Synchronization of chaotic systems and their machine-learning models

Tongfeng Weng,[1,*] Huijie Yang,[1] Changgui Gu,[1] Jie Zhang,[2] and Michael Small[3,4,†]
[1]*Business School, University of Shanghai for Science and Technology, Shanghai 200093, People's Republic of China*
[2]*Institute of Science and Technology for Brain-Inspired Intelligence, Fudan University, Shanghai 200433, China*
[3]*Complex Systems Group, Department of Mathematics and Statistics, The University of Western Australia, Crawley, Western Australia 6009, Australia*
[4]*Mineral Resources, CSIRO, Kensington, Western Australia 6151, Australia*

(Received 19 September 2018; revised manuscript received 16 February 2019; published 5 April 2019)

Recent advances have demonstrated the effectiveness of a machine-learning approach known as "reservoir computing" for model-free prediction of chaotic systems. We find that a well-trained reservoir computer can synchronize with its learned chaotic systems by linking them with a common signal. A necessary condition for achieving this synchronization is the negative values of the sub-Lyapunov exponents. Remarkably, we show that by sending just a scalar signal, one can achieve synchronism in trained reservoir computers and a cascading synchronization among chaotic systems and their fitted reservoir computers. Moreover, we demonstrate that this synchronization is maintained even in the presence of a parameter mismatch. Our findings possibly provide a path for accurate production of all expected signals in unknown chaotic systems using just one observational measure.

DOI: 10.1103/PhysRevE.99.042203

Synchronization of chaotic systems is a fundamental problem of nonlinear science which has attracted continuous interest over several decades [1–4]. This activity can be traced back to Pecora and Carroll's seminal work in which they found that synchronization can be realized by linking two chaotic systems with a common signal [1]. Subsequent works have described several types of synchronization features ranging from complete synchronization [5], phase synchronization [6], and lag synchronization [7] to generalized synchronization [8]. Meanwhile, intensive studies have demonstrated the potential applications of chaos synchronization in communication [9] and biological systems [3].

In general, previous studies of chaos synchronization rely on the fact that the equations of chaotic systems are known beforehand. This restriction is in practice impossible for dealing with real chaotic systems for which only one or some observational signals are usually available [10]. Recently, great progress has been achieved in model-free prediction of chaotic systems from data using a reservoir computing approach [11–14]. A growing number of studies have demonstrated that this effective technique can predict well low-dimensional chaotic systems [11], extract Lyapunov exponents from real data [12,13], and even capture the evolutionary rule of large spatiotemporally chaotic systems [14]. Here we take advantage of this machine-learning approach for modeling chaotic systems whose equations of motion are unknown. We find that by transmitting just one scalar signal, the well-trained reservoir computer will synchronize with the learned chaotic system. Meanwhile, we show that, following the same manner, synchronization of trained reservoir computers and a cascading synchronization can also be achieved. Results on two benchmark chaotic systems (i.e., the Rössler and Lorenz systems) confirm our findings, which suggests a new way of accurately constructing all signals in real chaotic systems using limited observational measures.

We start by introducing the reservoir computering approach for modeling chaotic systems. The basic architecture of a reservoir computer comprises an input layer, a reservoir network having N dynamical reservoir nodes, and a linear output layer. Here we follow Jaeger's design [11] and define the update equation of the state vector $\mathbf{r}$ of the reservoir network as follows:

$$\mathbf{r}(t+1) = (1-\alpha)\mathbf{r}(t) + \alpha\tanh\left(\mathbf{Ar(t)} + \mathbf{W}_{\text{in}}\begin{bmatrix} b_{\text{in}} \\ \mathbf{u}(t)\end{bmatrix}\right), \tag{1}$$

where $\mathbf{A}$ is the adjacency matrix of the reservoir network and $\mathbf{u}$ is the input vector fed into the reservoir network via the input weighted matrix $\mathbf{W}_{\text{in}}$. The parameter α is a "leakage" rate lying in the range (0,1) and $b_{\text{in}} = 1$. In addition to the input vector $\mathbf{u}$, the reservoir dynamics mainly depend on the matrices $\mathbf{W}_{\text{in}}$ and $\mathbf{A}$. Empirically, the elements in matrix $\mathbf{W}_{\text{in}}$ are drawn from a uniform distribution $[-\sigma, \sigma]$, while $\mathbf{A}$ is built from a sparse random Erdös-Rényi matrix whose nonzero entries are drawn randomly from a uniform distribution $[-1, 1]$. Consequently, the output vector $\mathbf{y}$ of the reservoir system is taken to be a linear function of the reservoir state and the input vector such that

$$\mathbf{y}(t) = \mathbf{W}_{\text{out}}\begin{bmatrix} b_{\text{out}} \\ \mathbf{u}(t) \\ \mathbf{r}(t)\end{bmatrix}, \tag{2}$$

where $\mathbf{W}_{\text{out}}$ is the solely adjusted matrix in the training stage and $b_{\text{out}} = 1$. During the training process, the reservoir

*wtongfeng2006@163.com
†michael.small@uwa.edu.au

computer first evolves based on Eq. (1) for τ time steps to eliminate transient states of the reservoir network. Suppose that $\{\mathbf{s}(t)|t = -\tau, -\tau + 1, \ldots, T\}$ is the training data for which we use $\mathbf{s}(t)$ to predict next forward $\mathbf{s}(t+1)$. Previous studies have shown that the output weights $\mathbf{W}_{\rm out}$ can be analytically given by [15,16]

$$\mathbf{W}_{\rm out} = \mathbf{Y}\mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top + \lambda\mathbb{I})^{-1}. \tag{3}$$

Here $\mathbb{I}$ is an identity matrix, λ is the ridge regresion parameter for avoiding overfitting, and $\mathbf{X}$ (respectively, $\mathbf{Y}$) is the matrix whose kth column is $[b_{\rm out}; \mathbf{s}(k); \mathbf{r}(k)]$ [respectively, $\mathbf{s}(k+1)$]. After training, the reservoir system can run autonomously based on Eqs. (1) and (2), where $\mathbf{y}(t)$ is adopted to approximate $\mathbf{u}(t)$.

We now address the reservoir computing approach for modeling a chaotic Rössler system,

$$dx/dt = -y - z, \tag{4}$$

$$dy/dt = x + ay, \tag{5}$$

$$dz/dt = b + z(x - c), \tag{6}$$

where $a = 0.2$, $b = 0.2$, and $c = 9.0$. We calculate a numerical solution for this system via the fourth-order Runge-Kutta technique and record 7200 time points with time step $\Delta t = 0.1$ after discarding the leading 4000 observations (to eliminate transient states). We use the first 2600 points with input vector $\mathbf{u} = (x, y, z)$ for training the reservoir computer with $\alpha = 0.5$, $N = 500$, $\tau = 100$, $\sigma = 1$, and $\lambda = 1 \times 10^{-8}$. The

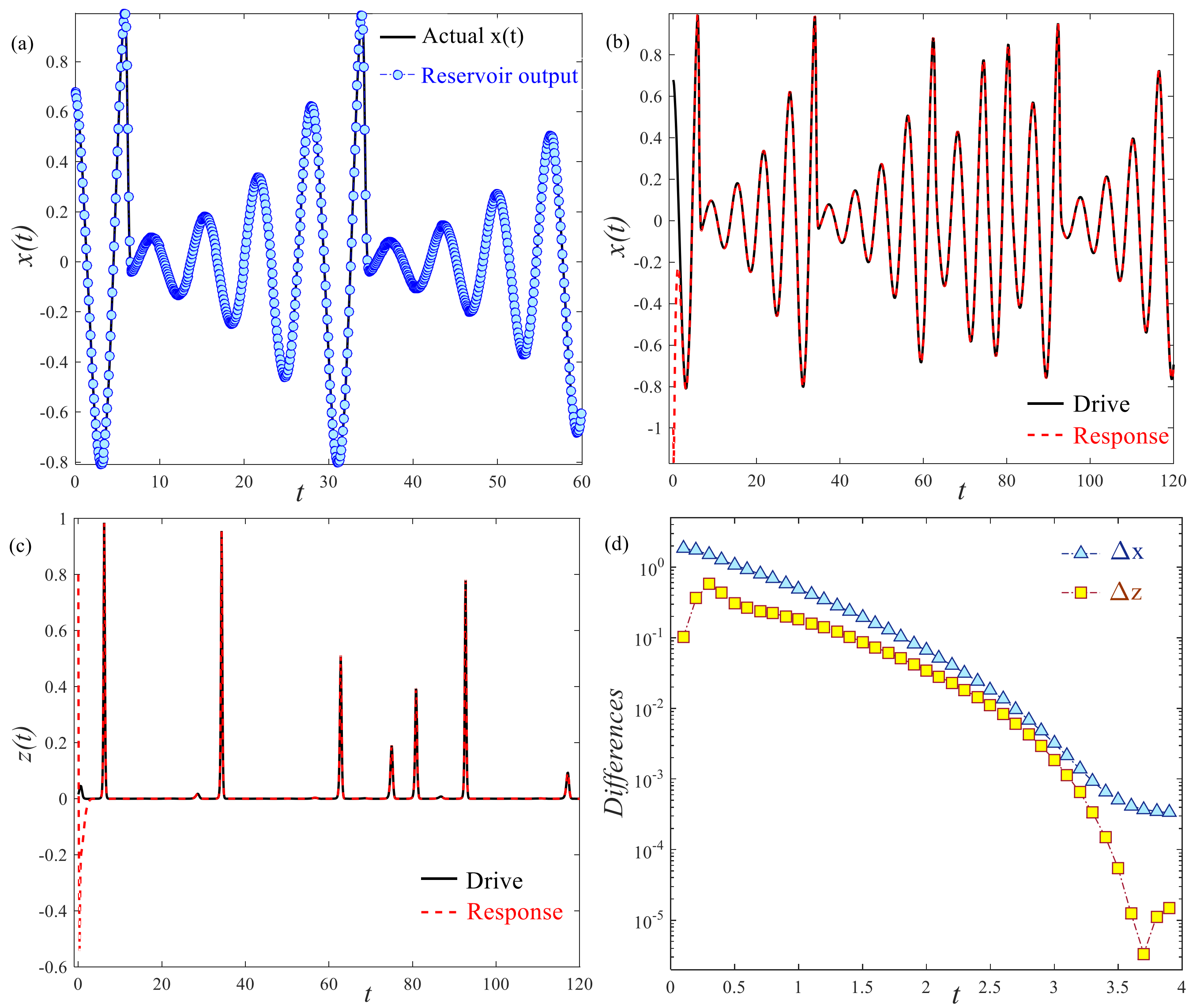

FIG. 1. (a) Prediction output of the trained reservoir computer overlaid with actual data from the Rössler system. (b) Complete synchronization of the Rössler system and the trained reservoir computer; the solid line (dashed line) represents x (x') in function of time. (c) Complete synchronization of the Rössler system and the trained reservoir computer in terms of the z variable. (d) The differences Δx and Δz between the response variables of the trained reservoir computer and their drive counterparts for the Rössler system.

trained reservoir system can infer the trajectory of the chaotic Rössler system accurately, where we show the x variable as an example, see Fig. 1(a). Then we use the y component of the Rössler system as the driving signal for which the input vector fed into the training reservoir computer becomes $\mathbf{u} = (x', y, z')$, where the initial values of x'_0 and z'_0 are chosen randomly. Based on Eqs. (1) and (2), we can generate the subsequent values x' and z' autonomously for which the output vector $\mathbf{y}(t) = (x'_t, y'_t, z'_t)$ of the reservoir computer is used as the next input vector $\mathbf{u}(t+1) = (x'_t, y_t, z'_t)$ with y'_t replacing by the driving signal y_t. Interestingly, we find that a response system given by the trained reservoir system will synchronize with a driving system given by the Rössler system. Specifically, we show the rapid convergence of the response to the drive for x and z components in Figs. 1(b) and 1(c). This is further supported by observing the differences $\Delta x = |x' - x|$ and $\Delta z = |z' - z|$, which converge toward zeros, as described in Fig. 1(d). Note that x' and z' are generated from the response system (i.e., the trained reservoir system). Results show that synchronization of the trained reservoir computer and the chaotic system can be achieved by sending a common signal.

We further investigate the above phenomena in the Lorenz system, which is given by

$$dx/dt = \sigma(y - z), \tag{7}$$

$$dy/dt = -xz + rx - y, \tag{8}$$

$$dz/dt = xy - bz. \tag{9}$$

We choose the parameters $(\sigma, b, r) = (10, 8/3, 60)$ to be in the chaotic regime and generate 1×10^4 data points via the fourth-order Runge-Kutta technique with step size $\Delta t = 0.02$. We use the first 2600 points with input vector $\mathbf{u} = (x, y, z)$ for training the reservoir computer with $\alpha = 0.2$, $N = 500$, $\tau = 100$, and $\lambda = 1 \times 10^{-8}$. Following the same procedure as before, we build a drive-response configuration using the x component as the driving signal. Clearly, it is shown that the variables of the response given by the trained reservoir computer rapidly approach that of the drive system, as described in Figs. 2(a) and 2(b). Moreover, we can see that although the response reservoir computer starts far from the

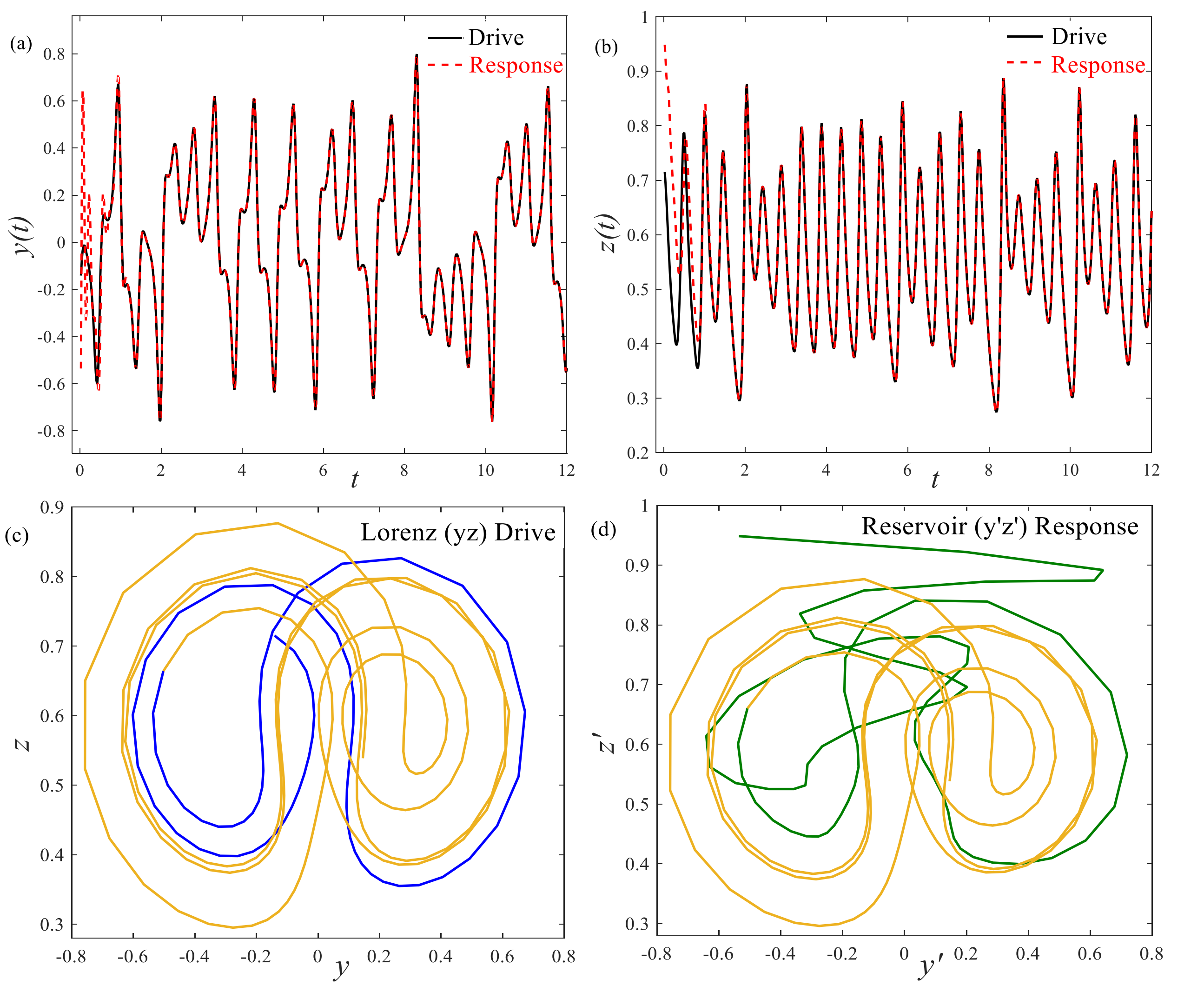

FIG. 2. Synchronization of the Lorenz system and the trained reservoir computer with the x-driven configuration: (a) Solid line (dashed line) represents y (y') in function of time. (b) Solid line (dashed line) represents z (z') in function of time. Trajectories of the x-driven (y, z) subsystems from (c) the drive and (d) the response. The initial transient states are shown by the blue line (c) and the green line (d), respectively.

TABLE I. A listing of the various subsystems and driving components of the Lorenz and Rössler systems and their sub-Lyapunov exponents reported in Refs. [1,4].

System	Drive	Response	Sub-Lyapunov exponents
Rössler	x	(y, z)	$(-8.89, 0.20)$
$a = 0.2, b = 0.2$	y	(x, z)	$(-8.81, -0.056)$
$c = 9.0$	z	(x, y)	$(0.10, 0.10)$
Lorenz	x	(y, z)	$(-1.86, -1.81)$
$\rho = 10, b = 8/3$	y	(x, z)	$(-9.99, -2.67)$
$r = 60$	z	(x, y)	$(-11.0, 0.01)$

drive values it soon spirals into the same type of the Lorenz (y, z) subsystem attractor [see Figs. 2(c) and 2(d)]. These findings further confirm that by transmitting a common signal, a trained reservoir computer can synchronize with its learned chaotic system. Note that in this case, synchronization will occur for either x or y driving, but z driving did not work. This can be referred to the sub-Lyapunov exponents as shown in Table I, where there is one positive conditional Lyapunov exponent for z driving. So this subsystem (x, y) is unstable which results in desynchrony [1,3]. It is a similar situation for the Rössler system, but only the y-driven configuration will synchronize, as shown in Fig. 1.

Moreover, we show that it is possible to maintain excellent synchronization between chaotic systems and their trained reservoir computers even when there is a parameter mismatch between them. Here, by parameter mismatch, we mean that the original dynamical system for training the reservoir computer is not the same dynamical system as the driving system. To address this, we take the previous Lorenz system in the presence of significant parameter mismatch as an

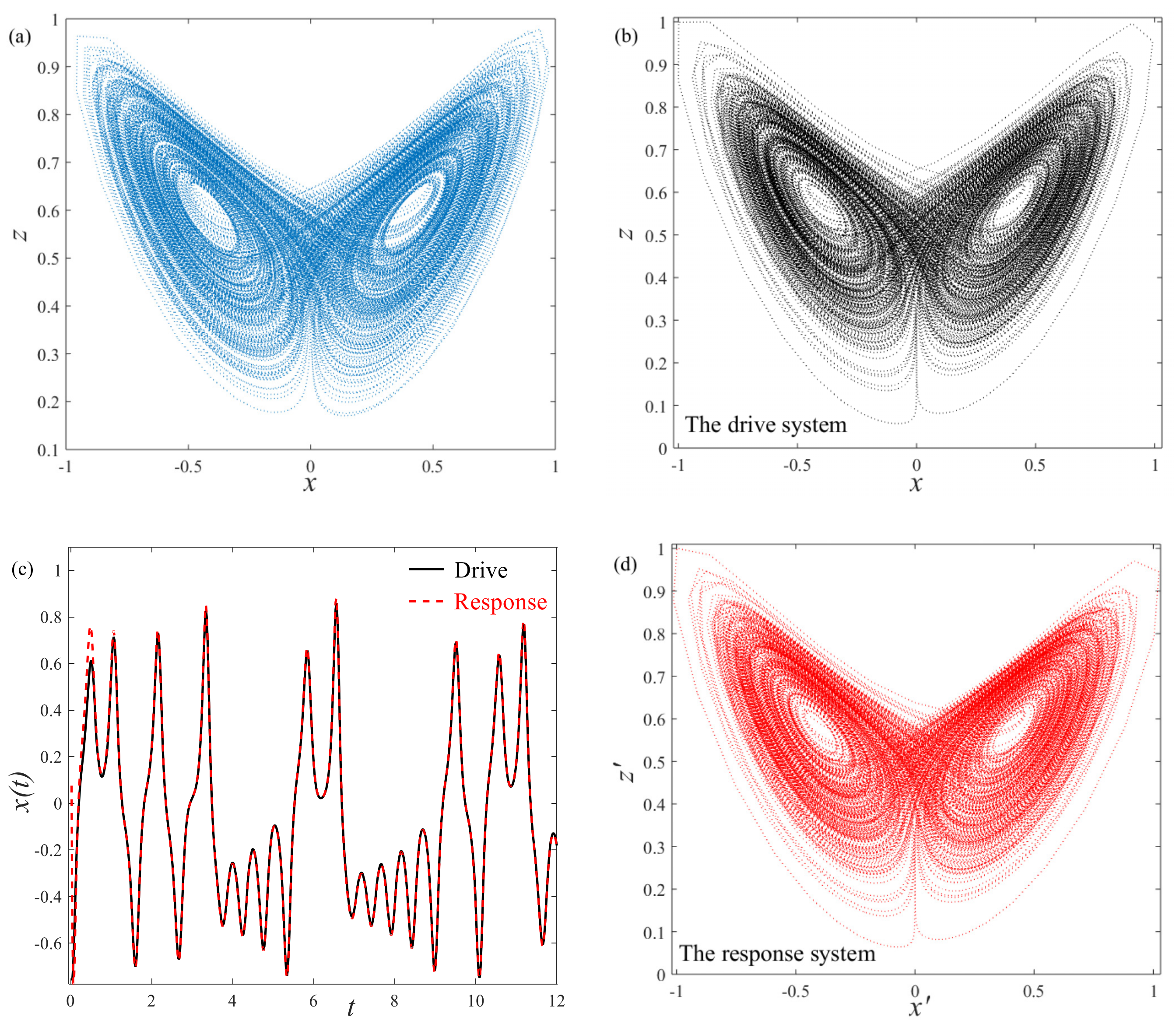

FIG. 3. The (x, z) subsystem attractors of (a) the original dynamical system for training the reservoir computer and (b) the drive system. (c) Complete synchronization of the Lorenz system and the trained reservoir computer in terms of the x variable in the case of the parameter mismatch. (d) The (x', z') subsystem attractor of the response system.

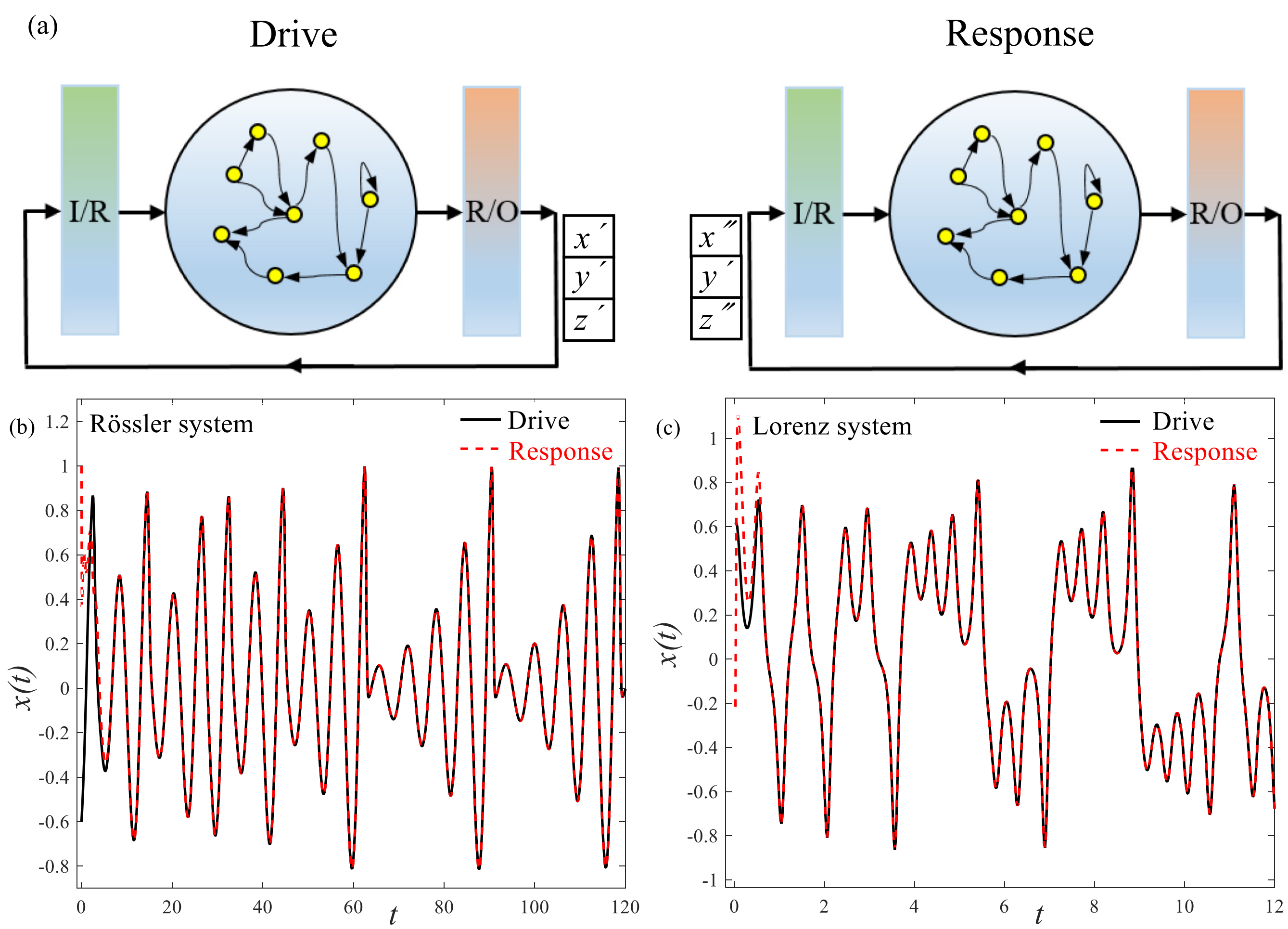

FIG. 4. (a) Schematic illustrations of our strategy to synchronize trained reservoir computers to the y' drive. Synchronization of the trained reservoir computers; the solid line (dashed line) represents x' (x'') in function of time for (b) the Rössler system and (c) the Lorenz system.

example. Specifically, we choose the parameters $(\sigma, b, r) = (10, 8/3, 60)$ for training the reservoir computer, while the parameters $(\sigma, b, r) = (10, 8/3, 50)$ are used for the drive system. In Figs. 3(a) and 3(b) we show the (x, z) subsystem attractors of the training and drive systems at different parameter settings. Interestingly, we find that by sending the y signal for driving, the x variable of the trained reservoir computer quickly approaches that of the drive system, see Fig. 3(c). Results show that synchronization between chaotic systems and their learned reservoir computers can be achieved even under a condition of substantially mismatched parameters. This is further supported by observing the response (x', z') subsystem, which successfully reproduces the drive subsystem with little perceivable distortion [see Fig. 3(d)].

Besides synchronization of a trained reservoir computer and its learned chaotic system, we further illustrate that these trained reservoir computers can also be synchronized. This is achieved by sharing a common variable of the trained reservoir computer, as described in Fig. 4(a). In particular, after training, the updated equation of the reservoir computer can be described as

$$\mathbf{u}(t+1) = g[\mathbf{u}(t)], \tag{10}$$

where $\mathbf{u} = (x', y', z')$ is the input vector. Now create another reservoir computer identical to this one and substitute the variable y' for the corresponding y'' in the function g with this new reservoir system, giving

$$\mathbf{u}'(t+1) = g[\mathbf{u}'(t)], \tag{11}$$

where $\mathbf{u}' = (x'', y', z'')$ is the input vector for which the initial values x''_0 and z''_0 are chosen randomly. The component y' is used as the driving signal linking them. At each time step, the output vector (x''_t, y''_t, z''_t) is used as the input vector $(x''_{t+1}, y'_{t+1}, z''_{t+1})$ with y''_{t+1} replaced by the driving signal y'_{t+1}. We test this idea on the previous selected chaotic systems (i.e., the Rössler system and the Lorenz system). Results show that although from different initial conditions, x'' quickly converges to x' as time progresses [Figs. 4(b) and 4(c)]. These findings reveal that synchronization of trained reservoir computers can be realized through linking with a common signal.

Finally, a cascaded synchronization is an important branch in the area of synchronization in chaotic systems [4,17]. We show that a cascading synchronization among chaotic systems and their trained reservoir computers can also be achieved. Here we test the cascaded scheme using the previous selected

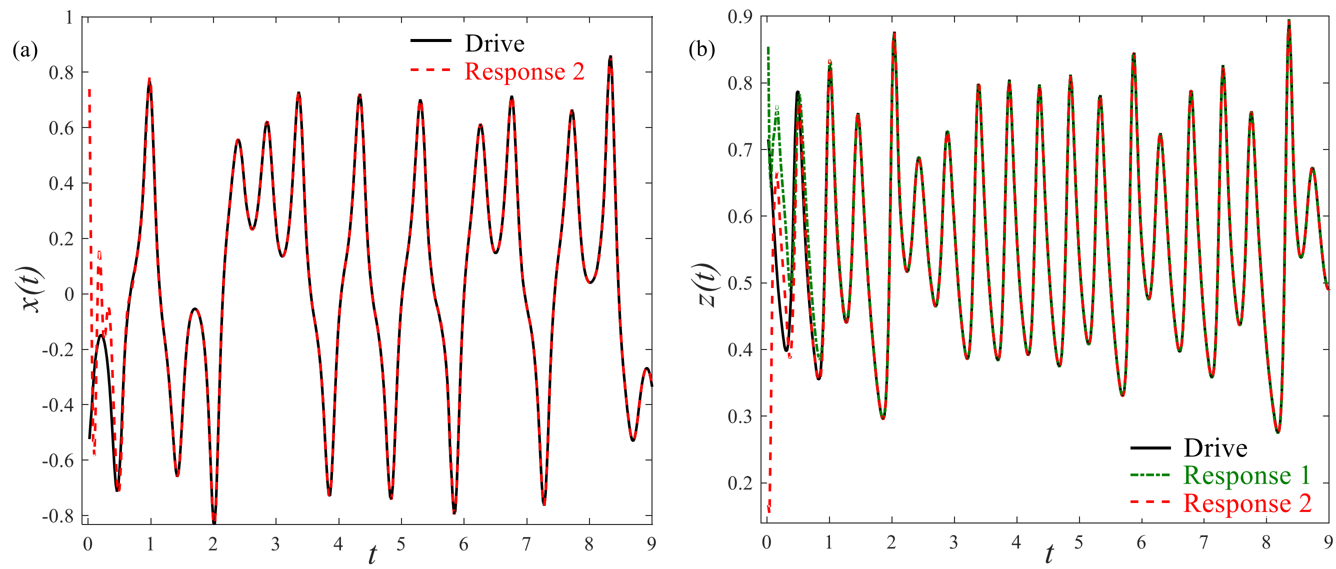

FIG. 5. (a) Convergence of the cascaded reservoir computer's x'' variable to the drive Lorenz systems's x variable. (b) Both the z'' of the second response and the z' of the first rsponse synchronize with the z variable of the Lorenz system.

Lorenz system which has two stable subsystems [17]. Specifically, we use the x component of the Lorenz system as a common signal to link the Lorenz system and first trained reservoir computer (named response 1), while from response 1 to the second reservoir computer (named response 2), y' is the driving signal as described by Eqs. (10) and (11). It is shown that the x'' variable of the second response rapidly approaches the x variable of the original Lorenz system as time progresses [Fig. 5(a)]. This finding implies that we can reproduce the incoming drive signal (i.e., x variable) since the second reservoir computer in the cascade has the original drive component as part of its response. Moreover, Fig. 5(b) shows that the three signals of the z component generated from the driving and response systems match exactly in the course of time, as expected.

In summary, we adopt a machine-learning technique using the reservoir computing approach for modeling chaotic systems. Besides the effectiveness of model-free prediction [14], we find that the trained reservoir computers can synchronize with their learned chaotic systems by sharing a common signal. We show that this synchronization is robust even in the case of a large parameter mismatch. Remarkably, we find that synchronization of the trained reservoir computers can also be achieved by sharing one common signal. Moreover, we further extend our work to cascading synchronization among chaotic systems and their fitted reservoir computers. Our findings show that by taking advantage of the reservoir computing approach, synchronization of chaotic systems whose equations of motion are unknown can be achieved. Our work opens interesting opportunities for accurately producing all signals of real dynamical systems for which limited data information is available.

We note that there is a long tradition of inferring a desired variable from known measurements in the control theory literature, where it is known as the observer problem [18–20]. However, most previous studies rely on the fact that they have a sufficiently accurate mathematical model of the chaotic system of interest. Here we investigate nonlinear dynamical systems for which mathematical models of them are unavailable by taking advantage of the reservoir computer technique for modeling. On this point, the motivation for our technique is similar to that of the "reservoir observer" reported in Ref. [21] but from different angles and with different channels. Specifically, the method in Ref. [21] treats the available measurements as the input vector to predict the desired variable directly for which the output of a trained reservoir computer accurately agrees with the desired variable for a long time and then gradually diverges. In contrast, we explore this problem from the synchronization perspective, where the prediction of a trained reservoir computer matches the desired signal exactly as time progress.

Note that the Matlab code used in this paper for Ref. [22].

T.F.W. was supported by National Science Foundation of China Grant No. 11805128. C.G.G. was supported by National Science Foundation of China Grants No. 11875042 and 11505114. This work is supported by the Shanghai project for construction of top disciplines under Grant No. USST-SYS-01 (H.Y., C.G., and T.W.).

[1] L. M. Pecora and T. L. Carroll, Phys. Rev. Lett. **64**, 821 (1990).
[2] Y. C. Lai and C. Grebogi, Phys. Rev. E **47**, 2357 (1993).
[3] S. Boccaletti, J. Kurths, G. Osipov, D. L. Valladares, and C. S. Zhou, Phys. Rep. **366**, 1 (2002).

[4] L. M. Pecora and T. L. Carroll, Chaos **25**, 097611 (2015).
[5] L. M. Pecora and T. L. Carroll, Phys. Rev. A **44**, 2374 (1991).
[6] M. G. Rosenblum, A. S. Pikovsky, and J. Kurths, Phys. Rev. Lett. **76**, 1804 (1996).
[7] M. G. Rosenblum, A. S. Pikovsky, and J. Kurths, Phys. Rev. Lett. **78**, 4193 (1997).
[8] L. Kocarev and U. Parlitz, Phys. Rev. Lett. **76**, 1816 (1996).
[9] C. Ling and S. Sun, IEEE Trans. Commun. **46**, 1433 (1998).
[10] S. Boccaletti, D. L. Valladares, L. M. Pecora, H. P. Geffert, and T. Carroll, Phys. Rev. E **65**, 035204(R) (2002).
[11] H. Jaeger and H. Haas, Science **304**, 78 (2004).
[12] J. Pathak, Z. X. Lu, B. R. Hunt, M. Girvan, and E. Ott, Chaos **27**, 121102 (2017).
[13] Z. X. Lu, B. Hunt, and E. Ott, Chaos **28**, 061104 (2018).
[14] J. Pathak, B. Hunt, M. Girvan, Z. X. Lu, and E. Ott, Phys. Rev. Lett. **120**, 024102 (2018).
[15] M. Lukoševičius and H. Jaeger, Comput. Sci. Rev. **3**, 127 (2009).
[16] R. S. Zimmermann and U. Parlitz, Chaos **28**, 043118 (2018).
[17] L. M. Pecora and T. L. Carroll, Physica D (Amsterdam) **67**, 126 (1993).
[18] R. Hermann and A. J. Krener, IEEE Trans. Autom. Control **22**, 728 (1977).
[19] P. So, E. Ott, and W. P. Dayawansa, Phys. Rev. E **49**, 2650 (1994).
[20] J. Schumann-Bischoff, S. Luther, and U. Parlitz, Phys. Rev. E **94**, 032221 (2016).
[21] Z. X. Lu, J. Pathak, B. Hunt, M. Girvan, R. Brockett, and E. Ott, Chaos **27**, 041102 (2017).
[22] http://staffhome.ecm.uwa.edu.au/00027830/code/reservoir.rar