

Reservoir computing with noise

Cite as: Chaos 33, 041101 (2023); doi: 10.1063/5.0130278

Submitted: 10 October 2022 · Accepted: 9 March 2023 ·

Published Online: 25 April 2023









View Online



Export Citation



CrossMark

Chad Nathe,¹  Chandra Pappu,²  Nicholas A. Mecholsky,³  Joe Hart,⁴  Thomas Carroll,⁴  and Francesco Sorrentino^{1,a)} 

AFFILIATIONS

¹Mechanical Engineering Department, University of New Mexico, Albuquerque, New Mexico 87131, USA

²Electrical, Computer and Biomedical Engineering Department, Union College, Schenectady, New York 12309, USA

³Department of Physics and Vitreous State Laboratory, The Catholic University of America, Washington, DC 20064, USA

⁴US Naval Research Laboratory, Washington, DC 20375, USA

^{a)} Author to whom correspondence should be addressed: fsorrent@unm.edu

ABSTRACT

This paper investigates in detail the effects of measurement noise on the performance of reservoir computing. We focus on an application in which reservoir computers are used to learn the relationship between different state variables of a chaotic system. We recognize that noise can affect the training and testing phases differently. We find that the best performance of the reservoir is achieved when the strength of the noise that affects the input signal in the training phase equals the strength of the noise that affects the input signal in the testing phase. For all the cases we examined, we found that a good remedy to noise is to low-pass filter the input and the training/testing signals; this typically preserves the performance of the reservoir, while reducing the undesired effects of noise.

Published under an exclusive license by AIP Publishing. <https://doi.org/10.1063/5.0130278>

In any practical application, noise will inevitably affect the signals that are processed; this is also true in machine learning. Therefore, the robustness of a machine learning technique to the noise-corruption of input and training data is an important question. In this work, we investigate the effect of noisy signals on the performance of a reservoir computer acting as an observer. A reservoir observer is an application of reservoir computing in which the internal state of a system is reconstructed from knowledge of one or more measured state variables. We find low-pass filtering of the noisy signals that are used by the reservoir observer to be an effective remedy against additive Gaussian noise, provided that the same type of filtering is applied in the training phase and in the testing phase.

I. INTRODUCTION

Noise is an unavoidable component in almost all practical applications. For example, signals obtained from biological systems are typically affected by a large amount of noise. Hence, it is important to understand how machine learning is affected by noise and what remedies can be put in place to contain its effects. In this paper, we focus on reservoir computing^{1,2} as a particular type of machine

learning and in particular on reservoir observers, which use knowledge of part of the state of a system to reconstruct the internal state of the same system.³

One thing that makes reservoir computers interesting is that they may be implemented as massively parallel devices in analog hardware. This, combined with the simplicity of training, makes them promising for applications such as drones or handheld sensors that require small size, low weight, and low power consumption. Reservoir computers that are all or partly analog include photonic systems,^{4–9} analog electronic circuits,¹⁰ mechanical systems,¹¹ and field programmable gate arrays.¹² Many other examples are included in the review paper Ref. 13.

A number of authors have considered how noise added to the input, testing, or training signals affects the performance of reservoir computing. In Ref. 14, the authors used the concept of consistency to show how added noise decreased the information processing capacity of reservoir computers. Vettelschoss *et al.*¹⁵ demonstrated the change in information processing capacity in a single-node reservoir computer. Shougat *et al.*¹⁶ considered noise as the masking function for a reservoir computer based on a Hopf oscillator. References 17 and 18 examined the effect of added noise on the ability to classify different chaotic signals. In Ref. 19, the authors sought to mitigate the effects of added noise by using a bistable function for the activation function to take advantage of the principle of stochastic

resonance. The authors of Ref. 3 studied the effects of noise on the input signal for the case of a reservoir observer. The authors of Ref. 20 investigated the impact of measurement noise on the input data for Lyapunov exponent estimation using reservoir computing. The authors of Ref. 21 studied the effects of noise on signal-to-noise ratio in reservoir computers with no nonlinearity. In Refs. 22–24, noise occupied a constructive role in attractor reconstruction tasks. In these papers, noise added in the training stage allowed the reservoir computer to learn behaviors in parameter ranges that were not part of the training data. A recent paper²⁵ has investigated the effects of noise in an experimental reservoir computer.

In this work, we study how measurement noise added to the reservoir input and/or output signals affects the reservoir computer’s performance for the so-called observer task.³ This would be the situation when using the reservoir computer to model any real-world system since the input data will be corrupted by some amount of measurement noise. We also consider the general situation in which the noise strength in the input and output signals can be different in the training and testing phases, as would be the case when training is done in the laboratory and the reservoir is deployed in the field.

We find that the reservoir performs best when the measurement noise on the input signal in the training and testing phases has the same strength. Since it may not always be possible to match the noise in the training phase to the noise in the testing phase, we propose applying a simple low-pass filter that is well-matched to the true signal spectrum to the input signal. We find that such a filter significantly mitigates the negative effect of the noise.

The rest of this paper is organized as follows. In Sec. II, we introduce the reservoir equations. In Sec. III, we describe the effects of noise on the training error and the testing error. In Sec. IV, we introduce low-pass filtering as a remedy to noise. Finally, the conclusions are given in Sec. V.

II. RESERVOIR COMPUTING IN THE PRESENCE OF NOISE

In this section, we present a general formulation of the reservoir dynamics in the presence of noise. A signal from the underlying system to be observed is used to drive the reservoir (the input signal), and the reservoir is trained to reproduce a second signal from the underlying system (the training signal). However, in our formulation, both the input and training signals are affected by noise. In particular, we consider the case of additive white Gaussian noise (AWGN) applied to the reservoir input and output. We call $\epsilon_1 \geq 0$ the noise strength added to the input signal in the training phase, $\epsilon_2 \geq 0$ the strength of the noise added to the input signal in the testing phase, $\epsilon_3 \geq 0$ the strength of the noise added to the training signal, and $\epsilon_4 \geq 0$ the strength of the noise added to the testing signal. All of these noise sources are independent and identically distributed (IID). This is illustrated in Fig. 1 which shows how noise can affect either the input or the output signals that interact with the reservoir. The motivation for considering $\epsilon_1 \neq \epsilon_2$ and $\epsilon_3 \neq \epsilon_4$ is to account for situations in which training is performed in a laboratory and testing in the field. It is thus expected that the level of noise may vary substantially between the training and the testing phases.

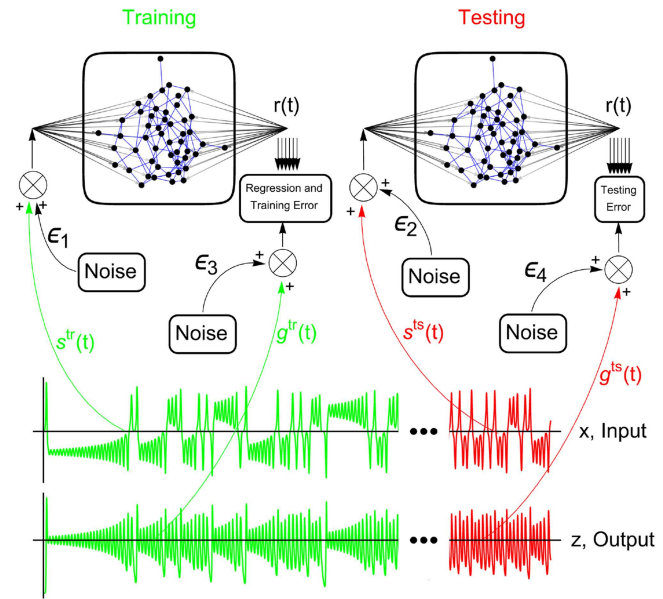


FIG. 1. Illustration of how noise can affect the input and output signals of a reservoir computer in the training and testing phase. We are performing an “observer task” in which we feed the x coordinate of the Lorenz system started from random initial conditions into the reservoir and the reservoir is trained to predict the z coordinate. We call ϵ_1 the strength of the noise affecting the input signal in the training phase, ϵ_2 the strength of the noise affecting the input signal in the testing phase, ϵ_3 the strength of the noise affecting the training signal, and ϵ_4 the strength of the noise affecting the testing signal.

In the rest of this paper, for a given noise-free signal $x(t)$, we will use the following notation: $\tilde{x}_\epsilon(t)$ is the noise-corrupted version of $x(t)$ with $\epsilon \geq 0$ being the noise strength and $\hat{x}_\epsilon(t)$ is the low-pass filtered (LPF) version of $\tilde{x}_\epsilon(t)$. We model the reservoir dynamics in discrete time, while the underlying physical system with which the reservoir interacts evolves in continuous time. Thus, input and output signals are sampled at each time step of the reservoir dynamics, with sampling period t_s . We normalize the noise-free input and output signals so that their mean is equal to zero and their standard deviation is equal to one. We then set the normalized noise-corrupted signal $\tilde{x}_\epsilon(t) = x(t) + \epsilon \sqrt{\frac{t_s}{T}} \zeta(t)$, where at each discrete time t , $\zeta(t)$ is a scalar drawn from a standard normal distribution and T is the approximate period over which the signal completes one oscillation. The weighting $\sqrt{t_s/T}$ is to renormalize the standard deviation of the sum of T/t_s standard normal values added to the reservoir.²⁶ Note that by taking the noise-free signal $x(t)$ to have mean equal zero and standard deviation equal one, the noise strength ϵ can be directly translated into a measure of signal-to-noise ratio (SNR), that is, $\text{SNR} = \epsilon^{-2}$.

The equation that models the reservoir dynamics is

$$\mathbf{r}(t + 1) = (1 - \alpha)\mathbf{r}(t) + \alpha \tanh(A\mathbf{r}(t) + \mathbf{w}_{\epsilon_1}^{zr}(t)), \quad (1)$$

where α is the leakage rate chosen in the range of $[0, 1]$, A is the coupling matrix, $\tilde{s}_{\epsilon_1}^{zr}(t)$ is the noise-corrupted input signal in

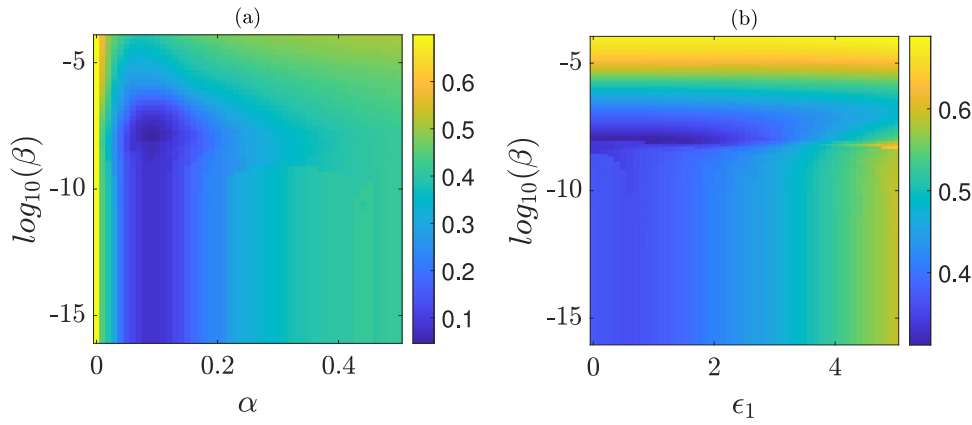


FIG. 2. Contour plot of the testing error for the Lorenz system, (a) as a function of α and β , for a case in which noise is absent from the training and testing phases and (b) as a function of ϵ_1 and β . Here, an optimized value of $\alpha = 0.1$ is considered.

the training phase, and \mathbf{w} is a vector of random elements drawn from a Gaussian distribution with mean 1 and standard deviation 0.1, i.e., $\mathcal{N}(1, 0.01)$. The matrix A is the adjacency matrix of an undirected and unweighted Erdos Renyi network with $N = 100$ nodes and connectivity probability $p = 0.5$. We set the elements on the main diagonal to be equal to zero; then normalize the matrix $A \leftarrow A/\rho(A)$, where $\rho(A)$ is the spectral radius so that the largest eigenvalue of the normalized matrix has modulus equal to 1. Overall, we found our results that follow not to be strongly affected by the particular choice of the network topology, see Sec. SIII of the [supplementary material](#) for a study of the effects of the network topology.

In this paper, we consider three different tasks (described in detail in Sec. SI in the [supplementary material](#)), which we briefly refer to as the Lorenz task, the Rössler task, and the Hindmarsh–Rose (HR) task. We optimize the parameter α with respect to the particular task assigned to the reservoir and set $\alpha = 0.1$ for the Lorenz task, $\alpha = 0.003$ for the Rössler task, and $\alpha = 0.01$ for the HR task. Further information on the optimization in α is presented in Sec. SIV in the [supplementary material](#).

From the solution of Eq. (1), one obtains the readout matrix

$$\Omega = \begin{pmatrix} r_1(1) & r_2(1) & \cdots & r_N(1) & 1 \\ r_1(2) & r_2(2) & \cdots & r_N(2) & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ r_1(T_1) & r_2(T_1) & \cdots & r_N(T_1) & 1 \end{pmatrix}, \quad (2)$$

where $r_i(t)$ is the readout of node i at time t and $t = T_1$ indicates the end of the training phase. The entries in the last column of Ω are set to 1 to account for any constant offset in the fit. We then relate the readouts to the training signal, \mathbf{g}^{tr} , with additive noise, via the unknown coefficients contained in the vector, $\boldsymbol{\kappa}$,

$$\Omega \boldsymbol{\kappa} = \tilde{\mathbf{g}}_{\epsilon_3}^{tr}, \quad (3)$$

where $\tilde{\mathbf{g}}_{\epsilon_3}^{tr}$ is the noisy training signal. We compute the unknown coefficients vector $\boldsymbol{\kappa}$ via the equation

$$\boldsymbol{\kappa} = \Omega^\dagger \tilde{\mathbf{g}}_{\epsilon_3}^{tr}. \quad (4)$$

Here, Ω^\dagger is given as

$$\Omega^\dagger = (\Omega^T \Omega + \beta \mathbf{I})^{-1} \Omega^T. \quad (5)$$

In the above equation, β is the ridge-regression parameter used to avoid overfitting³ and \mathbf{I} is the identity matrix. Next, we define the training fit signal as

$$\mathbf{h} = \Omega \boldsymbol{\kappa}. \quad (6)$$

Lastly, the training error is computed as

$$\Delta_{tr} = \frac{\langle \mathbf{h} - \tilde{\mathbf{g}}_{\epsilon_3}^{tr} \rangle}{\langle \tilde{\mathbf{g}}_{\epsilon_3}^{tr} \rangle}, \quad (7)$$

where the notation $\langle \rangle$ denotes the standard deviation.

In the testing phase, the reservoir evolves according to the equation

$$\mathbf{r}(t+1) = (1-\alpha)\mathbf{r}(t) + \alpha \tanh(\mathbf{A}\mathbf{r}(t) + \mathbf{w}\tilde{\mathbf{s}}_{\epsilon_2}^{ts}), \quad (8)$$

where $\tilde{\mathbf{s}}_{\epsilon_2}^{ts}$ is the noise-corrupted version of the input signal in the testing phase. Typically in our numerical experiments, we have the testing phase follow immediately after the training phase (but this is not a requirement). Similarly to what was done in the training phase, we compute the matrix

$$\tilde{\Omega} = \begin{pmatrix} r_1(1) & r_2(1) & \cdots & r_N(1) & 1 \\ r_1(2) & r_2(2) & \cdots & r_N(2) & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ r_1(T_2) & r_2(T_2) & \cdots & r_N(T_2) & 1 \end{pmatrix}, \quad (9)$$

where T_2 indicates the end of the testing phase and we typically set $T_2 = \frac{1}{2}T_1$. We then compute the testing fit signal by the equation

$$\tilde{\mathbf{h}} = \tilde{\Omega} \boldsymbol{\kappa}, \quad (10)$$

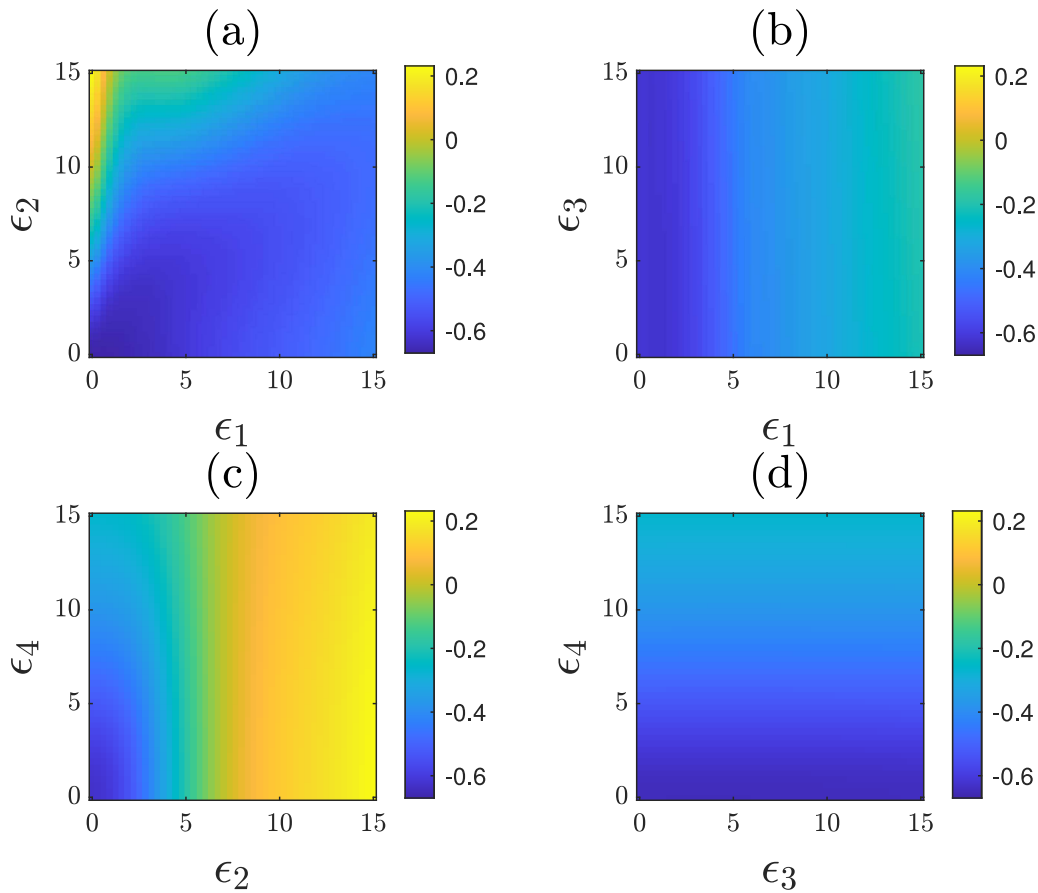


FIG. 3. Contour plots of the testing error for the Lorenz system, when (a) ϵ_1 and ϵ_2 are varied while $\epsilon_3 = \epsilon_4 = 0$, (b) ϵ_1 and ϵ_3 are varied while $\epsilon_2 = \epsilon_4 = 0$, (c) ϵ_2 and ϵ_4 are varied while $\epsilon_1 = \epsilon_3 = 0$, and (d) ϵ_3 and ϵ_4 are varied while $\epsilon_1 = \epsilon_3 = 0$. For illustrative purposes, we depicted the error as \log_{10} of the actual error.

where the vector κ is the one obtained in the training phase [Eq. (4)]. The testing error is equal to

$$\Delta_{ts} = \frac{\langle \mathbf{h} - \tilde{\mathbf{g}}_{\epsilon_4}^{ts} \rangle}{\langle \tilde{\mathbf{g}}_{\epsilon_4}^{ts} \rangle}, \tag{11}$$

where $\tilde{\mathbf{g}}_{\epsilon_4}^{ts}$ is the noise-corrupted the testing signal.

In the figures that follow, we average several simulations over the choice of the matrix A and over different noise realizations. Figure 2 illustrates our selection of the hyperparameters α and β of the reservoir computer for the Lorenz system. First, we computed the testing error as a function of the leakage rate α and of the ridge-regression parameter β . As can be seen from Fig. 2(a), in the absence of noise, the minimum testing error is obtained when $\alpha = 0.1$ and $\beta = 10^{-8}$, approximately. We then set α equal to the optimum value 0.1 and compute the testing error as a function of both β and ϵ_1 , i.e., the noise strength added to the input signal in the training phase, which is shown in Fig. 2(b). We see that as ϵ_1 is varied, the minimum testing error is always obtained when β is around 10^{-8} . Therefore, in what follows, we use the optimal values $\alpha = 0.1$ and $\beta = 10^{-8}$. We

conducted further simulations and verified that other choices of β such as 10^{-9} or 10^{-7} still produce similar qualitative results, but with only a slight change in the magnitude of the error. We conclude that our proposed methodology is not too sensitive to the particular choice of the hyperparameter β .

III. TESTING IN THE PRESENCE OF NOISE

We show in Fig. 3 contour-level plots of the testing error Δ_{ts} as we vary ϵ_1 and ϵ_2 in (a), ϵ_1 and ϵ_3 in (b), ϵ_2 and ϵ_4 in (c), and ϵ_3 and ϵ_4 in (d). In Fig. 3(a), the noise strengths of the training input signal ϵ_1 and testing input signal ϵ_2 are varied while keeping $\epsilon_3 = \epsilon_4 = 0$. Optimal performance is achieved when both ϵ_2 is roughly equal to ϵ_1 and they are both low, as expected. It is quite natural that training and testing should be performed under the same environmental conditions, which explains why the performance is good when $\epsilon_1 \approx \epsilon_2$.

When ϵ_1 is small and ϵ_2 is large (corresponding to a small input noise while training and a large input noise while testing), the testing error Δ_{ts} is very high (e.g., $\epsilon_1 < 5$ and $\epsilon_2 > 10$). On the other

hand, when the reservoir is trained with highly noisy signals and the testing input signal has lower noise levels (e.g., $\epsilon_1 > 10$ and $\epsilon_2 < 5$), the testing error is reduced by an order of magnitude. The important takeaway from Fig. 3(a) is that it is best to train a reservoir computer on an input signal with the same amount of noise as will be present during operation. In situations where an estimate of the noise that will be present during operation is difficult to obtain ahead of time, it is better to train a reservoir computer on an input signal with too much noise (compared to what it will receive in operation) than with too little.

In Fig. 3(b), ϵ_1 and ϵ_3 [the noise strength on the training signal $g^{tr}(t)$] are varied. In this case, $\epsilon_2 = \epsilon_4 = 0$ is considered. The reservoir computer is trained with input $\tilde{s}_{\epsilon_1}^{tr}(t)$, which affects the reservoir dynamics $\mathbf{r}(t)$. For a large ϵ_1 , $\tilde{s}_{\epsilon_1}^{tr}(t)$ has an increased amplitude and generates a qualitatively different response in the nonlinear reservoir than for the case of small ϵ_1 . Since the testing signal does not have any noise, Δ_{ts} proportionally increases to ϵ_1 . On the other hand, from Eq. (10), fitting a training signal is identical to averaging the uncorrelated noise from each reservoir node. Hence, Δ_{ts} is quite robust to changes in ϵ_3 .

In Fig. 3(c), ϵ_2 and ϵ_4 [the noise strength on the testing signal $g^{ts}(t)$] are varied and $\epsilon_1 = \epsilon_3 = 0$ is considered as 0. From equation (8), the reservoir dynamics in the testing phase is determined by $\tilde{s}_{\epsilon_2}^{ts}$ and is altered as ϵ_2 changes. However, $\tilde{g}_{\epsilon_4}^{ts}(t)$ appears only in the evaluation of the testing error Eq. (11) and does not affect the reservoir dynamics. Therefore, ϵ_2 dominates the testing error for higher values of ϵ_2 . An important takeaway from Figs. 3(b) and 3(c) is that noise on the reservoir input signal can be much more problematic than noise on the training or testing output signals.

In Fig. 3(d), the testing error is computed as a function of ϵ_3 and ϵ_4 while assuming $\epsilon_1 = \epsilon_2 = 0$. Similar to Fig. 3(b), the testing error Δ_{ts} is robust to ϵ_3 , due to the averaging in the training stage. However, with an increase in ϵ_4 , the testing error Δ_{ts} , which is a function of $\tilde{g}_{\epsilon_4}^{ts}(t)$, increases linearly.

IV. LOW-PASS FILTERING

In this section, in order to improve the performance of reservoir computing in the presence of noise, the noise-corrupted

versions of the input signal ($[s(t)]$) and of the output signal ($[g(t)]$) are driven through a low-pass filter (LPF). The LPF rejects high-frequency components while allowing the frequencies below the chosen cutoff frequency. The equation of a first-order LPF is

$$\frac{dV_{out}}{dt} = \frac{1}{\tau} [V_{in} - V_{out}], \text{ or equivalently,} \tag{12}$$

$$\dot{\hat{x}} = a [\tilde{x} - \hat{x}],$$

where $a = \frac{1}{\tau}$ is the cutoff frequency, $\tilde{x}(t) = V_{in}(t)$ is the LPF input, and $\hat{x}(t) = V_{out}(t)$ is the LPF output. We chose this type of filter because it is characterized by only a single parameter, and therefore reduces the complexity of the analysis and any potential physical implementations.

Adjusting the reservoir parameters such as α will alter the bandpass characteristics of the reservoir computer, but because α multiplies a nonlinear function on the right hand side of Eq. (1), adjusting α to alter the reservoir bandwidth will also change the reservoir nonlinearity so that it is no longer optimized for reproducing the Lorenz chaotic signal. Adding a separate linear low-pass filter allows us to alter the bandwidth of the reservoir computer without affecting its nonlinear characteristics.

This noise reduction method is particularly attractive because, in a field-deployed reservoir computer, low-pass filters are straightforward to implement, either with analog components or digital signal processing. In some types of photonic reservoir computers such as optoelectronic oscillators, a filter may even be directly integrated as part of the reservoir computer itself.^{27,28} Low-pass filters have previously been used on the individual node outputs to expand the reservoir,²⁹ but, to our knowledge, this is the first comprehensive investigation of the use of a low-pass filter to mitigate the effects of noise on reservoir computing performance.

If the input signals are low-pass filtered, Eq. (1) [Eq. (8)] are evolved with $\tilde{s}_{\epsilon_1}^{tr}(t)$ replaced by $\hat{s}_{\epsilon_1}^{tr}(t)$ [$\tilde{s}_{\epsilon_2}^{ts}(t)$ replaced by $\hat{s}_{\epsilon_2}^{ts}(t)$]. If the output signals are low-pass filtered, $\tilde{g}_{\epsilon_3}^{tr}(t)$ is replaced by $\hat{g}_{\epsilon_3}^{tr}(t)$ in Eqs. (4) and (7) [$\tilde{g}_{\epsilon_4}^{ts}(t)$ is replaced by $\hat{g}_{\epsilon_4}^{ts}(t)$ in Eq. (11)].

We first consider an ideal situation in which the underlying system is known *a priori*; then using statistical analysis, the optimal cutoff frequency \hat{a} for the LPF can be determined. The optimal cutoff

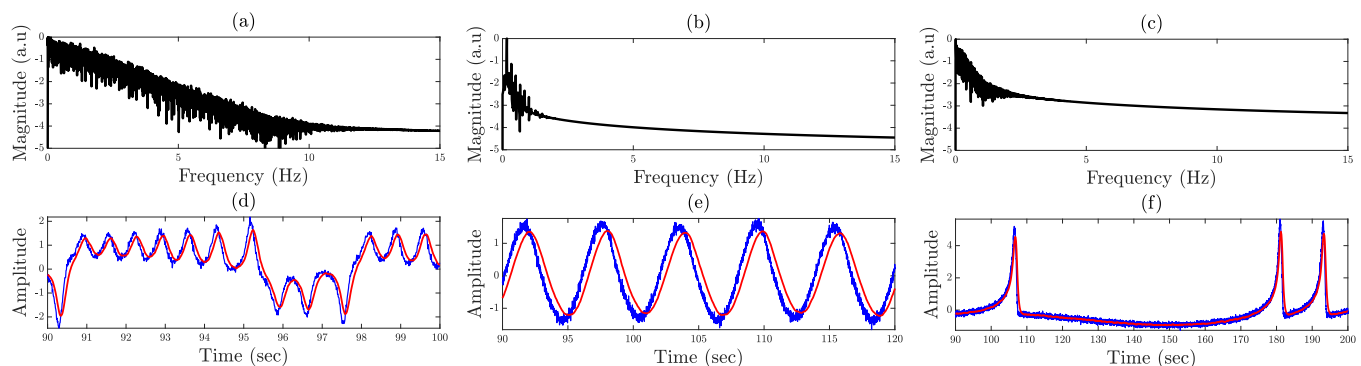


FIG. 4. The top panels show the spectrum of state variable $x(t)$ for the (a) Lorenz, (b) Rössler, and (c) Hindmarsh–Rose systems. The bottom panels show the time series plots of $\hat{x}(t)$ (blue color) and $\hat{x}(t)$ (red color) for the (a) Lorenz, (b) Rössler, and (c) Hindmarsh–Rose systems.

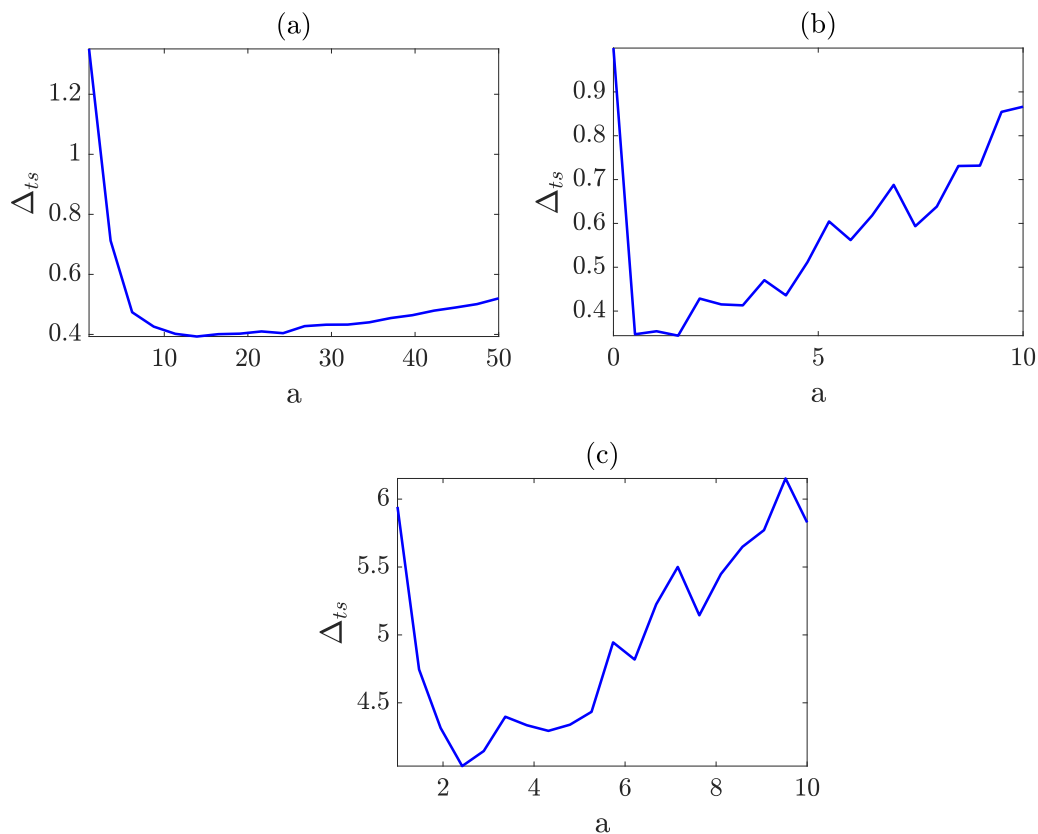


FIG. 5. Testing errors for the (a) Lorenz, (b) Rössler, and (c) HR systems, plotted as a function of the LPF cutoff frequency a .

frequency \hat{a} can be obtained from knowledge of the spectrum of the input signal, which is

$$X(f) = |\mathcal{F}[x(t)]|,$$

where \mathcal{F} is the Fourier transform operator.

Figure 4 shows the spectrum of the input signal for the cases of the Lorenz, Rössler, and Hindmarsh–Rose systems. The signals generated from chaotic systems have an invariant density function.

Consequently, the shape of the spectrum of $x(t)$ is not altered even by changing the system’s initial conditions. That is, for multiple chaotic signal realizations, their spectrum shape is invariant. The spectrum plot shown in Fig. 4 is generated from one such realization.

From Fig. 4, we observe that the maximum frequency component of $x(t)$ for the Lorenz system is at around 12 Hz. Similarly, the highest frequency for the Rössler system is at 2 Hz, and that of the Hindmarsh–Rose system is at 3 Hz. Therefore, the values of \hat{a} for

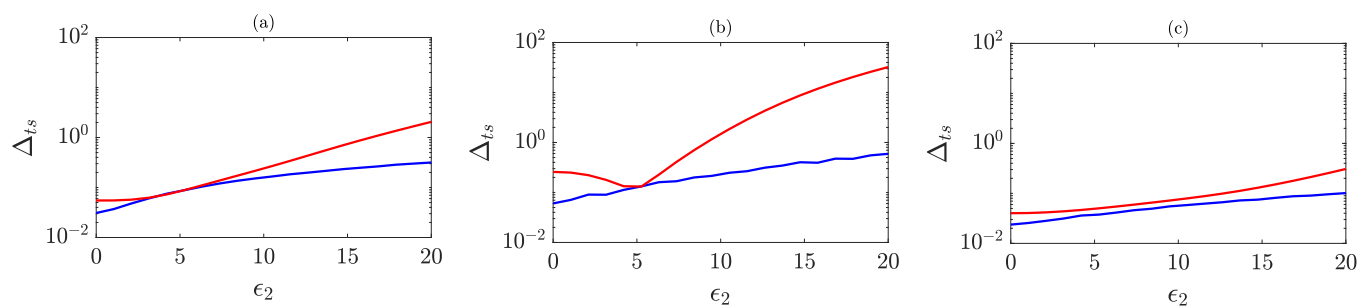


FIG. 6. Testing errors for the (a) Lorenz, (b) Rössler, and (c) Hindmarsh-Rose systems. The blue color plot is the error when the input signal is driven through the LPF and the red color plot is error when the input is not driven through the LPF.

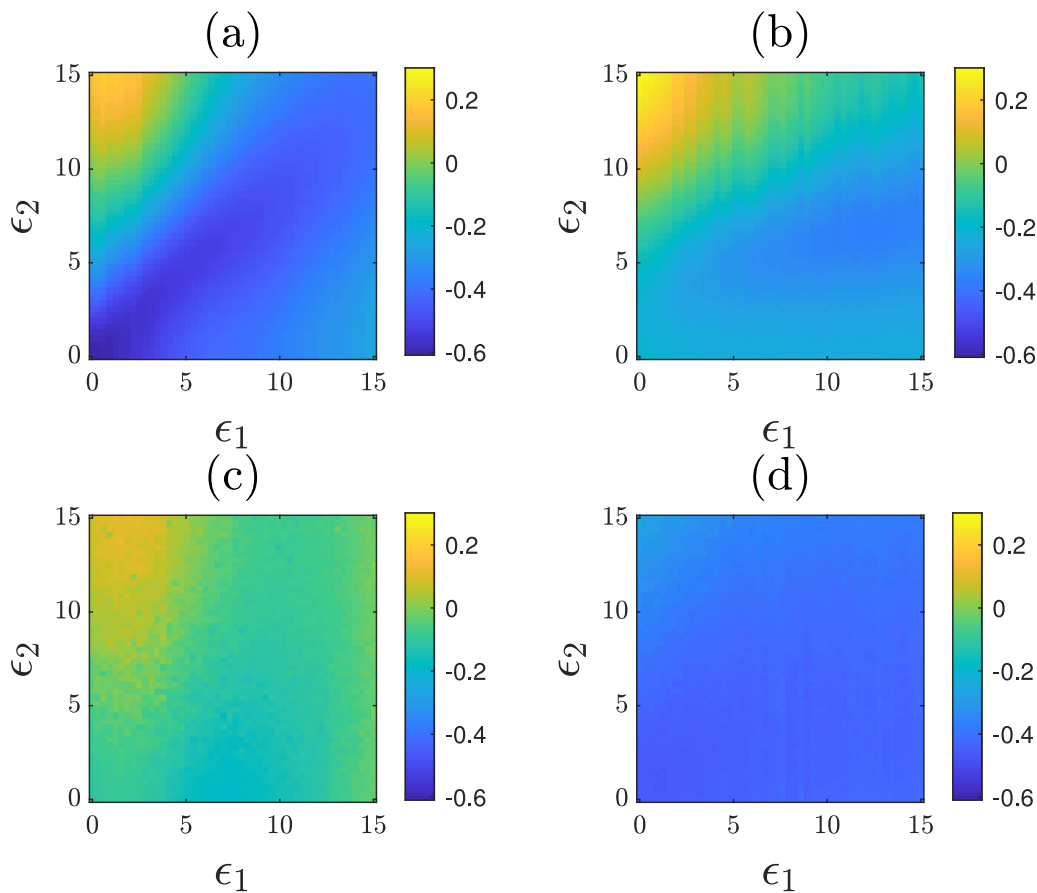


FIG. 7. Contour plots of the \log_{10} of the testing error for the Lorenz task and different cases: (a) the LPF is not used (b) only the input signal in the training phase $\tilde{s}^t(t)$ is driven through the LPF, (c) only the input signal in the testing phase $\tilde{s}^s(t)$ is driven through the LPF and (d) both signals $\tilde{s}^t(t)$ and $\tilde{s}^s(t)$ are driven through the LPF.

the Lorenz, Rössler, and Hindmarsh–Rose systems are 12, 2, and 3, respectively. If the cutoff frequency exceeds the optimal value, the filter allows more noise to go through. On the other hand, if it is less than the optimal value, it filters out the chaotic signal $x(t)$. Consequently, the minimum error is expected at \hat{a} , and the error increases when the choice of a is not optimal. The lower plots show the noise-corrupted version of the drive signal with $\epsilon_1 = 2$ and $\epsilon_2 = 5$ in blue and the filtered version of the input signal in red. It can be observed that \hat{x} (red) is delayed with respect to \tilde{x} (blue) while the noise is filtered.

In practice, one may only have access to the input signal corrupted with noise, and the underlying chaotic signal may not be known *a priori*. This prevents computation of the optimal cutoff frequency, hence it may become necessary to drive the input signal through the LPF while varying the cutoff frequency a . To assess the effects of the LPF, we consider the case where the noise strength of the training and testing signals are fixed, i.e., $\epsilon_1 = 5$, $\epsilon_2 = 20$. Figure 5 shows the testing error plotted against a . We see that the minimum testing error is obtained for the above-mentioned values of \hat{a} associated with each chaotic system. That is, for the Lorenz

system, the minimum Δ_{ts} is obtained when the cutoff frequency is around $a = 12$. Similarly, for the Rössler system and the Hindmarsh–Rose system, the minimum Δ_{ts} is obtained when $a = 2$ and $a = 3$, respectively.

Next, we investigate the effect of varying the noise strength. Namely, we fix the noise strength of the training phase $\epsilon_1 = 5$ and vary ϵ_2 . For each system, we set the cutoff frequency equal to \hat{a} . Figure 6 shows that, for all cases, the testing error is lower when the input signal is driven through the LPF, compared to the case when filtering is not applied. This is true also in the case that no noise is added to the input signal in the testing phase, i.e., $\epsilon_2 = 0$. The training error is independent of the LPF as it is independent of ϵ_2 , and we find that the training error for the filtered case is lower than for the unfiltered case.

We assessed the performance of reservoir computing by filtering the training and testing input signals individually. Figure 7 shows contour-level plots of the testing error for the Lorenz system task in various cases. In case (a), the error is computed without filtering the training or testing signals. Panel (a) of Fig. 7 is the same as panel (a) of Fig. 3, except we changed the color legend to be

consistent with the other plots in Fig. 7. A LPF with cutoff frequency $\hat{a} = 12$ is used for the rest of the cases. In case (b), the reservoir is trained with the filtered version of $\tilde{s}_{\epsilon_1}^{tr}(t)$, i.e., $\hat{s}_{\epsilon_1}^{tr}(t)$ [but the input signal in the testing phase is still $\tilde{s}_{\epsilon_2}^{ts}(t)$]. This indicates that noise is removed only from the input signal in the training phase (but not from the input signal in the testing phase). As a result, the performance of the reservoir is more robust to changes in ϵ_1 (especially for low values of ϵ_2) than to changes in ϵ_2 . For lower values of ϵ_2 , since the input signal in the training phase is filtered, the testing error is low. In case (c), the reservoir is trained with the noise-corrupted version of the input signal in the training phase, i.e., $\tilde{s}_{\epsilon_1}^{tr}(t)$, whereas, noise is removed only from the input signal in the testing phase, i.e., $\hat{s}_{\epsilon_2}^{ts}(t)$. We observed that when ϵ_1 is small, a small amount of overfitting is obtained. However, as ϵ_1 increases, the training signal becomes too noisy irrespective of the regression parameter and overfitting. Lastly, we consider the case where both $\tilde{s}_{\epsilon_1}^{tr}(t)$ and $\tilde{s}_{\epsilon_2}^{ts}(t)$ are filtered. Since the frequencies above \hat{a} are rejected, the impact of noise on the reservoir computer is effectively reduced. Consequently, we obtain a very low Δ_B over the entire ϵ_1, ϵ_2 plane, which is shown in case (d). We conclude that when no filtering operation is applied, the best performance is obtained for $\epsilon_1 = \epsilon_2$. However, for $\epsilon_1 \neq \epsilon_2$, the reservoir performance can be improved by filtering both the input signal in the training phase and in the testing phase with the same cutoff frequency \hat{a} .

We also considered the effects of picking different cutoff frequencies of LPF applied to the input signals in the training phase and in the testing phase. This is discussed in Sec. SII of the [supplementary material](#).

V. CONCLUSION

This paper presents a comprehensive investigation of the use of a low-pass filter to mitigate the effects of measurement noise on the performance of a reservoir observer. The effects of noise in both the training phase and in the testing phase were considered, and for both the cases that noise affects the input signals and the training and testing signals. Overall, low-pass filters are found to provide a good remedy against noise.

We first consider the case that filtering is not applied and find that the best performance is achieved when the noise strength affecting the input signal is about the same in the training and in the testing phases. The performance in the case that the amount of noise is the same is better than when the reservoir is trained with noise and tested in a noise-free environment. This motivates us to study possible remedies to implement when these amounts are not the same. We thus introduce low-pass filtering applied to the input signals in both the training phase and in the testing phase. We investigate the performance of the RC as the cutoff frequency of the low-pass filter is varied and find the optimal value of the cutoff frequency. We see a substantial improvement in the testing error, provided that the same type of filtering is applied in the training phase and in the testing phase.

One conclusion that we obtain is that it may be good to filter input and output signals, even when an estimate on the amount of noise that affects these signals is not available. In fact, low-pass filtering is typically not found to be detrimental, even when the signals are noise free. However, a more significant improvement in

performance is observed when a low-pass filter is applied to signals affected by increasing amount of noise. A question that is not addressed in this study is the role of process noise (as opposed to measurement noise) on the performance of a reservoir observer.

SUPPLEMENTARY MATERIAL

See the [supplementary material](#) for information about the different tasks that we assign to a reservoir computer in this paper and a study of the performance of a reservoir computer that uses different cutoff frequencies in the training phase and in the testing phase.

ACKNOWLEDGMENTS

This work was partly funded by the National Institutes of Health (NIH) under Grant No. 1R21EB028489-01A1 and by the Naval Research Lab's Basic Research Program.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Chad Nathe: Investigation (equal); Writing – original draft (equal). **Chandra Pappu:** Conceptualization (equal); Formal analysis (equal); Investigation (equal); Writing – original draft (equal). **Nicholas A. Mecholsky:** Conceptualization (equal); Formal analysis (equal); Investigation (equal); Methodology (equal); Visualization (equal). **Joe Hart:** Conceptualization (equal); Investigation (equal); Writing – original draft (equal). **Thomas Carroll:** Conceptualization (equal); Methodology (equal). **Francesco Sorrentino:** Conceptualization (equal); Investigation (equal); Methodology (equal); Writing – original draft (equal).

DATA AVAILABILITY

The data that support the findings of this study are available within the article.

REFERENCES

- ¹H. Jaeger, Bonn, Germany: German National Research Center for Information Technology GMD Technical Report (2001), Vol. 148, p. 13.
- ²W. Maass, T. Natschläger, and H. Markram, *Neur. Comput.* **14**, 2531 (2002).
- ³Z. Lu, J. Pathak, B. Hunt, M. Girvan, R. Brockett, and E. Ott, *Chaos* **27**, 041102 (2017).
- ⁴L. Appeltant, M. C. Soriano, G. V. der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. R. Mirasso, and I. Fischer, *Nat. Commun.* **2**, 468 (2011).
- ⁵L. Larger, M. C. Soriano, D. Brunner, L. Appeltant, J. M. Gutierrez, L. Pesquera, C. R. Mirasso, and I. Fischer, *Opt. Express* **20**, 3241 (2012).
- ⁶G. V. der Sande, D. Brunner, and M. C. Soriano, *Nanophotonics* **6**, 561 (2017).
- ⁷J. D. Hart, L. Larger, T. E. Murphy, and R. Roy, *Philos. Trans. R. Soc.* **377**, 20180123 (2019).
- ⁸Y. K. Chembo, D. Brunner, M. Jacquot, and L. Larger, *Rev. Mod. Phys.* **91**, 035006 (2019).
- ⁹A. Argyris, J. Bueno, and I. Fischer, *IEEE Access* **7**, 37017 (2019).
- ¹⁰F. Schurmann, K. Meier, and J. Schemmel, in *Advances in Neural Information Processing Systems* (MIT Press, 2004), Vol. 17, pp. 1201–1208.
- ¹¹G. Dion, S. Mejaouri, and J. Sylvestre, *J. Appl. Phys.* **124**, 152132 (2018).

- ¹²D. Canaday, A. Griffith, and D. J. Gauthier, *Chaos* **28**, 123119 (2018).
- ¹³G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose, *Neur. Netw.* **115**, 100 (2019).
- ¹⁴T. Jungling, T. Lymburn, and M. Small, *IEEE Trans. Neur. Netw. Learn. Syst.* **33**, 2586 (2022).
- ¹⁵B. Vettelschoss, A. Röhm, and M. C. Soriano, *IEEE Trans. Neur. Netw. Learn. Syst.* **33**, 2714 (2022).
- ¹⁶M. R. E. U. Shougat, X. Li, T. Mollik, and E. Perkins, *Sci. Rep.* **11**, 19465 (2021).
- ¹⁷T. L. Carroll, *Phys. Rev. E* **98**, 052209 (2018).
- ¹⁸T. L. Carroll, *Chaos, Solitons Fractals* **164**, 112688 (2022).
- ¹⁹Z. Liao, Z. Wang, H. Yamahara, and H. Tabata, *Chaos, Solitons Fractals* **153**, 111503 (2021).
- ²⁰J. Pathak, Z. Lu, B. R. Hunt, M. Girvan, and E. Ott, *Chaos* **27**, 121102 (2017).
- ²¹N. Semenova, X. Porte, L. Andreoli, M. Jacquot, L. Larger, and D. Brunner, *Chaos* **29**, 103128 (2019).
- ²²I. Estébanez, I. Fischer, and M. C. Soriano, *Phys. Rev. Appl.* **12**, 034058 (2019).
- ²³A. Röhm, D. J. Gauthier, and I. Fischer, *Chaos* **31**, 103127 (2021).
- ²⁴L.-W. Kong, H.-W. Fan, C. Grebogi, and Y.-C. Lai, *Phys. Rev. Res.* **3**, 013090 (2021).
- ²⁵G. Donati, A. Argyris, C. R. Mirasso, M. Mancinelli, and L. Pavesi, in *Integrated Optics: Devices, Materials, and Technologies XXVI* (SPIE, 2022), Vol. 12004, pp. 219–226.
- ²⁶F. Sorrentino and E. Ott, *Chaos* **19**, 033108 (2009).
- ²⁷H. Dai and Y. K. Chembo, *IEEE J. Quantum Electron.* **57**, 1 (2021).
- ²⁸H. Dai and Y. K. Chembo, *J. Lightwave Technol.* **40**(21), 7060–7071 (2022).
- ²⁹T. L. Carroll, *Phys. D* **416**, 132798 (2021).