scientific reports

Check for updates

OPEN

A Hopf physical reservoir computer

Md Raf E Ul Shougat✉, XiaoFu Li, Tushar Mollik & Edmon Perkins

Physical reservoir computing utilizes a physical system as a computational resource. This nontraditional computing technique can be computationally powerful, without the need of costly training. Here, a Hopf oscillator is implemented as a reservoir computer by using a node-based architecture; however, this implementation does not use delayed feedback lines. This reservoir computer is still powerful, but it is considerably simpler and cheaper to implement as a physical Hopf oscillator. A non-periodic stochastic masking procedure is applied for this reservoir computer following the time multiplexing method. Due to the presence of noise, the Euler–Maruyama method is used to simulate the resulting stochastic differential equations that represent this reservoir computer. An analog electrical circuit is built to implement this Hopf oscillator reservoir computer experimentally. The information processing capability was tested numerically and experimentally by performing logical tasks, emulation tasks, and time series prediction tasks. This reservoir computer has several attractive features, including a simple design that is easy to implement, noise robustness, and a high computational ability for many different benchmark tasks. Since limit cycle oscillators model many physical systems, this architecture could be relatively easily applied in many contexts.

Reservoir computing (RC) is a bio-inspired, supervised machine-learning computational framework based on artificial recurrent neural networks (RNNs), which utilizes naturally emergent dynamics of a physical resource[1–6]. Conventional machine learning schemes use backpropagation through time[7] to train an *entire* recurrent neural network. This method is computationally expensive, since all the weights of the network need to be updated to mimic a target function. *Echo state networks*[8] and *liquid state machines*[9] are two concepts that addressed this issue in the early 2000s. Reservoir computing merges these concepts. In reservoir computing, the neural network is formed from a set of coupled nonlinear nodes, where the network is divided into three parts: an input layer, the reservoir, and the readout layer. Unlike conventional RNNs, only the readout layer requires training by a simpler training algorithm, such as linear or ridge regression[10]. Thus, the RC architecture is much faster and more stable than conventional RNN methods, which is one of the key advantages of this information processing framework.

There are many real-world applications of reservoir computing, including bit-wise logical operations[11–13], speech recognition[6], handwritten digit recognition[14], wireless communications[1], complex and chaotic time series predictions[1,6,15–18], image recognition[19], emulation of nonlinear time series[4,10], and morphological computation[20,21]. The echo state architecture of a reservoir allows the use of physical systems as reservoir computers, also known as *physical reservoir computers* (PRCs). Many physical systems have been shown to perform as PRCs, including an array of nonlinear mechanical oscillators[11,22,23], soft robotic bodies[20,24–26], tensegrity structures[21,27], and origami structures[28,29].

Importantly, quantum systems can be used as PRCs. The natural disordered quantum dynamics of an ensemble system was utilized to emulate nonlinear time series, including a chaotic system[30]. A Kerr nonlinear oscillator was used in sine wave phase estimation using its complex amplitudes as computational nodes[31]. Nuclear-magnetic-resonance spin-ensemble system was used for nonlinear dynamics emulation task by implementing spatial multiplexing approach to increase computational power[32]. Dissipative quantum dynamics was used to build a quantum reservoir computer (QRC) for nonlinear temporal tasks[33].

Physical reservoir computers were initially constructed from only the coupled, real dynamic nodes. Later, a virtual node-based reservoir computing method was proposed by implementing a time multiplexing approach in which a delayed feedback was used as a single nonlinear dynamic node to perform computation[6]. This method simplifies the complexity of a reservoir built from an array of physical nonlinear nodes. This approach has been popularly used to construct physical reservoir computers for different tasks, such as an optoelectronic oscillator for optical information processing[34], a photonics-based passive linear fiber reservoir for signal processing[35], an FPGA implementation using a single autonomous Boolean logic element for pattern recognition[5], time-delay reservoirs for forecasting of stochastic nonlinear time series[36], a delayed Duffing silicon beam for parity tasks[12], and a semiconductor laser with delayed optical feedback for nonlinear time series prediction[37]. These reservoirs used a delay line to create the necessary nodes for computation. A simpler approach can be taken by creating

LAB2701: Nonlinear Dynamics Laboratory, Department of Mechanical and Aerospace Engineering, North Carolina State University, Raleigh, NC 27695, USA. ✉email: mdrafeulshougat@ncsu.edu

Description	Nomenclature	Description	Nomenclature
First state	x	Second state	y
Input	u	Mask	m
Resonance constant	ω_0	Amplitude of sinusoidal forcing	A
Frequency of sinusoidal forcing	Ω	Phase of sinusoidal forcing	ϕ
Parameter affecting limit cycle radius	μ	Noise amplitude	σ
White Gaussian noise	$\dot{W}$	Noise bias	β
Re-scaled x state	X	Identity matrix	I
Nodal state matrix	L	Target vector	M
Information rate	R	Pseudo-period	T_p
Output of PRC	o	Number of nodes	N

Table 1. List of parameters, states, and functions.

the nodes without the presence of any delay or feedback line[38]. This approach is studied less, though it makes the reservoir architecture much simpler.

Here, a Hopf oscillator is used as a physical reservoir. The Hopf oscillator can also be used as the building block for adaptive oscillators[39,40], which can natively learn information without any training. The Hopf oscillator can exhibit limit cycle motion, which provides a source of memory by storing information in its dynamic states. Although a binary periodic masking function is popularly used for time-multiplexed reservoir[6,10,12], noise can also be used as periodic mask[41]. In this paper, a Hopf oscillator PRC is constructed that uses a non-periodic stochastic mask. A Hopf oscillator physical reservoir computer is fabricated as an analog circuit, which is compared with Euler–Maruyama simulations[40,42,43]. This Hopf PRC can successfully complete benchmark machine learning tasks, including parity tasks, fundamental logic gate tasks[12], nonlinear dynamic emulation tasks[4], and various time series prediction tasks[44]. The information rate is used as the performance metric for logical tasks[11], and the normalized mean square error (NMSE) is used for the emulation and time series tasks[4].

The rest of the article is organized as follows. In "System equations for Hopf physical reservoir computer" section, the equations of motion for the stochastic Hopf oscillator PRC are presented. In "Mapping methodology" section, the methodology of mapping the oscillator's dynamics to an information processing scheme is discussed for an example task by using the Euler–Maruyama simulation. The effects of the pseudo-period and the noise on computational ability are discussed in "Pseudo-period and noise" section. In "Analog circuit experiment" section, the analog circuit experiment is described. In "Benchmark tasks for Hopf PRC" section, different benchmark tasks are performed with the numerical and experimental Hopf PRC, which includes logic tasks, emulation tasks of time series, and prediction tasks. The concluding remarks are stated in "Concluding remarks" section.

System equations for Hopf physical reservoir computer

The equations of motion for the Hopf oscillator are[45]:

$$\begin{aligned} \dot{x} &= \left(\mu - (x^2 + y^2)\right)x - \omega_0 y + A\sin(\Omega t + \phi) \\ \dot{y} &= \left(\mu - (x^2 + y^2)\right)y + \omega_0 x \end{aligned} \tag{1}$$

For this Hopf oscillator, x and y are the first and second states, respectively, and the sinusoidal forcing is given by $A\sin(\Omega t + \phi)$. A list of the parameters is given in Table 1. The information is first encoded as an input, $u(t)$, which will depend on the benchmark task being performed. The mask is defined by white Gaussian noise as:

$$m(t) = \sigma\dot{W} + \beta \tag{2}$$

Here, σ is the noise amplitude, $\dot{W}$ is white Gaussian noise, and β is a positive bias. It should be noted that $\dot{W}$ does not exist, but its differential form, dW, does[46].

To send information to the PRC to be processed, an external forcing function that contains the information signal, $u(t)$, and the stochastic mask, $m(t)$, is constructed as:

$$f(t) = 1 + u(t)m(t) \tag{3}$$

This external forcing function is injected into both the amplitude of the sinusoidal forcing, A, and the parameter affecting the limit cycle radius, μ. Including this force, the equations for the Hopf PRC are written as:

$$\begin{aligned} \dot{x} &= \Big(\mu f(t) - (x^2 + y^2)\Big)x - \omega_0 y + Af(t)\sin(\Omega t + \phi) \\ \dot{y} &= \Big(\mu f(t) - (x^2 + y^2)\Big)y + \omega_0 x \end{aligned} \tag{4}$$

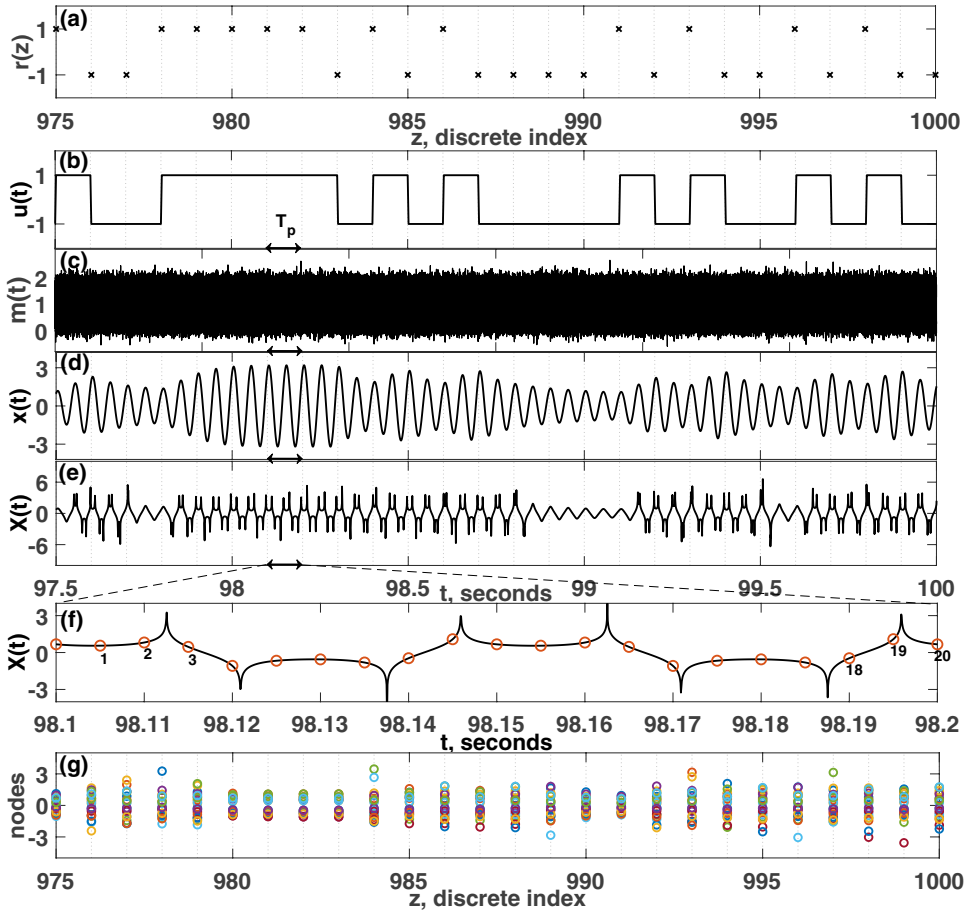

Figure 1. (**a**) Discrete random binary signal, $r(z)$. (**b**) Continuous input signal, $u(t)$. (**c**) Stochastic masking function, $m(t)$. (**d**) Time history of $x(t)$. (**e**) Rescaled time history, $X(t)$. (**f**) 20 equidistant nodes for a single pseudo-period, T_p, are denoted with circles. (**g**) Collected nodal states from the nodes for machine learning input data set. Different colors in (**g**) denote different nodes. For the simulation depicted here, the parameters were set such that: $\mu = 5$, $A = 0.5$, $\Omega = 40\pi$ rad/s, $\omega_0 = 40\pi$ rad/s, $T_p = 0.1$ seconds, $N = 20$ nodes, $\phi = \pi/3$ rad, $\sigma = 100$, $\beta = 1.0$.

Mapping methodology

To use the dynamics of the Hopf oscillator as a physical reservoir computer, the dynamics must first be mapped. To describe this mapping, an exclusive OR (XOR) logical task is used as an example. In this section, the Hopf PRC is simulated using an Euler–Maruyama scheme, since the mask is stochastic[42]. Shannon's information metric is used to quantify the performance of the reservoir when performing logical tasks, such as the XOR operation[11,43].

For this task, the binary "false" and "true" values are encoded as discrete negative ones and positive ones, respectively, in a discrete signal, $r(z)$. $r(z)$ is defined such that $z \in \mathbf{Z}^+$ and $r(z) \in \{-1, +1\}$, which is depicted in Fig. 1a. To input this into a continuous dynamical system, these values are first mapped to a continuous input function, $u(t)$, as follows:

$$u(t) = r(z) \text{ for } (n-1)T_p \leq t < (n)T_p, n \in \mathbf{Z}^+ \tag{5}$$

This function is depicted in Fig. 1b. T_p is a constant pseudo-period, in which $u(t)$ does not change its value. Thus, for the XOR logical task, the input function, $u(t) \in \{-1, +1\}$, is a random square wave with a pseudo-period, T_p. This implies that each of the "true" (e.g., +1) or "false" (e.g., −1) values affect the system for an amount of time, T_p. The mask function, $m(t)$, is depicted in Fig. 1c.

The Hopf PRC system described in Eq. (4) is numerically integrated using the Euler–Maruyama (EM) method, since the PRC is stochastic[42,47]. For these simulations, the integration time step, $dt = 10^{-5}$ seconds, the total simulation time in this case was $3000T_p = 300$ seconds, and $T_p = 0.1$ sec. This example simulation is shown in Fig. 1.

The time history of the x state obtained from the simulation is depicted in Fig. 1d. Next, $x(t)$ is re-scaled by subtracting the mean, μ_x, and dividing by the standard deviation, σ_x, using Eq. (6):

$$X = Re\left(tanh^{-1}\left(\frac{x-\mu_x}{\sigma_x}\right)\right) \tag{6}$$

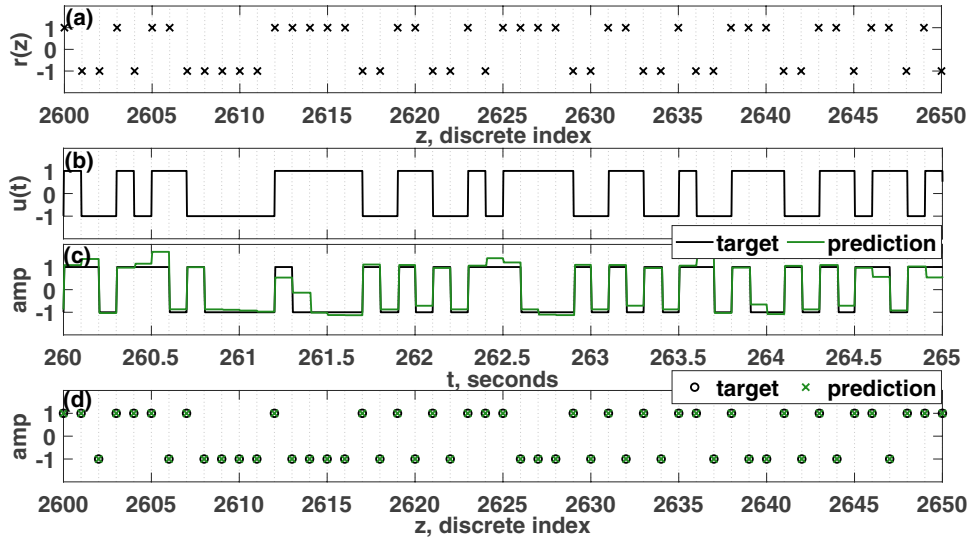

Figure 2. (**a**) Discrete random binary signal, $r(z)$. (**b**) Continuous input signal, $u(t)$. (**c**) target signal and continuous prediction. d) Discretized target and prediction. The calculated information metric is $R = 0.98$. For the simulation depicted here, the parameters were set such that: $\mu = 5$, $A = 0.5$, $\Omega = 40\pi$ rad/s, $\omega_0 = 40\pi$ rad/s, $T_p = 0.1$ s, $N = 20$, $\phi = \pi/3$ rad, $\sigma = 100$, $\beta = 1.0$.

In this equation, the inverse hyperbolic tangent function is used as a nonlinear activation function. Only the real part of $tanh^{-1}(\frac{x-\mu_x}{\sigma_x})$ is used for the subsequent steps. The time history of the X state is depicted in Fig. 1e.

Next, equidistant nodes are created by dividing each pseudo-period, T_p, equally into $N(= 20)$ nodes, as shown in Fig. 1f. Over each pseudo-period, T_p, the N node values are referred to as the nodal state, which is depicted in Fig. 1g.

The node matrix, S, is an $N \times K$ matrix; for this example, $N = 20$ is the number of nodes over a pseudo-period, and $K = 3000$ is the total number of pseudo-periods. Truncating the final 20% of this S matrix ($600T_p$), a new matrix, L ($480T_p$) is formed, which will be used in the training process. Throughout this paper, next, the reservoir computer is trained using ridge regression, as in Eq. (7):

$$\begin{aligned} w &= ML^T(LL^T + \lambda I)^{-1} \\ o(k) &= \sum_{i=1}^{N} w_i X_i(k) \end{aligned} \tag{7}$$

A target signal (the M vector) is created from the encoded input based on a benchmark task, which in this case is XOR task. For each pseudo-period, there will be one target value that is found by performing the XOR operation between the inputs, $r(z)$ and $r(z-1)$. In this way, the target vector, M, is found for the XOR task. Linear regression based training is then applied to the nodal state matrix, L, to map it to the desired output using Eq. (7). In Eq. (7), w is the weight vector found after training, I is the identity matrix, $\lambda = 10^{-1}$ is the regularization parameter used to avoid over-fitting, and $o(k)$ is the prediction of the reservoir computer at the kth pseudo-period. The discrete input, r(z) and continuous input, u(t) are given in Fig. 2a and b respectively. Figure 2c shows this prediction along with the corresponding target signal. In the final step, the prediction is binarized since XOR is a binary task, which is depicted in Fig. 2d. It should be noted that a nonlinear dynamic emulation task would not require this final step of discretization.

For a logical task, the efficacy of the reservoir computer is quantified using Shannon's *information rate*[48]. The information rate, R, can be defined as follows:

$$R = H(x) - H_y(x) \tag{8}$$

Here $H(x)$ is the *Shannon entropy*, which denotes how much information is encoded in a signal. This can be defined as follows:

$$H(x) = -\sum_i p_i \log_2(p_i) \tag{9}$$

In this equation, p_i is the probability of getting a particular bit, i. $H_y(x)$ is the *conditional entropy*, which denotes the probability of getting an incorrect bit in the target signal:

$$H_y(x) = -\sum_{i,j} p(i,j) \log_2(p_i(j)) \tag{10}$$

Here $p_i(j) = p(j|i) = \frac{p(i,j)}{\sum_j p(i,j)}$ and $p(i,j)$ is the joint probability distribution of the two variables, i and j, each of which can take a value of "1" or "− 1" for a logical task. i is a bit from the target, and j is a bit from the prediction.

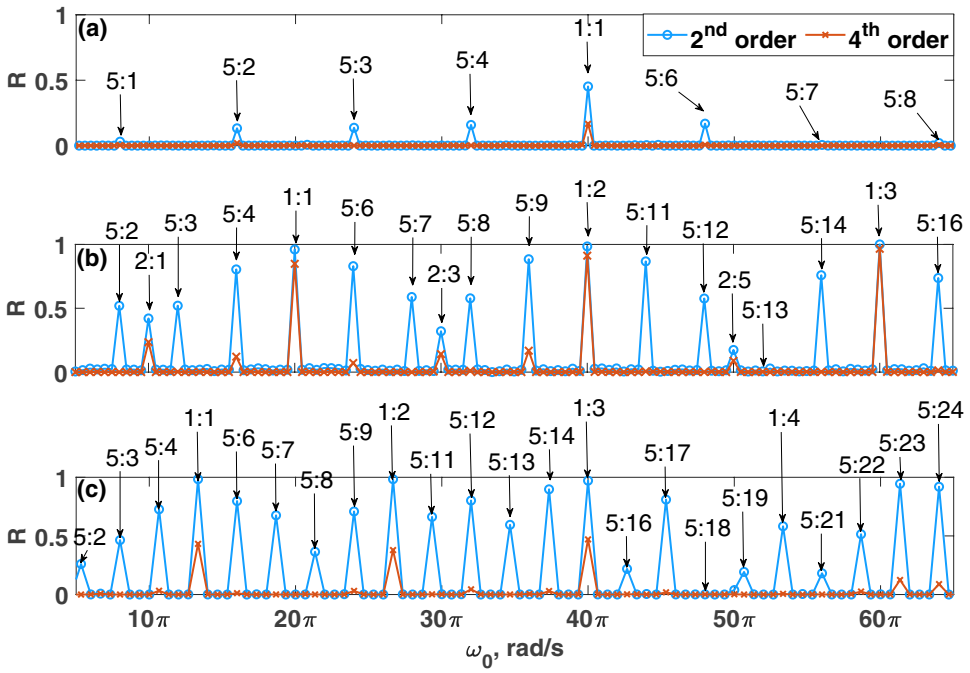

Figure 3. Comparison of the reservoir's computing performance, R, on the choice of the pseudo-period, T_p, and the natural frequency, ω_0 using 2nd and 4th order parity tasks. (**a**) $T_p = 0.05$ seconds, (**b**) $T_p = 0.1$ seconds, and (**c**) $T_p = 0.15$ seconds. Different ratios of the natural period and pseudo-period (e.g., $\frac{2\pi}{\omega_0} : T_p$) are simulated, and the ratios are depicted for peaks in the information metric. $\omega_0 = \Omega = 50\pi$ rad/s is the resonance case. Parameters were set such that: $\mu = 5$, $A = 0.5$, $\Omega = 50\pi$ rad/s, $N = 1000$ nodes, $\phi = \pi/3$ rad, $\sigma = 15$, $\beta = 1.0$.

The information rate, R, for this case was calculated to be 0.98 based on the prediction from the validation portion (not including in the training process). Due to the nature of this binary target signal, the Shannon entropy is 1.0, which marks the maximum value of the information rate for this task. It should be noted that the lower limit of R is zero, which would be achieved if every prediction was incorrect, while the upper limit of R depends on the task. For the parity tasks considered here, the upper limit of R is equal to one.

Pseudo-period and noise

In this section, the effects of pseudo-period and noise on the computational ability of the reservoir are explored. For this discussion, several parity tasks (defined in Eq. (12)) are used to understand the effects of the pseudo-period and noise on the computational ability of the reservoir.

The relationship between the pseudo-period, T_p, and the natural frequency of the oscillator, ω_0, is explored in Fig. 3 by using the 2nd and 4th order parity tasks. In Fig. 3a–c, the reservoir computer's performance is measured for three different values of T_p while varying the natural period, $\frac{2\pi}{\omega_0}$. It is found that the reservoir has better performance when the pseudo-period is an integer multiple of the natural period of the oscillator. The reservoir's performance is studied using both resonance ($\omega_0 = \Omega$) and non-resonance ($\omega_0 \neq \Omega$) conditions. It is found that both cases can result in strong or weak computational ability depending on the fractional relationship between the natural period and the pseudo-period. However, maintaining this design can still fail to make a robust reservoir computer when T_p is very low (e.g., $T_p = 0.05$ seconds). For the remainder of the paper, combinations of T_p and ω_0 are chosen such that the pseudo-period is an integer multiple of the natural period of the oscillator.

Noise is ubiquitous in physical systems. For this reason, noise is introduced into this system using a stochastic masking function. Figure 4 shows the relationship between the computational ability, as measured with R, the noise amplitude, σ, and the noise bias, β. The simulations presented in Fig. 4 are performed for the 4th order parity task (left) and 6th order parity task (right). The reservoir is found to be robust against a certain level of noise intensity, which demonstrates its potential to be implemented under the influence of environmental noise. However, increasing noise intensity does decrease the computational ability of the reservoir. This effect may be observed for a higher order task, which requires a longer memory (e.g., the 6th order parity task of Fig. 4). When $\beta = 0$, the computational ability was the lowest. Since the non-periodic noise mask with increasing noise intensity deteriorates the computational ability, it should be noted that the Hopf reservoir computer can also be built by excluding the noise mask ($\sigma = 0$).

Analog circuit experiment

To build a physical reservoir computer (PRC), an analog circuit implementation of Eq. (4) was designed, fabricated, and tested. The circuit's equations are given in Eq. (11):

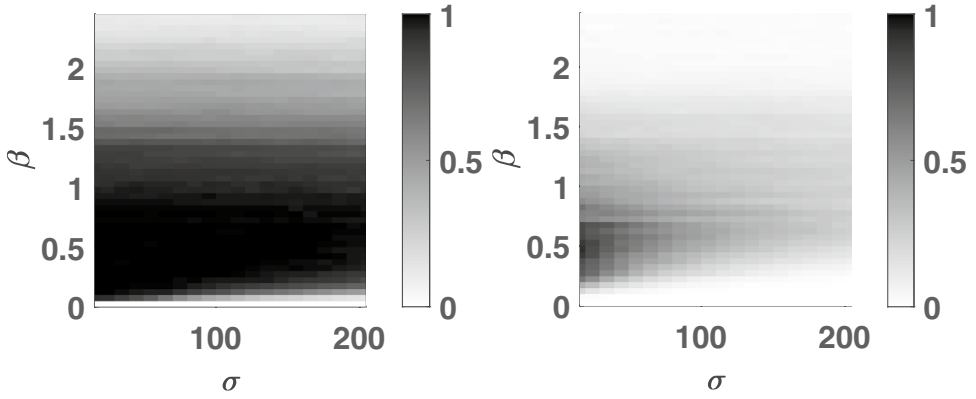

Figure 4. The reservoir computer is somewhat robust to noise. The effects of σ and β are shown. Left: 4th order parity task. Right: 6th order parity task. Parameters were set such that: $\mu = 5$, $A = 0.5$, $\Omega = 40\pi$ rad/s, $\omega_0 = 40\pi$ rad/s, $T_p = 0.1$ seconds, $N = 1000$ nodes, and $\phi = \pi/3$ rad.

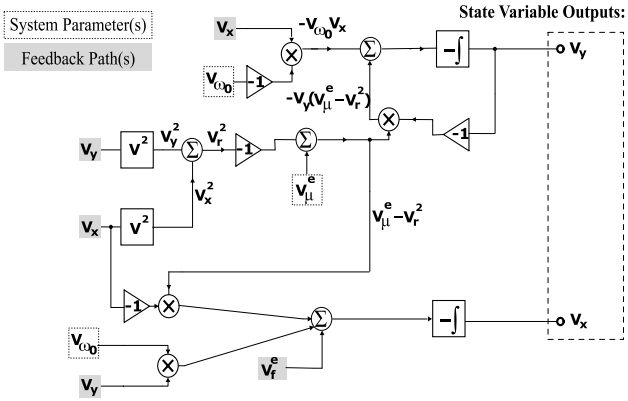

Figure 5. A simplified schematic for the Hopf PRC, with states V_x and V_y. $V_\mu^e = V_\mu(1 + V_u V_m)$ and $V_f^e = A(1 + V_u V_m)\sin(\Omega t + \phi)$.

$$\begin{aligned}\dot{V}_x &= -\tfrac{1}{R_1 C}\Big(V_\mu(1 + V_u V_m) - (V_x^2 + V_y^2)\Big)V_x + \tfrac{1}{R_1 C}V_{\omega_0}V_y - \tfrac{1}{R_2 C}\big(A(1 + V_u V_m)\sin(\Omega t + \phi)\big)\\ \dot{V}_y &= -\tfrac{1}{R_1 C}\Big(V_\mu(1 + V_u V_m) - (V_x^2 + V_y^2)\Big)V_y - \tfrac{1}{R_1 C}V_{\omega_0}V_x\end{aligned} \tag{11}$$

Here, V_u is the input voltage, V_m is the stochastic masking voltage, V_μ is the limit cycle radius voltage, V_{ω_0} is the resonance constant voltage. V_x and V_y are the states, which correspond to states x and y in Eq. (4). The circuit implementation used TL082 operational amplifiers and AD633 multipliers in standard integrator network configurations. The error tolerance is 1% for the resistors and 2% for the capacitors. The continuous input function, V_u, the stochastic masking function, V_m, and the sinusoidal forcing, $\sin(\Omega t + \phi)$, were created in MATLAB and sent to the circuit via a National Instrument (NI) cDAQ-9174. This cDAQ-9174 also collected the V_x and V_y states. A sampling frequency of 10^5 samples/s was used to collect data for all the experiments. The resistor values were chosen such that $R_1 = 10\,\text{k}\Omega$ and $R_2 = 100\text{k}\Omega$, and the capacitor values were chosen such that $C = 0.1\mu\text{F}$. A simplified schematic is shown in Fig. 5.

The V_x state will be treated in the same manner that the x state was treated in "Mapping methodology" section. That is, the V_x state will be rescaled using Eq. (6), and then the rescaled state will be used to form the nodal state matrix, L. The target signal vector, M, will be created following the same process discussed in "Mapping methodology" section. Finally, Eq. (7) will be used to train the PRC to map input data to the desired output values. As an example, the analog circuit Hopf PRC was used to solve the XOR task as in the previous section, which is depicted in Fig. 6. The information rate, R, for this case was calculated to be 1.0 based on the prediction from the validation portion (not including in the training process).

Benchmark tasks for Hopf PRC

The Hopf PRC is numerically and experimentally tested with three benchmark tasks: (1) *logic tasks*, (2) *emulation tasks of time series*, and (3) *prediction tasks*. Logic tasks include the fundamental logic gate tasks and parity tasks of different orders. Emulation tasks of time series will test the PRC's ability to reproduce nonlinear auto

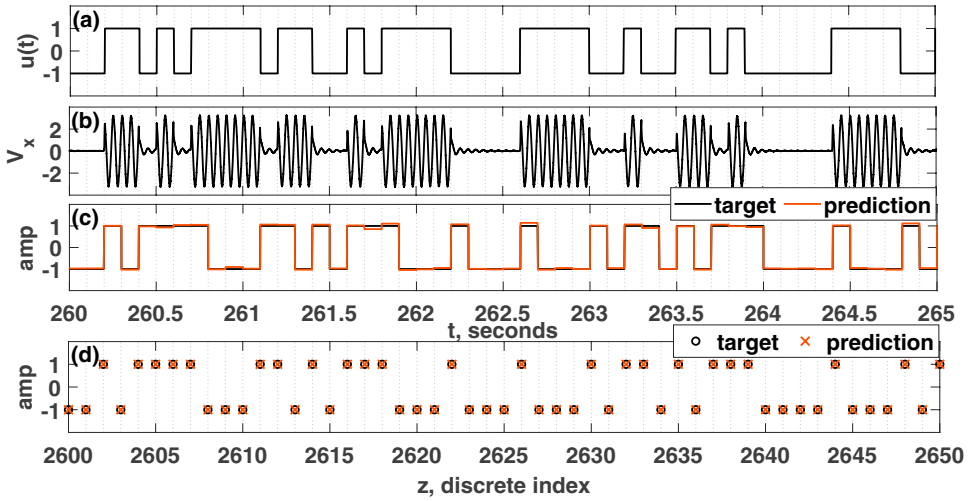

Figure 6. (**a**) Input voltage signal, V_u. (**b**) Time history of V_x. (**c**) XOR target signal, M, and the prediction. (**d**) Discretized prediction. The calculated information metric is $R = 1.0$. For the experimental results depicted here, the parameters were set such that: $V_\mu = 5$ volts, $A = 0.5$ volts, $\Omega = 40\pi$ rad/s, $V_{\omega_0} = 40\pi$ volts, $T_p = 0.1$ seconds, $N = 20$ nodes, $\phi = \pi/3$ rad, $\sigma = 10$ volts, $\beta = 1.0$ volts.

regressive moving average (NARMA) tasks of different orders. Prediction tasks include the Santa Fe time series and sunspot prediction tasks.

Logic benchmark tasks. *Parity tasks.* The computing efficacy of the reservoir is first evaluated with parity benchmark tasks. Since it is a logical task, the input function, $u(t)$, is generated with a random binary signal, $r(z)$, as discussed in "Mapping methodology" section. The nth order parity function, P_n, is defined by the following equation:

$$P_n(t) = \prod_{i=0}^{n-1} u(t - iT_p) \tag{12}$$

As n increases, this task will require more memory and nonlinearity from the reservoir. As given in "Mapping methodology" section, Shannon's information metric is used to measure the performance of the PRC for logic tasks. For $n = 1$, the first-order task does not require any memory from the input of the previous pseudo-period, so the task is linear. For $n > 1$, the task is nonlinear, which demands that the reservoir computer must also possess memory and the nonlinear separation ability. In Fig. 7, the ability of the Hopf PRC to follow parity tasks of 2nd to 5th order, both experimentally and in simulations. The initial $4000T_p = 400$ seconds are used for training, and the final $1000T_p = 100$ seconds are used for testing. The performance difference between the PRC experiment and the simulation could be due to the presence of nonlinear circuit components in the analog circuit, which are not represented in Eq. (11). For instance, V_u must jump between -1 and $+1$, but this instantaneous change takes a finite amount of time in the circuit.

Fundamental logic gate tasks. The computing performance of the reservoir is also assessed with fundamental logic gates: NOT ($\neg$), AND ($\wedge$), and OR ($\vee$). The input function, $u(t)$, is generated with a random binary signal as discussed in "Mapping methodology" section, and the Shannon's information metric is used again to measure the performance of this PRC. Figure 8 depicts the response of the Hopf PRC acting as fundamental logic gates, both experimentally and in simulations. In all cases, the Hopf PRC achieved an information rate that was maximal.

Emulation tasks. The reservoir is also evaluated with emulation tasks. The *nonlinear auto-regressive moving average* (NARMA) time series is used to test whether the reservoir possesses adequate nonlinearity and long time lags[4,24,26,28]. These tasks show the multi-tasking capability of the reservoir. NARMA tasks from the 2nd to 20th orders are used to test the reservoir. A NARMA task of order n is given in Eq. (13), where the initial target values are set to 0.19:

$$\begin{aligned} M_2(j+1) &= 0.4M_2(j) + 0.4M_2(j)M_2(j-1) + 0.6u^3(j\Delta t) + 0.1 \\ M_n(j+1) &= \alpha M_n(j) + \zeta M_n(j)\left(\sum_{i=0}^{n-1} M_n(j-i)\right) + \gamma u\Big(\big(j-(n-1)\big)\Delta t\Big)u(j\Delta t) + \delta \\ u(t) &= 0.2\sin\left(2\pi f_1 t\right)\sin\left(2\pi f_2 t\right)\sin\left(2\pi f_3 t\right) \end{aligned} \tag{13}$$

In Eq. (13), M_n is the target of the system. n is the order of NARMA task, $(f_1, f_2, f_3) = (\frac{2.11}{500}, \frac{3.73}{500}, \frac{4.33}{500})$, and $(\alpha, \zeta, \gamma, \delta) = (0.3, 0.05, 1.5, 0.1)$[26,28]. $u(t)$ is the continuous input that is used to force the Hopf PRC, which is

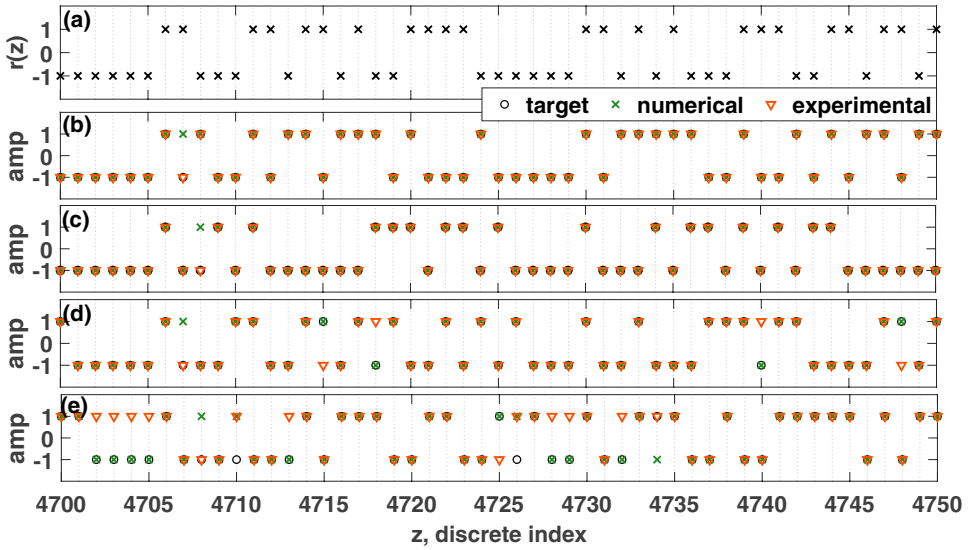

Figure 7. Comparison of the performance of the PRC for parity tasks. (**a**) Discrete input function, $r(z)$. (**b**) 2nd order parity task. Information metric: $R_{exp} = 1.00$, $R_{sim} = 0.98$. (**c**) 3rd order parity task. Information metric: $R_{exp} = 1.00$, $R_{sim} = 0.98$. (**d**) 4th order parity task. Information metric: $R_{exp} = 0.68$, $R_{sim} = 0.93$. (**e**) 5th order parity task. Information metric: $R_{exp} = 0.31$, $R_{sim} = 0.74$. Parameters were set such that: $V_\mu = \mu = 5$, $A = 0.5$, $\Omega = 40\pi$ rad/s, $V_{\omega_0} = \omega_0 = 40\pi$, $T_p = 0.1$ seconds, $N = 1000$ nodes, $\phi = \pi/3$ rad, $\sigma = 15$, $\beta = 1.0$, and a total time of $5000T_p = 500$ seconds (only a portion of the discrete prediction is shown).

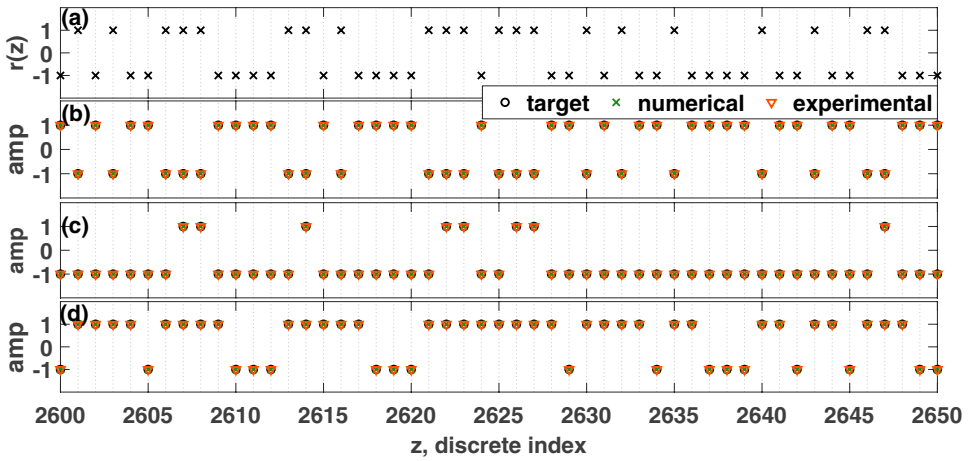

Figure 8. Comparison of the performance of the PRC for parity tasks. (**a**) Input function, $u(t)$. (**b**) NOT ($\neg$) gate. (**c**) AND ($\wedge$) gate, (**d**) OR ($\vee$) gate. For all numerical and experimental results, the information rate was at the theoretical maximum; the Hopf PRC can act as any of the fundamental logic gates. Parameters were set such that: $V_\mu = \mu = 5$, $A = 0.5$, $\Omega = 40\pi$ rad/s, $V_{\omega_0} = \omega_0 = 40\pi$ rad/s, $T_p = 0.1$ seconds, $N = 1000$ nodes, $\phi = \pi/3$ rad, $\sigma = 15$, $\beta = 1.0$, and a total time of $3000T_p = 300$ seconds. Only a portion of the response is shown here.

a function of a three sinusoidal functions. It should be noted that this formulation of the NARMA emulation task is non-standard. The $u(t)$ given in Eq. (13) was used for other dynamic systems in which inertia played a large role[4,26]. Similarly, this non-standard NARMA task is used here to evaluate this analog circuit reservoir. The reservoir emulates this nonlinear function, but it should be noted that the correlation present in Eq. (13) does not allow a definitive evaluation of the long-term memory characteristics of this reservoir.

In the simulations and experiments, $\Delta t = 0.1$ seconds, and the sampling rate was 10^5 samples/second. Figure 9 shows several NARMA tasks. Instead of the information rate, the *normalised mean square error* (NMSE) is used to evaluate the performance of the reservoir computer for the NARMA tasks:

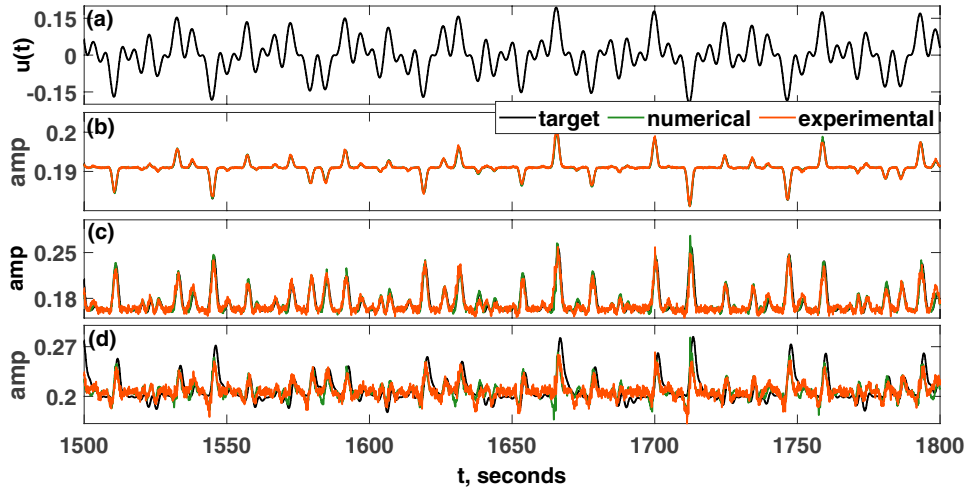

Figure 9. Comparison of the performance of the PRC for NARMA tasks. (**a**) Input function, $u(t)$. (**b**) 2nd order NARMA task. Performance metric: $NMSE_{exp} = 2.8181 \times 10^{-6}$, $NMSE_{sim} = 7.8199 \times 10^{-7}$. (**c**) 10th order NARMA task. $NMSE_{exp} = 0.0037$, $NMSE_{sim} = 5.0362 \times 10^{-4}$. (**d**) 20th order NARMA task. $NMSE_{exp} = 0.0060$, $NMSE_{sim} = 0.0033$. Only a portion of the result is shown in each figure. Parameters were set such that: $V_\mu = \mu = 5$, $A = 0.5$, $\Omega = 40\pi$ rad/s, $V_{\omega_0} = \omega_0 = 40\pi$, $T_p = 0.1$ seconds, $N = 1000$ nodes, $\phi = \pi/3$ rad, $\sigma = 15$, $\beta = 1.0$.

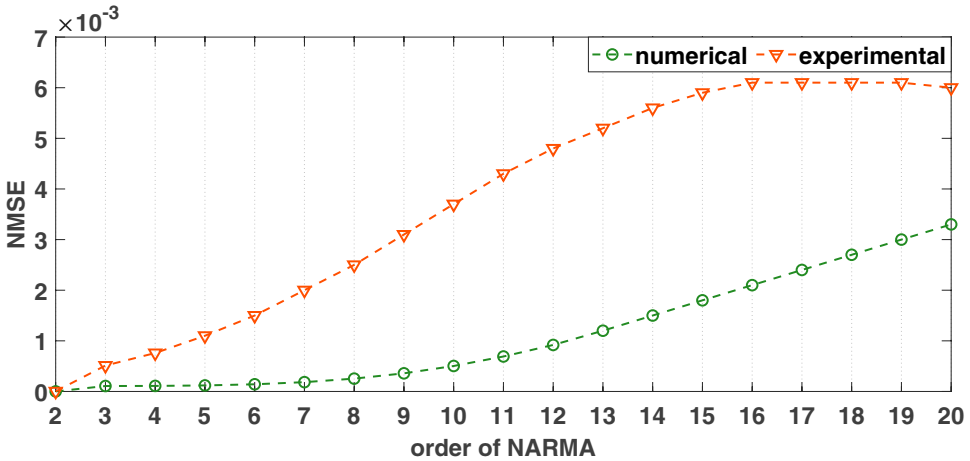

Figure 10. Plot of the NMSE of the 2nd to 20th order NARMA tasks for the simulation and experiment. Parameters were set such that: $V_\mu = \mu = 5$, $A = 0.5$, $\Omega = 40\pi$ rad/s, $V_{\omega_0} = \omega_0 = 40\pi$, $T_p = 0.1$ seconds, $N = 1000$, $\phi = \pi/3$ rad, $\sigma = 15$, $\beta = 1.0$.

$$NMSE = \frac{\sum_{j=j_0}^{j_f} \left(M_n(j+1) - o_n(j+1) \right)^2}{\sum_{j=j_0}^{j_f} M_n^2(j+1)} \tag{14}$$

The final 20% of the target signal (16,000–20,000 pseudo-periods) is used for the validation. M_n is the target, and o_n is the prediction from the reservoir computer. In Eq. (14), j_0 is the starting time step, and j_f is the ending time step from the test section. From the plots in Figs. 9 and 10, the numerical simulations of the Hopf PRC show superior performance as compared to the experiment. However, both have an acceptable performance until the 20th order task. The PRC can perform much higher order NARMA tasks than the order of the parity tasks.

Prediction tasks. *Santa Fe task.* Time series forecasting is an important benchmark for a reservoir. The Santa Fe time series was first used in a time series forecasting competition as a benchmark test. The Santa Fe time series data set *A* is a univariate time series found from the recorded intensity of a chaotic far-infrared-laser[49]. The target signal is generated to predict the value at the next time step based on the values of the current and previous time steps. Figure 11a shows the Hopf PRC's performance on this laser time series, for both the experiment and the numerical simulations. NMSE is used as the performance metric.

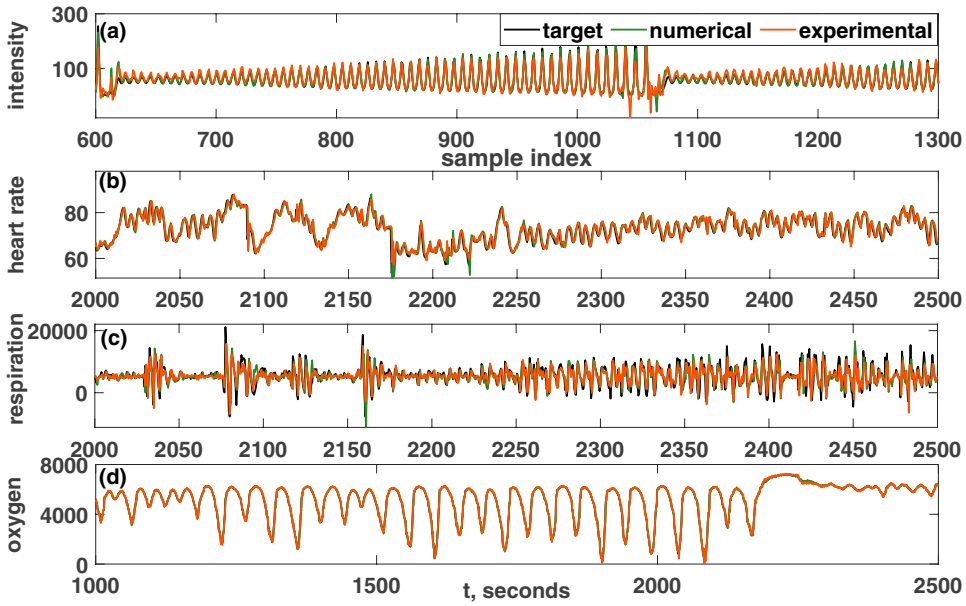

Figure 11. Comparison of the performance of the PRC for Santa Fe prediction tasks. (**a**) The Santa Fe chaotic time series of a laser intensity prediction task. Performance metric: $NMSE_{exp} = 0.0615$, $NMSE_{sim} = 0.02$. (**b**) The Santa Fe heart rate prediction task. $NMSE_{exp} = 6.0258 \times 10^{-4}$, $NMSE_{sim} = 6.5060 \times 10^{-4}$. (**c**) Santa Fe respiration force prediction task. $NMSE_{exp} = 0.1826$, $NMSE_{sim} = 0.1753$. (**d**) Santa Fe blood oxygen concentration prediction task. $NMSE_{exp} = 3.3287 \times 10^{-4}$, $NMSE_{sim} = 1.7 \times 10^{-4}$. Parameters were set such that: $V_\mu = \mu = 5$ volts, $A = 0.5$ volts, $\Omega = 40\pi$ rad/s, $V_{\omega_0} = \omega_0 = 40\pi$ volts, $T_p = 0.1$ seconds, $N = 1000$ nodes, $\phi = \pi/3$ rad, $\sigma = 15$ volts, $\beta = 1.0$ volts. Only a portion of the result is shown in each figure.

Santa Fe time series data set *B* is a multivariate time series found from the sleep laboratory of the Beth Israel Hospital (current name: Beth Israel Deaconess Medical Center) in Boston, Massachusetts[50,51]. This data set was taken from the MIT-BIH Polysomnographic Database record (slp60) and submitted to the Santa Fe Time Series Competition in 1991[52]. The heart rate, chest volume (respiration force), and blood oxygen concentration comprise the target.

For each of these time series, the target signal is again generated to predict the next step based on the values of the current and previous time steps. In each case, the original time series is normalized to use as the input. Figure 11b–d shows the reservoir computer's performance in predicting subsequent values of the heart rate, respiratory force, and blood oxygen concentration, respectively, through both experiments and numerical simulations. The NMSE is calculated in each case to evaluate the performance of the reservoir.

Sunspot prediction task. The prediction of the total number of sunspots (S_n) is also a one-step time series prediction task similar to the Santa Fe time series[10]. Daily and monthly total sunspot numbers were used in one step forecasting purpose by the reservoir computer. The necessary data set is taken from *WDC-SILSO*, Royal Observatory of Belgium, Brussels[53]. Again, for each of the time series, the target signal is generated to predict the next value based on the value of the current and previous time steps, and the original time series is normalized to use as the input to the oscillator. Figure 12 (top) shows the reservoir's performance in predicting the next steps of the daily total counted sunspots, and Fig. 12 (bottom) shows the performance in predicting monthly counted sunspots. Again, the NMSE is used to evaluate the reservoir's efficacy for this task.

Concluding remarks

In this paper, the Hopf oscillator is explored as a physical reservoir computer through employing a time-multiplexed, node-based architecture with a stochastic masking function. Discarding the regularly used delay lines, this Hopf PRC is a simple and cheap method for creating a physical reservoir computer. Since quantum systems are capable of limit cycle motion[54], this Hopf PRC formulation might be applicable for quantum PRCs. The Euler–Maruyama method was used for the numerical simulations of this Hopf PRC. An analog circuit of this Hopf PRC was developed, fabricated, and tested. The Hopf PRC was found to possess multi-tasking capability, since it was shown to perform logic operations, emulation tasks, and time series prediction tasks. Taking inspiration from adaptive oscillators, the input signal was injected into multiple locations, including the parameter that affects the limit cycle radius and the amplitude of the sinusoidal forcing. Additionally, the masking function used in this PRC is stochastic. Since this PRC architecture is tested with noise, it also suggests that this reservoir computer should be robust to environmental noises in practical implementations.

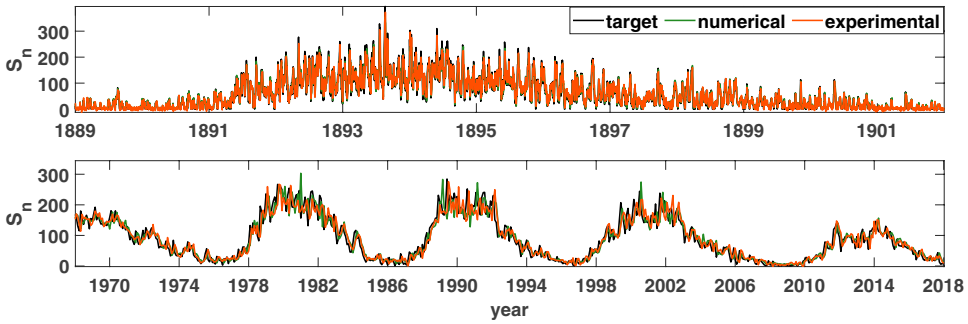

Figure 12. Comparison of the performance the sunspot prediction (S_n) task. Top: Daily total number of sunspot prediction task, for both the experiment and the numerical simulations. Performance metric: $NMSE_{exp} = 0.0548$, $NMSE_{sim} = 0.0534$. Bottom: Monthly mean total number of sunspot prediction task. Performance metric: $NMSE_{exp} = 0.0595$, $NMSE_{sim} = 0.0455$. Parameters were set such that: $V_\mu = \mu = 5$ volts, $A = 0.5$ volts, $\Omega = 40\pi$ rad/s, $V_{\omega_0} = \omega_0 = 40\pi$ rad/s, $N = 1000$ nodes, $\phi = \pi/3$ rad, $\sigma = 15$ volts, $\beta = 1.0$ volts. Only a portion of the result is shown in each figure.

Received: 21 July 2021; Accepted: 17 September 2021
Published online: 30 September 2021

References

1. Jaeger, H. & Haas, H. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science* **304**, 78–80 (2004).
2. Lukoševičius, M. & Jaeger, H. Reservoir computing approaches to recurrent neural network training. *Comput. Sci. Rev.* **3**, 127–149 (2009).
3. Nakajima, K. Physical reservoir computing-an introductory perspective. *Jpn. J. Appl. Phys.* **59**, 060501 (2020).
4. Nakajima, K., Hauser, H., Li, T. & Pfeifer, R. Information processing via physical soft body. *Sci. Rep.* **5**, 10487 (2015).
5. Haynes, N. D., Soriano, M. C., Rosin, D. P., Fischer, I. & Gauthier, D. J. Reservoir computing with a single time-delay autonomous boolean node. *Phys. Rev. E* **91**, 020801 (2015).
6. Appeltant, L. *et al.* Information processing using a single dynamical node as complex system. *Nat. Commun.* **2**, 1–6 (2011).
7. Werbos, P. J. Backpropagation through time: What it does and how to do it. *Proc. IEEE* **78**, 1550–1560 (1990).
8. Jaeger, H. The echo state approach to analysing and training recurrent neural networks-with an erratum note. *Bonn Ger. German Natl. Res. Center Inf. Technol. GMD Tech. Rep.* **148**, 13 (2001).
9. Natschläger, T., Maass, W. & Markram, H. The liquid computer: A novel strategy for real-time computing on time series. *Spec. Issue Found. Inf. Process. TELEMATIK* **8**, 39–43 (2002).
10. Appeltant, L. *et al. Reservoir computing based on delay-dynamical systems* (Vrije Universiteit Brussel/Universitat de les Illes Balears, These de Doctorat, 2012).
11. Shougat, M. R. E. U., Li, X., Mollik, T. & Perkins, E. An information theoretic study of a duffing oscillator array reservoir computer. *J. Comput. Nonlinear Dyn.* **16**, 081004 (2021).
12. Dion, G., Mejaouri, S. & Sylvestre, J. Reservoir computing with a single delay-coupled non-linear mechanical oscillator. *J. Appl. Phys.* **124**, 152132 (2018).
13. Laporte, F., Dambre, J. & Bienstman, P. Simulating self-learning in photorefractive optical reservoir computers. *Sci. Rep.* **11**, 1–10 (2021).
14. Du, C. *et al.* Reservoir computing using dynamic memristors for temporal information processing. *Nat. Commun.* **8**, 1–10 (2017).
15. Inubushi, M. & Yoshimura, K. Reservoir computing beyond memory-nonlinearity trade-off. *Sci. Rep.* **7**, 1–10 (2017).
16. Wang, R., Kalnay, E. & Balachandran, B. Neural machine-based forecasting of chaotic dynamics. *Nonlinear Dyn.* **98**, 2903–2917 (2019).
17. Pathak, J., Hunt, B., Girvan, M., Lu, Z. & Ott, E. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Phys. Rev. Lett.* **120**, 024102 (2018).
18. Rafayelyan, M., Dong, J., Tan, Y., Krzakala, F. & Gigan, S. Large-scale optical reservoir computing for spatiotemporal chaotic systems prediction. *Phys. Rev. X* **10**, 041037 (2020).
19. Borlenghi, S., Boman, M. & Delin, A. Modeling reservoir computing with the discrete nonlinear schrödinger equation. *Phys. Rev. E* **98**, 052101 (2018).
20. Urbain, G., Degrave, J., Carette, B., Dambre, J. & Wyffels, F. Morphological properties of mass-spring networks for optimal locomotion learning. *Front. Neurorobot.* **11**, 16 (2017).
21. Caluwaerts, K., DHaene, M., Verstraeten, D. & Schrauwen, B. Locomotion without a brain: Physical reservoir computing in tensegrity structures. *Artif. Life* **19**, 35–66 (2013).
22. Coulombe, J. C., York, M. C. & Sylvestre, J. Computing with networks of nonlinear mechanical oscillators. *PLoS ONE* **12**, e0178663 (2017).
23. Zheng, T. *et al.* Parameters optimization method for the time-delayed reservoir computing with a nonlinear duffing mechanical oscillator. *Sci. Rep.* **11**, 1–11 (2021).
24. Hauser, H., Ijspeert, A. J., Füchslin, R. M., Pfeifer, R. & Maass, W. Towards a theoretical foundation for morphological computation with compliant bodies. *Biol. Cybern.* **105**, 355–370 (2011).
25. Nakajima, K. *et al.* Computing with a muscular-hydrostat system. In *2013 IEEE International Conference on Robotics and Automation*, 1504–1511 (IEEE, 2013).
26. Nakajima, K. *et al.* A soft body as a reservoir: Case studies in a dynamic model of octopus-inspired soft robotic arm. *Front. Comput. Neurosci.* **7**, 91 (2013).
27. Caluwaerts, K. *et al.* Design and control of compliant tensegrity robots through simulation and hardware validation. *J. R. Soc. Interface* **11**, 20140520 (2014).
28. Bhovad, P. & Li, S. Physical reservoir computing with origami and its application to robotic crawling. *Sci. Rep.* **11**, 1–18 (2021).
29. Bhovad, P. & Li, S. Physical reservoir computing with origami: a feasibility study. In *Behavior and Mechanics of Multifunctional Materials XV*, vol. 11589, 1158903 (International Society for Optics and Photonics, 2021).

30. Fujii, K. & Nakajima, K. Harnessing disordered-ensemble quantum dynamics for machine learning. *Phys. Rev. Appl.* **8**, 024030 (2017).
31. Govia, L., Ribeill, G., Rowlands, G., Krovi, H. & Ohki, T. Quantum reservoir computing with a single nonlinear oscillator. *Phys. Rev. Res.* **3**, 013077 (2021).
32. Nakajima, K., Fujii, K., Negoro, M., Mitarai, K. & Kitagawa, M. Boosting computational power through spatial multiplexing in quantum reservoir computing. *Phys. Rev. Appl.* **11**, 034021 (2019).
33. Chen, J., Nurdin, H. I. & Yamamoto, N. Temporal information processing on noisy quantum computers. *Phys. Rev. Appl.* **14**, 024065 (2020).
34. Larger, L. *et al.* Photonic information processing beyond turing: An optoelectronic implementation of reservoir computing. *Opt. Express* **20**, 3241–3249 (2012).
35. Vinckier, Q. *et al.* High-performance photonic reservoir computer based on a coherently driven passive cavity. *Optica* **2**, 438–446 (2015).
36. Grigoryeva, L., Henriques, J., Larger, L. & Ortega, J.-P. Stochastic nonlinear time series forecasting using time-delay reservoir computers: Performance and universality. *Neural Netw.* **55**, 59–71 (2014).
37. Argyris, A., Schwind, J. & Fischer, I. Fast physical repetitive patterns generation for masking in time-delay reservoir computing. *Sci. Rep.* **11**, 1–12 (2021).
38. Marković, D. *et al.* Reservoir computing with the frequency, phase, and amplitude of spin-torque nano-oscillators. *Appl. Phys. Lett.* **114**, 012409 (2019).
39. Li, X. *et al.* A four-state adaptive Hopf oscillator. *PLoS ONE* **16**, e0249131 (2021).
40. Li, X. *et al.* Stochastic effects on a Hopf adaptive frequency oscillator. *J. Appl. Phys.* **129**, 224901 (2021).
41. Nakayama, J., Kanno, K. & Uchida, A. Laser dynamical reservoir computing with consistency: An approach of a chaos mask signal. *Opt. Express* **24**, 8679–8692 (2016).
42. Higham, D. J. An algorithmic introduction to numerical simulation of stochastic differential equations. *SIAM Rev.* **43**, 525–546 (2001).
43. Perkins, E. & Balachandran, B. Effects of phase lag on the information rate of a bistable duffing oscillator. *Phys. Lett. A* **379**, 308–313 (2015).
44. Nguimdo, R. M., Verschaffelt, G., Danckaert, J. & Van der Sande, G. Simultaneous computation of two independent tasks using reservoir computing based on a single photonic nonlinear node with optical feedback. *IEEE Trans. Neural Netw. Learn. Syst.* **26**, 3301–3307. https://doi.org/10.1109/TNNLS.2015.2404346 (2015).
45. Nayfeh, A. H. & Balachandran, B. *Applied Nonlinear Dynamics: Analytical, Computational, and Experimental Methods* (John Wiley & Sons, 2008).
46. Chorin, A. J. & Hald, O. H. Brownian motion. In *Stochastic Tools in Mathematics and Science*, 47–81 (Springer, 2009).
47. Perkins, E. Effects of noise on the frequency response of the monostable duffing oscillator. *Phys. Lett. A* **381**, 1009–1013 (2017).
48. Shannon, C. E. A mathematical theory of communication. *Bell Syst. Tech. J.* **27**, 379–423 (1948).
49. Gershenfeld, N. A. & Weigend, A. S. *The Future of Time Series: Learning and Understanding* (CRC Press, 2018).
50. Rigney, D. R. Multichannel physiological data description and analysis. *Time Ser. Predict.* (1994).
51. Goldberger, A. L. *et al.* Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. *Circulation* **101**, e215–e220 (2000).
52. Ichimaru, Y. & Moody, G. Development of the polysomnographic database on cd-rom. *Psychiatry Clin. Neurosci.* **53**, 175–177 (1999).
53. SILSO World Data Center. The international sunspot number. *International Sunspot Number Monthly Bulletin and online catalogue* (1818–2018).
54. Arosh, L. B., Cross, M. & Lifshitz, R. Quantum limit cycles and the Rayleigh and van der Pol oscillators. *Phys. Rev. Res.* **3**, 013130 (2021).

Acknowledgements

Partial support for this project from DARPA's Young Faculty Award is greatly appreciated. Research was sponsored by the Army Research Office and was accomplished under Grant No. W911NF-20-1-0336. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

Author contributions

E.P. conceived the numerical and physical experiment(s) and guided process; M.S. and X.L. built the circuit, M.S., X.L., and T.M. wrote programs and simulated the experiment(s); E.P. and M.S. analyzed the results. All authors reviewed the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to M.R.E.U.S.