Learning Hamiltonian dynamics with reservoir computing

Han Zhang, Huawei Fan, Liang Wang, and Xingang Wang*
School of Physics and Information Technology, Shaanxi Normal University, Xi'an 710119, China

(Received 24 April 2021; accepted 19 July 2021; published 6 August 2021)

Reconstructing the Kolmogorov-Arnold-Moser (KAM) dynamics diagram of Hamiltonian system from the time series of a limited number of parameters is an outstanding question in nonlinear science, especially when the Hamiltonian governing the system dynamics is unknown. Here we demonstrate that this question can be addressed by the machine learning approach knowing as reservoir computing (RC). Specifically, we show that without prior knowledge about the Hamilton equations of motion, the trained RC is able to not only predict the short-term evolution of the system state, but also replicate the long-term ergodic properties of the system dynamics. Furthermore, using the architecture of parameter-aware RC, we show that the RC trained by the time series acquired at a handful parameters is able to reconstruct the entire KAM dynamics diagram with a high precision by tuning a control parameter externally. The feasibility and efficiency of the learning techniques are demonstrated in two classical nonlinear Hamiltonian systems, namely, the double-pendulum oscillator and the standard map. Our study indicates that, as a complex dynamical system, RC is able to learn from data the Hamiltonian.

DOI: 10.1103/PhysRevE.104.024205

I. INTRODUCTION

Model-free prediction of chaotic dynamical systems using machine learning approaches has received broad research interest in recent years [1–5]. In particular, a technique knowing as reservoir computing (RC) has been widely adopted in the literature for predicting the state evolution of chaotic systems [6–16]. From the perspective of dynamical systems, RC can be regarded as a complex network of coupled dynamical elements, which, driven by the input data, generates the output data through a readout function. Except the parameters of the readout function, which are to be determined by the training process, all other parameters and the dynamics of RC are fixed at the construction. After training, the system is closed by using the output as the input, and then is evolving as an autonomous system for predictions. Evidence has shown that compared with the conventional prediction techniques in nonlinear science, RC has a clear advantage in both accuracy and efficiency. For instance, it is demonstrated that a well-trained RC can accurately predict the state evolution of chaotic systems for about a half-dozen Lyapunov times, which is much longer than the prediction horizon of the conventional techniques [6].

Besides predicting chaos evolutions, RC has also been exploited to address other long-standing questions in nonlinear science, such as reconstructing chaotic attractors and calculating Lyapunov exponents [17], synchronizing chaotic oscillators [8,18], predicting system collapses [19], reconstructing synchronization transition paths [20], and transferring knowledge between different systems [21,22], to name just a few. These studies, while demonstrating the power of RC in solving different nonlinear questions, also give insights into the working mechanisms of RC. For instance, although RC fails to predict the long-term evolution of chaotic systems, it can replicate the ergodic properties of the chaotic systems faithfully, e.g., the Lyapunov exponents and returned maps [17]. This ability, known as climate replication, suggests that it is the intrinsic dynamics of the chaotic system that RC essentially learns from the data, instead of the mathematical expressions describing the time series. Exploiting this ability, model-free techniques have been proposed in recent years to predict the bifurcation behaviors of nonlinear dynamical systems, e.g., reproducing the bifurcation diagram of classical chaotic systems [13,14], anticipating the critical points of system collapse [19], predicting the critical coupling for synchronization [20], etc. Another property revealed recently in exploiting RC is that knowledge can be transferred between different dynamical systems, namely, the ability of transfer learning [21,22]. Specifically, it is shown that the RC trained by the time series of system A can be used to infer the properties of system B, with the motions of A and B significantly different from each other. It is worth mentioning that the systems employed in these studies are dominantly dissipative, in the sense that the final dynamics of the system is independent on the initial conditions. The adoption of dissipative systems is natural, as the reservoir network itself can be regarded as a dissipative system. In particular, the memory-fading property of RC requires that the dynamics of the reservoir network should be converged onto a low-dimensional manifold independent of the network initial conditions [2].

From the astronomical scales to the quantum scales, many physical systems are described by the Hamiltonian formalism. Different from dissipative systems, the symplectic structure of the Hamiltonian requires that during the time course of system evolution, the phase-space volume of a closed surface should be preserved and, for a time-independent Hamiltonian, the total energy of the system should be conserved [23]. The concerns about volume preservation and energy conservation have led to the development of physics-enhanced

*wangxg@snnu.edu.cn

machine learning techniques in recent years, for instance, Hamiltonian neural networks (HNNs) [24–33]. The idea of HNNs was articulated about three decades ago [34], when methods for designing neural networks of Hamiltonian dynamics were proposed. Recent studies generalize this idea by incorporating the Hamiltonian mechanisms into the conventional approaches in machine learning, e.g., deep learning, resulting in improved learning performance [24–28]. The idea of HNNs has been also extended to systems described by Lagrangian formalism and generalized coordinates recently [30–32]. In particular, a parameter-aware architecture has been proposed for reconstructing the Kolmogorov-Arnold-Moser (KAM) dynamics diagram of Hamiltonian systems, in which the adaptability of HNNs has been explored [33]. It is worth mentioning that while HNNs have been proved to be efficient for learning Hamiltonian dynamics, the training of HNNs is time-consuming as compared with RC. For instance, due to the multiple-layered network structure, the number of parameters to be trained in HNNs is orders of magnitude larger to that of RC. In addition, in designing HNNs, physics constraints (e.g., the Hamilton equations of motion) are enforced to the machine, yet it remains unclear whether such an enforcement is necessary, and nor do we understand completely the mechanism underlying the improved performance.

From the standpoint of state evolution prediction, the mission of RC and HNNs is very similar, i.e., mimicking the function relating the input and output. Whereas the output of RC does not satisfy the symplectic properties of the Hamiltonian, a high-dimensional reservoir does have the ability to fit any form of function with a proper training of the output matrix [1]. Once the statistical properties of the time series are captured by the output of RC, the climate of the Hamiltonian dynamics will be replicated. Furthermore, using the property of transfer learning, we might also be able to reconstruct the KAM dynamics diagram with the RC trained at a limited number of parameters. If it is indeed the case, the computational cost as required by HNNs will be significantly reduced. Our main objective in the present work is just to verify this speculation. To be specific, using a recently proposed architecture of parameter-aware RC [19,20], we train RC with the time series of Hamiltonian systems acquired at a handful of parameters, and then use the trained RC to replicate the dynamics climates of different parameters. We are able to show that the trained RC not only is able to replicate the climates associated with the training parameters, but also is able to generate the entire KAM dynamics diagram with a high precision by tuning a control parameter externally.

In the following section, we will present the architecture of parameter-aware RC used for predicting Hamiltonian dynamics, together with the training method. The application of RC to the prediction of Hamiltonian dynamics and the reconstruction of the KAM diagram for two classical Hamiltonian systems, the double-pendulum oscillator and the standard map, will be reported in Sec. III. Discussion and a conclusion will be given in Sec. IV.

II. PARAMETER-AWARE RESERVOIR COMPUTING

We adopt the architecture of parameter-aware RC to learn the dynamics of Hamiltonian systems [19,20]. In this architecture, the RC is constituted by four modules: the I/R layer (input-to-reservoir), the parameter-control module, the reservoir network, and the R/O layer (reservoir-to-output). The I/R layer is characterized by the matrix $\mathbf{W}_{\text{in}} \in \mathbb{R}^{D_r \times D_{\text{in}}}$, which couples the input vector $\mathbf{u}_\beta(t) \in \mathbb{R}^{D_{\text{in}}}$ to the reservoir network. Here $\mathbf{u}_\beta(t)$ denotes the input vector that is acquired from the target system at time t and under the specific system parameter β. The elements of $\mathbf{W}_{\text{in}}$ are randomly drawn from a uniform distribution within the range $[-\sigma, \sigma]$. The parameter-control module is characterized by the vector $\mathbf{s} = \beta\mathbf{b}$, with β the control parameter and $\mathbf{b} \in \mathbb{R}^{D_r}$ the bias vector. In applications, the control parameter β can be regarded as an additional input channel marking the input vector $\mathbf{u}(t)$. In our studies, we choose β to be the initial conditions of the Hamiltonian system, as the variation of which can lead to different dynamical motions. The elements of $\mathbf{b}$ are drawn randomly from a uniform distribution within the range $[-\sigma, \sigma]$. The reservoir network contains D_r dynamical nodes, with the initial states of the nodes being randomly chosen from the interval $[-1, 1]$. The states of the nodes in the reservoir network, $\mathbf{r}(t) \in \mathbb{R}^{D_r}$, are updated according to the equation

$$\mathbf{r}(t + \Delta t) = (1 - \alpha)\mathbf{r}(t) + \alpha \tanh[\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{\text{in}}\mathbf{u}_\beta(t) + \beta\mathbf{b}]. \tag{1}$$

Here Δt is the time step for updating the reservoir, α is the leaking coefficient, and $\mathbf{A} \in \mathbb{R}^{D_r \times D_r}$ is the weighted adjacency matrix representing the coupling relationship between nodes in the reservoir. The adjacency matrix $\mathbf{A}$ is constructed as a sparse random Erdös-Rényi matrix: with the probability d, each element of the matrix is assigned a nonzero value drawn randomly from the interval $[-1, 1]$. The matrix $\mathbf{A}$ is rescaled to make its spectral radius equal to ρ. Before training, the reservoir is evolved for a transient period of T_0, so as to avoid the influence induced by the initial states of the nodes. The output layer is characterized by the matrix $\mathbf{W}_{\text{out}} \in \mathbb{R}^{D_{\text{out}} \times D_r}$, which generates the output vector, $\mathbf{v}(t) \in \mathbb{R}^{D_{\text{out}}}$, by the operation

$$\mathbf{v}(t + \Delta t) = \mathbf{W}_{\text{out}}\mathbf{r}(t + \Delta t), \tag{2}$$

with $\mathbf{W}_{\text{out}}$ the output matrix to be estimated through the training process. Except $\mathbf{W}_{\text{out}}$, all other parameters of the RC, e.g., $\mathbf{W}_{\text{in}}$ and $\mathbf{A}$, are fixed at the construction. Briefly, the purpose of the training process is to find a suitable output matrix $\mathbf{W}_{\text{out}}$ so that the output vector $\mathbf{v}(t + \Delta t)$ as calculated by Eq. (2) is as close as possible to the input vector $\mathbf{u}(t + \Delta t)$ for $t = (\tau + 1)\Delta t, \ldots, (\tau + L)\Delta t$, with $T_0 = \tau\Delta t$ the transient period to avoid the impact of the initial states of the reservoir and L the length of the training time series. This can be done by minimizing the cost function with respect to $\mathbf{W}_{\text{out}}$ [8,9,17]:

$$\mathbf{W}_{\text{out}} = \mathbf{U}\mathbf{V}^T(\mathbf{V}\mathbf{V}^T + \lambda\mathbb{I})^{-1}. \tag{3}$$

Here $\mathbf{V} \in \mathbb{R}^{D_r \times L}$ is the state matrix whose kth column is $\mathbf{r}[(\tau + k)\Delta t]$, $\mathbf{U} \in \mathbb{R}^{D_{\text{in}} \times L}$ is a matrix whose kth column is $\mathbf{u}[(\tau + k)\Delta t]$, $\mathbb{I}$ is the identity matrix, and λ is the ridge regression parameter for avoiding the overfitting. After training, the output matrix $\mathbf{W}_{\text{out}}$ will be fixed, and the RC is ready for prediction. In the predicting phase, first we set the control parameter β to a specific value of interest (not necessarily the parameters used in the training phase), and then we evolve the RC as an autonomous dynamical system by taking the output vector $\mathbf{v}(t)$ as the next input vector $\mathbf{u}_\beta(t)$.

Finally, with a fine tuning of the control parameter β, we attempt to reconstruct the KAM diagram of the system.

We note that the input data in the training phase consist of two time series: (1) the input vector $\mathbf{u}_\beta(t)$ that represents the state of the target system and (2) the control parameter $\beta(t)$ that labels the condition under which the input vector $\mathbf{u}_\beta(t)$ is acquired. More specifically, the input vector $\mathbf{u}_\beta(t)$ is composed of m segments of length T, while each segment is a time series obtained from the target system under a specific control parameter β. As such, $\beta(t)$ is a step function of time. In the predicting phase, besides replacing $\mathbf{u}_\beta(t)$ with $\mathbf{v}(t)$, we still need to input the control parameter $\beta(t)$, so as to guide the reservoir evolution. For convenience, we set in the present work $D_{\text{in}} = D_{\text{out}}$ for the input and output vectors.

While there are many choices for the control parameter in Hamiltonian systems, we choose the initial conditions. To be specific, we generate different motions by varying one of the initial conditions, while keeping the other initial conditions of the system unchanged. We are going to show that the parameter-aware RC is able to not only replicate the dynamics climates associated with the training parameters (initial conditions), but also reproduce the dynamics climates associated with other parameters (initial conditions), thereby reconstructing the entire KAM diagram of the Hamiltonian system.

III. RESULTS

A. The double-pendulum oscillator: Results for standard RC

We start by showing the capability of the standard RC in predicting and replicating the dynamics of Hamiltonian systems. As discussed above, a common sense in the existing studies of machine learning is that RC is applicable to only dissipative dynamical systems, and, to predict the state evolution and replicate the dynamics of Hamiltonian systems, physics constraints from the Hamiltonian mechanisms should be incorporated into the learning algorithm, e.g., the development of HNNs [24–28]. Therefore, before reconstructing the KAM diagram with the method of parameter-aware RC, we need to check first whether the dynamics of Hamiltonian systems can be learned with the standard RC, which is realized by setting $\mathbf{b} = 0$ in Eq. (1) and $m = 1$ in preparing the training data.

The first model of Hamiltonian system employed in our study is the double-pendulum oscillator, which is a classical textbook model for demonstrating the nonlinear dynamics of Hamiltonian systems [34,35]. The Hamiltonian of double-pendulum oscillator is $\mathcal{H} = E_k + E_p = [(m_1/6 + m_2/2)l_1^2\omega_1^2 + m_2 l_2^2\omega_2^2/6 + m_2 l_2 l_1 \omega_1\omega_2 \cos(\theta_1 - \theta_2)/2] - g[(m_1/2 + m_2)l_1\cos\theta_1 + l_2 m_2 \cos\theta_2/2]$, with $\omega_{1,2} = d\theta_{1,2}/dt$ the angular frequencies and $g = 9.8\ m/s^2$ the acceleration of gravity on Earth. The variables $\theta_{1,2}$, $m_{1,2}$, and $l_{1,2}$ denote the angular displacements, masses, and lengths of the two pendulums, respectively. By changing the initial values of the two pendulums, $\mathbf{u}(0) = [\theta_1(0), \omega_1(0), \theta_2(0), \omega_2(0)]^T$, the system can present rich dynamical behaviors, including quasiperiodic and chaotic motions. According to Lagrange's equation of the second kind, the dynamics of the double-pendulum oscillator is governed by equations

$$\left(\frac{m_1}{3} + m_2\right) l_1^2\dot{\omega}_1 + \frac{m_2 l_1 l_2}{2}\cos(\theta_1 - \theta_2)\dot{\omega}_2 + \frac{m_2 l_1 l_2}{2}\sin(\theta_1 - \theta_2)\omega_2^2 + \frac{(m_1 + 2m_2)gl_1}{2}\sin\theta_1 = 0,$$

$$\frac{m_2 l_1 l_2}{2}\cos(\theta_1 - \theta_2)\dot{\omega}_1 + \frac{m_2 l_2^2}{3}\dot{\omega}_2 - \frac{m_2 l_1 l_2}{2}\sin(\theta_1 - \theta_2)\omega_1^2 + \frac{m_2 g l_2}{2}\sin\theta_2 = 0. \quad (4)$$

Without the loss of generality, we set the two pendulums to be identical in mass and length, i.e., $m_1 = m_2 = m$ and $l_1 = l_2 = l$. By introducing the new time variable $t = \sqrt{g/l_1}t'$ [t' is the time value for Eq. (4)], the equations can be rewritten as

$$\dot{\omega}_1 = \left[9\cos(\theta_1 - \theta_2)\sin(\theta_1 - \theta_2)\omega_1^2 + 6\sin(\theta_1 - \theta_2)\omega_2^2 + 18\sin\theta_1 - 9\cos(\theta_1 - \theta_2)\sin\theta_2\right]/[9\cos^2(\theta_1 - \theta_2) - 16],$$

$$\dot{\omega}_2 = \left[24\sin(\theta_1 - \theta_2)\omega_1^2 + 9\cos(\theta_1 - \theta_2)\sin(\theta_1 - \theta_2)\omega_2^2 + 27\cos(\theta_1 - \theta_2)\sin\theta_1 - 24\sin\theta_2\right]/[16 - 9\cos^2(\theta_1 - \theta_2)]. \quad (5)$$

The total energy of the system now reads $E = 2\omega_1^2/3 + \omega_2^2/6 + [\omega_1\omega_2\cos(\theta_1 - \theta_2) - \cos(\theta_2) - 3\cos(\theta_1)]/2$. In simulations, Eq. (5) is evolved numerically by the symplectic algorithm, with the time step being set as $\Delta t = 0.2$. In model simulations, we fix the initial conditions $\theta_1(0) = 0.6$, $\omega_1(0) = 0$, $\omega_2(0) = 0$, while changing the initial condition $\theta_2(0)$ within the range $[-\pi, \pi)$ to generate different motions.

We demonstrate first the learning of an integrable Hamiltonian system. Setting $\theta_2(0) = 1.35$, the oscillator presents the quasiperiodic motion, as depicted in Fig. 1(a) (the black curves). By solving Eq. (5) numerically, we collect the system state $\mathbf{u}(t) = [\theta_1(t), \omega_1(t), \theta_2(t), \omega_2(t)]^T$ for a sequence of $\hat{T} = 3.1 \times 10^3$ time steps (about 100 oscillation cycles). The sequence is divided into three segments. The first segment of length $T_0 = 100$ is used to drive the reservoir out of the transient period, the second segment of length $T = 2 \times 10^3$ is used as the training data to calculate the output matrix $\mathbf{W}_{\text{out}}$, and the third segment of length $T' = 1 \times 10^3$ is used as the test data. In this case, the parameters of the reservoir are chosen as $(D_r, d, \rho, \alpha, \sigma, \lambda) = (500, 0.48, 1.48, 0.25, 1.52, 1 \times 10^{-9})$, which are obtained by the optimizer "optimoptions" in MATLAB. In the predicting phase, the final state of the reservoir network in the training phase is used as the initial state, and the reservoir is evolving according to Eqs. (1) and (2) by replacing $\mathbf{u}(t)$ with $\mathbf{v}(t)$. (This setting of the initial states of the reservoir nodes is for the purpose of predicting the state evolution, while for the purpose of climate replication the initial states can be randomly chosen.) The time evolution of the system state predicted with the trained RC is plotted in Fig. 1(a1) (the red curves). We see that the predictions are in good agreement with the results obtained from direct simulations of the model system for a

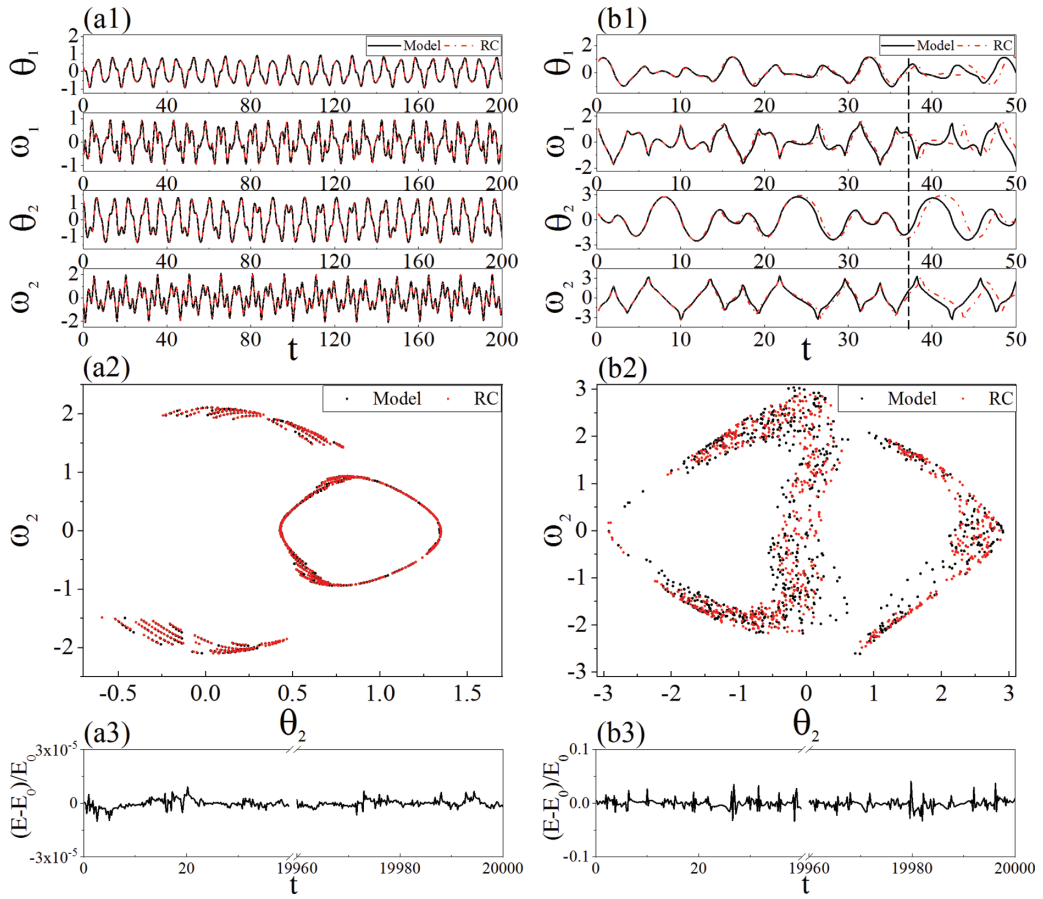

FIG. 1. Predicting the dynamics of double-pendulum oscillator by the standard RC. (a) Predicting the quasiperiodic motion generated by the initial condition $\theta_2(0) = 1.35$. (a1) The time evolution of the system state. (a2) The system trajectory detected by the Poincaré surface of section defined by $\omega_1 = 0$ and $\theta_1 > 0$. The similarity parameter between the true and predicated trajectories is about 0.97. (a3) The long-time evolution of the system energy. The true energy of the system is $E_0 = -1.3475$. (b) Predicting the chaotic motion generated by the initial condition $\theta_2(0) = 2.04$. (b1) The time evolution of the system state. (b2) The chaotic trajectories detected by the Poincaré surface of section. The similarity parameter between the true and predicated trajectories is about 0.52. (b3) The long-time evolution of the system energy. The true energy of the system is $E_0 = -1.01$. Results obtained by the direct simulations of the model system are colored in black. Results predicted by RC are colored in red.

long period. To check whether the statistical properties of the quasiperiodic motion is properly replicated by RC, we plot in Fig. 1(a2) the system trajectory for a period of $T = 1 \times 10^4$ time steps on the Poincaré surface of section defined by $\omega_1 = 0$ and $\theta_1 > 0$. We see that the trajectory predicted by the RC is well overlapped with the one obtained from the model system, manifesting the proper replication of the dynamics climate.

We generalize the similarity parameter introduced in Ref. [36] and use the generalized parameter to evaluate qualitatively the "degree of overlap" between the true and replicated trajectories. In doing this, we first divide the phase plane (θ_2, ω_2) into a grid of $N \times N$ cells, and then calculate for each trajectory the number of points within each cell. Denoting n_{ij}^t and n_{ij}^r as the number of points inside cell (i, j) for the true and replicated trajectories, respectively, the similarity parameter is calculated as

$$M = 1 - \frac{1}{C} \sum_{i,j}^{N} \left(\frac{n_{ij}^t - n_{ij}^r}{n_{ij}^t + n_{ij}^r} \right)^2, \tag{6}$$

with C the number of nonempty cells occupied by the two trajectories in the phase plane. Depending on the overlapping degree of the two trajectories, the value of M is varying within the range [0,1]. The similarity parameter has been proposed in the literature for quantifying the degree of measure synchronization between coupled Hamiltonian oscillators [36–38]. Here we employ it to quantify the overlapping degree of the true and replicated trajectories in machine learning. In general, the larger the value of M, the higher the overlap between the trajectories. To make the results comparable between different dynamical motions (the chaotic motions occupy a larger area in the phase plane than the quasiperiodic motions), in the present work we fix the boundaries of the phase plane as $\theta_2 \in (-3.5, 3.5)$ and $\omega_2 \in (-3.5, 3.5)$ and set the size of the grid as $N = 20$. For the trajectories shown in Fig. 1(a2), the similarity parameter is $M \approx 0.97$, indicating that the true and replicated trajectories are well overlapped. The results in Figs. 1(a1) and 1(a2) thus validate the capability of the standard RC in predicting and replicating the dynamics of an integrable Hamiltonian system.

We next demonstrate the learning of a nonintegrable Hamiltonian system. Setting $\theta_2(0) = 2.04$, the oscillator presents the chaotic motion, with the largest Lyapunov exponent being $\Lambda \approx 0.163$. In a similar way, we generate the training data by simulating Eq. (5), calculate the output matrix $\mathbf{W}_{\text{out}}$, and then use the trained RC to predict the state evolution and replicate the system dynamics. For this case, the set of parameters for the RC are chosen as $(D_r, d, \rho, \alpha, \sigma, \lambda) = (500, 0.36, 2.66, 0.24, 2.08, 5.4 \times 10^{-2})$. Figure 1(b1) shows the time evolution of the system state predicted by RC (the red curves), together with the results obtained from model simulations (the black curves). We see that the RC can predict accurately the system evolution for a period of $T \approx 35$ (about six Lyapunov times). Figure 1(b2) shows the system trajectory on the Poincaré surface of section defined by $\omega_1 = 0$ and $\theta_1 > 0$. We see that the predicted and true trajectories are overlapped in the space reasonably well ($M \approx 0.52$). (By increasing the length of the replicated trajectory, the value of M will be increased slightly, but the increment is small.) To confirm further the replication of the system climate, we calculate the largest Lyapunov exponent of the predicted trajectory by the numerical method proposed in Ref. [39]. The calculated result is $\Lambda \approx 0.166$, which agrees with the one obtained from direct simulations well.

In machine learning of Hamiltonian systems, a major concern is whether the system energy will be conserved in the long-term evolution [24,30]. In particular, it has been shown that if the physics constraints of the Hamiltonian and Lagrangian mechanisms are not imposed in the feedforward neural network, the system energy calculated from the predicted results of the machine will be gradually decreased as time increases. This concern has led to the development of HNNs and Lagrangian neural networks (LNNs) [24,30], with which the system energy is well conserved in the long-term evolution. The problem of energy conservation, however, remains as an open issue for RC, as the architecture of RC is completely different from that of the feed-forward neural networks. The dissipative nature of the reservoir suggests that the system energy predicted by the machine could be decreasing with time, whereas the results of evolution prediction and climate replication plotted in Fig. 1 suggest that the system energy is conserved. To check this, we plot in Figs. 1(a3) and 1(b3) the long-time evolution of the system energy for the quasiperiodic and chaotic motions, respectively. We see that the system energy is well conserved for both cases. For the quasiperiodic motion [see Fig. 1(a3)], the system energy is fluctuating around the true value E_0 by small amplitudes of the order of 10^{-5}; for the chaotic motion [see Fig. 1(b3)], the fluctuating amplitudes are of the order of 10^{-2}. We note that, by decreasing the time step Δt, the fluctuating amplitudes can be further decreased (not shown).

To check the generality of the standard RC in replicating the climate of Hamiltonian dynamics, we keep the other initial values of the double-pendulum oscillator unchanged, while choosing the initial value of $\theta_2(0)$ randomly within the range $[-\pi, \pi)$. As above, for each value of $\theta_2(0)$, we first train the RC by the time series obtained from model simulations, and then replicate the dynamics climate based on the RC outputs. $n = 34$ initial values are chosen in total. The trajectories of the replicated dynamics on the Poincaré surface of section ($\omega_1 = 0$ and $\theta_1 > 0$) are plotted in Fig. 2(a). We see that the reconstructed KAM diagram is mixed with chaotic and regular dynamics. By the same set of initial values of $\theta_2(0)$, we plot in Fig. 2(b) the KAM diagram based on the results of model simulations. The similarity parameter between Figs. 2(a) and 2(b) is $M \approx 0.83$, suggesting that the true KAM diagram is well replicated by RC. Figure 2(c) shows the relationship between the energies of the true and replicated dynamics for the $n = 34$ trajectories, we see that the two energies agree with each other very well for all the trajectories.

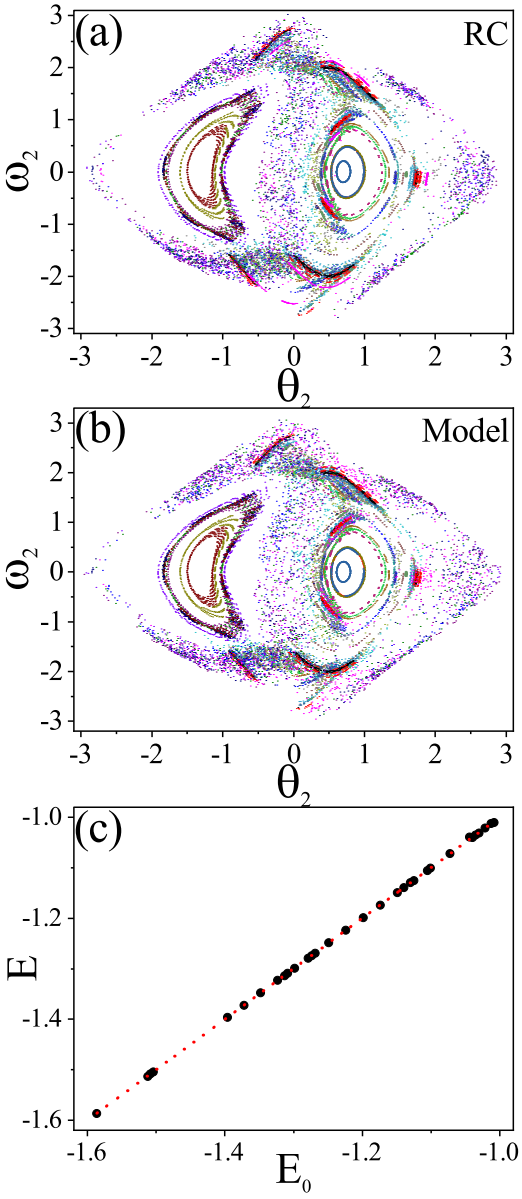

FIG. 2. The dynamics of $n = 34$ different double-pendulum oscillators on the Poincaré section surface. (a) The results predicted by the standard RC. (b) The results from direction simulations of the model system. In (a) and (b), trajectories with the same initial value $\theta_2(0)$ are represented by the same color. The similarity parameter between (a) and (b) is $M \approx 0.83$. (c) The relationship between the true energy E_0 and the energy of the replicated dynamics E for $n = 34$ trajectories. Each result of E is averaged over a time period of $T = 1 \times 10^4$. Dotted line: the diagonal line denoting $E = E_0$.

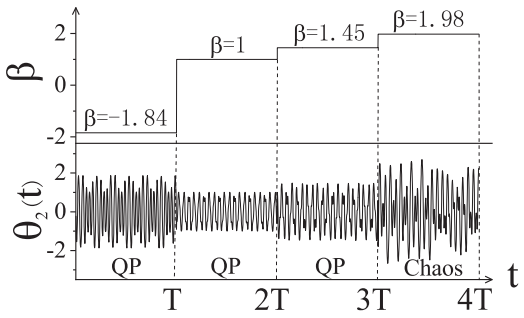

FIG. 3. For the double-pendulum oscillator, a schematic plot of the training data used in training the parameter-aware RC. The training data consist of $m = 4$ segments of equal length $T = 2.5 \times 10^3$. The segments are generated by the control parameters $\beta = -1.84$, 1, 1.45, and 1.98. For $\beta = -1.84$, 1, and 1.45, the system dynamics is quasiperiodic (QP); for $\beta = 1.98$, the system dynamics is chaotic.

B. The double-pendulum oscillator: Results for parameter-aware RC

We move on to study the capability of parameter-aware RC in replicating the KAM dynamics diagram, which now employs the parameter-control module (i.e., the vector $\beta\mathbf{b}$) in Eq. (1) [19,20,33]. The adoption of the parameter-aware RC is motivated by the fact that in many realistic situations only the time series of a limited number of system motions are available, while the mission is to predict the dynamics, or replicate the system climate, of many unknown motions. As the knowledge the machine learned from the training system (e.g., the output matrix) is used to predict the dynamics of other systems with different motions, parameter-aware RC thus can be regarded as another approach of transfer learning [40,41]. For the case of Hamiltonian dynamics, our mission here is replicating the KAM dynamics diagram (as shown in Fig. 2) based on the time series of a handful of system motions.

Different from the standard RC, in parameter-aware RC the training data are composed of m segments, with each segment being a time series of length T generated under a specific control parameter β by the target system. To generate the training data in simulations, we fix again the initial values $[\theta_1(0), \omega_1(0), \omega_2(0)] = (0.6, 0, 0)$, while changing $\theta_2(0)$ to adjust the system dynamics. In this application, we set $\theta_2(0)$ as the control parameter, i.e., setting $\beta = \theta_2(0)$ in Eq. (1). The m segments are then combined to form the new time series $\mathbf{u}_\beta(t) = [\theta_1(t), \omega_1(t), \theta_2(t), \omega_2(t)]^T_\beta$, which now has the length $L = mT$. The new time series and the time series of the control parameter $\beta(t)$ are fed into the reservoir for estimating the output matrix $\mathbf{W}_{\text{out}}$. As was done for the standard RC, the initial states of the reservoir network are still randomly chosen, and, before the training, the reservoir is evolved for a period of $T_0 = 100$ steps to discard the transient. As an illustration, we choose $m = 4$ control parameters and acquire for each parameter a time series of $T = 2.5 \times 10^3$ states. The four control parameters are $\beta = -1.84$, 1.0, 1.45, and 1.98, which are randomly chosen within the range $[-\pi, \pi)$. The system dynamics is quasiperiodic for $\beta = -1.84$, 1.0, and 1.45, and is chaotic for $\beta = 1.98$ (the largest Lyapunov exponent is $\Lambda \approx 0.16$). The structure of the training data is schematically shown in Fig. 3. In this application, the set of parameters for the parameter-aware RC are chosen as $(D_r, d, \rho, \alpha, \sigma, \lambda) = (1 \times 10^3, 0.97, 1.13, 0.64, 0.94, 2 \times 10^{-2})$.

We check first the feasibility of the trained RC in replicating the dynamics climates associated with the training parameters. This is implemented by changing the control parameter β to one of the training parameters, and then evolving the reservoir according to Eqs. (1) and (2) as described in Sec. II. We note that the major difference between the standard RC and the parameter-aware RC lies in the variability of the output matrix in the predicting phase. For the standard RC employed in Sec. III A, the output matrix is trained separately for each time series, and each output matrix is able only to replicate the dynamics climate associated with a specific control parameter. For the parameter-aware RC, the output matrix is trained only once, and in the predicting phase the same output matrix is used to reproduce the dynamics climate of any desired parameter. Another difference between the standard RC and parameter-aware RC is the setting of the initial states of the reservoir network at the beginning of the predicting phase. For the standard RC, the initial states are set as the final states of training phase. For the parameter-aware RC, the initial states are generated by one-step iteration of the reservoir network driven by the initial values of the model system, $\mathbf{u}(0) = [\theta_1(0), \omega_1(0), \theta_2(0), \omega_2(0)]^T$. (We note that this setting is only necessary when the purpose is to predict the system evolution. If the purpose is to replicate the dynamics climate and reconstruct the KAM diagram, the initial states of the reservoir can be randomly chosen.) Setting $\beta = -1.84$, we plot in Fig. 4(a1) the state evolution of the system predicted by the machine, together with the results obtained from model simulations. We see that the state evolution is well predicted by the machine for a long period. Figure 4(a2) shows the dynamics on the Poincaré surface of section. The similarity parameter of the true and replicated trajectories is $M \approx 0.68$, signifying that the dynamics climate of the true system is well replicated by the RC. The results for $\beta = 1.98$ are shown in Fig. 4(b). We see that the RC is able to predict the evolution for about six Lyapunov times [see Fig. 4(b1)], and the dynamics climate replicated by the RC is overlapped with the true one reasonably well [see Fig. 4(b2)]. The similar results are also observed for the other two training parameters, $\beta = 1$ and 1.45 (not shown).

We check next the capability of the trained RC in replicating the dynamics climate of a new parameter not included in the training set. As the demonstration, we choose $\beta = 2.0$, with which the model system shows chaotic motion and the largest Lyapunov exponent is $\Lambda \approx 0.1$. The results for this new parameter are plotted in Fig. 4(c). We see that the machine not only predicts accurately the short-time evolution of the system [see Fig. 4(c1)], but also replicates properly the system climate [see Fig. 4(c2)].

Having justified the capability of parameter-aware RC in replicating the climates of both the training and nontraining parameters, we finally exploit it to reconstruct the entire KAM dynamics diagram. In doing this, we keep the output matrix $\mathbf{W}_{\text{out}}$ unchanged, while changing the control parameter β to a number of values that are randomly chosen within the range $[-\pi, \pi)$. In this case, the initial states of the reservoir network

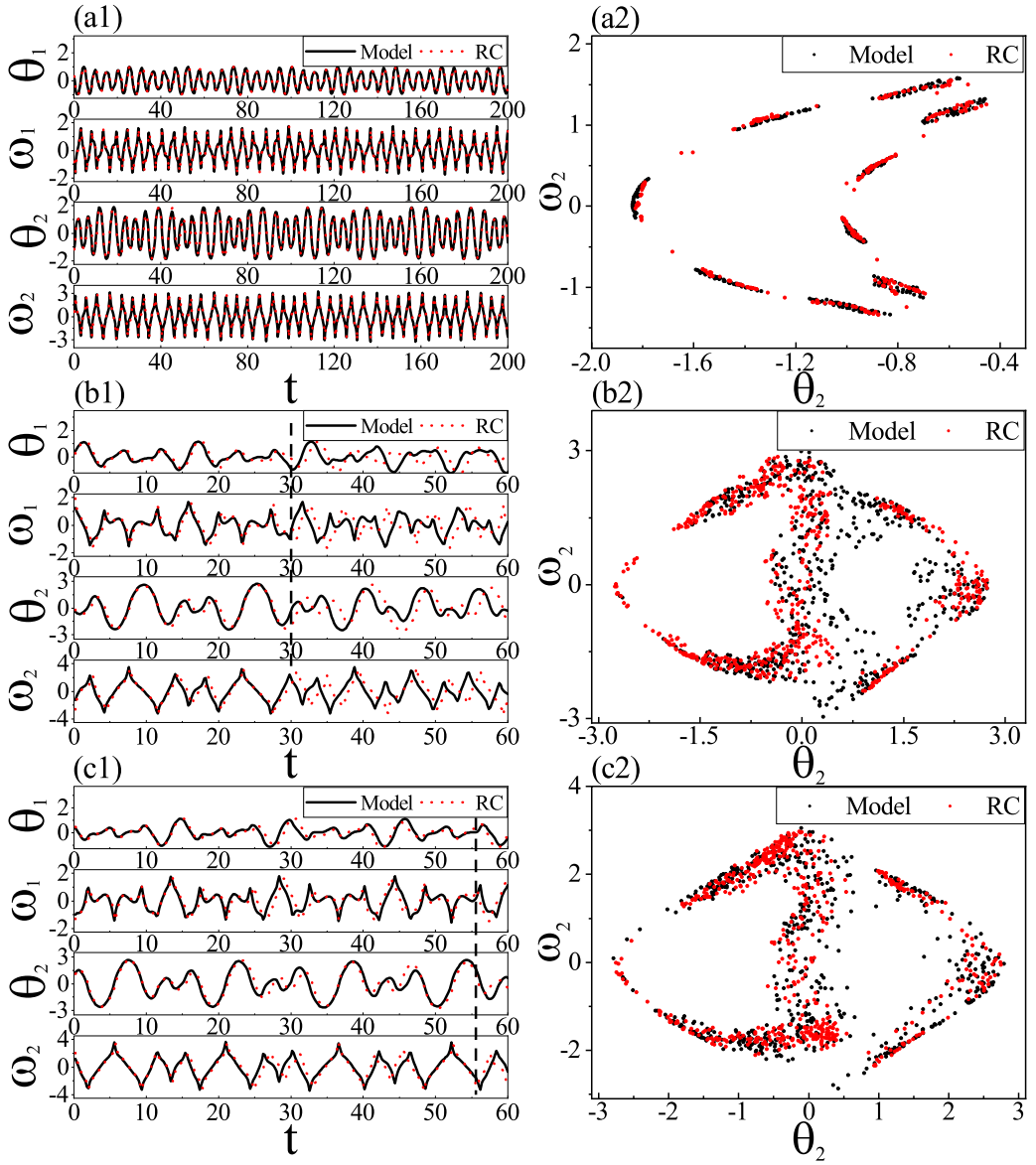

FIG. 4. Predicting the state evolution and replicating the climate of a double-pendulum oscillator with parameter-aware RC. (a) The results for the control parameter $\beta = -1.84$, which is one of the training parameters. (b) The results for the control parameter $\beta = 1.98$, which is also one of the training parameters. (c) The results for the control parameter $\beta = 2.0$, which is not included in the set of training parameters. The vertical lines in (b1) and (c1) denote the prediction horizons. In all graphs, the results predicted by RC are shown as red, and the results obtained from model simulations are shown as black. The similarity parameter between the true and replicated trajectories is $M \approx 0.68$ in (a2), $M \approx 0.50$ in (b2), and $M \approx 0.52$ in (c2).

are randomly chosen within the range $[-1, 1]$. For each value of β, we collect the output of the machine for $T = 1 \times 10^4$ steps, based on which we reconstruct the dynamics climate for this specific control parameter. To demonstrate, we choose $n = 31$ control parameters. The $n = 31$ trajectories on the Poincaré surface of section are plotted in Fig. 5(a). We see that the diagram is mixed with quasiperiodic and chaotic motions. With the same set of control parameters, we plot in Fig. 5(b) the KAM dynamics diagram based on the results of model simulations. The similarity parameter between Figs. 5(a) and 5(b) is $M \approx 0.69$, suggesting that the KAM diagram is well reconstructed by the machine. To confirm the performance of parameter-aware RC further, we plot in Fig. 5(c) the relationship between E_0 (the true energy) and E (the replicated energy) for all $n = 31$ trajectories. It is seen that the data are well fitted by the diagonal line, indicating a good replication of the energy by the parameter-aware RC.

C. The standard map

We generalize the above findings by employing the parameter-aware RC to reconstruct the KAM diagram of a Hamiltonian mapping system. The model we adopt is the standard map (also known as the kicked rotor). The Hamiltonian of standard map reads $\mathcal{H} = p_\theta^2/(2I) + K\cos\theta\sum_n \delta(t - n\tau)$, with I the rotational inertia of the bar, K the kicking strength,

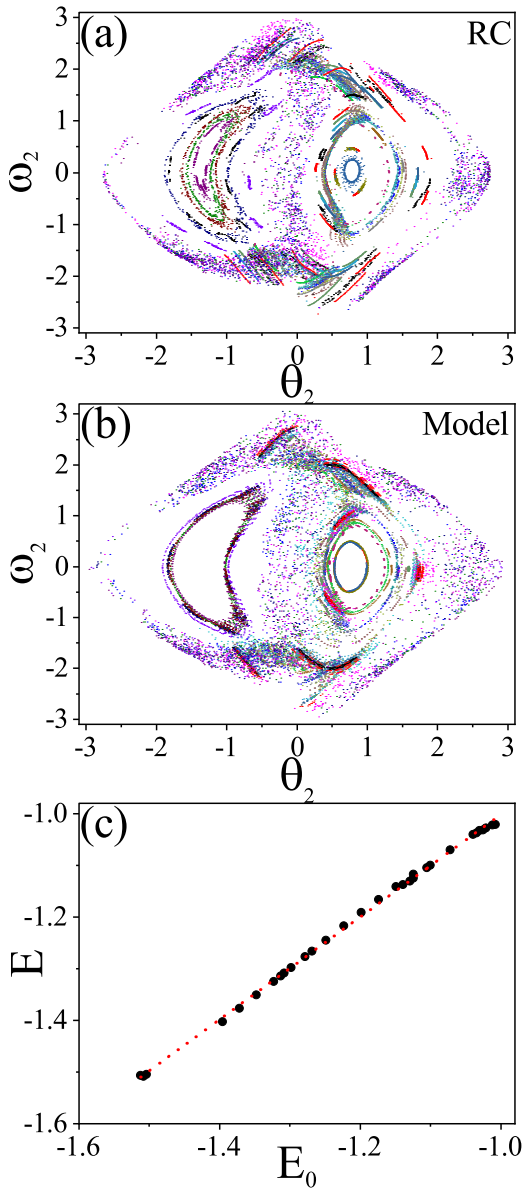

FIG. 5. The KAM diagram of the double-pendulum oscillator for $n = 31$ dynamics. (a) The results predicted by the parameter-aware RC. (b) The results obtained from model simulations. The same set of control parameters (initial conditions) are used in (a) and (b). The similarity parameter between (a) and (b) is $M \approx 0.69$. (c) The relationship between the true energy E_0 and the energy of the replicated dynamics E for $n = 31$ trajectories. Each result of E is averaged over a time period of $T = 1 \times 10^4$. Dotted line: the diagonal line denoting $E = E_0$.

$n\tau$ the moment of the nth kick, and $\delta(\ldots)$ the Dirac delta function. θ and p_θ represent the angular displacement and angular momentum of the bar, respectively. The Hamilton equations of motion read

$$dp_\theta/dt = K \sin\theta \sum_n \delta(t - n\tau), \tag{7}$$

$$d\theta/dt = p_\theta/I. \tag{8}$$

The integration of the above equations from $t = n\tau$ to $t = (n+1)\tau$ leads to the new equations

$$\theta_{n+1} = (\theta_n + p_n) \quad \text{modulo} \quad 2\pi, \tag{9}$$

$$p_{n+1} = (p_n + K \sin\theta_{n+1}) \quad \text{modulo} \quad 2\pi. \tag{10}$$

For the sake of simplicity, here we set $\tau = I$, and restrict θ and p to be within the range $[0, 2\pi)$. The KAM dynamics diagram of the standard map is dependent on the kicking strength K [23]. When K is small, the diagram is dominated by quasiperiodic dynamics. As K increases, more tori will be broken and the diagram is mixed with quasiperiodic and chaotic dynamics. When K is large, most of the tori will be broken and the diagram is dominated by chaotic motions. Our mission here is to reconstruct the KAM diagram for a fixed value of K based on the information of a handful of motions.

We demonstrate first the reconstruction of the KAM diagram for a small kicking strength. For demonstration purposes, we choose $K = 0.5$, by which the KAM diagram is dominated by quasiperiodic motions. In generating the training data, we fix the initial value of the angular displacement as $\theta_0 = \pi$, while changing the initial value of the angular momentum p_0 to $m = 8$ different values. The eight initial values of p_0 are randomly chosen within the range $[0, 2\pi)$, which are 1.76, 2.38, 3.2, 3.35, 3.73, 4.74, 5.28, and 5.77 in this case. For each value of p_0, we simulate the system dynamics according to Eqs. (9) and (10) and record the system state, $\mathbf{u}(n) = [\sin(\theta_n), \sin(p_n), \cos(\theta_n), \cos(p_n)]^T$ (which is generalized from the state $[\theta_n, p_n]^T$), for a time series of $T = 2 \times 10^3$ iterations. The training data therefore are of length $L = mT = 1.6 \times 10^4$, which, together with the corresponding time series of the control parameter ($\beta = p_0/2\pi$), are fed into the parameter-aware RC for estimating the output matrix. In the predicting phase, we keep the output matrix fixed, while tuning the control parameter to different values, and, based on the predictions, reconstruct the KAM diagram. In this application, the parameters of the reservoir are $(D_r, d, \rho, \alpha, \sigma, \lambda) = (1.5 \times 10^3, 3.6 \times 10^{-3}, 1.62, 0.95, 1.59, 8.2 \times 10^{-2})$. Figure 6(a) shows the dynamics climates predicted by the machine for the eight training parameters. The corresponding dynamics obtained from direct simulations of the model system are plotted in Fig. 6(b). The similarity parameter between the replicated and true diagrams is $M \approx 0.88$. [In calculating the similarity parameter for the standard map, the boundaries of the phase plane are set as $\theta \in (0, 2\pi)$ and $p \in (0, 2\pi)$, and the phase plane is also divided into a grid of 20×20 cells.] To reconstruct the entire KAM diagram, we change the control parameter β to $n = 26$ new values that are randomly chosen within the range (0,1). The dynamics climates of the $n = 26$ new parameters, together with the ones of the eight training parameters, are plotted in Fig. 6(c). The corresponding results obtained from model simulations are plotted in Fig. 6(d). The similarity parameter between the true and reconstructed trajectories is $M \approx 0.69$, signifying a good replication of the KAM diagram by machine.

We next demonstrate the reconstruction of KAM dynamics diagram for a relatively strong kicking strength, $K = 1$. For this kicking strength, many tori are destroyed and the diagram

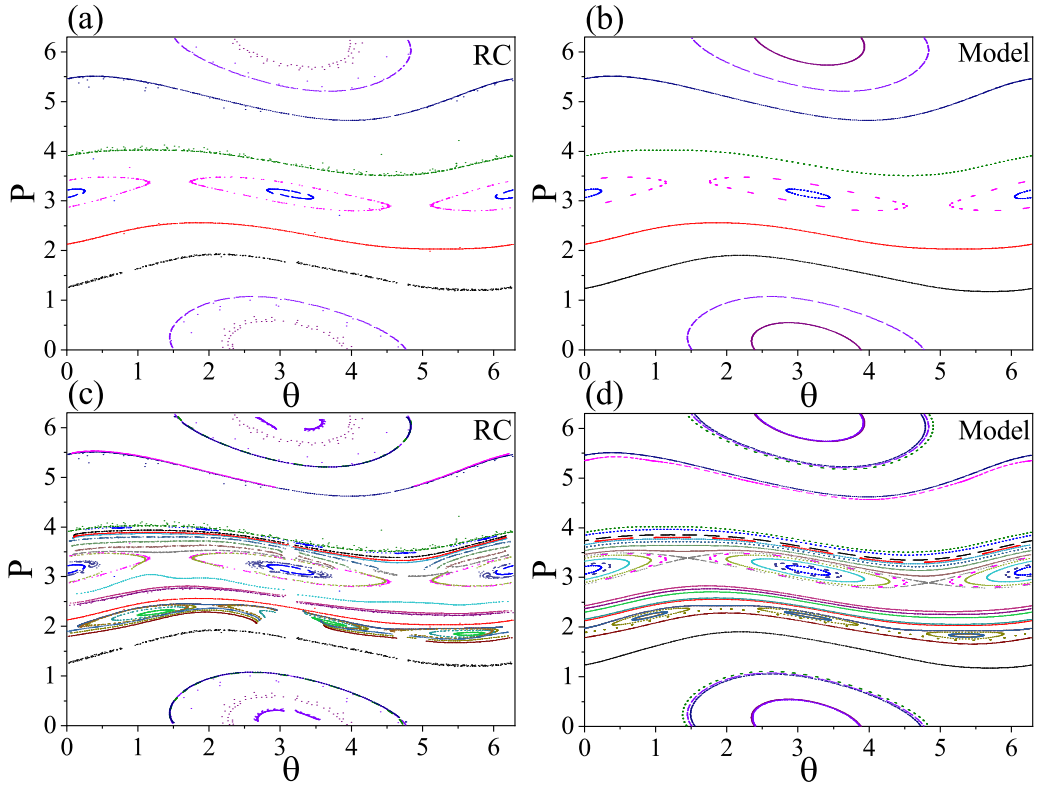

FIG. 6. Reconstructing the KAM diagram of the standard map under the kicking strength $K = 0.5$. The training data are generated by $m = 8$ training parameters. (a) The dynamics predicted by the machine for the training parameters. (b) The dynamics of the training parameters obtained from model simulations. The similarity parameter between (a) and (b) is $M \approx 0.88$. (c) The diagram reconstructed by the machine, which includes $m = 8$ dynamics from the training parameters and $n = 26$ dynamics from the additional control parameters. (d) For the same set of control parameters used in (c), the diagram obtained from model simulations. The similarity parameter between (c) and (d) is $M \approx 0.69$.

is mixed with quasiperiodic and chaotic motions. In this case, the training data are generated by $m = 6$ training parameters randomly chosen from the range $[0, 2\pi)$, $p_0 = (0.58, 2.07, 2.19, 3.35, 3.49, 4.1)$ (three of them generate chaotic motions). The set of parameters for the reservoir are $(D_r, d, \rho, \alpha, \sigma, \lambda) = (1 \times 10^3, 0.66, 0.77, 0.55, 3, 1 \times 10^{-9})$. The dynamics of the six training parameters predicted by the machine are plotted in Fig. 7(a). The corresponding results obtained from model simulations are plotted in Fig. 7(b). We see that the climates of the training parameters are well replicated by the machine ($M \approx 0.75$). By the trained RC, we replicate the climates for $n = 24$ additional control parameters randomly chosen from the range $[0, 2\pi)$. The replicated climates, together with the climates of the six training parameters, are plotted in Fig. 7(c). The corresponding results obtained from model simulations are plotted in Fig. 7(d). We see that the KAM diagram predicted by the machine captures the main features of the diagram obtained from model simulations ($M \approx 0.66$).

IV. DISCUSSION AND CONCLUSION

A few remarks on the performance of RC in learning Hamiltonian systems are in order. First, the purpose of the present work is to reconstruct the KAM dynamics diagram based on the time series acquired at a handful of training parameters, instead of a precise prediction of the state evolution of Hamiltonian systems. As such, the performance of RC is mainly evaluated by the replicability of the statistical properties of the system dynamics, namely, the climate, instead of the prediction horizon. Second, different from conventional RC in which the training and target systems are identical, in parameter-recognizant RC the target systems can be different from the training ones. From the standpoint of transfer learning [40], the fact that the KAM diagram can be reconstructed by the time series of a few training parameters implies the transferability of knowledge between different Hamiltonian motions. Finally, we would like to note that the performance of RC is dependent on the training data, including the number of the training parameters and the motions associated with these parameters. In general, the larger the number of the training parameters, the more accurate the reconstructed diagram. And, for the fixed number of training parameters, the more representative the motions of the training parameters, the more accurate the reconstructed diagram. For instance, for the example of standard map showing mixed dynamics (see Fig. 7), if all the $m = 6$ training parameters are of quasiperiodic motions, the trained RC will not be able to replicate the chaotic dynamics, and, as the consequence, the reconstructed diagram will be distinctly different from the true one.

In machine learning of chaotic systems, a topic under active debate in literature is what the machine really learns from the data—the mathematical expressions describing the given time series, the dynamics governing the system evolution,

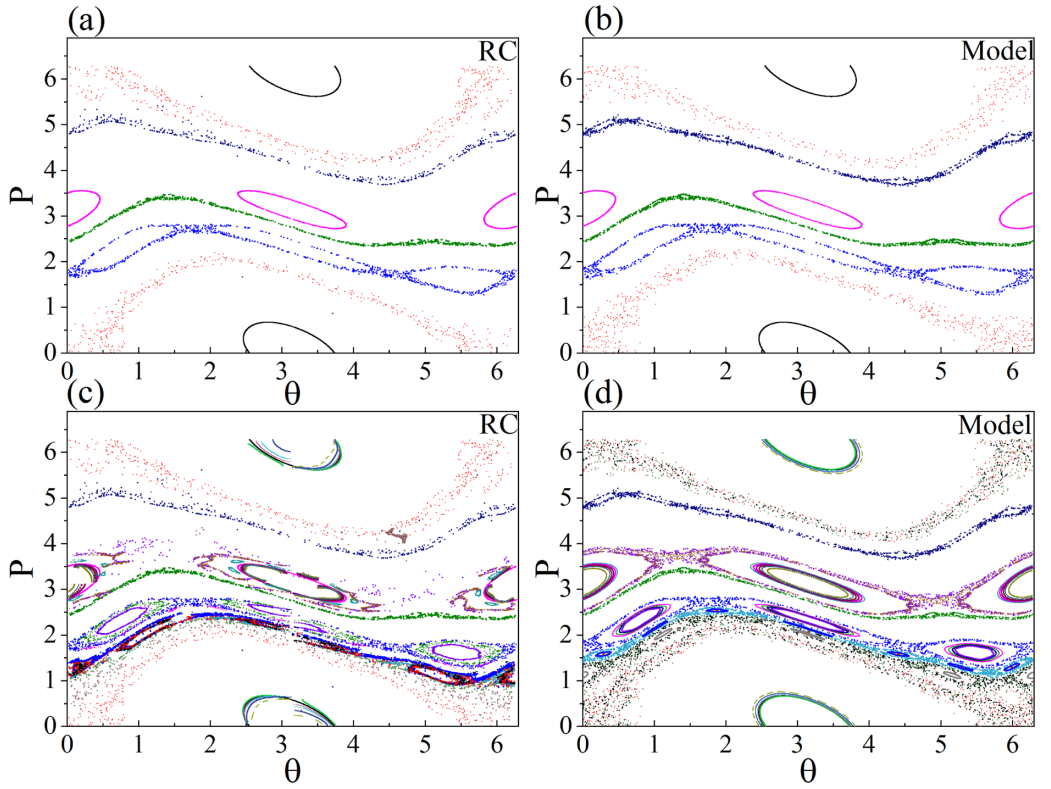

FIG. 7. Reconstructing the KAM diagram of the standard map under the kicking strength $K = 1$. The training data are generated by $m = 6$ training parameters. (a) The dynamics predicted by the machine for the training parameters. (b) The dynamics of the training parameters obtained from model simulations. The similarity parameter between (a) and (b) is $M \approx 0.75$. (c) The diagram reconstructed by the machine, which includes the dynamics from the training parameters and $n = 24$ dynamics from the additional control parameters. (d) For the same set of control parameters used in (c), the diagram obtained from model simulations. The similarity parameter between (a) and (b) is $M \approx 0.66$.

or the physical laws underlying the data set and dynamics. For the purpose of state evolution prediction, the goal can be accomplished by fitting the time series with a mathematical expression, which, given the reservoir is complex enough, can be normally achieved [1]. If this is the case, then the machine will be data-specific, e.g., the machine trained by periodic motions cannot be used to replicate the dynamics of chaotic motions. Yet recent studies on transfer learning of chaotic systems show that in some circumstances knowledge can be transferred between systems of different motions, e.g., using the RC trained by periodic logistic maps to replicate the climates of chaotic logistic maps [19]. The results of transfer learning suggest that it is the intrinsic dynamics that are learned by the machine, but not the mathematical expressions describing the motions. Our current study of Hamiltonian systems indicates that the machine (specifically the parameter-aware RC) might learn something "deeper" than mathematical expressions and dynamics, i.e., the physical laws. Physical laws in nature are characterized by symmetries and invariants, which are the essential rules defining the system dynamics and guiding the system evolutions. In learning physical systems, it is commonly believed that by incorporating the physical laws, the performance of the machines can be significantly improved, e.g., the development of HNNs. Yet there are also studies showing that, just like the human brain, a machine might be able to extract the physical laws and concepts from the data without any prior knowledge or assumptions about physics, kinematics, or geometry [42–45], such as Hamiltonians, Lagrangians, and other laws of geometric and momentum conservation. Our present work provides additional evidence for the automated learning of physical laws from data by showing that the RC, probably the simplest recurrent neural network in machine learning, is able to learn from the data the Hamiltonian mechanisms.

A brief discussion on the difference between HNNs and RC is necessary. HNNs are modified from the feedforward neural network by imposing the Hamiltonian mechanisms, in which the information is flowing from the input layer to the output layer in a one-way fashion. As such, HNNs can be essentially treated as a mapping function connecting the input and output vectors without intrinsic network dynamics. In particular, a HNN has no memory about the previous input data, and, at each time step of the reservoir evolution, the output of HNNs is completely determined by the input data. In contrast, RC is a type of recurrent neural network and is running essentially as a complex dynamical system. For RC, the output not only depends on the input, but also is affected by the dynamical state of the reservoir. Previous studies on machine learning of Hamiltonian systems are mainly based on HNNs, in which the Hamiltonian mechanisms must be known *a priori* and be imposed explicitly on the algorithm [24–33]. Yet in realistic situations the Hamiltonian is usually unknown, making the direct application of HNNs infeasible. Our current study shows that for RC, the Hamiltonian mechanisms could

be encoded in the reservoir in the training phase and be used to reconstruct the KAM diagram in the predicting phase. As such, the requirement of prior knowledge of the Hamiltonian mechanisms is removed in RC. Another advantage that RC enjoys in realistic applications is the reduced data size and computational cost. Due to the multiple-layered network structure, both the amount of data required for training HNNs and the number of parameters to be trained are orders of magnitude larger than that of RC. This raises the training cost and computational burden in applying HNNs. These problems, as demonstrated in our studies, do not exist for RC. The Hamiltonian-free, data-saving, and easy-training properties make RC a powerful technique for predicting Hamiltonian dynamics.

Summarizing up, we have studied the learning of Hamiltonian systems with the RC technique and found that, without prior knowledge of the Hamiltonian mechanisms, the trained RC is able to not only forecast accurately the short-term evolution of the system state (i.e., the relative error between the true and predicted states is less than 5% for several Lyapunov times), but also replicate reasonably the long-term ergodic properties of the system dynamics (i.e., for a period of $T = 1 \times 10^4$, the system energy and the largest Lyapunov exponent calculated from the replicated dynamics are close to those of the true dynamics, and the similarity parameter of the true and replicated trajectories is larger than 0.5). Furthermore, by the architecture of parameter-aware RC, we have demonstrated that based on the time series of a handful of training parameters, the trained RC is able to reconstruct the entire KAM dynamics diagram with a high precision. Although our studies are based on toy models, it is expected that the similar results can be also found in other Hamiltonian systems. The current study provides an alternative approach for learning Hamiltonian systems and gives insights into the intrinsic working mechanism of RC.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China under Grant No. 11875182.

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

[2] M. Lukosevicius and H. Jaeger, Reservoir computing approaches to recurrent neural network training, Comput. Sci. Rev. **3**, 127 (2009).

[3] G. Tanaka, T. Yamane, J. B. Heroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose, Recent advances in physical reservoir computing: A review, Neural Netw. **115**, 100 (2019).

[4] Y. Tang, J. Kurths, W. Lin, E. Ott, and L. Kocarev, Introduction to focus issue: When machine learning meets complex systems: Networks, chaos, and nonlinear dynamics, Chaos **30**, 063151 (2020).

[5] Z. C. Lipton, J. Berkowitz, and C. Elkan, A critical review of recurrent neural networks for sequence learning, arXiv:1506.00019.

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

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

[8] 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).

[9] 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).

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

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

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

[13] R. Follmann and E. Rosa, Predicting slow and fast neuronal dynamics with machine learning, Chaos **29**, 113119 (2019).

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

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

[16] J. Z. Kim, Z. Lu, E. Nozari, G. J. Pappas, and D. S. Bassett, Teaching recurrent neural networks to infer global temporal structure from local examples, Nat. Mach. Intell. **3**, 316 (2021).

[17] 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).

[18] 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).

[19] L.-W. Kong, H.-W. Fan, C. Grebogi, and Y.-C. Lai, Machine learning prediction of critical transition and system collapse, Phys. Rev. Res. **3**, 013090 (2021).

[20] H.-W. Fan, L.-W. Kong, Y.-C. Lai, and X. G. Wang, Anticipating synchronization with machine learning, Phys. Rev. Res. **3**, 023237 (2021).

[21] C. Klos, Y. F. Kalle Kossio, S. Goedeke, A. Gilra, and R.-M. Memmesheimer, Dynamical Learning of Dynamics, Phys. Rev. Lett. **125**, 088103 (2020).

[22] Y. L. Guo, H. Zhang, L. Wang, H. W. Fan, J. H. Xiao, and X. G. Wang, Transfer learning of chaotic systems, Chaos **31**, 011104 (2021).

[23] E. Ott, *Chaos in Dynamical Systems* (Cambridge University Press, Cambridge, 2002).
[24] S. Greydanus, M. Dzamba, and J. Yosinski, Hamiltonian neural networks, Adv. Neural Inf. Proc. Systems **32**, 15379 (2019).
[25] P. Toth, D. J. Rezende, A. Jaegle, S. Racanière, A. Botev, and I. Higgins, Hamiltonian generative networks, arXiv:1909.13789.
[26] T. Bertalan, F. Dietrich, I. Mezić, and I. G. Kevrekidis, On learning Hamiltonian systems from data, Chaos **29**, 121107 (2019).
[27] A. Choudhary, J. F. Lindner, E. G. Holliday, S. T. Miller, S. Sinha, and W. L. Ditto, Physics-enhanced neural networks learn order and chaos, Phys. Rev. E **101**, 062207 (2020).
[28] M. Mattheakis, D. Sondak, A. S. Dogra, and P. Protopapas, Hamiltonian neural networks for solving differential equations, arXiv:2001.11107.
[29] J. W. Burby, Q. Tang, and R. Maulik, Fast neural Poincaré maps for toroidal magnetic fields, Plasma Phys. Control. Fusion **63**, 024001 (2020).
[30] M. Cranmer, S. Greydanus, S. Hoyer, P. Battaglia, D. Spergel, and S. Ho, Lagrangian neural networks, arXiv:2003.04630.
[31] Z. Dulberg and J. Cohen, Learning canonical transformations, arXiv:2011.08822.
[32] A. Choudhary, J. F. Lindner, E. G. Holliday, S. T. Miller, S. Sinha, and W. L. Ditto, Forecasting Hamiltonian dynamics without canonical coordinates, Nonlinear Dyn. **103**, 1553 (2021).
[33] C.-D. Han, B. Glaz, M. Haile, and Y.-C. Lai, Adaptable Hamiltonian neural networks, Phys. Rev. Res. **3**, 023156 (2021).
[34] R. B. Levien and S. M. Tan, Double pendulum: An experiment in chaos, Am. J. Phys. **61**, 1038 (1993).
[35] I. Kovacic, M. Zukovic, and D. Radomirovic, Normal modes of a double pendulum at low energy levels, Nonlinear Dyn. **99**, 1893 (2020).
[36] S. Gupta, S. De, M. S. Janaki, and A. N. S. Lyengar, Exploring the route to measure synchronization in non-linearly coupled Hamiltonian systems, Chaos **27**, 113103 (2017).
[37] A. Hampton and D. H. Zanette, Measure Synchronization in Coupled Hamiltonian Systems, Phys. Rev. Lett. **83**, 2179 (1999).
[38] X. G. Wang, M. Zhan, C.-H. Lai, and G. Hu, Measure synchronization in coupled ϕ^4 Hamiltonian systems, Phys. Rev. E **67**, 066215 (2003).
[39] M. Sano and Y. Sawada, Measurement of the Lyapunov spectrum from a chaotic time series, Phys. Rev. Lett. **55**, 1082 (1985).
[40] S. J. Pan and Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. **22**, 1345 (2010).
[41] F. Zhuang, P. Luo, Q. He, and Z. Shi, Survey on transfer learning research, J. Soft **26**, 26 (2015).
[42] M. Schmidt and H. Lipson, Distilling free-form natural laws from experimental data, Science **324**, 81 (2009).
[43] T. Wu and M. Tegmark, Toward an artificial intelligence physicist for unsupervised learning, Phys. Rev. E **100**, 033311 (2019).
[44] R. Iten, T. Metger, H. Wilming, L. Rio, and R. Renner, Discovering Physical Concepts with Neural Networks, Phys. Rev. Lett. **124**, 010508 (2020).
[45] Y. Mototake, Interpretable conservation law estimation by deriving the symmetries of dynamics from trained deep neural networks, Phys. Rev. E **103**, 033303 (2021).