

Reservoir Computing Generalized

Tomoyuki Kubota,^{1, 2,*} Yusuke Imai,¹ Sumito Tsunegi,³ and Kohei Nakajima^{1, 2}

¹*Graduate School of Information Science and Technology, The University of Tokyo, Japan*

²*Next Generation Artificial Intelligence Research Center (AI Center), The University of Tokyo, Japan*

³*National Institute of Advanced Industrial Science and Technology, Japan*

(Dated: December 18, 2024)

A physical neural network (PNN) [1–3] has both the strong potential to solve machine learning tasks and intrinsic physical properties, such as high-speed computation and energy efficiency. Reservoir computing (RC) [4–6] is an excellent framework for implementing an information processing system with a dynamical system by attaching a trained readout, thus accelerating the wide use of unconventional materials for a PNN [7–9]. However, RC requires the dynamics to reproducibly respond to input sequence [4], which limits the type of substance available for building information processors. Here we propose a novel framework called generalized reservoir computing (GRC) by turning this requirement on its head, making conventional RC a special case. Using substances that do not respond the same to identical inputs (e.g., a real spin-torque oscillator), we propose mechanisms aimed at obtaining a reliable output and show that processed inputs in the unconventional substance are retrievable. Finally, we demonstrate that, based on our framework, spatiotemporal chaos, which is thought to be unusable as a computational resource, can be used to emulate complex nonlinear dynamics, including large scale spatiotemporal chaos. Overall, our framework removes the limitation to building an information processing device and opens a path to constructing a computational system using a wider variety of physical dynamics.

The physical instantiation of neural networks is an urgent challenge to solve issues caused by traditional computers, such as energy consumption for computation [10–12]. Unconventional materials have intrinsic physical properties that are not found in the traditional substance of semiconductors, such as low energy consumption [2, 3] and high-speed processing [3], potentially allowing us to build a computational system with innovative physical properties. From the perspective of information processing, a key factor is the dynamical state in the physical system, which works as memory and nonlinear processing units. One of the most remarkable frameworks to physicalize neural networks is physical reservoir computing (PRC) [13].

Reservoir computing (RC) [4–6] is a machine learning framework for implementing an information processing system using a dynamical system, which provides the theoretical basis for PRC. In the RC framework, we inject inputs into the system and obtain outputs by attaching a trained readout, which can be either a (frequently used) linear function [4] or a nonlinear one [5]. This simple configuration is applicable to any dynamical system, including natural physical dynamics, to realize various types of physical reservoirs (e.g., electronics [14–16], quantum systems [17–20], optical components [21, 22], spintronics [23–25], mechanical structures [26–28], and organisms [29–31]). However, RC imposes a condition of echo state property (ESP) [4], in which the responses of the system be a function of only the previous input sequence (See Materials and Methods for further details). We call this a time-invariant (TI) state [5, 32] throughout this paper, as it guarantees a reproducible response to the same input sequence. Using the TI state and the trained readout, we can realize a required input-output relation in the entire computational system. Conversely, this condition limits the range of computational resources to the TI state, thereby excluding a great variety of oscillatory or chaotic materials included in the time-variant (TV) state.

In this paper, we propose generalized reservoir computing (GRC), a novel framework for exploiting systems, either with or without ESP, as a reliable computational resource. To create an information processor with TI outputs, the conventional RC adopts a TI dynamical state as well as a linear or nonlinear readout (Fig. 1a) [4, 5]. However, reliable information processing requires that the ESP be in the output layer, not in the reservoir layer. GRC generates an output with ESP from a general dynamical system to introduce a time-invariant (TI) transformation, regardless of the ESP of the dynamical state in the reservoir (Fig. 1b).

TI TRANSFORMATION

We consider an input-driven dynamical system described by $\mathbf{x}_{t+1} = \mathbf{g}(\mathbf{x}_t, \mathbf{u}_t)$ with the N -dimensional state \mathbf{x}_t and the M -dimensional input \mathbf{u}_t . Its solution \mathbf{x}_t can be regarded as a function of time t and input history $\{\mathbf{u}_{t-1}, \mathbf{u}_{t-2}, \dots\}$ with an initial state \mathbf{x}_0 , $\mathbf{x}_t = \mathbf{h}(t, \mathbf{u}_{t-1}, \mathbf{u}_{t-2}, \dots; \mathbf{x}_0)$ [33]. The conventional RC imposes the ESP

* kubota@isi.imi.i.u-tokyo.ac.jp

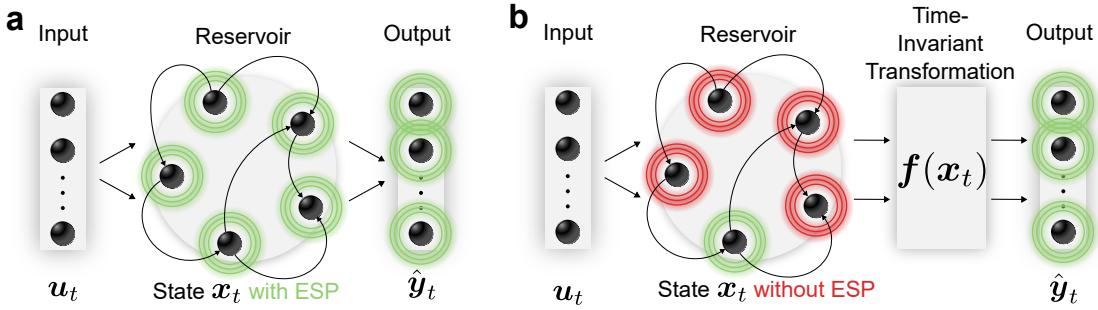


FIG. 1. **Conventional RC is a special case of GRC.** **a**, Conventional and **b**, GRC frameworks. The reservoir receives the input u_t and updates the state x_t . The outputs \hat{y}_t are calculated by **a**, a linear or nonlinear readout and **b**, a TI transformation, which would be realized through a nonlinear readout with memory $f(x_t)$. The green (red) ripples around the nodes represent the state with (without) node-wise ESP (See Supplementary Information S2 for further details). The conventional RC forms outputs with ESP using the reservoir states, thereby already fulfilling ESP, while the GRC can use not only states with ESP but also those without ESP, whose time-dependence is removed through TI transformation to make outputs equipped with ESP.

on the reservoir state, which is TI. Conversely, if all the states are functions of time and input history called TV [i.e., $x_{i,t} = h_i(t, u_{t-1}, u_{t-2}, \dots; x_0)$ ($i = 1, \dots, N$)], the conventional linear readout cannot always generate an output with ESP. The TV state can be transformed into TI outputs by nonlinearity with memory $f(x_t, x_{t-1}, \dots)$ in general (Note that linear weights can also be used for TI transformation in special cases [See Supplementary Information S2 for further details]). Unless otherwise noted, we focus on TI transformation without memory $f(x_t)$ for simplicity from now on (See Supplementary Information S6 for examples of TI transformation with memory). We call this transformation the TI transformation and define the GRC as follows:

$$x_{t+1} = g(x_t, u_t), \quad \hat{y}_t = f(x_t),$$

where the reservoir state x_t is either TI or TV, while the output \hat{y}_t is TI.

We introduce two analytical examples of the transformation mechanism. The first example removes the time dependence from the TV term $V(t, u_{t-1}, u_{t-2}, \dots; x_0)$ to form a TI term $T(u_{t-1}, u_{t-2}, \dots)$ (Fig. 2a, left). This transformation can be analytically shown using periodic dynamics as an example (Fig. 2a, middle), which is represented by the radius r_t and the angle θ_t . The analytical solution of the radius r_t is the TI function $r(u_{t-1}, u_{t-2}, \dots)$, and the angle is the linear function of time $\theta_t = \omega t$. The position on the orthogonal coordinate $x_t = (X_t, Y_t)^\top = (r_t \cos \theta_t, r_t \sin \theta_t)^\top$ depends both on time t and on input history $x_t = (r(u_{t-1}, u_{t-2}, \dots) \cos(\omega t), r(u_{t-1}, u_{t-2}, \dots) \sin(\omega t))^\top$, which is TV. Note that we have taken $(r_0, \theta_0) = (0, 0)$ and, from now on, we omit the dependence on the initial value from the following equations for simplicity. The TI and TV representations can be revealed by the temporal information processing capacity (TIPC) [20, 33], which expands a function F_t with TI function bases I_i and TV bases V_i as:

$$F_t = \sum_i \mathbf{a}_i I_i(u_{t-1}, u_{t-2}, \dots) + \sum_i \mathbf{b}_i V_i(t, u_{t-1}, u_{t-2}, \dots)$$

and evaluates the amount of processed inputs by the magnitude of coefficients $C_i^{\text{TI}} = \|\mathbf{a}_i\|^2$ and $C_i^{\text{TV}} = \|\mathbf{b}_i\|^2$. The sums of the TI (TV) capacities are illustrated by a bar graph without (with) hatchmarks for each order of input with different colors. For example, the state of periodic dynamics is expanded as $x_t = \sum_\tau (c_\tau u_{t-\tau} \cos(\omega t) + c_\tau u_{t-\tau} \sin(\omega t))$, which has only TV terms with zeroth- and first-order input (hatched purple and green in Fig. 2a, right). The conventional RC cannot form an output with ESP from these two states because time-dependent elements cannot usually be canceled out by a weighted sum of the states. To remove the time dependence, nonlinear transformation is required in the readout layer in this case. This state can be transformed into a TI state [e.g., $f(x_t) = X_t^2 + Y_t^2 = r_t^2(\cos^2 \theta_t + \sin^2 \theta_t) = r_t^2$, where r_t^2 is TI (non-hatched green and orange in Fig. 2a, right)].

The second example is to amplify the existing TI terms by removing TV terms. If the state is represented by a sum of TI and TV terms $I(u_{t-1}, u_{t-2}, \dots) + V(t, u_{t-1}, u_{t-2}, \dots)$, this creates the possibility of transforming the state into TI representation $T(u_{t-1}, u_{t-2}, \dots)$ (Fig. 2b, left). We demonstrate the amplification using the Lissajous knot system (Fig. 2b, middle). All three states $x_t = (X_t, Y_t, Z_t)^\top = (\cos(\omega t), \sin(\omega t), \sin(2\omega t) + \epsilon u_t)^\top$ include time-dependent terms, which make it impossible to form a TI output with linear readout, while a nonlinear readout [e.g., $f(x_t) = -2X_t Y_t + Z_t = \epsilon u_t$] enables us to calculate an output with ESP (Fig. 2b, right).

Finally, we introduce a numerical transformation with a nonlinear data-fitting technique to cover TI transformation that cannot be obtained in an explicit form, which includes majority of physical systems. To find the transformation,

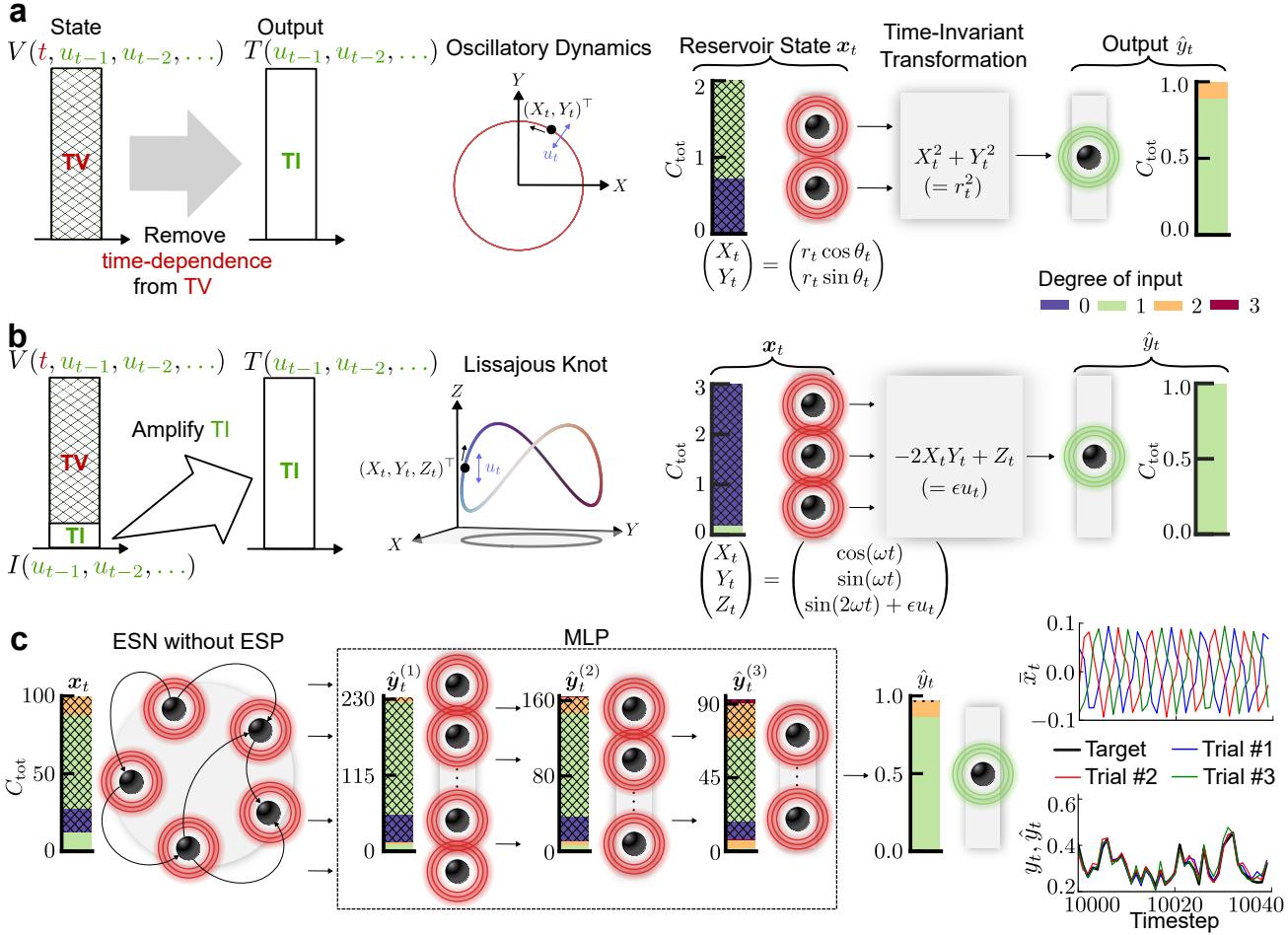


FIG. 2. **TI transformation of time-variant reservoirs.** **a, b**, Two analytical examples and **c**, a numerical transformation, are illustrated. **a** left, Removal of time-dependence from time-variant (TV, hatched bar) terms to form time-invariant (TI, non-hatched bar) terms. **a** middle, The trajectory of oscillatory dynamics, on which the state $(X_t, Y_t)^\top$ moves along a circular orbit with a fixed angular velocity and is perturbed by input u_t in the radial direction. **a** right, Time-dependent elements in the state are canceled out with coordinate transformation to form the TI output. The TIPC decomposition C_{tot} is depicted by color bars where the non-hatched and hatched bars represent the TI and TV capacities, respectively. The color represents the degree of input: 0 (purple), 1 (green), 2 (orange), and 3 (red). **b** left, Amplification of TI terms by removal of TV terms. **b** middle, The trajectory of the Lissajous knot, on which the state $(X_t, Y_t, Z_t)^\top$ shows the periodic orbit perturbed by the input u_t in the Z -direction. **b** right, The nonlinear readout enlarges the small TI term in the state by canceling out time-dependent functions (hatched purple). See Fig. S1 for further details of the TIPC decompositions in **a** and **b**. **c**, The numerical transformation using the ESN with TV states (its maximum conditional Lyapunov exponent was $\lambda_{\max} = 1.5 \times 10^{-2}$) and 4-layer MLP. The ripple color represents whether the node holds the ESP (all the node-wise ESP indices $\bar{d}_i < 0.3$, green) or not (red). The bar graphs represent the amount of TI and TV terms in the ESN and MLP layers. **c** right, Time series of the mean-field states \bar{x}_t and outputs \hat{y}_t with three different initial values (trial #1, blue; #2, red; #3, green) and target y_t (black). The normalized mean square errors between y_t and \hat{y}_t were 0.066 (#1), 0.068 (#2), and 0.071 (#3). Note that those with linear regression were 0.40 (#1), 0.40 (#2), and 0.41 (#3).

we adopted a multi-layer perceptron (MLP) with backpropagation for nonlinear readout. We utilized the echo state network (ESN) [4] with TV states to perform the NARMA10 benchmark task [34], whose required memories to solve are known [33]. Figure 2c depicts the network structure of the ESN with a 4-layer MLP and the TIPC decomposition for each layer. The ESN held both TI and TV inputs, which were converted via MLP layers to the TI output with linear and quadratic past inputs. The amounts of required terms are gradually tuned as an increase in layer id (See Supplementary Information S3 for further details). Furthermore, we solved the benchmark task with another ESN time series with different initial values. Although the three time series averaged over nodes showed totally different behaviors (Fig. 2c, right), both outputs \hat{y}_t emulated the target y_t well, indicating success in searching for the numerical transformation. These results suggest the possibility of using analytical or numerical transformation to generate the

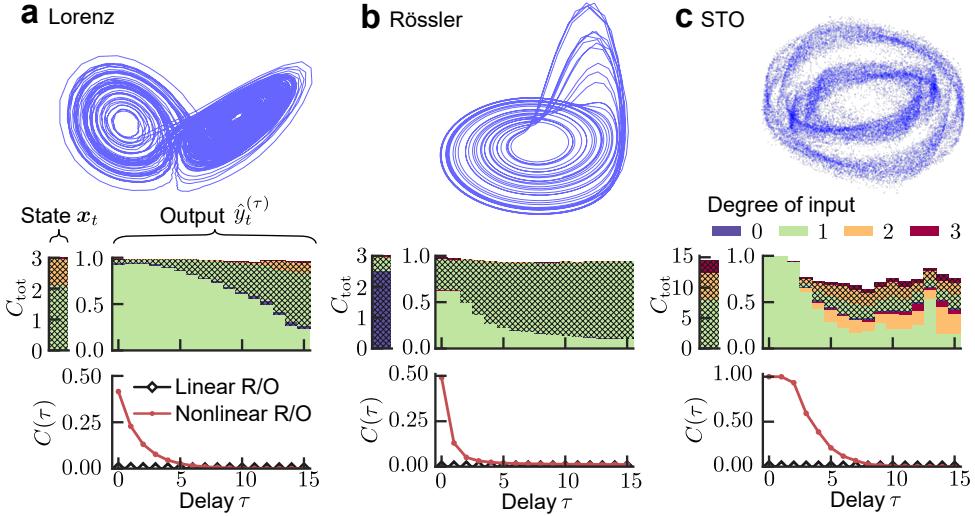


FIG. 3. Memory in systems without ESP. **a**, Lorenz model, **b**, Rössler model, and **c**, real STO. **a–c** illustrate a 3D-trajectory driven by input (upper), the TIPC decomposition of state and outputs (middle), and memory functions of a dynamical system without ESP (lower). **c** upper, The time-multiplexing technique was applied to the STO reservoir to make 100 virtual nodes, whose first three nodes were plotted. The TIPC decomposition of state x_t and outputs of a system that trained τ -delayed input $u_{t-\tau}$, using the same colors for the degree of input as in Fig. 2. The memory functions $C(\tau)$ with linear (black) and nonlinear readout (red). The memory capacities $MC = \sum_{\tau} C(\tau)$ with nonlinear readout are 0.97 (Lorenz), 0.97 (Rössler), and 4.3 (STO). See Fig. S2 for the relationship between the memory function and TIPC.

TI output from systems without ESP.

To show that our framework can leverage latent memory in simulated and physical systems without ESP, we calculated memory functions [35] with nonlinear readout. We adopted two chaotic systems (Lorenz and Rössler models) and a real spin-torque oscillator (STO) (Fig. 3, upper row), which receive a random input u_t and emulate τ -step delayed input $y_t^{(\tau)} = u_{t-\tau}$ by the nonlinear MLP readout (See Materials and Methods for detailed settings). As shown in Fig. 3, TI transformations can successfully extract past inputs from TV states in all three cases.

APPLICATIONS

Finally, we demonstrate that GRC enables us to construct an information processing system using a dynamical system without ESP. Using a high-dimensional atmospheric model of the Lorenz 96 system as a reservoir, which exhibits spatiotemporal chaos, we implemented three types of attractor embedding tasks. Figure 4a depicts the embedding task in which a reservoir emulates a target dynamical system. In the training phase, the reservoir receives the current target state \mathbf{u} to learn to predict the target at the next step. In the test phase, we remove the input and inject its output $\hat{\mathbf{u}}$ as the next input instead. First, to show that the GRC can extract past inputs from a TV state in a temporal task, we embedded the chaotic system of the Rössler model, which is described by three variables $\mathbf{u} = (u_1, u_2, u_3)^T$. In this demonstration, we inject u_1 into the Lorenz 96 reservoir for prediction of \mathbf{u} , which requires an input history of u_1 . After training, the Lorenz 96 exhibited chaos; however, it succeeded in reconstructing the rest of the variables (u_2, u_3) (Fig. 4b upper right, prediction in red; see Supplementary Information S4 and Video #2 for verification), indicating that the reservoir holds the history of u_1 . Second, to demonstrate that various types of target systems can be emulated by the chaotic system, we employed four periodic targets simultaneously. Figure 4c illustrates that four Lissajous curve-shaped trajectories $\hat{\mathbf{u}}$ with different frequencies were successfully embedded (yellow) in the Lorenz 96 reservoir. Even if the feedback signal is fixed at 0 during $8 \leq t \leq 16$ to perturb the Lorenz 96 state (green to blue), the output returns to the target trajectories (purple). Third, to further show the variation of targets in the GRC framework, we employed spatiotemporal chaos, which is the high-dimensional chaos representing complex behavior in nature (e.g., geophysical dynamics and fluid dynamics). We used the Kuramoto–Sivashinsky (KS) equation [37, 38], whose state \mathbf{u} is shown in Fig. 4d. In the test phase, the Lorenz 96 reservoir predicted \mathbf{u} and could keep a small error for over seven times the Lyapunov time of the KS model, implying that a similar model is embedded in the reservoir. These results suggest that the nonlinear readout allows us to extract memory from complex dynamical systems (e.g., chaos), which are available to solve various types of tasks.

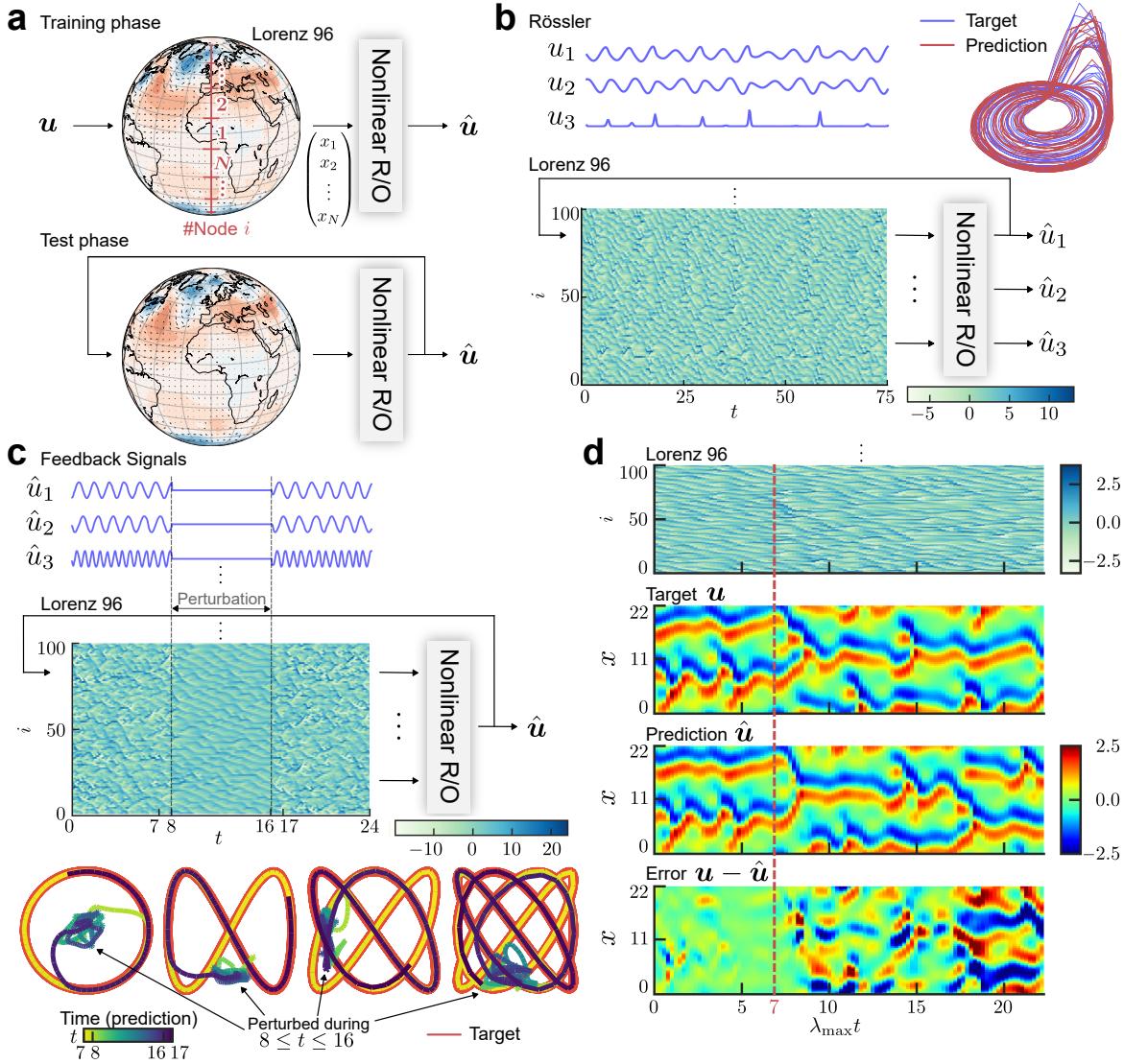


FIG. 4. Application of GRC to the reservoirs without ESP. **a**, Procedure of embedding task. Three target systems of **b**, Rössler model, **c**, Four Lissajous curves, and **d**, KS model were emulated by the Lorenz 96 model. **a**, In the training phase, the system receives the current target \mathbf{u} and predicts the target at the next step by training the nonlinear readout. In the test phase, the input is removed, and the output is fed back into the reservoir as input. **b-d** illustrate 100-dimensional time-series of Lorenz 96 out of 500–5120. **b**, Three variables (u_1, u_2, u_3) of the Rössler model (upper left) are the targets, and u_1 is the input (lower). The target (blue) and output (red) trajectories are plotted in the 3D space (upper right). After training, the Lorenz 96 exhibited chaos [the maximum conditional Lyapunov exponent in the training phase was $\lambda_{\max} = 1.18$, and the maximum Lyapunov exponent (MLE) in the test phase was $\lambda_{\max} = 1.53$]. **c** The reservoir state (middle) and output \mathbf{u} (upper) in $0 \leq t \leq 24$ are displayed. The feedback outputs were perturbed during $8 \leq t \leq 16$. In the test phase, the MLE of the Lorenz 96 reservoir was estimated to be $\lambda_{\max} = 3.5$. Note that the MLE of the embedded signal is estimated to be $\lambda_{\max} = 0.13$, implying that the embedded attractor is chaotic but has a similar shape as the original one [36]. **d** illustrates the target \mathbf{u} , the prediction $\hat{\mathbf{u}}$, and their error $\mathbf{u} - \hat{\mathbf{u}}$. The MLEs of the target and output were estimated to be $\lambda_{\max} = 0.13, 0.12$, respectively. Note that λ_{\max} in the horizontal axis represents the maximum Lyapunov exponent of the KS model and is the averaged time length where the initial value's error grows by a factor of e .

DISCUSSION

This paper extended the applicable range of RC to more general systems. Conventional RC imposes the ESP condition on the system state to effectively extract computational capabilities. The ESP of the state is required to obtain the ESP of output and strongly limits the type of dynamical systems, which leave a wide range of systems behind (e.g., periodic and chaotic dynamics as well as nonstationary ones [See Supplementary Information S6 for

further details]). This paper proposed a novel framework by changing the target of the ESP condition from the reservoir state to the final output and by introducing nonlinear readout with memory to utilize the systems without ESP. GRC is expected to effectively extract computational capabilities from many types of dynamics without ESP, including physical dynamics, such as spatiotemporal patterns in the brain composed of oscillatory spiking neurons and nonstationary synaptic plasticity.

The components in some types of computers (e.g., transistors in a traditional computer and artificial neurons in a conventional reservoir computer) show identical responses to identical inputs, while real neurons in a biological neural network seem to behave differently in every trial. However, living beings should generate meaningful outputs from these signals, which raises a challenge regarding information processing in their nervous systems. The proposed framework may provide a novel perspective on this problem. Even if the biological system receives an identical input, its neural network does not show identical responses, possibly due to its time dependence, although it certainly processes inputs [33]. These processed inputs can be transformed through the nonlinear responses of its neural circuit and body, which act as nonlinear readouts with memory, to eventually obtain meaningful outputs.

The use of alternative computing frameworks, such as neuromorphic computing, including PRC or a physical neural network, can take advantage of various properties of physics (e.g., durability and energy efficiency [23], computational speed [21, 39, 40], and robustness in extreme experiments, such as radio active ones [41]). GRC can also exert its power fully when implemented in a physical realm. For this purpose, we must develop a nonlinear readout using physical systems with a more refined architecture. For example, this can be composed using a physical deep neural network, in which a physical system is utilized as a neural network with training methods [42, 43]. In this paper, for the sake of functional verification of GRC, we adopted MLP with backpropagation. However, the readout selection is open to various types of functions. One approach is to specialize the readout dependent on the type of dynamics. For example, detrending for nonstationary systems and envelope extraction for periodic dynamics have already been successfully utilized in the post-processing of TV reservoirs, and we can consider these approaches as TI transformations (See Supplementary Information S6 for further details). Another strategy is to employ other nonlinear fitting schemes. For example, a dynamical system is also utilized as a readout that can realize TI transformation, holding past inputs, which naturally forms a deep reservoir architecture [44] and can be used for TI transformation. If deep physical reservoir architectures are introduced, some reservoirs can be considered to act as readouts of other connected reservoirs. This form of network is frequently found in nature. Its parameters can be tuned by a gradient-free learning scheme called augmented direct feedback alignment [43] if the precise model of physical reservoir is not obtained. Further studies on readout functions will open up more sophisticated computational systems.

The proposal of GRC paves the way to exploiting TV terms that have been missing to date. In the conventional RC, the reservoir state is required to be TI and can be utilized for computation with a linear or nonlinear readout. The computational capability retrievable with linear readout is limited [33, 45], whereas nonlinear readout eases this restriction. First, nonlinear readout can extract infinite past input. For instance, binary conversion can theoretically extract all the past inputs from a linear system with binary input [35]. Second, nonlinear readout can make any degree of nonlinearity on the input. If the readout has a universal approximation property [46], the past inputs can be converted into any function. In addition to the conventional properties, the mechanism of GRC revealed that the nonlinear readout allows us to utilize not only TI but also TV terms, which contribute strongly to a great variety of feasible outputs (Figs. 2 and 3).

MATERIALS AND METHODS

Memory Capacity Task

To evaluate the amount of past inputs held in dynamical systems, we performed the memory capacity task [35] using both linear and nonlinear readouts. We applied a uniform random input $u_t \in [-1, 1]$ and a binary random input $u_t \in \{-1, 1\}$ to the two chaotic models (Lorenz and Rössler models) and the STO, respectively, and set the target output to τ -delayed input $u_{t-\tau}$. Using the dynamical state \mathbf{x}_t , we trained the readout weight to form output \hat{y}_t such that the normalized mean-square error (NMSE) $\sum_{t=1}^{T_{\text{train}}}(y_t - \hat{y}_t)^2 / \sum_{t=1}^{T_{\text{train}}} y_t^2$ was minimized. The memory function was evaluated by the emulation error $C(\tau) = 1 - \sum_{t=T_{\text{train}}+1}^{T_{\text{train}}+T_{\text{test}}}(y_t - \hat{y}_t)^2 / \sum_{t=T_{\text{train}}}^{T_{\text{train}}+T_{\text{test}}} y_t^2$, and the memory capacity is $MC = \sum_{\tau} C(\tau)$.

Multi-Layer Perceptron

We employed multi-layer perceptron (MLP) to search for the TI transformation of the chaotic and physical systems. MLP uses the nonlinear activation function of the rectified linear unit (ReLU). We normalized the input \mathbf{x}_t to $\hat{\mathbf{x}}_t$, with zero mean and unit variance, and subsequently injected $\hat{\mathbf{x}}_t$ to the MLP. In the NARMA10 task and memory capacity task for the STO, the N_i -dimensional nodes $\hat{\mathbf{y}}_t^{(i)}$ in the i^{th} layer are described by

$$\hat{\mathbf{y}}_t^{(i)} = \begin{cases} \hat{\mathbf{x}}_t & (i=0) \\ \mathbf{R}(\mathbf{W}^{(i)}\hat{\mathbf{y}}_t^{(i-1)} + \mathbf{b}^{(i)}) & (i=1, \dots, L-1) \\ \mathbf{W}^{(L)}\hat{\mathbf{y}}_t^{(L-1)} + \mathbf{b}^{(L)} & (i=L) \end{cases}$$

where $\mathbf{W}^{(i)} \in \mathbb{R}^{N_i \times N_{i-1}}$ is the weight matrix and $\mathbf{b}^{(i)} \in \mathbb{R}^{N_i}$ is the bias vector; $\mathbf{R}(\mathbf{z}) = (R(z_1) \cdots R(z_N))^{\top}$ and $R(\cdot)$ is the ReLU activation function:

$$R(z_j) = \begin{cases} z_j & (z_j > 0) \\ 0 & (z_j \leq 0) \end{cases}.$$

We used four-layer MLPs ($L = 4$), and the numbers of nodes were $N_i \in [128, 512]$ ($i = 1, 2, 3$). In the memory capacity tasks of the Lorenz and Rössler models and attractor embedding tasks, we added skip connections for each layer, which worked as linear activation functions. The nodes are described by

$$\hat{\mathbf{y}}_t^{(i)} = \begin{cases} \hat{\mathbf{x}}_t & (i=0) \\ \mathbf{R}(\mathbf{W}^{(i)}\hat{\mathbf{y}}_t^{(i-1)} + \mathbf{b}^{(i)}) + \mathbf{W}_{\text{skip}}^{(i)}\hat{\mathbf{y}}_t^{(i-1)} + \mathbf{b}_{\text{skip}}^{(i)} & (i=1, \dots, L-1) \\ \mathbf{W}^{(L)}[\hat{\mathbf{y}}_t^{(L-1)\top} \cdots \hat{\mathbf{y}}_t^{(0)\top}]^{\top} + \mathbf{b}^{(L)} & (i=L) \end{cases}$$

where $\mathbf{W}_{\text{skip}}^{(i)} \in \mathbb{R}^{N_i \times N_{i-1}}$ is the weight matrix and $\mathbf{b}_{\text{skip}}^{(i)} \in \mathbb{R}^{N_i}$ is the bias vector. We used four- or five-layer MLPs ($L = 4, 5$), and the numbers of nodes were $N_i \in [64, 5120]$ ($i = 1, \dots, L-1$). In the training, the batch size was 128, the maximum epoch was 1,000, and weights with the best performance were selected. We used Adam optimizer with a learning rate of 0.001 and the loss function of the NMSE.

Memories in Systems without ESP

We calculated the memory functions of the Lorenz and Rössler models and the real STO (Fig. 3, upper row) with the nonlinear readout. We applied the random input u_t to the systems and emulated τ -step delayed input $y_t^{(\tau)} = u_{t-\tau}$ by the nonlinear MLP readout. The lower row in Fig. 3 illustrates the memory functions with nonlinear readout (red) as well as those with linear readout (black). The conventional linear readout cannot extract memory from these systems at all (the memory capacity $\text{MC} = \sum_{\tau} C(\tau) = 0$ for each system), while the nonlinear MLP readouts successfully recovered latent memories [$\text{MC} = 0.97$ (Lorenz), $\text{MC} = 0.97$ (Rössler), and $\text{MC} = 4.3$ (STO)]. The TIPC elaborates the processed inputs in state and output, revealing the amount and type of processed inputs (middle row in Fig. 3). In all three systems, the TIPC decomposition of state was composed of only TV terms. Note that the TIPCs were truncated with thresholds to remove numerical errors due to the finite length of the time series, implying that capacities smaller than the threshold potentially exist. Conversely, the decomposition of output includes TI terms. For each τ , the amount of first-order TI capacity (non-hatched green) was larger than the memory function with nonlinear readout, indicating that the output included not only $u_{t-\tau}$ but also inputs with another delay u_{t-s} ($s \neq \tau$) (See Fig. S2 for further details). These results imply that the MLP readouts numerically succeeded in removing time dependence from TV memories and/or in amplifying very small TI memories.

Prerequisite for RC

In this article, we employed the ESP to represent the condition of conventional RC. RC was proposed by integrating two types of recurrent neural networks: an ESN [4] and a liquid state machine (LSM) [5], which were independently developed but have similar prerequisites. The ESN imposes the network state on the ESP, in which the state is a function of only input history, as follows:

$$\mathbf{x}_t = \mathbf{h}(u_{t-1}, u_{t-2}, \dots).$$

The LSM requires that the state is approximated by a polynomial expansion of input history called the Volterra series. The state is expanded by polynomials of input history whose maximum degree is infinite, as follows:

$$\mathbf{x}_t = \mathbf{a} + \mathbf{b}_1 u_{t-1} + \mathbf{b}_2 u_{t-2} + \cdots + \mathbf{c}_1 u_{t-1}^2 + \mathbf{c}_2 u_{t-1} u_{t-2} + \cdots,$$

where $\mathbf{a}, \mathbf{b}_i, \mathbf{c}_i \in \mathbb{R}^N$ ($i = 1, 2, \dots$) are the coefficient vectors of zeroth-, first-, and second-order polynomials, respectively. Both prerequisites require that the state is independent of time and dependent on input history. To straightforwardly define the time-invariance of the state, the ESP was selected in this article.

A related concept to these prerequisites is the generalized synchronization [47], in which the dynamical state is synchronized with input. Under the circumstance, the state is a function of only input and thus is independent of time. The maximum conditional Lyapunov exponent (MCLE) is a measure to judge whether the state is synchronized with input or not. A negative MCLE means that the state is a function of only input and thus holds the ESP. Conversely, a positive MCLE implies that the state is not a function of only input and does not hold the ESP.

NARMA10 Task

To demonstrate the TI transformation with the nonlinear data fitting technique, we solved the NARMA10 benchmark task using the ESN [4] with TV states, where the spectral radius is controlled (set at 1.3) so that the ESP is broken down. The MCLE is estimated to be $\lambda_{\max} = 1.5 \times 10^{-2}$ (See Supplementary Information S3 for further details), which implies that the states did not hold the ESP [47]. The state $\mathbf{x}_t = (x_{1,t}, \dots, x_{N,t})^\top$ of the ESN is updated by

$$x_{i,t+1} = \tanh \left(\sum_{j=1}^N w_{ij} x_{j,t} + w_{\text{in},i} u_t \right), \quad (1)$$

where u_t denotes the uniform random input in the range of $[-1, 1]$; the input weights $w_{\text{in},i}$ were generated from a uniform random number in the range of $[-0.1, 0.1]$; the internal weight matrix $\mathbf{W} = [w_{ij}]$ was also generated from a uniform random number in $[-1, 1]$ and then was rescaled such that its spectral radius is equivalent to σ . In the results of Fig. 2c, we used $N = 100$ and $\sigma = 1.3$.

The 10th-order nonlinear autoregressive moving average (NARMA10) model [34] is described by

$$y_{t+1} = \alpha y_t + \beta y_t \sum_{k=0}^9 y_{t-k} + \gamma v_t v_{t-9} + \delta, \\ v_t = \sigma(u_t + 1)/2,$$

where y_t is the target output at the t^{th} step; u_t is the uniform random input in the range of $[-1, 1]$, which is injected into the reservoir, and v_t linearly converts the range of u_t to $[0, \sigma]$; and $(\alpha, \beta, \gamma, \delta, \sigma) = (0.3, 0.05, 1.5, 0.1, 0.45)$. Note that the target output y_t is a TI function mainly composed of linear past inputs $u_{t-\tau}$ ($\tau = 1, 2, 3, 10, 11, 12$) and quadratic past inputs $u_{t-\tau} u_{t-\tau-9}$ ($\tau = 1, 2, 3$) [33], which represent the input terms required to solve the task.

Attractor Embedding Task

The embedding tasks were performed with three types of target systems: the Rössler model, Lissajous curves, and the KS model. In the training phase, we fed the current state of the target system as input $\mathbf{u}(t)$ into the Lorenz 96 model and trained a readout weight to emulate the next state of target system $\mathbf{u}(t + \Delta t)$.

The Lorenz 96 model, which exhibits spatio-temporally chaotic behavior, was utilized as a reservoir to demonstrate the embedding tasks. The differential equation for the i^{th} state $x_i(t)$ is described by

$$\frac{dx_i}{dt} = (x_{i+1} - x_{i-2})x_{i-1} - x_i + \mu + \iota u_i \quad (i = 1, \dots, N),$$

where the states are cyclically ordered (i.e., $x_{-1} = x_{N-1}$, $x_0 = x_N$, and $x_{N+1} = x_1$); μ and ι are the bias and input intensity, respectively, and (μ, ι) were set to $(5, 5)$, $(8, 20)$, and $(5, 15)$ for the embedding tasks of the Rössler model, Lissajous curves, and the KS model, respectively; in the training phase of embedding task, $u_i(t)$ was the state of the target system, while in the test phase, $u_i(t)$ was the output of MLP fed back as input. In the training phase, the

MCLEs of the Lorenz 96 model with the target of Rössler model, Lissajous curves, and the KS model were estimated to be $\lambda_{\max} = 1.18, 3.67$, and 2.00 , respectively (See Supplementary Information S4 for further details). The positive MCLEs imply that the reservoir states do not hold the ESP [47].

We used different settings of input and target for each task. In the first task, the target Rössler model has the three-dimensional state (X, Y, Z) . We used only the one-dimensional state $u = X$ for input and target. In the second task with the Lissajous curves and the third task with the KS model, all of the five- and 64-dimensional states, respectively, were used as input and target.

Models

Periodic oscillator

The equation of the periodic oscillator is described on the polar coordinate. The radius r_t and angle θ_t are updated at each time step as follows:

$$\begin{aligned} r_{t+1} &= \rho r_t + \mu + \sigma u_t, \\ \theta_{t+1} &= \theta_t + \omega, \end{aligned} \tag{2}$$

where ρ is the time constant of relaxation, μ is the input bias, σ is the input intensity, and ω is the constant angular velocity of θ_t . Assuming that $r_0 = \theta_0 = 0$ and $t \rightarrow \infty$, the analytical solution of Eq. (2) is given by

$$\begin{aligned} r_t &= \frac{\mu}{1 - \rho} + \sigma \sum_{\tau=1}^{\infty} \rho^{\tau-1} u_{t-\tau}, \\ \theta_t &= \omega t. \end{aligned}$$

Therefore, we analytically obtained the TI radius $r_t = r(u_{t-1}, u_{t-2}, \dots)$ and the angle of function of time $\theta_t = \omega t$. The position $(X_t, Y_t) = (r_t \cos \theta_t, r_t \sin \theta_t)$ on the Cartesian coordinate is TV.

Lissajous knot

We demonstrated the second mechanism of the TI transformation using the Lissajous knot [48], which is a periodic system with multiple frequencies, as follows:

$$\begin{pmatrix} X_t \\ Y_t \\ Z_t \end{pmatrix} = \begin{pmatrix} \cos(\omega t) \\ \sin(\omega t) \\ \sin(2\omega t) + \epsilon u_t \end{pmatrix},$$

where u_t was the uniform random input, and the input intensity was set to $\epsilon = 10^{-3}$.

Lorenz model

We evaluated the memory capacity of the Lorenz model. The three states (X, Y, Z) are described by

$$\begin{aligned} \frac{dX}{dt} &= p(Y - X) + \iota u_t, \\ \frac{dY}{dt} &= -XZ + rX - Y, \\ \frac{dZ}{dt} &= XY - bZ, \end{aligned}$$

where $(p, b, r) = (10, 8/3, 28)$ are constant parameters, $u_t \in [-1, 1]$ is the uniform random input, and the input intensity was set to $\iota = 30$. We solved the equations using the fourth-order Runge-Kutta method with a step width of $\Delta t = 0.02$.

Rössler model

The Rössler system was used for memory capacity and the target of embedding tasks. The state variables (X, Y, Z) are updated by

$$\begin{aligned}\frac{dX}{dt} &= -Y - Z, \\ \frac{dY}{dt} &= X + aY, \\ \frac{dZ}{dt} &= b + XZ - cZ + \iota u_t,\end{aligned}$$

where the constant parameters were set to $(a, b, c) = (0.2, 0.2, 5.7)$; $u_t \in [-1, 1]$ is the uniform random input. The input intensity was set to $\iota = 0.2$ (0) for the memory capacity (embedding) task, and the equations were solved by the fourth-order Runge–Kutta method with a step width of $\Delta t = 0.1$ (0.2).

Lissajous curves

Four two-dimensional Lissajous curves are given by $(u_i(t), u_{i+1}(t))$ ($i = 1, 2, 3, 4$), where $u_i(t)$ is described by

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{pmatrix} = \begin{pmatrix} \cos(\omega t) \\ \sin(\omega t) \\ -\sin(2\omega t) \\ \sin(3\omega t) \\ -\sin(4\omega t) \end{pmatrix}.$$

Kuramoto–Sivashinsky model

The KS system is defined by the following partial differential equation for the state $u(x, t)$:

$$\frac{\partial u}{\partial t} + \frac{\partial^2 u}{\partial x^2} + \frac{\partial^4 u}{\partial x^4} + \frac{1}{2} \left(\frac{\partial u}{\partial x} \right)^2 = 0 \quad (3)$$

on a periodic domain $0 \leq x \leq L$ [i.e., $u(x, t) = u(x + L, t)$]. We evenly spanned the space to define the Q variables

$$\mathbf{u}(t) = (u(\Delta x, t), u(2\Delta x, t), \dots, u(Q\Delta x, t))^{\top}$$

with an interval of $\Delta x = L/Q$ and numerically solve Eq. (3) with a step width of $\Delta t = 0.25$. Note that $L = 22$ and $Q = 64$. Using the Rosenstein algorithm [49], we estimated the MLE of the KS model as $\lambda_{\max} = 0.13$.

Spin Torque Oscillator

STO is a device that converts nonlinear spin dynamics into electrical signals. We performed PRC on a system in which the device was fed back its own delayed signal. The device and feedback circuit were almost the same as those in a previous report [25]. The delay time of the circuit was about 29 ns, and the feedback gain was about 20 dB. The uniform random input u_t was injected through modulation of the driving voltage of the STO. The modulation amplitude and offset were 75 mV and 225 mV, respectively. The driving voltage was kept at 20 ns at each step of the input. Simultaneously, the STO signal was measured by an oscilloscope with a sampling rate of 5 Gsam/s. The measured voltage was treated as 100 virtual nodes, using the time-multiplexing method [14].

Temporal Information Processing Capacity

To reveal the TI and TV representations in the states and outputs, we adopted TIPC [20, 33], which comprehensively quantifies the amount of processed input in the system. A dynamical system with input updates the state through

state equation $\mathbf{x}_{t+1} = \mathbf{g}(\mathbf{x}_t, u_t)$, where $\mathbf{x}_t \in \mathbb{R}^N$ and $u_t \in \mathbb{R}$ are the state and input, respectively. We assume that the orthonormalized state $\hat{\mathbf{x}}_t \in \mathbb{R}^r$ ($r \leq N$) is a function of time and input history:

$$\hat{\mathbf{x}}_t = \mathbf{F}(t, u_{t-1}, u_{t-2}, \dots),$$

where r -normalized, linearly independent state $\hat{\mathbf{x}}_t$ is extracted from the N -dimensional state \mathbf{x}_t through singular value decomposition, and that the state can be completely expanded by orthonormal TI bases $I_i(u_{t-1}, u_{t-2}, \dots)$ and TV ones $V_i(t, u_{t-1}, u_{t-2}, \dots)$ as

$$\hat{\mathbf{x}}_t = \sum_i \mathbf{a}_i I_i(u_{t-1}, u_{t-2}, \dots) + \sum_i \mathbf{b}_i V_i(t, u_{t-1}, u_{t-2}, \dots),$$

where $\mathbf{a}_i, \mathbf{b}_i \in \mathbb{R}^r$ are coefficient vectors. I_i and V_i represent the function forms of processed inputs, whose amounts are calculated by their squared norms $C_i^{\text{TI}} = \|\mathbf{a}_i\|^2$ and $C_i^{\text{TV}} = \|\mathbf{b}_i\|^2$, respectively. The total capacity $C_{\text{tot}} = \sum_i C_i^{\text{TI}} + C_i^{\text{TV}}$ holds $C_{\text{tot}} \leq r$ (See Supplementary Information S1 for further details).

We used two types of orthonormal bases. First, we used the Legendre polynomial-chaos and sinusoidal terms [33] to derive the TIPC of the analytical solutions. Second, we adopted the Volterra-Wiener-Korenberg series [20, 50] as the orthonormal polynomial expansion for the TIPC of the numerical solutions.

To visually show the TI and TV representation, we used the TIPC decomposition, which sums up the TIPC for each degree n of input as follows:

$$C_{\text{tot},n}^{\text{TI}} = \sum_{\{i|d_i=n\}} C_i^{\text{TI}},$$

$$C_{\text{tot},n}^{\text{TV}} = \sum_{\{i|d_i=n\}} C_i^{\text{TV}},$$

where d_i denotes the degree of input in the i^{th} basis. We illustrated the TIPC decomposition using bar graphs, in which nonhatched and hatched areas show the amounts of TI and TV terms in $\hat{\mathbf{x}}_t$, respectively.

Lyapunov Exponent

We calculated the MCLE for ESN [Eq. (1)] based on the QR decomposition and the Jacobian matrix $\mathbf{J}_t = (\mathbf{I} - \text{diag}(\mathbf{x}_t \circ \mathbf{x}_t)) \cdot \mathbf{W}^\top$, where \circ represents the Hadamard product. For the t^{th} timestep, we calculated the QR decomposition $\mathbf{Q}_t \mathbf{R}_t = \mathbf{J}_t \mathbf{Q}_{t-1}$ to obtain the time series of \mathbf{R}_t , where \mathbf{Q}_t is the orthonormal matrix and \mathbf{R}_t is the upper triangular matrix. Subsequently, we calculated the Lyapunov exponent by $\lambda_i = \sum_{t=1}^T \ln |R_{ii,t}|/T$ ($i = 1, \dots, N$).

We also calculated the MCLE and MLE for the Lorenz 96 model using the Jacobian matrix (See Supplementary Information S4 for further details).

Node-Wise ESP Index

To find out whether each node of a dynamical system depends on an initial value or not, we calculated the node-wise ESP index. We ran the N -dimensional dynamical system twice using two different initial values. Let $x_{i,t}^{(1)}, x_{i,t}^{(2)}$ be the two system states at the t^{th} time step with node id $i (= 1, \dots, N)$. We defined the node-wise ESP index \bar{d}_i by the NMSE between $x_{i,t}^{(1)}$ and $x_{i,t}^{(2)}$ as follows:

$$\bar{d}_i = \frac{1}{\sigma_i^{(1)}} \sqrt{\frac{1}{T} \sum_{t=1}^T \left(x_{i,t}^{(1)} - x_{i,t}^{(2)} \right)^2},$$

where $\sigma_i^{(1)}$ is the standard deviation of $x_{i,t}^{(1)}$ during $t \in [1, T]$.

[1] R. H. Lee, E. A. Mulder, and J. B. Hopkins, Mechanical neural networks: Architected materials that learn behaviors, *Science Robotics* **7**, eabq7278 (2022).

- [2] A. Ross, N. Leroux, A. De Riz, D. Marković, D. Sanz-Hernández, J. Trastoy, P. Bortolotti, D. Querlioz, L. Martins, L. Benetti, *et al.*, Multilayer spintronic neural networks with radiofrequency connections, *Nature Nanotechnology* **18**, 1273 (2023).
- [3] Z. Xue, T. Zhou, Z. Xu, S. Yu, Q. Dai, and L. Fang, Fully forward mode training for optical neural networks, *Nature* **632**, 280 (2024).
- [4] H. Jaeger, 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).
- [5] W. Maass, T. Natschläger, and H. Markram, Real-time computing without stable states: A new framework for neural computation based on perturbations, *Neural computation* **14**, 2531 (2002).
- [6] K. Nakajima and I. Fischer, *Reservoir Computing* (Springer, 2021).
- [7] C. Kaspar, B. Ravoo, W. G. van der Wiel, S. Wegner, and W. Pernice, The rise of intelligent matter, *Nature* **594**, 345 (2021).
- [8] H. Yasuda, P. R. Buskohl, A. Gillman, T. D. Murphey, S. Stepney, R. A. Vaia, and J. R. Raney, Mechanical computing, *Nature* **598**, 39 (2021).
- [9] F. Cazettes, L. Mazzucato, M. Murakami, J. P. Morais, E. Augusto, A. Renart, and Z. F. Mainen, A reservoir of foraging decision variables in the mouse brain, *Nature neuroscience* , 1 (2023).
- [10] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, *nature* **521**, 436 (2015).
- [11] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, *et al.*, A million spiking-neuron integrated circuit with a scalable communication network and interface, *Science* **345**, 668 (2014).
- [12] M. Prezioso, F. Merrikh-Bayat, B. D. Hoskins, G. C. Adam, K. K. Likharev, and D. B. Strukov, Training and operation of an integrated neuromorphic network based on metal-oxide memristors, *Nature* **521**, 61 (2015).
- [13] K. Nakajima, Physical reservoir computing—an introductory perspective, *Japanese Journal of Applied Physics* **59**, 060501 (2020).
- [14] L. Appeltant, M. C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. R. Mirasso, and I. Fischer, Information processing using a single dynamical node as complex system, *Nature communications* **2**, 468 (2011).
- [15] C. Du, F. Cai, M. A. Zidan, W. Ma, S. H. Lee, and W. D. Lu, Reservoir computing using dynamic memristors for temporal information processing, *Nature communications* **8**, 2204 (2017).
- [16] Y. Zhong, J. Tang, X. Li, X. Liang, Z. Liu, Y. Li, Y. Xi, P. Yao, Z. Hao, B. Gao, *et al.*, A memristor-based analogue reservoir computing system for real-time and power-efficient signal processing, *Nature Electronics* **5**, 672 (2022).
- [17] K. Fujii and K. Nakajima, Harnessing disordered-ensemble quantum dynamics for machine learning, *Physical Review Applied* **8**, 024030 (2017).
- [18] S. Ghosh, K. Nakajima, T. Krisnanda, K. Fujii, and T. C. Liew, Quantum neuromorphic computing with reservoir computing networks, *Advanced Quantum Technologies* **4**, 2100053 (2021).
- [19] P. Mujal, R. Martínez-Peña, J. Nokkala, J. García-Beni, G. L. Giorgi, M. C. Soriano, and R. Zambrini, Opportunities in quantum reservoir computing and extreme learning machines, *Advanced Quantum Technologies* **4**, 2100027 (2021).
- [20] T. Kubota, Y. Suzuki, S. Kobayashi, Q. H. Tran, N. Yamamoto, and K. Nakajima, Temporal information processing induced by quantum noise, *Physical Review Research* **5**, 023057 (2023).
- [21] D. Brunner, M. C. Soriano, C. R. Mirasso, and I. Fischer, Parallel photonic information processing at gigabyte per second data rates using transient states, *Nature communications* **4**, 1364 (2013).
- [22] L. Larger, M. C. Soriano, D. Brunner, L. Appeltant, J. M. Gutiérrez, L. Pesquera, C. R. Mirasso, and I. Fischer, Photonic information processing beyond turing: an optoelectronic implementation of reservoir computing, *Optics express* **20**, 3241 (2012).
- [23] J. Torrejon, M. Riou, F. A. Araujo, S. Tsunegi, G. Khalsa, D. Querlioz, P. Bortolotti, V. Cros, K. Yakushiji, A. Fukushima, *et al.*, Neuromorphic computing with nanoscale spintronic oscillators, *Nature* **547**, 428 (2017).
- [24] S. Tsunegi, T. Taniguchi, K. Nakajima, S. Miwa, K. Yakushiji, A. Fukushima, S. Yuasa, and H. Kubota, Physical reservoir computing based on spin torque oscillator with forced synchronization, *Applied Physics Letters* **114** (2019).
- [25] S. Tsunegi, T. Kubota, A. Kamimaki, J. Grollier, V. Cros, K. Yakushiji, A. Fukushima, S. Yuasa, H. Kubota, K. Nakajima, *et al.*, Information processing capacity of spintronic oscillator, *Advanced Intelligent Systems* **5**, 2300175 (2023).
- [26] K. Nakajima, T. Li, H. Hauser, and R. Pfeifer, Exploiting short-term memory in soft body dynamics as a computational resource, *Journal of The Royal Society Interface* **11**, 20140437 (2014).
- [27] K. Nakajima, H. Hauser, T. Li, and R. Pfeifer, Information processing via physical soft body, *Scientific reports* **5**, 10487 (2015).
- [28] N. Akashi, Y. Kuniyoshi, T. Jo, M. Nishida, R. Sakurai, Y. Wakao, and K. Nakajima, Embedding bifurcations into pneumatic artificial muscle, *Advanced Science* , 2304402 (2024).
- [29] H. Cai, Z. Ao, C. Tian, Z. Wu, H. Liu, J. Tchieu, M. Gu, K. Mackie, and F. Guo, Brain organoid reservoir computing for artificial intelligence, *Nature Electronics* **6**, 1032 (2023).
- [30] T. Sumi, H. Yamamoto, Y. Katori, K. Ito, S. Moriya, T. Konno, S. Sato, and A. Hirano-Iwata, Biological neurons act as generalization filters in reservoir computing, *Proceedings of the National Academy of Sciences* **120**, e2217008120 (2023).
- [31] M. Ushio, K. Watanabe, Y. Fukuda, Y. Tokudome, and K. Nakajima, Computational capability of ecological dynamics, *Royal Society Open Science* **10**, 221614 (2023).
- [32] S. Boyd and L. Chua, Fading memory and the problem of approximating nonlinear operators with volterra series, *IEEE Transactions on circuits and systems* **32**, 1150 (1985).

- [33] T. Kubota, H. Takahashi, and K. Nakajima, Unifying framework for information processing in stochastically driven dynamical systems, *Physical Review Research* **3**, 043135 (2021).
- [34] A. F. Atiya and A. G. Parlos, New results on recurrent network training: unifying the algorithms and accelerating convergence, *IEEE transactions on neural networks* **11**, 697 (2000).
- [35] H. Jaeger, *Short term memory in echo state networks*, Vol. 5 (GMD-Forschungszentrum Informationstechnik, 2001).
- [36] T. Kabayama, Y. Kuniyoshi, K. Aihara, and K. Nakajima, Designing chaotic attractors: A semi-supervised approach, arXiv preprint arXiv:2407.09545 (2024).
- [37] Y. Kuramoto and T. Tsuzuki, Persistent propagation of concentration waves in dissipative media far from thermal equilibrium, *Progress of theoretical physics* **55**, 356 (1976).
- [38] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach, *Physical review letters* **120**, 024102 (2018).
- [39] K. Vandoorne, P. Mechet, T. Van Vaerenbergh, M. Fiers, G. Morthier, D. Verstraeten, B. Schrauwen, J. Dambre, and P. Bienstman, Experimental demonstration of reservoir computing on a silicon photonics chip, *Nature communications* **5**, 3541 (2014).
- [40] L. Larger, A. Baylón-Fuentes, R. Martinenghi, V. S. Udal'tsov, Y. K. Chembo, and M. Jacquot, High-speed photonic reservoir computing using a time-delay-based architecture: Million words per second classification, *Physical Review X* **7**, 011015 (2017).
- [41] N. Akashi, Y. Kuniyoshi, S. Tsunegi, T. Taniguchi, M. Nishida, R. Sakurai, Y. Wakao, K. Kawashima, and K. Nakajima, A coupled spintronics neuromorphic approach for high-performance reservoir computing, *Advanced Intelligent Systems* **4**, 2200123 (2022).
- [42] L. G. Wright, T. Onodera, M. M. Stein, T. Wang, D. T. Schachter, Z. Hu, and P. L. McMahon, Deep physical neural networks trained with backpropagation, *Nature* **601**, 549 (2022).
- [43] M. Nakajima, K. Inoue, K. Tanaka, Y. Kuniyoshi, T. Hashimoto, and K. Nakajima, Physical deep learning with biologically inspired training method: gradient-free approach for physical hardware, *Nature Communications* **13**, 7847 (2022).
- [44] C. Gallicchio and A. Micheli, *Deep reservoir computing. Reservoir Computing: Theory, Physical Implementations, and Applications* (Springer, 2021) pp. 77–95.
- [45] J. Dambre, D. Verstraeten, B. Schrauwen, and S. Massar, Information processing capacity of dynamical systems, *Scientific reports* **2**, 514 (2012).
- [46] K. Hornik, M. Stinchcombe, and H. White, Multilayer feedforward networks are universal approximators, *Neural networks* **2**, 359 (1989).
- [47] Z. Lu, B. R. Hunt, and E. Ott, Attractor reconstruction by machine learning, *Chaos: An Interdisciplinary Journal of Nonlinear Science* **28**, 061104 (2018).
- [48] M. Bogle, J. Hearst, V. Jones, and L. Stoilov, Lissajous knots., *Journal of Knot Theory & Its Ramifications* **3** (1994).
- [49] M. T. Rosenstein, J. J. Collins, and C. J. De Luca, A practical method for calculating largest lyapunov exponents from small data sets, *Physica D: Nonlinear Phenomena* **65**, 117 (1993).
- [50] M. J. Korenberg, Identifying nonlinear difference equation and functional expansion representations: the fast orthogonal algorithm, *Annals of biomedical engineering* **16**, 123 (1988).
- [51] <https://github.com/kubota0130/ipc>.
- [52] I. Shimada and T. Nagashima, A numerical approach to ergodic problem of dissipative dynamical systems, *Progress of theoretical physics* **61**, 1605 (1979).
- [53] C. R. Nelson and C. R. Plosser, Trends and random walks in macroeconomic time series: some evidence and implications, *Journal of monetary economics* **10**, 139 (1982).
- [54] M. Feldman, Hilbert transform in vibration analysis, *Mechanical systems and signal processing* **25**, 735 (2011).
- [55] J. Grollier, D. Querlioz, K. Camsari, K. Everschor-Sitte, S. Fukami, and M. D. Stiles, Neuromorphic spintronics, *Nature electronics* **3**, 360 (2020).

ACKNOWLEDGMENTS

This study is supported by JSPS KAKENHI Grant No. JP21KK0182 and JP23K18472, by JST CREST Grant No. JPMJCR2014, and by the Cross-ministerial Strategic Innovation Promotion Program (SIP) on the “Integrated Health Care System” Grant Number JPJ012425. Y. I. was supported by JSPS KAKENHI Grant No. JP23KJ0331.

DATA AVAILABILITY

The experimental data of the STO are available from the corresponding author on reasonable request.

CODE AVAILABILITY

The Python codes that were used for the analysis can be requested from the corresponding author.

SUPPLEMENTARY INFORMATION FOR RESERVOIR COMPUTING GENERALIZED

This supplementary information provides a detailed description of the analysis of computational capability and attractor embedding tasks and gives examples of time-invariant transformation.

S1. TEMPORAL INFORMATION PROCESSING CAPACITY

Temporal information processing capacity (TIPC) [33] is a measure that comprehensively quantifies the amount of processed inputs in a time-variant (TV) state.

Definition

TIPC was developed by extending the information processing capacity (IPC) [45], which is a computational measure that comprehensively evaluates processed inputs in a TI state. We consider a nonlinear dynamical system with input whose state equation is described by

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, u_t).$$

Under the condition that the state holds an ESP [4], the system state is a function of only input history:

$$\mathbf{x}_t = \mathbf{g}(u_{t-1}, u_{t-2}, \dots),$$

which holds delayed and nonlinearly transformed inputs. To quantify the amount of the processed inputs, we begin with the simplest case of memory capacity [35], which evaluates the amount of delayed inputs held in the state. We apply an independent and identically distributed (i.i.d.) random input u_t to the system and set a target output z_t to τ -delayed input $u_{t-\tau}$.

$$z_t = u_{t-\tau}.$$

The output \hat{z}_t is calculated by linear regression as follows:

$$\hat{z}_t = \hat{\mathbf{W}}\mathbf{x}_t, \quad \hat{\mathbf{W}} = \arg \min_{\mathbf{W}} \sum_t (z_t - \mathbf{W}\mathbf{x}_t)^2.$$

The memory function $C(\tau)$ is defined by the emulation accuracy of the past input as follows:

$$C(\tau) = 1 - \frac{\sum_t (z_t - \hat{z}_t)^2}{\sum_t z_t^2}, \quad (\text{S1})$$

where the second term on the right-hand side is the normalized mean square error (NMSE) between the target output z_t and output \hat{z}_t . If the state perfectly holds the past input $u_{t-\tau}$, the capacity is 1; if the state does not hold the input at all, the capacity is 0. Additionally, the memory capacity (MC) is defined by a sum of the memory function:

$$\text{MC} = \sum_{\tau} C(\tau).$$

Note that the input must be i.i.d. random because if the targets correlate with each other, $C(\tau)$ cannot represent the amount of past input held in the state. For example, if the state is represented by $x_t = u_{t-1}$ (i.e., the state holds only u_{t-1}), the memory function should be $C(\tau) = 1$ ($\tau = 1$) and $C(\tau) = 0$ ($\tau > 1$). However, if we apply non-i.i.d. input, such as sine wave input, we obtain a memory function $C(\tau) > 0$ ($\tau > 1$), which does not represent the input held in the state. Under the condition of i.i.d. input, the MC has an upper bound of rank r , i.e., $\text{MC} \leq r$, where r is the rank of the correlation matrix $\mathbf{X}^\top \mathbf{X}$.

The state of a nonlinear dynamical system can hold not only delayed inputs but also nonlinear ones, such as u_{t-1}^2 and $u_{t-1}u_{t-2}$. In the same manner as the memory capacity, we set the target to a nonlinear function of past inputs. Due to the same reason as the memory capacity, these targets must not be correlated with each other, i.e.,

$$\sum_t z_{i,t} z_{j,t} = 0 \quad (i \neq j). \quad (\text{S2})$$

To meet this condition, we can use an orthogonal polynomial as a target. For example, in the case of a uniform random number $u_t \in [-1, 1]$, the target polynomial is orthogonalized with the Legendre polynomial $P_n(u)$ [e.g., u_t , u_t^2 , and u_t^3 are transformed into $P_1(u_t) = u_t$, $P_2(u_t) = (3u_t^2 - 1)/2$, and $P_3(u_t) = (5u_t^3 - 3u_t)/2$, respectively]. These targets are univariate polynomials, while a multivariate polynomial (e.g., $u_{t-1}u_{t-2}, u_{t-1}^2u_{t-2}, u_{t-1}u_{t-2}u_{t-3}, \dots$) is orthogonalized by a product of univariate orthogonal polynomials $\prod_k P_{n_k}(u_{t-\tau_k})$. Using the orthogonal basis $z_{i,t}$, the IPC is defined by

$$C_i = 1 - \frac{\sum_t (z_{i,t} - \hat{z}_{i,t})^2}{\sum_t z_{i,t}^2}, \quad (\text{S3})$$

where the output is calculated by linear regression in the same manner as memory capacity. The total capacity is defined by

$$C_{\text{tot}} = \sum_i C_i, \quad (\text{S4})$$

which is also bounded by the rank r , i.e., $C_{\text{tot}} \leq r$. If the state is a function of only input history, the total capacity matches the rank. This property is called the completeness property. The source codes of IPC are available online [51], enabling users to easily evaluate the overall computational capabilities of the TI state.

The IPC connects to a Taylor-like polynomial expansion called the polynomial chaos expansion, in which the state is expanded by multivariate orthogonal polynomials of input. For example, in the case of the uniform random input, the orthogonal polynomial is the product of the Legendre polynomial, which is called Legendre polynomial chaos. The state is expanded by these bases, which are the same as the targets of IPC $z_{i,t}$. The state is expanded by the bases as follows:

$$\hat{x}_t = \sum_i c_i \hat{z}_{i,t}, \quad (\text{S5})$$

where the time series of $\mathbf{x}_t \in \mathbb{R}^N$ is orthonormalized to that of $\hat{\mathbf{x}}_t \in \mathbb{R}^r$ using the singular value decomposition. The state matrix $\mathbf{X} = [\mathbf{x}_1 \cdots \mathbf{x}_T]^\top \in \mathbb{R}^{T \times N}$ is decomposed into $\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^\top$, and $\mathbf{U} = [\hat{\mathbf{x}}_1 \cdots \hat{\mathbf{x}}_T]^\top \in \mathbb{R}^{T \times r}$ includes the orthonormal state vector $\hat{\mathbf{x}}_t$. Let the time series of $z_{i,t}$ be $\mathbf{z}_i = [z_{i,1} \cdots z_{i,T}]^\top$, and the time series of $z_{i,t}$ is normalized to that of $\hat{z}_{i,t}$ such that $\hat{z}_{i,t} = z_{i,t}/\|\mathbf{z}_i\|$. In Eq. (S5), the squared norm of the coefficient vector is equivalent to the IPC, as follows:

$$C_i = \|c_i\|^2,$$

which means that the magnitude of the coefficient in the expanded state represents the amount of processed input. Note that the target output in the temporal task should be a function of input history, i.e.,

$$y_t = h(u_{t-1}, u_{t-2}, \dots).$$

We can apply the IPC to the target and completely expand its polynomials of past input. The IPCs of state refer to the amount of processed input in the state, whereas IPCs of target refer to the required input terms to solve the task.

Even if the state is not TI, the inputs can be processed and held in the state. If the state is TV, i.e.,

$$\mathbf{x}_t = \mathbf{g}(t, u_{t-1}, u_{t-2}, \dots), \quad (\text{S6})$$

processed inputs are represented not only by input history $\{u_{t-1}, u_{t-2}, \dots\}$ but also by time t . To comprehensively find computational capabilities in the state, we must use not only TI targets but also TV ones. The TIPC is defined by the same equation as the IPC [Eq. (S3)]. If the targets span a complete orthogonal system for a TV system, and the state is completely expanded by the target bases, the total capacity reaches the rank in the same manner as the IPC.

In this article, we use two types of bases for the TIPC. The first is a combination of the Fourier series bases and the Legendre polynomial chaos. The bases are composed of three parts: the Legendre polynomial chaos $z_t = \prod_k P_{n_k}(u_{t-\tau_k})$, which are the orthogonal bases of only input history if the input is a uniform random number $u_t \in [-1, 1]$; the Fourier series bases $z_t = \cos(\omega_n t)$ and $z_t = \sin(\omega_n t)$ ($n = 1, 2, \dots$), which are time-dependent orthogonal bases; and the products of the Legendre polynomial chaos and Fourier series bases $z_t = \prod_k P_{n_k}(u_{t-\tau_k}) \cos(\omega_n t)$ and $z_t = \prod_k P_{n_k}(u_{t-\tau_k}) \sin(\omega_n t)$ ($n = 1, 2, \dots$), which are bases of both time and input history. All the bases satisfy the orthogonality as in Eq. (S2) [33] and work as targets of the TIPC. We use this type of basis to calculate the TIPC of analytical solutions, such as in the oscillatory dynamics and the Lissajous knot system.

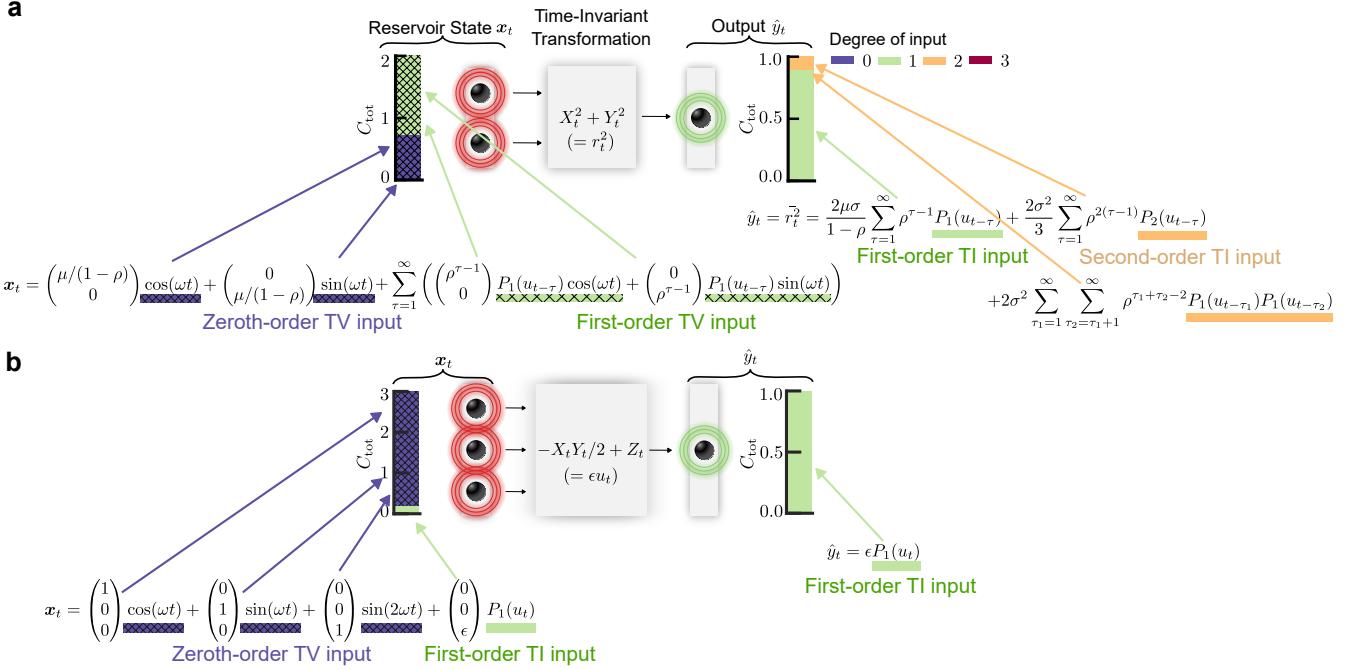


FIG. S1. **Explanation of TIPC with examples of TI transformation in Fig. 2.** The TI transformations of **a**, oscillatory dynamics and **b**, Lissajous knot converts the TV reservoir state x_t in Eqs. (S8) and (S10) into the TI output \hat{y}_t in Eqs. (S9) and (S11), respectively. The state and output are expanded by the TI and TV terms. After normalizing the state and orthogonal bases, the squared norm of coefficient vector represents the TIPC. The TIPCs are summarized as the TIPC decomposition C_{tot} , which is depicted by color bars, where the non-hatched and hatched bars represent the TI and TV capacities, respectively. The color of the TIPC represents the degree of input: 0 (purple), 1 (green), 2 (orange), and 3 (red). The orthogonal bases are underlined by the bar corresponding to that of the TIPC decomposition.

The second basis is the Volterra-Wiener-Korenberg (VWK) series [20, 50], which is an orthogonal expansion with polynomials of input and state histories. Assuming that the state is TV, as in Eq. (S6), the delayed state $x_{t-\tau}$ can also be regarded as a term that includes time-dependent elements. The VWK series expands the state by polynomials of past input and state and then orthogonalizes them using the Gram-Schmidt.

$$\begin{aligned} x_t &= \mathbf{a}_1^{(1)} u_{t-1} + \mathbf{a}_2^{(1)} u_{t-2} + \cdots + \mathbf{a}_1^{(2)} u_{t-1}^2 + \mathbf{a}_2^{(2)} u_{t-1} u_{t-2} + \cdots \\ &\quad + \mathbf{b}_1^{(1,1)} u_{t-1} x_{1,t-1} + \mathbf{b}_2^{(1,1)} u_{t-1} x_{2,t-1} + \cdots + \mathbf{b}_1^{(2,1)} u_{t-1}^2 x_{1,t-1} + \mathbf{b}_2^{(2,1)} u_{t-1} u_{t-2} x_{1,t-1} + \cdots \\ &\quad + \mathbf{c}_1^{(1)} x_{1,t-1} + \mathbf{c}_2^{(1)} x_{1,t-2} + \cdots + \mathbf{c}_1^{(2)} x_{1,t-1}^2 + \mathbf{c}_2^{(2)} x_{1,t-2}^2 + \cdots \\ &= \sum_i \gamma_i z_{i,t}, \end{aligned}$$

where the first to third rows and the fourth row represent expansions before and after orthogonalization, respectively; $\mathbf{a}_i^{(d)}$ is the coefficient vector of the i^{th} term of d^{th} -order input; $\mathbf{b}_i^{(d_1, d_2)}$ is the coefficient vector of the i^{th} product term of d_1^{th} -order input and d_2^{th} -order state; $\mathbf{c}_i^{(d)}$ is the coefficient vector of the i^{th} term of d^{th} -order state; $z_{i,t}$ represents the orthogonalized basis; and γ_i is the coefficient vector of $z_{i,t}$. These types of bases are used to calculate the TIPC of numerical solutions and time series data, such as the Lorenz model, the Rössler model, and real spin-torque oscillator (STO) data.

Derivation of TIPC of Oscillatory Dynamics

As shown in Fig. 2a of the main text, we calculate the TIPC of oscillatory dynamics, whose state equation is described by

$$\begin{aligned} r_{t+1} &= \rho r_t + \mu + \sigma u_t, \\ \theta_{t+1} &= \theta_t + \omega, \end{aligned} \quad (\text{S7})$$

where r_t and θ_t are the radius and angle on the polar coordinate, respectively, ρ (< 1) is the time constant of relaxation, μ is the input bias, σ is the input intensity, and ω is the constant angular velocity of θ_t . Assuming that $r_0 = \theta_0 = 0$ and $t \rightarrow \infty$, the analytical solution of Eq. (S7) is as follows:

$$\begin{aligned} r_t &= \frac{\mu}{1-\rho} + \sigma \sum_{\tau=1}^{\infty} \rho^{\tau-1} u_{t-\tau}, \\ \theta_t &= \omega t. \end{aligned}$$

The reservoir state on the Cartesian coordinate is

$$\mathbf{x}_t = \begin{pmatrix} X_t \\ Y_t \end{pmatrix} = \begin{pmatrix} r_t \cos \theta_t \\ r_t \sin \theta_t \end{pmatrix} = \left(\frac{\mu}{1-\rho} + \sum_{\tau=1}^{\infty} \rho^{\tau-1} u_{t-\tau} \right) \begin{pmatrix} \cos(\omega t) \\ \sin(\omega t) \end{pmatrix}.$$

Using the orthogonal bases, the state can be expanded as follows:

$$\mathbf{x}_t = \begin{pmatrix} \mu/(1-\rho) \\ 0 \end{pmatrix} \cos(\omega t) + \begin{pmatrix} 0 \\ \mu/(1-\rho) \end{pmatrix} \sin(\omega t) + \sum_{\tau=1}^{\infty} \left(\begin{pmatrix} \rho^{\tau-1} \\ 0 \end{pmatrix} P_1(u_{t-\tau}) \cos(\omega t) + \begin{pmatrix} 0 \\ \rho^{\tau-1} \end{pmatrix} P_1(u_{t-\tau}) \sin(\omega t) \right), \quad (\text{S8})$$

where $P_1(u_{t-\tau}) = u_{t-\tau}$ is the first-order Legendre polynomial. Letting $\mathbf{X} = (X_1 \cdots X_T)^{\top}$ and $\mathbf{Y} = (Y_1 \cdots Y_T)^{\top}$, we obtain

$$\|\mathbf{X}\| = \|\mathbf{Y}\| = \frac{1}{1-\rho} \sqrt{\frac{T}{6} \left(3\mu^2 + \frac{1-\rho}{1+\rho} \right)}.$$

The normalized state is expanded by the orthonormal bases as follows:

$$\begin{aligned} \hat{\mathbf{x}}_t &= \begin{pmatrix} X_t / \|\mathbf{X}\| \\ Y_t / \|\mathbf{Y}\| \end{pmatrix} = \frac{\mathbf{x}_t}{\|\mathbf{X}\|} \\ &= \begin{pmatrix} \frac{\mu \sqrt{T/2}}{(1-\rho) \|\mathbf{X}\|} \\ 0 \end{pmatrix} \frac{\cos(\omega t)}{\sqrt{T} \sigma_c} + \begin{pmatrix} 0 \\ \frac{\mu \sqrt{T/2}}{(1-\rho) \|\mathbf{X}\|} \end{pmatrix} \frac{\sin(\omega t)}{\sqrt{T} \sigma_s} + \sum_{\tau=1}^{\infty} \begin{pmatrix} \frac{\sqrt{T/6} \rho^{\tau-1}}{\|\mathbf{X}\|} \\ 0 \end{pmatrix} \frac{P_1(u_{t-\tau}) \cos(\omega t)}{\sqrt{T} \sigma_{uc}} \\ &\quad + \sum_{\tau=1}^{\infty} \begin{pmatrix} 0 \\ \frac{\sqrt{T/6} \rho^{\tau-1}}{\|\mathbf{X}\|} \end{pmatrix} \frac{P_1(u_{t-\tau}) \sin(\omega t)}{\sqrt{T} \sigma_{us}}. \end{aligned}$$

Therefore, the TIPCs for the TV bases $\cos(\omega t)$, $\sin(\omega t)$, $P_1(u_{t-\tau}) \cos(\omega t)$, and $P_1(u_{t-\tau}) \sin(\omega t)$ are given by

$$\begin{aligned} C_{\cos(\omega t)} &= C_{\sin(\omega t)} = \left(\frac{\sqrt{T/2} \mu}{(1-\rho) \|\mathbf{X}\|} \right)^2 = \frac{3\mu^2}{3\mu^2 + (1-\rho)/(1+\rho)}, \\ C_{P_1(u_{t-\tau}) \cos(\omega t)} &= C_{P_1(u_{t-\tau}) \sin(\omega t)} = \left(\frac{\sqrt{T/6} \rho^{\tau-1}}{\|\mathbf{X}\|} \right)^2 = \frac{\rho^{2(\tau-1)} (1-\rho)^2}{3\mu^2 + (1-\rho)/(1+\rho)}, \end{aligned}$$

respectively (Fig. S1a). Their total capacity is

$$C_{\text{tot}} = C_{\cos(\omega t)} + C_{\sin(\omega t)} + \sum_{\tau=1}^{\infty} (C_{P_1(u_{t-\tau}) \cos(\omega t)} + C_{P_1(u_{t-\tau}) \sin(\omega t)}) = 2,$$

which holds the completeness property. In the main text, the example of TI transformation for the periodic oscillator is

$$f(\mathbf{x}_t) = X_t^2 + Y_t^2 = r_t^2.$$

Expanding the output $\hat{y}_t = f(\mathbf{x}_t)$ by the orthonormal bases of Legendre polynomial chaos, we analytically derive its TIPC. The output is expanded by orthogonal linear and quadratic input terms as follows:

$$\begin{aligned} r_t^2 &= \left(\frac{\mu}{1-\rho} + \sigma \sum_{\tau=1}^{\infty} \rho^{\tau-1} u_{t-\tau} \right)^2 \\ &= \left(\frac{\mu}{1-\rho} \right)^2 + \frac{\sigma^2}{3(1-\rho^2)} + \frac{2\mu\sigma}{1-\rho} \sum_{\tau=1}^{\infty} \rho^{\tau-1} P_1(u_{t-\tau}) + \frac{2\sigma^2}{3} \sum_{\tau=1}^{\infty} \rho^{2(\tau-1)} P_2(u_{t-\tau}) \\ &\quad + 2\sigma^2 \sum_{\tau_1=1}^{\infty} \sum_{\tau_2=\tau_1+1}^{\infty} \rho^{\tau_1+\tau_2-2} P_1(u_{t-\tau_1}) P_1(u_{t-\tau_2}), \end{aligned}$$

where $P_1(u_{t-\tau}) = u_{t-\tau}$ and $P_2(u_{t-\tau}) = (3u_{t-\tau}^2 - 1)/2$ are the first- and second-order Legendre polynomials, respectively. We calculate the debiased output \bar{r}_t^2 by subtracting the constant terms from r_t^2 :

$$\bar{r}_t^2 = \frac{2\mu\sigma}{1-\rho} \sum_{\tau=1}^{\infty} \rho^{\tau-1} P_1(u_{t-\tau}) + \frac{2\sigma^2}{3} \sum_{\tau=1}^{\infty} \rho^{2(\tau-1)} P_2(u_{t-\tau}) + 2\sigma^2 \sum_{\tau_1=1}^{\infty} \sum_{\tau_2=\tau_1+1}^{\infty} \rho^{\tau_1+\tau_2-2} P_1(u_{t-\tau_1}) P_1(u_{t-\tau_2}). \quad (\text{S9})$$

Letting $\bar{\mathbf{r}}^2 = (\bar{r}_1^2 \cdots \bar{r}_T^2)^\top$, its norm is

$$\|\bar{\mathbf{r}}^2\| = \frac{2\sigma}{3} \sqrt{\frac{T}{1-\rho^2} \left(\frac{3\mu^2}{(1-\rho)^2} + \frac{\sigma^2}{5(1+\rho^2)} + \frac{\sigma^2\rho}{1-\rho} \right)}.$$

The normalized output is expanded by orthonormal bases as follows:

$$\begin{aligned} \frac{\bar{r}_t^2}{\|\bar{\mathbf{r}}^2\|} &= \frac{2\mu\sigma}{(1-\rho)\|\bar{\mathbf{r}}^2\|} \sqrt{\frac{T}{3} \sum_{\tau=1}^{\infty} \rho^{\tau-1} \frac{P_1(u_{t-\tau})}{\sqrt{T/3}}} + \frac{2\sigma^2}{3\|\bar{\mathbf{r}}^2\|} \sqrt{\frac{T}{5} \sum_{\tau=1}^{\infty} \rho^{2(\tau-1)} \frac{P_2(u_{t-\tau})}{\sqrt{T/5}}} \\ &\quad + \frac{2\sigma^2\sqrt{T}}{3\|\bar{\mathbf{r}}^2\|} \sum_{\tau_1=1}^{\infty} \sum_{\tau_2=\tau_1+1}^{\infty} \rho^{\tau_1+\tau_2-2} \frac{P_1(u_{t-\tau_1}) P_1(u_{t-\tau_2})}{\sqrt{T/3}}. \end{aligned}$$

The TIPCs for the bases $P_1(u_{t-\tau})$, $P_2(u_{t-\tau})$, and $P_1(u_{t-\tau_1}) P_1(u_{t-\tau_2})$ ($\tau_1 < \tau_2$) are given by

$$\begin{aligned} C_{P_1(u_{t-\tau})} &= \frac{3\mu^2(1-\rho^2)}{3\mu^2 + \sigma^2(1-\rho)^2/(5(1+\rho^2)) + \sigma^2\rho(1-\rho)} \rho^{2(\tau-1)}, \\ C_{P_2(u_{t-\tau})} &= \frac{\sigma^2(1-\rho^2)(1-\rho)^2/5}{3\mu^2 + \sigma^2(1-\rho)^2/(5(1+\rho^2)) + \sigma^2\rho(1-\rho)} \rho^{4(\tau-1)}, \\ C_{P_1(u_{t-\tau_1}) P_1(u_{t-\tau_2})} &= \frac{\sigma^2(1-\rho^2)(1-\rho)^2}{3\mu^2 + \sigma^2(1-\rho)^2/(5(1+\rho^2)) + \sigma^2\rho(1-\rho)} \rho^{2(\tau_1+\tau_2-2)}, \end{aligned}$$

respectively (Fig. S1a).

Derivation of TIPC of Lissajous Knot

As shown in Fig. 2b of the main text, we calculate the TIPC of the Lissajous knot, whose state is described by

$$\mathbf{x}_t = \begin{pmatrix} X_t \\ Y_t \\ Z_t \end{pmatrix} = \begin{pmatrix} \cos(\omega t) \\ \sin(\omega t) \\ \sin(2\omega t) + \epsilon u_t \end{pmatrix}.$$

The state is expanded by the orthogonal bases as follows:

$$\mathbf{x}_t = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \cos(\omega t) + \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \sin(\omega t) + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \sin(2\omega t) + \begin{pmatrix} 0 \\ 0 \\ \epsilon \end{pmatrix} P_1(u_t), \quad (\text{S10})$$

where $P_1(u_t) = u_t$ is the first-order Legendre polynomial. Let $\mathbf{X} = (X_1 \cdots X_T)^\top$, $\mathbf{Y} = (Y_1 \cdots Y_T)^\top$, and $\mathbf{Z} = (Z_1 \cdots Z_T)^\top$. Their norms are given by

$$\|\mathbf{X}\| = \|\mathbf{Y}\| = \sqrt{T/2}, \quad \|\mathbf{Z}\| = \sqrt{T(2\epsilon^2 + 3)/6}.$$

The normalized state is expanded by the orthonormal bases as follows:

$$\hat{\mathbf{x}}_t = \begin{pmatrix} X_t / \|\mathbf{X}\| \\ Y_t / \|\mathbf{Y}\| \\ Z_t / \|\mathbf{Z}\| \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \frac{\cos(\omega t)}{\sqrt{T/2}} + \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \frac{\sin(\omega t)}{\sqrt{T/2}} + \begin{pmatrix} 0 \\ 0 \\ \sqrt{3/(2\epsilon^2 + 3)} \end{pmatrix} \frac{\sin(2\omega t)}{\sqrt{T/2}} + \begin{pmatrix} 0 \\ 0 \\ \epsilon \sqrt{2/(2\epsilon^2 + 3)} \end{pmatrix} \frac{P_1(u_t)}{\sqrt{T/3}}$$

The TIPCs for the bases $\cos(\omega t)$, $\sin(\omega t)$, $\sin(2\omega t)$, and $P_1(u_t)$ are given by

$$C_{\cos(\omega t)} = C_{\sin(\omega t)} = 1, \quad C_{\sin(2\omega t)} = \frac{3}{2\epsilon^2 + 3}, \quad C_{P_1(u_t)} = \frac{2\epsilon^2}{2\epsilon^2 + 3},$$

respectively (Fig. S1b).

In the main text, the example of TI transformation for the Lissajous knot is

$$f(\mathbf{x}_t) = -2X_t Y_t + Z_t = \epsilon u_t.$$

The output is expanded by an orthogonal term, as follows:

$$f(\mathbf{x}_t) = \epsilon P_1(u_t), \quad (\text{S11})$$

where $P_1(u_t) = u_t$ is the first-order Legendre polynomial. The TIPC for the basis $P_1(u_t)$ is given by

$$C_{P_1(u_t)} = 1,$$

which is depicted in Fig. S1b.

Correspondence Between MC and TIPC

As shown in Fig. 3 in the main text, we demonstrate that, even if the reservoir holds only the TV terms, we can extract the TI terms using the nonlinear readout. We inject the input u_t into the TV reservoir and emulate the target $y_t^{(\tau)} = u_{t-\tau}$, whose accuracy is evaluated by the memory function $C(\tau)$, as in Eq. (S1) (bold red arrow). We also analyze the TIPC of the output (blue arrow), which represents the magnitude of the coefficient of the polynomial-expanded output. As shown in the upper-right position of Fig. S2, the TIPC decomposition C_{tot} includes both the TI and TV capacities. We extract the TI first-order capacities $C_1(s)$, which represent the amount of u_{t-s} in the output $\hat{y}_t^{(\tau)}$, and plot them as the forgetting curve (green arrow) in the middle-right position. We plot $C_1(s)$ with $\tau = 0, 14$, whose red point is $C_1(s)$ with $s = \tau$ and corresponds with the memory function $C(\tau)$ (thin red arrows). Since the non-hatched green bar in the TIPC decomposition is the sum of $C_1(s)$, it includes the memory function $C(\tau)$.

S2. NODE-WISE ECHO STATE PROPERTY

We consider a dynamical system with N -dimensional states $\mathbf{x}_t = (x_{1,t} \cdots x_{N,t})^\top \in \mathbb{R}^N$ that receives M -dimensional input $\mathbf{u}_t \in \mathbb{R}^M$, as follows:

$$\mathbf{x}_{t+1} = \mathbf{g}(\mathbf{x}_t, \mathbf{u}_t),$$

where $\mathbf{g} : \mathbb{R}^N \times \mathbb{R}^M \rightarrow \mathbb{R}^N$. We assume that the state is expanded by the TI function \mathbf{I} and TV function \mathbf{V} [33], as follows:

$$\mathbf{x}_t = \mathbf{I}(\mathbf{u}_{t-1}, \mathbf{u}_{t-2}, \dots) + \mathbf{V}(t, \mathbf{u}_{t-1}, \mathbf{u}_{t-2}, \dots),$$

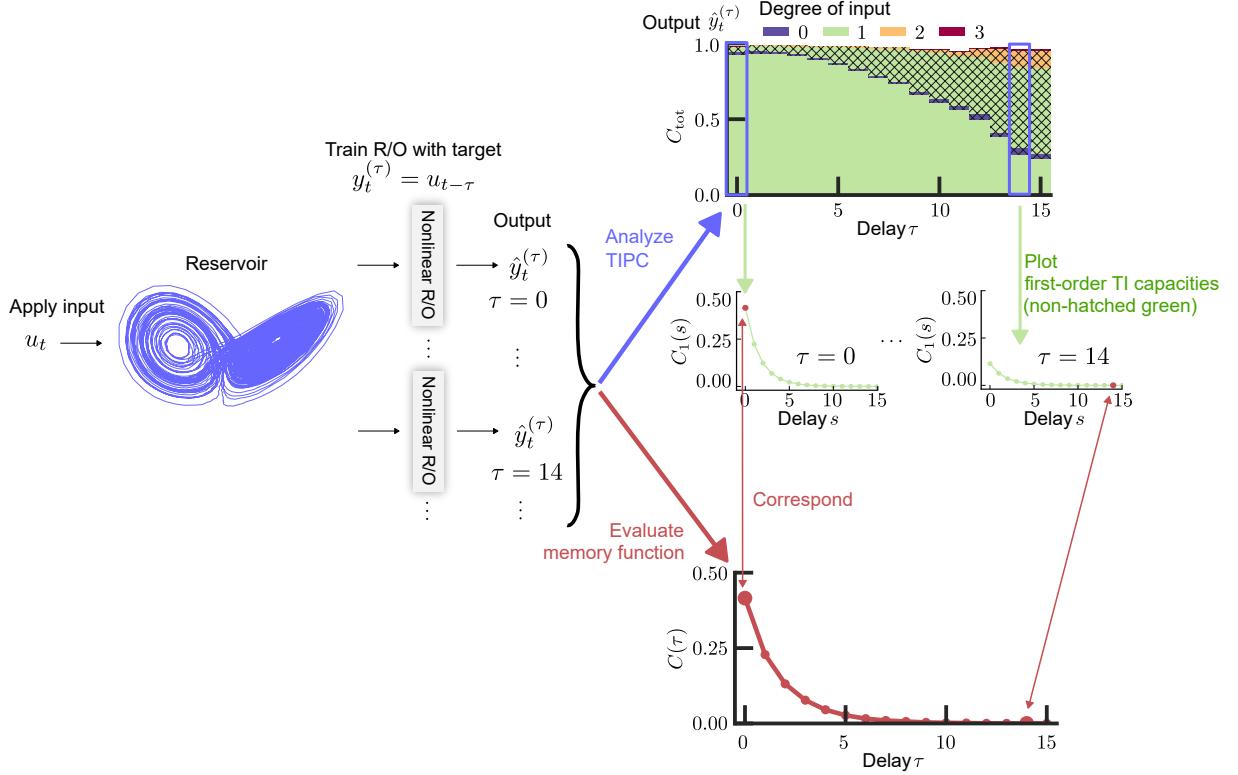


FIG. S2. Correspondence between MC and TIPC of output. A reservoir receives an input u_t , and targets $y_t^{(\tau)} = u_{t-\tau}$ ($\tau = 0, \dots, 15$) are emulated by nonlinear readout. The memory function is evaluated by $C(\tau) = 1 - \sum_t (y_t^{(\tau)} - \hat{y}_t^{(\tau)})^2 / \sum_t (y_t^{(\tau)})^2$ (bold red arrow). The outputs are also analyzed by the TIPC (blue arrow). The TIPC decomposition C_{tot} is depicted by color bars, where the non-hatched and hatched bars represent the TI and TV capacities, respectively. The color of the TIPC represents the degree of input: 0 (purple), 1 (green), 2 (orange), and 3 (red). The TIPC includes the TI first-order capacity $C_1(s)$, which is the capacity of u_{t-s} . Its sum is represented by the non-hatched green bar in the TIPC decomposition. In the middle-right position, $C_1(s)$ with $\tau = 0, 14$ is plotted. The TI first-order capacity $C_1(s)$ with $s = \tau$ corresponds with the memory function $C(\tau)$ (thin red arrows).

where $\mathbf{I} = (I_1 \cdots I_N)^\top$ and $\mathbf{V} = (V_1 \cdots V_N)^\top$. To obtain outputs with an echo state property (ESP), one or more nodes must have the ESP, under which the state is independent of time. We say that the i th state has a node-wise ESP if the following relation is held:

$$x_{i,t} = I_i(\mathbf{u}_t, \mathbf{u}_{t-1}, \dots).$$

Under the assumption that one or more nodes have node-wise ESP, linear regression can form time-invariant (TI) outputs:

$$\hat{\mathbf{y}}_t = \sum_{i \in S_I} \mathbf{w}_i x_{i,t} = \sum_{i \in S_I} \mathbf{w}_i I_i(\mathbf{u}_t, \mathbf{u}_{t-1}, \dots), \quad (\text{S12})$$

where S_I is an index set whose node is TI, and $\mathbf{w}_i (\neq \mathbf{0})$ is the weight vector for the i th node. If the node does not have node-wise ESP, the i th state is described by

$$x_{i,t} = I_i(\mathbf{u}_t, \mathbf{u}_{t-1}, \dots) + V_i(t, \mathbf{u}_t, \mathbf{u}_{t-1}, \dots).$$

If none of the states have node-wise ESP, the outputs are described by

$$\hat{\mathbf{y}}_t = \sum_{i \in S_V} \mathbf{w}_i x_{i,t} = \sum_{i \in S_V} \mathbf{w}_i I_i(\mathbf{u}_t, \mathbf{u}_{t-1}, \dots) + \sum_{i \in S_V} \mathbf{w}_i V_i(t, \mathbf{u}_t, \mathbf{u}_{t-1}, \dots). \quad (\text{S13})$$

where S_V is an index set whose node is TV, and $\mathbf{w}_i (\neq \mathbf{0})$ is the weight vector for the i th node. Even if none of the states have node-wise ESP, the output can be TI in a special case. Under the condition that the TV elements

V_i ($i \in S_V$) are linearly dependent from each other, i.e.,

$$V_i = \sum_{\substack{j \in S_V \\ j \neq i}} a_j V_j,$$

the output can form TI functions, as follows:

$$\hat{\mathbf{y}}_t = \mathbf{w} \left(x_{i,t} - \sum_{\substack{j \in S_V \\ j \neq i}} a_j x_{j,t} \right) = \mathbf{w} \left(I_i(\mathbf{u}_t, \mathbf{u}_{t-1}, \dots) - \sum_{\substack{j \in S_V \\ j \neq i}} a_j I_j(\mathbf{u}_t, \mathbf{u}_{t-1}, \dots) \right),$$

where $a_i \in \mathbb{R}$ and $\mathbf{w} \in \mathbb{R}^N$. Under the conditions that none of the states have node-wise ESP and the TV elements V_i are linearly independent from each other, the outputs in Eq. (S13) cannot form TI functions.

S3. ECHO STATE NETWORK WITH MULTI-LAYER PERCEPTRON

ESP Index and Conditional Lyapunov Exponent

In Fig. 2c of the main text, to examine whether nodes in each layer have ESP or not, we calculated the node-wise ESP index of the echo state network (ESN) with 4-layer multi-layer perceptron (MLP). The number of nodes are 100 (reservoir layer), 256 (MLP layer #1), 192 (MLP layer #2), 128 (MLP layer #3), and 1 (output layer). Figure S3a illustrates the distribution of indices for each layer. The averaged index $\langle \bar{d}_i \rangle$ (i.e., node-wise ESP index averaged over nodes in a layer) decreases with an increase in the MLP layer ID [$\langle \bar{d}_i \rangle = 0.96$ (reservoir), 1.42 (#1), 1.35 (#2), 1.24 (#3), and 0.287 (output)].

Next, to double-check whether the ESN is a function of input history or not, we evaluated the conditional Lyapunov exponent (CLE) of the reservoir layer. The state equation of the N -dimensional ESN is described by

$$x_{i,t+1} = \tanh \left(\sum_{j=1}^N W_{ij} x_{j,t} + w_{\text{in},i} u_t \right),$$

where $x_{i,t}$ is the i^{th} state ($i = 1, \dots, N$), u_t denotes the uniform random input in the range of $[-1, 1]$, the input weights $w_{\text{in},i}$ are generated from a uniform random number in the range of $[-0.1, 0.1]$, and the internal weight matrix $\mathbf{W} = (W_{ij})$ is also generated from a uniform random number in $[-1, 1]$ and then is rescaled such that its spectral radius is equivalent to σ . In this paper, we used $N = 100$ and $\sigma = 1.3$. The Jacobian matrix of ESN is given by

$$\mathbf{J}_t = \left(\frac{\partial x_{i,t+1}}{\partial x_{j,t}} \right) = (\mathbf{I} - \text{diag}(\mathbf{x}_{t+1} \circ \mathbf{x}_{t+1})) \cdot \mathbf{W}.$$

Using the QR decomposition, we calculate the orthonormal matrix \mathbf{Q}_t and the upper triangular matrix \mathbf{R}_t as follows:

$$\mathbf{J}_t \mathbf{Q}_{t-1} = \mathbf{Q}_t \mathbf{R}_t \quad (t = 1, 2, \dots).$$

Note that $\mathbf{J}_0 = \mathbf{Q}_0 \mathbf{R}_0$. Using the diagonal elements of \mathbf{R}_t , we calculate the Lyapunov exponent λ_i as follows:

$$\lambda_i = \frac{1}{T} \sum_{t=0}^{T-1} \ln |R_{t,ii}| \quad (i = 1, \dots, N).$$

As a result, we obtained the maximum conditional Lyapunov exponent (MCLE) $\lambda_{\max} = 1.5 \times 10^{-2}$. This positive exponent and the averaged node-wise ESP index $\langle \bar{d}_i \rangle = 1.7$ in the reservoir layer indicate that the ESN does not hold the ESP.

TIPC for Each Layer

To track the input terms transformed through the layers, we illustrated the first- and second-order capacities in each layer. The lower rightmost plot in Fig. S3b shows the TIPC decomposition of the NARMA10's target output,

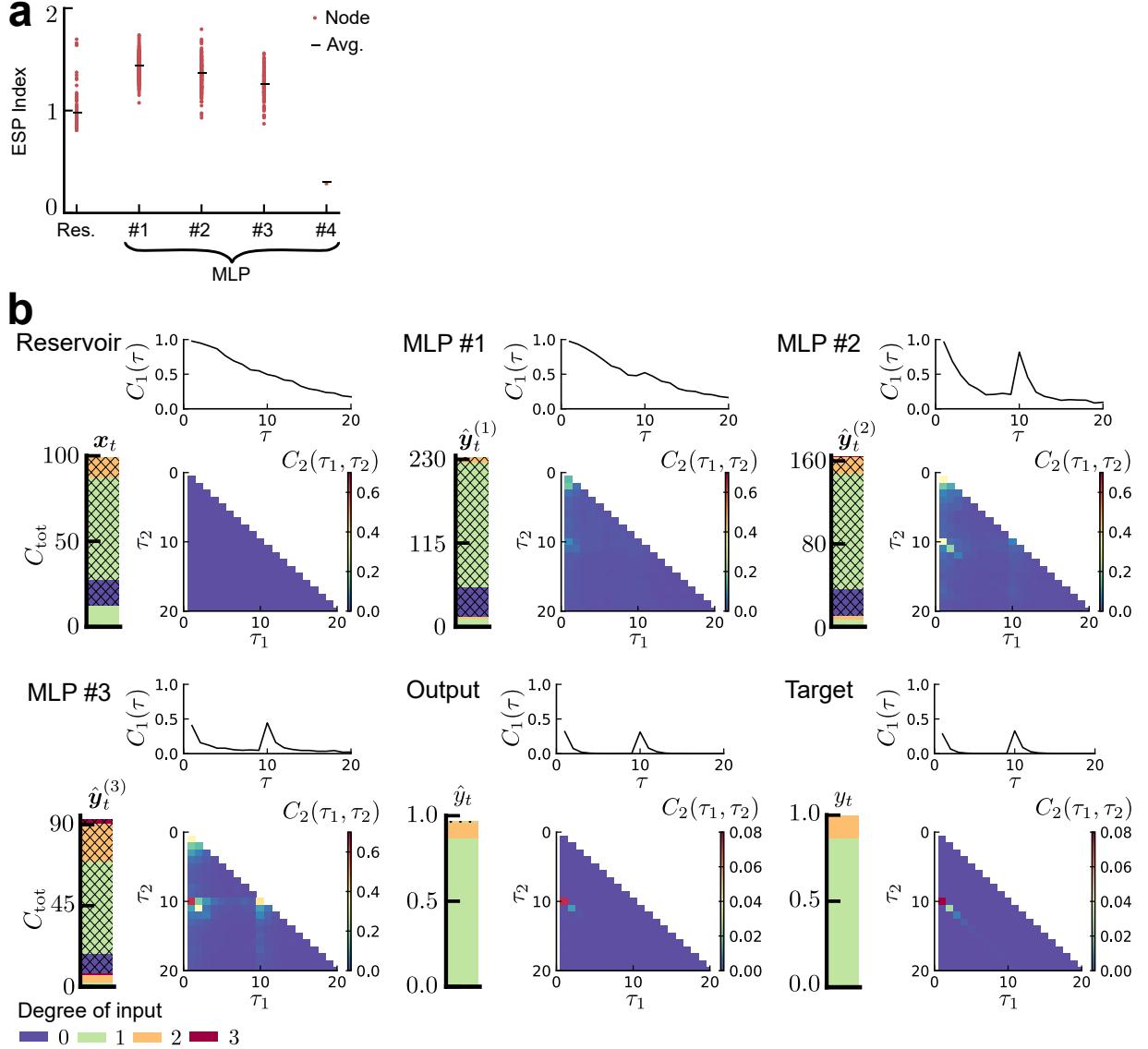


FIG. S3. **ESP Index and TIPC of ESN with MLP.** **a**, Node-wise ESP indices of ESN with 4-layer MLP. The vertical axis is the node-wise ESP index, and the horizontal axis is the layer name, in which “Res.” represents the reservoir layer. The numbers of nodes are (100, 256, 192, 128, 1) from Res. to #4. **b**, The TIPC decomposition (lower left) and its TI first-order capacity $C_1(\tau)$ (upper right; non-hatched green in the TIPC) and TI second-order capacity $C_2(\tau_1, \tau_2)$ (lower right; non-hatched orange in the TIPC) for each layer. $C_1(\tau)$ and $C_2(\tau_1, \tau_2)$ represent the capacities of orthogonalized $u_{t-\tau}$ and $u_{t-\tau_1} u_{t-\tau_2}$ ($\tau_1 \leq \tau_2$), respectively. The lower rightmost plots illustrate the capacities required to emulate the NARMA10 target. The TIPC decomposition C_{tot} is depicted by color bars, where the non-hatched and hatched bars represent the TI and TV capacities, respectively. The color of the TIPC represents the degree of input: 0 (purple), 1 (green), 2 (orange), and 3 (red).

which is composed of TI first- and second-order capacities. To solve the NARMA10 task, the output must include nine components [i.e., TI first-order terms $u_{t-\tau}$ ($\tau = 1, 2, 3, 10, 11, 12$) and second-order ones $u_{t-\tau} u_{t-\tau-9}$ ($\tau = 1, 2, 3$)] with an appropriate ratio [33]. To make these terms in the output layer, the MLP adjusts their amounts in each layer. The reservoir layer holds the TI first-order capacity $C_1(\tau)$. The MLP increases the required first-order terms with $\tau = 10, 11, 12$ in layers #1 and #2 and then adjusts the ratio of the first-order terms in layer #3 and the output layer. Conversely, the ESN does not have TI second-order terms at all. The MLP increases these terms in the layers #1–3 and adjusts their ratio in the output layer. The output layer does not sufficiently hold $u_{t-2} u_{t-11}$ and $u_{t-3} u_{t-12}$, which may deteriorate the NMSE score.

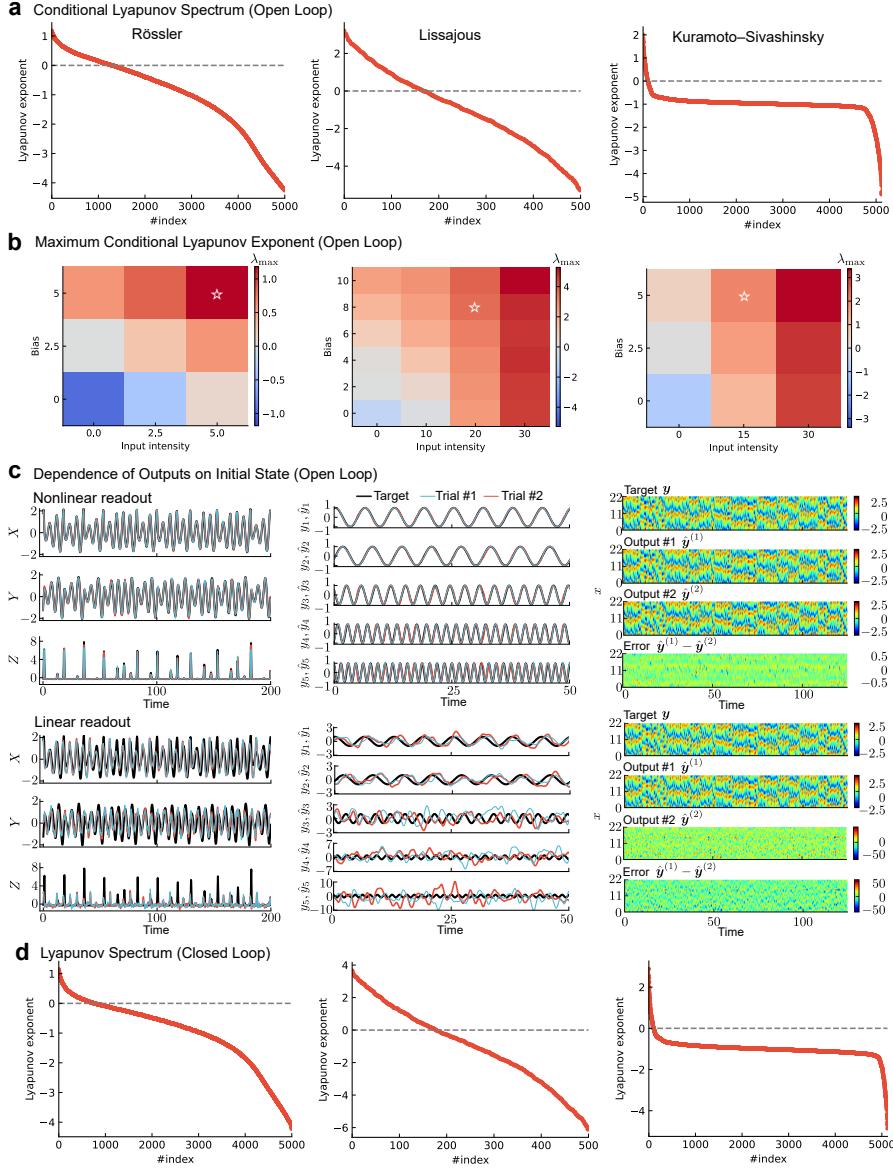


FIG. S4. Initial sensitivity of the Lorenz 96 reservoir. **a**, Conditional Lyapunov spectra, **b**, maximum conditional Lyapunov exponents, and **c**, time series of target and two outputs with different initial states of the open-loop Lorenz 96 reservoir with linear (lower) and nonlinear (upper) readouts, as well as **d**, Lyapunov spectra of the closed-loop Lorenz 96 reservoir with the target of the Rössler model, Lissajous curves, and the Kuramoto–Sivashinsky model. **a**, The horizontal axis is the index of nodes, and the vertical axis is the conditional Lyapunov exponent in the training phase. **b**, The horizontal axis is the input intensity ι , the vertical axis is the bias μ , and the color represents the maximum conditional Lyapunov exponent in the training phase. The star marks indicate the parameters used for the embedding tasks. **c**, In the case of the targets of the Rössler model and the Lissajous curves, the time series of target (black) and two outputs (blue and red) with different initial states of the Lorenz 96 model. In the case of the KS model, the target, two outputs, and error of the outputs, are depicted by colormaps. **d**, The horizontal axis is the index of nodes, and the vertical axis is the Lyapunov exponent in the test phase.

S4. ATTRACTOR ANALYSIS

Lorenz 96 Model

We calculated the CLE to confirm whether the Lorenz 96 reservoir was a function of only input history in the training phase (i.e., the Lorenz 96 reservoir is an open-loop system). As shown in Fig. 4 of the main text, we used three types of input: the Rössler model, Lissajous curves, and the Kuramoto–Sivashinsky (KS) model. The differential

equation of the Lorenz 96 model is described by

$$\frac{dx_i}{dt} = (x_{i+1} - x_{i-2})x_{i-1} - x_i + \mu + \iota u_i(t) \quad (i = 1, \dots, N),$$

where μ and ι represent the bias and input intensity, respectively. We calculated the conditional Lyapunov spectra using the algorithm described in [52]. The Jacobian matrix of the Lorenz 96 model is given by

$$\mathbf{J} = \left(\frac{\partial \dot{x}_i}{\partial x_j} \right) = \begin{pmatrix} -1 & x_N & 0 & \cdots & 0 & -x_N & x_2 - x_{N-1} \\ x_3 - x_N & -1 & x_1 & & 0 & 0 & -x_1 \\ \vdots & & & \ddots & & & \vdots \\ x_{N-1} & 0 & 0 & \cdots & -x_{N-1} & x_1 - x_{N-2} & -1 \end{pmatrix}.$$

We define N -orthonormal vectors $\delta \mathbf{x}_i(t) \in \mathbb{R}^N$ ($i = 1, \dots, N$) and a matrix $\delta \mathbf{X}(t) = [\delta \mathbf{x}_1(t) \cdots \delta \mathbf{x}_N(t)]$. $\delta \mathbf{X}(t)$ is updated through the following steps. First, we numerically solve the following differential equation using the fourth-order Runge-Kutta method:

$$\frac{d(\delta \mathbf{X})}{dt} = \mathbf{J} \delta \mathbf{X}$$

to obtain the matrix at the next timestep $\delta \mathbf{X}(t + \Delta t)$. Second, we orthonormalize the matrix using the QR decomposition. The matrix $\delta \mathbf{X}(t + \Delta t)$ is decomposed into

$$\delta \mathbf{X}(t + \Delta t) = \mathbf{Q}(t + \Delta t) \mathbf{R}(t + \Delta t),$$

where \mathbf{Q} is the orthonormal matrix and \mathbf{R} is the upper triangular matrix. $\delta \mathbf{X}$ is replaced by \mathbf{Q} as follows:

$$\delta \mathbf{X}(t + \Delta t) \leftarrow \mathbf{Q}(t + \Delta t).$$

Note that we set the matrix at the initial timestep to the identity matrix $\delta \mathbf{X}(0) = \mathbf{I}$. By repeating this procedure, we obtained $\{\mathbf{R}(\Delta t), \mathbf{R}(2\Delta t), \dots, \mathbf{R}(M\Delta t)\}$. The conditional Lyapunov exponent λ_j is calculated with diagonal elements of $\mathbf{R}(t)$ as follows:

$$\lambda_i = \frac{1}{M} \sum_{m=1}^M \ln |R_{ii}(m\Delta t)| \quad (i = 1, \dots, N).$$

Figure S4a shows the conditional Lyapunov spectra of the Lorenz 96 model in the three embedding tasks. The MCLEs of the Lorenz 96 model with the target of Rössler model, Lissajous curves, and the KS model were estimated to be $\lambda_{\max} = 1.18, 3.67$, and 2.00 , respectively. In all cases, the MCLEs were positive, indicating that the Lorenz 96 systems were not functions of only input history in the training phase. The MCLE of the Lorenz 96 model can be negative if we use different parameters, such as $(\mu, \iota) = (0, 0)$, and we selected the parameters with a positive MCLE for the embedding tasks (Fig. S4b).

We also calculate a global ESP index to confirm the ESP of output in the open-loop Lorenz 96 reservoir. We run the Lorenz 96 reservoir with two different initial states $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}$ and calculate outputs using both linear and nonlinear readouts to show that the output with nonlinear readout has an ESP. Let the M -dimensional target and outputs with the two initial states be $\mathbf{y} = (y_1 \cdots y_M)^\top$ and $\hat{\mathbf{y}}^{(i)} = (\hat{y}_1^{(i)} \cdots \hat{y}_M^{(i)})^\top$ ($i = 1, 2$), respectively. The global ESP index is defined by

$$d = \frac{\sum_{i=1}^M \text{MSE}(\hat{y}_i^{(1)}, \hat{y}_i^{(2)})}{\sum_{i=1}^M \text{Var}(y_i)} = \frac{\sum_{i=1}^M \sum_{t=1}^T (\hat{y}_{i,t}^{(1)} - \hat{y}_{i,t}^{(2)})^2}{\sum_{i=1}^M \sum_{t=1}^T (y_{i,t} - \bar{y}_i)^2},$$

where $\text{MSE}(\cdot, \cdot)$ and $\text{Var}(\cdot)$ represent the mean square error between the two time series and the variance of the time series, respectively, and $\bar{y}_i = \sum_{t=1}^T y_{i,t}/T$ is the time average of $y_{i,t}$. Figure S4c shows two output time series with the target of the Rössler model, Lissajous curves, and the KS model, in which the two outputs with nonlinear readout are consistent with the target, but those with linear readout have larger errors than the nonlinear cases. To quantify these errors, we calculate 10 indices with 10 different initial states of the Lorenz 96 model to average them. The averaged global ESP indices (mean \pm standard deviation) with nonlinear readout are $(1.53 \pm 0.01) \times 10^{-2}$ (Rössler model), $(5.95 \pm 0.54) \times 10^{-2}$ (Lissajous curves), and $(6.87 \pm 0.13) \times 10^{-2}$ (KS model), while those with linear readout are

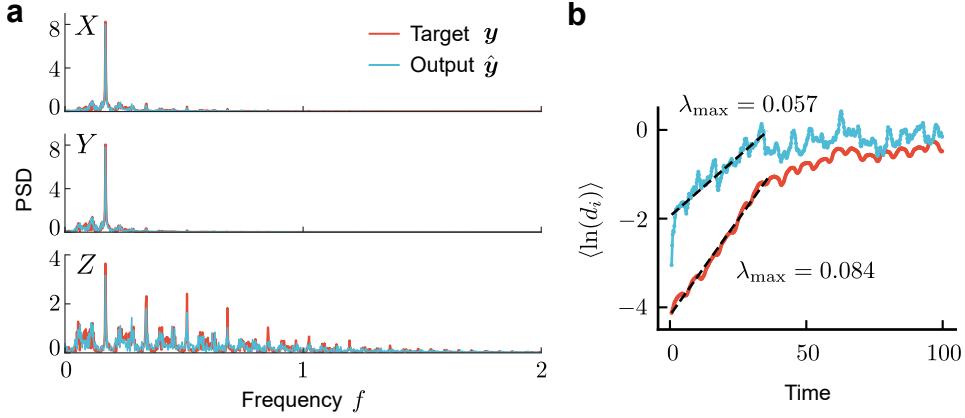


FIG. S5. **Attractor analysis in the embedding task of the Rössler model.** **a**, Power spectral density of (X, Y, Z) and **b**, time series of distance between two states beginning from a neighbor point. The target y and output \hat{y} are plotted in red and blue, respectively.

0.409 ± 0.007 (Rössler model), 1.92 ± 0.19 (Lissajous curves), and 0.246 ± 0.003 (KS model). These small indices with nonlinear readouts imply that the TI transformation successfully makes the output function of input history.

Finally, we calculate the MLEs of the Lorenz 96 model in the test phase to verify that the closed-loop Lorenz 96 reservoir is chaotic. As shown in Fig. S4d, the MLEs of the Lorenz 96 model with the target of Rössler model, Lissajous curves, and the KS model are $\lambda_{\max} = 1.15, 3.50$, and 2.88 , respectively. The positive exponents indicate that the Lorenz 96 reservoir is still chaotic with the feedback signal instead of the teacher forcing signal.

Rössler Model

As shown in Fig. 4 of the main text, we perform the three attractor embedding tasks to demonstrate that a system without ESP can be utilized as a computational resource by the nonlinear readout. First, we embed the Rössler attractor in the Lorenz 96 reservoir. To evaluate correspondence between the original and embedded attractors, we calculate their power spectral density (PSD) and maximum Lyapunov exponent.

To estimate the maximum Lyapunov exponent from the time series, we adopt the Rosenstein algorithm [49]. First, we find pairs of time $\{t_{i,1}, t_{i,2}\}$ ($i = 1, \dots, M$) for the nearest neighbors, such as $\|\hat{y}(t_{i,1}) - \hat{y}(t_{i,2})\| < \epsilon$. We calculate the time evolution of distance between $\|\hat{y}(t_{i,1}) - \hat{y}(t_{i,2})\|$ for each pair as follows:

$$d_i(k) = \|\mathbf{x}(t_{i,1} + k\Delta t) - \mathbf{x}(t_{i,2} + k\Delta t)\|_2.$$

We average the logarithm of distance over pairs as follows:

$$\langle \ln d_i(k) \rangle = \frac{1}{M} \sum_{i=1}^M \ln d_i(k).$$

We draw this time series and approximate its linear part by a linear regression whose slope represents the maximum Lyapunov exponent λ_{\max} .

Figure S5a shows the PSDs of the target and output, which match well. The maximum Lyapunov exponents of the output and target are different (the exponent of the output is $\lambda_{\max} = 0.057$; that of the target is $\lambda_{\max} = 0.084$). Therefore, the Lorenz 96 reservoir did not embed the attractor with the exact same properties as the original one.

Lissajous Curves

Next, we perform the attractor embedding task of the Lissajous curves with the Lorenz 96 reservoir. In the main text, comparing the shapes of the embedded attractors with those of the original ones, we show that the Lorenz 96 reservoir can embed the Lissajous curves. We also confirm that these attractors are stably embedded by disturbing the feedback signals. These embedded attractors do not hold original properties on a long-term basis. Figure S6a shows that the long time series of the target and output do not perfectly match. As shown in Fig. S6b, this difference

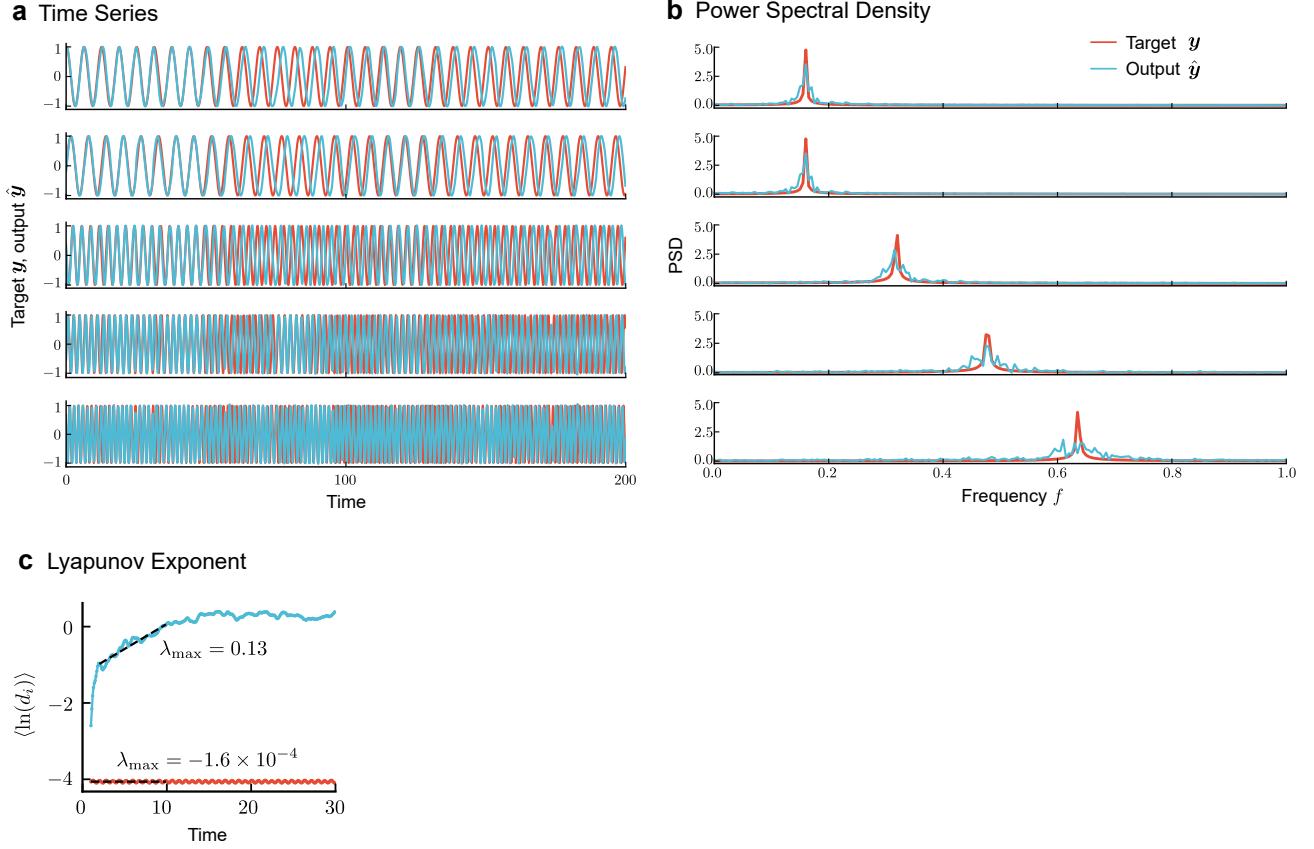


FIG. S6. **Attractor analysis in the embedding task of the Lissajous curves.** **a**, Time series, **b**, power spectral density of the Lissajous curves, and **c**, time series of distance between two states beginning from a neighbor point. The target \mathbf{y} and output $\hat{\mathbf{y}}$ are plotted in red and blue, respectively.

can be found by the PSDs of output, which are distributed around the peak of the target PSD but have a different distribution. The maximum Lyapunov exponent of the target is 0, but that of the output is positive ($\lambda_{\max} = 0.13$, Fig. S6c). This outcome is similar to the designed periodic chaos recently reported in [36].

Kuramoto–Sivashinsky Model

Finally, we perform the attractor embedding task of the KS model using the Lorenz 96 reservoir. The KS system is defined by the following partial differential equation for the state $y(x, t)$:

$$\frac{\partial y}{\partial t} + \frac{\partial^2 y}{\partial x^2} + \frac{\partial^4 y}{\partial x^4} + \frac{1}{2} \left(\frac{\partial y}{\partial x} \right)^2 = 0 \quad (\text{S14})$$

on a periodic domain $0 \leq x \leq L$ [i.e., $u(x, t) = u(x + L, t)$] with $L = 22$. We evenly span the space to define the $Q (= 64)$ variables

$$\mathbf{y}(t) = (y(\Delta x, t), y(2\Delta x, t), \dots, y(Q\Delta x, t))^{\top}$$

with an interval of $\Delta x = L/Q$. As shown in Fig. S7a, the output $\hat{\mathbf{y}}$ shows a similar spatiotemporal pattern to the target time series. To evaluate their correspondence, we calculate their PSDs for each position (Fig. S7b). Since Eq. (S14) is common for the position and the boundary is cyclic, we calculate the PSD averaged over position. The averaged PSD of the target and output match well. Furthermore, we evaluate the maximum Lyapunov exponents of target \mathbf{y} and output $\hat{\mathbf{y}}$ using the Rosenstein algorithm. As shown in Fig. S7c, the estimated maximum Lyapunov exponents for the target and output are $\lambda_{\max} = 0.12$ and $\lambda_{\max} = 0.13$, respectively, showing good agreement.

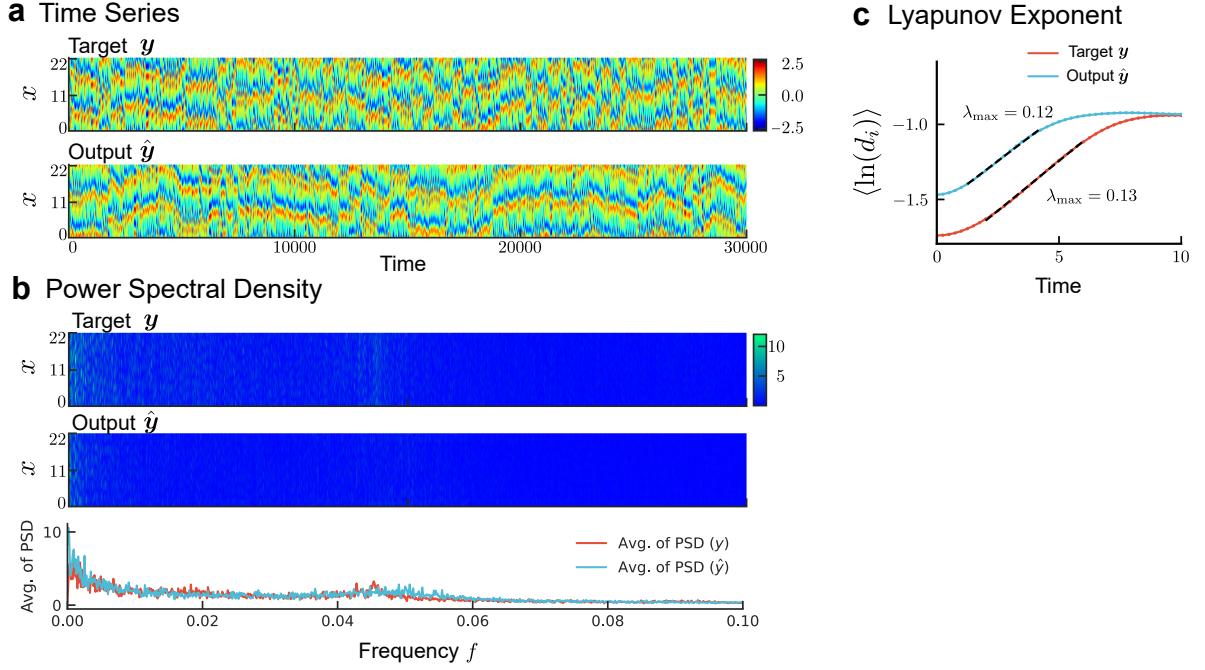


FIG. S7. **Attractor analysis in the embedding task of the Kuramoto–Sivashinsky model.** **a**, Time series, **b**, power spectral densities, and **c**, time series of distance between two states beginning from a neighbor point.

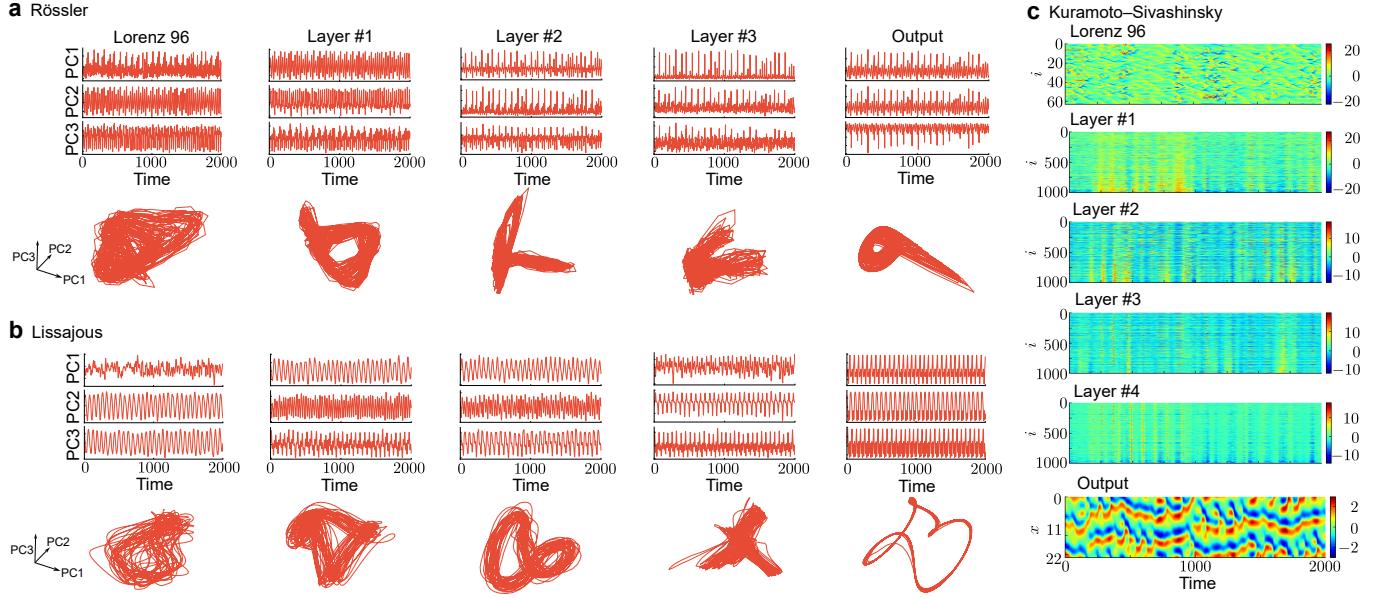


FIG. S8. **Attractor transition through the MLP layers.** Time series (upper) and three-dimensional plot (lower) of **a**, the Rössler model and **b**, Lissajous curves. Time series of the **c**, Kuramoto–Sivashinsky model. **a**, **b**, First three principal components (PC1, PC2, PC3) were plotted. **c**, Only the first 64 nodes in the Lorenz 96 reservoir layer were plotted. In layers #1–4, the nodes were sorted in the order of the two nodes' correlation and only the first 1,000 nodes were plotted.

S5. ATTRACTOR TRANSITION THROUGH MLP LAYERS

To visualize the transformation through the MLP layers, we depict a 3D plot or time series for each layer. Figure S8a and S8b illustrate the 3D trajectories for each layer in the attractor embedding task of the Rössler model and Lissajous curves, respectively. We perform principle component analysis to extract three-dimensional states from each layer and plot first three principal components (PC1, PC2, PC3). In the case of the embedding task of the KS model, we

depict the time series for each layer in Fig. S8c. In the three cases, each layer shows different trajectories or time series, which are linearly combined by the skip connections of the output layer to emulate the target.

S6. EXAMPLES OF TIME-INVARIANT TRANSFORMATION

We introduce TI transformation for nonstationary and periodic systems.

Nonstationary System with Detrending

A nonstationary system with input shows TV behavior. Here, we adopt the trend stationary model [53] to show its TI transformation. Let the input at the t^{th} step be u_t , and the state x_t is described by the sum of the function of time $f(t)$ and stationary process $g(u_{t-1}, u_{t-2}, \dots)$ as follows:

$$x_t = f(t) + g(u_{t-1}, u_{t-2}, \dots). \quad (\text{S15})$$

where $g(u_{t-1}, u_{t-2}, \dots)$ is TI and has a fixed time-averaged value. The detrending works as a TI transformation. For example, if the trend stationary model is represented by

$$x_t = at + b + cu_{t-1}, \quad (\text{S16})$$

we can calculate the average of N past states $\mathbf{x}_t = \{x_{t-N+1}, \dots, x_t\}$ as follows:

$$\bar{x}_t = \frac{1}{N} \sum_{i=0}^{N-1} x_{t-i} = at - \frac{N-1}{2}b + c\bar{u},$$

where $\bar{u} = \sum_{i=1}^N u_{t-i}/N$ converges to a fixed value if N is sufficiently large. Under this condition, the detrending can form a TI output:

$$f(\mathbf{x}_t) = x_t - \bar{x}_t = \frac{N+b}{2} + c(u_{t-1} - \bar{u}). \quad (\text{S17})$$

Note that Eq. (S17) works with the general representation of the stationary process $g(u_{t-1}, u_{t-2}, \dots)$ instead of cu_{t-1} in Eq. (S16). This transformation can be realized by a readout with memory. For example, if we use a dynamical system with linear readout as a readout of generalized reservoir computing, and it has a memory of N past states, the dynamical readout can realize Eq. (S17) by a weighted sum of delayed state series.

The detrending was utilized in ecological reservoir computing as post-processing [31]. In this study, the number of cultured unicellular organisms, *Tetrahymena*, was used as a reservoir state and was controlled by temperature input. The state increases with time, leading to a nonstationary state, which is transformed into a stationary state by the detrending technique. These manual operations remove the time dependency from the system states without ESP, heuristically converting them into states with ESP.

Periodic System with Envelope Extraction

We can transform a periodic system driven by input into a TI output using envelope extraction. Let the input at the t^{th} step be u_t . We assume that the state $x(t)$ is given by

$$x(t) = a(u_{t-1}, u_{t-2}, \dots) \cos(\Omega t),$$

where the amplitude $a(u_{t-1}, u_{t-2}, \dots)$ is a function of input, and Ω is the eigenfrequency. Let the Fourier transform of $a(u_{t-1}, u_{t-2}, \dots)$ be $A(\omega)$. Assuming that $A(\omega) = 0$ if $\omega > \Omega$, the Hilbert transform of $x(t)$ gives the following relation [54]:

$$H[x(t)] = \hat{x}(t) = a(u_{t-1}, u_{t-2}, \dots) \sin(\Omega t).$$

Finally, we can obtain the envelope as the time-invariant transformation

$$f[x(t)] = \sqrt{x(t)^2 + \hat{x}(t)^2} = a(u_{t-1}, u_{t-2}, \dots).$$

In the real STO reservoir, some studies employed the Hilbert transformation for the envelope extraction, which is applied as post-processing after observing the entire time series [25]. Others used an envelope detector with a diode [23, 55], which contributes to real-time computation.