Machine learning prediction of critical transition and system collapse

Ling-Wei Kong,[1] Hua-Wei Fan,[2] Celso Grebogi,[3] and Ying-Cheng Lai[1,4,*]

[1]*School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, Arizona 85287, USA*
[2]*School of Physics and Information Technology, Shaanxi Normal University, Xi'an 710062, China*
[3]*Institute for Complex Systems and Mathematical Biology, School of Natural and Computing Sciences, King's College, University of Aberdeen, Aberdeen, United Kingdom*
[4]*Department of Physics, Arizona State University, Tempe, Arizona 85287, USA*

(Received 28 May 2020; accepted 14 January 2021; published 28 January 2021)

To predict a critical transition due to parameter drift without relying on a model is an outstanding problem in nonlinear dynamics and applied fields. A closely related problem is to predict whether the system is already in or if the system will be in a transient state preceding its collapse. We develop a model-free, machine-learning-based solution to both problems by exploiting reservoir computing to incorporate a parameter input channel. We demonstrate that, when the machine is trained in the normal functioning regime with a chaotic attractor (i.e., before the critical transition), the transition point can be predicted accurately. Remarkably, for a parameter drift through the critical point, the machine with the input parameter channel is able to predict not only that the system will be in a transient state, but also the distribution of the transient lifetimes and their average before the final collapse, revealing an important physical property of transient chaos.

DOI: 10.1103/PhysRevResearch.3.013090

I. INTRODUCTION

In nonlinear and complex dynamical systems, a catastrophic collapse is often preceded by transient chaos. For example, in electrical power systems, voltage collapse can occur after the system enters into the state of transient chaos [1]. In ecology, slow parameter drift caused by environmental deterioration can induce a transition into transient chaos, after which species extinction follows [2,3]. A common route to transient chaos is a global bifurcation termed crisis [4,5], at which a chaotic attractor collides with its own basin boundary, is destroyed, and becomes a chaotic transient. In the real world, the accurate system equations are often unknown and only measured time series are available. Model-free and data-driven prediction of the critical transition and system collapse in advance of its occurrence, while the system is currently operating in a normal regime with a chaotic attractor, has been an outstanding problem. If the underlying equations of the system have a sparse representation in some suitable mathematical basis, then sparse optimization methods such as compressive sensing can be exploited to find the system equations [6,7] and consequently to predict the critical transition.

A closely related problem is to determine if the system is already in a transient state: the question "How do you know you are in a transient?" In nonlinear dynamics, this is one of the most difficult questions because the underlying system can be in a long transient in which all measurable physical quantities exhibit essentially the same behaviors as if the system were still in a sustained state with a chaotic attractor. Applying the traditional method of delay coordinate embedding [8] to such a case would yield estimates of dynamical invariants such as the Lyapunov exponents and fractal dimensions, but it would give no indication that the system is already in a transient and so an eventual collapse is inevitable. Developing a model-free, purely data-driven predictive paradigm to address this problem is of considerable value to solving some of the most pressing problems in modern times. Due to global warming and climate change, some natural systems may have already been in a transient state awaiting a catastrophic collapse to occur. A reliable determination at the present that the system has already passed the critical transition or a "tipping" point to a transient state would send a clear message to policy makers and the general public that actions must be taken immediately to avoid the otherwise inevitable catastrophic collapse.

In recent years, machine learning techniques have been proven useful for many tasks in the field of nonlinear dynamics. A research area based on reservoir computing, a class of recurrent neural networks [9–12], has gained considerable momentum as a powerful paradigm for model-free prediction of nonlinear and chaotic dynamical systems [13–29]. There have also been efforts in reconstructing the bifurcation diagrams of nonlinear dynamical systems using machine learning [30–32].

In this paper, we develop a machine learning framework based on reservoir computing to predict critical transition and transient chaos in nonlinear dynamical systems. Our articulated reservoir computing structure differs from the

*ying-cheng.lai@asu.edu

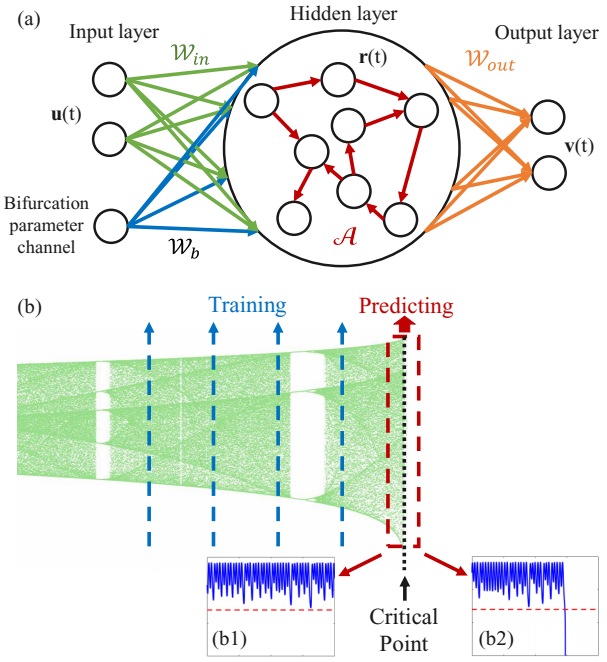

FIG. 1. Modified reservoir computing scheme and illustration of the idea to predict critical transition and collapse. (a) Structure of the modified reservoir computing machine with an input parameter channel. The system consists of three layers: the input layer, the hidden layer, and the output layer. Vectors $\mathbf{u}(t)$, $\mathbf{r}(t)$, and $\mathbf{v}(t)$ denote the set of input signals (measured time series from the dynamical variables of the target system), the dynamical state of the reservoir network (hidden layer), and the set of output signals representing the machine's prediction. The bifurcation parameter is designated as an additional input. (b) The reservoir computing machine is trained with time series taken from a few parameter values in the green regime in which the system operates normally, indicated by the three vertical blue dashed lines. After training, prediction can be made either for parameter values before the critical point (still in the green regime) or after, where in the former, the machine will predict that the system is safe (b1) but, importantly, for the latter, the machine will predict that the system will eventually collapse after transient chaos (b2).

conventional one in that we designate an additional input channel for the bifurcation parameter, as shown in Fig. 1(a). The basic idea of our framework is explained in Fig. 1(b), a schematic bifurcation diagram of a typical nonlinear system. In the green region, there is a chaotic attractor together with periodic windows, in which the system functioning is normal. A catastrophic bifurcation occurs at the critical parameter value defining the end of the green region, where the chaotic attractor is destroyed through a crisis transition. For a parameter value slightly beyond the critical point, there is transient chaos leading to collapse. Suppose that, currently, the system operates in the normal regime. Given a certain amount of parameter drift, the two goals are (i) to predict the transition point so as to determine whether the system will be in a transient chaotic regime heading to collapse and (ii) if yes, on average how long the system could survive, i.e., to predict the average lifetime of the chaotic transient. Because time-series data from multiple parameter values are needed, it is necessary to specify the parameter value at which the data are taken—a task that can be accomplished by inputting the parameter value to all nodes in the underlying neural network. We demonstrate that our proposed machine learning framework can accomplish the two goals with three examples: a three-species food chain model [2] in ecology in which a catastrophic transition leads to transient chaos and then species extinction, an electrical power system susceptible to voltage collapse through transient chaos [1], and the Kuramoto-Sivashinsky system [33,34] in the regime of transient spatiotemporal chaos [35]. We show that, training a reservoir network of reasonable size, e.g., 1000 nodes, with time-series data taken from three parameter values in the normal chaotic regime, the machine is able to predict not only the collapse point but also transient chaos for parameter values beyond the critical point. A remarkable feature is that, after the critical transition, the probability distribution of the transient lifetimes of the machine generated trajectories from an ensemble of initial conditions agrees with the true distribution, indicating that, for a given parameter drift into the transient chaos regime, the machine is able to predict, on average, how long the system can "survive" before collapse. Thus, not only can the machine predict that the system does exhibit a critical transition, it is also capable of revealing some basic physics about the system, i.e., the nature of transient chaos as characterized by the lifetime distribution.

II. RESERVOIR COMPUTING WITH AN ADDITIONAL PARAMETER CHANNEL

A reservoir computing system consists of three layers: the input layer, the hidden layer, and the output layer. As shown in Fig. 1(a), the input layer maps the low-dimensional time-series data $\mathbf{u}(t)$ into the high-dimensional hidden state $\mathbf{r}(t)$ through a matrix $\mathcal{W}_{\text{in}}$, and the output layer maps $\mathbf{r}(t)$ back into the low-dimensional time series $\mathbf{v}(t)$ through another matrix $\mathcal{W}_{\text{out}}$. An additional input parameter channel is connected to every node of the reservoir network via the matrix $\mathcal{W}_b$. For simplicity, we consider a catastrophic transition caused by variations in a single parameter. The reservoir network adjacency matrix $\mathcal{A}$ transfers information from the hidden states $\mathbf{r}(t)$ at t to the next time step $\mathbf{r}(t+\Delta t)$. The dynamical evolution of the reservoir computing machine is described by

$$\mathbf{r}(t+\Delta t) = (1-\alpha)\mathbf{r}(t) + \alpha \tanh[\mathcal{A}\cdot\mathbf{r}(t) + \mathcal{W}_{\text{in}}\cdot\mathbf{u}(t) + k_b\mathcal{W}_b(b+b_0)], \quad (1)$$

$$\mathbf{v}(t) = \mathcal{W}_{\text{out}}\cdot\mathbf{r}(t), \quad (2)$$

where b is the bifurcation parameter, α is the leakage parameter, and k_b and b_0 are two hyperparameters determining the scaling and the bias of the input of the bifurcation parameter b into the reservoir network.

The elements of the two input matrices $\mathcal{W}_{\text{in}}$ and $\mathcal{W}_b$ as well as those of the reservoir network matrix $\mathcal{A}$ are generated randomly *a priori* and fixed during the training and prediction phases. Training is administered to adjust only the output matrix $\mathcal{W}_{\text{out}}$ to minimize the difference between $\mathbf{v}(t)$ and $\mathbf{u}(t)$, so that the reservoir can predict the evolution of the target system into the future with input of the dynamical variables from the past. (A more detailed

description of the training and predicting processes is provided in Appendix A.) We train the reservoir machine using time series from a few distinct parameter values—all in the normal or safe regime where the system still possesses a chaotic attractor, as shown in Fig. 1(b). Because of the additional parameter input channel, the machine trained with data at different parameter values will gain the ability to "sense" the variation in the parameter and the associated changes in the time-series data from different parameter values. Such a well-trained machine is a high-dimensional representation of the original dynamical system. Finally, to predict with what parameter drift the system will exhibit transient chaos and then collapse, we simply input the parameter value of interest into the parameter input channel. Then, at each time step, we collect the one-step prediction from the output layer $\mathbf{v}(t)$ and feed it back to the input layer $\mathbf{u}(t) = \mathbf{v}(t)$. Now the reservoir machine in the predicting phase is a closed-loop dynamical system with one constant external drive—the bifurcation parameter of interest. The machine in the predicting phase will be able to predict the system collapse preceded by transient chaos for the input value of the bifurcation parameter.

To enable reliable predictions outside the training parameter region, a reservoir machine needs to be "well trained" in the sense that it can learn the different dynamical behaviors of the target system at all the training bifurcation parameter values. The goals of training are to make the reservoir machine learn the different attractors of the target system and to "teach" the machine to associate an attractor with a specific value of the bifurcation parameter. We find that, for a number of representative target systems described by nonlinear ordinary or partial differential equations, a reservoir machine so trained not only is able to predict the dynamics at all the training parameter points accurately (with relative errors less than 5%) for about four or five Lyapunov times, but also can predict the critical point and transient chaos reliably and accurately. It is thus reasonable to impose this requirement of prediction accuracy for all training parameter values as a criterion for estimating the hyperparameter values based on the training data. As will be demonstrated in Sec. III below, insofar as the training parameter values are reasonably close to the critical transition point to transient chaos, their occurrences can be predicted accurately.

To achieve accurate prediction within four or five Lyapunov times for all the training parameter values, hyperparameter optimization is required as it is significantly more difficult for a reservoir machine to learn several different attractors simultaneously than to learn a single attractor. Using Bayesian optimization [25], we optimize seven of the hyperparameters of the reservoir computing machine, which are the average degree d of the reservoir hidden network, the spectral radius ρ of the reservoir hidden network, the scaling factor k_{in} of the input matrix $\mathcal{W}_{\text{in}}$, the regularization parameter β used during the training of the output matrix $\mathcal{W}_{\text{out}}$, and parameters k_b, b_0, and α. (A more detailed description of these hyperparameters and their optimization can be found in Appendixes A and B.) In addition, the random nature of the input and hidden layers can cause fluctuations in the validation and prediction performance. We thus train five different random realizations of the reservoir machines at a time and keep the one with the lowest validation error only. (A more detailed discussion of the effects of input and reservoir matrices can be found in Appendix C.)

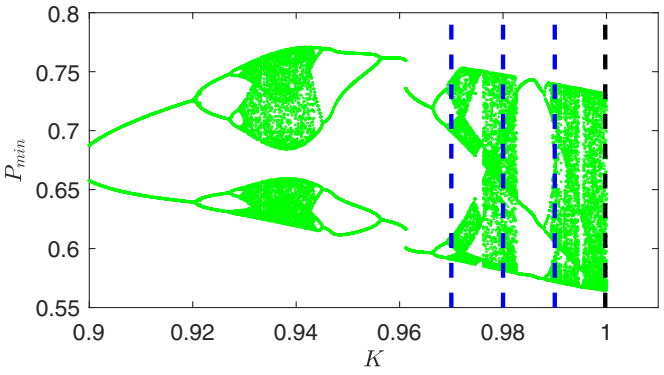

FIG. 2. Bifurcation diagram of the food chain system. The vertical black dashed line indicates the critical point $K_c = 0.99976$. The three vertical blue dashed lines specify the three values of bifurcation parameter K used for training the reservoir machine, which are $K = 0.97$, 0.98, and 0.99.

III. RESULTS

We demonstrate our machine learning approach to predicting critical transition and system collapse using three target systems: a chaotic food chain system, a power model system, and the spatiotemporal one-dimensional Kuramoto-Sivashinsky system.

A. Chaotic food chain system

Sudden extinction of species in ecological systems [36] has been occurring at an alarming rate. A possible reason for local extinction can be attributed to changes in environmental factors that lead to a shift in the system parameters. We consider a three-species food chain model [2]:

$$\frac{dR}{dt} = R\left(1 - \frac{R}{K}\right) - \frac{x_c y_c C R}{R + R_0}, \tag{3}$$

$$\frac{dC}{dt} = x_c C\left[\frac{y_c R}{R + R_0} - 1\right] - \frac{x_p y_p P C}{C + C_0}, \tag{4}$$

$$\frac{dP}{dt} = x_p P\left(\frac{y_p C}{C + C_0} - 1\right), \tag{5}$$

where R, C, and P are the population densities of the resource, consumer, and predator species, respectively. K is a parameter characterizing the environmental capacity of the resource species and is chosen to be the bifurcation parameter. x_c, y_c, x_p, y_p, R_0, and C_0 are other parameters in the system we assume to be constants. A bifurcation diagram of this food chain system is shown in Fig. 2.

As the environmental capacity K of the resource species is varied, a catastrophic bifurcation and subsequent transient chaos leading to sudden species extinction occurs. A boundary crisis occurs at the critical value $K = K_c = 0.99976$. Figures 3(a1) and 3(a2) show typical behaviors of the predator density P for $K < K_c$ and $K > K_c$, where there is sustained chaos in the former and transient chaos leading to species extinction in the latter.

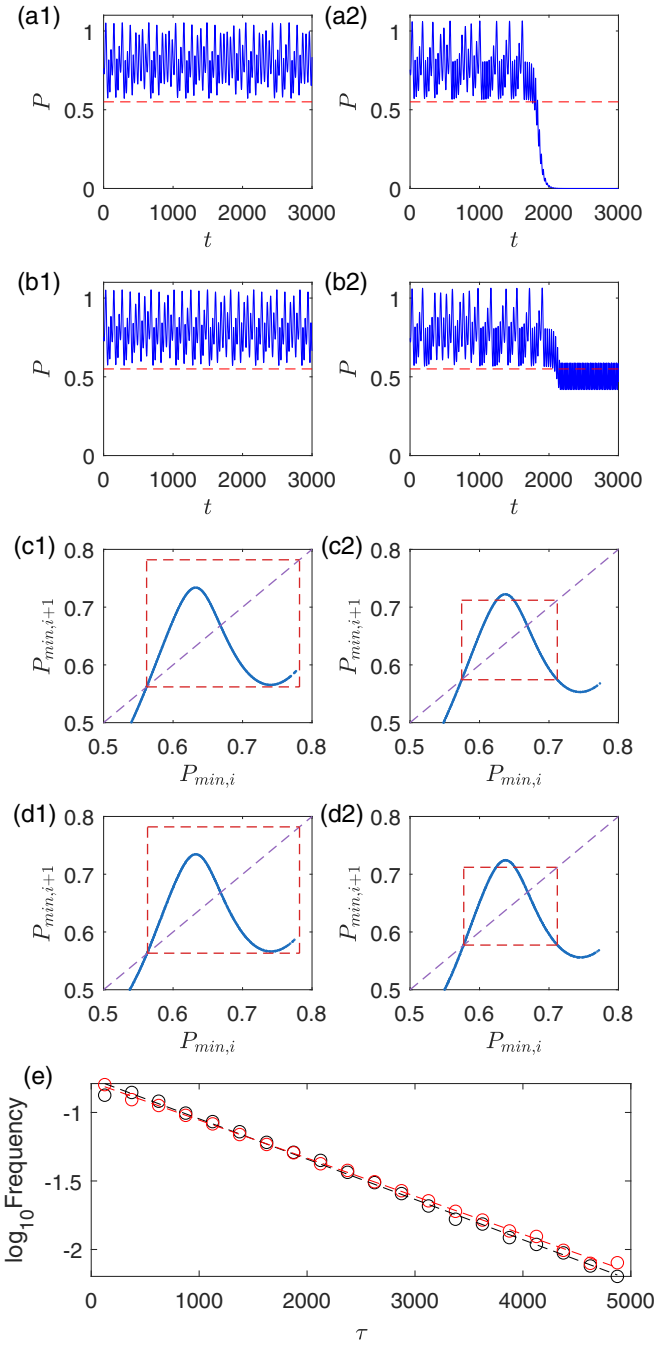

FIG. 3. Predicting transient chaos and collapse of a food chain system. (a1), (a2) Typical time series of the predator density P in the normal and transient regimes, respectively, for $K = 0.997 < K_c$ and $K = 1.01 > K_c$. (b1), (b2) Predicted time series of P for the same values of K as in (a1) and (a2), respectively. The machine predicts correctly the sudden collapse for $K > K_c$. (c1), (c2) Return maps constructed from the local minima of $P(t)$ from the true time series, where the red dashed squares define an interval in which an invariant set exists: (c1) a chaotic attractor at $K = 0.997 < K_c$ or (c2) a nonattracting chaotic set at $K = 1.01 > K_c$ due to the escaping region about the critical point leading to transient chaos. (d1), (d2) Predicted return maps for the same values of K as in (c1) and (c2), respectively. (e) Actual (black) and predicted (red) transient lifetime distributions at $K = K_c + 2 \times 10^{-4}$.

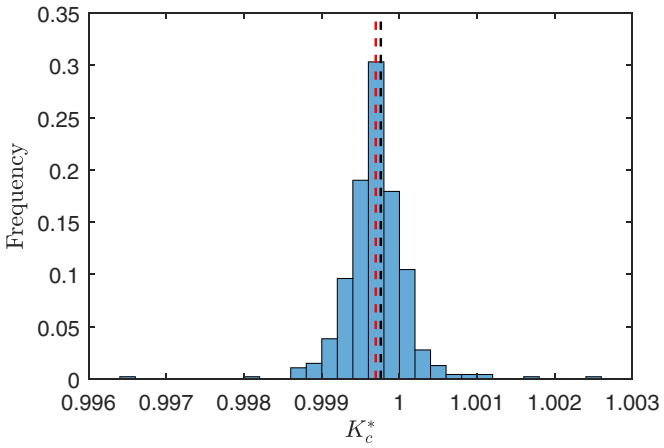

FIG. 4. Histogram of the predicted critical point K_c^* for the chaotic food chain system. The number of random realizations is 500. The black and red vertical dashed lines show the true value of the critical point $K_c = 0.99976$ and the average predicted value $\langle K_c^* \rangle = 0.99970$, respectively.

We train the reservoir machine at three different values of the bifurcation parameter: $K = 0.97$, 0.98, and 0.99, all in the sustained chaos regime (Fig. 2). For each K value, training is done such that the reservoir machine is able to predict the exact state evolution of the original system for several Lyapunov times. After training, we apply some parameter change ΔK and test, for each resulting parameter value, whether the predicted system state is a chaotic attractor or a chaotic transient. An exemplary pair of the predicted state for $K < K_c$ and $K > K_c$ is shown in Figs. 3(b1) and 3(b2), respectively. It is remarkable that the reservoir machine is able to predict the collapse after a chaotic transient, as shown in Fig. 3(b2). Examining the prediction results for a set of systematically varied ΔK values enables determination of the predicted critical bifurcation point, denoted as K_c^*. Averaging over an ensemble of 500 different random reservoir realizations, we obtain the value of the critical point as $K_c^* = 0.9997 \pm 4 \times 10^{-4}$, as shown in Fig. 4, which agrees well with the actual value $K_c = 0.99976$. Our machine learning framework can also predict a basic statistical characteristic of transient chaos: the lifetime distribution. To demonstrate this, we set the control parameter of the reservoir to be $K = K_c^* + 2 \times 10^{-4}$ so that the system is in the transient chaos regime and the distribution of transient lifetime in the target system is exponential. The reservoir system predicts correctly the exponential distribution, as shown in Fig. 3(e), where 100 stochastic realizations of the reservoir machines and 400 random initial conditions with each machine are used. The predicted average transient lifetime is about 1.35×10^3, which agrees well with the true value (1.33×10^3), demonstrating the power of our reservoir computing scheme for predicting transient chaos and system escape (collapse).

The values of the constant parameters in the food chain system are [1,2] $x_c = 0.4$, $y_c = 2.009$, $x_p = 0.08$, $y_p = 2.876$, $R_0 = 0.16129$, and $C_0 = 0.5$. The hyperparameters of the reservoir system are $n = 900$, $d = 4$, $\rho = 2.3$, $k_{\text{in}} = 3.6$, $k_b = 0.5$, $b_0 = -2.2$, $\alpha = 0.30$, and $\beta = 3 \times 10^{-5}$. The time step is

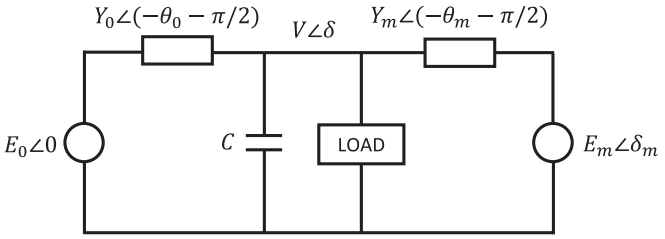

FIG. 5. Power system model in which a voltage collapse following transient chaos can occur. The bifurcation parameter is Q_1, a load parameter.

$\Delta t = 1$, and the training length for each value of the bifurcation parameter is $t_{\text{train}} = 3200$, which contains approximately 80 oscillation cycles. The validation time length is $t_{\text{validating}} = 1200$ corresponding to about 30 periods of oscillations. Averaged over an ensemble of 100 random reservoir machines, the validation root-mean-square error (RMSE) is about 0.17. Typically the reservoir system is able to predict about four to six Lyapunov times with error less than 5% of the oscillation range of the dynamical variables.

While our approach has yielded good predictions of the critical transition point and the transient lifetime distribution, the reservoir machine is unable to predict the actual final state after the collapse correctly [e.g., Fig. 3(b2)]. A reason is that only the information about the sustained chaotic attractor of the target system has been learned by the machine through the training data set, and the machine has never been exposed to any information about the collapsed state of the system after the critical point. This phenomenon also occurs for the power system in Sec. III B.

B. Power system model

Voltage collapse is a major type of instability in electrical power systems [37], where the dynamical variables of the system fluctuate chaotically for a finite amount of time, i.e., exhibiting transient chaos, before suddenly collapsing to zero. Initially, the system operates normally with a stable attractor. Disturbances cause a change in the system parameter through the critical value at which a boundary crisis occurs [38], driving the system into transient chaos in which the final asymptotic state is a voltage collapse. A generic model of the electrical power system with voltage collapse, as illustrated in Fig. 5, uses four ordinary differential equations [37,38]:

$$\dot{\delta}_m = \omega, \tag{6}$$

$$M\dot{\omega} = -d_m\omega + P_m - E_m V Y_m \sin(\delta_m - \delta), \tag{7}$$

$$K_{qw}\dot{\delta} = -K_{qv2}V^2 - K_{qv}V + Q(\delta_m, \delta, V) - Q_0 - Q_1, \tag{8}$$

$$\begin{aligned} TK_{qw}K_{pv}\dot{V} = {} & K_{pw}K_{qv2}V^2 + (K_{pw}K_{qv} - K_{qw}K_{pv})V \\ & + K_{qw}[P(\delta_m, \delta, V) - P_0 - P_1] \\ & - K_{pw}[Q(\delta_m, \delta, V) - Q_0 - Q_1], \end{aligned} \tag{9}$$

where $V\angle\delta$ is the load voltage, the generator terminal voltage is $E_m\angle\delta_m$, the infinite bus has terminal voltage $E_0\angle 0$, and ω is the speed of the generator rotor. The load includes a constant

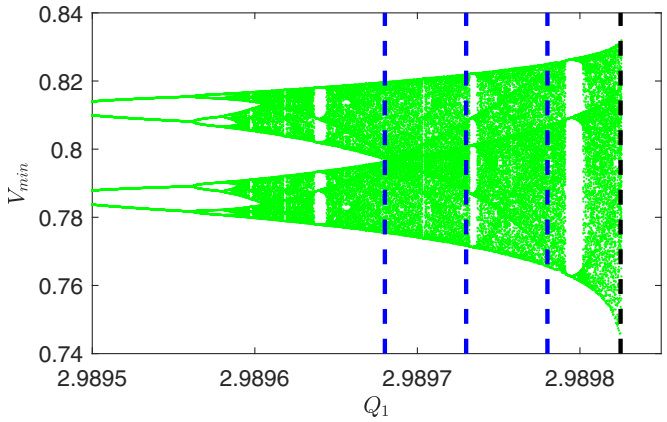

FIG. 6. Bifurcation diagram of the power system. The vertical black dashed line represents the critical point $Q_{1c} = 2.9898256$. The three vertical blue dashed lines specify the three values of Q_1 used for training the reservoir machine: $Q_1 = 2.98968$, 2.98973, and 2.98978.

PQ load in parallel with an induction motor. The real and reactive powers supplied to the load by the network are

$$P(\delta_m, \delta, V) = -E_0' V Y_0' \sin\delta + E_m V Y_m \sin(\delta_m - \delta), \tag{10}$$

$$\begin{aligned} Q(\delta_m, \delta, V) = {} & E_0' V Y_0' \cos\delta - (Y_0' + Y_m)V^2 \\ & + E_m V Y_m \cos(\delta_m - \delta). \end{aligned} \tag{11}$$

These equations contain a Thévenin equivalent circuit with the following adjusted values:

$$E_0' = \frac{E_0}{\left(1 + C^2 Y_0^{-2} - 2CY_0^{-1}\cos\theta_0\right)^{1/2}}, \tag{12}$$

$$Y_0' = Y_0\left(1 + C^2 Y_0^{-2} - 2CY_0^{-1}\cos\theta_0\right)^{1/2}, \tag{13}$$

$$\theta_0' = \theta_0 + \tan^{-1}\left(\frac{CY_0^{-1}\sin\theta_0}{1 - CY_0^{-1}\cos\theta_0}\right). \tag{14}$$

The constant system parameters are set as [38] $M = 0.01464$, $C = 3.5$, $E_m = 1.05$, $Y_0 = 3.33$, $\theta_0 = 0$, $\theta_m = 0$, $K_{pw} = 0.4$, $K_{pv} = 0.3$, $K_{qw} = -0.03$, $K_{qv} = -2.8$, $K_{qv2} = 2.1$, $T = 8.5$, $P_0 = 0.6$, $P_1 = 0.0$, $Q_0 = 1.3$, $E_0 = 1.0$, $Y_m = 5.0$, $P_m = 1.0$, and $d_m = 0.05$.

Figure 6 shows the relevant bifurcation diagram. A representative and often studied bifurcation parameter of the system is Q_1, which measures the reactive power demand at the load bus [37,38]. A boundary crisis occurs at the critical value $Q_{1c} = 2.9898256$, where there is a chaotic attractor for $Q_1 < Q_{1c}$ and there is transient chaos leading to an eventual collapse for $Q_1 > Q_{1c}$, as shown in Figs. 7(a1) and 7(a2).

We train the reservoir machine at three different values of the bifurcation parameter, $Q_1 = 2.98968$, 2.98973, and 2.98978, all in the normal operation regime with a chaotic attractor, as denoted by the three vertical blue dashed lines in Fig. 6. (The reason that the Q_1 values used for training appeared close to the critical value is that the value range of Q_1 exhibiting chaos is narrow, which is an intrinsic feature of this type of power system, as shown in the bifurcation diagram in Fig. 6.) After training, we apply systematic changes in Q_1 and test, for each resulting parameter value, whether the predicted

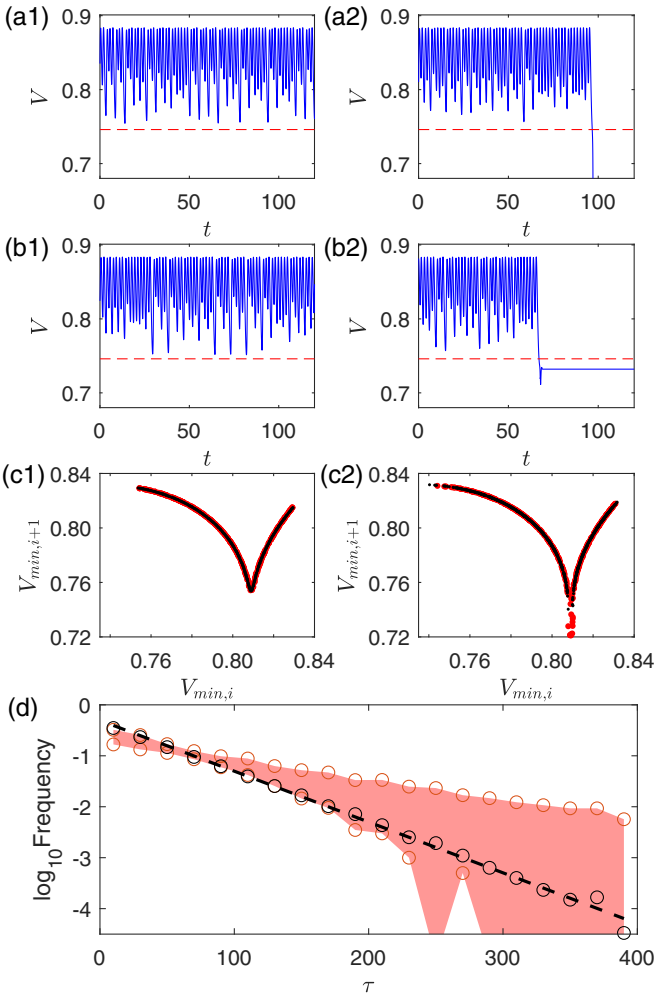

FIG. 7. Predicting voltage collapse in an electrical power system. (a1), (a2) True time evolution of the load voltage V before and after a voltage collapse for $Q_1 = 2.989823 < Q_{1,c}$ and $Q_1 = 2.989828 > Q_{1,c}$. There is transient chaos preceding the collapse for $Q_1 > Q_{1,c}$. (b1), (b2) The corresponding time series predicted by reservoir computing. In spite of training done exclusively in the precollapse regime $Q_1 < Q_{1,c}$, the neural machine successfully predicts the occurrence of the collapse in $Q_1 > Q_{1c}$. (c1), (c2) Black dots denote return maps constructed from the local minima of $V(t)$ from the actual time series for $Q_1 = 2.989818 < Q_{1,c}$ and $Q_1 = 2.989832 > Q_{1c}$, respectively. The small gap about $V_{\min,i} = 0.81$ in (c2) is the "escaping" channel in the phase space through which trajectories escape the "safe" region, leading to voltage collapse. Red dots denote the corresponding return maps predicted by the machine. (d) Actual (black) and predicted (red) transient lifetime distributions, where the shaded region indicates the range of the machine learning prediction.

result is a chaotic attractor or a chaotic transient. A pair of predicted voltage time series for $Q_1 < Q_{1c}$ and $Q_1 > Q_{1c}$ are shown in Figs. 7(b1) and 7(b2). While the asymptotic value of the predicted voltage in the transient regime is not exactly zero, it is remarkable that the reservoir machine is able to predict a sudden drop in the voltage after a chaotic transient, as shown in Fig. 7(b2). Examining the prediction results for a set of systematically varied ΔQ_1 values enables determination of the predicted critical bifurcation point, denoted as

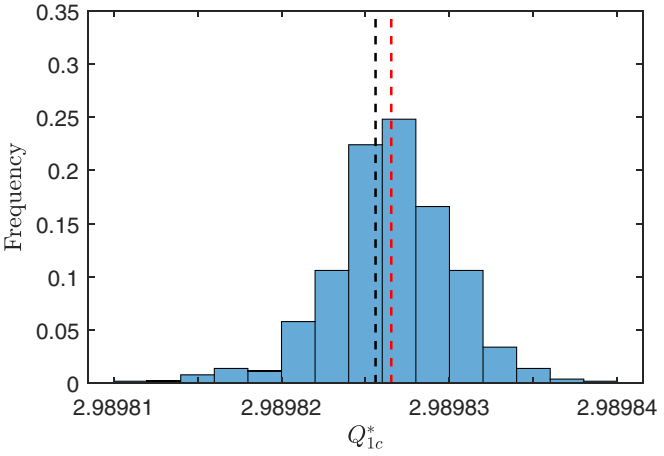

FIG. 8. Histogram of the predicted critical point Q^*_{1c}, with 500 random realizations of the reservoir machine. The black and red vertical dashed lines indicate the true critical value $Q_{1c} = 2.9898256$ and the average predicted value $\langle Q^*_{1c}\rangle = 2.9898265$, respectively.

Q^*_{1c}. Averaging over 500 independent random realizations of the reservoir configurations, we get $Q^*_{1c} = 2.9898265 \pm 4 \times 10^{-6}$, with relative error $\delta = 2\%$ (the relative error δ in this work is defined as the error in the predicted critical point divided by the distance between the real critical point and the nearest training point). A histogram of the predicted critical points Q^*_{1c} from 500 statistical realizations is shown in Fig. 8. The position of the peak of the distribution and the mean value of the predicted critical point both are close to its true value Q_{1c}. Our framework can also predict the fundamental statistical characteristic of transient chaos: the lifetime distribution. To demonstrate this, we set $Q_1 = Q_{1c} + 5 \times 10^{-6}$ so that the system is in the transient chaotic regime and the distribution of the transient lifetime is exponential. The reservoir system predicts correctly the exponential distribution, as shown in Fig. 7(d), where 100 stochastic realizations of the reservoir system and 2000 random initial conditions for each realization are used to generate the range of the predicted distribution. We see that the true distribution is contained in the range of predictions, demonstrating the predictive power of our reservoir computing scheme for transient chaos.

The hyperparameters of the reservoir machine are $n = 800$, $d = 250$, $\rho = 1.6$, $k_{\text{in}} = 2.1$, $k_b = 1.6$, $b_0 = -3.1$, $\alpha = 1$, and $\beta = 1 \times 10^{-4}$. The time step with which the reservoir state is updated is $\Delta t = 0.05$. The length of the training time for each value of the bifurcation parameter is $t_{\text{train}} = 500$, during which the system exhibits approximately 300 oscillation cycles. The validating time is $t_{\text{validate}} = 25$. Averaged over an ensemble of 100 random reservoir machines, the validation RMSE is about 0.07. Usually the reservoir machine is able to accurately predict the system state for about four to six Lyapunov times, with the absolute prediction error in the voltage variable $V(t)$ less 5% of its oscillation range.

C. One-dimensional Kuramoto-Sivashinsky system

We demonstrate that our "parameter-aware" reservoir computing machine can predict transient chaos in spatiotemporal dynamical systems. In particular, we consider systems

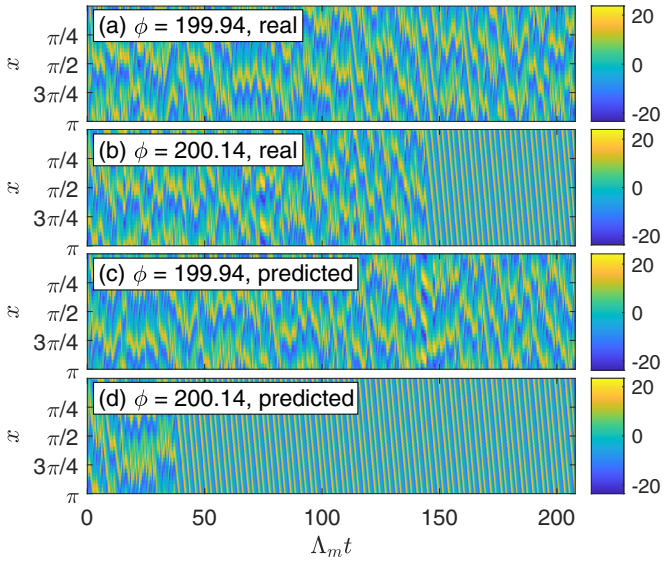

FIG. 9. Predicting transient chaos in spatiotemporal systems described by the one-dimensional KSE. (a), (b) Spatiotemporal evolution of the field $u(x,t)$ in the sustained ($\phi < \phi_c$) and transient ($\phi > \phi_c$) chaotic regime, respectively, where the time (horizontal) axis is in units of the Lyapunov time and ϕ is the bifurcation parameter. In the transient regime, the spatiotemporal chaotic solution eventually collapses into a regular, traveling wave solution. (c), (d) Reservoir computing predicted spatiotemporal evolution patterns for the same parameter value as in (a) and (b), respectively, where the machine successfully predicts the collapse.

described by the one-dimensional (1D) Kuramoto-Sivashinsky equation (KSE):

$$\frac{\partial u}{\partial t} + \upsilon \frac{\partial^4 u}{\partial x^4} + \phi\left(\frac{\partial^2 u}{\partial x^2} + u\frac{\partial u}{\partial x}\right) = 0, \tag{15}$$

where $u(x,t)$ is a scalar field, and υ and ϕ are the system parameters. While the 1D KSE has been used in recent studies as a paradigmatic model to demonstrate model-free prediction of spatiotemporal chaotic evolution [20], in the parameter regime studied there is only sustained chaos. Transient chaos can arise in the 1D KSE [35] but in a quite different region in the parameter plane (υ, ϕ). To generate transient spatiotemporal chaos in the 1D KSE, we set $\upsilon = 4$ and let ϕ be the bifurcation parameter. The spatial domain is $0 \leqslant x \leqslant \pi$ with a periodic boundary condition. A crisis occurs at $\phi_c \approx 200.04$, where there is sustained and transient spatiotemporal chaos for $\phi \lesssim \phi_c$ and $\phi \gtrsim \phi_c$, respectively. The two types of solutions are shown in Figs. 9(a) and 9(b), respectively.

We train the reservoir machine at three different values of the bifurcation parameter: $\phi = 196$, 197, and 198, all in the regime of sustained spatiotemporal chaos. Figures 9(c) and 9(d) show the predicted spatiotemporal evolution patterns corresponding to the values of ϕ in Figs. 9(a) and 9(b), respectively. It can be seen that the reservoir machine is able to predict correctly transient spatiotemporal chaos.

Because of the high dimensionality of the KSE system and the need to test a large number of values of the bifurcation parameter for any given reservoir structure, exploiting a large number of reservoir realizations to obtain the predicted critical bifurcation point and statistical distribution of the transient lifetime is computationally infeasible. Our solution is to divide the test range into six coarse subregions according to the relative error δ in the prediction of the critical point ϕ_c^*: $\delta < -10\%$, $-10\% < \delta < -5\%$, $-5\% < \delta < 0$, $0 < \delta < +5\%$, $+5\% < \delta < +10\%$, and $\delta > +10\%$. We then determine, for each random realization, to which subregion the predicted transition point ϕ_c^* belongs by observing the prediction of the trained reservoir machine for $\phi = 199.84$, 199.94, 200.04, 200.14, and 200.24. The result with 250 random realizations is shown in Fig. 10, where about 74% of the predicted ϕ_c^* values have relative error $|\delta| < 5\%$. However, if the accuracy is relaxed to $|\delta| < 10\%$, the fraction of correct prediction becomes 98%.

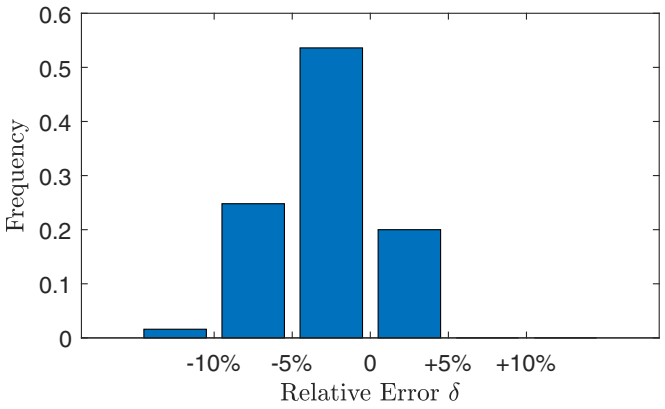

FIG. 10. Distribution of the relative error in the predicted transition point for the KSE system. The number of random realizations of the reservoir network is 250. About 74% of the predicted ϕ_c^* values have relative error $|\delta| < 5\%$. Within the accuracy $|\delta| < 10\%$, the fraction of correct prediction is 98%.

The hyperparameters of the reservoir system are $n = 4000$, $d = 450$, $\rho = 0.89$, $k_{\text{in}} = 0.057$, $k_b = -0.052$, $b_0 = -185$, $\alpha = 1$, and $\beta = 8 \times 10^{-5}$. The spatial dimension is evenly discretized into 32 nodes, i.e., $D_{\text{in}} = 32$. The reservoir's working time step is 2×10^{-5}. The largest Lyapunov exponent of the KSE system at the training parameter values is about $\Lambda_m \simeq 520$. The training and validation time lengths for each value of the bifurcation parameter are $t_{\text{train}} = 0.24$ and $t_{\text{validation}} = 10$ Lyapunov times ≈ 0.019. The prediction horizon is usually about four or five Lyapunov times.

D. An alternative machine learning approach: Measurements with a continuously varying bifurcation parameter

In all the results above, the reservoir machines are trained at a few discrete values of the bifurcation parameter. In many real-world systems, parameters may never be constant and in fact can drift continuously in time. For instance, suppose one wishes to train a reservoir machine with the climate data with the global yearly averaged temperature being the bifurcation parameter to predict if this temperature may exceed a critical point in the future after which the climate system would collapse. The global temperature is not a constant but increases with time in a continuous fashion. A viable approach is then to train the reservoir with a single time series collected in time with a continuously varying bifurcation parameter. We use

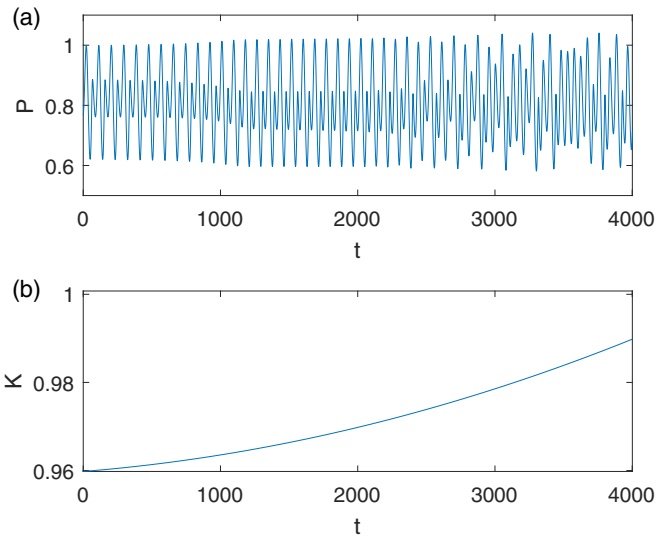

FIG. 11. Training a reservoir machine with a continuously varying bifurcation parameter for the chaotic food chain system. (a) Predator density P in the training time series. (b) Time series of the bifurcation parameter $K(t)$.

the same reservoir settings as above, but the input parameter channel now receives a time series representing the variations of the bifurcation parameter with time.

To demonstrate the power of the reservoir machine to predict the critical transition point to system collapse subject to continuous parameter drifting, we use the chaotic food chain system as an example. For training, the bifurcation parameter $K(t)$ is assumed to vary quadratically with time from $K = 0.96$ to $K = 0.99$, fully contained in the regime of sustained chaos or periodical oscillations. The single time series $K(t)$ has the length $T = 4000$, which contains about 100 oscillation cycles, as shown in Fig. 11. An average over an ensemble of 150 reservoir realizations yields the predicted critical point $K_c^* = 1.000 \pm 0.002$ (relative error $\delta = 2\%$), attesting to the power of the reservoir to predict transient chaos and system collapse subject to continuous parameter drifts.

IV. DISCUSSION

If predicting system collapse is viewed as a binary classification problem (i.e., with or without collapse), it would be useful to train the neural machine with data from both below and above the critical point. A difficulty is that, beyond the critical point, the system will collapse after a transient chaotic phase of random duration. Practically, it is infeasible to obtain sufficient training data from the system in the postcritical regime. Moreover, our machine learning method can be used to assess the likelihood of the occurrence of a crisis in the near future where no postcritical data are available.

Our results suggest that the reservoir machine trained with data from a few distinct values of the bifurcation parameter represents a "regression" between the dynamical behavior of the target system and the bifurcation parameter. The machine is able to make statistically accurate predictions outside the training region. Since training is done on as few as three different values of the bifurcation parameter, the prediction of, e.g., the critical transition point from any individual reservoir realization will involve large errors. However, the collective prediction from an ensemble of statistically independent reservoir machines can be quite accurate. In general, the prediction error would increase if the training parameter values are further from the critical transition point, suggesting that reservoir machines represent a low-order approximation of the real dynamical systems about the training points of the bifurcation parameter. This is a natural and inevitable trade-off.

In summary, we have articulated a parameter-aware scheme of reservoir computing to predict collapse as a result of parameter drift driving the system into transient chaos by designating an additional input channel to accommodate the bifurcation parameter, which is equivalent to introducing adjustable biases between the input and the hidden layers. With parameter-dependent training, all in the regime of sustained chaos, the reservoir machine acquires the ability to capture the variations in the "climate" of the target system, thereby gaining the power to predict the system state for different parameter values. When a parameter drift pushes the system through a critical point into a regime of transient chaos where collapse is inevitable, our design of machine learning is capable of accurate prediction of the critical value of the parameter, and of the statistical characteristics of transient chaos and the eventual collapse. These features are demonstrated using a food chain model in ecology, an electrical power system, and the one-dimensional Kuramoto-Sivashinsky system. Not only can our parameter-aware reservoir computing machine predict the critical transition point, it can also predict the distribution of the transient lifetime in the parameter regime beyond the transition, which is a fundamental physical characteristic of transient chaos.

ACKNOWLEDGMENTS

We would like to acknowledge support from the Vannevar Bush Faculty Fellowship program sponsored by the Basic Research Office of the Assistant Secretary of Defense for Research and Engineering and funded by the Office of Naval Research through Grant No. N00014-16-1-2828.

APPENDIX A: PROPOSED "PARAMETER-AWARE" RESERVOIR COMPUTING FRAMEWORK

Our proposed reservoir computing framework for predicting transient chaos and system collapse involves various types of parameters. It is useful to clarify the meanings of these parameters. For convenience, we call the nonlinear dynamical system that the reservoir machine is designed and trained to predict the *target system*. We classify all parameters involved in the target and reservoir systems into five categories:

(1) Bifurcation parameter: This parameter belongs to the target system. Varying the bifurcation parameter can drive the system out of normal operation with sustained chaos into the regime of transient chaos. The values of the bifurcation parameter are fed into the reservoir neural machine through the input parameter channel to empower the machine with "parameter awareness" to achieve the goal of predicting transient chaos and system collapse.

(2) Fixed parameters of the target system: This includes all parameters (except the bifurcation parameter) involved in defining the target system. In the present work, the values of these parameters are preassigned and fixed.

(3) Hyperparameters of the reservoir machine: This is the set of predetermined parameters defining the machine, which includes the size, average degree, and spectral radius of the neural network, as well as a set of parameters controlling the training process. There are standard methods such as Bayesian optimization [39,40], surrogate optimization (SGO) [41–43], particle swarm optimization [44,45], and genetic algorithm [46–48], which can be used to determine an optimal set of hyperparameters.

(4) Elements of the two input matrices and of the reservoir adjacency matrix [cf. Fig. 1(a)]: These are chosen randomly prior to training and fixed during training and prediction for any given reservoir realization. Their distributions are controlled by the hyperparameters.

(5) Training parameters: These are the set of parameters associated with the reservoir machine, whose values are to be determined through training. The values of these parameters depend on the training data sets.

Figure 1(a) illustrates the proposed reservoir computing machine. Differing from conventional reservoir computing used to predict chaotic systems [13–28], we have an additional "parameter sensing" input channel through which the specific values of the bifurcation parameter are fed into the reservoir neural network. A reservoir computing machine is a recurrent neural network of three layers: an input layer, a hidden recurrent layer, and an output layer. The input layer has two components, one receiving the time series constituting the $D_{\rm in}$-dimensional input vector $\mathbf{u}(t)$ and another taking in the value of the bifurcation parameter of the target system. The weighted, $D_r \times D_{\rm in}$ matrix $\mathcal{W}_{\rm in}$ maps $\mathbf{u}(t)$ to the D_r-dimensional state vector $\mathbf{r}(t)$ of the hidden layer. The $D_r \times D_b$ matrix $\mathcal{W}_b$ specifies the connection weights between the input bifurcation parameter to $\mathbf{r}(t)$, where we use $D_b = 1$ in this work. The $D_r \times D_r$ adjacency matrix $\mathcal{A}$ defines the structure of the reservoir network in the hidden layer, where the dynamics of each node are described by an internal state and a nonlinear (hyperbolic tangent) activation function. The weighted $D_{\rm out} \times D_r$ matrix $\mathcal{W}_{\rm out}$ maps the state vector $\mathbf{r}(t)$ to the $D_{\rm out}$-dimensional output vector $\mathbf{v}(t)$. We set $D_{\rm in} = D_{\rm out}$. The matrices $\mathcal{W}_{\rm in}$, $\mathcal{W}_b$, and $\mathcal{A}$ are generated randomly prior to training, while all elements of $\mathcal{W}_{\rm out}$ are to be determined through training.

A step-by-step description of the training, validating, and testing processes of our prediction framework is as follows.

Training phase I: The preparation phase. The elements of the matrices $\mathcal{W}_{\rm in}$, $\mathcal{W}_b$, and $\mathcal{A}$ are generated randomly and fixed, a process controlled by four hyperparameters $k_{\rm in}$, n, d, and ρ, where $k_{\rm in}$ is a scaling factor of the elements of the two input matrices $\mathcal{W}_{\rm in}$ and $\mathcal{W}_b$, n is the number of hidden nodes ($D_r = n$), and d and ρ are the average degree and spectral radius of $\mathcal{A}$, respectively. When the target system is low dimensional, the input matrix $\mathcal{W}_{\rm in}$ is dense: each node in the input layer is connected to all nodes in the reservoir network. In this case, we set the reservoir network to be an undirected (symmetric) random network and choose the elements of $\mathcal{A}$ from a zero mean normal distribution. If the target system is a spatiotemporal dynamical system (of arbitrarily high dimension), we make each input node connected only to a fraction of $1/D_{\rm in}$ of nodes in the hidden layer, and we choose the reservoir network to be a directed (asymmetric) random network with weighed elements chosen from a uniform distribution between zero and a fixed threshold value. (These choices were made rather arbitrarily, and other choices may also work.) In both cases, the nonzero elements of $\mathcal{W}_{\rm in}$ and all elements of $\mathcal{W}_b$ are generated from a uniform distribution in the interval $[-k_{\rm in}, k_{\rm in}]$.

Training phase II: Feed-forward process from input to hidden layer. We input the time series constituting the $\mathbf{u}(t)$ vector, feed it forward to the hidden layer, and record the state vector $\mathbf{r}(t)$ as a function of time. The dynamical updating rule for $\mathbf{r}(t)$ is given by Eq. (1), where Δt is the time step, the vector $\tanh(\mathbf{p})$ is defined to be $[\tanh(p_1), \tanh(p_2), \ldots]^T$ for a vector $\mathbf{p} = [p_1, p_2, \ldots]^T$, and b stands for the bifurcation parameter associated with the current time series. The quantities k_b and b_0 are two hyperparameters associated with the parameter input matrix $\mathcal{W}_b$. The initial condition for the network state evolution is set to be $\mathbf{r}(t = 0) = \mathbf{0}$.

With respect to training, the main feature that distinguishes our work from previous ones is training at multiple values of the bifurcation parameter. In particular, after the reservoir has been trained with data associated with one value of the bifurcation parameter, we reset the time and the initial states to zero, and repeat the training with data from another value of the parameter. After training is done for the available data from all preassigned values of the bifurcation parameter, we obtain multiple recordings of $\mathbf{r}(t)$, irrespective of the order of training.

The bifurcation parameter can be viewed as an additional dimension of the input time-series data, where Eq. (1) can be rewritten by combining $\mathcal{W}_{\rm in}$ and $\mathcal{W}_b$ into a single matrix $D_r \times (D_{\rm in} + D_b)$. Likewise, the two input vectors $\mathbf{u}$ and $k_b(b + b_0)$ can be combined. Alternatively, we may regard $k_b(b + b_0)\mathcal{W}_b$ as an adjustable bias matrix between the input layer and the hidden layer. In this point of view, for different values of the bifurcation parameter, the reservoir has a different set of inner parameters, with a global bias for each and every node as determined by the value of the bifurcation parameter.

In dealing with spatiotemporal chaotic systems, for computational feasibility, every input dimension from the target system's time series is connected to only a subset of the nodes in the hidden layer, but the bifurcation parameter is still connected with all hidden nodes. From the point of view of the input vectors, there is then an asymmetry between the input time series of the variables of the target system and the bifurcation parameter. However, for predicting transient chaos and collapse in a low-dimensional system, this asymmetry between state variable and parameter does not arise because the input matrix $\mathcal{W}_{\rm in}$ is dense in the sense that every input dimension is connected to all nodes in the hidden layer. It is worth emphasizing that, even if the input parameter channel is connected to only a fraction of the hidden nodes, the reservoir computing scheme is still capable of the prediction task.

Training phase III: Regression. We determine the elements of $\mathcal{W}_{\rm out}$ through a regression between the true data vector $\mathbf{u}(t)$ and the network state vector $\mathbf{r}(t)$. We stack the multiple recordings of $\mathbf{r}(t)$ from different values of the bifurcation

parameter on top of each other in the temporal dimension (after removing a small segment, e.g., ten time steps, of the recording to minimize the effect of possible transient behavior of the reservoir system) to form a vector function $\mathbf{r}_{\rm all}(t)$ and treat the data vector $\mathbf{u}(t)$ accordingly to form another vector function $\mathbf{u}_{\rm all}(t)$. Following the suggested regression method for predicting chaotic systems [16], we first replace $\mathbf{r}_{\rm all}(t)$ by $\mathbf{r}'_{\rm all}(t)$, where $\mathbf{r}'_{\rm all}(t)_i = \mathbf{r}_{\rm all}(t)_i$ for odd rows and $\mathbf{r}'_{\rm all}(t)_i = \mathbf{r}_{\rm all}(t)_i^2$ for even rows. We then perform a linear regression between $\mathbf{u}_{\rm all}(t)$ and $\mathbf{r}'_{\rm all}(t)$ with l_2 regularization by minimizing the loss function

$$\mathcal{L} = \sum_t ||\mathbf{u}_{\rm all}(t) - \mathcal{W}_{\rm out} \cdot \mathbf{r}'_{\rm all}(t)||^2 + \beta ||\mathcal{W}_{\rm out}||^2, \quad \text{(A1)}$$

where $\beta > 0$ is the regularization coefficient—a hyperparameter. The regularized regression can be accomplished by calculating

$$\mathcal{W}_{\rm out} = \mathcal{U} \cdot \mathcal{R}'^T (\mathcal{R}' \cdot \mathcal{R}'^T + \beta \mathcal{I})^{-1}, \quad \text{(A2)}$$

where $\mathcal{I}$ is the identity matrix of dimension D_r, and $\mathcal{U}$ and $\mathcal{R}'$ are the matrix forms of $\mathbf{u}_{\rm all}(t)$ and $\mathbf{r}'_{\rm all}(t)$, respectively, with each column being the values of the vector at a certain t and different rows representing different dimensions, so $\mathcal{U}$ and $\mathcal{R}'$ have $D_{\rm in}$ and D_r rows, respectively. Both $\mathcal{U}$ and $\mathcal{R}'$ have $n_b(T_{\rm train} - T_{\rm cut})$ columns, where $n_b = 3$ is the number of different values of the bifurcation parameter trained on the reservoir, $T_{\rm train}$ is the number of time steps of the training phase for each parameter value, and $T_{\rm cut} = 10$ is the number of time steps removed to ensure that the reservoir system has passed the transient phase.

Validation. For validation, we use the set of training bifurcation parameter values to predict the dynamical evolution of the target system at the same set of parameter values. Specifically, we replace the original input data vector by the output vector $\mathbf{v}(t)$, while keeping the parameter channel intact. The iterative equation [Eq. (1)] becomes

$$\mathbf{r}(t + \Delta t) = (1 - \alpha)\mathbf{r}(t) + \alpha \tanh[\mathcal{A} \cdot \mathbf{r}(t) + \mathcal{W}_{\rm in} \cdot \mathcal{W}_{\rm out} \cdot \mathbf{r}'(t) + k_b \mathcal{W}_b (b + b_0)], \quad \text{(A3)}$$

making the reservoir system a self-evolving dynamical system with external parameter input. During the validation process, for each value of the bifurcation parameter trained, we set the state vector of the hidden nodes at the last step of training, denoted as $\mathbf{r}_{\rm last}$, as the initial condition and let the reservoir system evolve according to Eq. (A3) for a certain number of time steps. The predicted output vector can then be compared with the true vector for these time steps, generating the average RMSE. We calculate the RMSE for all values of the training bifurcation parameters and use the largest RMSE value among them to measure the performance of training. For instance, in this work we are using three bifurcation parameters for training. Thus the validation would first result in three RMSE values for each parameter, and then we only pick the largest one among the three as the validation error. When the target is a spatiotemporal dynamical system, to reduce the computational burden, we calculate the RMSE at each time step and record the minimal time required for the RMSE to reach a certain tolerance threshold. We use this effective prediction time to measure the training performance.

Testing. Testing differs from validation in that we now set the bifurcation parameter to values that have *never* been trained before: these parameter values are completely "new" to the reservoir machine. This generalizability of the reservoir computing scheme is key to predicting transient chaos and system collapse. Specifically, the dynamical evolution of the reservoir machine is still governed by Eq. (A3), but with a different value of the bifurcation parameter b. To set the initial condition for the reservoir machine, we use a short period of the real time series from the target system (e.g., several oscillation cycles), taking from an arbitrary value of the bifurcation parameter in the sustained chaos regime as the input to "warm up" the neural network. After this "warmup," we set the hidden states at the last time step $\mathbf{r}_{\rm last}$ as the initial condition for testing.

Taken together, there are eight hyperparameters: $k_{\rm in}$, n, d, ρ, k_b, b_0, α, and β. Their values for the three examples in this paper are listed in Sec. III in the main text, with the optimization process behind explained in Appendix B below.

APPENDIX B: SENSITIVITY TO HYPERPARAMETERS

A well-trained reservoir machine is key to its ability to predict transient chaos and system collapse. Here by "well trained" we mean that the reservoir, after training, is able to accurately predict the dynamical evolution of the target system for several Lyapunov times with relative error less than 5% for all the selected values of the bifurcation parameter (validation process). As is known [13–28], the dependence of the prediction results on the hyperparameters can be quite sensitive. This is especially so in our study because of the necessity for the reservoir machine to possess the predictive power for several values of the bifurcation parameter. To equip the reservoir machine with the power to predict the behavior of the target system at bifurcation parameter values other than those used in training, it is necessary to optimize the hyperparameters. We have used the Bayesian optimization method [25] that is contained in the PYTHON package "skopt" [49]. An issue is that the optimization algorithm typically gives multiple sets of hyperparameter values. Our solution is to use these hyperparameter values to train multiple reservoirs and obtain the average validation RMSE that can be fed back to the Bayesian algorithm where, for each set of the hyperparameter values, we repeat the training and validation processes multiple times with different random realizations of the reservoir to reduce the fluctuations in RMSE. After several hundreds of iterations of the Bayesian algorithm, we choose the hyperparameter values with the lowest validation RMSE in all the iterations (not necessarily the hyperparameter values from the last iteration). It is worth emphasizing that all the training and validation data used in the optimization process are for the values of the bifurcation parameter before the transition point (in the regime of sustained chaos). That is, the system has no knowledge of the crisis and possible system collapse during the optimization process.

A deficiency of the Bayesian optimization algorithm is that sometimes it generates solutions that are not optimal, which occurs when the solution trajectory is trapped in a local minimum of the landscape of the cost function, especially when the RMSE from the validation process has large fluctuations. An

TABLE I. Hyperparameter values for comparison and the corresponding results of predicted transition point K_c^* in the chaotic food chain system.

Hyperparameters	Set A	Set B	Set C
n	900	900	900
d	4	48	27
ρ	2.3	1.5	1.9
k_{in}	3.6	3.8	3.1
k_b	0.5	0.8	0.7
b_0	−2.2	−1.8	−1.8
α	0.3	0.3	0.3
β	3×10^{-5}	4×10^{-5}	3×10^{-5}
$K_{c,\text{predicted}}$	0.99967	0.99978	0.99975
Relative error, δ	0.9%	0.2%	0.1%
Standard deviation	4.2×10^{-4}	2.4×10^{-4}	3.0×10^{-4}

empirical solution is to run the whole Bayesian optimization process independently a number of times and choose the best result with the smallest error.

A question is this: If one has different sets of hyperparameter values with comparable validation performance, will they have similar testing performance as well? To address this question, we test the chaotic food chain system with three different sets of hyperparameter values, with the results listed in Table I and shown in Fig. 12, where A denotes the set of hyperparameter values used to generate the results in the main text. The quantities examined are the predicted average crisis point and the average transient lifetime beyond the critical point for the bifurcation parameter value $K = K_C + 2 \times 10^{-4}$. We obtain uniformly small errors for all the three sets of hyperparameter values.

Another important issue is whether the hyperparameter optimization would require substantially more data than the data used for training. Our results suggest that the answer is no. The process of hyperparameter optimization is performed by repeating the training and validating processes many times with different values of the hyperparameters, and the values with the lowest validating errors are identified. In fact, optimization can be done for a fixed pair of training and validation data sets. Table II and Fig. 13 show the results of three sets of hyperparameters optimized in this way for the chaotic food chain system, which lead to accurate prediction of both the critical point and the transient lifetime distribution. For the Ikeda map system, the hyperparameter values are optimized with a single data set as well. In general, when the hyperparameter values are not optimized, the predictions are unstable in the sense that they are sensitive to small variations in the parameters with large fluctuations. In contrast, for optimal hyperparameter values, the prediction results are typically stable. It is thus quite feasible to assess if the hyperparameters have been optimized.

APPENDIX C: EFFECTS OF INPUT AND RESERVOIR MATRICES

The elements of the two input matrices $\mathcal{W}_{\text{in}}$ and $\mathcal{W}_b$ as well as those of the reservoir network matrix $\mathcal{A}$ are generated randomly, which can vary among the different realizations of the reservoir machines. We find that the performances during validation and testing are dependent on the matrices, even when learning the same system with the same set of optimal hyperparameter values. The relationship between the elements in the random layers and the reservoir's performance is an important but not yet solved problem. A simple solution to reducing the uncertainties in the prediction is to generate a number of different random realizations of the reservoir and choose the ones with the lowest validation RMSE. In this work, we generate five different reservoir machines at a time and abandon four of them and only record the results from the one left. Besides, suppose that the reservoirs have been well trained in the sense that they have learned the chaotic attractors at the selected values of the bifurcation parameter. When predicting the system dynamics outside the training bifurcation parameter set, fluctuations can still arise among the different reservoir realizations. Conceptually, this can be understood by viewing the prediction problem as some sort of "regression" between the system dynamics (attractor) and the bifurcation parameter. While the reservoir machine has learned the dynamics of the target system at several different

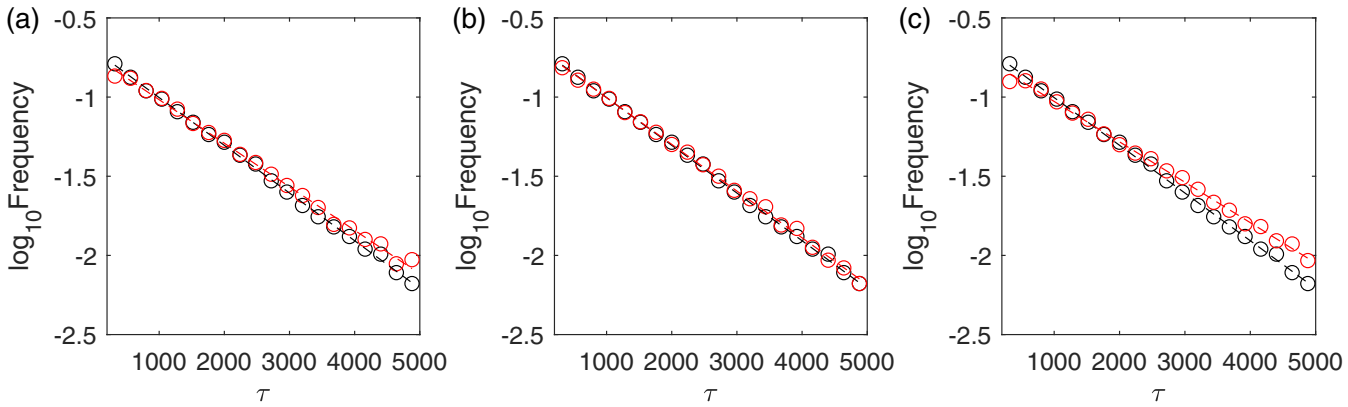

FIG. 12. Results of the transient lifetime distributions with the hyperparameter values from Table I for the chaotic food chain system. (a)–(c) Results from the parameter sets A, B, and C, respectively. Red and black dots represent the predicted and actual distributions, respectively. Dashed lines are the linear fits. The distributions are obtained for $K = K_C + 2 \times 10^{-4}$, where the results in each panel are the averages over 50 different reservoir realizations, each with 400 different initial conditions.

TABLE II. Hyperparameter values optimized by a single pair of training and validating data sets and the corresponding predicted value of the critical transition point K_c^* for the chaotic food chain system.

Hyperparameters	Set D	Set E	Set F
n	900	900	900
d	673	724	109
ρ	1.6	0.48	1.2
k_{in}	2.9	3.0	2.5
k_b	1.5	1.6	2.0
b_0	−1.1	−1.5	−0.97
α	0.51	0.43	0.40
β	7×10^{-5}	6×10^{-5}	9×10^{-5}
$K_{c,\text{predicted}}$	1.00054	1.00051	1.00104
Relative error, δ	8%	8%	13%
Standard deviation	5.1×10^{-4}	4.6×10^{-4}	5.2×10^{-4}

values of the bifurcation parameter, there are infinitely many ways to "connect" these dynamical behaviors (attractors). This bears similarities to the case of fitting three data points with a polynomial: there are infinite possibilities if the polynomial has a degree larger than 2. It is thus necessary to average the prediction results in the testing phase from an ensemble of reservoir realizations. For predicting the critical transition point, we use the simple algebraic average. In particular, for the electrical power and the chaotic food chain systems, we carry out an average over 500 realizations, leading to the relative errors of $\delta = 2\%$ and $\delta = 0.6\%$, respectively. For predicting the average lifetime of the chaotic transient, the simple averaging method can generate results only to the correct order of magnitude, due to the sophisticated relation between the error in the average lifetime and that in the value of the critical point. Another source of error comes from the fact that the dimensionality of the reservoir machine is typically much larger than that of the target chaotic system and, as a result, the reservoir machine may generate "fake" dynamical behaviors that do not arise in the real system. For example, for the electrical power system, the return map from the original system is approximately one dimensional [1] but the reservoir machine may generate a return map with multiple branches, where some branches correspond to sustained but not transient dynamics! We find that many reservoir realizations give the value of the average transient lifetime twice as large as the real value due to the occurrence of the additional branches in the predicted return map that correspond to sustained chaos.

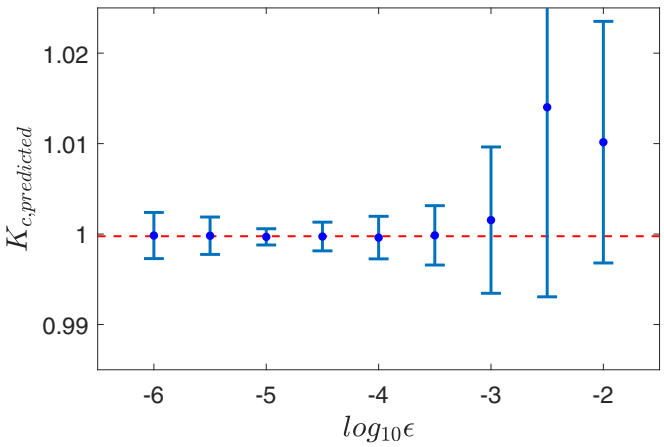

FIG. 14. Accuracy of the predicted values of the critical point K_c^* versus the magnitude of observational noise for the food chain system. The red dashed line represents the real value of the critical point, $K_c = 0.99976$. The error bars are obtained from 400 different random realizations.

APPENDIX D: NOISE RESISTANCE

We study the effect of measurement noise on prediction using the chaotic food chain system. The training time series of all the dynamical variables are subject to additive Gaussian noise of zero mean and standard deviation ϵ. Figure 14 shows the predicted critical transition point versus ϵ, where 400 reservoir realizations are used. It can be seen that, for $\epsilon < 10^{-3}$ (the oscillation amplitude of the target system can often be as small as about 0.2), the predicted critical point is relatively accurate.

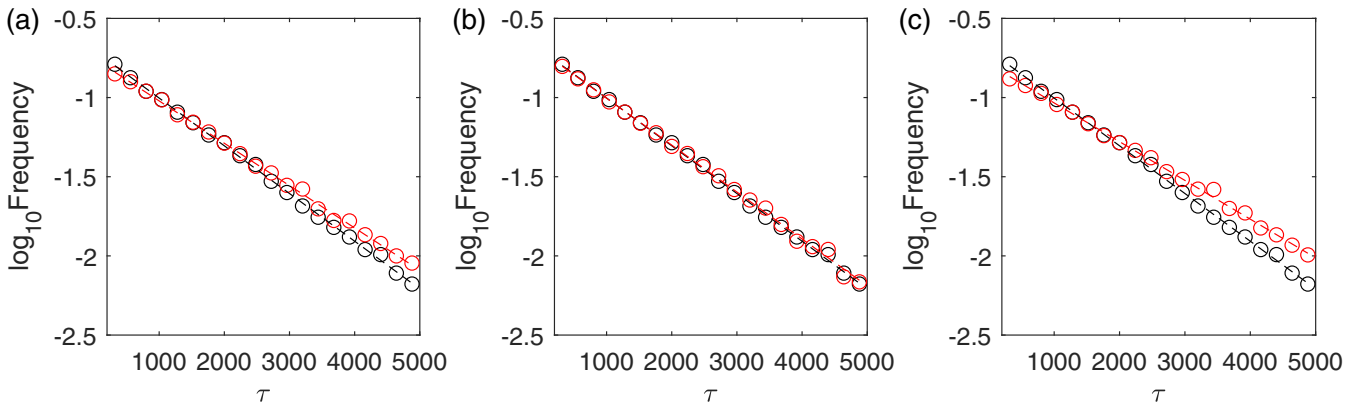

FIG. 13. Results of the transient lifetime distributions with the hyperparameter values from Table II for the chaotic food chain system, which are optimized by a single pair of training and validating sets. (a)–(c) Results from the parameter sets D, E, and F, respectively. Red and black dots represent the predicted and actual distributions, respectively. Dashed lines are linear fits. The distributions are obtained for $K = K_C + 2 \times 10^{-4}$, where the results in each panel are the averages over 50 different reservoir realizations, each with 400 different initial conditions.

[1] M. Dhamala and Y.-C. Lai, Controlling transient chaos in deterministic flows with applications to electrical power systems and ecology, Phys. Rev. E **59**, 1646 (1999).

[2] K. McCann and P. Yodzis, Nonlinear dynamics and population disappearances, Am. Nat. **144**, 873 (1994).

[3] A. Hastings, K. C. Abbott, K. Cuddington, T. Francis, G. Gellner, Y.-C. Lai, A. Morozov, S. Petrivskii, K. Scranton, and M. L. Zeeman, Transient phenomena in ecology, Science **361**, eaat6412 (2018).

[4] C. Grebogi, E. Ott, and J. A. Yorke, Crises, sudden changes in chaotic attractors and chaotic transients, Physica D **7**, 181 (1983).

[5] Y.-C. Lai and T. Tél, *Transient Chaos—Complex Dynamics on Finite Time Scales* (Springer, New York, 2011).

[6] W.-X. Wang, R. Yang, Y.-C. Lai, V. Kovanis, and C. Grebogi, Predicting Catastrophes in Nonlinear Dynamical Systems by Compressive Sensing, Phys. Rev. Lett. **106**, 154101 (2011).

[7] W.-X. Wang, Y.-C. Lai, and C. Grebogi, Data based identification and prediction of nonlinear and complex dynamical systems, Phys. Rep. **644**, 1 (2016).

[8] F. Takens, Detecting strange attractors in fluid turbulence, in *Dynamical Systems and Turbulence*, Lecture Notes in Mathematics Vol. 898, edited by D. Rand and L. S. Young (Springer-Verlag, Berlin, 1981), pp. 366–381.

[9] H. Jaeger, The echo state approach to analysing and training recurrent neural networks—with an erratum note, German National Research Center for Information Technology GMD Technical Report 148, 2001 (unpublished), p. 13.

[10] W. Mass, T. Nachtschlaeger, and H. Markram, Real-time computing without stable states: A new framework for neural computation based on perturbations, Neural Comput. **14**, 2531 (2002).

[11] H. Jaeger and H. Haas, Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication, Science **304**, 78 (2004).

[12] G. Manjunath and H. Jaeger, Echo state property linked to an input: Exploring a fundamental characteristic of recurrent neural networks, Neural Comput. **25**, 671 (2013).

[13] N. D. Haynes, M. C. Soriano, D. P. Rosin, I. Fischer, and D. J. Gauthier, Reservoir computing with a single time-delay autonomous Boolean node, Phys. Rev. E **91**, 020801(R) (2015).

[14] L. Larger, A. Baylón-Fuentes, R. Martinenghi, V. S. Udaltsov, Y. K. Chembo, and M. Jacquot, High-Speed Photonic Reservoir Computing Using a Time-Delay-Based Architecture: Million Words per Second Classification, Phys. Rev. X **7**, 011015 (2017).

[15] J. Pathak, Z. Lu, B. Hunt, M. Girvan, and E. Ott, Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data, Chaos **27**, 121102 (2017).

[16] Z. Lu, J. Pathak, B. Hunt, M. Girvan, R. Brockett, and E. Ott, Reservoir observers: Model-free inference of unmeasured variables in chaotic systems, Chaos **27**, 041102 (2017).

[17] T. Duriez, S. L. Brunton, and B. R. Noack, *Machine Learning Control-Taming Nonlinear Dynamics and Turbulence* (Springer, Berlin, 2017).

[18] Z. Lu, B. R. Hunt, and E. Ott, Attractor reconstruction by machine learning, Chaos **28**, 061104 (2018).

[19] J. Pathak, A. Wilner, R. Fussell, S. Chandra, B. Hunt, M. Girvan, Z. Lu, and E. Ott, Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model, Chaos **28**, 041101 (2018).

[20] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, Model-Free Prediction of Large Spatiotemporally Chaotic Systems from Data: A Reservoir Computing Approach, Phys. Rev. Lett. **120**, 024102 (2018).

[21] T. L. Carroll, Using reservoir computers to distinguish chaotic signals, Phys. Rev. E **98**, 052209 (2018).

[22] K. Nakai and Y. Saiki, Machine-learning inference of fluid variables from data using reservoir computing, Phys. Rev. E **98**, 023111 (2018).

[23] Z. S. Roland and U. Parlitz, Observing spatio-temporal dynamics of excitable media using reservoir computing, Chaos **28**, 043118 (2018).

[24] T. Weng, H. Yang, C. Gu, J. Zhang, and M. Small, Synchronization of chaotic systems and their machine-learning models, Phys. Rev. E **99**, 042203 (2019).

[25] A. Griffith, A. Pomerance, and D. J. Gauthier, Forecasting chaotic systems with very low connectivity reservoir computers, Chaos **29**, 123108 (2019).

[26] J. Jiang and Y.-C. Lai, Model-free prediction of spatiotemporal dynamical systems with recurrent neural networks: Role of network spectral radius, Phys. Rev. Research **1**, 033056 (2019).

[27] P. R. Vlachas, J. Pathak, B. R. Hunt, T. P. Sapsis, M. Girvan, E. Ott, and P. Koumoutsakos, Backpropagation algorithms and Reservoir Computing in Recurrent Neural Networks for the forecasting of complex spatiotemporal dynamics, Neural Networks **126**, 191 (2020).

[28] H. Fan, J. Jiang, C. Zhang, X. Wang, and Y.-C. Lai, Long-term prediction of chaotic systems with machine learning, Phys. Rev. Research **2**, 012080(R) (2020).

[29] C. Zhang, J. Jiang, S.-X. Qu, and Y.-C. Lai, Predicting phase and sensing phase coherence in chaotic systems with machine learning, Chaos **30**, 083114 (2020).

[30] R. Falahian, M. M. Dastjerdi, M. Molaie, S. Jafari, and S. Gharibzadeh, Artificial neural network-based modeling of brain response to flicker light, Nonlinear Dyn. **81**, 1951 (2015).

[31] R. Cestnik and M. Abel, Inferring the dynamics of oscillatory systems using recurrent neural networks, Chaos **29**, 063128 (2019).

[32] Y. Itoh, S. Uenohara, M. Adachi, T. Morie, and K. Aihara, Reconstructing bifurcation diagrams only from time-series data generated by electronic circuits in discrete-time dynamical systems, Chaos **30**, 013128 (2020).

[33] Y. Kuramoto, Diffusion-induced chaos in reaction systems, Prog. Theor. Phys. Suppl. **64**, 346 (1978).

[34] G. I. Sivashinsky, On flame propagation under conditions of stoichiometry, SIAM J. Appl. Math. **39**, 67 (1980).

[35] J. M. Hyman, B. Nicolaenko, and S. Zaleski, Order and complexity in the Kuramoto-Sivashinsky model of weakly turbulent interfaces, Physica D **23**, 265 (1986).

[36] S. L. Pimm and S. Pimm, *The Balance of Nature: Ecological Issues in the Conservation of Species and Communities* (University of Chicago Press, Chicago, 1991).

[37] I. Dobson and H.-D. Chiang, Towards a theory of voltage collapse in electric power systems, Syst. Control Lett. **13**, 253 (1989).

[38] H. O. Wang, E. H. Abed, and A. M. Hamdan, Bifurcations, chaos, and crises in voltage collapse of a model power system, IEEE Trans. Circuits Syst. I Fundam. Theor. Appl. **41**, 294 (1994).

[39] J. Snoek, H. Larochelle, and R. P. Adams, Practical Bayesian optimization of machine learning algorithms, in *Advances in Neural Information Processing Systems*, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Curran Associates, Inc., 2012), Vol. 25, pp. 2951–2959.

[40] M. A. Gelbart, J. Snoek, and R. P. Adams, Bayesian optimization with unknown constraints, in *Uncertainty in Artificial Intelligence - Proceedings of the 30th Conference, UAI 2014*, edited by N. L. Zhang and J. Tian (AUAI Press, 2014), pp. 250–259.

[41] H.-M. Gutmann, A radial basis function method for global optimization, J. Global Optim. **19**, 201 (2001).

[42] R. G. Regis and C. A. Shoemaker, A stochastic radial basis function method for the global optimization of expensive functions, INFORMS J. Comput. **19**, 497 (2007).

[43] Y. Wang and C. A. Shoemaker, A general stochastic algorithmic framework for minimizing expensive black box objective functions based on surrogate models and sensitivity analysis, arXiv:1410.6271.

[44] J. Kennedy and R. Eberhart, Particle swarm optimization, in *Proceedings of ICNN'95—International Conference on Neural Networks*, Vol. 4 (IEEE, Piscataway, NJ, 1995), pp. 1942–1948.

[45] E. Mezura-Montes and C. A. C. Coello, Constraint-handling in nature-inspired numerical optimization: Past, present and future, Swarm Evol. Comput. **1**, 173 (2011).

[46] A. R. Conn, N. I. Gould, and P. Toint, A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds, SIAM J. Numer. Anal. **28**, 545 (1991).

[47] A. Conn, N. Gould, and P. Toint, A globally convergent Lagrangian barrier algorithm for optimization with general inequality constraints and simple bounds, Math. Comput. **66**, 261 (1997).

[48] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. (Addison-Wesley Longman Publishing Co., Inc., USA, 1989).

[49] github.com/scikit-optimize/scikit-optimize