Chaos
An Interdisciplinary Journal of Nonlinear Science

RESEARCH ARTICLE | APRIL 05 2024

The influence of hyperchaoticity, synchronization, and Shannon entropy on the performance of a physical reservoir computer

Lucas A. S. Rosa ; Eduardo L. Brugnago ; Guilherme J. Delben ; Jan-Michael Rost ; Marcus W. Beims

Chaos 34, 043120 (2024)
https://doi.org/10.1063/5.0175001

The influence of hyperchaoticity, synchronization, and Shannon entropy on the performance of a physical reservoir computer

Cite as: Chaos 34, 043120 (2024); doi: 10.1063/5.0175001
Submitted: 4 September 2023 · Accepted: 21 March 2024 ·
Published Online: 5 April 2024

Lucas A. S. Rosa,[1,a] Eduardo L. Brugnago,[2,b] Guilherme J. Delben,[3,c] Jan-Michael Rost,[4,d] and Marcus W. Beims[1,4,e]

AFFILIATIONS

[1]Departamento de Física, Universidade Federal do Paraná, 81531-980 Curitiba, Paraná, Brazil
[2]Instituto de Fí sica, Universidade de São Paulo, 05508-090 São Paulo, SP, Brazil
[3]Departamento de Ciências Naturais e Sociais, Universidade Federal de Santa Catarina, 89520-000 Curitibanos, SC, Brazil
[4]Max-Planck Institute for the Physics of Complex Systems, Nöthnitzerstr.38, 01187 Dresden, Germany

[a)]lucasasouzar@gmail.com
[b)]elb@if.usp.br
[c)]guilherme.delben@ufsc.br
[d)]rost@pks.mpg.de
[e)]**Author to whom correspondence should be addressed:** mbeims@fisica.ufpr

ABSTRACT

In this paper, we analyze the dynamic effect of a reservoir computer (RC) on its performance. Modified Kuramoto's coupled oscillators are used to model the RC, and synchronization, Lyapunov spectrum (and dimension), Shannon entropy, and the upper bound of the Kolmogorov–Sinai entropy are employed to characterize the dynamics of the RC. The performance of the RC is analyzed by reproducing the *distribution* of random, Gaussian, and quantum jumps series (shelved states) since a replica of the time evolution of a completely random series is not possible to generate. We demonstrate that hyperchaotic motion, moderate Shannon entropy, and a higher degree of synchronization of Kuramoto's oscillators lead to the best performance of the RC. Therefore, an appropriate balance of irregularity and order in the oscillator's dynamics leads to better performances.

This work uses modified coupled Kuramoto oscillators to model a physical reservoir computer (PRC). Our main goal is to analyze the characteristics underlying the nonlinear dynamical properties of the PRC when it is capable of learning a specific task. In other words, what happens inside the PRC when its learning performance is acceptable and trustworthy? The nonlinear dynamic of the PRC is examined by the degree of synchronization of the oscillators, hyperchaoticity, Lyapunov spectra, and dimension. We show that better performance of the PRC to reproduce distributions of classical and quantum time series is obtained for the hyperchaotic cases with certain degrees of synchronization. Furthermore, the performances are shown to be proportional to the Shannon entropy of the distributions. Consequently, an order-hyperchaos mixing in the oscillator's dynamics provides improved performances.

I. INTRODUCTION

Over the last few decades, artificial intelligence (AI) has found widespread application in scientific and societal activities. Moreover, technological progress, as well as the growing number of applications, has been contributing significantly to the promotion of several sub-areas within AI, of which stand out computer vision,[1] natural language processing,[2] and the one we are going to approach in the present paper, machine learning (ML).[3]

In ML, several techniques can be used to solve tasks with applications in Physics,[4–11] Chemistry,[12–14] Biology,[15,16] Medicine,[17,18] and Economics,[19–21] among others. While the goal when performing such tasks is often to obtain optimal performance, i.e., a balance between the quality of the result and the incurring costs, in the present work, we focus on the underlying dynamics adopted by the machine when performing some tasks. Among the most broadly

used and efficient information processing techniques within ML, artificial neural networks (ANNs) stand out. They consist of a set of information processing units, i.e., mathematical models inspired by biological neurons. As the real ones, the artificial neurons communicate through synaptic connections. They receive stimuli through their input nodes and, in response, produce one or more outputs, whereas activation functions are responsible for this conversion according to the specific network architecture.[22,23] There are two main ways to connect those neurons: feedforward neural networks (FNNs) and recurrent neural Networks (RNNs). An FNN depends necessarily on the presence of an input to make a decision and cannot maintain self-sustaining dynamics. Since in this study we want to build a network with its own dynamics, independent of some input, only an RNN presents the desired characteristics for this study.

The RNN's main characteristic is the existence of cycles in its topology, leading to temporal dynamics of the network, self-sustained in its connections, even without the presence of an external input signal. This renders an RNN a dynamical system. If there is an input signal, an RNN preserves a non-linear transformation of its history in the internal states, thus being able to process information with temporal dependence.[24,25]

A clear disadvantage of RNNs is the high computational cost in the training stage due to their cyclic connections. A distinct approach, called reservoir computing (RC),[26] aims at overcoming this deficiency. Its two main variants, echo state networks (ESN)[27] and liquid state machines (LSMs),[28] have similar structures as conventional RNNs, but with the difference and advantage of having trained only the connections between the output and the neural units. Keeping the internal connections between the neurons and with the input signal unchanged facilitates not only the training stage but also allows one to use real physical systems as reservoirs, dubbed physical reservoir computing (PRC). Physical systems of diverse nature have been efficient as PRCs, such as mechanical,[29,30] biological[31,32] and quantum systems[33,34] as well as analog circuits.[35,36] One can obtain the neural states directly from the physical systems within an experimental approach or through computational simulations.

In this paper, the reservoir comprises Kuramoto oscillators,[37] which are widely used to investigate the synchronization effects in real biological networks.[38] The simplicity of Kuramotos's oscillator model supports a deeper understanding of the reservoir dynamics regarding its computational AI performance, our main task. To select the parameter values in the model, we have used the metaheuristic technique named genetic algorithm (GA).[39,40] This method is usually preferred when there are a lot of parameters involved and/or computational limitations in performing particular tasks. The model and the selection of parameters are described in Sec. II. Specifically, we investigate the relationship between the dynamics of the Kuramoto oscillator reservoir and its ability to learn the distribution of randomly generated binary series and series with a Gaussian distribution, as described in Sec. III. With *learning*, we refer to the ability of the PRC to generate an output with the same distribution as the reference series without necessarily making exact predictions at each step. As it will turn out, for the performance of the PRC, an important element is the information gain measured by the Shannon entropy.[41] Further relevant elements are contributed through the dynamical point of view of the reservoir, namely, the *synchronization* between Kuramoto oscillators, quantified by the order parameter,[42–44] as well as the *hyperchaoticity*, using the spectrum of Lyapunov exponents (and its dimension) and the upper bound for the Kolmogorov–Sinai entropy,[45,46] namely, the sum of all positive Lyapunov exponents. We emphasize that we do not intend to present a more efficient method for generating pseudorandom sequences, and we are interested in studying the dynamics inside the PCR.

As an application, we analyze a series based on the result of a quantum jump experiment involving shelved states[47] in Sec. IV. Quantum jumps have been studied for decades because their dynamics enable a better understanding of fundamental quantum theory related to the question of quantum measurement and transition between states. In these studies, in general, shelved quantum systems are used to describe the system, as it represents the quantum jump to a metastable state. In other words, the shelved level is a metastable state that can retain its energy for a prolonged period.[48–50] Such a system has attracted significant attention in quantum information processing due to its potential applications in quantum computing and quantum communication. The existence of the shelved level enables long coherence times, which is essential for the implementation of quantum algorithms. Additionally, this system can be used as a quantum memory, a superconducting qubit or a trapped-ion qubit, providing a platform for studying quantum coherence and entanglement.[51–54] In Sec. V, we summarize our findings.

II. KURAMOTO'S RESERVOIR COMPUTER AND METHODOLOGY

We start by describing the nonlinear dynamical system used to model the reservoir computer, followed by a detailed description of the learning processes used in this work.

A. The modified Kuramoto oscillators reservoir computer

The original Kuramoto system is composed of N all-to-all coupled oscillators.[37] In the present work, we used the K-nearest-neighbors coupling for which the oscillator states are described as

$$\dot{\theta}_i = \omega_i + \frac{\kappa}{2K}\sum_{j=1}^{N} A_{ij}\sin(\theta_j - \theta_i),\ i = 1, 2, \ldots, N, \tag{1}$$

where θ_i is the phase and ω_i is the natural frequency of oscillation of the ith oscillator, κ is the coupling strength, j is the index that indicates which oscillator will couple, and A represents the adjacency matrix, which holds the information about the connections. In our study, $A_{ij} = 1$ if $|i - j| \leq K$, and 0 otherwise.

To increase diversity in the output values, we added to Eq. (1) a periodic term depending on the difference between the phase of the

specific oscillator and a mean-field element according to

$$\dot{\theta}_i = \omega_i + \frac{\kappa}{2K}\sum_{j=1}^{N} A_{ij}\sin(\theta_j - \theta_i) + \beta\cos\left[\zeta\left(\theta_i - \sum_{j=1}^{N}\frac{\theta_j}{N}\right) - \gamma\right], \quad i = 1, 2, \ldots, N, \tag{2}$$

where β, ζ, and γ are parameters of the system. While β quantifies the intensity of the mean-field, ζ provides the oscillation frequency of each phase deviation around the mean phase, and γ is a phase shift. The above added term aims to perturb the network, causing oscillators whose phases are close to the mean field to suffer a larger perturbation than those that are in complete anti-phase. The purpose is to avoid total synchronization of the network, ensuring the existence of oscillators in different phases.[55] Inside the black box in Fig. 1, we show a sketch representing the connections of the neural units to each other and with the output signal. Equation (2) are the modified Kuramoto's oscillators used in the present work.

B. About the learning

In this section, we present some technical and relevant details about the numerical simulations, reference series, and the learning procedure. In general, to select a set of parameters that lead the system to a good performance, we use the well-known GA. It is a technique inspired by evolutionary biology and presents some characteristics, such as hereditary, mutation, natural selection, and recombination.[39,40,56] The number of oscillators and neighbors were kept fixed at $N = 250$ and $K = 8$, respectively. Empirically, we tested different values for N and K and observed that the above numbers are the smallest necessary to mimic the proposed random distributions. Details about the GA are presented in Appendix A, and a flow chart of the learning procedure is provided in Appendix B.

1. Numerical integration

For the numerical integration of Eq. (2), we used the fourth-order Runge–Kutta integrator (RK4) with a stepsize of 10^{-2} and discarded a transient of 10^4 steps.[57] The phases θ_i of the oscillators, which represent the neural states, are recorded at every eighth step, rendering the changes larger and, therefore, making the learning process more effective. A few checks were done using steps larger than 8, but no significant improvement in efficiency was observed. The output y is the result of an activation function φ applied to a weighted sum of these phases and the bias (see the magnification of Fig. 1). Throughout this paper, we refer iterations to each time we record the phases and not to the step's integrator. For the natural frequencies ω_i, we adopt the Lorentzian distribution, as typically done in dealing with Kuramoto oscillators. In contrast to usual studies with neural networks, there are no activation functions for each reservoir unit since the phases of the oscillators representing the system states are already restricted to values between 0 and 2π.

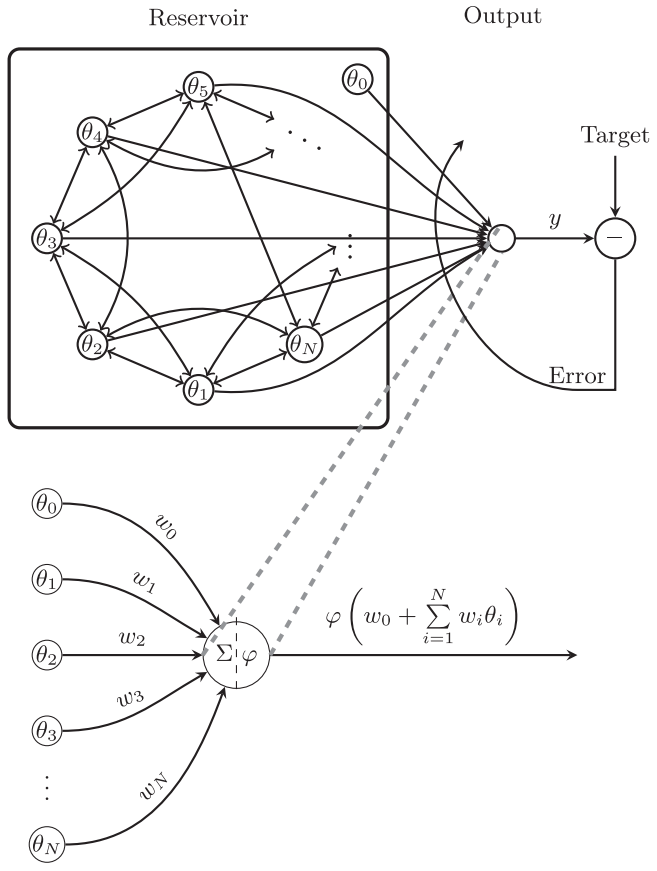

FIG. 1. Graph of the signal flow between the nodes (units) of the RNN within the reservoir. The link between the N network units in the reservoir is given by K-nearest-neighbours, with $K = 2$. The magnification highlights that the output y results from an activation function φ applied to a weighted sum of the states θ_i of the reservoir units and the bias. The connection weights between the reservoir units and the output are represented by w_i. The bias is always $\theta_0 = 1$.

2. Treating the reference series

For the output signal, we have chosen the Heaviside function $\varphi(\cdot)$ as activation,

$$y(\tau) = \varphi(W_{\text{out}}\vec{\theta}(\tau)), \tag{3}$$

with

$$\varphi(W_{\text{out}}\vec{\theta}(\tau)) = \begin{cases} 1, & \text{if } W_{out}\vec{\theta}(\tau) \geq 0 \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

where W_{out} is the matrix containing the weights connecting the reservoir units and the output. Although the system dynamics evolve continuously, we address time as a discrete variable $\tau = 1, 2, \ldots$, corresponding to the iterations defined above.

When the reference series has more than two values, we break the interval into equally spaced sub-intervals or *segments*. An integer labels the segments, and after representing them in binary numbers, we do the learning bit by bit. To update the weights

$w_0, w_1, w_2, \ldots, w_N$, components of the matrix W_{out}, we use the gradient descendent method. Thus, at each iteration, we compute the cost function

$$J(\tau, w_0, w_1, \ldots, w_N) = \frac{1}{2}\left[d(\tau) - y(\tau)\right]^2, \tag{5}$$

where $d(\tau)$ is the target obtained from the reference series and $y(\tau)$ is the output generated by the model. The weights were updated by summing the cost function's partial derivative with respect to themselves,

$$w_i(\tau + 1) = w_i(\tau) + \alpha\left[d(\tau) - y(\tau)\right]\theta_i(\tau), \tag{6}$$

where

$$\frac{\partial}{\partial w_i} J(\tau, W_{\text{out}}) = \left[d(\tau) - y(\tau)\right]\theta_i(\tau), \tag{7}$$

and α is the convergence constant of the gradient, which we have kept fixed at $\alpha = 1$ for all simulations.

3. The considered reference series

We tested the performance of our model and its relation to the underlying dynamics of Kuramoto's oscillators using three different series. The first one is a binary **Random series (A)**. This function provides a sequence of pseudorandom numbers within the interval $[0, 1[$. We break the interval into two segments and assign 1 to values greater or equal to a number we chose between 0 and 1, and 0 otherwise. We note that even though the random series appears to be too simple, a complete aleatory series, in general, is not possible to replicate with ML techniques. Second, we generate a **Gaussian series (B)**. To this end, we use the above function again but feed it to the Box–Muller transform to generate three series with approximate Gaussian distributions with different widths. However, this time, instead of dividing into two intervals, we broke their range of values into 16 equally spaced segments. Finally, for the **Quantum jumps (C)**, we generate a numerical series that simulates experimental light bursts generated by quantum jumps.

From the reference series, we take $T_{\text{training}} = 200$ samples for the random and Gaussian series and $T_{\text{training}} = 309$ samples for the quantum jumps series. The T_{training} values are used to update the weights ω_i between the reservoir nodes. If, after the T_{training} samples, the result is not satisfactory, we repeat the training process for the T_{training} samples, updating again the weights ω_i. Every time we repeat this procedure with the same sample set, we name it an *epoch* ($\mathfrak{E}$). The maximally allowed epochs are $\mathfrak{E}_{\text{max}} = 200$. After the learning stage, we use the data *validation* set of V elements to verify if there is an agreement between the extrapolation of our program and the reference series. All values analyzed after the validation belong to the *test interval* T_{test}.

Larger and smaller values than $T_{\text{training}} = 200$ were tested. For smaller values, the amount of data from the time series is not enough for the learning procedure. For larger values, the increasing computer time does not compensate the small gain in the learning process.

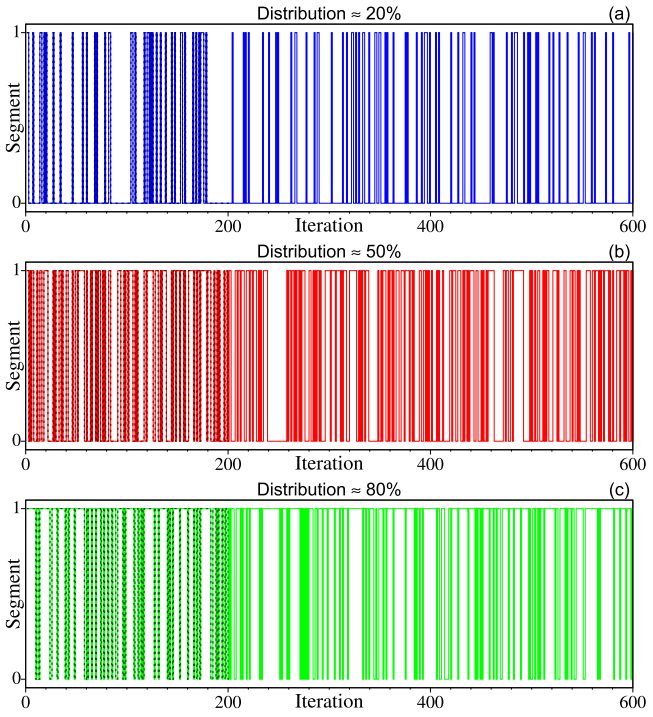

FIG. 2. Time series with approximated distributions of 20% (blue) in panel (a), 50% (red) in panel (b), and 80% (green) in panel (c). Dashed lines represent the reference series, and filled lines represent the PRC's output.

4. Evaluation method and errors

Conventionally, in studies of this field, it is expected that the machine learns the reference series in a way that is capable of faithfully reproducing it at each step. However, because we are dealing with randomly generated series whose signals are irreproducible step by step, we propose an evaluation method based on the program's ability to reproduce more general aspects of the reference series, namely, their distribution.

Our method evaluates three aspects: (i) the first one deals with the sufficient number of epochs to emulate the series in the same way as previous methods; (ii) the second one measures the model's ability to maintain the number of values within each segment of the reference series, in the training segment; (iii) the third one is similar to the previous aspect, but it looks at the sequences of two consecutive values (this prevents, for example, that the model presents a binary series with half of the consecutive points equal to 0, and the other half consecutive points equal to 1, while the reference series contains the same values, but arranged alternately, and obtain a good performance). We have done a weighted and normalized sum over the errors corresponding to the three aspects, assigning 10% for (i), 60% for (ii), and 30% for (iii).

To quantify the agreement between extrapolations and reference series, our *performance*, we calculate the error ($\mathbb{E}$) as

follows:

$$\mathbb{E} = \frac{\sqrt{\mathbb{E}_{\text{training}} + \mathbb{E}_1 + \mathbb{E}_2}}{\sqrt{W^2_{\text{training}} + W^2_{\text{predic1}} + W^2_{\text{predic2}}}}, \tag{8}$$

where

$$\begin{aligned} \mathbb{E}_{\text{training}} &= \left(W_{\text{training}} \cdot \frac{\mathfrak{E}_{\text{training}}}{\mathfrak{E}_{\max}} \right)^2, \\ \mathbb{E}_1 &= \left(W_{\text{predic1}} \cdot \mathbb{E}_{\text{predic1}} \right)^2, \\ \mathbb{E}_2 &= \left(W_{\text{predic2}} \cdot \mathbb{E}_{\text{predic2}} \right)^2, \end{aligned} \tag{9}$$

and $W_{\text{training}} = 0.1$, $W_{\text{predic1}} = 0.6$, and $W_{\text{predic2}} = 0.3$ are the weights for the first, second, and third terms in the weighted sum. The number of epochs the PRC uses for complete learning of the training interval from the reference series is given by $\mathfrak{E}_{\text{training}}$. To define $\mathbb{E}_{\text{predic1}}$, we count the number of elements in each class, both in the reference series (range from 0 to 200) and in the output of the PRC (range from 201 to 400), take the difference between these counts, and divide it by the number of elements compared. Similarly, we define the term $\mathbb{E}_{\text{predic2}}$, but instead of counting the number of elements in each class, we count the sequences of two consecutive values. The numbers 1 and 2 at the end of $\mathbb{E}_{\text{predic1}}$ and $\mathbb{E}_{\text{predic2}}$ refer to one value and two values, respectively.

C. Physical properties characterizing the dynamics of Kuramoto's oscillators

The synchronization of the oscillators, quantified by the order parameter r,[42–44] the Lyapunov spectrum $\{\lambda_i\}$,[58,59] and the associated upper bound of the *Kolmogorov–Sinai (KS) entropy* $h^{\text{u}}_{\text{KS}} = \sum_{\lambda_i > 0} \lambda_i$ (the sum of positive Lyapunov exponents λ_i[60]) are typical properties to characterize quantitatively a complex system, such as the Kuramoto model (2). We will determine the combination of their values that optimizes the performance of the Kuramoto oscillators as a PRC by varying the parameters β, γ, ζ, and κ in (2). When the sum of all Lyapunov exponents $J = \sum_{\lambda_i} \lambda_i$ is zero, the reservoir dynamics is conservative. For $J < 0$, it is dissipative. The Lyapunov spectrum is obtained using the standard Gram–Schmidt orthogonalization procedure.[60] The quantity h^{u}_{KS} is, therefore, obtained directly from the Lyapunov exponents, obtained from the tangent space associated with the equations of motion. No numerical estimator[61] was used to determine h^{u}_{KS}. Furthermore, it is possible to determine the *Lyapunov dimension*,[62,63]

$$D = k + \frac{\sum_{i=1}^{k} \lambda_i}{|\lambda_{k+1}|}, \quad \text{with} \quad k = \max\left\{ \sum_{i=1}^{n} \lambda_i \geq 0 \right\}. \tag{10}$$

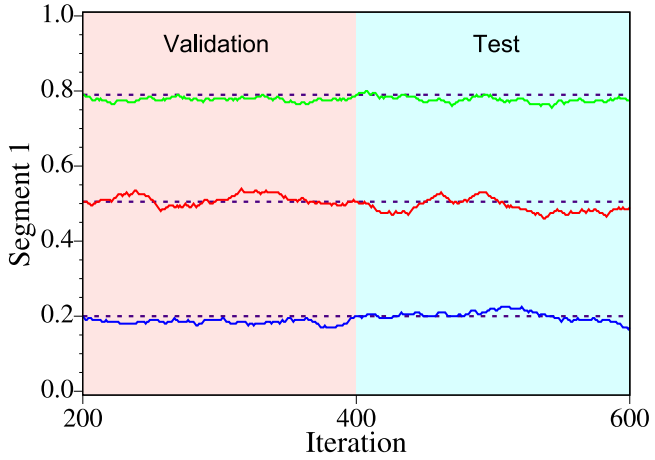

FIG. 3. Distributions along the time series. Purple dashed lines correspond to the distribution of the reference series (number of times the value 1 appears) in the training interval (first 200 samples). Solid lines are related to the distribution of the PRC's output considering only a block of 200 previous samples to each iteration. The colors blue, red, and green are related to distributions 20%, 50%, and 80%, respectively.

It has been argued[63] that D is the dimension of the highest dimensional infinitesimal hypersphere, which is deformed in time with the volume preserved on average.

To observe the network synchronization, we followed the temporal evolution of the oscillators, at every iteration, during 100 iterations. Then, the *order parameter* is obtained from

$$re^{i\Phi} = \frac{1}{N} \sum_{m=1}^{N} e^{i\theta_m}, \tag{11}$$

where Φ is the average phase, θ_m is the phase of the mth oscillator, and r, known as the order parameter, represents the phase-coherence of the population of oscillators. When the oscillators are fully coupled, the value of r tends to 1, and 0 when the distribution of the phases is uniform.

III. RESULTS FOR RANDOM AND GAUSSIAN SERIES

In this section, we apply the above-described methodology for specific times series, namely, the random and Gaussian series, considered from now on as the reference series. Since this involves two strands of optimization (finding the optimal reservoir in terms of

TABLE I. Parameters selected by the GA, the resulting errors ($\mathbb{E}$), the order parameter (r), the upper bound of the KS-entropy (h^{u}_{KS}), the dissipation ($-J$), the approximated partial Shannon-entropy (S_{p_i}) obtained directly from the random series, and the Lyapunov dimension (D). Related to Figs. 2–4.

Dist	κ	β	γ	ζ	$\mathbb{E}$	r	h^{u}_{KS}	$-J$	S_{p_i}	D
20%	0.8103	1.2552	4.7674	0.3588	0.008 408	0.81	16.1	75.36	0.464	113.7
50%	0.3866	1.7340	5.0040	0.4485	0.015 399	0.84	24.4	9.800	0.502	225.1
80%	1.4674	1.5198	4.3417	0.6502	0.006 997	0.85	11.8	329.6	0.269	56.34

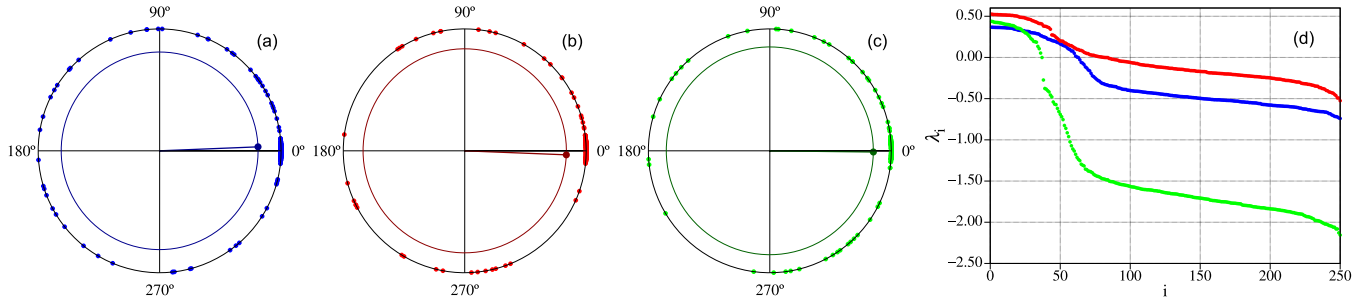

FIG. 4. The points on the outer circles of panels (a)–(c) are a representation of the phases of the oscillators at a given iteration. The point in the inner circle represents the average of the phases. The outer circle is unitary, while the inner circle has a radius equal to the order parameter r. Panels (a)–(c) are related to distributions of 20%, 50%, and 80%, respectively. Panel (d) shows the Lyapunov spectra of the Kuramoto system, in descending order, with the parameters (see Table I) selected for the reference series with distributions of 20% (blue points), 50% (red points), and 80% (green points).

the parameters $\beta, \gamma, \kappa, \zeta$ by a genetic algorithm and training the network), we provide a schematic summary of the procedure in Fig. 15 in Appendix B.

A. Random series

Initially, we generate binary series for numbers between 0 and 1. These numbers are divided into two segments: 1 if the number given by the function is greater or equal to a threshold and 0 otherwise. According to the chosen threshold, we determine the series distribution. For example, if we choose the threshold 0.2, approximately 80% of the series is 1. If we choose 0.4, approximately 60% is 1 and so on.

Figure 2 presents the time series whose thresholds are 0.8, 0.5, and 0.2, from the top to the bottom, respectively. According to the previous explanation, it corresponds to the approximated distributions of 20%, 50%, and 80% of values 1. In dashed lines (first 200 points), we have the reference series, and in filled lines, the output is generated by our model. Since this is a binary series, the segments were represented only by 0 and 1. As we can see, in Fig. 2(a), the series remains longer at 0 than at 1 since its distribution (of values equal to 1) is expected to be somewhere around 20%. The opposite behavior occurs in Fig. 2(c), while in Fig. 2(b), there is a balance between numbers 0 and 1. For each of the three cases, we apply the GA procedure to select a set of parameters that lead the system to a good performance. Table I shows the selected parameter sets for each reference series, along with the measured errors, described in Subsection. II B.

The errors are generally very small, with the most significant error occurring for the case 50%. The errors in the cases 20% and 80% become consecutively smaller compared to the 50% case. Such errors can be explained using the Shannon-entropy,[41] $S(X) = \sum_i^N S_{p_i} = -\sum_i^N p_i(x) \log_2 p_i(x)$, which quantifies the average gain of information, surprise, or uncertainty of the possible outcomes of the random variable X. p_i is the probability of obtaining the specific outcome x_i and $S_{p_i} = -p_i(x_i) \log_2 p_i(x_i)$ we define as the *partial* Shannon-entropy, which is the information gain of the specific outcome x_i. Returning to our random series, if the probability of obtaining the number 1 is $p = 1/5 = 0.2$ (20%), then the partial Shannon-entropy is $S_{1/5} = -0.2 \log_2(0.2) = 0.464$. For the other cases, we obtain $S_{1/2} = 0.5$ for 50%, and $S_{4/5} = 0.257$ for 80%. Compared to the performance (see Table I), there is an explicit relation between the errors made by the RC and the partial Shannon-entropy. Table I also presents the approximated partial Shannon-entropy (S_{p_i}) obtained directly from the random series and is in good agreement with the above values of $S_{1/5}, S_{1/2}$, and $S_{4/5}$. We mention that the usual Shannon-entropy is $S = -0.2 \log_2(0.2) - 0.8 \log_2(0.8) = 0.723$ for 20% *and* 80% and $S = 1.0$ for 50%, which are not directly related to the performance of the RC.

Figure 3 shows the model's ability to maintain the distribution of the training segment of the reference series. The blue, red, and green lines are related to the output and correspond, respectively, to the approximate distributions (how often the value 1 is observed) of 20%, 50%, and 80%, considering a block of 200 previous samples to each iteration. For example, iteration 210 shows the proportion of elements in segment 1 compared to the total number of elements starting from iteration 11. For reference, the dashed lines, in purple, correspond to the distribution that the reference series presents in its first 200 samples. The background colors highlight the intervals: salmon, referring to the validation interval, and cyan, referring to the test interval, that is, the interval where there is neither network training nor validation of the choice of parameters. The latter is the RC forecasting of the distribution. We observed that, in all cases, our model efficiently maintained the distribution of the reference series.

Regarding synchronization of Kuramoto's oscillators, we analyzed the space–time evolution of the oscillator. We observe a large interchange between phases oscillating mainly close and around 0° and 360°. A significantly smaller amount of oscillators have 180°. In all three cases, we notice a similar behavior. The oscillators that have ω_i close to 0 keep their phases close to 0°, apparently being more coupled than those with larger ω_i, in absolute value, which presents more heterogeneous dynamics.

To quantify the synchronization between the oscillators, we reckon the order parameter r using the parameters selected for each distribution. In Figs. 4(a)–4(c), we observe the phases of the oscillators at a given iteration, represented by points on the outer unit circle, while the average of all phases is shown by the point in the inner circle whose radius is the order parameter r. As we have

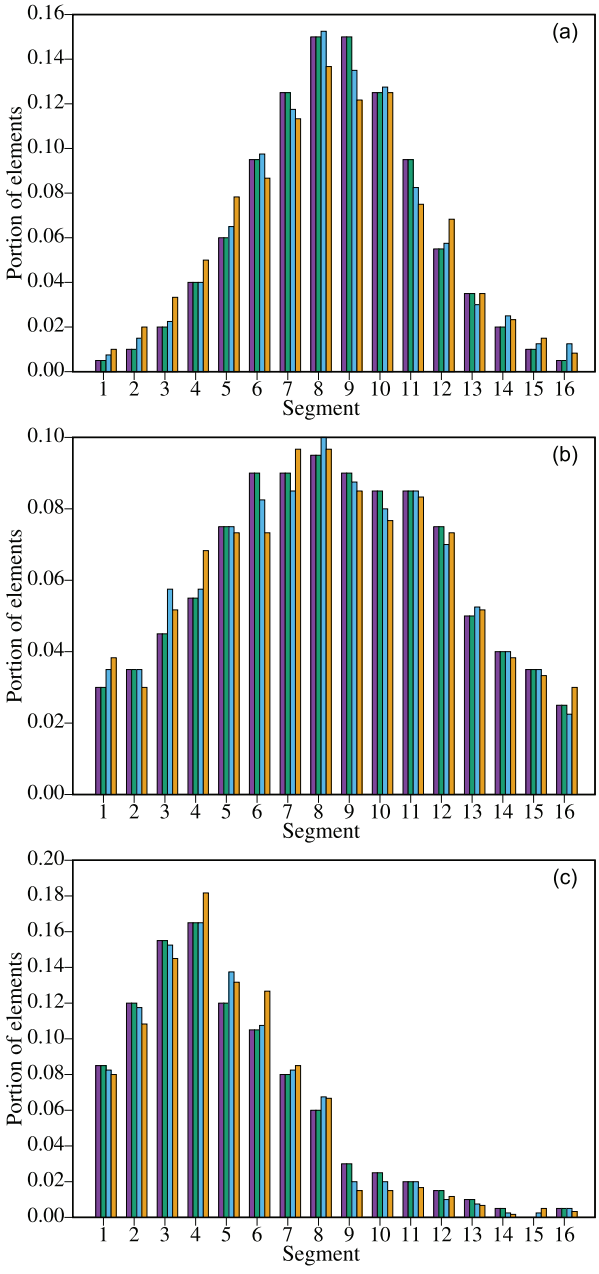

FIG. 5. Histograms with the number of elements in each segment of the reference series (purple) and of the PRC's outputs in the training (green), validation (blue), and test (orange) intervals. The three rows of Table II are related, respectively, to the panels (a)–(c).

seen, most oscillators are close to 0° and 360°. The panels 4(a)–4(c), respectively, are associated with the parameters selected by the GA for the three distributions (20%, 50%, and 80%). The order parameters for the three cases are $r = 0.81$ in panel 4(a), $r = 0.84$ in panel 4(b), and $r = 0.85$ in panel 4(c), summarized in Table I. It demonstrates that although the oscillators are not fully coupled ($r = 1.00$), the network presents a high degree of synchronization.

The last panel on the right of Fig. 4 presents the spectra of Lyapunov exponents (λ_i) for the three rows of Table I, respectively, related to the blue, red, and green points. In all three cases, we have more than one positive Lyapunov exponent, characterizing the dynamics of the network of oscillators as hyperchaotic. Table I shows the KS-entropy upper bound, according to the definition provided in Sec. II C. The sum of all Lyapunov exponents J is negative, demonstrating that the PRC dynamics is dissipative. Only a few Lyapunov exponents are close to zero, while most are negative. We notice that the prediction errors are larger (smaller) when more (fewer) oscillators have positive Lyapunov exponents. Thus, in this case, the performance of the PRC is proportional to the upper bound of the KS-entropy, the Shannon-entropy, and the Lyapunov dimension, as can be checked in Table I.

B. Gaussian series

Here, we use the random function again but manipulate it inside the Box–Muller function[64] to generate three series with approximately Gaussian distributions and distinct shapes from each other. After generating the series, we divided them into 16 segments. As previously mentioned, we represented the output in binary numbers and performed the learning bit by bit as if they were b independent outputs for each series value, where b is the minimum number of bits sufficient to represent the number of segments. In the case of 16 segments, we have $b = 4$.

In Fig. 5, we present in histograms the normalized proportion of elements in each segment of the reference series (purple columns) and PRC's outputs until the iterations 200, 400, and 600, represented by the green, blue, and orange columns, respectively. Indeed, we notice that all columns approximate Gaussian distributions with different shapes. Table II shows the sets of parameters selected by GA and also contains the errors evaluated by the model's ability to maintain the distribution of the reference series, as described in Subsection. III A.

Simulations demonstrate that the network synchronization is similar to the case described in Subsection. III A, where the performance was measured similarly. These similarities are demonstrated quantitatively with the order parameters shown in Fig. 6(a) for $r = 0.80$, Fig. 6(b) for $r = 0.80$, and Fig. 6(c) for $r = 0.75$, which are close to those presented in Subsection. III A. In Fig. 6, r is assigned to the radius of the inner circles positioned at the average of the oscillator phases, which are represented individually by the points in the outer unit circle. Furthermore, we can observe again the clustering of the phases, for the most part, close to 0° and 360°.

Finally, the panel on the right of Fig. 6 displays the Lyapunov spectra. For the three cases, the reservoir shows hyperchaotic behavior since more than one Lyapunov exponent is positive. As in Fig. 4, only a few Lyapunov exponents are close to zero, while most are negative. Table II provides the values of the upper bound of the KS-entropy, Shannon-entropy, dissipation, and Lyapunov dimension. In this case, we observe again that a larger Shannon-entropy

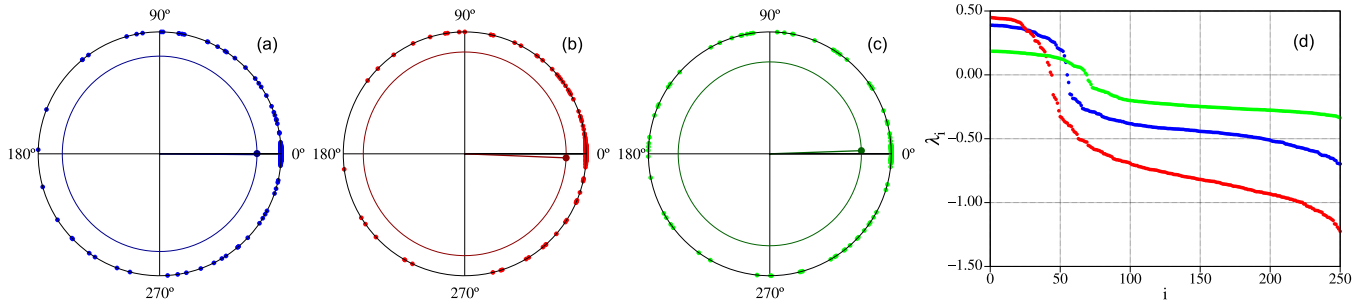

FIG. 6. The points on the outer circles of panels (a)–(c) are a representation of the phases of the oscillators at a given iteration. The average is represented by the point in the inner circle. The outer circle is unitary, while the inner circle has a radius equal to the order parameter r. The three rows of Table II are related, respectively, to panels (a)–(c). Panel (d) displays the corresponding Lyapunov spectra.

is related to the lower performance of the PRC. However, in distinction to the case discussed before, better performance is related to a higher Lyapunov dimension.

C. Performance and dynamics of the reservoir

The present section aims to analyze the dynamics of the computer reservoir when the parameters chosen by the GA are not optimal. The performance was measured based on the PRC's ability to maintain the distribution of elements in each segment of the reference series. We observed above that for the three series used, the parameters selected by the GA led the reservoir to exhibit similar dynamics. Therefore, we restrict ourselves to the first of the Gaussian series for this section. The parameters selected by the GA are presented in Table II. The two panels in Fig. 7 display the parameter spaces plot with the coupling strength κ *vs* β, the coefficient of the mean-field term. The color bars on the right represent in Fig. 7(a) the error presented by the PRC and in Fig. 7(b) the synchronization. We highlighted the parameters pair selected by the GA ($\kappa = 0.1526$ and $\beta = 1.3421$) and chose two other points, P1 and P2 (see Table III), to investigate the reservoir dynamics and evaluate the PRC performance. We observe in Fig. 7(a) that there are no well-defined regions for which the PRC's performance is larger. The parameters with the best performances (yellow to white points) appear to be random. However, the pair of parameters corresponding to the GA point has led the PRC to the best performance among all points in the parameter space. Overall, the best performances are presented by the parameter pairs in which the value of β predominates over κ. That is, as κ increases, the error tends to increase, as observed in the upper left corner of the parameter space. However, with the increase of β, the error tends to decrease again.

To investigate the relationship between the performance and parameters, and consequently, with the reservoir dynamics, we chose two other points from Fig. 7 where we kept $\beta = 0.25$ constant and increased the coupling strength κ. Table III contains the parameter pairs GA, P1, and P2, along with the errors associated with the PRC's performance under these configurations. We notice that, in this case, the best performance is obtained with a parameter pair where β predominates.

As mentioned before, the determining factor for performance evaluation, in this case, is not the emulation of the reference series iteration by iteration but rather the maintenance of its distribution. Therefore, in Fig. 8, we present the proportions of elements in each segment. The purple columns are related to the reference series, while the distributions of the segments from the PRC output, up to iterations 200, 400, and 600, are represented by the blue, green, and orange columns, respectively. The loss of performance in Figs. 8(a) and 8(b), compared to Fig. 5(a), is not as evident as it is in Table III. However, by carefully observing the green and orange columns, it becomes apparent that there is a greater distortion of the reference series' distribution (purple columns), especially in the columns at the edges, where there are fewer elements, and these differences become more visible.

Regarding synchronization, for the parameters selected by the GA point, the network's order parameter is $r = 0.80$ [see Fig. 4(a)]. When setting the coefficient of the mean-field term equal to 0.25 (point P1 in Fig. 7), the synchronization level is reduced to $r = 0.35$ [Fig. 10(a)]. In this configuration, as observed in Table III and

TABLE II. Parameters selected by the GA, the resulting errors ($\mathbb{E}$), the order parameter (r), the upper bound of the KS-entropy (h^{u}_{KS}), the dissipation ($-J$), the Shannon entropy (S), and the Lyapunov dimension (D). Related to Figs. 5–6.

κ	β	γ	ζ	$\mathbb{E}$	r	h^{u}_{KS}	$-J$	S	D
0.1526	1.3421	4.2427	0.6387	0.179 962	0.80	17.3	69.78	0.904	110.2
0.5442	1.5851	3.9802	0.6282	0.228 920	0.80	14.8	148.2	0.975	77.79
0.1942	1.0105	4.8636	0.4782	0.170 048	0.75	9.63	34.31	0.816	123.9

TABLE III. Parameters of the model, the resulting errors ($\mathbb{E}$), the order parameter (r), the upper bound of the KS-entropy (h^u_{KS}), the dissipation ($-J$), the Shannon entropy (S), and the Lyapunov dimension (D). Related to Figs. 7–10.

Points	κ	β	$\mathbb{E}$	r	h^u_{KS}	$-J$	S	D
GA	0.1526	1.3421	0.179 962	0.80	17.3	69.78	0.904	110.2
P1	0.1526	0.2500	0.331 862	0.35	4.27	7.718	0.944	209.8
P2	1.5000	0.2500	0.349 928	0.64	2.88	182.1	0.906	81.16

Panel 8(a), there is a loss of PRC performance. Keeping $\beta = 0.25$ and setting $\kappa = 1.5$ (point P2 in Fig. 7), the homogeneity in the network increases, as quantified by the order parameter $r = 0.65$ [Fig. 10(b)]. However, this parameter pair led the PRC to exhibit

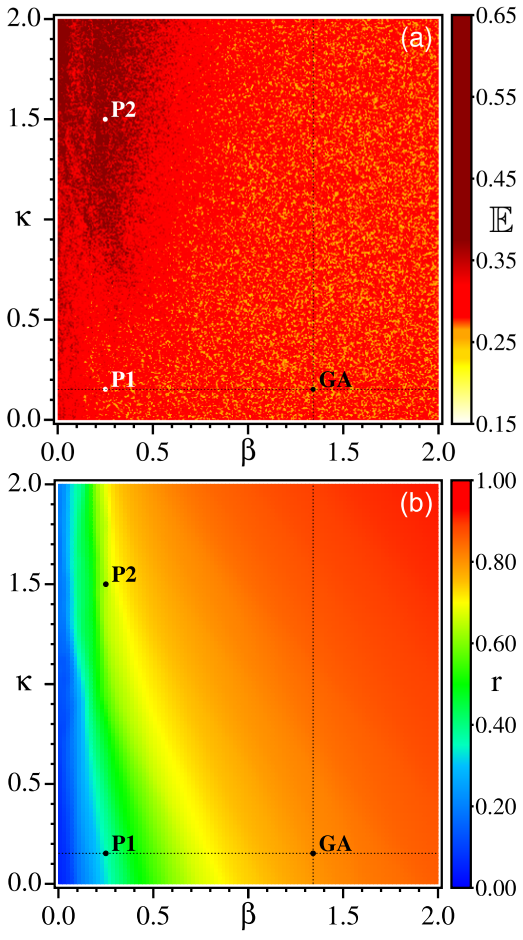

FIG. 7. Parameter space for (a) PRC errors and (b) sycnhronization represented by colors, as we vary the values of the coupling strength (κ) and the coefficient of the mean-field term (β), ranging from 0.000 to 2.000, with steps of size 5×10^{-3}. The location of points P1, P2, and GA is presented in Table III.

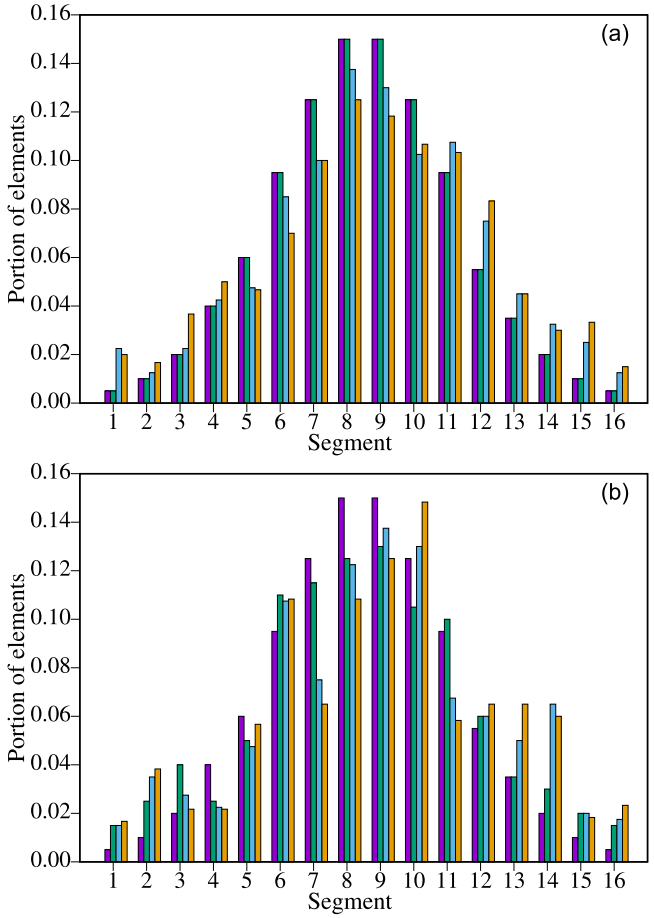

FIG. 8. Histograms with the number of elements in each segment of the reference series (purple) and of the PRC's outputs in the training (green), validation (blue), and test (orange) intervals. The panels are associated with the parameter pairs *P1* in (a) and *P2* in (b).

the worst performance among the investigated points [Table III and Fig. 8(b)]. To provide a more general possible relation between $\mathbb{E}$ and synchronization, we determine the latter in the whole analyzed parameter space, as shown in Fig. 7(b). Results demonstrate that the synchronization increases with β and that this effect is stronger for larger values of κ. However, no apparent relation is seen between results from Figs. 7(a) and 7(b). To check this, Fig. 9(a) combines data from Figs. 7(a) and 7(b) and displays the error $\mathbb{E}$ obtained for each value of the synchronization. The red line in Fig. 9(a) is the linear regression given by $\mathbb{E} \sim -0.09\, r + 0.36$, showing that the error slowly diminishes for larger synchronization. The main observation is that the most probable points are those with smaller errors ($\sim$0.27) and higher synchronization.

Furthermore, the distribution of errors is shown in Fig. 9(b), showing that the tail on the right is non-Gaussian, and the decay

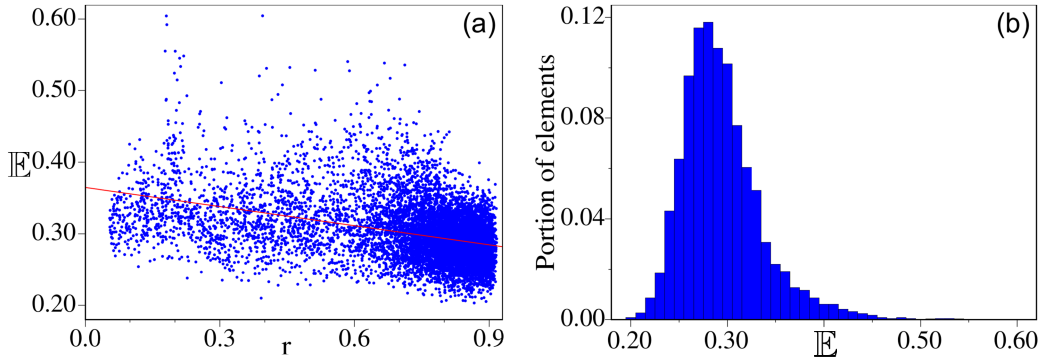

FIG. 9. Plotted is (a) the error ($\mathbb{E}$) *vs* the synchronization (r) and (b) the distribution of $\mathbb{E}$..

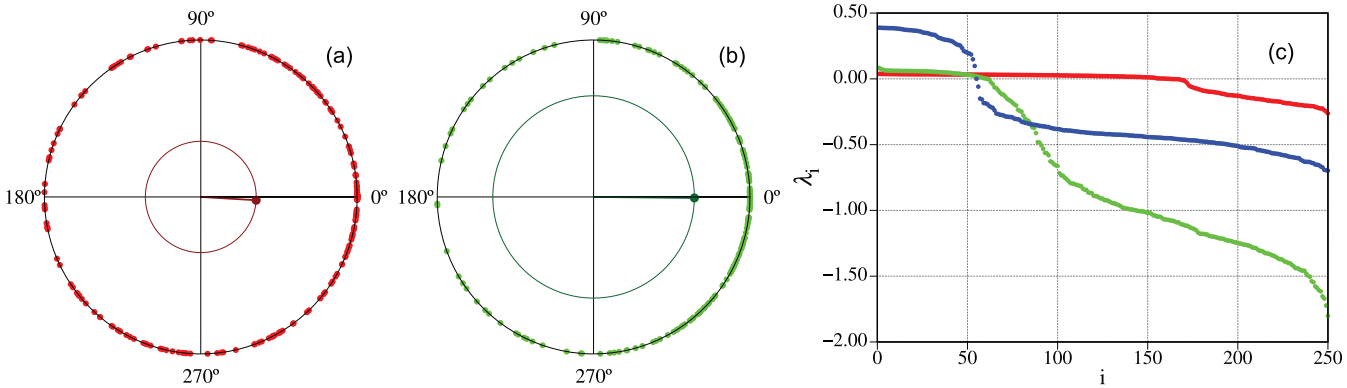

FIG. 10. The points on the outer circles of panels (a) and (b) are a representation of the phases of the oscillators at a given iteration. The average is represented by the point in the inner circle. The outer circle is unitary, while the inner circle has a radius equal to the order parameter r. In (a), the system parameters describing the dynamics were those represented by the P1 point and in (b) by the P2 point. See Table III and Fig. 7. Panel (c) displays the corresponding Lyapunov spectra with the blue points repeating the Lyapunov spectrum from Fig. 6.

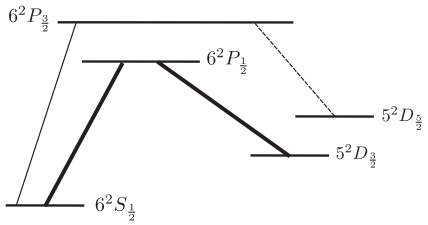

FIG. 11. Level structure of Ba^+. The metastable level is $5^2D_{\frac{5}{2}}$. The bold lines show the excitations by the lasers. The thin solid line represents the excitation caused by the hollow cathode lamp, while the subsequent decay to the $5^2D_{\frac{5}{2}}$ level is indicated by the dashed line.

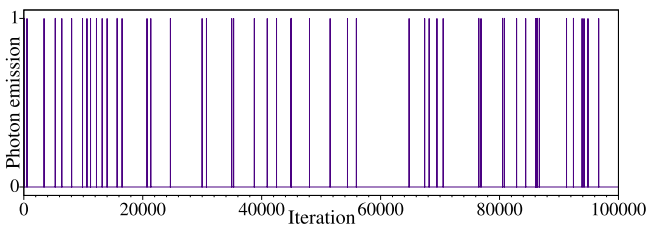

FIG. 12. Emission of photons by the Ba^+ ion obtained by solving the Schrödinger equation with the Hamiltonian given by Eq. (12). The metastable is the $5^2D_{\frac{5}{2}}$ state.

is slower than a Gaussian. Such heavy-tailed distribution possesses a higher probability for rare events than a similar Gaussian distribution.[65] Thus, larger errors become less probable (rare) events and tend to occur for smaller synchronizations, as can be checked in Fig. 9(a).

We also analyzed the behavior of the Lyapunov spectra, displayed on the right panel of Fig. 10. Blue points for the GA point, red for P1, and green for P2. It becomes evident that the number of positive Lyapunov exponents is crucial for the performance. For the red points, a very small number of positive Lyapunov exponents exist. Finally, and in distinction to previous results, comparing cases P1 and P2, the worst performance is related to a smaller Shannon-entropy and Lyapunov dimension.

IV. APPLICATION: SERIES GENERATED BY QUANTUM JUMPS

In Ref. 47, the authors demonstrate a direct observation of quantum jumps between the $6^2S_{\frac{1}{2}}$ and $5^2D_{\frac{5}{2}}$ levels of a single Ba^+ ion confined in a radio-frequency trap. In this experiment, an individual ion (Ba^+) with two excited states is trapped in a radio-frequency trap. Two lasers drive the ion to one of these excited states ($6^2P_{\frac{1}{2}}$), in which the ion tends to remain for an average of 1 s. Each time the ion returned, it emitted enough photons to be detected by the experimental apparatus approximately every 1 s, making the detected signal approximately constant. To take the ion to the other excited state ($6^2P_{\frac{3}{2}}$), a hollow cathode lamp with a filter for the desired frequency was used. Upon returning to the ground state, in the case of this second excited state, the ion occasionally went to a metastable state $5^2D_{\frac{5}{2}}$, which had the peculiarity of having a longer dwell time (on average 30 s) than the other states involved in the experiment. While the ion was in this state, no photon emission occurred, which is a direct observation of the phenomenon. A visual model of these Ba^+ level structures is presented schematically in Fig. 11.

The main interest in this context is to analyze how many times and for how long the system stays in the shelved state with no photon emission. For this, the authors in Ref. 47 counted the dwell times in the metastable state for 203 consecutive times and presented them in a histogram, along with a theoretical exponential model. They observed that the distribution of these dwell times tends to be a negative exponential. Therefore, we aim to reproduce such decay in the distribution of dwell times. To obtain results similar to those obtained in Ref. 47, we calculate the probability of photon emission in the archived state by defining the five-level quantum system, as shown schematically in Fig. 11. We can write the effective

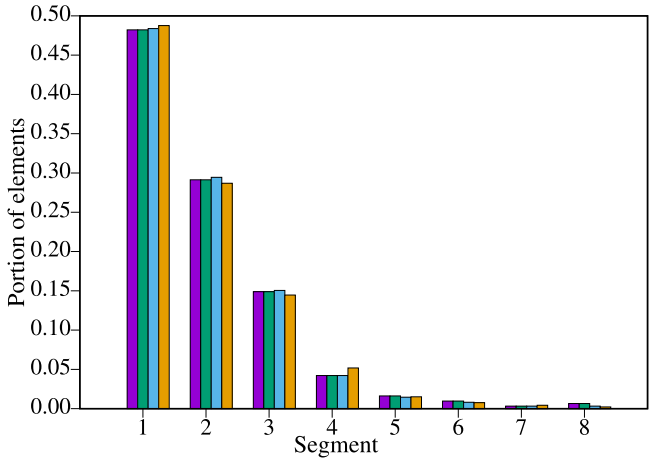

FIG. 13. Histograms with the number of elements in each segment of the reference series (purple) and the PRC's outputs in the training (green), validation (blue), and test (orange) intervals.

Hamiltonian of the system as[50]

$$H_{\text{eff}} = \sum_{i=1}^{5} \hbar\omega_i |i\rangle\langle i| + \sum_{i\neq j} \left(\hbar g_{ij} |i\rangle\langle j| + h.c.\right) + \sum_{i=1}^{2} \hbar\Omega_{i,i+1}\left(|i\rangle\langle i+1| + |i+1\rangle\langle i|\right) + \hbar\Omega_b\left(|1\rangle\langle 2| + |2\rangle\langle 1|\right), \tag{12}$$

where ω_i are the states energies, g_{ij} is the coupling constant between the states $|i\rangle$ and $|j\rangle$, *h.c.* represents the conjugate Hermitian term, Ω_{ij} is the laser term, and Ω_b is the lamp excitation term. For the numerical simulations, we use energy states defined by $|1\rangle$ as ground state $6^2S_{1/2}$, $|2\rangle$ excited state $6^2P_{1/2}$, $|3\rangle$ excited state $6^2P_{3/2}$, $|4\rangle$ intermediate state $5^2D_{3/2}$, and $|5\rangle$ metastable state $5^2D_{5/2}$. The constants are $\Omega_{12} = 0.5$, $\Omega_{23} = 0.05$, and $\Omega_b = 1$. By solving the time-dependent Schrödinger equation, we evolved the system, observed the emission of photons as a function of time, and generated the reference series, as shown in Fig. 12, with 3×10^4 points in the dwell state. Then, we divided this set of values into eight equally spaced segments, meaning that smaller times belong to segment 1,

TABLE IV. Parameters selected by the GA, the resulting errors ($\mathbb{E}$), the the upper bound of the KS-entropy (h^{u}_{KS}), the dissipation ($-J$), Shannon entropy (S), and the dimensionality of the attractor (D). Related to Figs. 11–14.

κ	β	γ	ζ	$\mathbb{E}$	r	h^{u}_{KS}	$-J$	S	D
1.2066	1.4989	4.4838	0.5731	0.062 347	0.86	13.5	244.4	0.596	60.58

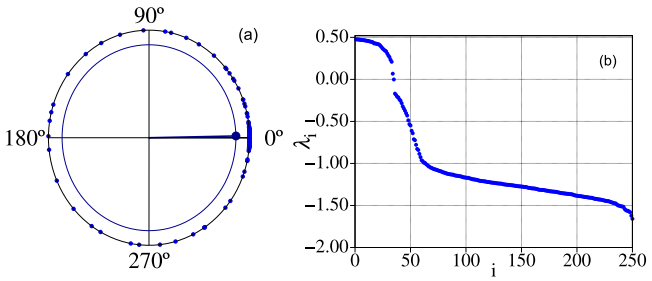

FIG. 14. In panel (a), points on the outer circle are a representation of the phases of the oscillators at a given iteration. The average is represented by the point in the inner circle. The outer circle is unitary, while the inner circle has a radius equal to the order parameter r. Panel (b) shows the corresponding Lyapunov spectrum.

and larger times belong to segment 8. That was the reference series used for the learning process.

Figure 13 presents the counting of consecutive dwell times in the metastable state, divided into eight segments as described earlier, both from the reference series in the training interval, which in this case goes up to sample 309 (purple column), and from the output of the PRC in the same interval (green column), up to sample 618 (blue column), and up to sample 927 (orange column). We observed that the PRC satisfactorily generated an output whose distribution in each segment closely resembles the reference series. Larger and smaller numbers of segments than 8 were checked, but the distributions obtained were inadequate.

The performance is evaluated based on the PRC's ability to maintain the distribution that the reference series presents in the training segment. Both the error and the parameters selected by the GA are shown in Table IV.

For the order parameter, we found $r = 0.86$ (see left panel from Fig. 14), a value close to those presented in Secs. III A and III B. For the spectrum of Lyapunov exponents (right panel from Fig. 14), it can be observed that the network composing the reservoir exhibits hyperchaotic behavior. A few exponents are near zero, several are positive, and the majority are negative. The Shannon-entropy and the Lyapunov dimension are small and comparable to the random series (see Table I).

V. CONCLUDING REMARKS

In this paper, we aimed for an improved understanding of the intermediate processes of machine learning, which leads to the best performance in specific tasks. To achieve this, we propose a physical reservoir computing model, where the units of the neural network forming the reservoir are modified Kuramoto oscillators, and their phases represent the values that sum up to form the output. We investigated the behavior of the reservoir with the parameters of Kuramoto's model that led the PRC to exhibit the best performances in the learning reference series. Additionally, we changed the parameters of Kuramoto's model and analyzed the dynamics of the reservoir when the performance was unsatisfactory, aiming to identify characteristics that relate the dynamics to the performance.

It is common in reservoir computing publications to choose reference series that resemble external inputs. However, we chose not to propose external inputs, maintaining the reservoir with the same configuration for all tasks. Therefore, we adjusted only the connection weights between the units of the reservoir and the output as well as the parameter values, which were selected for each task using the genetic algorithm.

We proposed a new method of evaluating performance, which measures how much it can *maintain the distribution* of the series (number of elements in each segment). Regarding the relationship between the ability of the PRC to maintain the distributions and its underlying nonlinear dynamics, we found the following:

(i) For equally distributed random series, the performance (measured by the error) is proportional to the upper bound of the KS-entropy of the Kuramoto oscillators, the partial Shannon-entropy, and the Lyapunov dimension D (see Table I).
(ii) For Gaussian distributed random series, the performance is proportional to the Shannon-entropy and inversely proportional to the Lyapunov dimension (see Table II). Best performances present a high degree of synchronization of Kuramoto's oscillator, around $r \sim 0.8$.
(iii) The above results are compared to PRC's performance for two cases (points P1 and P2 in Fig. 7) where the GA algorithm did not choose the parameters. In both cases, the performance decreases together with a substantial decrease in the upper bound of the KS-entropy and the amount of synchronization but an increase in the Shannon-entropy (see Table III). Therefore, hyperchaotic motion, high degrees of synchronization, and smaller Shannon-entropy are directly related to the best performance of the PRC.
(iv) In the quantum application, good performance of the PRC is obtained in maintaining the time distribution of shelved states. This is also a consequence of the hyperchaotic dynamics of Kuramoto's oscillators, their high degree of synchronization, and small Shannon-entropy (see Table IV).

To summarize, our results demonstrate that a balance of certain nonlinearity and synchronization between elements of the network in a computer reservoir is necessary to obtain a good performance. The coexistence of regular and chaotic motions in the same dynamical systems is known as mixed dynamics and is characteristic of, for example, typical Hamiltonian systems,[66–72] which may contain memory effects, sticky motion, anomalous transport, and fractional derivates among other properties. Furthermore, smaller amounts of information gain of the series, measured by the (partial) Shannon-entropy, are directly related to a better performance of the RC. Future developments may consider, in more detail, and within a more general context, the relation between the performance of other PRCs and the upper bound of the KS-entropy, the Shannon-entropy, the dissipation, and the Lyapunov dimension.

We would like to mention a possible generalization of our results. We may intuitively argue that completely hyperchaotic dynamics, with zero time correlation and vanishing synchronization, cannot reproduce or mimic a given signal. The dynamic is too

irregular to provide information on the signal. On the other hand, the limit of a completely regular dynamics cannot mimic a random signal, since the regular dynamics cannot change and adequately react to external signals. Therefore, in agreement with our results, we conjecture that the *mixture* of regular and hyperchaotic dynamics is most likely capable of reacting for adequate predictions of complex signals. However, we emphasize that such conjecture is based on our study model, and more studies are necessary to propose generalization. From the machine learning methodology point of view, our results suggest that it is possible to model an efficient computer reservoir without any input signal.

ACKNOWLEDGMENTS

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES, Brazil)—Finance Code 001. M.W.B. acknowledges Conselho Nacional de Desenvolvimento Cientúifico e Tecnológico (CNPq, Brazil) for financial support (Grant No. 310294/2022-3). E.L.B. acknowledges São Paulo Research Foundation (FAPESP) under Grant Nos. 2021/12232-0 and 2018/03211-6.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

L. A. S. Rosa and E. L. Brugnago developed the ideas, series, and codes for the ML and wrote the main initial version of the paper. G. J. Delben realized the quantum simulations and described the related part. M. W. Beims added interpretations related to the entropies. J. M. Rost and M. W. Beims suggested the main proposal of the paper and revised all results and conclusions. All authors revised the paper.

Lucas A. S. Rosa: Data curation (lead); Formal analysis (lead); Investigation (lead); Methodology (lead); Software (lead); Validation (equal); Writing – original draft (lead); Writing – review & editing (equal). **Eduardo L. Brugnago:** Conceptualization (lead); Formal analysis (lead); Investigation (lead); Methodology (lead); Software (lead); Supervision (equal); Writing – original draft (lead); Writing – review & editing (supporting). **Guilherme J. Delben:** Investigation (equal); Methodology (equal); Software (equal); Writing – review & editing (equal). **Jan-Michael Rost:** Conceptualization (equal); Visualization (equal); Writing – review & editing (equal). **Marcus W. Beims:** Conceptualization (lead); Methodology (equal); Project administration (equal); Supervision (lead); Validation (equal); Writing – original draft (supporting); Writing – review & editing (lead).

DATA AVAILABILITY

Data openly available in a public repository that does not issue DOIs.

APPENDIX A: GENETIC ALGORITHM

In our model, we use the GA to choose the parameters κ, β, ζ, and γ from Eq. (2). First, we randomly generated 10 sets of parameters, named *specimens*, and performed the learning process. The parameters β, κ, γ, and ζ were generated randomly with a uniform distribution. The lower and upper bounds were, respectively, 0 and 2 for β and κ and 0 and 2π for γ and ζ. To spawn 40 more specimens, we raffle pairs of the four parameters from the 10 specimens and combine them. These pairs are called *genitors* since their combination creates a new specimen. To combine parameters, we convert them from real type with double precision to integers, within the range from 0 to $2^{15} - 1$, in binary numbers. Then, we raffle a value between 2 and 15 to be the bit where the parameter cut will occur, i.e., until that bit, one genitor gives his values, after that bit, the other genitor is the donor. Next, we ranked the 50 specimens by performance, selected the best 10, and repeated the procedure for $G = 50$ *generations*. After this, to avoid some local minima, to each parameter, there is a 50% chance for a mutation to occur, i.e., one of the bits that compose the parameter has its value changed (if it was zero, it becomes one and vice versa). We repeated this entire procedure in every generation.

We have also used GA to select the initial conditions and the natural frequency of oscillation of the new specimens but with a slight difference in the recombination process. In this case, we opted to raffle a number within 2 and the number of oscillators N to do the cut. Until that point, one of the genitors gives his values, and from it, the other. In the first generation, the initial conditions of the 250 oscillators of the initial 10 specimens were randomly generated from a uniform distribution, with values limited between 0 and 2π. The natural frequencies of these oscillators were also randomly generated, but we opted for a Lorentzian distribution based on what is usual in the literature when working with Kuramoto systems. The lower and upper bounds were 10^{-2} and 10^{2}, respectively. After the 50th generation, we selected the parameters of the best specimen obtained by the genetic algorithm, i.e., the parameters that led our model to present the best performance in these first 50 generations.

Since every time we initialize the code again, the random number function generates new values, and we have run the same code 50 times for each task. In Sec. III, we have studied the reservoir dynamics with the parameter sets that led our model to get the best performances.

APPENDIX B: FLOW CHART

To outline the steps used in this article, we present in Fig. 15 a scheme that summarizes all the stages of our study. The red blocks, rectangles with rounded edges, correspond to the beginning and end of the procedure for each chosen reference series. The blue trapezoids indicate the initial information and the final results of the cycles, the orange rectangles are the intermediate processes, and the green diamonds are the decisions that guide us through the sequence of processes. The blocks on the left, surrounded by dashed blue lines, summarize the analysis performed for each reference series, while the blocks on the right, painted with less intense colors and surrounded by dashed orange lines, are a flow chart of the genetic algorithm.

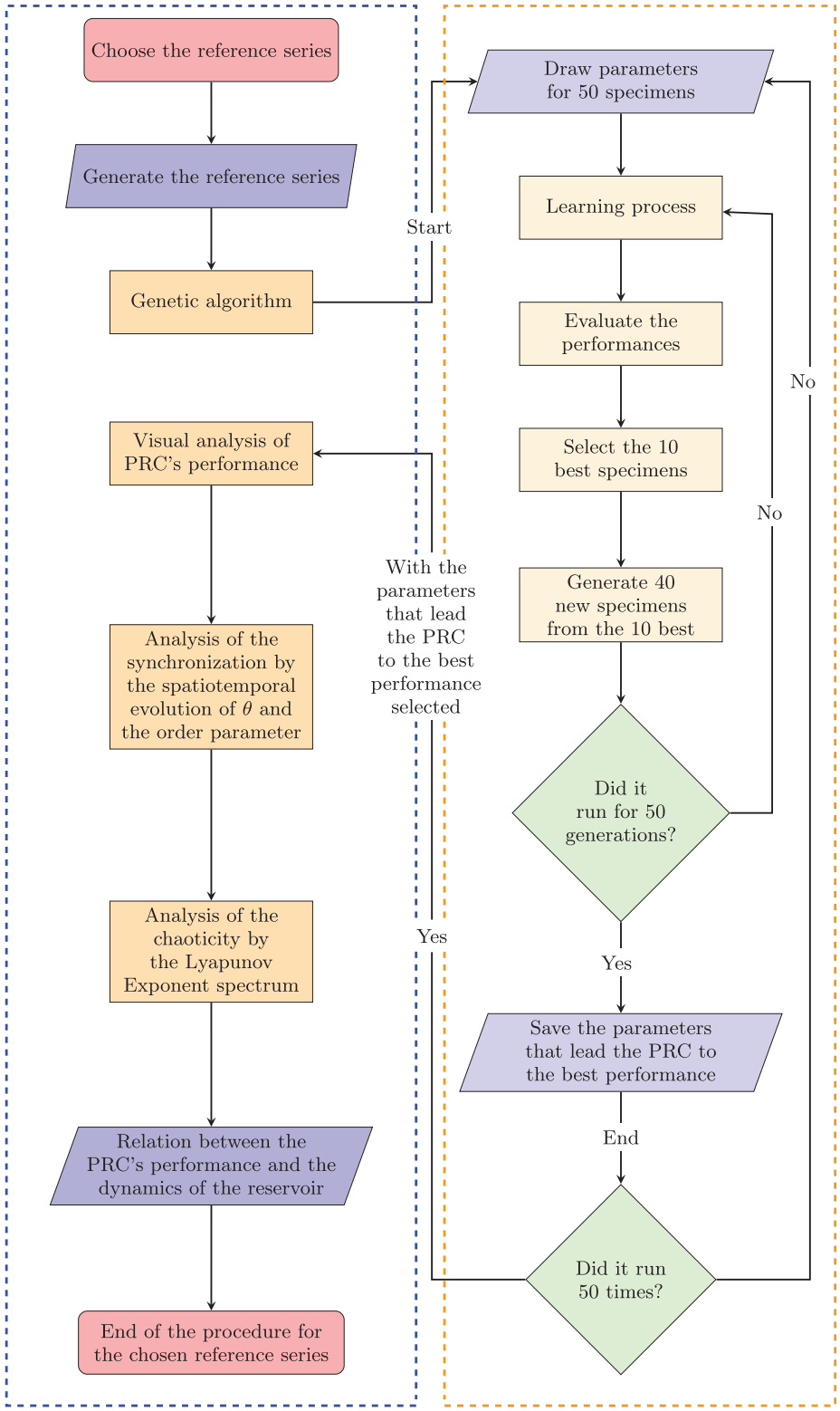

FIG. 15. On the left, inside the blue dashed rectangle, a summary of the analysis performed for each reference series. On the right, inside the orange frame, is a flow chart of the procedure adopted by the genetic algorithm.

REFERENCES

[1]D. H. Ballard and C. M. Brown, *Computer Vision* (Prentice-Hall, 1981).
[2]A. K. Joshi, "Natural language processing," Science **253**, 5025 (1991).
[3]M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," Science **349**, 255 (2015).
[4]C. Cao, S. Y. Hou, N. Cao, and B. Zeng, "Supervised learning in hamiltonian reconstruction from local measurements on eigenstates," J. Phys. Condens. Matter **33**, 064002 (2020).
[5]P. Huembeli, A. Dauphin, and P. Wittek, "Identifying quantum phase transitions with adversarial neural networks," Phys. Rev. B **97**, 134109 (2018).
[6]J. Behler and M. Parrinello, "Generalized neural-network representation of high-dimensional potential-energy surfaces," Phys. Rev. Lett. **98**, 146401 (2007).
[7]M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," J. Comput. Phys. **378**, 686 (2019).
[8]E. L. Brugnago, C. C. Felicio, and M. W. Beims, "Forecasting the duration of three connected wings in a generalized Lorenz model," Int. J. Bifurcat. Chaos **32**, 2230031 (2022).
[9]E. L. Brugnago, J. A. C. Gallas, and M. W. Beims, "Predicting regime changes and durations in Lorenz's atmospheric convection model," Chaos **30**, 103109 (2020).
[10]E. L. Brugnago, J. A. C. Gallas, and M. W. Beims, "Machine learning, alignment of covariant Lyapunov vectors and predictability in Rikitake's geomagnetic dynamo model," Chaos **30**, 083106 (2020).
[11]E. L. Brugnago, T. A. Hild, D. Weingaertner, and M. W. Beims, "Classification strategies in machine learning techniques predicting regime changes and durations in the Lorenz system," Chaos **30**, 053101 (2020).
[12]G. Schneider and P. Wrede, "Prediction of the secondary structure of proteins from the amino acid sequence with artificial neural networks," Angew. Chem. Int. Ed. English **32**, 1141 (1993).
[13]K. T. Butler, D. W. Davies, H. Cartwright, O. Isayev, and A. Walsh, "Machine learning for molecular and materials science," Nature **559**, 547 (2018).
[14]J. Behler, "First principles neural network potentials for reactive simulations of large molecular and condensed systems," Angew. Chem. Int. Ed. **56**, 12828 (2017).
[15]C. Sommer and D. W. Gerlich, "Machine learning in cell biology-teaching computers to recognize phenotypes," J. Cell Sci. **126**, 5529–5539 (2013).
[16]C. Angermueller, T. Pärnamaa, L. Parts, and O. Stegle, "Deep learning for computational biology," Mol. Syst. Biol. **12**, 878 (2016).
[17]G. S. Handelman, H. K. Kok, R. V. Chandra, A. H. Razavi, M. J. Lee, and H. Asadi, "Edoctor: Machine learning and the future of medicine," J. Intern. Med. **284**, 603 (2018).
[18]A. Rajkomar, J. Dean, and I. Kohane, "Machine learning in medicine," N. Engl. J. Med. **380**, 1347 (2019).
[19]J. H. Holland and J. H. Miller, "Artificial adaptive agents in economic theory," Am. Econom. Rev. **81**, 365–370 (1991); available at http://www.jstor.org/stable/2006886.
[20]J. Kleinberg, J. Ludwig, S. Mullainathan, and Z. Obermeyer, "Prediction policy problems," Am. Econom. Rev. **105**, 491 (2015).
[21]J. Kleinberg, H. Lakkaraju, J. Leskovec, J. Ludwig, and S. Mullainathan, "Human decisions and machine predictions," Quarterly J. Econom. **133**, 237–293 (2018).
[22]S. O. Haykin, *Neural Networks and Learning Machines* (Pearson Education, 2011).
[23]S. Russel and P. Norvig, *Artificial Intelligence: A Modern Approach* (Prentice Hall, 2012).
[24]D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature **323**, 533–536 (1986).
[25]M. I. Jordan, "Chapter 25 - serial order: A parallel distributed processing approach," in *Neural-Network Models of Cognition*, edited by J. W. Donahoe and V. P. Dorsel (North-Holland, 1997), Vol. 121, pp. 471–495.
[26]M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," Comput. Sci. Rev. **3**, 127–149 (2009).
[27]H. Jaeger, *Tutorial on Training Recurrent Neural Networks, Covering Bppt, RTRL, EKF and the Echo State Network Approach* (GMD Forschungszentrum Informations technik, 2002).
[28]W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," Neural Computat. **14**, 2531 (2002).
[29]J. C. Coulombe, M. C. A. York, and J. Sylvestre, "Computing with networks of nonlinear mechanical oscillators," PLoS One **12**, e0178663 (2017).
[30]G. Urbain, J. Degrave, B. Carette, J. Dambre, and F. Wyffels, "Morphological properties of mass-spring networks for optimal locomotion learning," Front. Neurorobot. **11**, 16 (2017).
[31]D. Nikolić, S. Häusler, W. Singer, and W. Maass, "Distributed fading memory for stimulus properties in the primary visual cortex," PLoS Biol. **7**, e1000260 (2009).
[32]M. Dranias, H. Ju, E. Rajaram, and A. M. J. VanDongen, "Short-term memory in networks of dissociated cortical neurons," J. Neurosci. **33**, 1940 (2013).
[33]K. Nakajima, K. Fujii, M. Negoro, K. Mitarai, and M. Kitagawa, "Boosting computational power through spatial multiplexing in quantum reservoir computing," Phys. Rev. Appl. **11**, 034021 (2019).
[34]S. Ghosh, A. Opala, M. Matuszewski, T. Paterek, and T. C. H. Liew, "Quantum reservoir processing," npj Quant. Inform. **5**, 6 (2019).
[35]L. Appeltant, M. C. Soriano, G. V. D. Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. R. Mirasso, and I. Fischer, "Information processing using a single dynamical node as complex system," Nat. Commun. **2**, 468 (2011).
[36]M. L. Alomar, M. C. Soriano, M. Escalona-Morán, V. Canals, I. Fischer, C. R. Mirasso, and J. L. Rosselló, "Digital implementation of a single dynamical node reservoir computer," IEEE Trans. Circuits Systems II Express Briefs **62**, 977–981 (2015).
[37]Y. Kuramoto, "Self-entrainment of a population of coupled non-linear oscillators," in *International Symposium on Mathematical Problems in Theoretical Physics*, edited by H. Araki (Springer Berlin Heidelberg, 1975), pp. 420–422.
[38]M. Breakspear, S. Heitmann, and A. Daffertshofer, "Generative models of cortical oscillations: Neurobiological implications of the Kuramoto model," Front. Hum. Neurosci. **4**, 1 (2010).
[39]J. H. Holland, "Genetic algorithms," Sci. Am. **267**, 66 (1992).
[40]Z. Michalewicz, *Genetic Algorithms + Data Structures= Evolution Programs* (Springer, 1996).
[41]C. E. Shannon, "A mathematical theory of communication," Bell Syst. Tech. J. **27**, 379 (1948).
[42]K. V. Mardia, *Statistics of Directional Data* (Academic Press, 1972).
[43]H. Haken, *Synergetics: An Introduction: Nonequilibrium Phase Transitions and Self-organization in Physics, Chemistry, and Biology* (Springer, 1983).
[44]P. A. Tass, *Phase Resetting in Medicine and Biology: Stochastic Modelling and Data Analysis* (Springer Berlin Heidelberg, 2007).
[45]G. Benettin, L. Galgani, and J. M. Strelcyn, "Kolmogorov entropy and numerical experiments," Phys. Rev. A **14**, 2338 (1976).
[46]D. Ruelle, "An inequality for the entropy of differentiable maps," Bol. Soc. Bras. Mat **9**, 83 (1978).
[47]W. Nagourney, J. Sandberg, and H. Dehmelt, "Shelved optical electron amplifier: Observation of quantum jumps," Phys. Rev. Lett. **56**, 26 (1986).
[48]P. Zoller, M. Marte, and D. F. Walls, "Quantum jumps in atomic systems," Phys. Rev. A **35**, 198 (1987).
[49]B. M. Garraway, M. S. Kim, and P. L. Knight, "Quantum jumps, atomic sheving and Monte Carlo fluorescence spectra," Opt. Commun. **117**, 560 (1995).
[50]M. B. Plenio and P. L. Knight, "The quantum-jump approach to dissipative dynamics in quantum optics," Rev. Modern Phys. **70**, 101 (1998).
[51]P. M. Billangeon, J. S. Tsai, and Y. Nakamura, "Circuit-qed-based scalable architectures for quantum information processing with superconducting qubits," Phys. Rev. B **91**, 094517 (2015).
[52]Y. Yariv and A. C. Aashish, "Shelving-style QND phonon-number detection in quantum optomechanics," New J. Phys. **19**, 033014 (2017).
[53]J. P. Garrahan and M. Guță, "Catching and reversing quantum jumps and thermodynamics of quantum trajectories," Phys. Rev. A **98**, 052137 (2018).
[54]J. L. Pei, M. W. Brendan, A. C. Andrew, L. D. Matthew, and S. Crystal, "Practical trapped-ion protocols for universal qudit-based quantum computing," Phys. Rev. Res. **2**, 033128 (2020).
[55]A. S. Reis, K. C. Iarosz, F. A. S. Ferrari, I. L. Caldas, A. M. Batista, and R. L. Viana, "Bursting synchronization in neuronal assemblies of scale-free networks," Chaos, Solitons Fractals **142**, 110395 (2021).

[56]S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: Past, present, and future," Multimedia Tools Appl. **80**, 8091 (2021).
[57]A. S. Reis, I. L. Caldas, and R. L. Viana, "Sobre vagalumes, pedestres e neurônios: A sincronização de osciladores de fase," Revista Brasileira de Ensino de Física **44**, e20210368 (2022).
[58]S. H. Strogatz, *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering* (Levant Books, 2007).
[59]E. Ott, *Chaos in Dynamical Systems* (Cambridge University Press, 2002).
[60]Ja. B. Pesin, "Families of invariant manifolds that correspond to nonero characteristic exponents," Mathemat. USSR-Izvestiya **10**, 1261–1305 (1976).
[61]P. Grassberger and I. Procaccia, "Estimation of the Kolmogorov entropy from a chaotic signal," Phys. Rev. A **28**, 2591 (1983).
[62]J. L. Kaplan and J. A. Yorke, *Chaotic Behavior of Multidimensional Difference Equations, in Functional Differential Equations and Approximations of Fixed Points* (Springer, 1979).
[63]R. Engelken, F. Wolf, and L. F. Abbot, "Lyapunov spectra of chaotic recurrent neural networks," Phys. Rev. Res. **5**, 043044 (2023).
[64]G. E. P. Box and M. E. Muller, "A note on the generation of random normal deviates," Ann. Math. Stat. **29**, 610 (1958).
[65]K. Holger, *Rare and Extreme Events*, edited by H. Atmanspacher and S. Maasen (John Wiley & Sons, 2016).
[66]G. Zaslavsky, *Hamiltonian Chaos and Fractional Dynamics* (University Press, 2006).
[67]M. W. Beims, C. Manchein, and J. M. Rost, "Origin of chaos in soft interactions and signatures of nonergodicity," Phys. Rev. E **76**, 056203 (2007).
[68]C. Manchein, M. W. Beims, and J. M. Rost, "Characterizing the dynamics of higher dimensional nonintegrable conservative systems," Chaos **22**, 033137 (2012).
[69]C. Manchein, M. W. Beims, and J. M. Rost, "Characterizing weak chaos in non-integrable Hamiltonian systems: The fundamental role of stickiness and initial conditions," Physica A **400**, 186 (2014).
[70]M. S. Custódio and M. W. Beims, "Intrinsic stickiness and chaos in open integrable billiards: Tiny border effects," Phys. Rev. E **83**, 056201 (2011).
[71]H. A. Oliveira, C. Manchein, and M. W. Beims, "Soft wall effects on interacting particles in billiards," Phys. Rev. E **78**, 046208 (2008).
[72]R. M. da Silva, C. Manchein, and M. W. Beims, "Characterizing weak chaos using time series of Lyapunov exponents," Phys. Rev. E **91**, 062907 (2015).