

RESEARCH ARTICLE | SEPTEMBER 04 2025

Denoising and reconstruction of nonlinear dynamics using truncated reservoir computing

Special Collection: [Nonlinear Dynamics of Reservoir Computing: Theory, Realization and Application](#)

Omid Sedehi   ; Manish Yadav  ; Merten Stender  ; Sebastian Oberst 



Chaos 35, 093103 (2025)

<https://doi.org/10.1063/5.0273505>



Articles You May Be Interested In

Optimizing reservoir computing to study nonlinear systems

Scilight (September 2025)

Comparative study of modal decomposition and dynamic equation reconstruction in data-driven modeling

AIP Advances (August 2021)

Dynamics of the stochastic Lorenz chaotic system with long memory effects

Chaos (December 2015)



Chaos

Special Topics Open for Submissions

[Learn More](#)

Denoising and reconstruction of nonlinear dynamics using truncated reservoir computing



Cite as: Chaos 35, 093103 (2025); doi: 10.1063/5.0273505

Submitted: 31 March 2025 · Accepted: 11 August 2025 ·

Published Online: 4 September 2025



View Online



Export Citation



CrossMark

Omid Sedehi,^{1,a)} Manish Yadav,^{2,b)} Merten Stender,^{2,c)} and Sebastian Oberst^{1,d)}

AFFILIATIONS

¹Centre for Audio, Acoustics and Vibration (CAAV), School of Mechanical and Mechatronic Engineering, University of Technology Sydney, Ultimo, NSW 2007, Australia

²Cyber-Physical Systems in Mechanical Engineering, Technische Universität Berlin, Straße des 17. Juni 135, 10623 Berlin, Germany

Note: This paper is part of the Special Topic on Nonlinear Dynamics of Reservoir Computing: Theory, Application and Realization.

a) Author to whom correspondence should be addressed: omid.sedehi@uts.edu.au

b) Electronic mail: manish.yadav@tu-berlin.de

c) Electronic mail: merten.stender@tu-berlin.de

d) Electronic mail: sebastian.oberst@uts.edu.au

ABSTRACT

Measurements acquired from distributed physical systems are often sparse and noisy. Therefore, signal processing and system identification tools are required to mitigate noise effects and reconstruct unobserved dynamics from limited sensor data. However, this process is particularly challenging because the fundamental equations governing the dynamics are largely unavailable in practice. Reservoir Computing (RC) techniques have shown promise in efficiently simulating dynamical systems through an unstructured and efficient computation graph comprising a set of neurons with random connectivity. However, the potential of RC to operate in noisy regimes and distinguish noise from the primary smooth or non-smooth deterministic dynamics of the system has not been fully explored. This paper presents a novel RC method for noise filtering and reconstructing unobserved nonlinear dynamics, offering a novel learning protocol associated with hyperparameter optimization. The performance of the RC in terms of noise intensity, noise frequency content, and drastic shifts in dynamical parameters is studied in two illustrative examples involving the nonlinear dynamics of the Lorenz attractor and the adaptive exponential integrate-and-fire system. It is demonstrated that denoising performance improves by truncating redundant nodes and edges of the reservoir, as well as by properly optimizing hyperparameters, such as the leakage rate, spectral radius, input connectivity, and ridge regression parameter. Furthermore, the presented framework shows good generalization behavior when tested for reconstructing unseen and qualitatively different attractors. Compared to the extended Kalman filter, the presented RC framework yields competitive accuracy at low signal-to-noise ratios and high-frequency ranges.

© 2025 Author(s). All article content, except where otherwise noted, is licensed under a Creative Commons Attribution-NonCommercial-NoDerivs 4.0 International (CC BY-NC-ND) license (<https://creativecommons.org/licenses/by-nc-nd/4.0/>). <https://doi.org/10.1063/5.0273505>

Reconstruction of unobserved dynamical responses from sparse and noisy observations is a crucial and challenging task when dealing with unknown nonlinear systems. Filtering techniques have commonly been employed to handle such problems, but they often demand a thorough understanding of the underlying processes, such as the governing equations in terms of a physics-based state-space model. This paper presents a novel truncated reservoir computing approach to simulate nonlinear

dynamics and implicitly recover the underlying physics from noisy measurements. It is proposed that reservoir computers can effectively distinguish noise from dynamics and reconstruct unmeasured dynamical responses from noisy ones without having to involve any supplementary noise reduction techniques. While the Reservoir Computing acts as a model-free mathematical modeling technique, it offers competitive accuracy compared to the conventional filtering techniques at certain noise levels.

I. INTRODUCTION

Measurements of physical quantities from distributed systems are often noisy and incomplete in covering all response quantities. This issue has inspired the development of a wide range of denoising and reconstruction methods, which fall under two major classes, namely, model-free and model-based methods, depending on whether they demand prior knowledge of underlying physics. A good example of the former is wavelet-based denoising,¹ which does not require incorporating physical knowledge explicitly, whereas stochastic filtering techniques, including Kalman-type filters,² often demand a general understanding of the physics, characterized via state-space models. However, the foundation of both methods primarily rests upon a key hypothesis, implying that the desired signal and the noise have distinguishable low- and high-dimensional features in a specific latent space.

Reservoir computing (RC) is a computational framework rooted in the broader domain of recurrent neural networks (RNNs) and dynamical systems theory. It leverages the dynamics of a fixed, randomly initialized high-dimensional system, known as the “reservoir,” to transform sequential input data into a latent space representation. In this paradigm, only the output layer is trained through ridge regression,³ which linearly combines the reservoir states to minimize prediction errors. This significantly reduces the computational complexity associated with the training, mainly because only a system of linear equations needs to be solved coupled with a proper regularization. RC has been particularly effective in time-series processing,^{4–6} speech recognition,^{3,7} and chaotic system modeling^{8–10} due to its ability to exploit the reservoir’s rich temporal dynamics and memory capacity. Heuristics of building the reservoir formulation, e.g., including an output feedback or a delayed input, have been shown to improve the results.¹¹ The generalization behavior of RC has recently been studied, where an RC approach is employed for making predictions on unexplored attractors.^{12,13} However, the optimal size and structure of the reservoir remain an open question since reservoir networks are mostly initialized considering random connectivity. Only lately researchers have started investigating the structure–function relationships to discover network properties that are beneficial for a given learning task within the RC framework.^{14,15}

Recent research has explored the impact of noise on the RC’s performance, revealing its strength and weaknesses for signal reconstruction across various contexts. Rarely RCs have been employed specifically for simultaneous denoising and signal reconstruction tasks. Semenova *et al.*¹⁶ have studied the propagation of noise from the input nodes toward the reservoir outputs in recurrent and multilayer networks. Estébanez *et al.*¹⁷ have shown that, while noise can degrade the performance of RC-based analog computations, its introduction in the training phase can lead to improved robustness and accuracy. Similarly, Wikner *et al.*¹⁸ have shown that adding noise to the training data can improve the stability of predictions. Röhm *et al.*¹⁹ have shown that an RC trained based on a single noisy trajectory may still be able to reproduce unexplored attractors. Nathe *et al.*²⁰ have investigated the impact of measurement noise on the performance of RC when used for reconstructing chaotic systems. Lee *et al.*²¹ have employed RC along with a nonlinear forward operator for efficient image denoising. Choi and Kim²² have separated the signal from noise as the predictable part modeled by RC and acquired information about the additive or

multiplicative nature of the noise, as well as noise statistical properties. Despite these advances in RC-based denoising approaches, a thorough investigation is still required to explore the potential of RC for discarding different noise processes and factors affecting denoising performance.

This work introduces a novel learning protocol for denoising and reconstruction of nonlinear dynamics based on a truncated RC framework. This approach begins with training the reservoir on a specific dataset using the conventional ridge regression, followed by a new optimization strategy that calibrates key hyperparameters, including the ridge regression coefficient, leakage rate, and spectral radius, aiming to enhance performance. Once the optimized reservoir is obtained, we truncate redundant nodes and edges according to the model accuracy on the validation dataset using a novel pruning strategy. Two numerical examples are provided to demonstrate the proposed protocol and study the impact of noise intensity and frequency content on the performance of the RC. The generalization behavior of the proposed networks is demonstrated against drastic changes in the dynamical parameters and Signal-to-Noise Ratio (SNR). Comparison with the Extended Kalman Filter (EKF), a near optimal filter,² is provided for the Lorenz attractor example, considering different noise levels. Finally, the method is further demonstrated using the non-smooth deterministic dynamics of the adaptive exponential integrate-and-fire (AdEx) model.

This study primarily investigates the potential of reservoir computing for denoising and state reconstruction. In addition, it applies hyperparameter optimization and explores the pruning of redundant nodes and edges. The pruning of small reservoir networks, specifically the removal of redundant nodes and connections, has received relatively little attention in previous studies. Notably, Dutoit *et al.*²³ have shown that the pruning of the readout nodes is an effective strategy to improve the prediction accuracy and reduce hardware and software computational demands. Scardapane *et al.*²⁴ have characterized the significance of reservoir synapses according to the correlation between the input and output of the neurons and specified the significance of neurons based on the importance of their neighboring synapses. Then, based on this criterion, they have presented a pruning approach for reducing the reservoir dimensionality without calculating the error gradients.²⁴ Haluszczynski *et al.*²⁵ have shown that removing the nodes with negative impact on the reservoir’s response to the input can enhance the accuracy and reduce instabilities. Compared to these works, this study provides a more systematic and generic approach to reducing the reservoir nodes and edges.

The remainder of this paper is organized as follows: First, the mathematical foundation of the RC framework will be reviewed, presenting the conventional ridge regression method. Then, the reservoir’s hyperparameters will be identified, for which an optimization approach will be presented. Truncation of the RC through a pruning approach will be discussed, and further comparisons will be made between the proposed RC framework and conventional filtering techniques from a theoretical standpoint. The presented approaches will be demonstrated using two nonlinear systems, aiming to address simultaneous state reconstruction and denoising tasks. Finally, further discussions and conclusions will be drawn about the denoising performance of the proposed truncated RC framework.

II. RESERVOIR COMPUTING FRAMEWORK

A. Echo State Network

Several neural network architectures are available for constructing reservoir computers, among which Echo State Network (ESN)²⁶ and Liquid State Machines^{27,28} are perhaps the most widely used ones. In this paper, ESN is employed for creating a reservoir whose formulation is described in terms of the latent dynamical states,

$$\mathbf{r}(t_{i+1}) = (1 - \alpha)\mathbf{r}(t_i) + \alpha f(\mathbf{W}_{\text{res}}\mathbf{r}(t_i) + \mathbf{W}_{\text{in}}\mathbf{u}(t_i) + \mathbf{b}), \quad (1)$$

where $f(\cdot)$ is the activation function, considered $\tanh(\cdot)$, $\mathbf{r}(t_i)$ is the reservoir state at time t_i , α is the leakage rate representing how much of the preceding state contributes to the present state, \mathbf{W}_{res} is the reservoir weight matrix, considered a random weighted Erdős–Rényi (ER) graph matrix with p internal connectivity probability, \mathbf{W}_{in} is the input weight matrix scaled by a factor of ζ , $\mathbf{u}(t_i)$ is the input vector at time t_i , and \mathbf{b} is the bias term. Notably, an important parameter of the reservoir matrix (\mathbf{W}_{res}) is its spectral radius, denoted by γ and calculated as the greatest eigenvalue of the reservoir matrix.²⁹

Given the above equation, a recursive formulation is available for calculating the reservoir states. By collecting all states into a matrix like $\mathbf{R}_n = [\mathbf{r}(t_1), \dots, \mathbf{r}(t_n)]^T$, simulated responses of the physical system at different time steps comprised in $\mathbf{Y}_n = [\mathbf{y}(t_1), \dots, \mathbf{y}(t_n)]^T$ can be described as

$$\mathbf{Y}_n = \mathbf{R}_n \mathbf{W}_{\text{out}}. \quad (2)$$

In the training phase of the RC framework, it is customarily assumed that the reservoir is fixed, whereas the readout matrix \mathbf{W}_{out} must be trained. Thus, the optimization will simplify into a linear regression, which requires minimizing the following loss function:

$$L = \|\hat{\mathbf{Y}}_n - \mathbf{R}_n \mathbf{W}_{\text{out}}\|_2^2 + \lambda \|\mathbf{W}_{\text{out}}\|_2^2, \quad (3)$$

where $\hat{\mathbf{Y}}_n$ is a vector of actual measurements fed into the RC as input. This formulation, referred to as linear ridge regression,³⁰ can readily calibrate the readout nodes through an explicit solution, i.e.,

$$\hat{\mathbf{W}}_{\text{out}} = (\mathbf{R}_n^T \mathbf{R}_n + \lambda \mathbf{I})^{-1} \mathbf{R}_n^T \hat{\mathbf{Y}}_n, \quad (4)$$

where $\hat{\mathbf{W}}_{\text{out}}$ is the optimal readout matrix and λ is the ridge coefficient, often selected to be a small coefficient like 10^{-8} . In Eq. (3), it is evident that the first term, $\|\hat{\mathbf{Y}}_n - \mathbf{R}_n \mathbf{W}_{\text{out}}\|_2^2$, provides a measure of fitting accuracy, whereas the second term, $\lambda \|\mathbf{W}_{\text{out}}\|_2^2$, involves a penalty over the non-zero elements of \mathbf{W}_{out} , implementing a smooth sparsity. Note that in problems involving different levels of noise, λ must be inferred properly based on the data.

B. Proposed learning protocol

The reservoir is assumed to take noisy and incomplete measured data $\mathbf{U}_n = [\mathbf{u}(t_1), \dots, \mathbf{u}(t_n)]^T$ as the input and return filtered and complete states \mathbf{Y}_n as the output. Figure 1 provides an overview of this specific implementation of the reservoir. At first glance, it might be difficult to train such an RC since both noisy and clean data are required in the training phase. However, such data can be obtained via computer simulations wherein the dynamics are

considered entirely known for specific chaotic orbits. Another possibility is to employ simpler system dynamics for acquiring noisy data and attempting to clean those measurements through conventional filtering techniques.¹ In both cases, the RC trained based on these datasets is anticipated to generalize across a large neighborhood such that it can be employed for the denoising and reconstruction of unexplored nonlinear dynamics while being trained on limited clean data.

C. Hyperparameter optimization

The RC framework presented in Sec. II A is straightforward to follow and implement because it only requires performing a linear regression for the training of the readout unit. However, the performance of the reservoir computer often benefits from the fine-tuning of the reservoir and optimizing its underlying parameters. Such parameters include the number of neurons, leakage rate, spectral radius, input scaling, and node connectivity, whose optimization can be performed through a greedy algorithm that minimizes the loss function of Eq. (3) for the underlying parameters while considering a held-out dataset. To formalize such an optimization, let the hyperparameters of the RC be denoted by ϕ , specifically comprising the number of reservoir nodes (N), the leakage rate (α), the spectral radius (γ), the input scaling ratio (ζ), and the reservoir internal connectivity rate (p). The optimal values of these hyperparameters should be determined from

$$\hat{\phi} = \arg \min_{\phi} (L), \quad (5)$$

where the loss function is given by Eq. (3). It should be noted that this minimization might encounter computational difficulties if gradient-descent-based optimization approaches³¹ are employed. In effect, calculating the gradients requires implementing backpropagation³² through the hidden states of the reservoir, which is often deemed challenging if not impossible, especially when the number of time steps in the training data is large.³³ Therefore, grid-based³⁴ or random search algorithms³⁵ are often preferred in the literature. In this study, surrogate-assisted optimization techniques are employed along with cross-validation algorithms, which avoid computing the gradients³⁶ using the Python-based package *scikit-optimize*.³⁷

D. Tuning the ridge coefficient

The ridge regression coefficient (λ) becomes important when dealing with noisy data. This coefficient is inversely proportional to the level of noise contained in the data, and in the machine learning literature, it is often calibrated through cross-validation techniques.³⁸ It should be noted that this coefficient can also be treated as a hyperparameter and calibrated via Eq. (5). However, due to the unknown nature of noise level and the wide range of potential values for this parameter, the inclusion of this parameter in the optimization algorithm might not be feasible. Therefore, an empirical optimization algorithm is preferred, wherein the optimal value of this parameter is selected from a relatively coarse one-dimensional grid. For this purpose, a set of exponents of ten is introduced to the cross-validation algorithm, i.e., $\{10^{-15}, 10^{-14}, \dots, 10^{20}\}$, and then, the best choice is specified through computing the cross-validation

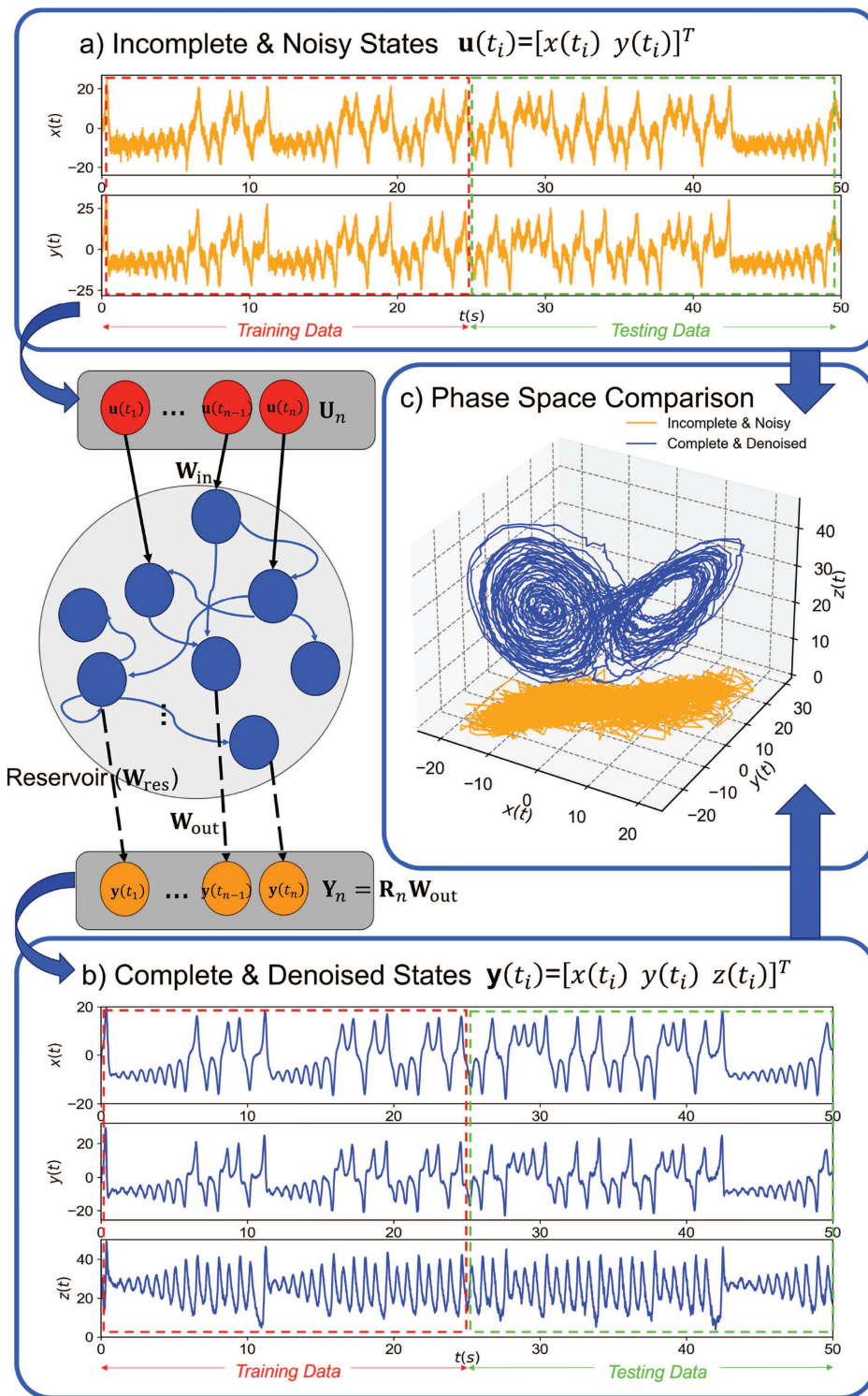


FIG. 1. Overview of the proposed RC protocol for denoising and reconstruction of nonlinear dynamics. Blue, red, and orange circles show reservoir nodes, input nodes, and readout nodes, respectively. The blue arrows show the edges of the reservoir chosen randomly. The solid black arrows show the input connection to specific nodes of the reservoir, and the black dashed arrows show the readout connections. (a) Incomplete and noisy dynamics of the Lorenz system considered as the input to the RC. (b) Reproduced Lorenz dynamics obtained through the RC framework. (c) Comparison of the input and output of the reservoir in the phase space, representing how three-dimensional dynamics (blue curve) are reconstructed from two-dimensional noisy signals (orange curve).

score. Although this approach might not be precise in terms of simultaneously optimizing all hyperparameters, it offers significant computational merits as it retains the closed-form solution of Eq. (3).

E. Pruning and evolution mechanisms

The above optimization approach can be used to specify an optimal number of nodes for the RC while edges, representing the internal connectivity in the reservoir, are selected at random. Due to this arbitrary initialization, such an RC might not be in its minimal form, and there could be redundant nodes and edges, degrading the prediction performance. In this sense, it is reasonable to explore ways to reduce redundancies through truncating unnecessary nodes and edges.^{14,15} Attaining minimality in the size and structure of the reservoir is also important for hardware development when a digital circuit is going to be designed for implementation via a physical RC.³⁹ Therefore, to remove ineffective nodes and edges, one can first train the reservoir and its hyperparameters, considering a specific number of nodes and random edges, and then perform an exhaustive search for discarding a predefined percentage of nodes and connections, subject to satisfying a performance metric. The second stage can be implemented considering another optimization scheme, where the mean-squared error (MSE) calculated for the validation dataset should be minimized while the pruning of nodes and edges takes place. However, this pruning strategy requires establishing a ranking mechanism across the nodes and edges so that the elimination can start from the least influential nodes. Five measures are considered for weighing the nodes' importance:

- Absolute mean state: The absolute average of the latent states is a metric to assess a node's activity over the entire training period. Higher average values indicate greater node activity.
- State variance: A higher variance in nodal states may indicate higher node activity.
- Number of edges: Nodes with more connections may have a greater influence on the RC's performance.
- Clustering: Nodes with a larger clustering coefficient can be more important as they better connect neighboring nodes.
- Page-rank: This web-inspired scoring approach integrates the influence of inbound and outbound edges, as well as the quality of edges as measured in terms of their relative weight, frequency of interaction, and contextual relevance within the network.⁴⁰

The above measures can be normalized to show a score in each category between zero and one. Then, the summation of these factors can be used for selecting the nodes nominated for elimination. Based on a preselected truncation percentage, redundant nodes will be eliminated, and the RC hyperparameters will be recalibrated. This process continues until a target performance is either achieved or the maximum number of trials is attained. Here, we hypothesize the usefulness of such metrics based on our intuition, but choosing other metrics for pruning nodes and edges is also possible. However, the decision whether to remove or retain the nominated nodes or edges must be made based on the performance.

In contrast to the pruning approach, we might encounter cases in which the number of nodes appears insufficient to describe the dynamics properly, and the performance can still be improved by

adding extra nodes and connections. Although increasing the number of nodes can be an option always available at the expense of extra computational costs, the performance increase may be limited since nodes and edges are initially selected at random in the classical RC framework. Additionally, the presented pruning approach is essentially a backward truncation approach, which might not cause a significant change in the performance indices. In such cases, it is reasonable to start with a simple RC with minimum dimensionality and consider an evolutionary approach for the growth of the RC toward a more complex network architecture.¹⁵ This evolution can be implemented by following the same greedy approach as above, but with one difference, that is, the addition of nodes and edges shall undergo a similar post-processing optimization.

III. DISCRETE-TIME EXTENDED KALMAN FILTER

To provide a competitive baseline, we consider a qualitatively different filtering technique based on explicit modeling of the system dynamics. A method widely applied to nonlinear systems is EKF, a well-known tool for the denoising and reconstruction of dynamic responses from incomplete noisy measurements.² EKF operates on state-space models of distributed systems and requires a clear mathematical model of the system to discard noise. In [Appendix A](#), a mathematical exposition of the EKF is provided for reference.

It should be noted that the most significant demerit of these methods lies in the fact that they require a thorough understanding of the dynamics in terms of a multi-dimensional state-space model. In contrast, the RC framework presented earlier is a model-free approach, which does not require any explicit dynamical model and provides a general mathematical framework to implicitly learn the dynamics directly from the data. In [Sec. IV](#), we benchmark the results of the proposed RC framework against the EKF, showing how these methods compare in terms of validation accuracy.

IV. RESULTS

A. Lorenz chaotic dynamics

The Lorenz system is attractive to study in this work as it qualitatively exhibits different dynamics along the period-doubling route to chaos (see [Appendix B](#) for its governing equations and details). For generating training data, $\sigma = 10$, $\rho = 28$, and $\beta = 8/3$ are considered. By employing a time interval of $\Delta t = 0.005$ s and a recording duration of $T = 50$ s, discrete-time dynamical datasets were generated considering initial conditions $x_0 = y_0 = z_0 = 1$ and polluted with specific levels of additive Gaussian White Noise (GWN). Across these datasets, the intensity of noise is considered variable as the noise standard deviation ranges from 0.25% (52 dB) to 100% (0 dB) of the root-mean-square (RMS) of the original signal's components. The noise-contaminated recordings of $x(t)$ and $y(t)$ are considered observables of a hypothetical measurement system and used as the input to the reservoir computer, whereas the clean version of all states is regarded as its output during training ([Fig. 1](#)). Hence, the RC must learn both denoising the inputs and reconstructing a third state. Given this specification, the generated data of $t \in [0, 25]$ s act as the training data, and that corresponding to $t \in [25, 50]$ s serve as the validation set. The reservoir hyperparameters are optimized through searching within specific intervals,

including the leakage rate from $\alpha \in (0.01, 1)$, the spectral radius from $\gamma \in (0.01, 1)$, the input scaling from $\zeta \in (0.1, 2)$, and the internal connectivity of reservoir nodes from $p \in (0.1, 0.9)$.

Figures 1(a) and 1(b) represent the inputs and outputs of the RC for reconstructing Lorenz chaotic dynamics. The results presented correspond to a reservoir computer whose hyperparameters and readout nodes are first trained, with its internal network being then truncated through the method explained above. The output of the reservoir is considerably smoother than the original noisy signals (Fig. 1). The comparison of the noisy and reproduced dynamics in phase space [Fig. 1(c)] shows the capability of RC in separating the dynamics of noise and responses.

1. Optimization of nodes and edges

This section examines how the hyperparameter optimization and node removal approach can enhance the reservoir performance, particularly compared to a classical RC, in which the edges are random and only readout nodes are subject to calibration. The MSE is considered a measure of accuracy when the same validation set is used to compare different training schemes. However, due to the difference in the order of magnitude of the responses, a normalization over the mean-squared of the signal is included, yielding a dimensionless measure as follows:

$$\text{NMSE} = \frac{\|\hat{\mathbf{Y}}_n - \mathbf{R}_n \hat{\mathbf{W}}_{\text{out}}\|_2^2}{\|\hat{\mathbf{Y}}_n\|_2^2}. \quad (6)$$

Given the above measure, Fig. 2 shows improvement in the logarithm of NMSE for three training strategies considering different SNRs in the validation dataset. Note that SNR is defined here as the RMS of the signal over the RMS of noise. The vanilla reservoir, whose readout nodes are only trained, provides the largest NMSE, whereas further optimization of the reservoir can improve the accuracy up to several orders of magnitude in some cases. More specifically, the hyperparameter tuning presented above can considerably reduce the prediction errors, and truncating nodes and edges can further improve the accuracy. However, this superior prediction performance comes at the expense of extra computational costs.

The performance of these methods is also compared with that of the EKF. As can be seen in Fig. 2, the presented truncated RC can outperform EKF at SNRs of 1 and 4 although no explicit knowledge of the dynamics is introduced to the RC, which is contrary to the EKF that relies on the accuracy of the state-space model. Based on this result, it can be concluded that, specifically at low SNRs, the RC can offer competitive accuracy when compared with the EKF. However, it should be noted that this comparison is made while the EKF is in an advantageous position. In effect, the EKF uses physics-driven state-space models to discard noise, whereas the RC explicitly separates the desired dynamics from noise through the proposed learning protocol.

2. Characterization of denoising gain

In this section, we investigate when the reservoir is trained on a dataset with a specific SNR and how its accuracy generalizes when tested on validation datasets with smaller or larger SNRs. For this purpose, RCs and their hyperparameters are trained using datasets

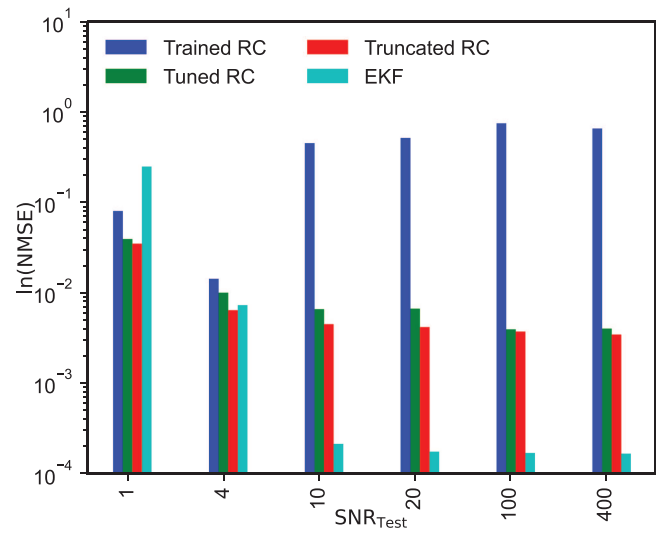


FIG. 2. Performance evaluation of different training methods considering varying SNRs. Trained RC: Only the readout nodes of the RC are updated through ridge regression using 500 reservoir nodes with random edges. Tuned RC: The reservoir hyperparameters, e.g., leakage rate, spectral radius, and node connectivity ratios, were subjected to optimization while training the readout nodes as well (the number of reservoir nodes was fixed at 500). Truncated RC: The tuned RC is further optimized by reducing nodes and edges through a separate optimization.

contaminated with varying levels of noise. Then, the calibrated RCs are used for denoising and reconstruction of held-out datasets with different levels of noise. To compare the denoising capability in these cases, a gain ratio is introduced as follows:

$$\text{Denoising Gain} = \frac{\text{SNR}_{\text{Reconstructed}}}{\text{SNR}_{\text{Test}}}, \quad (7)$$

where $\text{SNR}_{\text{Reconstructed}}$ represents the amount of noise present in the predictions made by the RC, and SNR_{Test} represents the amount of noise in the test dataset. When this index is greater than 1.0, it can be concluded that the RC can discard the noise, whereas values equal to or smaller than 1.0 represent cases where the RC adversely increases the amount of noise in the reconstructed data. The color-map matrix in Fig. 3 visualizes the denoising gain, where the green and red colors show positive and negative denoising performance, respectively. Values on the minor diagonal of the matrix show cases where the training and testing datasets have had the same level of noise intensity. The best denoising gain is achieved when the training and testing datasets correspond to the same level of noise. Additionally, the denoising gain is better for large SNRs. For $\text{SNR}_{\text{Train}} = 100$ and 400 that indicates the level noise in the training set, the RC is rather unsuccessful in denoising the signals, although it still has merits in terms of reconstructing the unobserved dynamics. Additionally, the color map is completely non-symmetrical, which implies that, for a specific level of noise contained in the test dataset, those RCs trained on smaller noise perform relatively better.

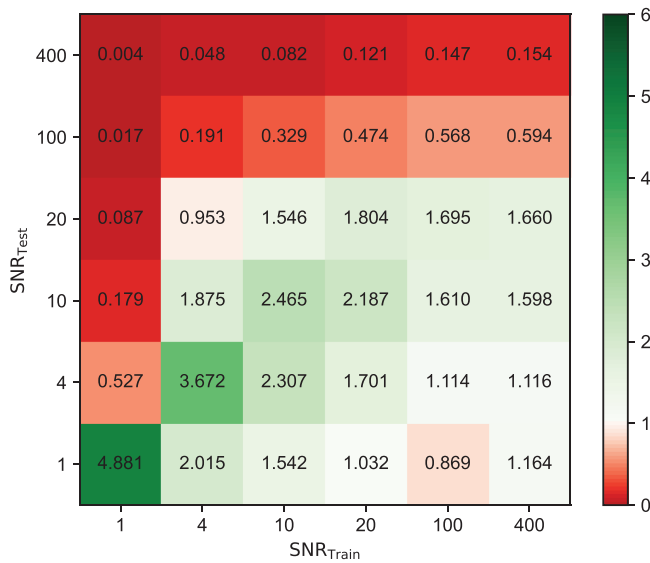


FIG. 3. Denoising gain matrix obtained for training and testing datasets with different levels of noise. Note that 10 separate reservoirs are trained and tested, and the average gain values are presented to reduce the bias due to random initialization of the RC.

3. Generalization of denoising performance

As demonstrated, the accuracy of the RC depends on the intensity of noise contained in the data. Similarly, it can be concluded that the accuracy might also be related to the Lorenz qualitative dynamics. To analyze such effects, the denoising gain is presented in Fig. 4 for validation datasets generated with the Prandtl parameter varying within $\sigma \in [0 - 20]$, specifically when the RC and its hyperparameters were trained on SNR = 4 and $\sigma = 10$. As can be seen, the best gain is achieved in the vicinity of the selected Prandtl parameter ($\sigma = 10$) used for training, considering the same noise level as the training data. Within a relatively wide neighborhood of $\sigma = 10$, the gain is still significantly larger than one, although the denoising gain decreases as we deviate from the center. However, there is a pattern change for values of $\sigma < 4.0$, where the denoising gain increases. This observation is attributed to the gradual change in the nonlinear dynamics in the vicinity of $\sigma = 4.0$, as demonstrated in the bifurcation diagram of Fig. 4(c). This figure highlights that the denoising gain also depends on the nonlinear dynamics considered for training the RC, and it will not necessarily show monotonous behavior as we deviate from the neighborhood of the parameter used for training the RC.

It is also worth investigating how the denoising gain varies across the domain of the Prandtl parameter, considering two additional datasets for the training of the RC and its hyperparameters. One dataset is generated using $\sigma = 8$ with SNR = 1, and the other is generated using the same parameters as before, i.e., $\sigma = 10$ but using SNR = 1. Note that all other factors are considered the same as the original dataset. Figure 5 presents the denoising gain for the validation datasets when the above settings are used for generating the training data. It is clear that the denoising gain considerably

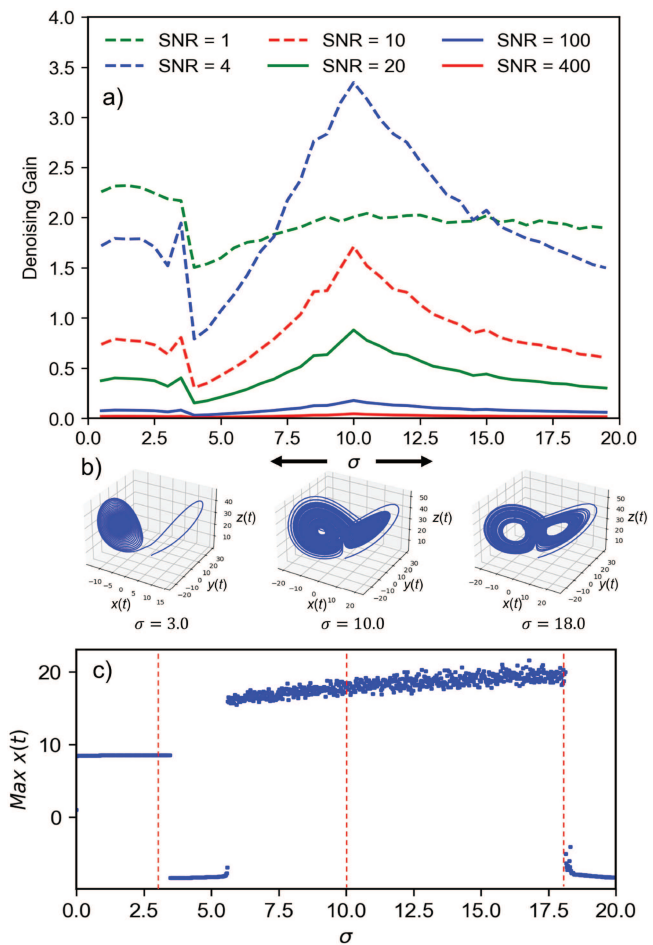


FIG. 4. Generalization of denoising gain in terms of variation in Prandtl parameter. (a) Denoising gain is presented when different levels of SNR and Prandtl parameters are examined for a given RC trained on a dataset with $\sigma = 10$ and SNR = 4. (b) Phase space representation of a Lorenz attractor for specific choices of σ . (c) Bifurcation diagram showing variation in the maximum $x(t)$. Results in sub-figure (a) are averaged over 10 randomly configured RCs to reduce the impact of randomly initialized RCs.

improves around $\sigma = 8$ and particularly for SNR = 1, exhibiting a better generalization performance across the parameter domain as well. As can be seen, by including extra training data, it is possible to enhance the denoising gain and obtain more accurate computing reservoirs. This result promotes the potential of using a reservoir network trained on a few simple chaotic orbits when applied for reconstructing unseen nonlinear dynamics and highlights how the denoising gain improves as additional data are included.

B. AdEx model

This section examines the performance of the proposed RC framework when having stiff and non-smooth dynamics. For this

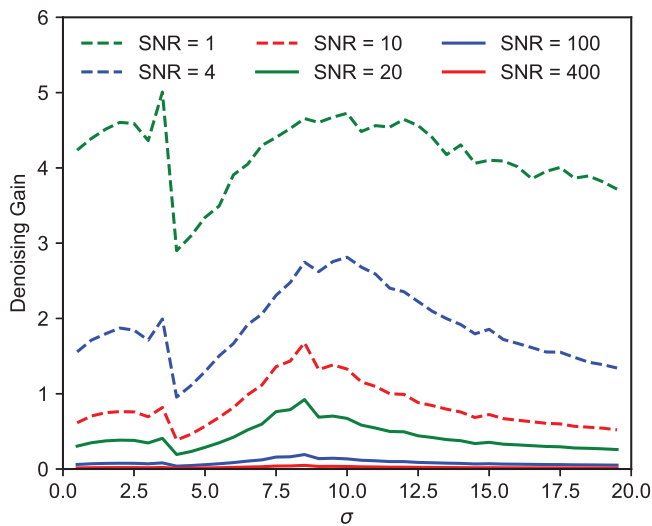


FIG. 5. Improvement in denoising gain when additional datasets with $\sigma = 8$ and $\text{SNR} = 4$ are included in the training data. Results are averaged over 10 randomly initialized RCs.

purpose, the AdEx model is selected due to its bursting and adaptation behavior, which finds significant interest in studying insect locomotion and sensory threshold.⁴¹ Appendix C presents the non-linear differential equations of this model, explaining the sensory behavior of the neurons in terms of the variation in the membrane potential $V(t)$ when driven by the current input $I(t)$. The membrane potential also depends on an independent variable, named adaptation parameter $w(t)$, simulating non-trivial firing and adaptation neuronal behavior.

Three different noise processes are considered, varying in terms of the frequency content, comprising GWN, violet, and pink noise processes. The first 200 ms is treated as the training data, whereas the next 200 ms is considered as validation data. Reservoir nodes are not considered fixed and are released to vary between 50 and 100 to represent cases in which the number of reservoir nodes is variable. Other hyperparameters are allowed to vary within the intervals specified earlier for the Lorenz attractor. According to Appendix C, simulated data are generated at 100 kHz sampling rate for a period of 400 ms and polluted with 10% RMS measurement noise (20 dB). The simulated current is considered a step function starting at 10 ms for a fixed interval of 390 ms, with a magnitude of 65 pA. Firing patterns are simulated through Dirac delta functions, implemented in the adaptation parameter equation. Figure 6 shows the ground truth

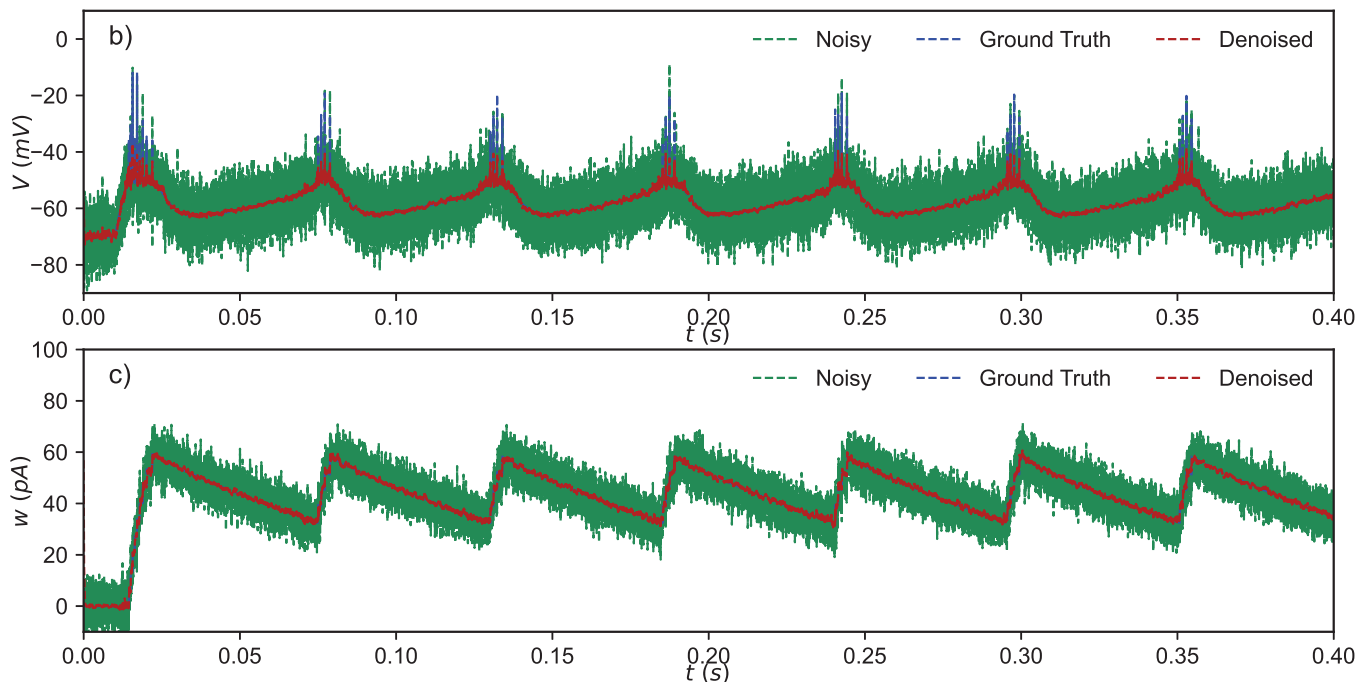


FIG. 6. Reproducing noisy measurements of the AdEx model through the proposed RC framework ($\text{SNR} = 20$ dB). The ground truth data appear in dashed blue lines, representing a neuronal behavior subjected to an initial firing pattern, followed by subsequent adaptation. The noisy and denoised curves are shown in green and red colors. (a) Membrane potential is shown subject to fixed current input and firing patterns triggered through Dirac delta functions. (b) Adaptation variable is specified in the same substance as the input current with initial firing patterns and periodic spiking behavior.

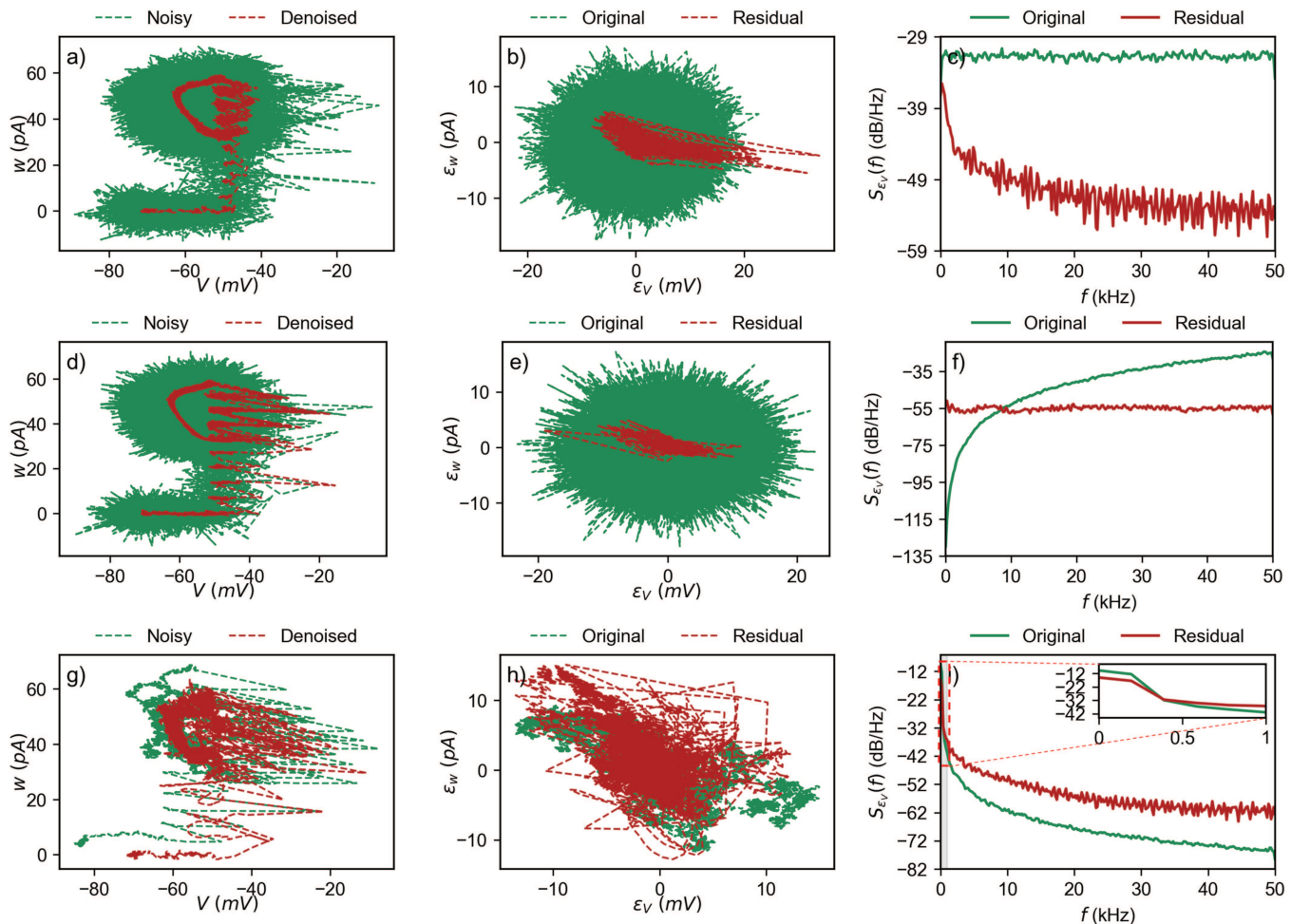


FIG. 7. Denoising performance comparison between different additive noise processes: (a)–(c) GWN $S(f) = S_0$; (d)–(f) violet noise $S(f) \propto f$; (g)–(i) pink noise $S(f) \propto 1/f$. The left side subplots show the denoised membrane potential and adaptation variable in the phase space compared to the noisy ones. The middle column subplots compare residual noise with the originally added noise in the phase space. The right side subplots show the averaged PSD of the residual and original noise processes.

membrane potential and adaptation variable along with dynamics contaminated with GWN. The results of the pruned RC are also shown in Fig. 6, presenting accurate results across both training and testing datasets when compared with the ground truth. A substantial amount of noise is discarded, and the dynamic patterns are made more evident through the presented RC framework. However, the most significant mismatch between the denoised and the ground truth membrane potential prevails at the bursting intervals, where sharp peaks are not well captured. This issue can originate from the mathematical expression of ESN, which seemingly simulates smooth dynamics better.

The GWN process contains the same level of noise across the entire frequency band, but this assumption can be relaxed by considering colored noise. Thus, we investigate the denoising performance

under violet and pink noise, where the high- and low-frequency components are dominant, respectively. The power spectral density (PSD) of violet noise varies linearly with the frequency, i.e., $S(f) \propto f$, whereas the PSD of pink noise is inversely proportional to the frequency, i.e., $S(f) \propto 1/f$. Figure 7 compares the denoised dynamics with the noisy ones in the phase space for the three noise processes. The denoising residual errors are also indicated in the phase space and the frequency domain for comparison purposes. These errors are denoted by $\varepsilon_V(t) = V(t) - \hat{V}(t)$ and $\varepsilon_w(t) = w(t) - \hat{w}(t)$, where $\hat{V}(t)$ and $\hat{w}(t)$ represent the ground truth membrane potential and adaptation parameter, respectively.

Specifically for GWN, the noise has been reduced considerably, as shown in Fig. 7(a). The subtraction of the denoised signal and the ground truth one is shown in Fig. 7(b), highlighting the capability of

RC to discard GWN to a large extent. As can be seen, the errors are more pronounced along the horizontal axis. This observation can be justified based on the time-domain comparison shown in Fig. 6, where the voltage errors $\varepsilon_V(t)$ are large around the sharp peaks, whereas the adaptation parameter errors $\varepsilon_w(t)$ are small. Appendix D provides further comparison between the PSD curves of the noisy and denoised signals in comparison to the ground truth PSD.

The PSD of the membrane potential error, $S_{\varepsilon_V}(f)$, is shown in Fig. 7(c). Based on this figure, the reduction in the noise magnitude varies between 5 and 25 dB, depending on the frequency range. In effect, the residual noise is stronger in the low-frequency range of [0–5 kHz], but it reaches a weaker and stable regime around –50 dB/Hz at frequencies above 5 kHz, as shown in Fig. 7(c). Similar comparisons are made for violet noise in Figs. 7(d)–7(f). The denoising performance is good. Notably, the residual noise appears similar to a GWN process with an almost fixed PSD equal to –55 dB/Hz. The original additive noise has a smaller magnitude at frequencies below 10 kHz, meaning that the RC produces extra noise in this particular frequency range. However, above 10 kHz, the denoising performance becomes significant, improving up to 30 dB/Hz.

The pink noise primarily appears as some shift in the phase space, as shown in Fig. 7(g). The RC mitigates noise at frequencies below 350 Hz, but above this range, it produces extra noise. Overall, the results display limited merits for RC in discarding pink noise.

The results presented above correspond to one specific configuration of reservoir nodes and edges. Figure 8 shows the distribution of denoising gain [see Eq. (7)] when RCs are initialized with random nodal and connectivity configurations and allowed to have an optimal number of nodes between 50 and 100. The results remain qualitatively consistent across different reservoir configurations. Additionally, it can be confirmed that the best denoising gain (mean: 12.68; SD: 1.13) is achieved for violet noise, wherein high-frequency components are greatly dominant. For GWN, an average denoising gain of 5.50 is acquired with a small standard deviation of 0.36. Finally, for pink noise, the average denoising gain is 1.32 with 0.07 standard deviation, which is still above 1.0, although the merits of denoising in this case are minimal.

V. DISCUSSION

The examples above primarily demonstrate the RC system's ability to discard additive noise. Denoising performance is evaluated using both the smooth dynamics of the Lorenz attractor and the spiking patterns of the AdEx model. While RC performs well in both scenarios, it achieves higher accuracy with smooth dynamics. This is attributed to the mathematical formulation of RCs, where the state is expressed as a linear combination of the previous state and input using a smooth activation function.

To better handle discontinuities, alternative activation functions may be needed, ones that can approximate abrupt changes more efficiently. In this context, architectures based on spiking neuro-physical models, such as liquid state machines, could offer greater flexibility for simulating bursting nonlinear dynamics.²⁸ The mathematical structure of RC appears to be problem-specific and should be tailored accordingly.

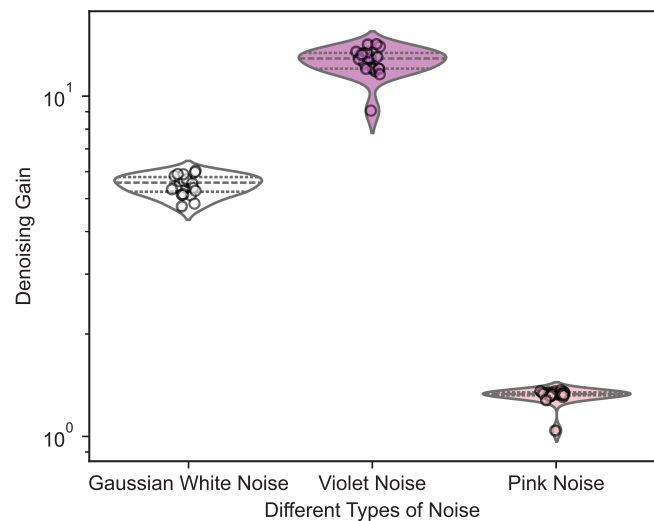


FIG. 8. Distribution of denoising gain for different types of noise, showing sensitivity of the results to the initial configuration of the reservoir.

The denoising performance also varies with the SNR. For lower SNRs, the RC significantly reduces noise, yielding a denoising gain well above unity. However, at very high SNRs, the RC struggles to distinguish low-amplitude noise from dominant nonlinear dynamics, sometimes resulting in adverse denoising effects (see Fig. 3). Therefore, adequate care should be given to avoid misusing the RC.

The frequency characteristics of noise play a significant role in the denoising performance of reservoir computing. The RC demonstrates strong results when dealing with GWN and violet noise, effectively suppressing high-frequency components. However, its performance deteriorates with pink noise, where low-frequency components dominate. This limitation is expected, as low-frequency noise tends to overlap with the underlying nonlinear dynamics, making it difficult for the RC to distinguish signal from noise. This observation suggests that the denoising capabilities of RC are most effective in scenarios involving high-frequency noise at low SNR levels. Consequently, the primary hypothesis of this study requires slight refinement to reflect that RC's denoising strengths are frequency-dependent.

These findings open up new avenues toward combining the RC with wavelet scattering⁴² and less-studied nonlinear phase space-based filtering methods like the GHKSS approach.⁴³ While these hybrid strategies were not exercised herein, they represent compelling directions for enhancing RC's performance in more complex noise environments.

Another important feature of the RC is its generalization capability when the dynamical parameters change. The RC maintains its denoising gain above one across a relatively wide neighborhood of the training parameters. Including extra training datasets with varying dynamical parameters allows for acquiring higher accuracy when dealing with potential parameter changes. This adaptability is

crucial for real-world applications where system dynamics may not remain constant.

While expanding the training dataset improves generalization, it also increases computational costs. Specifically, when a hyperparameter optimization is required, the cost scales cubically with the dataset size due to matrix inversion in Eq. (4). Therefore, careful selection of training parameters is essential to balance accuracy and computational efficiency.

The computational cost of the ridge regression is proportional to $O(N^3)$ due to matrix multiplication for calculating the reservoir states, where N is the number of reservoir nodes. The ridge regression requires a matrix multiplication and inversion of the same dimension as the reservoir matrix, having a computational cost proportional to $O(N^3)$ if the Gauss–Jordan approach is employed for the matrix inversion.⁴⁴ Pruning reservoir nodes can reduce the dimensionality, and if the number of pruned nodes is M , the computational complexity can be characterized as $O[(N - M)^3]$. This reduction can be considerably large depending on the number of pruned nodes. However, this assessment of the computational cost requires accounting for the computational efforts related to pruning the reservoir. In fact, at the training stage, the cost of pruning nodes and edges is considerable as it scales up with the number of iterations required to search for M ineffective reservoir nodes.

The pruning method used to reduce the number of reservoir nodes is effective only when the node count is relatively small. The current greedy approach identifies redundant nodes based on a performance criterion like MSE, but its scalability is limited. This limitation could be improved by adopting graph-based optimization techniques, which allow for global reduction of reservoir nodes and edges without exhaustive point-by-point evaluation.⁴⁵

VI. CONCLUSION

A new RC framework was proposed for the reconstruction and denoising of nonlinear dynamics from noisy measured data. The performance of the computing reservoir was investigated in terms of the noise intensity, the sensitivity to dynamical parameters, and the noise frequency content. The optimization of the hyperparameters, the reservoir nodes, the input connectivity, the readout nodes, and the ridge regression coefficient was proven to be necessary and effective to attain a well-trained reservoir computer. Reducing the reservoir dimensionality and removing ineffective edges through a pruning technique significantly improved denoising accuracy, surpassing the performance of the classical reservoir with a fixed number of nodes and random connectivity. At high noise levels, the proposed RC outperformed an EKF implementation. This superior performance was achieved despite the fact that the EKF had an advantage as it used precise physical knowledge through the state-space model, while the RC implicitly learned the dynamics solely from the training data. Since the EKF is a near-optimal filter, having observed superior performance for the RC is of great importance from an optimality standpoint.

The denoising performance of RC was observed to be sensitive to the dynamical parameters, but the RC maintained its accuracy in the vicinity of the parameters used for generating the training data. As the validation dataset deviates from the trained neighborhood of the parameters, the denoising gain drops considerably. When

additional data were incorporated into the training set by using the same nonlinear dynamics but with an altered physical parameter, the denoising gain improved, showing a bi-modal pattern with two peaks positioned in the vicinity of the parameters used for training. This improvement can become the basis of training RC on dedicated training data to acquire generalization across a wider interval in the domain of physical parameters.

The pruning technique used for reducing the dimension of the RC and eliminating redundant edges proved to be an effective approach to further reduce computational costs while enhancing accuracy. However, since the optimization was implemented based on searching for the most ineffective nodes and edges, it involves high computational costs, which scale up with the reservoir dimension. Our future works will address this problem by transitioning into a graph-based optimization algorithm, which can potentially overcome the drawbacks of searching across all nodes and edges.

ACKNOWLEDGMENTS

This research has received funding from the Australian Research Council Linkage Project under Grant No. LP200301196. Additionally, the first draft of this manuscript was prepared during O. Sedehi's visit to TU Berlin, supported by the Key Technology Partnership (KTP) SEED funding scheme. This financial support is gratefully appreciated. M. Stender and M. Yadav acknowledge financial support from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under the Special Priority Program (SPP 2353: Daring More Intelligence—Design Assistants in Mechanics and Dynamics, Project No. 501847579).

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Omid Sedehi: Conceptualization (lead); Data curation (lead); Formal analysis (lead); Investigation (lead); Methodology (lead); Software (lead); Validation (lead); Visualization (lead); Writing – original draft (lead); Writing – review & editing (lead). **Manish Yadav:** Investigation (equal); Methodology (equal); Visualization (equal); Writing – review & editing (equal). **Merten Stender:** Conceptualization (equal); Funding acquisition (equal); Investigation (equal); Methodology (equal); Supervision (equal); Visualization (equal); Writing – review & editing (equal). **Sebastian Oberst:** Conceptualization (equal); Funding acquisition (lead); Methodology (equal); Project administration (lead); Supervision (equal); Writing – review & editing (equal).

DATA AVAILABILITY

The data that support the findings of this study are openly available in GitHub at <https://github.com/Omid-Sedehi/Reservoir-Computer>, Ref. 48.

APPENDIX A: EXTENDED KALMAN FILTER

It is assumed that the dynamical system under consideration can be described in discrete-time through a nonlinear state-space relationship given as

$$\mathbf{x}_k = \mathbf{g}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) + \mathbf{v}_k, \quad (\text{A1})$$

where \mathbf{x}_k is the dynamical state at discrete time $t_k = k\Delta t$, k takes on values from $\{1, 2, \dots, N\}$, representing individual time steps, \mathbf{u}_k is the dynamical input applied at time t_k , and \mathbf{v}_k is the process noise considered to be zero-mean GWN with \mathbf{Q} covariance matrix.

Likewise, a non-linear observation model can be established by relating the state and input vectors to the observed data. This essentially leads to

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}_k. \quad (\text{A2})$$

In this equation, \mathbf{y}_k is the measured vector, $\mathbf{h}(\cdot)$ is the nonlinear observation model, and \mathbf{w}_k is the observation noise considered zero-mean GWN with \mathbf{R} covariance matrix.

Through a linearization around the existing system state, it is possible to estimate the system states using the EKF. By doing so, the following sequential algorithm can be obtained:²

$$\mathbf{x}_{k|k-1} = \mathbf{g}(\mathbf{x}_{k-1|k-1}, \mathbf{u}_k), \quad (\text{A3})$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_{k-1} \mathbf{P}_{k-1|k-1} \mathbf{F}_{k-1}^T + \mathbf{Q}, \quad (\text{A4})$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R})^{-1}, \quad (\text{A5})$$

$$\mathbf{x}_{k|k} = \mathbf{x}_{k|k-1} + \mathbf{K}_k (\mathbf{z}_k - \mathbf{h}(\mathbf{x}_{k|k-1}, \mathbf{u}_k)), \quad (\text{A6})$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{H}_k \mathbf{P}_{k|k-1}. \quad (\text{A7})$$

Here, $\mathbf{x}_{k|k-1}$ is the predicted state, $\mathbf{x}_{k|k}$ is the estimated state, $\mathbf{P}_{k|k-1}$ is the prediction error covariance matrix, $\mathbf{P}_{k|k}$ is the updated error covariance matrix, \mathbf{K}_k is the Kalman gain matrix, \mathbf{F}_{k-1} is the derivative of $\mathbf{g}(\cdot)$ with respect to \mathbf{x} evaluated at $\mathbf{x}_{k-1|k-1}$, and \mathbf{H}_k is the derivative of the observation model $\mathbf{h}(\cdot)$ with respect to \mathbf{x}_k evaluated at $\mathbf{x}_{k|k-1}$.

Given the noise covariance matrices, the above sequential formulations provide an efficient estimate of the system states.

APPENDIX B: NONLINEAR EQUATIONS OF THE LORENZ SYSTEM

The Lorenz system consists of three major system state variables, which govern the atmospheric convection through the following set of equations:⁴⁶

$$\frac{dx}{dt} = \sigma(y - x), \quad (\text{B1})$$

$$\frac{dy}{dt} = x(\rho - z) - y, \quad (\text{B2})$$

$$\frac{dz}{dt} = xy - \beta z, \quad (\text{B3})$$

where σ is the Prandtl number, ρ is the Rayleigh number, and β is the dissipation rate. Note that selecting different parameters can entirely change the dynamic regimes. The dynamics of the Lorenz system are nonlinear, chaotic, and smooth deterministic, which depend heavily on the initial conditions.

APPENDIX C: MATHEMATICAL FORMULATION OF THE AdEx MODEL

The nonlinear differential equations describing the adaptive exponential integrate-and-fire model are provided in the following:⁴⁷

$$\tau_m C \frac{dV}{dt} = -(V - V_r) + \Delta_T \exp\left(\frac{V - V_T}{\Delta_T}\right) - R w + R I, \quad (\text{C1})$$

$$\tau_w \frac{dw}{dt} = a(V - V_r) - w + b \tau_w \sum_{t_f} \delta(t - t_f), \quad (\text{C2})$$

where V is the voltage, I is the input current, w is the adaptation parameter, V_r is the leak reversal potential, t_f is the firing times at which the voltage exceeds V_T threshold, and $\delta(\cdot)$ is the Dirac delta function. Other parameters are constant coefficients: $\tau_m = 5$ ms is membrane time scale, $\tau_w = 100$ ms is adaptation time constant, $R = 500$ M Ω is membrane resistance, $V_r = -55$ mV is rest potential, $V_T = -51$ mV is voltage threshold, $\Delta_T = -2$ mV is sharpness threshold, $a = -0.5$ ns is adaptation voltage coupling, and $b = 7$ pA is spike-triggered adaptation increment. At spike time $t = t_f$, there will be a sharp rise in the adaptation parameter equal to b , as described via the Dirac delta function. The dynamics of the AdEx model are nonlinear and can be chaotic, showing non-smooth deterministic behavior, owing to the threshold-reset mechanism.

APPENDIX D: COMPARISON OF PSD CURVES IN AdEx EXAMPLE

The results presented in Fig. 7 compare the PSD of the membrane potential prediction errors, i.e., $S_{\varepsilon_V}(f)$. Figure 9 supplements those results, showing the PSD of the ground truth membrane potential, $S_V(f)$, compared to those of the noisy and denoised signals. Note that the difference of the noisy and denoised signals from the ground truth signal is indicated in the PSD curves of Fig. 7 and labeled as “original” and “residual.” These results further confirm the above conclusions, showing the performance of RC in handling different types of noise and emphasizing that the RC is more successful in discarding high-frequency noise.

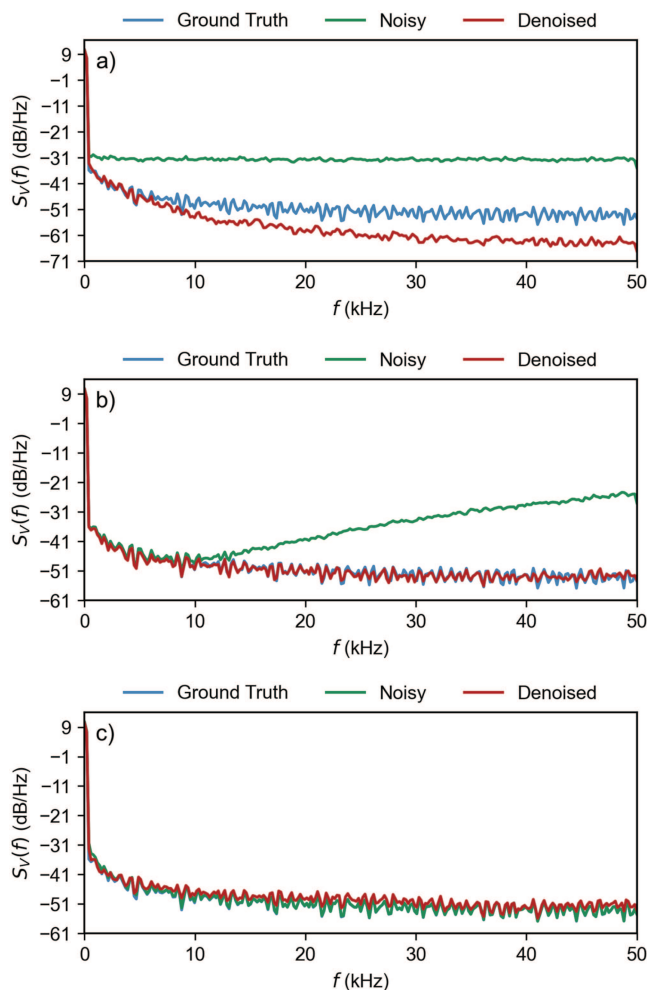


FIG. 9. PSD curves of the membrane potential when considering additive (a) GWN, (b) violet noise, and (c) pink noise. “Ground Truth” corresponds to the noise-free original signal; “Noisy” represents the noisy signal fed into the RC framework; and “Denoised” represents the reconstructed signal.

REFERENCES

- ¹D. L. Donoho, “De-noising by soft-thresholding,” *IEEE Trans. Inf. Theory* **41**, 613–627 (1995).
- ²B. D. Anderson and J. B. Moore, *Optimal Filtering*, Prentice-Hall Information and System Sciences Series (Prentice-Hall, 1979).
- ³A. Ghani, T. M. McGinnity, L. Maguire, L. McDaid, A. Belatreche, and N. Shabtai, “Neuro-inspired speech recognition based on reservoir computing,” in *Advances in Speech Recognition* (IntechOpen, 2010).
- ⁴D. J. Gauthier, E. Bollt, A. Griffith, and W. A. S. Barbosa, “Next generation reservoir computing,” *Nat. Commun.* **12**, 1–8 (2021).
- ⁵B. Thorne, T. Jüngling, M. Small, D. Corrêa, and A. Zaitouny, “Reservoir time series analysis: Using the response of complex dynamical systems as a universal indicator of change,” *Chaos* **32**, 033109 (2022).
- ⁶S. Shahi, F. H. Fenton, and E. M. Cherry, “A machine-learning approach for long-term prediction of experimental cardiac action potential time series using an autoencoder and echo state networks,” *Chaos* **32**, 063117 (2022).

- ⁷F. A. Araujo, M. Riou, J. Torrejon, S. Tsunegi, D. Querlioz, K. Yakushiji, A. Fukushima, H. Kubota, S. Yuasa, M. D. Stiles, and J. Grollier, “Role of non-linear data processing on speech recognition task in the framework of reservoir computing,” *Sci. Rep.* **10**, 328 (2020).
- ⁸J. Pathak, Z. Lu, B. R. Hunt, M. Girvan, and E. Ott, “Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data,” *Chaos* **27**, 121102 (2017).
- ⁹S. Krishnagopal, M. Girvan, E. Ott, and B. R. Hunt, “Separation of chaotic signals by reservoir computing,” *Chaos* **30**, 023123 (2020).
- ¹⁰T. L. Carroll, “Creating new chaotic signals with reservoir computers,” *Chaos, Solitons Fractals* **164**, 112688 (2022).
- ¹¹L. Jaurigue, E. Robertson, J. Wolters, and K. Lüdge, “Reservoir computing with delayed input for fast and easy optimisation,” *Entropy* **23**, 1560 (2021).
- ¹²A. Flynn, V. A. Tsachouridis, and A. Amann, “Multifunctionality in a reservoir computer,” *Chaos* **31**, 013125 (2021).
- ¹³U. Parlitz, “Learning from the past: Reservoir computing using delayed variables,” *Front. Appl. Math. Stat.* **10**, 1221051 (2024).
- ¹⁴M. Yadav, S. Sinha, and M. Stender, “Evolution beats random chance: Performance-dependent network evolution for enhanced computational capacity,” *Phys. Rev. E* **111**, 014320 (2025).
- ¹⁵M. Yadav, S. Chauhan, M. D. Shrimali, and M. Stender, “Predicting multi-parametric dynamics of an externally forced oscillator using reservoir computing and minimal data,” *Nonlinear Dyn.* **113**, 5977–5990 (2025).
- ¹⁶N. Semenova, X. Porte, L. Andreoli, M. Jacquot, L. Larger, and D. Brunner, “Fundamental aspects of noise in analog-hardware neural networks,” *Chaos* **29**, 103128 (2019).
- ¹⁷I. Estébanez, I. Fischer, and M. C. Soriano, “Constructive role of noise for high-quality replication of chaotic attractor dynamics using a hardwarebased reservoir computer,” *Phys. Rev. Appl.* **12**, 034058 (2019).
- ¹⁸A. Wikner, J. Harvey, M. Girvan, B. R. Hunt, A. Pomerance, T. Antonsen, and E. Ott, “Stabilizing machine learning prediction of dynamics: Novel noise-inspired regularization tested with reservoir computing,” *Neural Networks* **170**, 94–110 (2024).
- ¹⁹A. Röhm, D. J. Gauthier, and I. Fischer, “Model-free inference of unseen attractors: Reconstructing phase space features from a single noisy trajectory using reservoir computing,” *Chaos* **31**, 103127 (2021).
- ²⁰C. Nathe, C. Pappu, N. Mecholsky, J. Hart, T. Carroll, and F. Sorrentino, “Reservoir computing with noise,” *Chaos* **33**, 041101 (2023).
- ²¹I. Lee, Y. Kim, T. Kim, H. Choi, S. Y. Yang, and Y. Kim, “Recursive reservoir concatenation for salt-and-pepper denoising,” *Pattern Recognition* **160**, 111196 (2025).
- ²²J. Choi and P. Kim, “Signal-noise separation using unsupervised reservoir computing,” *arXiv:2404.04870* (2024).
- ²³X. Dutoit, B. Schrauwen, J. Van Campenhout, D. Stroobandt, H. Van Brussel, and M. Nutton, “Pruning and regularization in reservoir computing,” *Neurocomputing* **72**, 1534–1546 (2009).
- ²⁴S. Scardapane, D. Comminiello, M. Scarpiniti, and A. Uncini, “Significance-based pruning for reservoir’s neurons in echo state networks,” in *Advances in Neural Networks: Computational and Theoretical Issues* (Springer International Publishing, Cham, 2015), pp. 31–38.
- ²⁵A. Haluszczynski, J. Aumeier, J. Herteux, and C. Râth, “Reducing network size and improving prediction stability of reservoir computing,” *Chaos* **30**, 063136 (2020).
- ²⁶H. Jaeger, “Echo state network,” *Scholarpedia* **2**, 2330 (2007).
- ²⁷W. Maass, T. Natschlaeger, and H. Markram, “Real-time computing without stable states: A new framework for neural computation based on perturbations,” *Neural Comput.* **14**, 2531 (2002).
- ²⁸H. Hazan and L. M. Manevitz, “Topological constraints and robustness in liquid state machines,” *Expert Syst. Appl.* **39**, 1597–1606 (2012).
- ²⁹B. Ren and H. Ma, “Global optimization of hyper-parameters in reservoir computing,” *Electron. Res. Arch.* **30**, 2719–2729 (2022).
- ³⁰A. Hoerl and R. Kennard, “Ridge regression: Biased estimation for nonorthogonal problems,” *Technometrics* **12**, 55–67 (1970).
- ³¹L. Bottou, “On-line learning and stochastic approximations,” in *On-Line Learning in Neural Networks* (Cambridge University Press, 1999), pp. 9–42.

- ³²D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature* **323**, 533–536 (1986).
- ³³B. Hanin, "Which neural net architectures give rise to exploding and vanishing gradients?" *Adv. Neural Inform. Process. Syst.* **31**, 582–591 (2018).
- ³⁴S. Shahi, F. H. Fenton, and E. M. Cherry, "Prediction of chaotic time series using recurrent neural networks and reservoir computing techniques: A comparative study," *Machine Learn. Appl.* **8**, 100300 (2022).
- ³⁵P. Kaelo and M. M. Ali, "Some variants of the controlled random search algorithm for global optimization," *J. Optim. Theory Appl.* **130**, 253–264 (2006).
- ³⁶X. Li, M. Small, and Y. Lei, "Reservoir computing with higher-order interactive coupled pendulums," *Phys. Rev. E* **108**, 064304 (2023).
- ³⁷J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.* **13**, 281–305 (2012).
- ³⁸T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistic Learning: Data Mining, Inference, and Prediction* (Springer, 2017).
- ³⁹R. M. Kent, W. A. S. Barbosa, and D. J. Gauthier, "Controlling chaos using edge computing hardware," *Nat. Commun.* **15**, 3886 (2024).
- ⁴⁰A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using Networkx," in *Proceedings of the 7th Python in Science Conference (SciPy2008)*, edited by G. Varoquaux, T. Vaught, and J. Millman (Scipy, Pasadena, CA, 2008), pp. 11–15.
- ⁴¹T. van der Veen, Y. Cohen, E. Chicca, and V. Dürr, "A spiking neural network model for proprioception of limb kinematics in insect locomotion," bioRxiv:10.1101/2024.09.27.615365 (2024).
- ⁴²S. Mallat, "Group invariant scattering," *Commun. Pure Appl. Math.* **65**, 1331–1398 (2012).
- ⁴³P. Grassberger, R. Hegger, H. Kantz, C. Schaffrath, and T. Schreiber, "On noise reduction methods for chaotic data," *Chaos* **3**, 127–141 (1993).
- ⁴⁴G. H. Golub and C. F. V. Loan, *Matrix Computations*, Johns Hopkins Studies in the Mathematical Sciences, 4th ed. (Johns Hopkins University Press, Baltimore, 2013).
- ⁴⁵F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.* **20**, 61–80 (2009).
- ⁴⁶E. N. Lorenz, "Deterministic nonperiodic flow," *J. Atmosph. Sci.* **20**, 130–141 (1963).
- ⁴⁷W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition* (Cambridge University Press, 2014).
- ⁴⁸O. Sedehi, "ReservoirComputer (Truncated Reservoir Computer)," GitHub repository, see <https://github.com/Omid-Sedehi/ReservoirComputer> (accessed August, 17 2025).