

<https://doi.org/10.1038/s42005-024-01885-2>

Suppressing unknown disturbances to dynamical systems using machine learning

Juan G. Restrepo¹, Clayton P. Byers² & Per Sebastian Skardal³✉

Identifying and suppressing unknown disturbances to dynamical systems is a problem with applications in many different fields. Here we present a model-free method to identify and suppress an unknown disturbance to an unknown system based only on previous observations of the system under the influence of a known forcing function. We find that, under very mild restrictions on the training function, our method is able to robustly identify and suppress a large class of unknown disturbances. We illustrate our scheme with the identification of both deterministic and stochastic unknown disturbances to an analog electric chaotic circuit and with numerical examples where a chaotic disturbance to various chaotic dynamical systems is identified and suppressed.

Identifying and suppressing an unknown disturbance to a dynamical system is a problem with many existing and potential applications in engineering^{1–9}, ecology^{10,11}, fluid mechanics^{12–14}, and climate change¹⁵. Traditional control theory disturbance identification and suppression techniques usually assume either an existing model for the dynamical system, linearity, or that the disturbance can be observed (for reviews of existing methods see, for example, refs. 4,16). In this Article we present a method for real-time disturbance identification and suppression that relies solely on observations of the dynamical system when forced with a known training forcing function. Our method is based on the application of machine-learning techniques to dynamical systems. Such techniques have found many applications, including the forecast of chaotic spatiotemporal¹⁷ and networked¹⁸ dynamics, estimation of dynamical invariants from data¹⁹, control of chaos²⁰, network structure inference²¹, and prediction of extreme events²² and crises in non-stationary dynamical systems^{23,24}. For a review of other applications and techniques, see refs. 25–27. In most of these previous works, a machine learning framework is trained to replicate the nonlinear dynamics of the system based on a sufficiently long time series of the dynamics.

In this Article we use machine learning to identify and subsequently suppress an unknown disturbance. Without knowledge of an underlying model for the dynamical system, and only based on observations of the system under a suitable known forcing function, our method allows us to reliably identify and suppress a large class of disturbances. Recent work²⁸ considers the problem of predicting the response of a system based on knowledge of the forcing (the disturbance) and the system's response after training with known functions. That problem can be thought of as the “forward” problem, while the problem addressed here can be considered as

the “inverse” problem. While both approaches are complementary, they apply to very different situations. In addition to this fundamental difference, the main additional differences between our results and those of ref. 28 are that we present a method to suppress the unknown forcing, that our method works for stochastic signals, and that we show that the training functions can be extremely simple (e.g., piecewise constant functions). We also demonstrate our method with an experimental analog chaotic circuit in addition to numerical simulations.

Results

System setup, disturbances, and reservoir computers

Consider an N -dimensional dynamical system

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}(\mathbf{x}) + \mathbf{g}(t), \quad (1)$$

where $\mathbf{x} \in \mathbb{R}^N$ is the state vector, $\mathbf{F} \in \mathbb{R}^N$ represents the intrinsic dynamics of the system, and $\mathbf{g}(t) \in \mathbb{R}^N$ represents an unknown (and usually undesired) disturbance. Our goal is to develop a scheme by which the disturbance can be identified and the system can be brought approximately to satisfy the undisturbed dynamics $d\mathbf{x}/dt = \mathbf{F}(\mathbf{x})$. We assume that we can observe the state vector \mathbf{x} , but we don't need to assume knowledge of the intrinsic dynamics \mathbf{F} or the disturbance function \mathbf{g} . Assuming that we can force the system with a known *training forcing function* $\mathbf{f}(t)$, as

$$\frac{d\tilde{\mathbf{x}}}{dt} = \mathbf{F}(\tilde{\mathbf{x}}) + \mathbf{f}(t), \quad (2)$$

¹Department of Applied Mathematics, University of Colorado at Boulder, Boulder, CO, 80309, USA. ²Department of Engineering, Trinity College, Hartford, CT, 06106, USA. ³Department of Mathematics, Trinity College, Hartford, CT, 06106, USA. ✉e-mail: persebastian.skardal@trincoll.edu

and observe $\hat{\mathbf{x}}(t)$ for a long enough time, our goal is to train a machine learning system to approximate $\mathbf{f}(t)$ given $\hat{\mathbf{x}}(t)$, and subsequently infer $\mathbf{g}(t)$ from observations of $\mathbf{x}(t)$ obtained from system (1). As we will show below, we find that we can recover a large class of forcing functions $\mathbf{g}(t)$ with very mild restrictions on the choice of training functions $\mathbf{f}(t)$. Once we infer $\mathbf{g}(t)$, we implement a self-consistent control scheme to suppress it from the dynamics. Our method works when the intrinsic dynamics are chaotic, periodic, or stationary.

We begin by outlining our method for identifying the unknown disturbance $\mathbf{g}(t)$. We will illustrate our technique using reservoir computing, a type of machine learning framework particularly suited for dynamical systems problems²⁶. In our implementation, we assume that we run the system in Eq. (2) during the “training” interval $[-T, 0]$, and collect a time-series of the observed state vector $\{\hat{\mathbf{x}}(-T), \hat{\mathbf{x}}(-T + \Delta t), \dots, \hat{\mathbf{x}}(0)\}$. These variables are fed to the reservoir, a high-dimensional dynamical system with internal variables $\mathbf{r} \in \mathbb{R}^M$, where M is the size of the reservoir. Here, following^{17,19}, we implement the reservoir as the map

$$\mathbf{r}(t + \Delta t) = \tanh[\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{\text{in}}\hat{\mathbf{x}}(t) + \beta], \quad (3)$$

where the $M \times M$ matrix \mathbf{A} is a sparse matrix representing the internal structure of the reservoir network and the $M \times N$ matrix \mathbf{W}_{in} is a fixed input matrix. Here we choose the bias parameter $\beta=1$, which for our purposes nearly optimizes results (see Methods). The reservoir output \mathbf{u} is constructed from the internal states as $\mathbf{u} = \mathbf{W}_{\text{out}}\mathbf{r}$, where the $N \times M$ output matrix \mathbf{W}_{out} is chosen so that \mathbf{u} approximates as best as possible the known training forcing function $\mathbf{f}(t)$. The optimization can be done by minimizing the cost function

$$\sum_{n=0}^{T/\Delta t} \|\mathbf{f}(-n\Delta t) - \mathbf{u}(-n\Delta t)\|^2 + \lambda \text{Tr}(\mathbf{W}_{\text{out}}\mathbf{W}_{\text{out}}^T) \quad (4)$$

via a ridge regression procedure, where the constant $\lambda \geq 0$ prevents over-fitting. With this procedure, the reservoir is trained to identify the forcing function $\mathbf{f}(t)$ given the observed values of $\hat{\mathbf{x}}(t)$. The reservoir can then be presented with a time series of the observed variables taken from (1), i.e., it can be evolved as

$$\mathbf{r}(t + \Delta t) = \tanh[\mathbf{A}\mathbf{r}(t) + \mathbf{W}_{\text{in}}\mathbf{x}(t) + 1], \quad (5)$$

The reservoir output $\mathbf{u}(t) = \mathbf{W}_{\text{out}}\mathbf{r}$ will be, if the method is successful, a good approximation to the unknown disturbance, $\mathbf{u} \approx \mathbf{g}$. As we will see, the reservoir robustly identifies disturbances it has not observed previously. Fig. 1 illustrates our procedure in the training phase (top row) and recovery phase (bottom row).

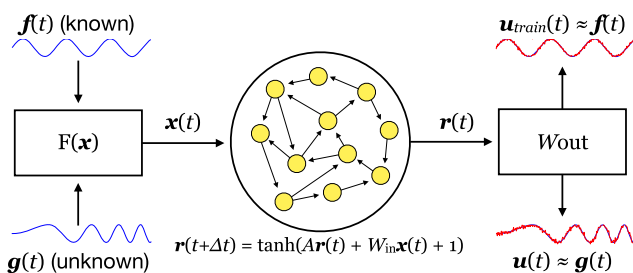


Fig. 1 | Schematic illustration of our method. In the training phase (top row), a nonlinear system is forced with a training function $\mathbf{f}(t)$. Observations of the forced system are used to train a reservoir to approximate the training function, $\mathbf{u}_{\text{train}}(t)$. The reservoir subsequently identifies unknown disturbance function $\mathbf{g}(t)$ with an approximate disturbance function $\mathbf{u}(t)$ (bottom row).

In our numerical examples, the reservoir matrix \mathbf{A} is a random matrix of size $M=1000$ where each entry is uniformly distributed in $[-0.5, 0.5]$ with probability $6/M$ and 0 otherwise, and rescaled so that its spectral radius is 1.2. The input matrix \mathbf{W}_{in} is a random matrix where each entry is uniformly distributed in $[-0.01, 0.01]$. The ridge regression regularization constant is $\lambda = 10^{-6}$. We train the reservoir for $T = 150$ time units and use Euler’s method to solve the differential equations with a time step $\Delta t = 0.002$.

Simulated examples: deterministic and stochastic disturbances

We first demonstrate our method with numerical simulations. For the numerical simulations, we consider a system where the intrinsic dynamics are given by the Lorenz system²⁹, i.e., system (1) is

$$\frac{dx_L}{dt} = \sigma(y_L - x_L) + g_x(t), \quad (6)$$

$$\frac{dy_L}{dt} = x_L(\rho - z_L) - y + g_y(t), \quad (7)$$

$$\frac{dz_L}{dt} = x_L y_L - \beta z_L + g_z(t), \quad (8)$$

with $\rho=28$, $\sigma=10$, and $\beta=8/3$. For the unknown disturbance we consider two examples: (i) a deterministic forcing $[g_x, g_y, g_z]^T = [x_R/10, y_R/10, 0]^T$, where $x_R(t)$ and $y_R(t)$ are the x and y coordinates of an auxiliary Rössler system³⁰ (assumed to be unknown),

$$\frac{dx_R}{dt} = -y_R - z_R, \quad (9)$$

$$\frac{dy_R}{dt} = x_R + ay_R, \quad (10)$$

$$\frac{dz_R}{dt} = b + z_R(x_R - c), \quad (11)$$

with $a=0.2$, $b=0.2$, and $c=5.7$, and (ii) a stochastic forcing $[x_S, y_S, 0]^T$ where both x_S and y_S satisfy the Langevin equations

$$\frac{dx_S}{dt} = -\frac{x_S}{2} + \eta_x(t), \quad (12)$$

$$\frac{dy_S}{dt} = -\frac{y_S}{2} + \eta_y(t), \quad (13)$$

where η_x and η_y are both white noise terms satisfying $\langle \eta(t)\eta(t') \rangle = 2D\delta(t - t')$, with $D = 1.25$.

We present our results in Fig. 2 demonstrating the performance of the reservoir in recovering the unknown disturbance for different choices of training forcing function $[f_x(t)f_y(t), 0]^T$. (For simplicity of visualization we assume it is known that the forcing in the z coordinate is zero). Along the top row, i.e., panels (a)–(c), we plot the trajectory of the unknown disturbance functions to be reconstructed as black curves, the reconstructed disturbances as red curves, and the training forcing functions as blue curves and circles. (Fig. 2c only shows the last 1/8 portion of the time-series.) From left to right we have trained the reservoirs with forcing functions consisting of a sine/cosine pair $[f_x(t), f_y(t)]^T = [\cos(t/20), \sin(t/20)]^T$, a slightly offset pair of cosine functions $[f_x(t), f_y(t)]^T = [\cos(t/20), \cos((t-1)/20)]^T$, and piecewise constant functions $[f_x(t), f_y(t)]^T = [\text{sign}(\cos(t/20)), \text{sign}(\sin(t/20))]^T$. Time series for the unknown and recovered disturbances, $g_i(t)$ and $u_i(t)$, are compared in the bottom row, (d)–(f), plotted in solid black and dashed red, respectively.

Remarkably, our results show that the reservoir can identify a chaotic or stochastic forcing function to the Lorenz system even when it was trained with a periodic function [Fig. 2(a)], or a piecewise constant function with

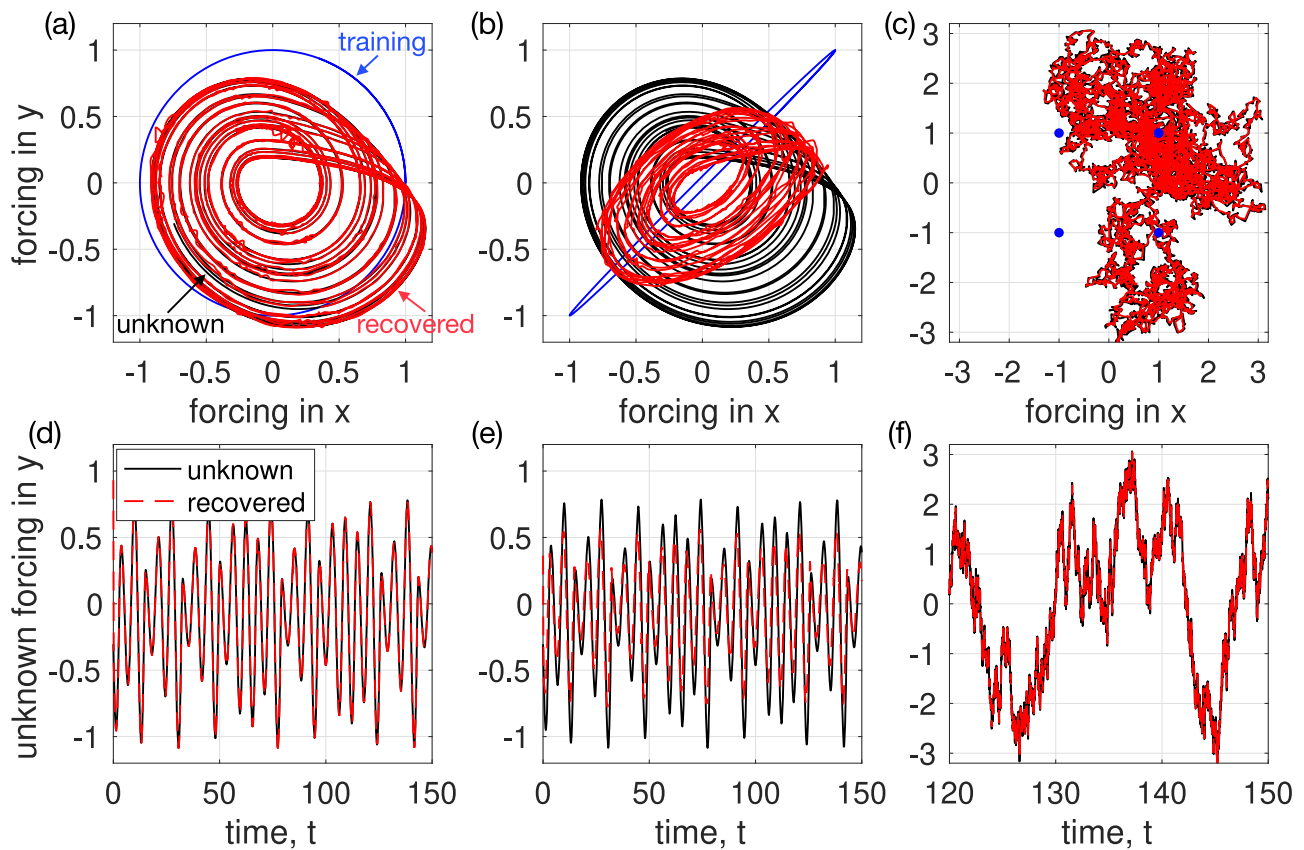


Fig. 2 | Identifying unknown disturbances. **a–c** Unknown and reconstructed disturbance functions $[g_x(t), g_y(t)]$ (black curve) and $[u_x(t), u_y(t)]$ (red curve) along with the training forcing functions $[f_x(t), f_y(t)]$ (thick blue curves and symbols). For each case the reservoir was trained with **(a)** $[f_x(t), f_y(t)] = [\cos(0.05t), \sin(0.05t)]$, **(b)** $[f_x(t), f_y(t)] = [\cos(0.05t), \cos(0.05t - 0.05)]$, and **(c)** $[f_x(t), f_y(t)] = [\text{sign}(\cos(0.05t)), \text{sign}(\sin(0.05t))]$ and disturbed with **(a, b)** Rossler dynamics and **(c)** Langevin dynamics. **d, e** Time series for the unknown (solid black) and recovered (dashed red) disturbance functions in the y component, $g_y(t)$ and $u_y(t)$.

only four different values [Fig. 2c]. Figure 2b illustrates the limitations on the forcing functions used to train the reservoir. In this example, the forcing functions satisfy $f_x \approx f_y$. Given this limited training, the reservoir has trouble extrapolating to functions away from the manifold $f_x = f_y$, and the reconstruction of the disturbance suffers.

Next we present some additional results demonstrate generalizability of our mechanism. First, we consider an inverted version of our first example, namely we consider system dynamics defined by the Rössler system [i.e., Eqs. (9–11)] that are disturbed by time series arising from the Lorenz system [i.e., Eqs. (6–8)]. Specifically, we set $[g_x(t), g_y(t)]^T = [x_L(t)/20, y_L(t)/20]^T$ and use sinusoidal forcing, as before, for training, namely $[f_x(t), f_y(t)]^T = [\cos(t/20), \sin(t/20)]^T$. Results for this inverted example are plotted in Figs. 3a and c, and show good agreement between the unknown and recovered disturbances. Second, to illustrate the efficacy of the methodology in a higher-dimensional system, we consider the Lorenz 96 model³¹ whose variables x_i for $i = 1, \dots, N$ evolve according to

$$\frac{dx_i}{dt} = (x_{i+1} - x_{i-2})x_{i-1} - x_i + F. \quad (14)$$

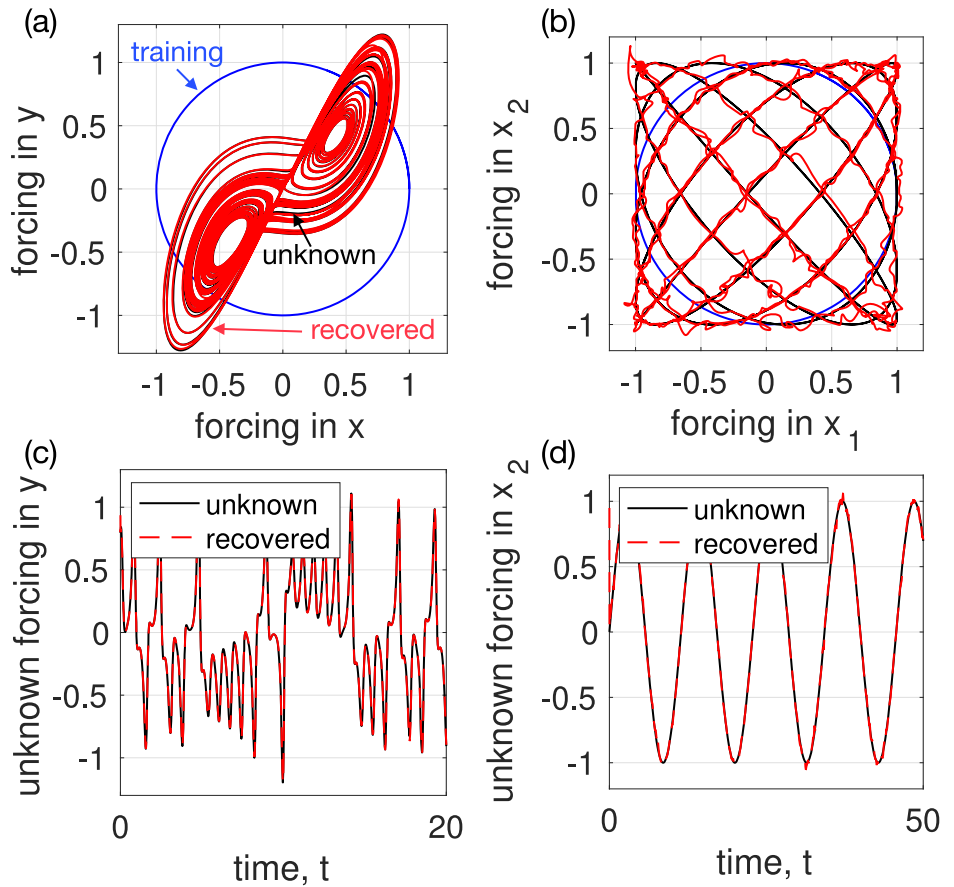
Here we choose the dimension $N=8$ and set $F=8$ to realize high-dimensional chaos. We train the system with known forcing in the first two variables as in the prior example, $[f_{x_1}(t), f_{x_2}(t)]^T = [\cos(t/20), \sin(t/20)]^T$, then use an unknown disturbance of $[g_{x_1}(t), g_{x_2}(t)]^T = [\cos(t/2), \sin(11t/20)]^T$. Due to the high dimensionality we use a reservoir of twice the size as in other examples, namely, $M = 2000$. Results for this example are plotted in Fig. 3b and d and show good agreement between the unknown and recovered disturbances.

Experimental examples: a chaotic circuit

In addition to the numerical simulations presented above, we also demonstrate that our method can recover unknown disturbances in an experimental setting. An analog electric circuit which reproduces the dynamics of the Lorenz equations was built following ref. 32 (see Methods) and arbitrary waveform generators were used to introduce various types of additive forcing terms in both the x and y variables as in Eqs. (6–7). The circuit variables x , y , and z were sampled at a rate of 10 kHz for 20 s when forced with various choices of \mathbf{f} and \mathbf{g} . In Fig. 4a and b we present the results obtained from training the reservoir using the dynamics of the circuit under piecewise constant and sinusoidal forcing, respectively, and recovering the more complicated unknown disturbance. To alleviate noise effects, the recovered disturbance is a moving average of the reservoir prediction with a window of 20 ms. Time series for the unknown disturbance and the noise-filtered recovered disturbance are shown in Fig. 2c, d. Despite some noise, the reservoir robustly recovers the disturbances.

An important question is what training forcing function \mathbf{f} should one use in order to recover an a priori unknown disturbance \mathbf{g} . In our numerical experiments, we have found that the reservoir computer is able to identify disturbances with range in a region approximately 5 times larger than the convex hull of the set $\{\mathbf{f}(-n\Delta t)\}_{n=0}^{T/\Delta t}$, with the same center. This condition is very mild and can be met with a variety of simple forcing functions, for example a piecewise constant function with only three values. Intuitively, if the range of the training forcing function \mathbf{f} does not contain enough information for the reservoir computer to extrapolate and infer the disturbance, the process will fail. In this Article we have not attempted a rigorous or more general analysis of the conditions that training forcing functions should satisfy, and leave this for future research. In addition to the

Fig. 3 | Identifying unknown disturbances: Rössler and Lorenz 96 systems. **a, b** Unknown and reconstructed disturbance functions $[g_x(t), g_y(t)]$ (black curve) and $[u_x(t), u_y(t)]$ (red curve) along with the training forcing functions $[f_x(t), f_y(t)]$ (blue curves). For each case the reservoir was trained with $[f_x(t), f_y(t)] = [\cos(0.05t), \sin(0.05t)]$ and disturbed with **(a)** Lorenz dynamics and **(b)** sinusoidal functions. **c, d** Time series for the unknown (solid black) and recovered (dashed red) disturbance functions in the second component, $g_y(t)$ and $u_y(t)$ (or $g_{x_2}(t)$ and $u_{x_2}(t)$).



example above where the unknown disturbance is chaotic, we have also successfully identified temporally localized, constant, periodic, and slowly varying, non-oscillatory forcing functions $\mathbf{g}(t)$.

Suppression of disturbances

Next we consider the problem of suppressing the undesired disturbance function $\mathbf{g}(t)$ with the aim of recovering the approximate undisturbed dynamics $d\mathbf{x}/dt = \mathbf{F}(\mathbf{x})$. For this we assume that the procedure described above has been successful, and that $\mathbf{u}(t) = W_{\text{out}}\mathbf{r}$ approximates the forcing to the system. We motivate our subsequent method by first considering a scheme where Eq. (1) is modified to

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}(\mathbf{x}) + \mathbf{g}(t) - \alpha\mathbf{u}(t), \quad (15)$$

where α is the control gain, and \mathbf{u} is obtained by feeding \mathbf{x} to the trained reservoir. We refer to this scheme as the *simple control* scheme. Since the reservoir was trained to identify the forcing, in principle we have a self-consistent relationship

$$\mathbf{u} \approx \mathbf{g} - \alpha\mathbf{u}, \quad (16)$$

with solution

$$\mathbf{u}(t) \approx \frac{\mathbf{g}(t)}{1 + \alpha}. \quad (17)$$

The effective forcing $\mathbf{g}(t) - \alpha\mathbf{u}(t)$ in Eq. (15) reduces to $\mathbf{g}(t)/(1 + \alpha)$. In principle, then, choosing $\alpha \gg 1$ suppresses the forcing. However, this control scheme becomes unstable for moderate values of α . To understand this, we assume momentarily that \mathbf{g} is constant, and study the stability of the

control scheme. On a given time step, the reservoir tries to approximate the forcing in Eq. (15), which is based on the previous reservoir output. Therefore, Eq. (16) needs to be treated as a dynamical system. A first approximation is

$$\mathbf{u}(t + \Delta t) \approx \mathbf{g} - \alpha\mathbf{u}(t), \quad (18)$$

which assumes that the reservoir approximates its own output at the previous time. In reality, the right-hand side of Eq. (18) might depend on previous history. Therefore, we regard Eq. (18) as a rough approximation to guide us in constructing a useful control scheme. Under Eq. (18), the fixed point (17), and therefore the control scheme, becomes unstable for $\alpha > 1$. In our example, we find numerically that the scheme becomes unstable at $\alpha \approx 2.5$, presumably due to the fact that Eq. (18) is only an approximation. Additional tests using non-constant \mathbf{g} show the same behavior.

In order to create a more robust control scheme, we modify (15) to

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}(\mathbf{x}) + \mathbf{g}(t) - \alpha\mathbf{v}(t), \quad (19)$$

$$\frac{d\mathbf{v}}{dt} = \frac{1}{\tau}(\mathbf{u} - \mathbf{v}), \quad (20)$$

where τ is a control parameter. We refer to this scheme as the *delayed control* scheme, since \mathbf{v} represents an exponentially weighted average of the previous values of \mathbf{u} . Now we repeat our previous approximation to this scheme. If, for example, the dynamics are solved using Euler's method, Eq. (18) now becomes

$$\mathbf{u}(t + \Delta t) \approx \mathbf{g} - \alpha\mathbf{v}(t), \quad (21)$$

Fig. 4 | Experimental results. **a, b** Unknown and reconstructed disturbance functions $[g_x(t), g_y(t)]$ (black curve) and $[u_x(t), u_y(t)]$ (red curve) along with the training forcing functions $[f_x(t), f_y(t)]$ (thick blue curves and symbols). For each case the reservoir was trained with **(a)** a 5 Hz square wave out of phase by $\pi/2$ and **(b)** $[f_x(t), f_y(t)] = [\cos(10\pi t), \sin(10\pi t)]$ and disturbed with combinations of sinusoidal functions. **c, d** Time series for the unknown (solid black) and recovered (dashed red) disturbance functions in the y component, $g_y(t)$ and $u_y(t)$.

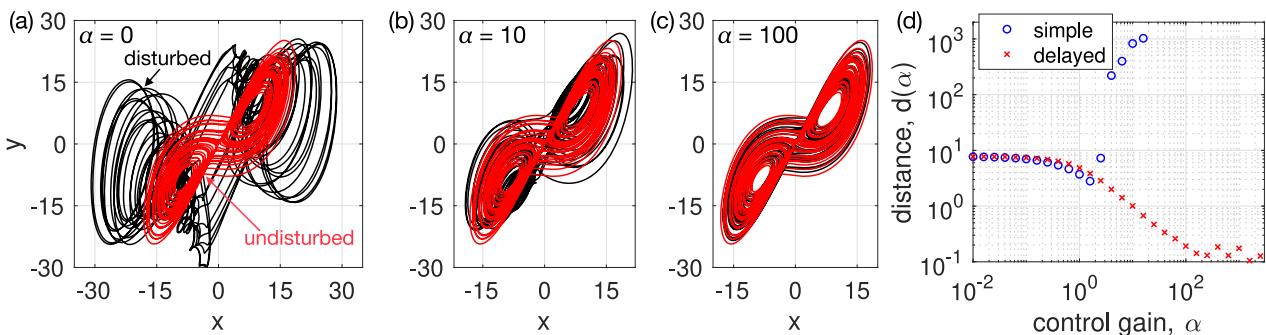
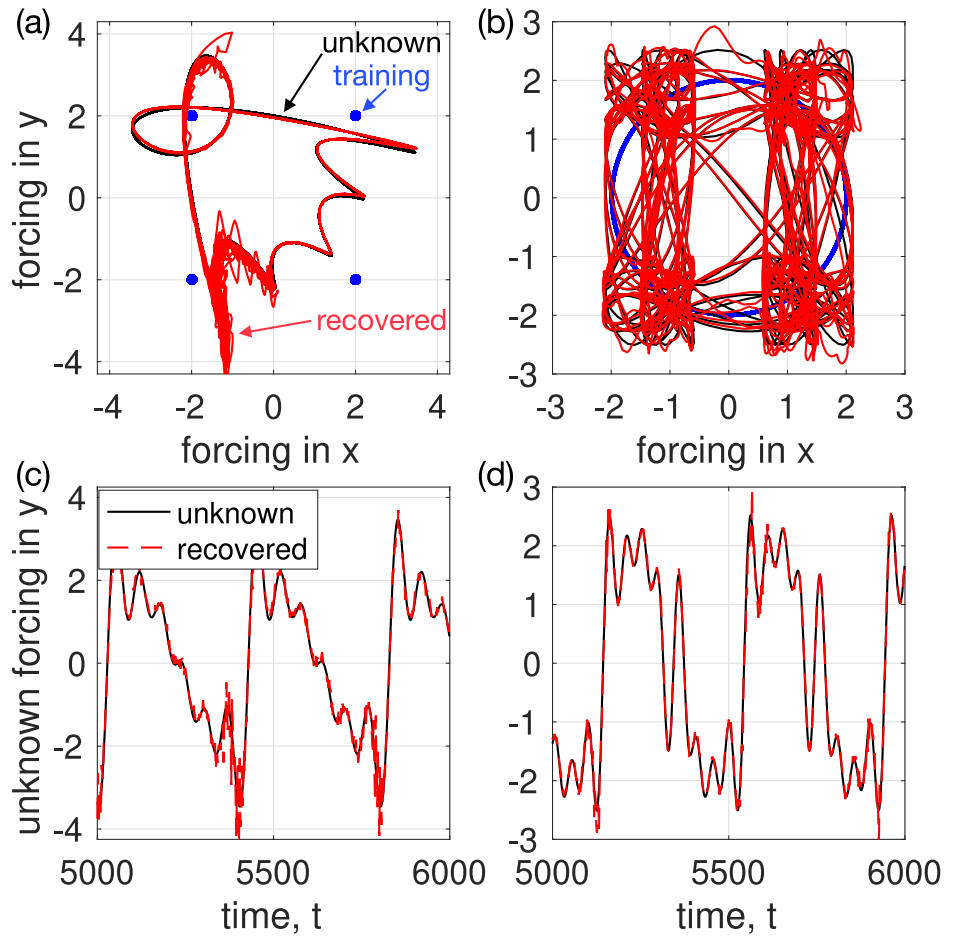


Fig. 5 | Suppressing unknown disturbances. **a–c** For control gains $\alpha = 0, 10$, and 100 , the disturbed attractor obtained from delayed control (black curves) compared to the undisturbed reference attractor (red curves). **d** As a function of the control

gain α , the average distance $d(\alpha)$ between the disturbed and original attractors as simple and delayed control (blue circles and red crosses) is applied to the disturbed attractor.

$$\mathbf{v}(t + \Delta t) \approx \mathbf{v}(t) + \frac{\Delta t}{\tau} [\mathbf{u}(t) - \mathbf{v}(t)]. \quad (22)$$

Again assuming constant \mathbf{g} , a linear stability analysis shows that when $\tau/\Delta t > 1$ the fixed point $\mathbf{u} = \mathbf{v} = \mathbf{g}/(1 + \alpha)$ is linearly stable as long as $\alpha < \tau/\Delta t$. While we don't expect this estimate to be exact, we expect that the range of values of α for which the control scheme is stable will be greatly expanded when $\tau/\Delta t$ is large. Interestingly, in contrast to typical control problems, the presence of delays increases the stability of the control scheme. In summary, the delayed control algorithm for suppressing a disturbance $\mathbf{g}(t)$ is as follows: (i) Force the system with a known training forcing function \mathbf{f} , and train a reservoir computer so that its output \mathbf{u} approximates \mathbf{f} based on

observations of the state variables $\hat{\mathbf{x}}$. (ii) Add a term $-\alpha \mathbf{v}$ to the disturbed system, where \mathbf{v} satisfies Eq. (20) with large τ .

Simulated examples: suppressing deterministic disturbances

In order to demonstrate the suppression method discussed above, we return to our example of a Lorenz system forced by a Rössler system, except that the forcing applied to Eqs. (6–8), are greatly amplified, namely, $[g_x, g_y, g_z]^T = [24x_R, 24y_R, 0]^T$. (In order to illustrate the power of our method, the disturbance terms are chosen to be much larger than in the previous example.) After training the reservoir with the sinusoidal forcing $[f_x(t), f_y(t), f_z(t)]^T = [\cos(0.05t), \sin(0.05t), 0]^T$, we run the control

scheme in Eqs. (19–20). In Fig. 5a–c, we plot for control gains $\alpha = 0, 10$, and 100 the disturbed Lorenz system in black curves as well as the undisturbed Lorenz system in red curves for comparison. Note that for $\alpha = 0$ (i.e., no control) the disturbed system attractor bears little resemblance to the undisturbed system attractor, but as α is increased the control method begins to effectively mitigate the disturbances, with little effective difference for $\alpha = 100$. In order to quantify the effectiveness of the control method, we measure the distance between the disturbed and controlled attractor and the undisturbed attractor as follows. We solve Eqs. (6–8) with $g_x = g_y = g_z = 0$ for $T = 150$ time units using Euler's method with a timestep $\Delta t = 0.002$ after discarding a sizable transient and create a reference time series $\{\mathbf{x}_0(0), \mathbf{x}_0(\Delta t), \mathbf{x}_0(2\Delta t), \dots, \mathbf{x}_0(T/\Delta t)\}$ representing an approximation of the undisturbed attractor. Next, for a given value of α , again after discarding a transient, we compute a time series for the disturbed and controlled system, $\{\mathbf{x}(0), \mathbf{x}(\Delta t), \mathbf{x}(2\Delta t), \dots, \mathbf{x}(T/\Delta t)\}$. Then we compute the average distance between the points on the controlled trajectory and the reference time series

$$d(\alpha) = \frac{1}{T/\Delta t} \sum_{i=0}^{T/\Delta t} \min_j \|\mathbf{x}(i\Delta t) - \mathbf{x}_0(j\Delta t)\|. \quad (23)$$

In Fig. 5d we plot the distance $d(\alpha)$ versus α for both the simple control scheme (15) (blue circles) and for the delayed control scheme (19)–(20) (red crosses) for the deterministic disturbance. The simple control scheme reduces the error until it becomes unstable at approximately $\alpha \sim 2.5$. In contrast, the delayed control scheme reduces the error to very small levels for large values of α , before it also becomes unstable at approximately $\alpha \sim 2500$. (For both methods, the values of α for which no data are shown resulted in numerical instability.) In practice, a suitable value of α could be chosen either by comparing the controlled attractor to the undisturbed one, if it is available, or by choosing α large enough that the controlled attractor doesn't

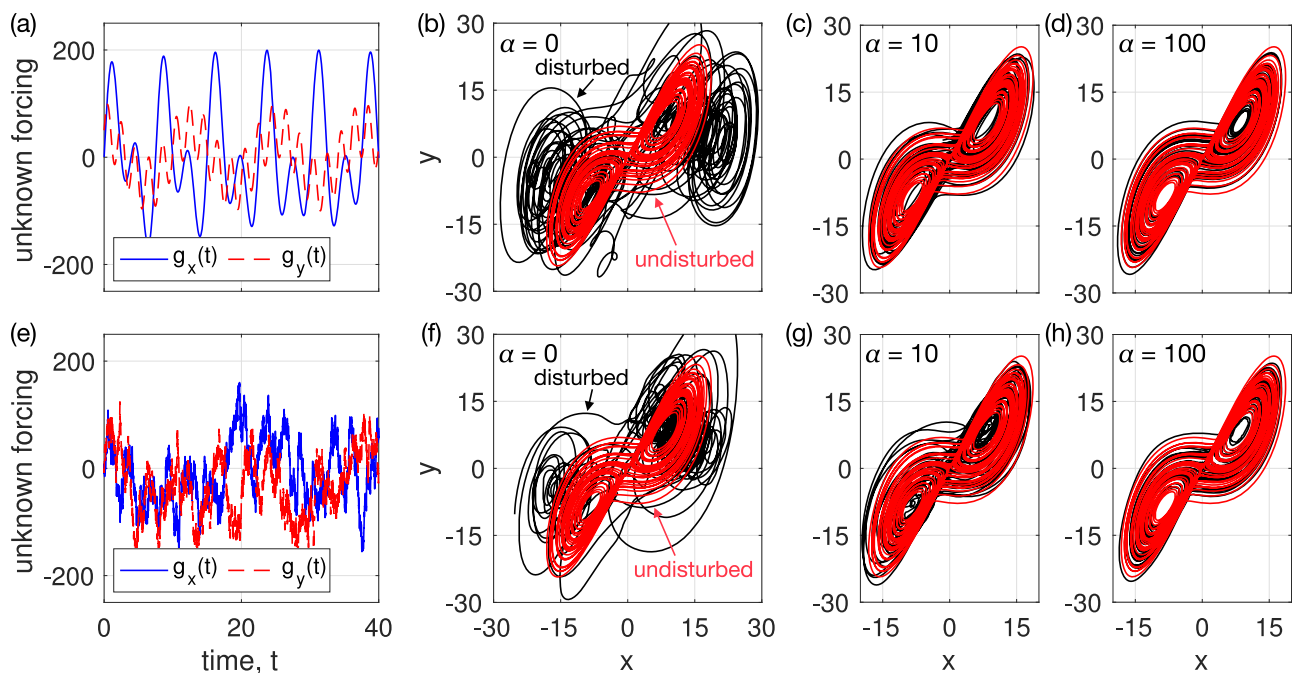


Fig. 6 | Suppressing unknown disturbances: Quasi-periodic and stochastic disturbances. For quasi-periodic (a–d) and stochastic (e–h) disturbances, the effectiveness of delayed control. In (a) and (e) the disturbances applied to the Lorenz

system. For each case, (b–d) and (f–h) the disturbed (black) and undisturbed (red) attractors when using control gains $\alpha = 0, 10$, and 100.

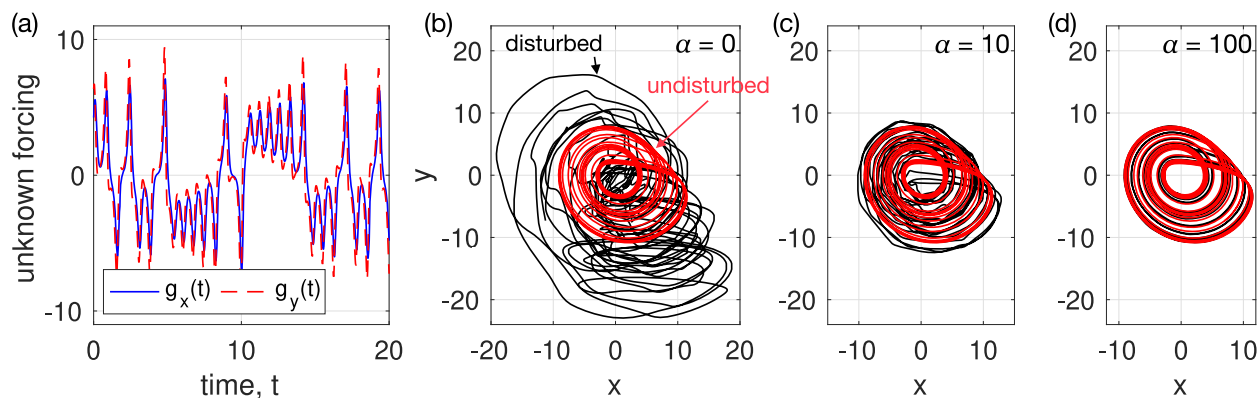


Fig. 7 | Suppressing unknown disturbances: Rössler system. a The disturbances applied to the Rössler system. b–d the disturbed (black) and undisturbed (red) attractors when using control gains $\alpha = 0, 10$, and 100.

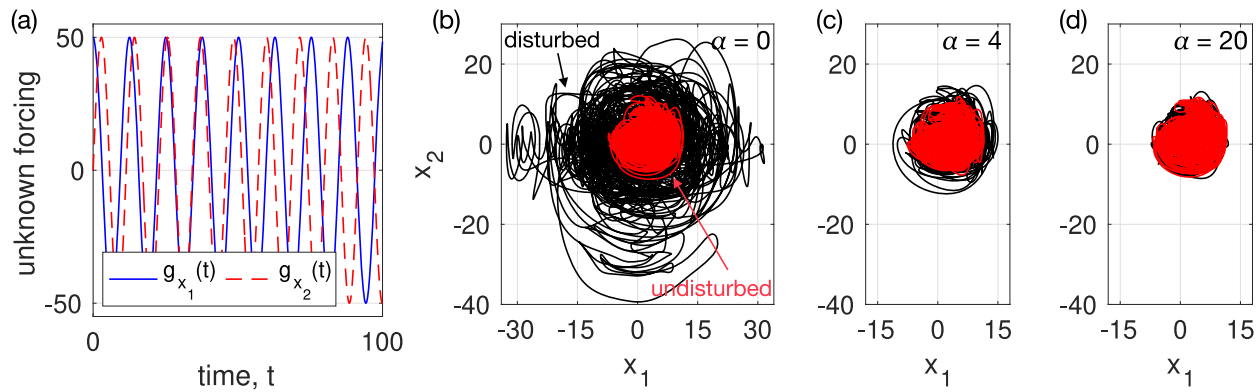


Fig. 8 | Suppressing unknown disturbances: Lorenz 96 system. **a** The disturbances applied to the Lorenz 96 system. **b–d** the disturbed (black) and undisturbed (red) attractors when using control gains $\alpha = 0, 4$, and 20 .

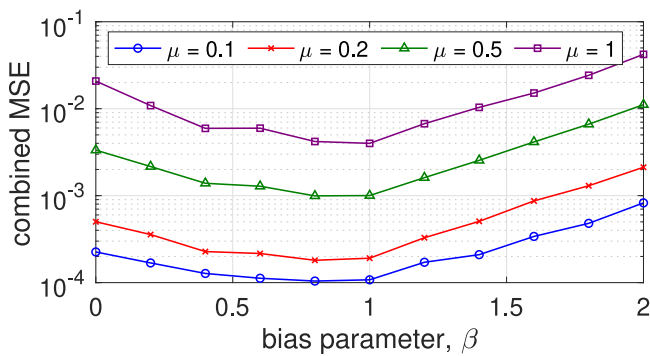


Fig. 9 | Effect of the bias parameter. As a function of the bias parameter β , the combined mean squared error (MSE) between the unknown disturbances, g_x and g_y , and those recovered by the reservoir, u_x and u_y . Unknown disturbances were presented at different magnitudes, scaled by μ as $[g_x, g_y, g_z]^T = [\mu x_R, \mu y_R, 0]^T$.

change appreciably when increasing α further, as it is often done with the time-step of numerical ODE solvers.

We also present some additional results that demonstrate the generalizability of the suppression mechanism in the face of different types of disturbances. In particular, while we keep the undisturbed system defined by the Lorenz system, we consider first quasi-periodic disturbances that are composed from mismatched sinusoids, namely $[g_x, g_y, g_z]^T = [200 \cos(2t/5) \sin(2\pi t/5), 50 \cos(t/2) + 50 \sin(\pi t), 0]^T$, and, second, stochastic disturbances $[g_x, g_y, g_z]^T = [x_S(t), y_S(t), 0]^T$ as defined in Eqs. (12) and (13) but with a stronger stochastic component, specifically $D = 75$. We plot the results from these two cases under delayed control in Fig. 6 on the top and bottom, respectively. First, in panels (a) and (e) we plot the time series of the disturbances applied to the Lorenz system, depicting the disturbances to the x and y components in solid blue and dashed red, respectively. Next, in panels (b) and (f) we plot the disturbed attractor without control (i.e., using $\alpha = 0$) in black and also plotting the undisturbed attractor in red for comparison. Then in panels (c) and (g) we plot the disturbed and undisturbed attractors for $\alpha = 10$, then in panels (d) and (h) for $\alpha = 100$. As we increase the control gain α we see the disturbed attractor begins to resemble more so the undisturbed attractor.

Lastly we consider control of disturbances to other systems, specifically the Rössler system and the Lorenz 96 system with $N = 8$ and $F = 8$. In both cases we train the systems with sinusoidal forcing, $[f_x(t), f_y(t)]^T = [\cos(0.05t), \sin(0.05t)]^T$. Next, we disturb the x and y components of the Rössler system with the x and y components of the Lorenz system, $[g_x, g_y]^T = [2x_L(t/5, 2y_L(t/5)]^T$ and we disturb the first two components of the Lorenz 96 system with sinusoids with offset frequencies, $[g_{x_1}(t), g_{x_2}(t)]^T = [50 \cos(t/2), 50 \sin(11t/20)]^T$. In Figs. 7 and 8 we plot the results for the Rössler system and the Lorenz 96 system, respectively,

plotting in panel (a) the disturbances applied to each, then in panels (b)–(d) the disturbed (black) and undisturbed (red) attractors as the control gain is increased: for the Rössler system we use $\alpha = 0, 10$, and 100 and for the Lorenz 96 system we use $\alpha = 0, 4$, and 20 . (Note that for the Lorenz 96 system we were able to suppress disturbances quite well with even smaller control gains, thus the smaller values of α used.) In both cases we see that as the control gain is increased the disturbed dynamics get closer to the undisturbed dynamics.

Discussion

In summary, we have presented and demonstrated both numerically and experimentally a method that allows an unknown disturbance to an unknown dynamical system to be identified and suppressed in real-time, based only on previous observations of the system forced with a known forcing function. Our method is applicable, for example, to the problem of identifying node and line disturbances in networked dynamical systems such as power grids^{7–9}, and more broadly to the various fields where disturbances need to be suppressed in real-time⁴. While our method does not require knowledge of the underlying dynamics of the system, it requires one to be able to force it with the addition of a known training forcing function, and subsequently with the term $-\alpha v$. The consideration of nonlinear disturbances is left for another manuscript³³. In addition, we assumed that all the variables of the system can be observed. In principle, one could use our method by training the reservoir using an observed function $\mathbf{H}(\mathbf{x})$ of the state vector, but we have not explored this generalization. Another important research direction is to determine the class of appropriate training forcing functions, given a dynamical system and the anticipated characteristics of the disturbance.

Methods

Choice of the bias parameter, β

To explore the choice that the bias parameter β [see Eq. (3)] has on the ability of the reservoir computer to recover unknown disturbances, we return to our first example of a Lorenz system with sinusoidal known forcing used as training and then disturbed by Rössler dynamics. All other system and reservoir computer parameters are the same, training is set to $[f_x(t), f_y(t), f_z(t)]^T = [\cos(t/20), \sin(t/20), 0]^T$, but we consider varying both the bias parameter β and the magnitude of the Rössler forcing, namely, we introduce a magnitude parameter μ that scales the unknown disturbance as $[g_x, g_y, g_z]^T = [\mu x_R, \mu y_R, 0]^T$. (Note that in the main text we first used $\mu = 1/10$.) To examine the effect of the bias parameter we then train the reservoir with different β , then with that chosen value of β try to extract the unknown disturbance at different levels of μ . We evaluate the success of the reservoir in extracting the disturbances by calculating the sum of the mean squared error (MSE) in both the x and y components over the time window, namely, $\text{MSE} = (\int_0^T [g_x(t) - u_x(t)]^2 dt + \int_0^T [g_y(t) - u_y(t)]^2 dt) / T$. In Fig. 9 we plot the the combined MSE as a function of the bias parameter β for

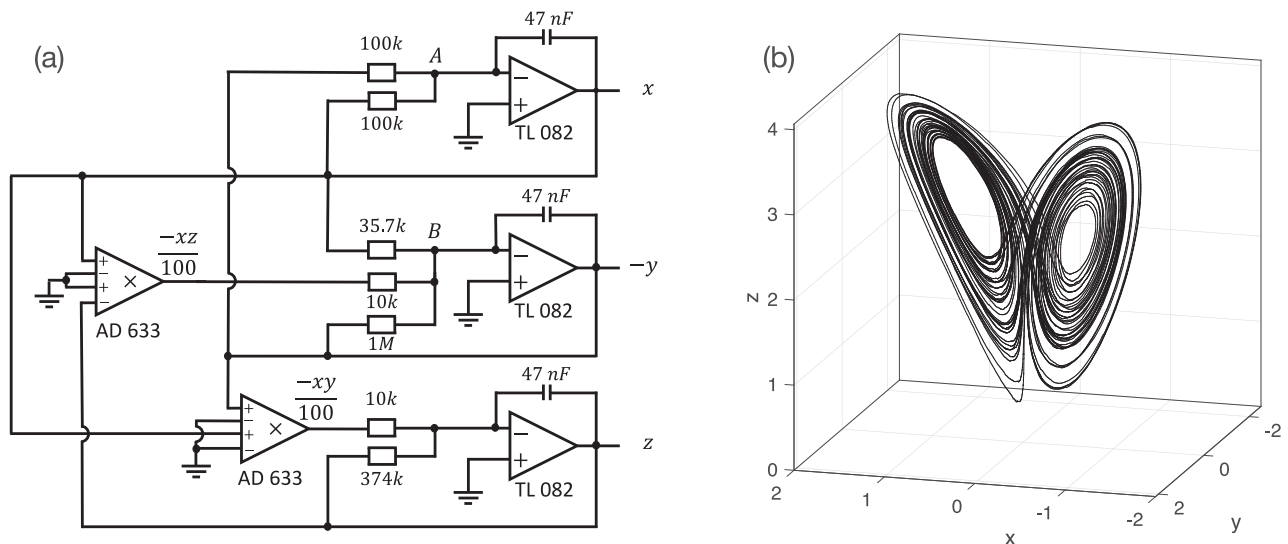


Fig. 10 | Experimental setup. **a** Circuit for Lorenz attractor. Integrators are based on the TL082 operational amplifier while the multipliers are based on the AD633 chip. Circuit recreated from ref. 32. **b** Output of the circuit, where units are in volts.

a number of choices of the magnitude parameter, specifically $\mu = 0.1$ (blue circles), 0.2 (red crosses), 0.5 (green triangles), and 1 (purple squares). Results demonstrate that a bias parameter is optimal near $\beta = 1$, which informs the choice made in this paper.

Experimental implementation

We constructed an analog electric circuit to replicate the Lorenz equations through three integrators and two multipliers, following the implementation described in refs. 32,34 and shown schematically in Fig. 10a. In this implementation, the variables x , y , and z are the voltages shown in the diagram in Fig. 10a and correspond to the respective variables in the Lorenz system scaled down by a factor of 10 (the equations the system models are modified accordingly for this scaling).

The values used for the resistors are chosen to produce the appropriate coefficients in the Lorenz equations of $\sigma = 10$, $\beta = 8/3$, and $\rho = 28$. The integrating capacitors of 47 nF were chosen to provide oscillations on the order of 30 Hz . Resistors had a component tolerance of 1% , while the capacitors have a 5% tolerance. The multiplication was done with an AD633 analog multiplier, which has an error of 2% of full scale, while the integrating circuit was based on an TL082 operational amplifier. The output of this circuit is shown in Fig. 10b. The characteristic butterfly shape is readily apparent, with each output swinging around 4 Vpp . This analog circuit represents the “undisturbed system” described in the main text. While it is constructed to obey approximately the (scaled) Lorenz equations, the component tolerances make it, for practical purposes, an unknown system from which we can measure the state variables x , y , and z . Electronic noise and uncertainty from the analog multipliers adds an additional complication not present in our numerical simulations.

The external forcing is introduced into the circuit through the two points marked A and B in Fig. 10a. A function generator produces two signals at magnitudes that were approximately 4 Vpp to closely match the magnitude of the variables of the unforced circuit. These forcing signals were each passed through a unity gain buffer and a 1 MOhm resistor before being added to the signal at the input of the x and y integrators at A and B, respectively. The value of 1 MOhm allows the forcing signal to be of a comparable amplitude to the x , y , and z signals, and adds this function unscaled into the first two integrators.

Data availability

The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Code availability

The code used during the current study is available from the corresponding author on reasonable request.

Received: 27 June 2024; Accepted: 21 November 2024;

Published online: 19 December 2024

References

- Nudell, T. R. & Chakraborty, A. A graph-theoretic algorithm for disturbance localization in large power grids using residue estimation. *2013 American Control Conference*, 3467–3472 (IEEE, 2013).
- Upadhyaya, S. & Mohanty, S. Power quality disturbance localization using maximal overlap discrete wavelet transform. *2015 Annual IEEE India Conference (INDICON)*, 1–6 (IEEE, 2015).
- Mathew, A. T. & Aravind, M. Pmu based disturbance analysis and fault localization of a large grid using wavelets and list processing. *2016 IEEE Region 10 Conference (TENCON)*, 879–883 (IEEE, 2016).
- Chen, W.-H., Yang, J., Guo, L. & Li, S. Disturbance-observer-based control and related methods-an overview. *IEEE Trans. Ind. Electron.* **63**, 1083–1095 (2015).
- Ferreira, V. et al. A survey on intelligent system application to fault diagnosis in electric power system transmission lines. *Electr. Power Syst. Res.* **136**, 135–153 (2016).
- Lee, H.-W., Zhang, J. & Modiano, E. Data-driven localization and estimation of disturbance in the interconnected power system. *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, 1–6 (IEEE, 2018).
- Wang, D., Wang, X., Zhang, Y. & Jin, L. Detection of power grid disturbances and cyber-attacks based on machine learning. *J. Inf. security Appl.* **46**, 42–52 (2019).
- Delabays, R., Pagnier, L. & Tyloo, M. Locating line and node disturbances in networks of diffusively coupled dynamical agents. *N. J. Phys.* **23**, 043037 (2021).
- Delabays, R., Pagnier, L. & Tyloo, M. Locating fast-varying line disturbances with the frequency mismatch. *IFAC-PapersOnLine* **55**, 270–275 (2022).
- Meurant, G. The ecology of natural disturbance and patch dynamics (Academic press, 2012).
- Battisti, C., Poeta, G. & Fanelli, G. An introduction to disturbance ecology. *Cham: Springer* 13–29 (2016).

12. Bewley, T. R. & Liu, S. Optimal and robust control and estimation of linear paths to transition. *J. Fluid Mech.* **365**, 305–349 (1998).
13. Bewley, T. R., Temam, R. & Ziane, M. A general framework for robust control in fluid mechanics. *Phys. D: Nonlinear Phenom.* **138**, 360–392 (2000).
14. Juillet, F., McKeon, B. & Schmid, P. J. Experimental control of natural perturbations in channel flow. *J. fluid Mech.* **752**, 296–309 (2014).
15. Verbesselt, J., Hyndman, R., Zeileis, A. & Culvenor, D. Phenological change detection while accounting for abrupt and gradual trends in satellite image time series. *Remote Sens. Environ.* **114**, 2970–2980 (2010).
16. Canfield, J. C. Active disturbance cancellation in nonlinear dynamical systems using neural networks (University of New Hampshire, 2003).
17. 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).
18. Srinivasan, K. et al. Parallel machine learning for forecasting the dynamics of complex networks. *Phys. Rev. Lett.* **128**, 164101 (2022).
19. 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).
20. Canaday, D., Pomerance, A. & Gauthier, D. J. Model-free control of dynamical systems with deep reservoir computing. *J. Phys. Complex.* **2**, 035025 (2021).
21. Banerjee, A., Pathak, J., Roy, R., Restrepo, J. G. & Ott, E. Using machine learning to assess short term causal dependence and infer network links. *Chaos: Interdiscip. J. Nonlinear Sci.* **29**, 121104 (2019).
22. Pyragas, V. & Pyragas, K. Using reservoir computer to predict and prevent extreme events. *Phys. Lett. A* **384**, 126591 (2020).
23. Kong, L.-W., Weng, Y., Glaz, B., Haile, M. & Lai, Y.-C. Reservoir computing as digital twins for nonlinear dynamical systems. *Chaos: Interdisciplinary J. Nonlinear Sci.* **33**, 033111 (2023).
24. Patel, D. & Ott, E. Using machine learning to anticipate tipping points and extrapolate to post-tipping dynamics of non-stationary dynamical systems. *Chaos: Interdiscip. J. Nonlinear Sci.* **33**, 023143 (2023).
25. Tanaka, G. et al. Recent advances in physical reservoir computing: A review. *Neural Netw.* **115**, 100–123 (2019).
26. Nakajima, K. & Fischer, I. Reservoir computing (Springer, 2021).
27. Carroll, T. L. & Pecora, L. M. Network structure effects in reservoir computers. *Chaos: Interdiscip. J. Nonlinear Sci.* **29**, 083130 (2019).
28. Mandal, S. & Shrimali, M. D. Learning unidirectional coupling using an echo-state network. *Phys. Rev. E* **107**, 064205 (2023).
29. Lorenz, E. N. Deterministic nonperiodic flow. *J. Atmos. Sci.* **20**, 130–141 (1963).
30. Roessler, O. An equation for continuous chaos. *Phys. Lett. A* **57**, 397–398 (1976).
31. Lorenz, E. N. Predictability: a problem partly solved. *Proc. Seminar on predictability*, vol. 1 (1996).
32. Horowitz, P. & Hill, W. The Art of Electronics: The x Chapters, vol. 1 (Cambridge University Press, Cambridge, 2020).
33. Skardal, P. S. & Restrepo, J. G. Detecting disturbances in network-coupled dynamical systems with machine learning. *Chaos* **33**, 103137 (2023).
34. Fitch, A. L., Lu, H. H. & Lu, D. D. An analog computer for electronic engineering education. *IEEE Trans. Educ.* **54**, 550–557 (2010).

Acknowledgements

J.G.R. acknowledges support from NSF Grant DMS-2205967. P.S.S. acknowledges support from NSF grant MCB2126177.

Author contributions

J.G.R. and P.S.S. conceived the research. J.G.R., C.P.B., and P.S.S. performed the research. J.G.R., C.P.B., and P.S.S. wrote the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to Per Sebastian Skardal.

Peer review information *Communications Physics* thanks the anonymous reviewers for their contribution to the peer review of this work.

Reprints and permissions information is available at <http://www.nature.com/reprints>

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

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2024