Journal of Industrial Information Integration 35 (2023) 100503

Contents lists available at ScienceDirect

Journal of Industrial Information Integration

journal homepage: www.elsevier.com/locate/jii

Full length article

Prediction and identification of nonlinear dynamical systems using machine learning approaches

Leisheng Jin [a], Zhuo Liu [a], Lijie Li [b,*]

[a] College of Integrated Circuit Science and Engineering, Nanjing University of Posts and Telecommunications, Nanjing, 210023, Jiangsu, China
[b] College of Engineering, Swansea University, Swansea, SA1 8EN, Wales, United Kingdom

ARTICLE INFO

Keywords:
Prediction
Chaotic dynamical systems
Identification
Reservoir computing
Runge–Kutta

ABSTRACT

Nonlinear dynamical systems are widely implemented in many areas. The prediction and identification of these dynamical systems purely based on observational data are of great significance for practical applications. In the work, we develop a machine learning based approach called Runge–Kutta guided next-generation reservoir computing (RKNG-RC). The proposed scheme can process data information generated by the most complicated nonlinear dynamical systems such as chaotic Lorenz63 system even with noise, and experimental systems such as chaotic Chua's electronic circuit, showing an outstanding ability for prediction tasks. More importantly, the RKNG-RC is found to have distinctive interpretability that from the trained weights the ordinary differential equation governing the observable data can be deduced, which is beyond the processing capacities of traditional approaches. The work provides an efficient platform for processing information generated by various dynamical systems.

1. Introduction

The prediction and identification of dynamical models that underpin systems in industry [1,2], physics [3], engineering [4], biology [5], and social network [6,7] has been an important and challenging topic. Most of traditional methods rely heavily on expert intuition and suffer a large computational cost. As a result, it is seen recently a focused attention on developing more efficient data-driven modeling methods [8–11]. Particularly, machine learning offers a shortcut for analyzing limited observational data and learning the underlying nonlinear dynamics [12–14]. Various of neural networks are proposed for discovering representations of complex systems, closure modeling and chaotic time-series prediction [15–17]. Among these, the reservoir computing (RC) [18], as a type of recurrent neural networks (RNN), is deemed as one of the most attractive platforms for processing information generated by various of dynamical systems [19–22].

The RCs have been developed from the original Echo-state network (ESN)-based to the time-delayed based [23], and until recently the next generation of reservoir computing (NG-RC) [24]. The core idea of RCs is to map the limited state values (input) into a much higher dimensional networks realized by a reservoir, where the reservoir could be time-delayed, or constructed by randomly connected neurons, or replaced by specially designed feature vectors, and a weight matrix that couples the reservoir state and output layer is trained to find the desired function between the input and output. Previous studies have shown that RC is competent for tasks such as prediction of chaotic systems, reconstruction chaotic attractors, nonlinear behaviors controlling [21,25–28] etc. However, in general, there is a lack of interpretability of using RC in the sense that the RC behaves like a black-box having many hyperparameters to be adjusted. The trained RC can be seen as a data-driven model but it is failed to give explicit information about the mathematical structure behind the observable data.

In this paper, we propose a new type of NG-RC based framework in which the structure construction is guided by Runge–Kutta algorithm and training procedures is optimized. The explored method is capable of predicting chaotic time series that might be noise polluted or generated by experiment during which the data is somehow affected by practical equipment and measurement, as well as reconstructing chaotic attractors behaving like a data-driven model. Both numerical simulations and real experiment have been conducted to verify the proposed method. More importantly, the proposed scheme is proven to be competent for capturing the internal relationships between input states, and one can identify the correct form of ODE together with relevant system parameters based on the Runge–Kutta guided interpretative process. The data generated by Lorenz chaotic systems with/without noise and the real Chua's circuit are used for the numerical validations. Comparisons in both the prediction and identification task with the most popular Sparse Identification of Nonlinear Dynamical systems

* Corresponding author.
E-mail addresses: jinls@njupt.edu.cn (L. Jin), l.li@swansea.ac.uk (L. Li).

https://doi.org/10.1016/j.jii.2023.100503
Received 18 April 2023; Received in revised form 19 June 2023; Accepted 19 July 2023
Available online 24 July 2023
2452-414X/© 2023 The Author(s). Published by Elsevier Inc. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

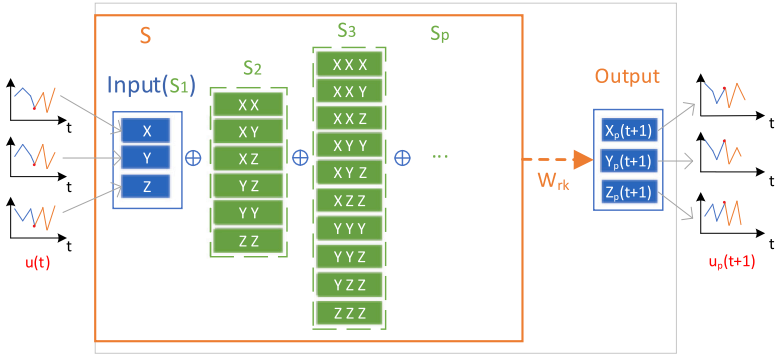


Fig. 1. The structure of the RKNG-RC. The relationship between $\boldsymbol{S}$ and $\boldsymbol{S_2}, \boldsymbol{S_3}, \ldots, \boldsymbol{S_p}$ is described by Eq. (2). The $\oplus$ denotes vector concatenation operation.

(SINDy) algorithm are also given. Though the proposed method in this work is limited for studying ODE, it can shed light for developing machine learning based data-driven methods that could be extended into in the fields such as aerodynamics, fluid dynamics described by partial differential equations (PDEs).

2. Theories

2.1. Problem description

We consider a general dynamical process that can be described by the following ordinary differential equation (ODE) (Eq. (1)):

$$\dot{\boldsymbol{u}}(\boldsymbol{t}) = f(\boldsymbol{u}(\boldsymbol{t})) \tag{1}$$

where the f is unknown, the state vector $\boldsymbol{u}(\boldsymbol{t}) \in R^m$ and time $t \in [0, T]$. Suppose that the state $\boldsymbol{u}(\boldsymbol{t})$ is observable only for a limited time range of $[0, T_1]$, and we have corresponding time series data of state $\boldsymbol{u}(\boldsymbol{t}), i.e., \ (\boldsymbol{u}(\boldsymbol{t_0}), \boldsymbol{u}(\boldsymbol{t_1}), \boldsymbol{u}(\boldsymbol{t_2}), \ldots, \boldsymbol{u}(\boldsymbol{t_{T1}}))$. For practical consideration, we assume that the observable data can be noise-polluted or generated by real experiment. Our objective is two-fold. First, it aims to construct a distinctive NG-RC framework that can learn from the observable data, and be further seen as a data-driven model which can predict the future state evolution and reproduce the underlying dynamics such as chaos attractor. Second, it is expected that the proposed method can infer the explicit ODE structure and relevant system parameters in f.

2.2. Structure of the proposed RKNG-RC

The traditional RC is generally composed of three parts: an input layer, a reservoir and an output layer. The input layer feeds the input vector via a weight matrix $\boldsymbol{W_{in}}$ to the reservoir composed of N interconnected nodes. The output vector can be derived through weight matrix $\boldsymbol{W_{out}}$ which couples the dynamical state of reservoir nodes and the output layer.

The structure of the RKNG-RC is shown in Fig. 1, which learns the observable data over a limited time range and extract the relationship between state at time t and $t + 1$. Specifically, it uses $\boldsymbol{u}(\boldsymbol{t})$ at time t to create a so-called feature vector $\boldsymbol{S}$ that is composed of some sub-vectors of $\boldsymbol{S_2}, \boldsymbol{S_3}, \ldots, \boldsymbol{S_p}$, where p indicates the number of states involved in every nonlinear terms in sub-vectors. The $\boldsymbol{S}$ can be written by (Eq. (2)):

$$\boldsymbol{S} = \boldsymbol{S_1} \oplus \boldsymbol{S_2} \ldots \oplus \boldsymbol{S_p} \tag{2}$$
$$\boldsymbol{S_1} = [X, Y, Z]$$
$$\boldsymbol{S_2} = [XX, XY, XZ, YY, YZ, ZZ]$$

where $\oplus$ represents vector concatenation operation. As in the original NG-RC, the feature vector is generally composed of the linear part and nonlinear part involving states extended into k time delayed steps (i.e., nonlinear vector autoregression (NVAR)), which hinders the interpretability. We can narrow down the feature library of the original NG-RC by referring the idea that when using Runge–Kutta formula to solve an ODE the state variable at time step $t + 1$ can be calculated iteratively based on state variables at time step t. For example, a second-order Runge–Kutta formulas (Eqs. (3)–(5)) is given by:

$$\boldsymbol{y}(\boldsymbol{t_{m+1}}) = \boldsymbol{y}(\boldsymbol{t_m}) + h(\boldsymbol{k_1} + \boldsymbol{k_2})/2 \tag{3}$$
$$\boldsymbol{k_1} = f(t_m, \boldsymbol{y}(\boldsymbol{t_m})) \tag{4}$$
$$\boldsymbol{k_2} = f(t_m + h, \boldsymbol{y}(\boldsymbol{t_m}) + \boldsymbol{k_1} h) \tag{5}$$

where f represents the function linking state variables and their derivatives, $\boldsymbol{y}(\boldsymbol{m})$ presents the state values at step m. t_m is the time at step m, and h is the step length. From the formula, it can be concluded that the state values at $t + 1$ are actually functions of various nonlinear terms constructed by state variables at t. Therefore, it is reasonable for using all the possible combinations of state variables at t to construct sub-vectors $\boldsymbol{S_p}(p = 1, 2, ..)$ that are combined to create the feature vector $\boldsymbol{S}$. Based on $\boldsymbol{S}$, an output weight matrix W_{rk} should be existed for satisfying the following relation (Eq. (6)):

$$\boldsymbol{u}(\boldsymbol{t+1}) = \boldsymbol{W_{rk}} \times \boldsymbol{S} + \boldsymbol{u}(\boldsymbol{t}) \tag{6}$$

Next, a training process is specially designed and conducted for deriving the desired $\boldsymbol{W_{rk}}$. The training process is conducted through the following steps: (take a three-dimensional dynamical system for example).

Step 1: Preliminary ridge regression. With the feature vector $\boldsymbol{S}$, a preliminary weight matrix $\boldsymbol{W}$ can be calculated by the ridge regression [20], that is:

$$\boldsymbol{W} = [\boldsymbol{u}(\boldsymbol{t+1}) - \boldsymbol{u}(\boldsymbol{t})]\boldsymbol{S}^T(\boldsymbol{S}\boldsymbol{S}^T + \lambda_1 \boldsymbol{I})^{-1} \tag{7}$$

In Eq. (7), $\boldsymbol{u}$ is the desired output, $\boldsymbol{I}$ is an identity matrix and λ_1 is ridge parameter for preventing over-fitting.

Step 2: Effective weights marking. Based on the $\boldsymbol{W}$ derived in Step 1, weight values that play a more important role during the ridge regression should be emphasized, and a marking procedure is introduced here, which can be expressed as in Eq. (8):

$$\boldsymbol{W_v} = h(\boldsymbol{W} - c) \tag{8}$$

where h is the step function, $\boldsymbol{W_v}$ is 0–1 matrix (Boole) which can indicate whether the corresponding item in the matrix participates in the operation, c is predefined weight threshold. It can be seen that only if the value of terms in $\boldsymbol{W}$ reach a certain level, the corresponding terms in $\boldsymbol{W_v}$ will be set to 1.

Step 3: Second ridge regression with marked terms. Based on step 1 and 2, a second regression process is developed, which is given by Eq. (9):

$$\boldsymbol{W^i_{rk}} = [\boldsymbol{u}(\boldsymbol{t+1}) - \boldsymbol{u}(\boldsymbol{t})]^i(\boldsymbol{S} \cdot \boldsymbol{W_v}^i)^T[(\boldsymbol{S} \cdot \boldsymbol{W_v}^i)(\boldsymbol{S} \cdot \boldsymbol{W_v}^i)^T + \lambda_2 \boldsymbol{I}]^{-1} \tag{9}$$

where $\boldsymbol{u}^i$ means the i_{th} row of the matrix $\boldsymbol{u}$, $\boldsymbol{W_{rk}}$ is the final output weight matrix based on ridge regression, and $'\cdot'$ means the multiplying

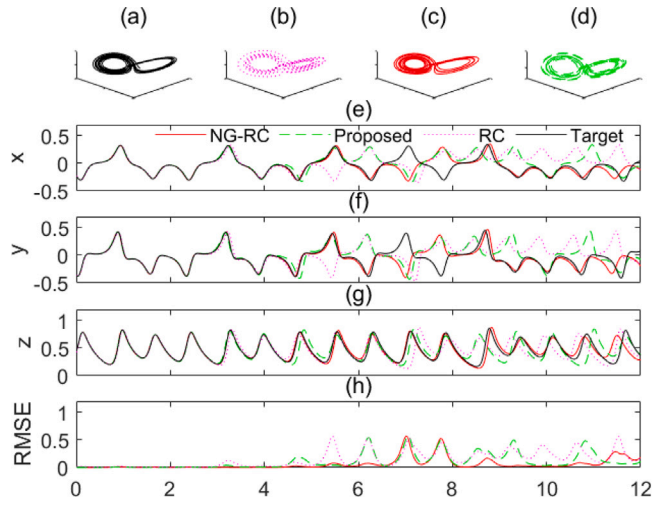


Fig. 2. Performance comparisons of using RC, NG-RC and the RKNG-RC to reconstruct chaotic attractor and predict time series based on data generated by chaotic Lorenz system. The RMSE calculation is given in (h). The result corresponding to each method is plotted by a specific color. Some parameters are taken as follows: $p = 3, \lambda_1 = 4 \times 10^{-3}, c = 6 \times 10^{-3}, \lambda_2 = 2 \times 10^{-7}$ and $\lambda_{NG-RC} = 2 \times 10^{-8}$ (λ_{NG-RC} in the following is the ridge parameter of the original NG-RC.). RC parameters in this test and following two tests: spectral radius is 0.6; connection degree is 1.5; reservoir size is 1000.

operation between all corresponding entries in the matrix. During the second regression, only the output weights with values greater than the predefined threshold are considered to be participated in the regression process.

Following the above steps to train the model, a final output weight matrix $\boldsymbol{W_{rk}}$ can be found, based on which the trained RKNG-RC can be further used for prediction and identification tasks, i.e.,: (1) prediction of the evolution of state variables using limited observable data and chaos attractor reconstruction; (2) identifying the ODE structure and system parameter values by adopting a reverse Runge–Kutta procedure. Note that during the prediction process, the $\boldsymbol{W_{rk}}$ is fixed, and one can make the output vector $\boldsymbol{u'}$ feeding back into the input layer, the RKNG-RC can automatically run itself for generating future states. The output vector is calculated by Eq. (10):

$$\boldsymbol{u_p}(t+1) = \boldsymbol{u}(t) + \boldsymbol{W_{rk}}\boldsymbol{S} \tag{10}$$

where $\boldsymbol{u_p}(t+1)$ represents the predicted value of the studied model.

3. Numerical and experimental validation

3.1. Numerical case: Lorenz63 system

The Lorenz63 system is firstly used for verifying the RKNG-RC. The equation of Lorenz63 system is given by [29]:

$$\begin{aligned} \dot{x} &= \sigma(y - x) \\ \dot{y} &= \gamma x - y - xz \\ \dot{z} &= xy - \beta z \end{aligned} \tag{11}$$

where $\sigma = 10$, $\gamma = 28$ and $\beta = 8/3$, and in such a parameter setting the system works in chaotic regime. A training data set D containing 20 000 time steps of states (x, y, z) with discarding the transient states is generated using Adams–Bashforth–Moulton PECE solver(ode113) [30] based on Eq. (11). Part of D is seen as observable data (training data), and the rest is used as target signal for testing the prediction ability of the RKNG-RC. During the training process, the feature vector S in this case is constructed with $\boldsymbol{S_1}, \boldsymbol{S_2}, \boldsymbol{S_3}$, i.e., $p = 3$. After training, the RKNG-RC can predict all the state (x, y, z) evolution precisely for a certain time, as shown by the green dotted line in Fig. 2(e)–(g). Apart form

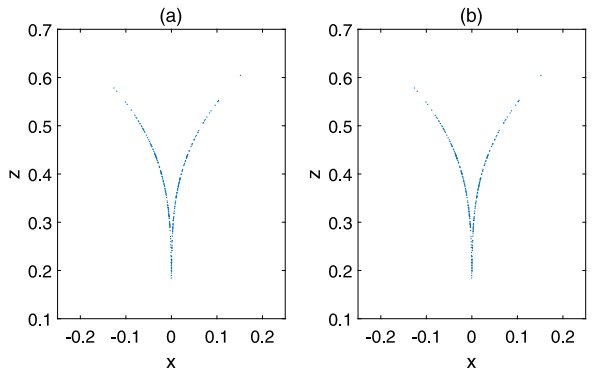


Fig. 3. Poincare section of the original system (a), and Poincare section based the data generated by RKNG-RC after being trained (b).

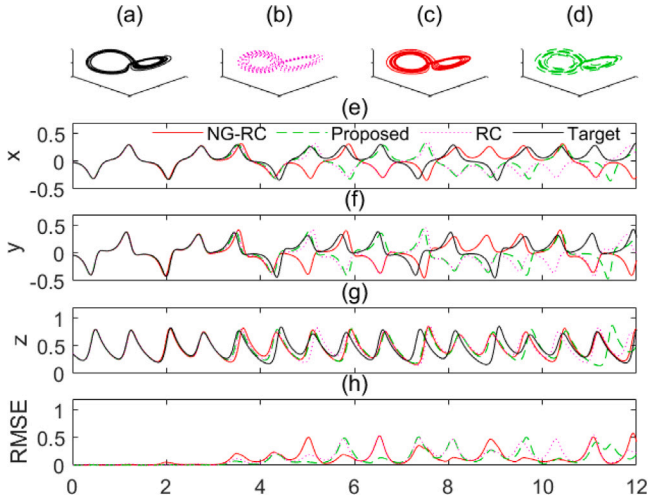
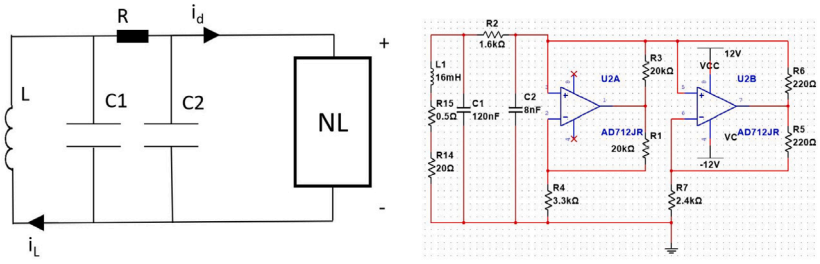


Fig. 4. Performance comparisons of using RC, NG-RC and RKNG-RC on chaotic attractor reconstruction and time series prediction based on data generated by chaotic Lorenz system with adding noise. The RMSE calculation is given in (h). The result corresponding to each method is plotted by a specific color. Some parameters are taken as follows: $p = 3, \lambda_1 = 8 \times 10^{-3}, c = 2 \times 10^{-3}, \lambda_2 = 2 \times 10^{-5}$ and $\lambda_{NG-RC} = 2 \times 10^{-7}$.

prediction, the reconstruction of chaotic attractor of Lorenz system based on the RKNG-RC is also studied. The result is shown with green color in Fig. 2(d), where it is seen that the RKNG-RC can reconstruct chaotic attractor as the original one shown in black color in Fig. 2(a). To compare, we have also calculated the prediction performance using the original NG-RC and RC, respectively, and the results are present by red solid and pink dotted line, respectively, in Fig. 2(e)–(g). It is shown that our RKNG-RC is outperformed the original NG-RC and RC in terms of prediction length. In addition, the attractors reconstructed by the RKNG-RC and RC are also given in Fig. 2(b) and (c), respectively. The RMSE of different methods is present in Fig. 2(h). The results in Fig. 2 proves that the RKNG-RC is competent for chaotic series prediction. To achieve such a comparable performance, it is worth mentioning that the calculation efficiency of the RKNG-RC is higher than the original NG-RC and RC because of effective weights marking.

In addition, the Poincaré's plots ($y = 0$) of the original Lorenz system and that reproduced using data generated by the trained RKNG-RC by D are also compared. As shown in Fig. 3(b), the reproduced one using 600 000 generated data points is overlapping with the one plotted by the original system, which proves that the long-time evolution behavior of the NG-RC can capture the nonlinear dynamics feature of the Lorenz63 system.

To verify the noise robustness of the RKNG-RC, the case of data with noise affected is used for training. Taking the Lorenz63 system



(a) Chua's circuits. (b) Design of Chua's circuits.

(c) The hardware realization of chaotic Chua's circuit (d) The chaotic attractor of the Chua's circuits displaying in an oscilloscope.

Fig. 5. The design of chaotic Chua's circuit and its experimental implementation.

subjected to noise as the data generator (Eq. (12)):

$$\begin{aligned}\dot{x} &= \sigma(y-x)+R\\ \dot{y} &= \gamma x - y - xz + R\\ \dot{z} &= xy - \beta z + R\end{aligned} \tag{12}$$

where R is randomly distributed at $[-10, 10]$ representing the white noise outside the system. By increasing the training error tolerance, the RKNG-RC can also work well. The results for this case are shown in Fig. 4. Both the prediction and reconstruction of chaotic attractor prove the noise robustness of the RKNG-RC. In addition, we compared performance with other methods such as the original NG-RC and RC. The results are also given in Fig. 4. Again, it can be seen that the RKNG-RC outperformed.

3.2. Experimental case: Chua's chaotic circuit

The Chua's circuit is one of the most representative electronic system that can exhibit chaos. The dynamical equation (Eq. (13)) for describing the Chua's circuit is given by:

$$\begin{aligned}L\dot{i}_L &= -v_1 - r_L i_L\\ C_1\dot{v}_1 &= (v_1 - v_2)/R + i_L\\ C_2\dot{v}_2 &= (v_1 - v_2)/R - i_d(v_2)\end{aligned} \tag{13}$$

where v_i is the voltage across capacitor C_i, i_L is the current in the inducer, and the current flowing through Chua's diode is given by Eq. (14):

$$i_d = \begin{cases} k_1 v_2 & abs(v_2) < v\\ k_2 v_2 + k_1 v - k_2 v & abs(v_2) > v\end{cases} \tag{14}$$

where v is the threshold of v_2. The v can be controlled by different slopes k_2 and k_1. In this section, we verify the RKNG-RC in a more practical environment by building a real Chua's chaotic circuit. The circuit diagram, design and experimental implementation are given in Fig. 5. The circuit consists of an inducer L with r as its inner resistor, two capacitors C_1 and C_2, a linear resistor R, and a piecewise linear resistor R_n. The only nonlinear element R_n is a two-terminal piecewise linear resistor denoted by 'Chua's diode'. The ideal values of all components are shown in Fig. 5(b). The hardware realization of the Chua's circuit is shown in Fig. 5(c). The Fig. 5(d) shows the chaotic attractor of the experimental Chua's circuit displaying on an oscilloscope.

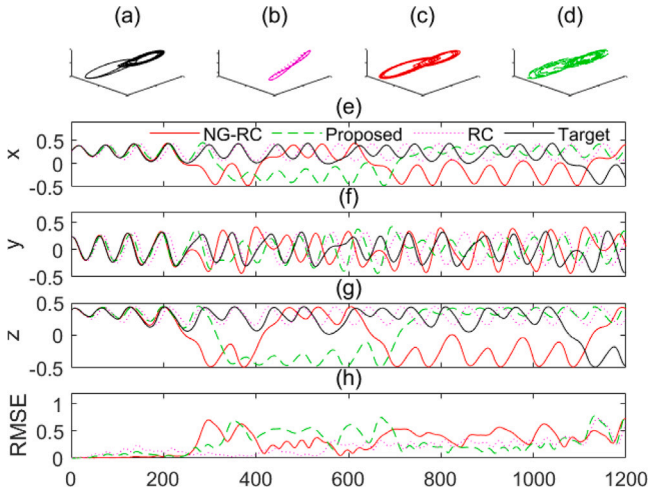


Fig. 6. The reconstruction of chaotic attractor of Chua's circuits and prediction of chaotic series using RC, NG-RC and the proposed NG-RC. The training data having 8000 time steps (2 us per step) is generated by real experiment. Some parameters are taken as follows: $p = 5, \lambda_1 = 1.6\times10^{-2}, c = 1.6\times10^{-2}, \lambda_2 = 2\times10^{-3}$ and $\lambda_{NG-RC} = 2\times10^{-4}$.

During the experiment, the data acquisition is accomplished by an oscillator. The collected data, far from ideal but affected by practical equipment, is divided as training and testing part. Subsequently, we try to validate that the RKNG-RC can learn the practical data and predict the state variables' evolution. By setting $p = 5$ and using 8000 time steps to train the proposed NG-RC, the prediction performance is then studied. The result is shown in Fig. 6(e)–(g). It can be seen that the RKNG-RC performs well to predict the real chaotic data in all dimensions. The reconstruction of Chua's chaotic attractor using the RKNG-RC is validated and compared with the one plotted by the original model, as shown in top row of Fig. 6. It is proved that the RKNG-RC after being trained can learn the dynamics behind the

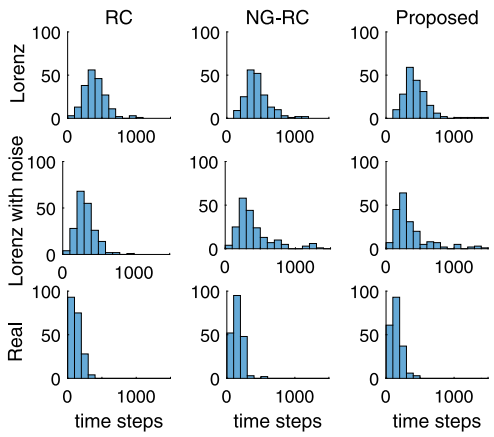


Fig. 7. The histogram of prediction length distributions. The horizontal axis denotes the prediction length. Data sets corresponding the above studied cases (Lorenz, Lorenz with noise and real Chua's circuit) are investigated. Methods between RC, NG—RC and the proposed are compared. It is shown that the proposed method has similar performances with the NG-RC. The actual training time with the new method is much shorter than the other two.

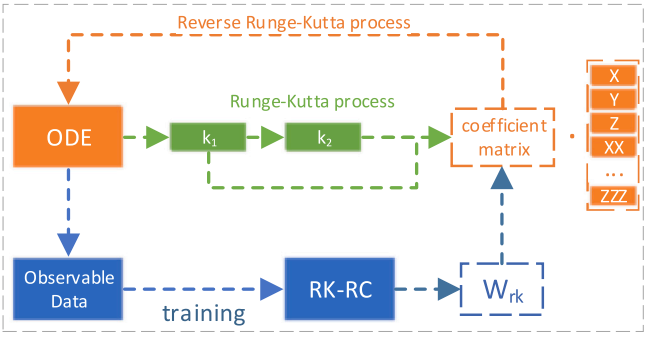


Fig. 8. Process of deducing the ODE.

real data and generate the similar attractor. To compare, the performance for predicting real data using the original NG-RC and RC are also investigated. The results, plotted with red solid and pink dotted line, respectively, are given in Fig. 6(e)–(g). It shows again that the RKNG-RC outperformed over RC and demonstrated at least comparable performance with the original NG-RC. However, the RKNG-RC has the merit that it is more efficient for processing the data due to the feature vector optimization.

Furthermore, To validate the proposed method, we study the predication performance by using about 200 trajectories that start from the scattered points in chaotic attractor of Lorenz system. And the distribution of prediction length, defined by the time step when the prediction differs from the true value by 0.25 (normalized value), is investigated and given in Fig. 7. To have a comprehensive comparison, the above studied numerical and experimental cases: Lorenz, Lorenz with noise and the real Chua's circuit, using three different methods: RC, the NG-RC and the proposed, are considered. In Fig. 7, it is seen that the proposed method can achieve a robust and comparable prediction. Here, we would like to emphasize that the advantage of employing the proposed NG-RC for conducting prediction task is that it can provide higher efficiency, which is characterized by reduction of multitude of metaparameters, less training data and shorter training time required. To achieve a predication length that is comparable with RC and the original NG-RC, the proposed method consumes less training time because of the effective weights marking.

4. ODE identification behind the data

4.1. Deducing the ODE structure

In this section, we illustrate how to infer ODE structure behind the data using the RKNG-RC. The specific process for deducing ODE structure using the RKNG-RC is shown in Fig. 8. The derived weight W_{rk} after being trained by observational data can reflect the relations between input state variables. Based on the values of W_{rk}, the explicit information about how terms in sub-vectors in the RKNG-RC play their role can be obtained. Further, by utilizing the reverse Runge–Kutta formula, the W_{rk} corresponding to their counterparts in coefficient matrix that is deduced from Runge–Kutta formula, indicated by green line in Fig. 8, can be used for deducing the original ODE structure.

We now take the Lorenz chaotic system as an example. In Table 1, the very left column lists all the selected terms in feature vector S corresponding to dimension: x, y and z. The columns titled with x_1, y_1 and z_1 represent the coefficient values linking the state variables (x, y and z) at time t and $t+1$ in the time-stepping form of Lorenz system ($\sigma = 10, \gamma = 28, \beta = 8/3$) calculated by Runge–Kutta method. The columns titled with x_2, y_2 and z_2 represent the derived weight values after the RKNG-RC being trained by data generated by Lorenz system using ode113. The columns titled with x_3, y_3 and z_3 represent the derived weight values after the RKNG-RC being trained by data generated by Lorenz system with noise using ode113.

Based on the weight values given in Table 1, we can deduce the explicit ODE structure step by step according to the process given in Fig. 8. Assume that we have the weight values listed in columns of x_2, y_2 and z_2, the original ODE structure can be deduced through the three steps. **Step 1:** Given the existence of 3-order polynomial terms such as yxx and zxx in dimension y and z, it can be inferred that the original ODE structure should have 2-order polynomial terms xz in dimension y and z. **Step 2:** According to the 2-order polynomial term zx appeared in dimension x and y, it can be inferred that the dimension y of the original ODE should have 2-order polynomial term xz, and the dimension x of the original ODE should have term y. **Step 3:** Based on the step 2 and according to the weight terms in dimension z, the z dimension of the original ODE should have 2-order polynomial terms xy, and the dimension z and x of the original ODE should have terms x and y.

Alternatively, the selection criteria of the proposed method can be put with a more straight and general way, which can include two steps, i.e.,:

Step 1: In Table 1, identify the nonlinear terms with largest and second largest weight values and all the linear terms with non-zero weight values, the ODE structure can then be constructed based on these terms directly;

Step 2: Using the RK2 to discretize the deduced ODE, and check if the coefficients of the main terms are consistent with the weight values obtained from the training by real data.

Follow the above mentioned steps, the inferred structure of ODE is given as:

$$\begin{aligned} \dot{x} &= ay - bx \\ \dot{y} &= cx + dy - exz \\ \dot{z} &= fxy - gz \end{aligned} \tag{15}$$

Note that during the above deducing process, the 3-order polynomial terms yyx in dimension x, zzx in dimension y and zyy in dimension z are unused and seen as noise.

There is other method to infer the ODE structure. For example, one can directly identify the weight values that are specially larger as the key terms for deducing the original ODE, such as 0.4595 corresponding to yx, −0.4898 corresponding to zx. These two values can actually reflect the original structure information on the premise that one is aware of the coefficient reduction principle during the iteration process of Runge–Kutta formula.

Table 1
The coefficient values (columns of 2,3,4) are derived during the time-stepping calculation of Lorenz system using by Runge–Kutta method with $h = 0.01$; Columns of 4,5,6 (7,8, 9) list the weight values of W_{rk} after the RKNG-RC being trained by the data generated by Lorenz system without and with noise.

Terms in S	x_1	y_1	z_1	x_2	y_2	z_2	x_3	y_3	z_3
xx	0	0	0.0630	0	0	0.0712	0.0049	0.0220	0.0695
yx	0	0	0.4731	0	0	0.4595	−0.0015	−0.0130	0.4770
yy	0	0	0.0248	0	0	0.0304	0	0	0.0247
zx	−0.0250	−0.4665	0	−0.0242	−0.4898	0	−0.0248	−0.4988	0
zy	0	−0.0243	0	0	−0.0168	0	0	−0.0173	0
zz	0	0	0	0	0	0	0	0	0
xxx	0	0	0	0	−0.0028	0	0.0015	−0.0314	0
yxx	0	−0.1125	0	0	−0.1413	0	0	−0.1114	0.0032
yyx	0	−0.0125	0	−0.0036	0.0078	0	−0.0033	0	−0.0031
yyy	0	0	0	0	0	0	0	0	0
zxx	0	0	−0.1125	0	0	−0.1224	−0.0052	−0.0287	−0.1194
zyx	0	0	−0.0125	0	0	−0.0065	0	0.0156	−0.0151
zyy	0	0	0	0	0	−0.0110	0	0.0079	−0.0095
zzx	0	0	0	0	0.0171	0	0	0.0277	0
zzy	0	0	0	0	0	0	0	0	0
zzz	0	0	0	0	0	0	0	0	0
x	−0.0810	0.2646	0	−0.0814	0.2727	0	−0.0810	0.2749	0
y	0.0945	0.0041	0	0.0947	0.0040	0	0.0945	0	0
z	0	0	−0.0263	0	0	−0.0263	0	0	−0.0263

Table 2
The calculation of coefficient values based on Runge–Kutta method.

Terms in S	x	y	z
xx	0	0	$\gamma h^2/2 - \gamma h^3\sigma/2$
yx	0	0	$h + h^3\sigma/2 - h^2/2 - h^2\sigma/2 - \beta h^2/2 + \gamma h^3\sigma/2$
yy	0	0	$h^2\sigma/2 - h^3\sigma/2$
zx	$-h^2\sigma/2$	$+h^2/2 - h + \beta h^2/2 + h^2\sigma/2 - \beta h^3\sigma/2$	0
zy	0	$-h^2\sigma/2 + \beta h^3\sigma/2$	0
zz	0	0	0
xxx	0	0	0
yxx	0	$h^3\sigma/2 - h^2/2$	0
yyx	0	$-h^3\sigma/2$	0
yyy	0	0	0
zxx	0	0	$h^3\sigma/2 - h^2/2$
zyx	0	0	$-h^3\sigma/2$
zyy	0	0	0
zzx	0	0	0
zzy	0	0	0
zzz	0	0	0
x	$h^2\sigma^2/2 - h\sigma + \gamma h^2\sigma/2$	$\gamma h - \gamma h^2/2 - \gamma h^2\sigma/2$	0
y	$-h^2\sigma/2 - h^2\sigma^2/2 + h\sigma$	$h^2/2 - h + \gamma h^2\sigma/2$	0
z	0	0	$\beta^2 h^2/2 - \beta h$

4.2. Deriving system parameters

Assuming that the ODE structure deduced in Section 4.1 is the Lorenz chaotic system, Eq. (15) can be rewritten as the commonly seen form which has parameters σ, γ, and β. The system parameters can be further derived using Runge–Kutta method. First, we can write the following relations (Eqs. (16)–(18)):

$$x_{t+1} = x_t + h^2\sigma^2 x_t/2 - h^2\sigma y_t/2 - h^2\sigma^2 y_t/2 - h\sigma x_t + h\sigma y_t + \gamma h^2\sigma x_t/2 - h^2\sigma x_t z_t/2 \tag{16}$$

$$y_{t+1} = y_t + h^2 y_t/2 - h y_t - \gamma h^2 x_t/2 + h^2 x_t z_t/2 - h^2 x_t^2 y_t/2 + \gamma h x_t - h x_t z_t - \gamma h^2\sigma x_t/2 + \gamma h^2\sigma y_t/2 + \beta h^2 x_t z_t/2 + h^2\sigma x_t z_t/2 - h^2\sigma y_t z_t/2 - h^3\sigma x_t y_t^2/2 + h^3\sigma x_t^2 y_t/2 - \beta h^3\sigma x_t z_t/2 + \beta h^3\sigma y_t z_t/2 \tag{17}$$

$$z_{t+1} = z_t + \beta^2 h^2 z_t/2 - h^2 x_t y_t/2 + \gamma h^2 x_t^2/2 + h^2\sigma y_t^2/2 - h^3\sigma y_t^2/2 - h^2 x_t^2 z_t/2 - \beta h z_t + h x_t y_t - \beta h^2 x_t y_t/2 - h^2\sigma x_t y_t/2 + h^3\sigma x_t y_t/2 - \gamma h^3\sigma x_t^2/2 + h^3\sigma x_t^2 z_t/2 - h^3\sigma x_t y_t z_t/2 + \gamma h^3\sigma x_t y_t/2 \tag{18}$$

Based on these equations, the coefficient of each term is listed in Table 2.

Matching the weight values listed in columns of x_2, y_2 and z_2 in Table 1 with Table 2, we can easily have a set of equations to calculate the system parameters of the Lorenz system (Eq. (19)), i.e.,:

$$\begin{aligned}
(-h^2\sigma/2 - h^2\sigma^2/2 + h\sigma) \times K &= 0.0947 \times K \\
(h^2/2 - h + \gamma h^2\sigma/2) \times K &= 0.0040 \times K \\
(\beta^2 h^2/2 - \beta h) \times K &= -0.0263 \times K
\end{aligned} \tag{19}$$

where K is the zoom factor meaning the coefficient by which the original data is multiplied during data normalization. Here, K is set to 50. Based on equations, we can get $\sigma = 10.022, \gamma = 27.839$ and $\beta = 2.6655$.

With the identification results the prediction task results are given in Fig. 9. The calculated parameters achieve an error rate of 1% within the original ODE used for generating the data. And if h is unknown, relevant parameters can still be solved by introducing other equations. The errors of the derived parameters are attributed to the numerical calculation schemes employed for data generation. There are unavoidable errors existed during the calculation process, e.g., the error brought by numerical solver leads to error in the weight values that are later used for inferring the system parameters.

We can also use the weight values listed in columns of x_3, y_3 and z_3 in Tables 1 and 2 to calculate the system parameters of the Lorenz system subjected to noise. Similar to the above procedures, we choose the coefficients corresponding to the first-order items such as x, y, z to

Table 3
$\lambda_1 = 2e-3,\ \lambda_2 = 2e-5;\ (1)c = 2e-3, (2)c = 8e-3, (3)c = 24e-3.$

Terms in S	x_1	y_1	z_1	x_2	y_2	z_2	x_3	y_3	z_3
xx	0.0023	0.0012	0.0676	0	0	0.0726	0	0	0.0178
yx	0	0	0.4571	0	0	0.4477	0	0	0.4810
yy	0	0	0.0337	0	0	0.0373	0	0	0.0353
zx	−0.0236	−0.4709	0	−0.0249	−0.4747	0	−0.0249	−0.4951	0
zy	0	−0.0198	0	0	−0.0175	0	0	−0.0056	0
zz	0	0	0.0024	0	0	0.0029	0	0	0.0151
xxx	0.0174	0.0483	0.0088	0	0.0459	0	0	−0.0559	0
yxx	−0.0123	−0.0925	0	0	−0.0905	0	0	0	0
yyx	−0.0089	−0.0688	−0.0038	0	−0.0670	0	0	−0.0627	0
yyy	0.0048	0.0173	0	0	0.0159	0	0	0	0
zxx	−0.0033	−0.0019	−0.1075	0	0	−0.1101	0	0	0
zyx	0.0034	0	0.0036	0	0	−0.0123	0	0	−0.0629
zyy	−0.0034	0	−0.0154	0	0	−0.0190	0	0	−0.0097
zzx	−0.00019	0	0	0	0.0023	0	0	0.0317	0
zzy	0	−0.0055	0	0	−0.0076	0	0	−0.0233	0
zzz	0	0	−0.0033	0	0	−0.0041	0	0	−0.0186
x	−0.0813	0.2670	0	−0.0808	0.2681	0	−0.0808	0.2709	0
y	0.0946	0.0026	0	0.0942	0.0020	0	0.0942	0.0005	0
z	0	0	−0.0263	0	0	−0.0268	0	0	−0.0292

infer the ODE coefficients, which is:

$$\begin{aligned}(-h^2\sigma/2 - h^2\sigma^2/2 + h\sigma) \times K &= 0.0945 \times K\\ (\gamma h - \gamma h^2/2 - \gamma h^2\sigma/2) \times K &= 0.2749 \times K\\ (\beta^2 h^2/2 - \beta h) \times K &= -0.0263 \times K\end{aligned} \tag{20}$$

By calculating Eq. (20), we can get $\sigma = 10.001, \gamma = 29.1, \beta = 2.6655$, where only the γ has a large error (3.929%). It is confirmed that the RKNG-RC can also derive the precise parameter values even the data used for training is noise polluted.

4.3. To compare with SINDy

The sparse identification of nonlinear dynamical systems [31] is deemed as one of the most popular method in statistical field. To compare with it, the major difference is that the "library" in SINDy is used for regress-fitting between the measured variables and their's derivatives, while in the modified (original) NG-RC the feature vector, playing the same role as "library" in SINDy, is used for mapping variables at time t and t + 1. The numerical approximation to derivatives from the data can be saved in NG-RC implementation.

Both in prediction and identification tasks, we have compared the performance of using the proposed method and SINDy. Taking the same example of Lorenz in [31], The prediction performance is investigated and compared firstly, as shown in Fig. 9(a).

The ODEs identified using the proposed method (Eq. (21)) and SINDy (Eq. (22)) are derived respectively:

$$\begin{aligned}\dot{x} &= 10.022 \times (y - x)\\ \dot{y} &= 27.839x - y + xz\\ \dot{z} &= 2.6655z + xy\end{aligned} \tag{21}$$

$$\begin{aligned}\dot{x} &= 10.002 \times (y - x)\\ \dot{y} &= 28.01x - y + xz\\ \dot{z} &= 2.669z + xy\end{aligned} \tag{22}$$

It can be concluded that our method can be used equivalently as SINDy for ODE structure and parameters deductions. Moreover, Based on the deduced ODE, the chaotic attractors are generated in a short and long time scale, respectively, as shown in Fig. 9(b) and (c), from which one can further compare the identification ability between SINDy and the proposed method. It is worth to emphasize that the proposed method does not need to know the time scale of the observed data for deducing the correlations between state variables. While the SINDy cannot identify ODE from real data without the knowledge of the time step during the data generation. The original NG-RC and RC also failed to identify the ODE structure in this real task because of the unmanageable feature selection and poor interpretability. The chaotic system is sensitive to the initial value. We have studied the prediction performance by using about 200 trajectories that start from the scattered points in the attractor (indicated by '+' in Fig. 10(a)). And the distribution of prediction length, defined by the time step when the prediction differs from the true value by 0.25 (normalized value), is investigated and given in Fig. 10(b), where it is seen that our method can achieve a robust prediction.

5. Conclusion

A Runge–Kutta guided NG-RC that is able to not only predict chaotic time series but also reveal the explicit information about the mathematical structure behind the observable data is developed. The method shows outstanding performance in dealing with data information generated with noise and/or by practical experimental systems. Both the numerical simulation and real experiment have been conducted for validation. The robustness and comprehensive comparisons are extensively studied and conducted. Overall, the work provides an efficient platform for prediction and identification of nonlinear dynamical systems that are distributed widely in many applications. The method proposed can be applied for processing dynamical information of a single nonlinear unit or network, also sheds light for developing data-driven methods affiliated with practical applications.

CRediT authorship contribution statement

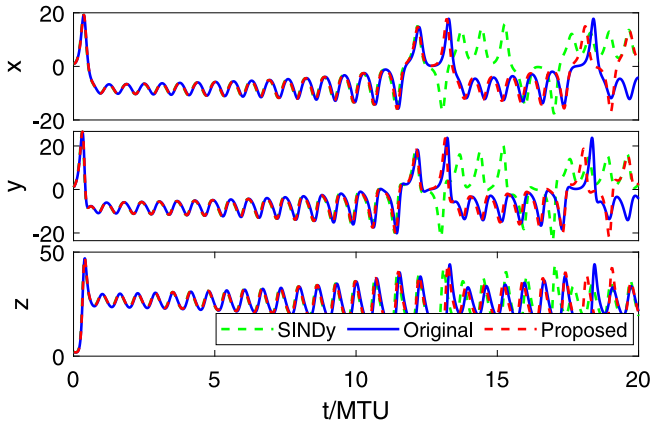
Leisheng Jin: Conceptualization, Methodology, Software, Writing – review & editing. **Zhuo Liu:** Data curation, Methodology, Writing – original draft. **Lijie Li:** Writing – review & editing, Conceptualization, Supervision.

Declaration of competing interest

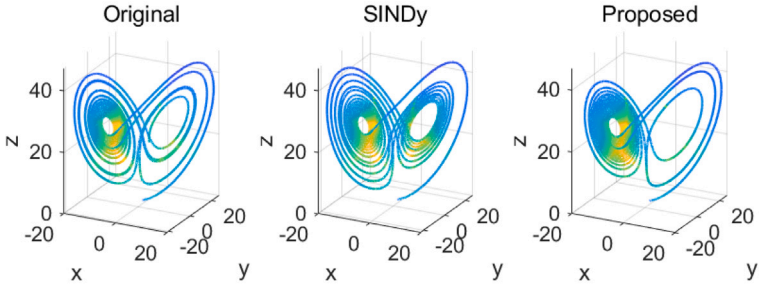
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

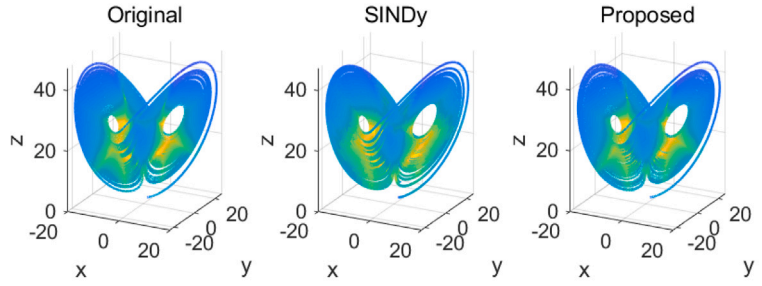
Data will be made available on request.



(a) The dynamical evolution of three-dimensional Lorenz system in 20 MTUs.



(b) The attractors in (0,20) MTU.



(c) The attractors in (0,120) MTU.

Fig. 9. Comparison on the prediction of the dynamical evolution of three-dimensional Lorenz system in 20 MTUs (model time unit) in (a); Comparison on the chaotic attractors generated by the ODE identified by the proposed method, SINDy and the original in short (b) and long time scale (c), respectively.

Acknowledgment

The authors would like to thank Nanjing University of Posts and Telecommunications, Swansea University and STFC (ST/T006455/1) for their support.

Appendix A. The influence of threshold value on deriving the W_{rk}

The threshold value during the optimization process of the feature library can be modulated to eliminate more unused non-zero entries. In SINDy, it also needs to set a threshold to eliminate unnecessary entries during its iterations. The Table 3 below shows the results when adjusting the threshold in the training process.

Appendix B. Identification of the real Chua's chaotic circuit

The most complicated case would come from the real system so that we conducted the practical experiment to verify the identification capability of the proposed idea. For the experimental case of Chua's chaotic circuit, the following ODE can be inferred from the real observing data without any other information (time step, derivative...) during the practical process (Time step in RK is set to 1). The weight matrix W_{rk} is given in Table 4. The structure of ODE can be identified as in Eq. (23):

$$\begin{aligned}\dot{x} &= ax + by + cz\\ \dot{y} &= dx + ey + fz\\ \dot{z} &= gx + hy + iz + jyzz + kz^5\end{aligned} \tag{23}$$

Then use RK2 to calculate the above ODE to obtain the connection between time t and t + 1, and make the coefficients of the above terms equal to the coefficients of the corresponding terms of the weight matrix. Next the ODE's parameter can be obtained by solving the equations and the final identified is derived as in Eq. (24):

$$\begin{aligned}\dot{i_L} &= -0.0312i_L + 0.0823u_1 + 0.0222u_2\\ \dot{u_1} &= -0.3794i_L + 0.0890u_1 + 0.3901u_2\\ \dot{u_2} &= -0.0033i_L + 0.1125u_1 + 0.2779u_2 - 0.2630yzz - 0.2738z^5\end{aligned} \tag{24}$$

Table 4
The 5-order reconstruction weights of real system. $\lambda_1 = 1.6e-2$, $\lambda_2 = 2e-3$, $c = 1.6e-2$.

	x	y	z
x^2	0	0.11636	−0.057377
xy	0.00026428	0.081869	−0.077527
y^2	0	0.025564	−0.027815
xz	0	−0.14693	0.087053
yz	0	−0.041921	0.040019
z^2	0	0.076497	−0.035401
x^3	0.031111	0	−0.015895
x^2y	0	−0.006533	0.079803
xy^2	−0.059174	0.089888	0.13059
y^3	−0.022806	0.072207	0
x^2z	0	−0.073173	0
y^2z	0.037927	−0.13565	−0.028435
xz^2	−0.0046239	−0.16166	−0.06322
yz^2	0.025238	0.070074	−0.26411
z^3	−0.020364	−0.00063083	−0.11588
x^4	0	−0.10134	−0.0037079
x^3y	0	−0.07449	0
xy^3	0	0.049035	−0.042049
y^4	0.016015	−0.079301	0.023202
x^3z	0	−0.032444	0.00065232
x^2yz	0	−0.040504	0
xy^2z	0	0.059149	−0.025187
y^3z	0	0	−0.032519
xyz^2	0	−0.011295	0.061475
y^2z^2	−0.0062025	−0.080974	0.040945
xz^3	0.012158	0.01851	0
yz^3	0	−0.055883	0.078919
z^4	−0.0072263	−0.096409	0.025319
x^5	0	0.06031	0
x^4y	0	0.062291	0.084628
x^3y^2	0	0.063087	0.055211
x^2y^3	0	−0.036913	0
y^5	0	0	0.062986
x^4z	0	0.055879	0
x^3yz	0	0.058498	0.067083
x^2y^2z	0	0.046138	0.063895
xy^3z	0	−0.060715	0
y^4z	0	−0.045635	0
x^3z^2	0	0.047332	0
x^2yz^2	0	0.042269	0.039381
xy^2z^2	0	0	0.057045
y^3z^2	0	−0.083435	0.015554
xz^4	0	0	−0.062458
yz^4	0	0	−0.10933
z^5	0	0	−0.17447
x	−0.026623	−0.23089	−0.031669
y	0.050195	−0.036007	0.067522
z	0.016682	0.19395	0.05287

It should be noted that Because the right-hand side of the standard Chua's circuit model contains nonlinear function i_d rather than the normal polynomials, it is reasonable as well as inevitable that the proposed method deduces linear and cubic terms for representing the original nonlinear function i_d. That is why the inferred equation looks differently from the original one. In the inferred equation, the term of $-0.2630yzz$ and $-0.2738z^5$ and some of linear related terms are generated corresponding to the piecewise function i_d. To prove this point, the fitting of i_d with third-order and fifth-order polynomials are investigated, as shown in the following figure (Fig. 11).

Table 5
RC parameters and prediction length. (Lorenz without noise).

Spectral radius	Connection degree	Reservoir size	Average prediction length
0.9	0.96	1600	336.2
0.9	0.96	1000	361.1
0.9	0.96	200	380.0
0.9	1.5	1600	345.3
0.9	1.5	1000	363.2
0.9	1.5	200	388.9
0.9	0.6	1600	332.0
0.9	0.6	1000	347.0
0.9	0.6	200	286.7
0.6	0.96	1600	371.2
0.6	0.96	1000	377.0
0.6	0.96	200	381.4
0.6	1.5	1600	377.2
0.6	1.5	1000	394.1
0.6	1.5	200	384.9
0.6	0.6	1600	322.2
0.6	0.6	1000	351.1
0.16	0.96	1600	394.0
0.16	0.96	1000	396.4
0.16	0.96	200	379.8
0.16	1.5	1600	391.8
0.16	1.5	1000	388.0
0.16	1.5	200	388.8
0.16	0.6	1600	353.4
0.16	0.6	1000	352.2
0.16	0.6	200	344.4

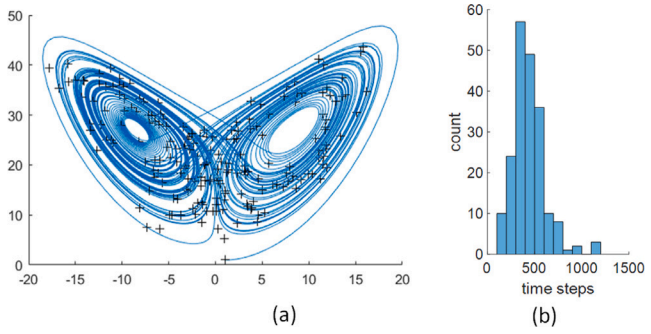


Fig. 10. Study of the impact on the prediction distribution by the initial values of the chaotic systems. The RC hyperparameter are taken as follows: spectral radius (max eigenvalue) of the adjacency matrix: 0.6, connection degree: 0.96, reservoir size: 1600, input weight scaling: 0.5. (a) Attractors with different initial values. (b) Distribution of prediction length.

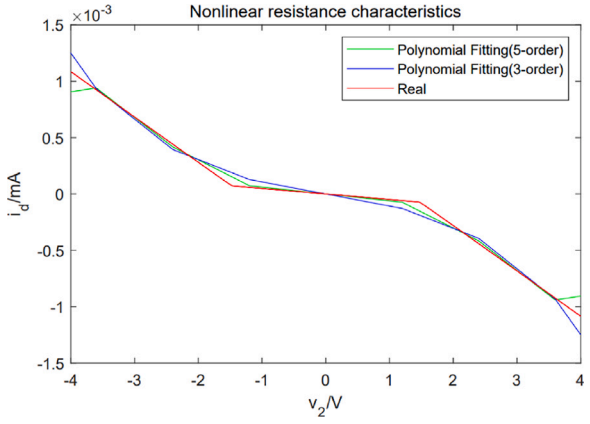


Fig. 11. The polynomials fitting for the nonlinear function i_d.

Appendix C. RC's parameters selection

The RC hyperparameters used in main text have been tested extensively, and the detailed information about the selection are given below in Table 5.

References

[1] Y. Cheng, K. Chen, H. Sun, Y. Zhang, F. Tao, Data and knowledge mining with big data towards smart production, J. Ind. Inf. Integr. 9 (2018) 1–13.

[2] L. Gao, X. Deng, W. Yang, Smart city infrastructure protection: real-time threat detection employing online reservoir computing architecture, Neural Comput. Appl. 34 (2022) 833–842.

[3] J.G. Zheng, Y.D. Cui, Z.J. Zhao, J. Li, B.C. Khoo, Investigation of airfoil leading edge separation control with nanosecond plasma actuator, Phys. Rev. Fluids 1 (2016) 073501.

[4] T.L. Carroll, Adding filters to improve reservoir computer performance, Physica D 416 (2021) 132798.

[5] H. Zou, A.C. Slim, A. Neild, Role of multiple-contact miscibility in drainage from a two-dimensional porous medium, Phys. Rev. Appl. 15 (2021) 054040.

[6] J. Reneuve, L. Chevillard, Flow of spatiotemporal turbulentlike random fields, Phys. Rev. Lett. 125 (2020) 014502.

[7] L.M. Alonso, M.O. Magnasco, Complex spatiotemporal behavior and coherent excitations in critically-coupled chains of neural circuits, Chaos 28 (9) (2018) 093102.

[8] M. Modiri, M.M. Homayounpour, M.M. Ebadzadeh, Reservoir weights learning based on adaptive dynamic programming and its application in time series classification, Neural Comput. Appl. 34 (2022) 13201–13217.

[9] T.S. Das, D. Wilson, Data-driven phase-isostable reduction for optimal nonfeedback stabilization of cardiac alternans, Phys. Rev. E 103 (2021) 052203.

[10] S. Maddu, B.L. Cheeseman, C.L. Müller, I.F. Sbalzarini, Learning physically consistent differential equation models from data using group sparsity, Phys. Rev. E 103 (2021) 042310.

[11] X. Yuzhu, D. Guoli, S. Xueli, Data-based reconstruction of chaotic systems by stochastic iterative greedy algorithm, Math. Probl. Eng. 2020 (2020) (2020) 6718304.

[12] M.L. Alomar, E.S. Skibinsky-Gitlin, C.F. Frasser, et al., Efficient parallel implementation of reservoir computing systems, Neural Comput. Appl. 32 (2020) 2299–2313.

[13] Y. Tang, J. Kurths, W. Lin, E. Ott, L. Kocarev, Introduction to Focus Issue: When machine learning meets complex systems: Networks, chaos, and nonlinear dynamics, Chaos 30 (6) (2020) 063151.

[14] N. Duan, L.-Z. Liu, X.-J. Yu, Q. Li, S.-C. Yeh, Classification of multichannel surface-electromyography signals based on convolutional neural networks, J. Ind. Inf. Integr. 15 (2019) 201–206.

[15] H. Arbabi, T. Sapsis, Generative stochastic modeling of strongly nonlinear flows with non-Gaussian statistics, 2019, arXiv. https://doi.org/10.48550/ARXIV.1908.08941.

[16] D. Nguyen, S. Ouala, L. Drumetz, R. Fablet, EM-like learning chaotic dynamics from noisy and partial observations, 2019, arXiv. https://doi.org/10.48550/ARXIV.1903.10335.

[17] J. Jeon, P. Kim, B. Jang, Y. Kim, PDE-guided reservoir computing for image denoising with small data, Chaos 31 (7) (2021) 073103.

[18] H. Jaeger, H. Haas, Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication, Science 304 (5667) (2004) 78–80.

[19] T.-F. Weng, X.-X. Cao, H.-J. Yang, Complex network perspective on modelling chaotic systems via machine learning, Chin. Phys. B 30 (6) (2021) 060506.

[20] R.S. Zimmermann, U. Parlitz, Observing spatio-temporal dynamics of excitable media using reservoir computing, Chaos 28 (4) (2018) 043118.

[21] J. Pathak, B. Hunt, M. Girvan, Z. Lu, E. Ott, Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach, Phys. Rev. Lett. 120 (2018) 024102.

[22] B. Penkovsky, X. Porte, M. Jacquot, L. Larger, D. Brunner, Coupled nonlinear delay systems as deep convolutional neural networks, Phys. Rev. Lett. 123 (2019) 054101.

[23] L. Appeltant, M.C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C.R. Mirasso, I. Fischer, Information processing using a single dynamical node as complex system, Nature Commun. 2 (2011) 468.

[24] D.J. Gauthier, E. Bollt, A. Griffith, W.A.S. Barbosa, Next generation reservoir computing, Nature Commun. 12 (2021) 5564.

[25] Y.-C. Bo, P. Wang, X. Zhang, An asynchronously deep reservoir computing for predicting chaotic time series, Appl. Soft Comput. 95 (2020) 106530.

[26] M.U. Kobayashi, K. Nakai, Y. Saiki, N. Tsutsumi, Dynamical system analysis of a data-driven model constructed by reservoir computing, Phys. Rev. E 104 (2021) 044215.

[27] C.-N. Wang, J. Ma, Y. Liu, L. Huang, Chaos control, spiral wave formation, and the emergence of spatiotemporal chaos in networked Chua circuits, Nonlinear Dynam. 67 (2012) 139–146.

[28] R. Wang, Y. Zhang, Y. Chen, X. Chen, L. Xi, Fuzzy neural network-based chaos synchronization for a class of fractional-order chaotic systems: an adaptive sliding mode control approach, Nonlinear Dynam. 100 (2020) 1275–1287.

[29] E. Lorenz, Deterministic nonperiodic flow, J. Atmos. Sci. 20 (1963) 130–141.

[30] L.F. Shampine, M.W. Reichelt, The MATLAB ODE suite, SIAM J. Sci. Comput. 18 (1) (1997) 1–22.

[31] S.L. Brunton, J.L. Proctor, J.N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, Proc. Natl. Acad. Sci. 113 (15) (2016) 3932–3937.