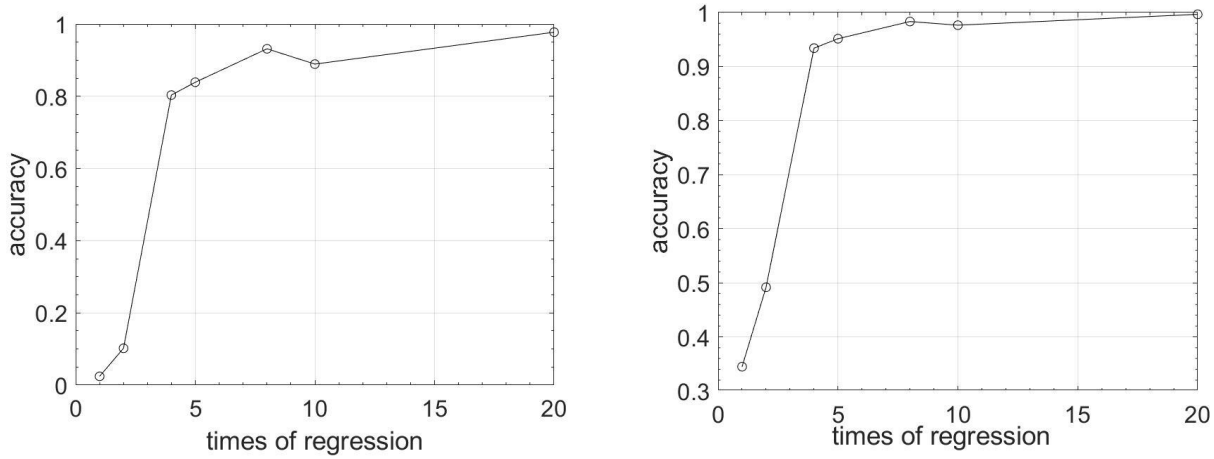


Fig. 8. Accuracy of nonlinear transformation as a function of linear regression segmentation for the doubled–frequency sine wave task using: (a) a 100-node network; and (b) a 700-node network.

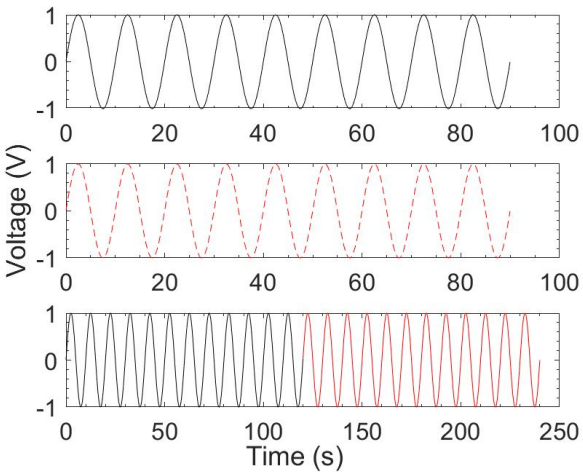


Fig. 9. Sine wave generation by a 700-node network. Top panel – target sine wave signal (teacher input). Middle panel – training by linear regression. Bottom panel – pre-activation period (30 s) and training period (90 s, black) followed by generating period (120s, red). Accuracy is $\approx 100\%$.

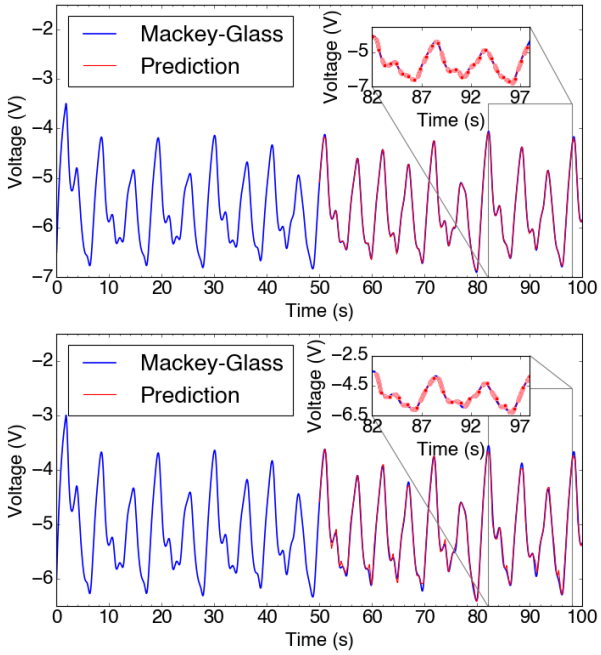


Fig. 10. Mackey–Glass time series prediction task. The masked Mackey-Glass input signal (blue) followed by the predicted signal (red) interspersed with masked signal updates. Top panel – 60-time-point prediction with 30-time-point update (accuracy 97.8%); Bottom panel – 50-time-point prediction with 40-time-point update (accuracy 98.7%). The zoom–in shows a period during generation to better visualize the prediction and update components.

training period (90 s) and indeed, the period of auto–generation can be prolonged further for hundreds of seconds.

3) Mackey–Glass next-step prediction: Fig. 10 shows results for the Mackey–Glass task for two different prediction/update relative timestep lengths: 60/30 and 50/40 (using 100 timesteps/s), with accuracies 97.8% and 98.7%, respectively. The network is first initialized by the masked input signal (in blue) before autonomous generation commences (at $t = 50$ s). The update steps are required to renormalize the predicted signal, as small deviations amplify exponentially due to the chaotic dynamics, which makes Mackey–Glass forecasting particularly challenging. Errors can be further mitigated by increasing the network reservoir size (these results are for 700-nanwoire nodes, 14,533 memristive junctions), the number of readout nodes (20 used here), and/or the number of virtual nodes used for next–step prediction (50 used here, (11)). Our simulated system is similar to that of Moon *et al.* [15], who demonstrated Mackey–Glass forecasting in an experimental system comprised of 20 memristors, each expanded with 50 virtual nodes. They achieved long–term forecasting using periodic updates of 25 timesteps after every 50 timesteps of prediction.

IV. Conclusions

Neuro–memristive switch junctions in neuromorphic nanowire networks adaptively redistribute voltage throughout the network circuitry. The resulting transient dynamics in this high–dimensional system are ideally suited to reservoir computing applications. We presented various reservoir computing implementations of nanowire networks for temporal information processing. We demonstrated the ability of these networks to nonlinearly transform an input sinusoidal signal into non–sinusoidal periodic signals, a second-harmonic signal and a phase–shifted sinusoidal signal. We found that piecewise linear regression can significantly improve accuracy in these tasks. We also demonstrated the ability of these networks to autonomously generate signals, both periodic and non-periodic, as for the case of the Mackey–Glass chaotic time series. In that case, we achieved a prediction accuracy of $\approx 98\,\%$ with the use of periodic updates to mitigate errors.

Overall, these results demonstrate that self–assembled silver nanowire networks represent a unique class of physical reservoir for neuromorphic information processing, particularly for temporal data. Work is ongoing to develop the hardware implementation of reservoir computing on these neuromorphic systems.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Computer Science Review*, vol. 3, no. 3, pp. 127–149, 2009.

[3] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural Computation*, vol. 14, no. 11, pp. 2531–2560, 2002.

[4] H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note," *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, vol. 148, no. 34, p. 13, 2001.

[5] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, no. 5667, pp. 78–80, 2004.

[6] H. Jaeger, M. Lukoševičius, D. Popovici, and U. Siewert, "Optimization and applications of echo state networks with leaky-integrator neurons," *Neural Networks*, vol. 20, no. 3, pp. 335–352, 2007.

[7] G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose, "Recent advances in physical reservoir computing: A review," *Neural Networks*, 2019.

[8] M. C. Soriano, D. Brunner, M. Escalona-Morán, C. R. Mirasso, and I. Fischer, "Minimal approach to neuro-inspired information processing," *Frontiers in Computational Neuroscience*, vol. 9, p. 68, 2015.

[9] L. Chua, "Memristor-the missing circuit element," *IEEE Transactions on circuit theory*, vol. 18, no. 5, pp. 507–519, 1971.

[10] K. Terabe, T. Hasegawa, T. Nakayama, and M. Aono, "Quantized conductance atomic switch," *Nature*, vol. 433, no. 7021, p. 47, 2005.

[11] T. Ohno, T. Hasegawa, T. Tsuruoka, K. Terabe, J. K. Gimzewski, and M. Aono, "Short-term plasticity and long-term potentiation mimicked in single inorganic synapses," *Nature Materials*, vol. 10, no. 8, p. 591, 2011.

[12] M. Aono, "Nanoionics," *Nature Materials*, vol. 6, no. 11, pp. 833–840, 2007.

[13] R. Waser, R. Dittmann, G. Staikov, and K. Szot, "Redox-based resistive switching memories–nanoionic mechanisms, prospects, and challenges," *Advanced Materials*, vol. 21, no. 25-26, pp. 2632–2663, 2009.

[14] C. Du, F. Cai, M. A. Zidan, W. Ma, S. H. Lee, and W. D. Lu, "Reservoir computing using dynamic memristors for temporal information processing," *Nature Communications*, vol. 8, no. 1, p. 2204, 2017.

[15] J. Moon, W. Ma, J. H. Shin, F. Cai, C. Du, S. H. Lee, and W. D. Lu, "Temporal data classification and forecasting using a memristor-based reservoir computing system," *Nature Electronics*, pp. 1–8, 2019.

[16] C. H. Bennett, D. Querlioz, and J.-O. Klein, "Spatio-temporal learning with arrays of analog nanosynapses," in *2017 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*. IEEE, 2017, pp. 125–130.

[17] A. V. Avizienis, H. O. Sillin, C. Martin-Olmos, H. H. Shieh, M. Aono, A. Z. Stieg, and J. K. Gimzewski, "Neuromorphic atomic switch networks," *Plos One*, vol. 7, no. 8, p. e42772, 2012.

[18] A. Z. Stieg, A. V. Avizienis, H. O. Sillin, C. Martin-Olmos, M. Aono, and J. K. Gimzewski, "Emergent criticality in complex turing b-type atomic switch networks," *Advanced Materials*, vol. 24, no. 2, pp. 286–293, 2012.

[19] H. O. Sillin, R. Aguilera, H.-H. Shieh, A. V. Avizienis, M. Aono, A. Z. Stieg, and J. K. Gimzewski, "A theoretical and experimental study of neuromorphic atomic switch networks for reservoir computing," *Nanotechnology*, vol. 24, no. 38, p. 384004, 2013.

[20] A. Z. Stieg, A. V. Avizienis, H. O. Sillin, R. Aguilera, H.-H. Shieh, C. Martin-Olmos, E. J. Sandouk, M. Aono, and J. K. Gimzewski, "Self-organization and emergence of dynamical structures in neuromorphic atomic switch networks," in *Handbook of Memristor Networks*. Springer, 2014, pp. 391–427.

[21] E. C. Demis, R. Aguilera, H. O. Sillin, K. Scharnhorst, E. J. Sandouk, M. Aono, A. Z. Stieg, and J. K. Gimzewski, "Atomic switch networks—nanoarchitectonic design of a complex system for natural computing," *Nanotechnology*, vol. 26, no. 20, p. 204003, 2015.

[22] H. G. Manning, F. Niosi, C. G. da Rocha, A. T. Bellew, C. O'Callaghan, S. Biswas, P. F. Flowers, B. J. Wiley, J. D. Holmes, M. S. Ferreira *et al.*, "Emergence of winner-takes-all connectivity paths in random nanowire networks," *Nature Communications*, vol. 9, no. 1, p. 3219, 2018.

[23] Z. Kuncic, I. Marcus, P. Sanz-Leon, R. Higuchi, Y. Shingaya, M. Li, A. Stieg, J. Gimzewski, M. Aono, and T. Nakayama, "Emergent brain-like complexity from nanowire atomic switch networks: Towards neuromorphic synthetic intelligence," in *2018 IEEE 18th International Conference on Nanotechnology (IEEE-NANO)*. IEEE, 2018, pp. 1–3.

[24] A. Diaz-Alvarez, R. Higuchi, P. Sanz-Leon, I. Marcus, Y. Shingaya, A. Z. Stieg, J. K. Gimzewski, Z. Kuncic, and T. Nakayama, "Emergent dynamics of neuromorphic nanowire networks," *Scientific Reports*, vol. 9, no. 1, pp. 1–13, 2019.

[25] A. Loeffler, R. Zhu, J. Hochstetter, M. Li, K. Fu, A. Diaz-Alvarez, T. Nakayama, J. M. Shine, and Z. Kuncic, "Topological properties of neuromorphic nanowire networks," *Frontiers in Neuroscience*, vol. 14, p. 184, 2020.

[26] A. Diaz-Alvarez, R. Higuchi, Q. Li, Y. Shingaya, and T. Nakayama, "Associative routing through neuromorphic nanowire networks," *AIP Advances*, vol. 10, no. 2, p. 025134, 2020.

[27] Chung-Wen Ho, A. Ruehli, and P. Brennan, "The modified nodal approach to network analysis," *IEEE Transactions on Circuits and Systems*, vol. 22, no. 6, pp. 504–509, June 1975.

[28] V. E. Mczgee and W. T. Carleton, "Piecewise regression," *Journal of the American Statistical Association*, vol. 65, no. 331, pp. 1109–1124, 1970.

[29] L. Appeltant, M. C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. R. Mirasso, and I. Fischer, "Information processing using a single dynamical node as complex system," *Nature Communications*, vol. 2, no. 1, pp. 1–6, 2011.

[30] L. Appeltant, G. Van der Sande, J. Danckaert, and I. Fischer, "Constructing optimized binary masks for reservoir computing with delay systems," *Scientific Reports*, vol. 4, p. 3629, 2014.

[31] E. C. Demis, R. Aguilera, K. Scharnhorst, M. Aono, A. Z. Stieg, and J. K. Gimzewski, "Nanoarchitectonic atomic switch networks for unconventional computing," *Japanese Journal of Applied Physics*, vol. 55, no. 11, p. 1102B2, 2016.