scientific reports

Check for updates

OPEN

Controlling nonlinear dynamical systems into arbitrary states using machine learning

Alexander Haluszczynski[1,2]✉ & Christoph Räth[3]

Controlling nonlinear dynamical systems is a central task in many different areas of science and engineering. Chaotic systems can be stabilized (or chaotified) with small perturbations, yet existing approaches either require knowledge about the underlying system equations or large data sets as they rely on phase space methods. In this work we propose a novel and fully data driven scheme relying on machine learning (ML), which generalizes control techniques of chaotic systems without requiring a mathematical model for its dynamics. Exploiting recently developed ML-based prediction capabilities, we demonstrate that nonlinear systems can be forced to stay in arbitrary dynamical target states coming from any initial state. We outline and validate our approach using the examples of the Lorenz and the Rössler system and show how these systems can very accurately be brought not only to periodic, but even to intermittent and different chaotic behavior. Having this highly flexible control scheme with little demands on the amount of required data on hand, we briefly discuss possible applications ranging from engineering to medicine.

The possibility to control nonlinear chaotic systems into stable states has been a remarkable discovery[1,2]. Based on the knowledge of the underlying equations, one can force the system from a chaotic state into a fixed point or periodic orbit by applying an external force. This can be achieved based on the pioneering approaches by Ott et al.[1] or Pyragas[3]. In the former, a parameter of the system is slightly changed when it is close to an unstable periodic orbit in phase space, while the latter continuously applies a force based on time delayed feedback. There have been many extensions of those basic approaches (see e.g. Boccaletti et al.[4] and references therein) including "anti-control" schemes[5], that break up periodic or synchronized motion. However, all of them do not allow to control the system into well-specified, yet more complex target states such as chaotic or intermittent behavior. Further, these methods either require exact knowledge about the system, i.e. the underlying equations of motion, or rely on phase space techniques for which very long time series are necessary.

In recent years, tremendous progress has been made in the prediction of nonlinear dynamical systems by means of machine learning (ML). It has been demonstrated that not only exact short-term predictions over several Lyapunov times become possible, but also the long-term behavior of the system (its "climate") can be reproduced with unexpected accuracy[6–12]—even for very high-dimensional systems[13–15]. While several ML techniques have successfully been applied to time series prediction, reservoir computing (RC)[16,17] can be considered as the so far best approach, as it combines often superior performance with intrinsic advantages like smaller network size, higher robustness, fast and comparably transparent learning[18] and the prospect of highly efficient hardware realizations[19–21].

Combining now ML-based predictions of nonlinear systems with manipulation steps, we propose in this study a novel, fully data-driven approach for controlling nonlinear dynamical systems. In contrast to previous methods, this allows to obtain a variety of target states including periodic, intermittent and chaotic ones. Furthermore, we do not require the knowledge of the underlying equations. Instead, it is sufficient to record some history of the system that allows the machine learning method to be sufficiently trained. As previously outlined[22], an adequate learning requires orders of magnitude less data than phase space methods.

[1]Department of Physics, Ludwig-Maximilians-Universität, Schellingstraße 4, 80799 Munich, Germany. [2]Allianz Global Investors, risklab, Seidlstraße 24, 80335 Munich, Germany. [3]Institut für Materialphysik im Weltraum, Deutsches Zentrum für Luft- und Raumfahrt, Münchner Str. 20, 82234 Wessling, Germany. ✉email: alexander.haluszczynski@gmail.com

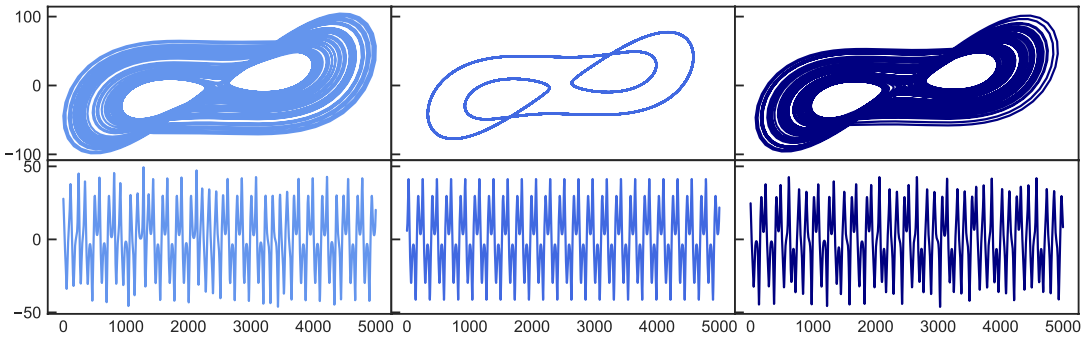

Figure 1. Periodic to chaotic control. Top: 2D attractor representation in the x–y plane. Bottom: X coordinate time series. Left plots show the original chaotic state which changes to a periodic state (middle) after tuning the order parameter. After applying the control mechanism, the system is forced into a chaotic state again (right).

Results

We define the situation that requires to be controlled in the following way: A dynamical system with trajectory $\mathbf{u}$ is in state $\mathbf{X}$, which may represent e.g. periodic, intermittent or chaotic behavior. Then, the system behavior changes into another state $\mathbf{Y}$ as a consequence of order parameter changes or some uncontrollable external force. The aim of a control mechanism is now to push the system back into its original state $\mathbf{X}$, while the cause for the initial change in state is still present. This can be achieved by deriving a suitable control force $\mathbf{F}(t)$ which is applied while the system is in state $\mathbf{Y}$. Deriving $\mathbf{F}(t)$ requires the knowledge of how the trajectory $\mathbf{u}(t)$ of the system would have evolved if the system was still in state $\mathbf{X}$ instead. This 'what if' scenario can be obtained by training a suitable machine learning technique on past observations of the system while being in state $\mathbf{X}$. In this study, this is achieved by using reservoir computing[23], which is a recurrent neural network based approach. In principle, any other prediction method could be used instead as long as it is able to deliver good predictions. Once trained and synchronized, it can create predictions $\mathbf{v}(t)$ of arbitrary length from which the control force $\mathbf{F}(t)$ is derived as

$$\mathbf{F}(t) = K(\mathbf{u}(t) - \mathbf{v}(t)), \tag{1}$$

where K scales the magnitude of the force. Since $\mathbf{F}$(t) only depends on the (measured) coordinates $\mathbf{u}$(t) and the ML prediction $\mathbf{v}$(t), no mathematical model is required to control the system and thus the method is generally applicable as long as good predictions are available. The definition of the control force being dependent on the distance between the actual coordinate and a target coordinate is similar to what has been originally proposed by Pyragas[3]. However, in our case the control is not limited to periodic orbits but can achieve a variety of dynamical target states. A step by step description of the method is given in Section 0.2. The control of nonlinear dynamical system is studied on the example of the Lorenz system[24], which is a model for atmospheric convection. Depending on the choice of parameters, the system exhibits e.g. periodic, intermittent or chaotic behavior. The equations read

$$\dot{x} = \sigma(y - x); \quad \dot{y} = x(\rho - z) - y; \quad \dot{z} = xy - \beta z, \tag{2}$$

and $\boldsymbol{\pi} \equiv (\sigma, \rho, \beta)$ are the order parameters that lead to a certain state and the trajectory is thus described by $\mathbf{u}(t) = (x(t), y(t), z(t))^T$. First, we simulate the Lorenz system with parameters $\boldsymbol{\pi}$ such that we obtain the desired initial state $\mathbf{X}$. Second, we train reservoir computing on the resulting trajectory until time step t_{train}. Then, the parameters are shifted to $\boldsymbol{\pi}^*$ such that the system behavior changes to state $\mathbf{Y}$ at time step t_{shift}. If $t_{shift} \geq t_{train}$, the RC system is synchronized accordingly with the trajectory since t_{train}. Synchronization means that the scalar states of the reservoir (see Eq. 5) are updated but the system is not re-trained. To control the system now back into state $\mathbf{X}$, the correction force $\mathbf{F}(t)$ is derived in each time step based on the prediction $\mathbf{v}(t)$ and applied to the system by solving the differential equations of the system for the next time step including $\mathbf{F}(t)$

$$\mathbf{u}(t + \Delta t) = \int_t^{t+\Delta t} (\dot{f}(\mathbf{u}(\tilde{t}), \boldsymbol{\pi}^*) + \mathbf{F}(\tilde{t}))d\tilde{t}, \tag{3}$$

where $\dot{f}$ is defined in Eq. (2). The knowledge of $\dot{f}$ is only required for the model system examples in this study but not for real world applications. The equations are solved using the 4th order Runge–Kutta method with a time resolution $\Delta t = 0.02$. Since still the parameters $\boldsymbol{\pi}^*$ are used, the system would continue to exhibit the undesired state $\mathbf{Y}$ if the control force was 0. For the Lorenz system, the scaling constant set to $K = 25$. We did not optimize for K and empirically found that our method works for a wide range of choices. It is important to emphasize that a smaller choice for K does not necessarily mean that a smaller force is needed, because smaller values may allow for more separation of $\mathbf{u}$(t) and $\mathbf{v}$(t).

Figure 1 shows the results for the Lorenz system originally (left side) being in a chaotic state $\mathbf{X}$ ($\boldsymbol{\pi} = [\sigma = 10.0, \rho = 167.2, \beta = 8/3]$), which then changes to periodic behavior (middle) $\mathbf{Y}$ after ρ is changed to $\rho = 166$. Then, the control mechanism is activated and the resulting attractor again resembles the original chaotic state (left). While 'chaotification' of periodic states has been achieved in the past, the resulting attractor generally did not correspond to a certain specified target state but just exhibited some chaotic behavior. Since we would like to not only rely on a visual assessment, we characterize the attractors using quantitative

	Largest Lyapunov exponent λ			Correlation dimension ν		
	λ_{orig}	$\lambda_{changed}$	$\lambda_{controlled}$	ν_{orig}	$\nu_{changed}$	$\nu_{controlled}$
Periodic → *Chaotic*	0.851 ± 0.070	0.080 ± 0.075	0.841 ± 0.074	1.700 ± 0.065	1.052 ± 0.071	1.700 ± 0.061
Chaotic → *Intermittent*	0.571 ± 0.096	0.853 ± 0.053	0.614 ± 0.101	1.321 ± 0.086	1.678 ± 0.055	1.351 ± 0.091
$Chaotic_B \rightarrow Chaotic_A$	0.479 ± 0.060	0.643 ± 0.075	0.478 ± 0.067	1.941 ± 0.038	1.948 ± 0.047	1.933 ± 0.040
$Chaotic_D \rightarrow Chaotic_C$	0.819 ± 0.092	0.884 ± 0.058	0.822 ± 0.052	1.855 ± 0.069	1.959 ± 0.037	1.866 ± 0.050
Periodic ← *Chaotic*	- 0.003 ± 0.012	0.844 ± 0.059	0.028 ± 0.110	1.001 ± 0.065	1.700 ± 0.071	1.001 ± 0.061
Chaotic ← *Intermittent*	0.851 ± 0.070	0.550 ± 0.094	0.828 ± 0.067	1.700 ± 0.086	1.326 ± 0.055	1.698 ± 0.091
$Chaotic_B \leftarrow Chaotic_A$	0.629 ± 0.069	0.446 ± 0.068	0.629 ± 0.066	1.948 ± 0.037	1.939 ± 0.049	1.956 ± 0.037
$Chaotic_D \leftarrow Chaotic_C$	0.881 ± 0.092	0.836 ± 0.058	0.880 ± 0.052	1.958 ± 0.069	1.864 ± 0.038	1.951 ± 0.050

Table 1. Statistical simulation over $N = 100$ random realizations of the systems evaluated in terms of the mean values of the largest Lyapunov exponent and the correlation dimension with corresponding standard deviations. The subscript *orig* denotes the initial state of the system, while *changed* refers to the new state after parameters changed and *controlled* means the system controlled back into the original state. The description left to the arrow is the original state that also will be achieved again after controlling the system whereas the state written right to the arrow corresponds to the changed condition.

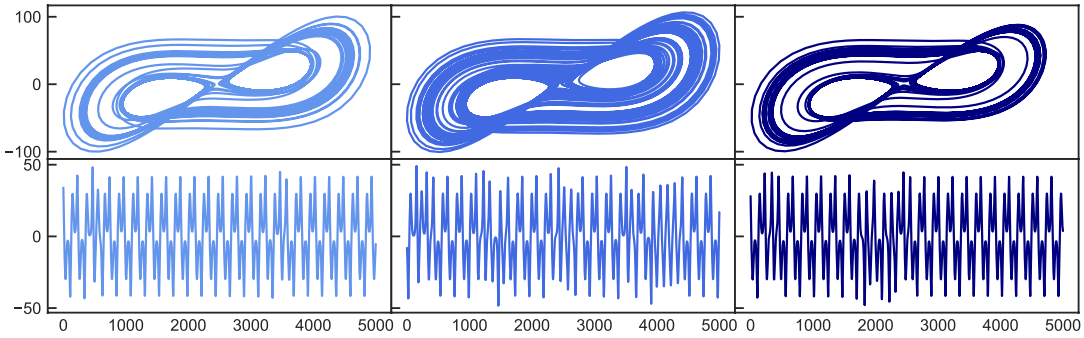

Figure 2. Chaotic to intermittent control. Top: 2D attractor representation in the x–y plane. Bottom: X coordinate time series. Left plots show the original intermittent state which changes to a chaotic state (middle) after tuning the order parameter. After applying the control mechanism, the system is forced into an intermittent state again (right).

measures. First, we calculate the largest Lyapunov exponent, which quantifies the temporal complexity of the trajectory, where a positive value indicates chaotic behavior. Second, we use the correlation dimension to assess the structural complexity of the attractor. Based on the two measures, the dynamical state of the system can be sufficiently specified for our analysis. Both techniques are described in the supporting information. Because a single example is not sufficiently meaningful, we perform our analysis statistically by evaluating 100 random realizations of the system at a time. The term 'random realization' refers to different random drawings of the reservoir $\mathbf{A}$ and the input mapping $\mathbf{W}_{in}$, as well as the initial conditions for the Lorenz system. The first line in Table 1 shows the respective statistical results for the setup shown in Figure 1. The largest Lyapunov exponent of the original chaotic system $\lambda_{orig} = 0.851$ significantly reduces to $\lambda_{changed} = 0.080$ when the parameter change drives the system into a periodic state. After the control mechanism is switched on, the value for the resulting attractor moves back to $\lambda_{controlled} = 0.0841$ and thus is within one standard deviation from its original value. Same applies to the correlation dimension, which resembles its original value after control very well.

Since there is a clear distinction between the chaotic- and the periodic state, with the latter being simple in terms of its dynamics, the next step is to control the system between more complex dynamics. Therefore, we start simulate the Lorenz system again with parameters $\boldsymbol{\pi} = [\sigma = 10.0, \rho = 166.15, \beta = 8/3]$ that lead to intermittent behavior[25]. This is shown in Fig. 2 on the left. Now ρ is changed to $\rho = 167.2$, which results in a chaotic state (middle plots). The control mechanism is turned on and the resulting state shows again the intermittent behavior (right plots) as in the initial state. This is particularly visible in the lower plots where only the X coordinate is shown. While the trajectory mostly follows a periodic path, it is interrupted by irregular burst that occur from time to time. It is remarkable that bursts do not seem to occur more often given the chaotic dynamics of the underlying equations and parameter setup. However, the control works so well that it exactly enforces the desired dynamics. This observation can again be confirmed by looking at the statistical results in Table 1.

Just like in the first two examples, it was not possible before to control a system from one chaotic state to another particular chaotic state. To do this, we start with the parameter set ($\boldsymbol{\pi} = [\sigma = 10.0, \rho = 28.0, \beta = 8/3]$) leading to a chaotic attractor which we call $Chaotic_A$. When changing ρ to $rho = 50.0$ we obtain a different chaotic attractor $Chaotic_B$. This time we use a different range of values for ρ compared to the previous examples in order to present a situation where not only the chaotic dynamics change, but also the size of the attractor significantly

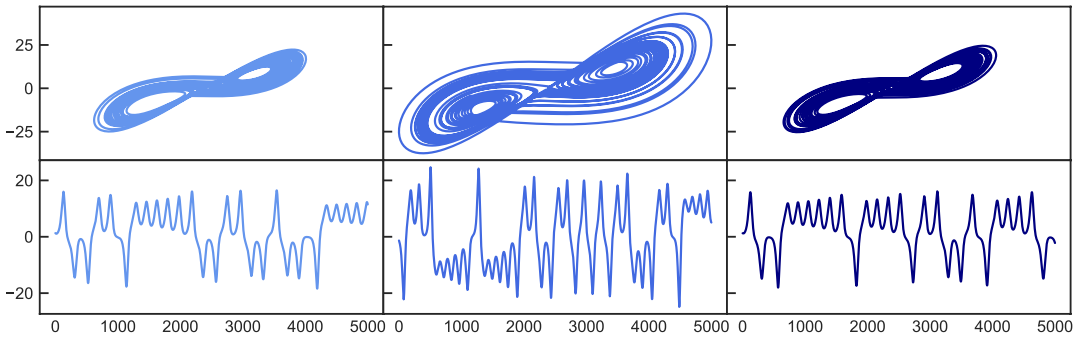

Figure 3. Chaotic to chaotic control. Top: 2D attractor representation in the x–y plane. Bottom: X coordinate time series. Left plots show the original chaotic state which changes to a different chaotic state (middle) after tuning the order parameter. After applying the control mechanism, the system is forced into the initial chaotic state again (right).

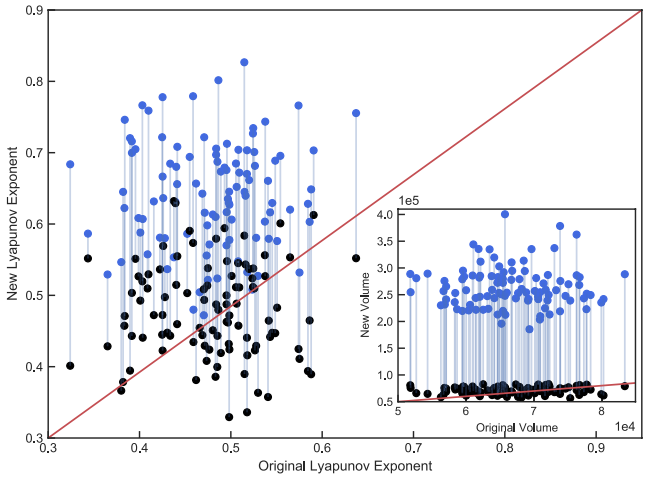

Figure 4. Chaotic to chaotic control (ρ changed). Values on the x-axis denote the largest Lyapunov exponent λ_{max} of the original system state before parameter change for $N = 100$ random realizations. Y-axis reflects the values for λ_{max} after parameters changed from $\rho = 28$ to $\rho = 50$. The blue dots correspond to the uncontrolled systems, while the black dots represent the controlled systems. Inlay plot shows the same for the volume of the attractor.

varies between the two states. The goal of the control procedure now is to not only force the dynamics of the system back to the behavior of the initial state $Chaotic_A$, but also to return the attractor to its original size. Figure 3 shows that both goals succeed. This is also confirmed by the statistical results, indicating that the largest Lyapunov exponent of the controlled system is perfectly close to the one of the uncontrolled original state. For the correlation dimension, however, there are no significant deviations between the two chaotic states. To give a more striking illustration of the statistical analysis, we show the results for each of the 100 random realizations in Fig. 4. The main plot scatters the largest Lyapunov exponents as measured for the original parameter set $\boldsymbol{\pi}$ against those measured after the parameters have been changed to $\boldsymbol{\pi}^*$. While the blue dots represent the situation where the control mechanism is not active, the control has been switched on for the black dots. Furthermore, each pair of points is connected with a line that belongs to the same random realization. It is clearly visible that the control leads to a downwards shift of the cloud of points towards the diagonal, which is consistent to the respective average values of the largest Lyapunov exponent shown in Table 1. In addition, the inlay plot shows the same logic but for the volume of the attractors being measured in terms of the smallest cuboid that covers the attractor. The control mechanism consistently works for every single realization and reduces the volume of the attractor back towards the initially desired state. We successfully applied our approach to other examples of controlling a chaotic state to another chaotic state, e.g. by varying the parameter σ as shown in the supporting information.

The bottom half of Table 1 proves that our statements are also valid if one reverses the direction in the examples. For example, *Periodic* → *Chaotic* in the upper half of the table means, that an initially chaotic system changed into a periodic state and then gets controlled back into its initial chaotic state. In contrast, *Periodic* ← *Chaotic* in the lower half now means that the system initially is in the periodic state. It then shows chaotic behavior after the parameter change and finally is controlled back into the original periodic state—thus the opposite direction as above. It is evident that all examples also succeed in the opposite direction. This supports our claim that the prediction based control mechanism works for arbitrary states.

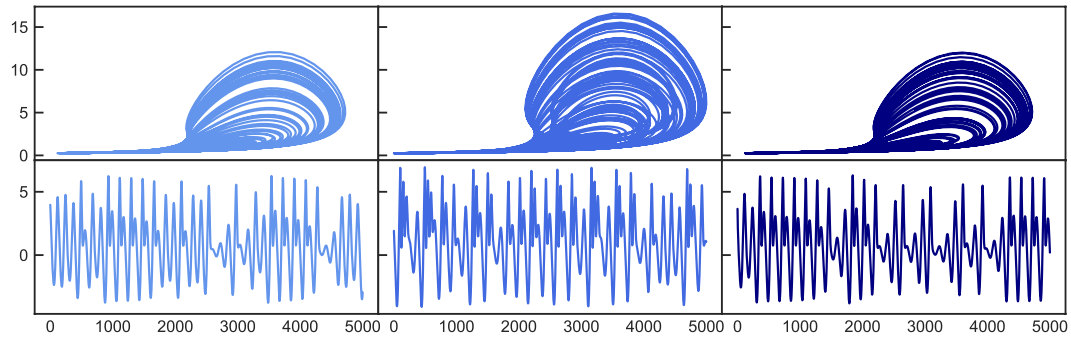

Figure 5. Chaotic to chaotic control for the Roessler system. Top: 2D attractor representation in the x–z plane. Bottom: X coordinate time series. Left plots show the original chaotic state which changes to a different chaotic state (middle) after tuning the order parameter. After applying the control mechanism, the system is forced into the initial chaotic state again (right).

In addition to the Lorenz system we also applied the method to another popular chaotic attractor: the Roessler system[26]. The equations read

$$\dot{x} = -(y+z); \quad \dot{y} = x + ay; \quad \dot{z} = b + (x-c)z \tag{4}$$

and we use parameters $\boldsymbol{\pi} = [a = 0.5, b = 2.0, c = 4.0]$ leading to a chaotic behavior. This serves as our initial state and the dynamics change to another chaotic state after the parameters are changed to $\boldsymbol{\pi}^* = [a = 0.55, b = 2.0, c = 4.0]$. For the Roessler system, we use a time resolution of $\Delta t = 0.05$ and $K = 20$. It can be seen in Fig. 5 that the control mechanism is successful. Again, the left plots represent the initial attractor resulting from the parameter set $\boldsymbol{\pi}$. Switching to $\boldsymbol{\pi}^*$ (middle plots) not only increases the size of the attractor in the x–z plane, but also significantly changes the pattern of the x-coordinate time series. Both, the appearance of the attractor and its x-coordinate pattern become similar to the initial attractor again after the control mechanism is active (right plots). The initial state with parameters $\boldsymbol{\pi}$ has properties $[\lambda_{max} = 0.13, \nu = 1.59]$, which become $[\lambda_{max} = 0.14, \nu = 1.75]$ after parameters have been changed to $\boldsymbol{\pi}^*$. Turning on the control mechanism leads to $[\lambda_{max} = 0.12, \nu = 1.64]$.

Discussion

Our method has a wide range of potential applications in various areas. For example, in nonlinear technical systems such as rocket engines it can be used to prevent the engine from critical combustion instabilities[27,28]. This could be achieved by detecting them based on the reservoir computing predictions (or any other suitable ML technique) and subsequently controlling the system into a more stable state. Here, the control force can be applied to the engine via its pressure valves. Another example would be medical devices such as pacemakers. The heart of a healthy human does not beat in a purely periodic fashion but rather shows features being typical for chaotic systems like multifractality[29] that vary significantly among individuals. While pacing protocols developed so far aim at keeping the diastolic interval constant[30], our general control scheme will emulate the patient-specific full behavior of the heart in healthy conditions. The control scheme could therefore be used to develop personalized pacemakers that do not just stabilize the heartbeat to periodic behavior[31–33], but may rather adjust the heartbeat to the individual needs of the patients.

In conclusion, our machine learning enhanced method allows for an unprecedented flexible control of dynamical systems and has thus the potential to extend the range of applications of chaos inspired control schemes to a plethora of new real-world problems.

Methods

Reservoir computing. RC or echo state networks[17,34,35] is an artificial recurrent neural network based approach, which builds on a static internal network called *reservoir* $\mathbf{A}$. Static means that the nodes and edges are kept fixed once the network has been initially created. This property makes RC computationally very efficient, as only its linear output layer is being optimized in the training process. The reservoir $\mathbf{A}$ is constructed as a sparse Erdös–Renyi random network[36] with $D_r = 300$ nodes that are connected with a probability $p = 0.02$. In order to feed the $D = 3$ dimensional input data $\mathbf{u}(t)$ into the reservoir $\mathbf{A}$, we set up an $D_r \times D$ input mapping matrix $\mathbf{W}_{in}$, which defines how strongly each input dimension influences every single node. The dynamics of the network are represented by its $D_r \times 1$ dimensional scalar states $\mathbf{r}(t)$ evolving according to the recurrent equation

$$\mathbf{r}(t + \Delta t) = tanh(\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{in}\mathbf{u}(t)). \tag{5}$$

Output $\mathbf{v}(t + \Delta t)$ is created by mapping back $\mathbf{r}(t)$ using a linear output function $\mathbf{W}_{out}$ such that

$$\mathbf{v}(t) = \mathbf{W}_{out}(\tilde{\mathbf{r}}(t), \mathbf{P}) = \mathbf{P}\tilde{\mathbf{r}}(t), \tag{6}$$

where $\tilde{\mathbf{r}} = \{\mathbf{r}, \mathbf{r}^2\}$. The matrix $\mathbf{P}$ is determined in the training process. This is done by acquiring a sufficient number of reservoir states $\mathbf{r}(t_w \ldots t_w + t_T)$ and then choosing $\mathbf{P}$ such that the output $\mathbf{v}$ of the reservoir is as close as possible to the known real data $\mathbf{v}(t_w \ldots t_w + t_T)$. For this we use Ridge regression, which minimizes

$$\sum_{-T \le t \le 0} \parallel \mathbf{W}_{out}(\tilde{\mathbf{r}}(t), \mathbf{P}) - \mathbf{v}_R(t) \parallel^2 - \beta \parallel \mathbf{P} \parallel^2, \tag{7}$$

where β is the regularization constant that prevents from overfitting by penalizing large values of the fitting parameters. The training process only involves the linear output layer and therefore is fast compared to other ML methods. Replacing $\mathbf{u}(t)$ in the *tanh* activation function above by $\mathbf{P}\tilde{\mathbf{r}}(t)$ allows to create predictions of arbitrary length due to the recursive equation for the reservoir states $\mathbf{r}(t)$:

$$\begin{aligned} \mathbf{r}(t + \Delta t) &= tanh(\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{in}\mathbf{W}_{out}(\tilde{\mathbf{r}}(t), \mathbf{P})) \\ &= tanh(\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{in}\mathbf{P}\tilde{\mathbf{r}}(t)). \end{aligned} \tag{8}$$

Further details including the choices for the hyperparameters are presented in the supporting information. We use a washout phase of 1000 time steps, a training period of 5000 time steps and let the parameter change of the dynamical system from $\boldsymbol{\pi}$ to $\boldsymbol{\pi}^*$ happen immediately after the training period and thus the prediction is needed from this moment on. However, it is not necessary that the network is trained on the full history until the parameter change happened. In general, it needs to be sufficiently trained and can then be synchronized based on the recorded trajectory after the training ended. The prediction is carried out for 10,000 time steps.

It has been shown by Bompas et al.[18] that the performance of reservoir computing does not strongly depend on the precision of the data. Hence, measurement noise and sensitive dependence on initial conditions for chaotic systems is not a problem when it comes to real world applications of the proposed method.

Control mechanism. The concrete steps of the application of the control mechanism to the examples in our study are shown in Algorithm 1. This is the simplest setup possible, where only one long prediction for $\mathbf{v}$(t) is performed before the control force is activated. We also successfully tested multiple more complicated setups, e.g. where the control force is not immediately switched on and the system is running on the new parameters $\boldsymbol{\pi}^*$ (and thus state $\mathbf{Y}$) for a while, where the reservoir computing prediction is updated after synchronizing the RC model with the realized trajectory since the last training or where the force is not applied in every time step. The control phase is run for 10,000 time steps.

These steps also apply for real world systems, where no mathematical model is available. The only requirement is sufficient data of the system recorded while being in the desired dynamical state $\mathbf{X}$.

Algorithm 1: Control mechanism

1 Simulate the system with initial parameters $\boldsymbol{\pi}$ such that the system is in a certain dynamical state $\mathbf{X}$
2 Train reservoir computing onto this data
3 Change the parameters to $\boldsymbol{\pi}^*$ such that the system is now in dynamical state $\mathbf{Y}$ with coordinates $\mathbf{u}(t_0)$
4 Predict how $\mathbf{u}(t_0)$ would evolve if the system was still in state $\mathbf{X}$ using RC from *Step 2* - this yields $\mathbf{v}$(t)
5 For $t > t_0$
6 Measure $\mathbf{u}$(t) and determine $\mathbf{v}$(t)
7 Compute the control force $\mathbf{F(t)}$,

$$\mathbf{F(t)} = K(\mathbf{u}(t) - \mathbf{v}(t))$$

8 Apply the control force when simulating

$$\mathbf{u}(t + \Delta t) = \int_t^{t+\Delta t} (\dot{f}(\mathbf{u}(\tilde{t}), \boldsymbol{\pi}^*) + \mathbf{F}(\tilde{t}))d\tilde{t} \ ,$$

9 Repeat *Steps 6 to 8* in every time step

Correlation dimension. To characterize the attractor and therefore its dynamical state we rely on quantitative measures. For this, we are looking at the long-term properties of the attractor rather than its short-term trajectory. One important aspect of the long-term behavior is the structural complexity. This can be assessed by calculating the correlation dimension of the attractor, where we measure the dimensionality of the space populated by the trajectory[37]. The correlation dimension is based on the correlation integral

$$C(r) = \lim_{N\to\infty} \frac{1}{N^2} \sum_{i,j=1}^{N} \theta(r - |\mathbf{x}_i - \mathbf{x}_j|) = \int_0^r d^3r' c(\mathbf{r}'), \tag{9}$$

where θ is the Heaviside function and $c(\mathbf{r}')$ denotes the standard correlation function. The correlation integral represents the mean probability that two states in phase space are close to each other at different time steps. This is the case if the distance between the two states is less than the threshold distance r. The correlation dimension ν is then defined by the power-law relationship

$$C(r) \propto r^{\nu}. \tag{10}$$

For self-similar strange attractors, this relationship holds for a certain range of r, which therefore needs to be properly calibrated. As we are finally only interested in comparisons, precision with regards to absolute values is not essential here. We use the Grassberger Procaccia algorithm[38] to calculate the correlation dimension.

Lypunov exponents. The temporal complexity of a system can be measured by its Lyapunov exponents λ_i, which describe the average rate of divergence of nearby points in phase space, and thus measure sensitivity to initial conditions. There is one exponent for each dimension in phase space. If the system exhibits at least one positive Lyapunov exponent, it is classified as chaotic. The magnitudes of λ_i quantify the time scale on which the system becomes unpredictable[39,40]. Since at least one positive exponent is the requirement for being classified as chaotic, it is sufficient for our analysis to calculate only the largest Lyapunov exponent λ_{max}

$$d(t) = Ce^{\lambda_{max}t}. \tag{11}$$

This makes the task computationally much easier than determining the full Lyapunov spectrum. We use the Rosenstein algorithm[41] to obtain it. In essence, we track the distance $d(t)$ of two initially nearby states in phase space. The constant C normalizes the initial separation. As for the correlation dimension, we are interested in a relative comparison that characterizes states of the system rather than the exact absolute values. It is important to point out that both measures—the correlation dimension and the largest Lyapunov exponent—are calculated purely based on data and do not require any knowledge of the underlying equations.

Data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Received: 4 April 2021; Accepted: 7 June 2021
Published online: 21 June 2021

References

1. Ott, E., Grebogi, C. & Yorke, J. A. Controlling chaos. *Phys. Rev. Lett.* **64**, 1196 (1990).
2. Shinbrot, T., Grebogi, C., Yorke, J. A. & Ott, E. Using small perturbations to control chaos. *Nature* **363**, 411–417 (1993).
3. Pyragas, K. Continuous control of chaos by self-controlling feedback. *Phys. Lett. A* **170**, 421–428 (1992).
4. Boccaletti, S., Grebogi, C., Lai, Y.-C., Mancini, H. & Maza, D. The control of chaos: Theory and applications. *Phys. Rep.* **329**, 103–197 (2000).
5. Schiff, S. J. *et al.* Controlling chaos in the brain. *Nature* **370**, 615–620 (1994).
6. Chattopadhyay, A., Hassanzadeh, P. & Subramanian, D. Data-driven predictions of a multiscale lorenz 96 chaotic system using machine-learning methods: Reservoir computing, artificial neural network, and long short-term memory network. *Nonlinear Process. Geophys.* **27**, 373–389 (2020).
7. Vlachas, P. R. *et al.* Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics. *Neural Netw.* **126**, 191–217 (2020).
8. Sangiorgio, M. & Dercole, F. Robustness of LSTM neural networks for multi-step forecasting of chaotic time series. *Chaos Solitons Fractals* **139**, 110045 (2020).
9. Herteux, J. & Räth, C. Breaking symmetries of the reservoir equations in echo state networks. *Chaos Interdiscip. J. Nonlinear Sci.* **30**, 123142 (2020).
10. Haluszczynski, A. & Räth, C. Good and bad predictions: Assessing and improving the replication of chaotic attractors by means of reservoir computing. *Chaos Interdiscip. J. Nonlinear Sci.* **29**, 103143 (2019).
11. Griffith, A., Pomerance, A. & Gauthier, D. J. Forecasting chaotic systems with very low connectivity reservoir computers. *Chaos Interdiscip. J. Nonlinear Sci.* **29**, 123108 (2019).
12. Lu, Z., Hunt, B. R. & Ott, E. Attractor reconstruction by machine learning. *Chaos Interdiscip. J. Nonlinear Sci.* **28**, 061104 (2018).
13. Pathak, J., Hunt, B., Girvan, M., Lu, Z. & Ott, E. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Phys. Rev. Lett.* **120**, 024102 (2018).
14. Zimmermann, R. S. & Parlitz, U. Observing spatio-temporal dynamics of excitable media using reservoir computing. *Chaos Interdiscip. J. Nonlinear Sci.* **28**, 043118 (2018).
15. Baur, S. & Räth, C. Predicting high-dimensional heterogeneous time series employing generalized local states (2021). arXiv:2102.12333.
16. Maass, W., Natschläger, T. & Markram, H. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Comput.* **14**, 2531–2560 (2002).
17. Jaeger, H. & Haas, H. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science* **304**, 78–80 (2004).
18. Bompas, S., Georgeot, B. & Guéry-Odelin, D. Accuracy of neural networks for the simulation of chaotic dynamics: Precision of training data vs precision of the algorithm. *Chaos Interdiscip. J. Nonlinear Sci.* **30**, 113118 (2020).

19. Marcucci, G., Pierangeli, D. & Conti, C. Theory of neuromorphic computing by waves: Machine learning by rogue waves, dispersive shocks, and solitons. *Phys. Rev. Lett.* **125**, 093901 (2020).
20. Tanaka, G. *et al.* Recent advances in physical reservoir computing: A review. *Neural Netw.* **115**, 100–123 (2019).
21. Carroll, T. L. Adding filters to improve reservoir computer performance. *Physica D* **416**, 132798 (2021).
22. Pathak, J., Lu, Z., Hunt, B. R., Girvan, M. & Ott, E. Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data. *Chaos Interdiscip. J. Nonlinear Sci.* **27**, 121102 (2017).
23. Lukoševičius, M. & Jaeger, H. Reservoir computing approaches to recurrent neural network training. *Comput. Sci. Rev.* **3**, 127–149 (2009).
24. Lorenz, E. N. Deterministic nonperiodic flow. *J. Atmos. Sci.* **20**, 130–141 (1963).
25. Pomeau, Y. & Manneville, P. Intermittent transition to turbulence in dissipative dynamical systems. *Commun. Math. Phys.* **74**, 189–197 (1980).
26. Rössler, O. E. An equation for continuous chaos. *Phys. Lett. A* **57**, 397–398 (1976).
27. Kabiraj, L., Saurabh, A., Wahi, P. & Sujith, R. Route to chaos for combustion instability in ducted laminar premixed flames. *Chaos Interdiscip. J. Nonlinear Sci.* **22**, 023129 (2012).
28. Nair, V., Thampi, G. & Sujith, R. Intermittency route to thermoacoustic instability in turbulent combustors. *J. Fluid Mech.* **756**, 470 (2014).
29. Ivanov, P. C. *et al.* Multifractality in human heartbeat dynamics. *Nature* **399**, 461–465 (1999).
30. Kulkarni, K., Walton, R. D., Armoundas, A. A. & Tolkacheva, E. G. Clinical potential of beat-to-beat diastolic interval control in preventing cardiac arrhythmias. *J. Am. Heart Assoc.* e020750 (2021).
31. Garfinkel, A., Spano, M. L., Ditto, W. L. & Weiss, J. N. Controlling cardiac chaos. *Science* **257**, 1230–1235 (1992).
32. Hall, K. *et al.* Dynamic control of cardiac alternans. *Phys. Rev. Lett.* **78**, 4518 (1997).
33. Christini, D. J. *et al.* Nonlinear-dynamical arrhythmia control in humans. *Proc. Natl. Acad. Sci.* **98**, 5827–5832 (2001).
34. Jaeger, H. The, "echo state" approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report* **148**, 13 (2001).
35. Maass, W., Natschlaeger, T. & Markram, H. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Comput.* **14**, 2531–2560. https://doi.org/10.1162/089976602760407955 (2002).
36. Erdos, P. On random graphs. *Publicationes mathematicae* **6**, 290–297 (1959).
37. Grassberger, P. & Procaccia, I. Measuring the strangeness of strange attractors. In *The Theory of Chaotic Attractors*, 170–189 (Springer, 2004).
38. Grassberger, P. Generalized dimensions of strange attractors. *Phys. Lett. A* **97**, 227–230 (1983).
39. Wolf, A., Swift, J. B., Swinney, H. L. & Vastano, J. A. Determining Lyapunov exponents from a time series. *Physica D* **16**, 285–317 (1985).
40. Shaw, R. Strange attractors, chaotic behavior, and information flow. *Zeitschrift für Naturforschung A* **36**, 80–112 (1981).
41. Rosenstein, M. T., Collins, J. J. & De Luca, C. J. A practical method for calculating largest Lyapunov exponents from small data sets. *Physica D* **65**, 117–134 (1993).

Acknowledgements

We would like to thank Y. Mabrouk, J. Herteux, S. Baur and J. Aumeier for helpful discussions.

Author contributions

C.R. initiated the research. A.H. and C.R. designed the study. A.H. performed the calculations. A.H. and C.R. interpreted the results and wrote the manuscript.

Funding

Open Access funding enabled and organized by Projekt DEAL.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-021-92244-6.

Correspondence and requests for materials should be addressed to A.H.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.