Sculpting Deep Attractors in Reservoir Networks via Oja's Plasticity

Yuji Kawai
Symbiotic Intelligent Systems Research Center,
Institute for Open and Transdisciplinary Research Initiatives,
The University of Osaka,
Suita, Osaka, Japan
kawai@otri.osaka-u.ac.jp

Hiroshi Atsuta
Symbiotic Intelligent Systems Research Center,
Institute for Open and Transdisciplinary Research Initiatives,
The University of Osaka,
Suita, Osaka, Japan
hiroshi.atsuta@otri.osaka-u.ac.jp

Minoru Asada
International Professional University of Technology in Osaka
Kita-ku, Osaka, Japan
Symbiotic Intelligent Systems Research Center, Institute for Open and Transdisciplinary Research Initiatives,
The University of Osaka
Suita, Osaka, Japan
Chubu University Academy of Emerging Sciences
Kasugai, Aichi, Japan
asada@otri.osaka-u.ac.jp

***Abstract*—Recurrent neural networks are powerful tools for learning and generating complex temporal dynamics yet training them to efficiently produce stable and robust patterns remains a significant challenge. In this study, we propose a novel learning method that integrates a local Hebbian-like plasticity mechanism, Oja's rule, with the global feedback provided by first-order reduced and controlled error (FORCE) learning within a reservoir computing model. We demonstrate that this combined approach enhances the network's ability to learn periodic target dynamics, achieving significantly faster convergence from arbitrary initial states than the conventional FORCE model while generating dynamics that are robust to noise. Through eigenmode analysis of the recurrent weight matrix, we show that the learning process systematically forms a low-rank connectivity structure characterized by the emergence of dominant outlying eigenvalues. We further demonstrate that Oja's rule captures essential modes of the network dynamics and embeds them within this low-dimensional structure. This work establishes a clear pathway from local synaptic learning rules, through the formation of a global low-rank architecture, to the emergence of robust computational function in recurrent neural networks.**

***Index Terms*—reservoir computing, echo state networks, FORCE learning, synaptic plasticity, Oja's plasticity rule, low-rank connectivity structure**

I. Introduction

Understanding the neural principles underlying the brain's remarkable ability to generate and control complex temporal patterns remains a central challenge in neuroscience. Artificial neural networks, particularly recurrent neural networks, have emerged as valuable computational models for elucidating the neural mechanisms that support dynamic cognitive and behavioral processes [1], [2], such as working memory [3], [4] and rhythmic pattern generation [5], [6].

Reservoir computing is an efficient paradigm for recurrent neural networks in which only the output (readout) weights are trained, whereas the high-dimensional recurrent connectivity (the 'reservoir') remains fixed [7], [8]. When output is fed back into the reservoir, the network learns to perform one-step-ahead prediction of a target time series. After training, the system can autonomously generate the learned pattern, including complex and chaotic dynamics, by iteratively feeding its own output back as input [8]–[12]. A widely used algorithm for this purpose is first-order reduced and controlled error (FORCE) learning, an online method that trains the readout weights to minimize the error between the network output and a target signal [9]. A key feature of this approach is its use of strong output feedback, which stabilizes neural activity within the reservoir and enables robust learning and reproduction of a wide range of complex patterns.

In essence, this capability means that the network forms a stable attractor corresponding to the target pattern. Even when the network state is initialized far from this pattern, the closed-loop dynamics guide the output to converge toward the target. This property of forming stable attractors has been exploited to generate trajectories for robot movements [13]. For example, a reservoir computer can be trained on a desired trajectory for a robotic arm, which subsequently drives the robot. If the robot's state, such as its position, is abruptly perturbed by an external force, including a collision, the network automatically generates a trajectory that smoothly returns the system to the target path.

One promising approach to enhancing the performance of

This work was supported by JST PRESTO, Grant Number JPMJPR23S5, and JSPS KAKENHI, Grant Number 25H02612, Japan.

reservoir computing is the introduction of unsupervised, local plasticity rules within the reservoir network. Previous studies have shown that Hebbian-type plasticity, which strengthens connections between co-active neurons [14], improves the reservoir's computational capabilities [15], [16]. In addition, the effectiveness of anti-Hebbian plasticity has been demonstrated in this context [17], [18]. However, it remains unclear how such local plasticity mechanisms interact with the attractor dynamics established by global, error-correcting algorithms such as FORCE learning. This gap motivates a key question: Can integrating a local, biologically plausible plasticity rule facilitate learning guided by a global teaching signal? More specifically, can a Hebbian mechanism operating on recurrent connections accelerate the formation of a target pattern attractor while simultaneously shaping network connectivity into a more robust and efficient structure?

To address this question, this study proposes a novel learning framework that integrates Oja's rule, a canonical and stabilized form of Hebbian plasticity [19], into the FORCE model. In this model, recurrent synapses are updated via Oja's rule, while readout weights are adapted using the recursive least squares (RLS) algorithm to track a target pattern. Oja's rule is well known for performing online principal component analysis (PCA), effectively extracting the highest-variance dimensions from input signals [19]. We hypothesize that as the RLS algorithm steers network activity along a target trajectory, Oja's rule captures the principal modes of this activity and embeds them into the network's connectivity structure. We demonstrate that this synergistic approach not only accelerates convergence to the target pattern from arbitrary initial states but also forms a robust low-rank connectivity matrix. Eigenmode analysis reveals that this structural adaptation manifests as the emergence of dominant outlying eigenvalues, establishing a clear structural substrate for the network's stable attractor. This work thus elucidates a mechanism through which local plasticity and global feedback cooperate to construct resilient neural representations of temporal patterns.

II. Methods

A. FORCE learning

The proposed method utilizes the FORCE learning algorithm with a firing-rate neuron model [9], as illustrated in Fig. 1. A reservoir consisting of a random recurrent neural network is designed to spontaneously generate chaotic neural activity. The network output is calculated as a weighted linear sum of the states of all neural units, and the readout weights are updated online to minimize the error between the output and a target time series. This output is then fed back into the reservoir, suppressing chaos and stabilizing network dynamics. Consequently, the trained reservoir autonomously generates the target pattern. In contrast to standard reservoir computing, in which recurrent weights remain fixed, this study introduces synaptic plasticity in these connections. This plasticity is applied during a training phase, and Gaussian noise is injected into each neural unit during both training and test phases.

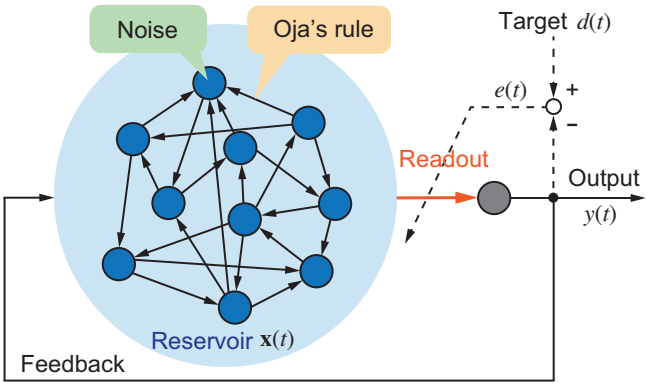

Fig. 1. Overview of the model architecture. The network output $y(t)$ is computed as a weighted linear sum of the reservoir activity $\mathbf{x}(t)$ and is subsequently fed back into the reservoir. During the training phase, the readout weights are updated using the RLS algorithm, indicated by dashed arrows. Simultaneously, recurrent connections within the reservoir are modified according to Oja's plasticity rule, while noise is injected into each neural unit. In the test phase, readout training and synaptic plasticity are deactivated.

We consider a reservoir composed of N neural units that generate N_{o}-dimensional output. The dynamics of the reservoir state $\mathbf{x}(t) \in \mathbb{R}^N$ are governed by the following equations:

$$\begin{aligned} \mathbf{x}(t+1) &= \left(1-\frac{1}{\tau}\right)\mathbf{x}(t) \\ &\quad +\frac{1}{\tau}\left(\mathbf{W}(t)\mathbf{r}(t)+\mathbf{W}_{\mathrm{fb}}\mathbf{y}(t)+\boldsymbol{\epsilon}(t)\right), \end{aligned} \tag{1}$$

$$\mathbf{r}(t) = \tanh(\mathbf{x}(t)), \tag{2}$$

where τ, $\mathbf{y}(t) \in \mathbb{R}^{N_{\mathrm{o}}}$, $\mathbf{W}(t) \in \mathbb{R}^{N\times N}$, $\mathbf{W}_{\mathrm{fb}} \in \mathbb{R}^{N\times N_{\mathrm{o}}}$, and $\boldsymbol{\epsilon}(t) \in \mathbb{R}^N$ denote the time constant, output, recurrent weight matrix, feedback weight matrix, and the noise term, respectively. The initial $\mathbf{x}(0)$ is drawn from a uniform distribution over the interval $[-1, 1]$. The initial $\mathbf{W}(0)$ is a sparse matrix with connection probability p. Its nonzero elements are drawn from a Gaussian distribution with zero mean and a standard deviation (SD) of $g/\sqrt{pN}$, where g denotes the gain of the recurrent weights. Finally, all diagonal elements of $\mathbf{W}(0)$ are set to zero to prevent self-connections. The elements of $\mathbf{W}_{\mathrm{fb}}$ are drawn from a Gaussian distribution with zero mean and an SD of $g_{\mathrm{fb}}/\sqrt{N_{\mathrm{o}}}$, where g_{fb} denotes the gain of the feedback weights, and is held fixed during training and testing. The vector $\boldsymbol{\epsilon}(t)$ represents Gaussian noise, drawn from a multivariate normal distribution with a zero mean vector and a diagonal covariance matrix, as shown below:

$$\boldsymbol{\epsilon}(t) \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}), \tag{3}$$

where σ and $\mathbf{I} \in \mathbb{R}^{N\times N}$ denote the SD, or amplitude of the noise, and the identity matrix, respectively.

As a linear readout of $\mathbf{r}(t)$, $\mathbf{y}(t)$ is computed:

$$\mathbf{y}(t) = \mathbf{W}_{\mathrm{o}}(t)\mathbf{r}(t), \tag{4}$$

where $\mathbf{W}_{\mathrm{o}}(t) \in \mathbb{R}^{N_{\mathrm{o}}\times N}$ denotes the readout weight matrix, which is initialized with all zeros.

TABLE I
PARAMETER VALUES

Parameter	Description	Value
N	Network size of reservoir	500
N_{o}	Number of output dimensions	2
g	Gain of reservoir weights	1.5
g_{fb}	Gain of feedback weights	3.0
τ	Time constant	10
σ	Noise amplitude	1
p	Connection probability	0.1
α	Initial value for RLS	1
η	Learning rate for Oja's rule	5.6e−5

B. Learning algorithms

During the training phase, $\mathbf{W}_{\mathrm{o}}(t)$ is updated using the RLS algorithm [20]. The update is performed independently for each output dimension. For the i-th output dimension, let $\mathbf{W}_{\mathrm{o},i}(t)$ denote the i-th row vector of $\mathbf{W}_{\mathrm{o}}(t)$, $y_i(t)$ be the corresponding output, and $d_i(t)$ be the target signal. The error is defined as $e_i(t) = y_i(t) - d_i(t)$. The vector $\mathbf{W}_{\mathrm{o},i}(t)$ is then updated at each time step t according to:

$$\mathbf{W}_{\mathrm{o},i}(t+1) = \mathbf{W}_{\mathrm{o},i}(t) - e_i(t)\mathbf{P}(t)\mathbf{r}(t). \tag{5}$$

This update rule requires an auxiliary matrix, $\mathbf{P}(t) \in \mathbb{R}^{N\times N}$, which is updated concurrently:

$$\mathbf{P}(t+1) = \mathbf{P}(t) - \frac{\mathbf{P}(t)\mathbf{r}(t)\mathbf{r}^{\top}(t)\mathbf{P}(t)}{1+\mathbf{r}^{\top}(t)\mathbf{P}(t)\mathbf{r}(t)}. \tag{6}$$

The matrix is initialized as $\mathbf{P}(0) = \alpha^{-1}\mathbf{I}$, where α denotes a positive constant.

Concurrently with the RLS update, the nonzero elements of $\mathbf{W}(t)$ are updated according to Oja's plasticity rule [19]. Assuming that $w_{ij}(t)$ represents the synaptic weight from the j-th neural unit to the i-th neural unit, the update rule is based on their respective activities, $r_i(t)$ and $r_j(t)$. The update at each time step is given by:

$$w_{ij}(t+1) = w_{ij}(t) + \eta r_i(t+1)\left(r_j(t) - r_i(t+1)w_{ij}(t)\right), \tag{7}$$

where η is the learning rate. The term $r_i(t+1)r_j(t)$ acts as a Hebbian component that strengthens correlated activity, while the term $-r_i(t+1)^2 w_{ij}(t)$ serves as a stabilizing factor that prevents unbounded growth of the weights.

III. Experimental settings

A. Simulation setting

All simulations were performed using the parameter values listed in Table I. These values were primarily based on previous studies on FORCE learning [9], [12]. As an exception, η was determined empirically. Unless otherwise specified, σ was set to 1.

During the training phase, the RLS algorithm (Eqs. (5) and (6)) and Oja's plasticity rule (Eq. (7)) were applied simultaneously at intervals of two time steps. The target signal, $\mathbf{d}(t) \in \mathbb{R}^{N_{\mathrm{o}}}$, was a two-dimensional circular trajectory with a period of 1000 time steps, given by:

$$\mathbf{d}(t) = \left[\cos(ct)\ \sin(ct)\right]/1.2, \tag{8}$$

where $c = 2\pi/10^3$. The network output was initialized as $\mathbf{y}(0) = \mathbf{d}(0) = [0.83\ 0]$. Each training trial was conducted for 1000 time steps, and this procedure was repeated five times.

During the test phase, the weight matrices $\mathbf{W}$ and $\mathbf{W}_{\mathrm{o}}$ were held fixed. In contrast to the training phase, the initial output, $\mathbf{y}(0)$, was systematically varied. Specifically, each dimension of $\mathbf{y}(0)$ was initialized to one of five values ranging from -1 to 1 in increments of 0.5. This procedure resulted in a 5×5 grid of 25 distinct initial conditions, and the network evolution was simulated for 1000 time steps from each of these conditions.

B. Analysis and evaluation

First, we analyzed $\mathbf{W}(t)$. We compared the distributions of the nonzero elements in $\mathbf{W}(t)$ before and after the training phase. A critical parameter governing reservoir dynamics and computational performance is the spectral radius of this matrix, $\rho(\mathbf{W})$, defined as its largest absolute eigenvalue [7].

To investigate the impact of Oja's plasticity, we tracked the temporal evolution of $\rho(\mathbf{W}(t))$ by calculating it at each time step. To examine how information is represented in the structure of $\mathbf{W}(t)$ after training, we analyzed the similarity between the learned weights and network activity patterns. First, we performed PCA for $\mathbf{W}(t)$ to extract the fist and second principal component (PC) vectors ($\mathbf{v}_1$ and $\mathbf{v}_2$). Next, we recorded time-series data of three key input variables while the trained network was operating:

1) Feedback input, $\mathbf{W}_{\mathrm{fb}}\mathbf{y}(t)$.
2) Inputs from other neurons, $\mathbf{W}(t)\mathbf{r}(t)$.
3) Full inputs, $\mathbf{W}(t)\mathbf{r}(t) + \mathbf{W}_{\mathrm{fb}}\mathbf{y}(t)$.

For each of these time-series datasets, we performed PCA and extracted the first and second PC vectors ($\mathbf{p}_1$ and $\mathbf{p}_2$). Finally, to quantify structural correspondence, we calculated the cosine similarity between $\mathbf{v}_1$ or $\mathbf{v}_2$ and $\mathbf{p}_1$ or $\mathbf{p}_2$. A high similarity value close to 1 indicates that the weight matrix is structured to align with a specific activity pattern. This analysis allowed us to determine which aspect of network dynamics, the target signal or the resulting internal activity, $\mathbf{W}(t)$ adapted to as a result of training.

Furthermore, we evaluated the network ability to reproduce the target trajectory during the test phase. To quantify performance, we calculated the minimum Euclidean distance, $\delta(t)$, between $\mathbf{y}(t)$ and any point on the target $\mathbf{d}$ at each time step t:

$$\delta(t) = \min_{n} ||\mathbf{y}(t) - \mathbf{d}(n)||. \tag{9}$$

To quantitatively evaluate network performance, we defined the following four metrics based on $\delta(t)$:

1) **Convergence time** (t_{c}): The first time step at which $\delta(t)$ drops below 0.01. This metric measures the speed of attraction.
2) **Transient distance**: The mean of $\delta(t)$ during the transient phase, that is, for $t < t_{\mathrm{c}}$. This metric quantifies the average error during the convergence process.
3) **Steady-state distance**: The mean of $\delta(t)$ during the steady-state phase, that is, for $t \geq t_c$. This metric

quantifies the long-term precision of the network in tracking the target trajectory.

4) **Fixed-point stability**: A binary metric indicating whether the output has settled into a stationary state, that is, a fixed point. This metric is assigned a value of 1 if the range, defined as the maximum minus the minimum, of each component of $\mathbf{y}(t)$ remains below 0.01 over the final 100 time steps of the simulation; otherwise, it is assigned a value of 0.

Metrics (1)–(3) were first averaged over the 25 initial conditions, whereas metric (4) was summed over the 25 initial conditions. Subsequently, all four resulting metrics were averaged across 50 independent simulations with different random seeds.

To evaluate the performance of our proposed model, we compared it against two baseline models. The first baseline incorporated an anti-Oja plasticity rule, which was implemented by inverting the sign of the learning rate ($\eta = -5.6\mathrm{e}-5$). The second was a non-plastic model where the weight matrix $\mathbf{W}(t)$ was fixed at its initial value, $\mathbf{W}(0)$. All other parameters for these baseline models were kept identical to those of the proposed model.

IV. Results

A. Analysis of recurrent weights

The initial structure of $\mathbf{W}(t)$ was compared with its structure after training. The weights were initialized by drawing from a zero-mean normal distribution, with the variance scaled by g of 1.5, resulting in an initial mean $\rho(\mathbf{W}(0))$ of 1.54 (SD = 0.026). As shown in a representative example (Fig. 2A), the overall shape of the weight distribution did not change significantly as a result of training. The distribution closely resembled that of the anti-Oja model (Fig. 2B).

However, $\rho(\mathbf{W}(t))$ increased substantially during training. Fig. 3 shows the temporal evolution $\rho(\mathbf{W}(t))$ during the training phase. The mean initial $\rho(\mathbf{W}(0))$ was 1.54 (SD = 0.026), whereas the final value $\rho(\mathbf{W}(5000))$ was 3.43 (SD = 0.27). This increase in $\rho(\mathbf{W}(t))$ was a robust outcome of learning and was independent of σ. By contrast, $\rho(\mathrm{W}(t))$ of the anti-Oja model showed little change, settling at a final mean value of 1.65 (SD = 0.026).

To understand the mechanism underlying the increased $\rho(\mathbf{W}(t))$, we analyzed the eigenvalue spectrum of $\mathbf{W}(t)$ before and after training (Fig. 4A). Initially, the eigenvalues of $\mathbf{W}(0)$ were uniformly distributed within a circle of radius 1.5. In contrast, training induced a pronounced change in this spectrum. While most eigenvalues remained within the initial circle, two distinct outlying eigenvalues emerged on the real axis at approximately 2.5 and 3.5. This spectral transformation indicates that the learning process embedded two dominant low-rank structural modes into $\mathbf{W}(t)$. Conversely, no such outliers were observed in the eigenvalue spectrum of the anti-Oja model (Fig. 4B).

To characterize the functional role of these emergent low-rank structures, we performed a structural analysis. We computed the cosine similarity between the principal axes of

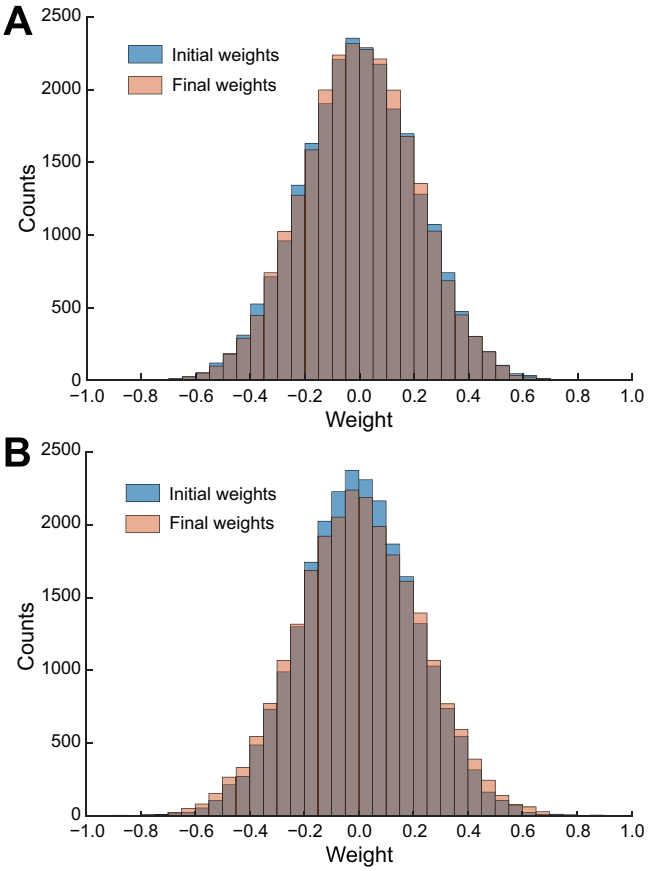

Fig. 2. Example histograms of all nonzero elements of the recurrent weights $\mathrm{W}(t)$ before training (blue, $t = 0$) and after training (orange). (A) Histogram of the proposed model. (B) Hisrogram of the anti-Oja model.

$\mathbf{W}(t)$, that is, the PC vectors, and the PC vectors of three key network signals: (1) $\mathbf{W}_{\mathrm{fb}}\mathbf{y}(t)$, (2) $\mathbf{W}(t)\mathbf{r}(t)$, and (3) $\mathbf{W}_{\mathrm{fb}}\mathbf{y}(t) + \mathbf{W}(t)\mathbf{r}(t)$. The results, summarized in Table II, reveal a striking transformation. Before training, the randomly initialized $\mathbf{W}(0)$ showed no significant alignment with any activity PCs, with all similarity scores near zero. After training, however, a clear structural relationship emerged. The first PC vector of the learned $\mathbf{W}(t)$ became aligned with the first PC vector of the inputs from other neurons (2), and the second PC vector aligned with its second PC. This demonstrates that Oja's plasticity and RLS-based learning do not embed a copy of the feedback, target-related signal into the weights. Rather, this learning process organizes $\mathbf{W}(t)$ to internalize and autonomously generate the principal dynamic modes experienced by the neural units during training, explaining why the network can function without the target signal after training.

B. Analysis of convergence speed

Fig. 5 shows example output trajectories originating from 25 different initial conditions. In all panels, the trajectories entered the target circular pattern, were initially drawn toward the origin, and then converged outward to the target circle. The eventual convergence to the target trajectory from all tested

TABLE II
MEAN COSINE SIMILARITY (SD)

			PC vector $\mathbf{W}_{\mathrm{fb}}\mathbf{y}(t)$		PC vector $\mathbf{W}(t)\mathbf{r}(t)$		PC vector $\mathbf{W}_{\mathrm{fb}}\mathbf{y}(t) + \mathbf{W}(t)\mathbf{r}(t)$	
			First	Second	First	Second	First	Second
PC vector of $\mathbf{W}(t)$	Initial	First	0.040 (0.027)	0.031 (0.022)	0.053 (0.040)	0.079 (0.061)	0.042 (0.031)	0.058 (0.043)
		Second	0.042 (0.029)	0.038 (0.027)	0.065 (0.048)	0.067 (0.052)	0.044 (0.039)	0.054 (0.036)
	Final	First	0.37 (0.16)	0.24 (0.16)	**0.97 (0.0082)**	0.036 (0.031)	0.90 (0.030)	0.067 (0.085)
		Second	0.28 (0.17)	0.32 (0.13)	0.034 (0.030)	**0.90 (0.039)**	0.070 (0.086)	0.80 (0.049)

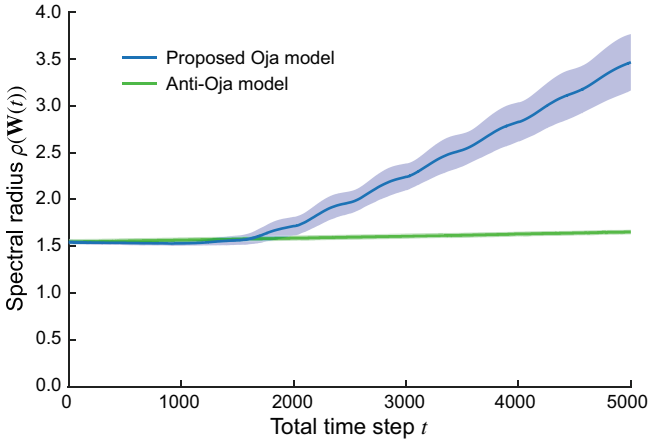

Fig. 3. Time evolution of the spectral radius of the recurrent weight matrix $\mathbf{W}(t)$ during the training phase. Shaded areas indicate the SD.

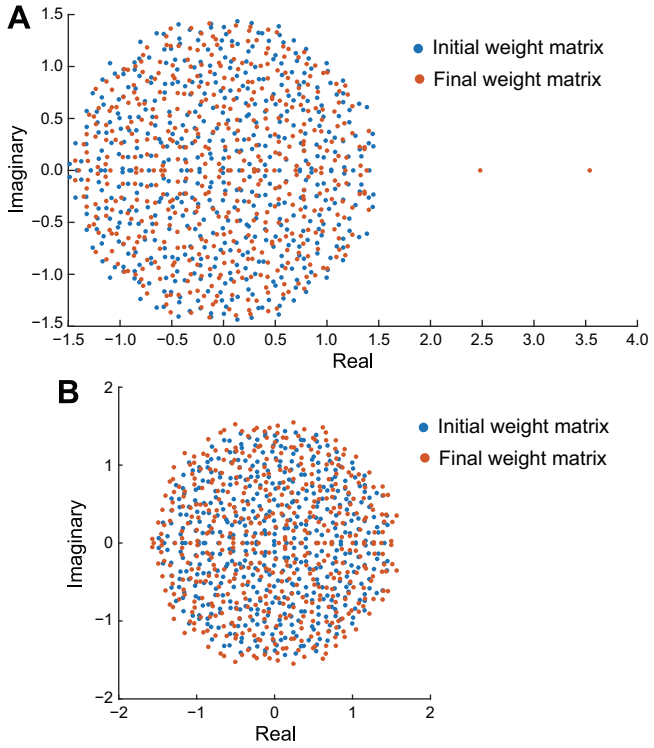

Fig. 4. Example eigenvalue spectra of the recurrent weight matrices before training (blue, $t = 0$) and after training (orange). (A) Spectral map of the proposed model. (B) Spectral map of the anti-Oja model.

initial conditions indicates that FORCE learning successfully created a stable attractor corresponding to the target dynamics. Notably, the initial inward penetration was less pronounced in the proposed method (A) than in the baseline without plasticity (B). The anti-Oja model produced noisy trajectories that did not precisely converge to the target circle (Fig. 5 C).

Fig.6 shows the convergence process by plotting $\delta(t)$. We compared the performance of the proposed model with plasticity with that of a baseline non-plastic model. To enable a rigorous comparison, we also tested a non-plastic model in which the gain was set to $g = 3.5$ to match the final spectral radius of the trained plastic network, $\rho(\mathbf{W}(t)) = 3.6$ (SD = 0.056). Initially, both the proposed and baseline models exhibited a sharp increase in $\delta(t)$, reflecting the dynamics transitioning into the interior of the target circle, before subsequently converging toward zero. Crucially, the proposed model converged to the target more rapidly and accurately. In contrast, the anti-Oja model and non-plastic model with $g = 3.5$ converged slowly and were unable to precisely track the target. This performance difference, despite similar spectral radii, demonstrates that the advantage of the proposed method is not merely due to a global shift in dynamics governed by the spectral radius, but rather to the specific structural changes induced by plasticity.

Fig. 7 shows the effect of σ, varied from 10^{-3} to 10^{1}, on the model outputs. The results for the anti-Oja model were excluded from Fig. 7 because its output distance, $\delta(t)$, never reached the specified threshold of 0.01 for any output. As illustrated in Fig. 7A, t_{c} of the proposed model was substantially shorter than that of the baseline model. This result indicates that Oja's plasticity contributed to rapid convergence. For the baseline non-plastic model with $g = 1.5$, t_{c} decreased as σ increased, suggesting that noise can accelerate convergence. However, this dependence on noise was not observed in the proposed model. Instead, the proposed model demonstrated robustness by maintaining a short convergence time even at large σ.

Fig. 7B compares the mean transition distance before convergence ($t < t_{\mathrm{c}}$). Across the full range of σ, the proposed model consistently exhibited a slightly smaller transition distance than the baseline models. Fig. 7C presents the mean convergence distance after convergence ($t \geq t_{\mathrm{c}}$). At small σ, the convergence distance of the non-plastic model with $g = 1.5$ was marginally smaller than that of the proposed

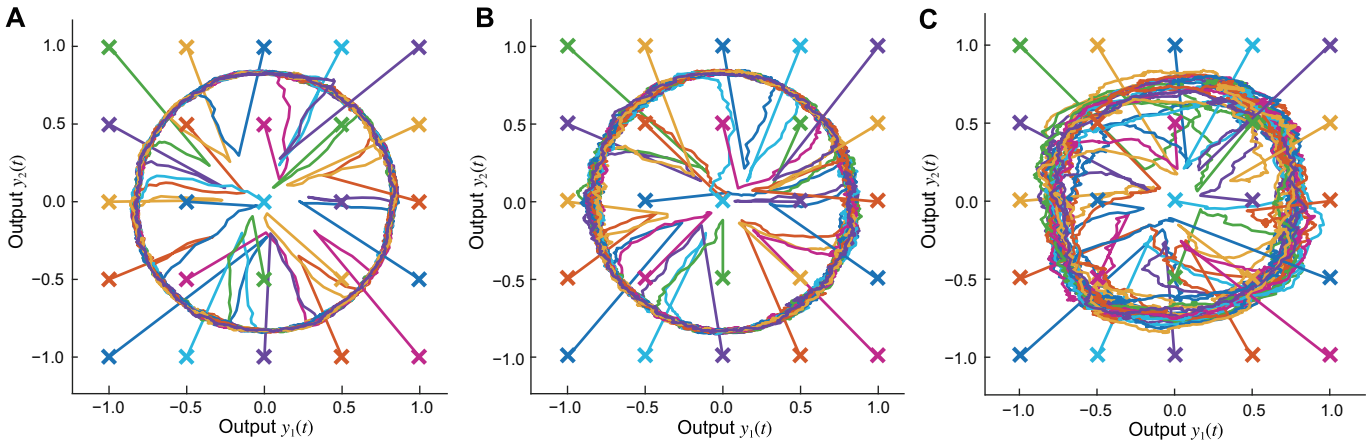

Fig. 5. Examples of network output trajectories from 25 different initial conditions (indicated by 'x' markers). (A) Proposed model with Oja's plasticity. (B) Model without plasticity. (C) Model with anti-Oja's plasticity.

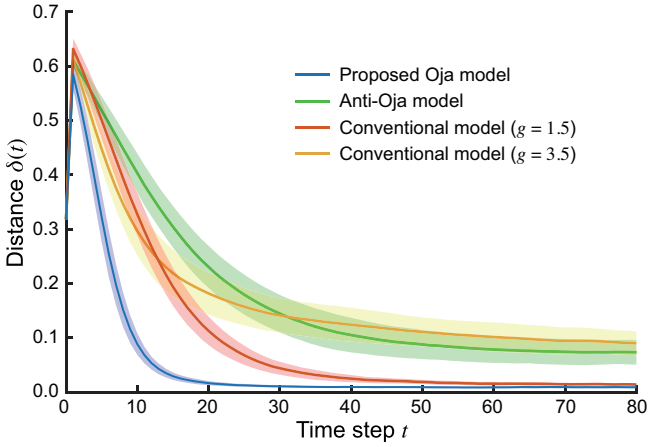

Fig. 6. Time evolution of the distance $\delta(t)$. The proposed model with plasticity (blue) is compared with the anti-Oja model (green) and the baseline without plasticity, initialized with $g = 1.5$ (red) and $g = 3.5$ (yellow). Shaded areas indicate the SD.

model, although the difference was small. In contrast, under large σ, the baseline models produced large deviations from the target, whereas the proposed model showed smaller deviations.

Finally, across all simulations, fixed-point stability scores were zero, indicating that the networks consistently avoided settling into fixed-point attractors and were therefore able to produce the desired time-varying outputs.

V. Discussion & Conclusion

In this study, we demonstrated that integrating Oja's plasticity rule into the FORCE learning framework enhances the ability of a reservoir computer to learn and generate periodic target dynamics. The proposed model achieved faster convergence from arbitrary initial conditions than the conventional model, even when the final spectral radii were matched. This accelerated convergence suggests that the learning process sculpts an effective attractor landscape characterized by a wider and deeper basin of attraction for the target trajectory. Furthermore, the resulting network exhibited robustness to noise, a critical feature for practical applications and a hallmark of a well-defined attractor.

The structural origin of this dynamical enhancement was revealed through spectral analysis of the recurrent weight matrix. After training, we observed the emergence of two outlying eigenvalues that separated from the circular bulk of the initial random matrix. This phenomenon indicates the formation of a low-rank structure embedded within the recurrent connectivity, which represents a core mechanism for encoding stable dynamics in recurrent neural networks [21]. Our analysis further shows that these dominant eigenmodes were systematically shaped by the reservoir activity.

The key to this effective structure formation lies in the computational role of Oja's rule, which performs an online form of PCA [19]. In this framework, the plastic rule guides the recurrent weights to capture the PCs of network activity. As the FORCE algorithm steers activity to follow the target dynamics, Oja's rule ensures that the dominant variance, corresponding to the most informative aspects of the reservoir dynamics, is encoded in the principal eigenmodes of the weight matrix. This process effectively carves out a low-rank subspace aligned with the reservoir dynamics, which manifests as the observed outlying eigenvalues. These results establishes a clear link between local synaptic learning, network structure, and the emergence of complex computational function.

In contrast, these phenomena—the emergence of eigenmodes and accelerated convergence—were not observed with the anti-Oja rule. Previous studies have demonstrated that the anti-Oja rule enhances the performance of echo state networks in tasks such as chaotic time series prediction and nonlinear function approximation [17], [18]. This is attributed to the fact that the anti-Oja rule updates recurrent weights to decorrelate neural activities, which might increase the complexity of the network's dynamics. Conversely, Oja's rule appears to embed neural activity into the recurrent weights, thereby constraining

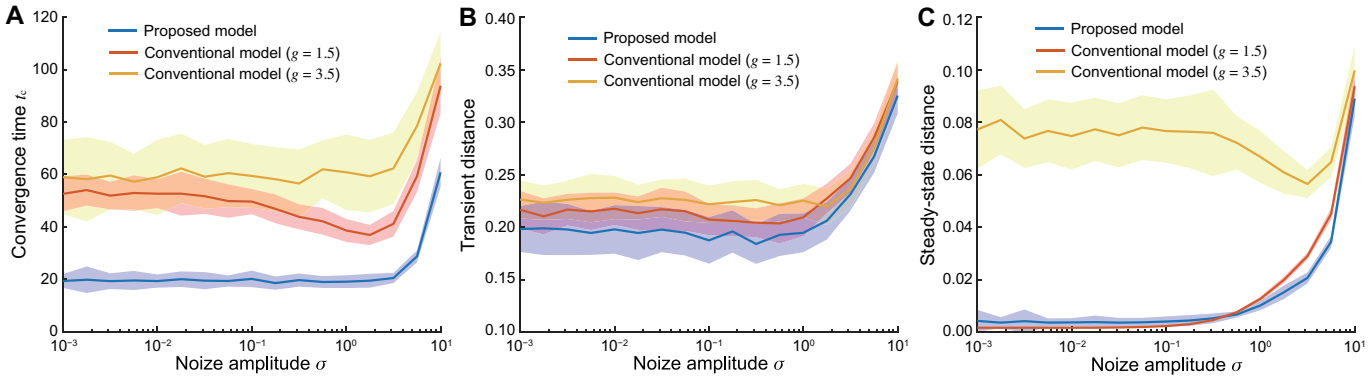

Fig. 7. Performance comparison between the proposed and baseline models. (A) Convergence time t_c. (B) Transient distance. (C) Steady-state distance. The proposed model with plasticity (blue) is compared with the baseline model without plasticity, initialized with $g = 1.5$ (red) and $g = 3.5$ (yellow). Shaded areas indicate the SD.

the network dynamics to represent the target time series. Clarifying these distinct functional roles will be a key area, potentially enabling the selection of the most suitable plasticity model for a given task.

This study opens several promising avenues for future research. Although the focus was on Oja's rule, exploring the impact of other biologically plausible plasticity mechanisms on attractor formation represents a natural next step. For example, the Bienenstock–Cooper–Munro rule, which enables bidirectional synaptic modification depending on postsynaptic activity levels, could provide a more flexible learning regime [22], [23]. In addition, homeostatic mechanisms such as intrinsic plasticity, which regulate overall network activity and neuronal firing rates, may further improve neural network performance [24], [25]. Investigating a multifaceted plasticity framework could therefore lead to even more robust and efficient learning systems.

References

[1] D. Sussillo, "Neural circuits as computational dynamical systems," *Current Opinion in Neurobiology*, vol. 25, pp. 156–163, 2014.

[2] D. Durstewitz, G. Koppe, and M. I. Thurm, "Reconstructing computational system dynamics from neural data with recurrent neural networks," *Nature Reviews Neuroscience*, vol. 24, no. 11, pp. 693–710, 2023.

[3] R. Pascanu and H. Jaeger, "A neurodynamical model for working memory," *Neural Networks*, vol. 24, no. 2, pp. 199–207, 2011.

[4] P. Enel, E. Procyk, R. Quilodran, and P. F. Dominey, "Reservoir computing properties of neural dynamics in prefrontal cortex," *PLOS Computational Biology*, vol. 12, no. 6, p. e1004967, 2016.

[5] K. Zemlianova, A. Bose, and J. Rinzel, "Dynamical mechanisms of how an RNN keeps a beat, uncovered with a low-dimensional reduced model," *Scientific Reports*, vol. 14, no. 1, p. 26388, 2024.

[6] Y. Kawai, S. Fujii, and M. Asada, "Modeling the rhythmic complexity of professional drumming with an oscillation-driven reservoir computer," *bioRxiv*, 2026.

[7] H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks," GMD-148, German National Research Center for Information Technology, Tech. Rep., 2001.

[8] H. Jaeger and H. Haas, "Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, no. 5667, pp. 78–80, 2004.

[9] D. Sussillo and L. F. Abbott, "Generating coherent patterns of activity from chaotic neural networks," *Neuron*, vol. 63, no. 4, pp. 544–557, 2009.

[10] J. Pathak, Z. Lu, B. R. Hunt, M. Girvan, and E. Ott, "Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, no. 12, p. 121102, 2017.

[11] J. Z. Kim, Z. Lu, E. Nozari, G. J. Pappas, and D. S. Bassett, "Teaching recurrent neural networks to infer global temporal structure from local examples," *Nature Machine Intelligence*, vol. 3, no. 4, pp. 316–323, 2021.

[12] Y. Kawai, T. Morita, J. Park, and M. Asada, "Oscillations enhance time-series prediction in reservoir computing with feedback," *Neurocomputing*, vol. 648, p. 130728, 2025.

[13] H. Atsuta, Y. Kawai, and M. Asada, "Enhancement of the robustness of redundant robot arms against perturbations by inferring dynamical systems using echo state networks," in *Proceedings of 2024 International Joint Conference on Neural Networks*, 2024, pp. 1–6.

[14] D. O. Hebb, *The Organization of Behavior*. New York: Wiley & Sons, 1949.

[15] A. Lazar, G. Pipa, and J. Triesch, "SORN: a self-organizing recurrent neural network," *Frontiers in computational neuroscience*, vol. 3, p. 800, 2009.

[16] T. Cazalets and J. Dambre, "Reshaping reservoirs with unsupervised Hebbian adaptation," *Nature Communications*, vol. 17, no. 450, 2025.

[17] Š. Babinec and J. Pospíchal, "Improving the prediction accuracy of echo state neural networks by anti-Oja's learning," in *Proceedings of the 17th International Conference on Artificial Neural Networks*, vol. 4668, 2007, pp. 19–28.

[18] X. Shen, S. Cheng, F. Li, and J. Feng, "Computational analysis of synaptic plasticity in echo state network," in *Proceedings of the 9th International Conference on Cognitive Computation and Systems*, 2025, pp. 272–282.

[19] E. Oja, "Simplified neuron model as a principal component analyzer," *Journal of Mathematical Biology*, vol. 15, no. 3, pp. 267–273, 1982.

[20] S. Haykin, *Neural Networks and Learning Machines*, 3rd ed. Upper Saddle River, NJ, USA: Pearson, 2009.

[21] F. Mastrogiuseppe and S. Ostojic, "Linking connectivity, dynamics, and computations in low-rank recurrent neural networks," *Neuron*, vol. 99, no. 3, pp. 609–623, 2018.

[22] E. L. Bienenstock, L. N. Cooper, and P. W. Munro, "Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex," *Journal of Neuroscience*, vol. 2, no. 1, pp. 32–48, 1982.

[23] M.-H. Yusoff, J. Chrol-Cannon, and Y. Jin, "Modeling neural plasticity in echo state networks for classification and regression," *Information Sciences*, vol. 364, pp. 184–196, 2016.

[24] J. Triesch, "Synergies between intrinsic and synaptic plasticity in individual model neurons," *Advances in Neural Information Processing Systems*, vol. 17, 2004.

[25] X. Wang, Y. Jin, and K. Hao, "Echo state networks regulated by local intrinsic plasticity rules for regression," *Neurocomputing*, vol. 351, pp. 111–122, 2019.