university of groningen

University Medical Center Groningen

University of Groningen

Co-evolutionary control of a class of coupled mixed-feedback systems

Venegas-Pineda, Luis Guillermo; Jardón-Kojakhmetov, Hildeberto; Cao, Ming

Published in:
Chaos

DOI:
10.1063/5.0230879

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2025

Link to publication in University of Groningen/UMCG research database

Citation for published version (APA):
Venegas-Pineda, L. G., Jardón-Kojakhmetov, H., & Cao, M. (2025). Co-evolutionary control of a class of coupled mixed-feedback systems. *Chaos*, *35*(3), Article 033155. https://doi.org/10.1063/5.0230879

Copyright
Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license. More information can be found on the University of Groningen website: https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment.

Take-down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): http://www.rug.nl/research/portal. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

Download date: 20-06-2025

Chaos
An Interdisciplinary Journal of Nonlinear Science

RESEARCH ARTICLE | MARCH 25 2025

Co-evolutionary control of a class of coupled mixed-feedback systems

Special Collection: Advances in Adaptive Dynamical Networks

Luis Guillermo Venegas-Pineda ; Hildeberto Jardón-Kojakhmetov ; Ming Cao

Check for updates

Chaos 35, 033155 (2025)
https://doi.org/10.1063/5.0230879

View Online
Export Citation

Articles You May Be Interested In

Functional differentiations in evolutionary reservoir computing networks
Chaos (January 2021)

Evolutionary game dynamics of controlled and automatic decision-making
Chaos (July 2015)

Rumor propagation in the framework of evolutionary game analysis
Chaos (March 2025)

03 April 2025 11:34:27

AIP Publishing
Chaos
Special Topics Open for Submissions
Learn More

AIP Publishing

Co-evolutionary control of a class of coupled mixed-feedback systems

Cite as: Chaos 35, 033155 (2025); doi: 10.1063/5.0230879
Submitted: 26 July 2024 · Accepted: 8 March 2025 ·
Published Online: 25 March 2025

View Online Export Citation CrossMark

Luis Guillermo Venegas-Pineda,[1] Hildeberto Jardón-Kojakhmetov,[1,a)] and Ming Cao[2]

AFFILIATIONS

[1]Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence, University of Groningen, Nijenborgh 9, 9747AG Groningen, The Netherlands
[2]Engineering and Technology Institute Groningen, University of Groningen, Nijenborgh 4, 9747AG Groningen, The Netherlands

Note: This paper is part of the Focus Issue on Advances in Adaptive Dynamical Networks.
[a)]**Author to whom correspondence should be addressed:** h.jardon.kojakhmetov@rug.nl. CogniGron (Groningen Cognitive Systems and Materials Center), University of Groningen (Univ Groningen), Nijenborgh 4, NL-9747 AG Groningen, Netherlands.

ABSTRACT

Oscillatory behavior is ubiquitous in many natural and engineered systems, often emerging through self-regulating mechanisms. In this paper, we address the challenge of stabilizing a desired oscillatory pattern in a networked system where neither the internal dynamics nor the interconnections can be changed. To achieve this, we propose two distinct control strategies. The first requires the full knowledge of the system generating the desired oscillatory pattern, while the second only needs local error information. In addition, the controllers are implemented as co-evolutionary, or adaptive, rules of some edges in an extended plant-controller network. We validate our approach in several insightful scenarios, including synchronization and systems with time-varying network structures.

© 2025 Author(s). All article content, except where otherwise noted, is licensed under a Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/). https://doi.org/10.1063/5.0230879

03 April 2025 11:34:27

This work tackles the challenge of stabilizing a desired oscillatory pattern in a network with a fixed structure. We propose two distributed controllers to achieve this. These controllers are fundamentally different: the first is highly robust and eliminates nonlinearities but requires full knowledge of the reference system. The second, inspired by the neuromodulation of biological systems, only requires local error information. Moreover, the controllers are implemented as co-evolutionary rules for edges connecting an oscillatory node with the controlled system. This effectively results in an adaptive network where the adaptation rules shape the dynamics of the controlled nodes. We showcase the co-evolutionary controllers by exploring the synchronization of the controlled system, showing that our approach is independent of the network structure, and testing the performance of the controllers in time-varying networks. Our theoretical and numerical results show two distributed strategies to induce a desired oscillatory pattern by adaptively modulating node> dynamics.

I. INTRODUCTION

Inducing and regulating a desired oscillatory behavior in networked systems is a fundamental problem in control theory, with ample relevance for both natural and engineered systems. Oscillations are ubiquitous in many real-world phenomena including industrial applications,[1–3] chemistry,[4,5] neural networks,[6,7] psychology,[8,9] and biology.[10,11] More specifically, in biological systems, rhythmic signals govern important physiological processes, including the heartbeat's regulation, circadian rhythms, central pattern generators, and neuronal signaling.[12–15] From the engineering side, several technologies also need such periodic signals, for example, to achieve the coordination of automation processes, including robotic locomotion,[16,17] or in the developments of synthetic biological systems.[18–20]

Several naturally occurring oscillatory patterns are self-regulated and emerge from the intrinsic dynamics of the system. In contrast, there are many scenarios, both natural and engineered,

© Author(s) 2025

where external control is required to induce or modify oscillations in a desired way. This is, for instance, relevant in neuromorphic engineering,[21] where particular oscillatory patterns of biologically inspired neuronal networks are critical, for example, to mimic cognitive functions, coordinate sensorimotor functions, synchronize spiking neural networks, and enable efficient signal encoding, to name a few.[22–24] While our interests in this paper are mainly theoretical, designing controllers that can reliably induce desired oscillations could lead to advances in neuromorphic engineering and design, brain-inspired control systems, and adaptive network dynamics, among others.

In this paper, we design two controllers that impose a desired oscillatory pattern on a networked system *without modifying its internal dynamics or interconnections*. Our focus is on a class of coupled mixed-feedback systems where the interplay of (fast) positive and (slow) negative feedback plays a key role in the generation of sustained oscillations [see (1) and further details below].

The manuscript is organized as follows: in Sec. II, we motivate and describe the problem at hand and propose two kinds of controllers. The first controller, while robust, needs complete information from the reference. In contrast, the second one only requires local error information. Next, in Sec. III, we embed the designed controllers into a networked framework, implementing them as a co-evolutionary, or adaptive, rule of the edges connecting a node (the controller's node) to the nodes of the controlled system. We compare and evaluate the controllers' performance in three key scenarios: synchronization, arbitrary network structures, and time-varying network structures. We conclude in Sec. IV, where we also discuss potential extensions of our work.

Notation: $\mathbb{R}$ and $\mathbb{C}$ denote the fields of real and complex numbers, respectively. We use sgn for the sign function and $\mathcal{O}$ to denote the big-Oh order symbol. An adjacency matrix is denoted by A, and a superscript r (for "reference") or p (for "plant"), as in A^r and A^p, is used whenever it is necessary to distinguish the adjacency matrices of the reference and of the plant. Further notation is clarified when necessary.

II. PROBLEM SETUP AND DESIGN OF THE CONTROLLERS

Our analysis is concerned with a class of dynamic networks given by the $2n$-dimensional system:

$$\begin{aligned}\frac{\mathrm{d}x_i}{\mathrm{d}t} &= -x_i - y_i + S\left(\alpha x_i + \beta \sum_{j=1}^{n} A_{ij} x_j\right),\\ \frac{\mathrm{d}y_i}{\mathrm{d}t} &= \varepsilon(x_i - y_i),\end{aligned} \tag{1}$$

where $n \geq 1$ is the number of nodes (oscillators) in the network, $0 < \varepsilon \ll 1$ is a small parameter describing the timescale separation between the variables, and S is a locally odd sigmoid function. This means, in particular, that the function $S : \mathbb{R} \to \mathbb{R}$ satisfies: (a) $S(0) = 0$, (b) $\frac{\mathrm{d}S}{\mathrm{d}x}(x) > 0\ \forall x \in \mathbb{R}$, (c) $\operatorname{argmax}(S'(x)) = 0$, and (d) $S(x) = -S(-x)$ for $x \in U$ with U a neighborhood of the origin. Moreover, $\alpha > 0$, $\beta > 0$, and $A = [A_{ij}] \in \mathbb{R}^{n\times n}$ is the adjacency matrix of the underlying graph.

Model (1) is an example of a coupled mixed-feedback system.[25,26] Each isolated node, namely,

$$\begin{aligned}\frac{\mathrm{d}x_i}{\mathrm{d}t} &= -x_i - y_i + S(\alpha x_i),\\ \frac{\mathrm{d}y_i}{\mathrm{d}t} &= \varepsilon(x_i - y_i),\end{aligned} \tag{2}$$

exhibits a (fast) positive feedback through $S(\alpha x_i)$ (recall that $\alpha > 0$) and (slow) negative feedback through the tendency of y_i to follow x_i.

Remark 1. *The nodes of (1), which are given by (2), undergo a supercritical Hopf bifurcation of the origin for $\alpha = \alpha^* := \frac{1}{S'(0)}$. This means that the response of each isolated node converges to the origin for $\alpha < \alpha^*$ and to a limit cycle when $\alpha > \alpha^*$.*

As in (2), a key feature of mixed-feedback systems is the interplay between positive and negative feedback loops across different timescales, which is commonly observed in many biological models, particularly in neuronal dynamics. For example, it is argued[26] that mixed feedback is *the* fundamental mechanism for excitability and explains how biological systems are able to exhibit sustained and robust oscillations.[27,28] Moreover, mixed-feedback architectures have been recently exploited to develop control methods for neuromorphic systems.[25]

Our main motivation to consider systems given by (1) is the theory developed in Ref. 29. Relevant to us is that they propose a procedure that allows one, for example, to design an adjacency matrix such that the output $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ corresponds to a *rhythmic profile of the network* (defined as the *inverse problem* in Ref. 29). While brief, let us be more precise: a rhythmic profile is an n-tuple $(\sigma_1 e^{\imath\phi_1}, \ldots, \sigma_1 e^{\imath\phi_1}) \in \mathbb{C}^n$, where $\sigma_i \in \mathbb{R}$ represents the amplitudes and $\phi_i \in [0, 2\pi)$ phases. If for all $i = 1, \ldots, n$, the solutions $x_i(t)$ of (1) converge to some periodic function with amplitude σ_i and phase ϕ_i defined as *oscillating function*,[29] then one says that the network is rhythmic. Assume further that $\sigma_1 > 0$ and that $\sigma_1 \geq \sigma_i$, for all $i = 2, \ldots, n$. The n-tuple $(1, \rho_2 e^{\imath\theta_2}, \ldots, \rho_n e^{\imath\theta_n})$, where $\rho_i = \frac{\sigma_i}{\sigma_1}$ and $\theta_i = \phi_i - \phi_1$ for all $i = 2, \ldots, n$, is called *a relative rhythmic profile*. By following the procedure summarized below, one can construct an adjacency matrix such that the solutions of (1) (locally) converge to a particular rhythmic profile:

1. Let $\omega_x = (1, \rho_2 e^{\imath\theta_2}, \ldots, \rho_n e^{\imath\theta_n})$ with $\theta_i \neq \{0, \pi\} \mod 2\pi$.
2. Pick $\mu_1 = a + \imath b$ with $a, b > 0$. Further choose $\mu_3, \ldots, \mu_n \in \mathbb{R}$ with $\mu_i < a$ for all $i = 3, \ldots, n$. Define $D = \operatorname{diag}(\mu_1, \overline{\mu_1}, \mu_3, \ldots, \mu_n)$.
3. Find an *invertible* matrix $Q = \begin{bmatrix} w_x & \overline{w_x} & B \end{bmatrix} \in \mathbb{C}^{n\times n}$, where $B \in \mathbb{R}^{n\times(n-2)}$.
4. Define $A = QDQ^{-1}$.

Remark 2.

- *We refer to an adjacency matrix obtained from the above algorithm as rhythmic.*
- *Once one picks μ_1, one can then choose the parameters α and β such that the origin of (1) is unstable, leading to the rhythmic profile through a Hopf bifurcation, recall Remark 1 and see Ref. 29, Sec. VIII.*
- *In Ref. 29, the slow-fast as well as the mixed-feedback structure of (1) plays a fundamental role. This structure will also be relevant for the controllers presented in Secs. II and III.*

03 April 2025 11:34:27

© Author(s) 2025

We now consider the following problem: suppose we are given a reference rhythmic network and a plant network, both of type (1). The adjacency matrix of the plant cannot be adjusted, but we want to render the plant rhythmic with respect to the reference's profile. Since the adjacency matrix of the plant is fixed, we shall design a controller that solves our problem.

Remark 3. *While our focus is theoretical, we notice that the described situation frequently appears in real-life scenarios: (a) many neuromorphic devices have fixed hardwired interconnections but allow modulation through externally controllable nodes; (b) deep brain stimulation consists of applying external electrical pulses to the (fixed) brain inducing a desired rhythmic activity. Similar challenges arise in robotics, where fixed mechanical structures coordinate with body motion, or the synchronization of power grids, where the underlying network remains unchanged while controllers regulate, for example, frequency stability.*

The control problem at hand is schematized in Fig. 1 and is described by the following system of equations:

$$\begin{aligned}
\frac{\mathrm{d}X_i}{\mathrm{d}t} &= -X_i - Y_i + S\left(\alpha_r X_i + \beta_r \sum_{j=1}^{n} A^r_{ij} X_j\right),\\
\frac{\mathrm{d}Y_i}{\mathrm{d}t} &= \varepsilon(X_i - Y_i),\\
\frac{\mathrm{d}x_i}{\mathrm{d}t} &= -x_i - y_i + S\left(\alpha_p x_i + \beta_p \sum_{j=1}^{n} A^p_{ij} x_j + u_i\right),\\
\frac{\mathrm{d}y_i}{\mathrm{d}t} &= \varepsilon(x_i - y_i),
\end{aligned} \tag{3}$$

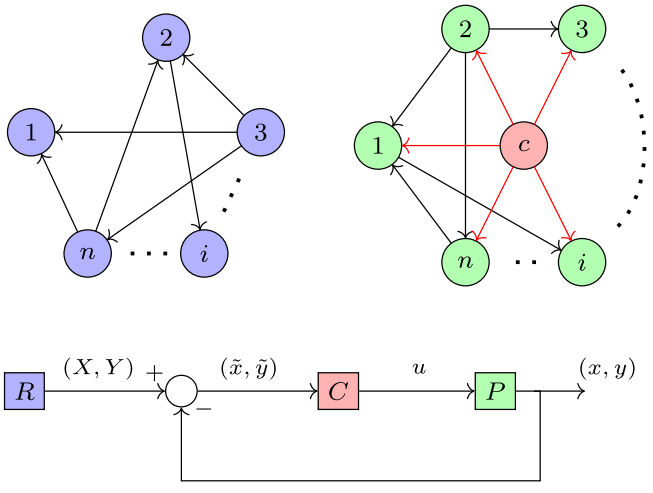

FIG. 1. Network (upper) and block diagram (lower) representations of our control problem (3), in which elements of the reference, plant, and control are presented in blue, green, and red, respectively. For the networks, the effect of the controller on the plant is displayed in red, while the rest of the connections are provided for illustrative purposes only, as any topology is allowed in our methodology. For the block diagram, uppercase (X, Y) and lowercase (x, y) represent the variables of the reference and of the plant, respectively, from which the error variables $\tilde{x} := X - x$ and $\tilde{y} := Y - y$ are defined for our analysis.

where $(X, Y) = (X_1, \ldots, X_n, Y_1, \ldots, Y_n) \in \mathbb{R}^{2n}$ are the states of the reference, $(x, y) = (x_1, \ldots, x_n, y_1, \ldots, y_n) \in \mathbb{R}^{2n}$ are the states of the plant, A^r and A^p denote, respectively, the adjacency matrices of the reference and of the plant, and u_i is the controller input into the ith node of the plant. The parameters $\alpha_r > 0$ and $\beta_r > 0$ are chosen according to the procedure sketched in Sec. I and together with A^r render the reference network rhythmic. The parameters $\alpha_p > 0$, $\beta_p > 0$, and A^p of the plant do not need to lead to a rhythmic profile. We also mention that the controller u_i is proposed to directly influence the plant's nodes in the same way as its internal connections and not as an independent mechanism due to the way the controller will be implemented in Sec. III.

Let us define the errors $(\tilde{x}, \tilde{y}) \in \mathbb{R}^{2n}$ by

$$\tilde{x}_i := X_i - x_i, \quad \tilde{y}_i := Y_i - y_i.$$

The corresponding error dynamics read as

$$\begin{aligned}
\frac{\mathrm{d}\tilde{x}_i}{\mathrm{d}t} &= -\tilde{x}_i - \tilde{y}_i + S\left(\alpha_r X_i + \beta_r \sum_{j=1}^{n} A^r_{ij} X_j\right)\\
&\quad - S\left(\alpha_p x_i + \beta_p \sum_{j=1}^{n} A^p_{ij} x_j + u_i\right),\\
\frac{\mathrm{d}\tilde{y}_i}{\mathrm{d}t} &= \varepsilon(\tilde{x}_i - \tilde{y}_i).
\end{aligned} \tag{4}$$

Remark 4. *The error system given in (4) is not in closed form. We shall deal with this when we design the controllers in Secs. II B–II C.*

In the following, we propose two different kinds of controllers that render the origin of (4) locally stable. The first one eliminates the sigmoidal nonlinearities, which results in a robust controller but requires complete knowledge of the reference. For comparison purposes, we propose an additional controller inspired by the synaptic modulation of neuronal systems. Such a controller does not require any knowledge of the reference. For clarity of our exposition, these two types of controllers are first provided without considering their implementation, see Propositions 1 and 2. After that, we propose a particular way of implementing them in a networked system where all nodes are of type (1), including the controller. This will mean that the controller action is implemented via the adaptation of the weights connecting the controller node and the plant's nodes; see more details in Sec. III.

A. General aspects of the simulations

We describe some generalities about the simulations that we present below. When required, more details are provided, and all codes are available in Ref. 30. In our simulations, the function S is chosen as the odd sigmoid $S(\cdot) = \tanh(\cdot)$, the small parameter ε is fixed at $\varepsilon = \frac{1}{100}$, while many of the other parameters are chosen at random. When we say that a parameter is chosen "at random," we always mean it under a uniform distribution and specify the interval of possible values. The following steps align with the algorithm described inSec. I:

03 April 2025 11:34:27

© Author(s) 2025

1. The relative amplitudes ρ_i, $i = 2, \ldots, n$, are chosen at random within the interval $(\frac{1}{3}, 1)$. The lower value $\frac{1}{3}$ is simply chosen so that the ith relative amplitude is not too small in the simulations. The phases θ_i, $i = 2, \ldots, n$, are chosen at random within the interval $(0, 2\pi)$. The probability of $\theta_i = \pi$ is zero.
2. For the leading eigenvalue $\mu_1 = a + \imath b$, we let $a = 1$ and b is randomly chosen within the interval $(\frac{1}{100}, \frac{1}{10})$. The choice of b is so that the period of the rhythmic profile, which is[29] $T \approx 2\pi(\beta \Im(\mu_1))^{-1}$, is not too small. The rest of the eigenvalues $\mu_i \in \mathbb{R}$, $i = 3, \ldots, n$, are chosen at random within the interval $(0, \frac{9}{10}a)$. This provides $D = \operatorname{diag}(\mu_1, \overline{\mu_1}, \mu_3, \ldots, \mu_n)$.
3. The complement matrix B is a randomly generated sparse matrix with nonzero coefficients within the interval $(0, 1)$. This matrix is generated until $Q = \begin{bmatrix} w_x & \overline{w_x} & B \end{bmatrix}$ is invertible.
4. The adjacency matrices A^r and A^p are, thus, given via the previous algorithm as $A^\bullet = QDQ^{-1}$. Since the algorithm to obtain the adjacency matrices is mostly random, both matrices are, with probability 1, different. Likewise, since B is sparse and randomly generated, the topologies of the adjacency matrices are, with probability 1, different. We emphasize that, from the results we present below, the plant's adjacency matrix A^p does not have to be rhythmic. To keep all systems (reference, plant, and controller) within the same context, we have opted to keep it rhythmic for the simulations.

In addition, and due to the choice of μ_1, $\beta_\bullet$ is randomly chosen within the interval $(0, 1)$, and the corresponding $\alpha_\bullet$ is set as $\alpha_\bullet = 1 + \varepsilon - \beta_\bullet + \frac{1}{100}$ (see Ref. 29, Theorem 1).

To better see the effect of the controllers, we present our simulations in the following way: for the first 300 time units, the controller is off, and so we see the open-loop response. At $t = 300$, the controller is turned on, and so from thereon, we see the closed-loop response. In our algorithms, this is realized by multiplying the controller by $l(t - t_c)$, where $l(x) = \frac{1}{1+e^{-x}}$ and $t_c = 300$ is the time at which the controller is turned on. Since we chose some parameters at random, the plots we show are representative of several simulation runs.

B. Elimination of nonlinearities

The first controller we propose follows from the next very simple observation: suppose that the controller can eliminate the sigmoidal nonlinearities of (4). The resulting error system is linear,

$$\begin{aligned} \frac{\mathrm{d}\tilde{x}_i}{\mathrm{d}t} &= -\tilde{x}_i - \tilde{y}_i, \\ \frac{\mathrm{d}\tilde{y}_i}{\mathrm{d}t} &= \varepsilon(\tilde{x}_i - \tilde{y}_i), \end{aligned} \tag{5}$$

and the origin is globally exponentially stable. Hence, the ideal controller that achieves this can be simply computed as follows:

$$\begin{aligned} u_i &= -\alpha_p x_i - \beta_p \sum_{j=1}^{n} A^p_{ij} x_j + \alpha_r \underbrace{(\tilde{x}_i + x_i)}_{X_i} + \beta_r \sum_{j=1}^{n} A^r_{ij} \underbrace{(\tilde{x}_j + x_j)}_{X_j} \\ &= \alpha_r \tilde{x}_i + (\alpha_r - \alpha_p) x_i + \sum_{j=1}^{n} \beta_r A^r_{ij} \tilde{x}_j (\beta_r A^r_{ij} - \beta_p A^p_{ij}) x_j. \end{aligned} \tag{6}$$

For future reference, let

$$F_i := \alpha_r \tilde{x}_i + (\alpha_r - \alpha_p) x_i + \sum_{j=1}^{n} \beta_r A^r_{ij} \tilde{x}_j (\beta_r A^r_{ij} - \beta_p A^p_{ij}) x_j. \tag{7}$$

For our implementations in Sec. III, it will be convenient that the controller is of integral type. Hence, we propose the closed-loop system:

$$\begin{aligned} \frac{\mathrm{d}X_i}{\mathrm{d}t} &= -X_i - Y_i + S\left(\alpha_r X_i + \beta_r \sum_{j=1}^{n} A^r_{ij} X_j\right), \\ \frac{\mathrm{d}Y_i}{\mathrm{d}t} &= \varepsilon(X_i - Y_i), \\ \frac{\mathrm{d}x_i}{\mathrm{d}t} &= -x_i - y_i + S\left(\alpha_p x_i + \beta_p \sum_{j=1}^{n} A^p_{ij} x_j + u_i\right), \\ \frac{\mathrm{d}y_i}{\mathrm{d}t} &= \varepsilon(x_i - y_i), \\ \frac{\mathrm{d}u_i}{\mathrm{d}t} &= k_i(F_i - u_i). \end{aligned} \tag{8}$$

Indeed, we have the following:

Proposition 1. *Consider (8) with F_i given by (7). If $k_i \geq k > 0$ for all $i = 1, \ldots, n$ and with k sufficiently large, then* $\lim_{t\to\infty} |\tilde{x}(t)| = \mathcal{O}(\frac{1}{k})$.

Proof. It suffices to let $k_1 = \cdots = k_n = k$. For $k \gg 1$ sufficiently large, the dynamics of u_i evolve in a fast timescale. In the limit when $k \to \infty$, F_i is constant, and the equilibrium of $\frac{\mathrm{d}u_i}{\mathrm{d}\tau} = (F_i - u_i)$, where $\tau = kt$ is the fast time, is hyperbolic. This is equivalent to saying that the critical manifold associated with (8), namely, $\mathcal{C} := \{u_i = F_i\}$ is normally hyperbolic[31] and globally exponentially stable. The restriction of (8) to the critical manifold leads precisely to the linear error dynamics (5), whose origin is globally exponentially stable. Since the equilibria of both the fast and the slow dynamics are exponentially stable, the result follows from, for example, Fenichel's (or Tikhonov's) theorem.[31,32] □

A simulation of (8) with F_i given by (7) is provided in Fig. 2; recall the general considerations described in Sec. II A. Figure 2 shows in the first two rows and Fig. 2(a) the time series for $x_i(t)$, $X_i(t)$, $u_i(t)$ and the mean norm of the error $\frac{1}{n}|\tilde{x}|$ in logarithmic scale for $n = 4$ nodes. In this and all following simulations, we prefer to show $\frac{1}{n}|\tilde{x}|$ because we keep all the controller constants the same, even for different numbers of nodes. Therefore, for comparison purposes, it is convenient to normalize the error. In Fig. 2(b), we show the mean error for a simulation similar to the previous one but for $n = 100$. Figure 2(c) also shows the mean error for a simulation with $n = 100$ nodes but for time-varying adjacency matrices. More precisely, for this simulation, each entry $A^\bullet_{ij}$ (for both the plant and the reference) of the adjacency matrices is of the form $A^\bullet_{ij} = A^\bullet_{ij}(t) = \bar{A}^\bullet_{ij}(1 + \frac{1}{5}\sin(\omega_{ij} t))$ where ω_{ij} is some random frequency within the interval $(0, 1)$ and $\bar{A}$ is a random rhythmic matrix. Since the controller fully uses these adjacency matrices, its performance is also good. Finally, on Fig. 2(d), we show another simulation for $n = 100$ nodes but now with a mismatch between the parameters used by

03 April 2025 11:34:27

© Author(s) 2025

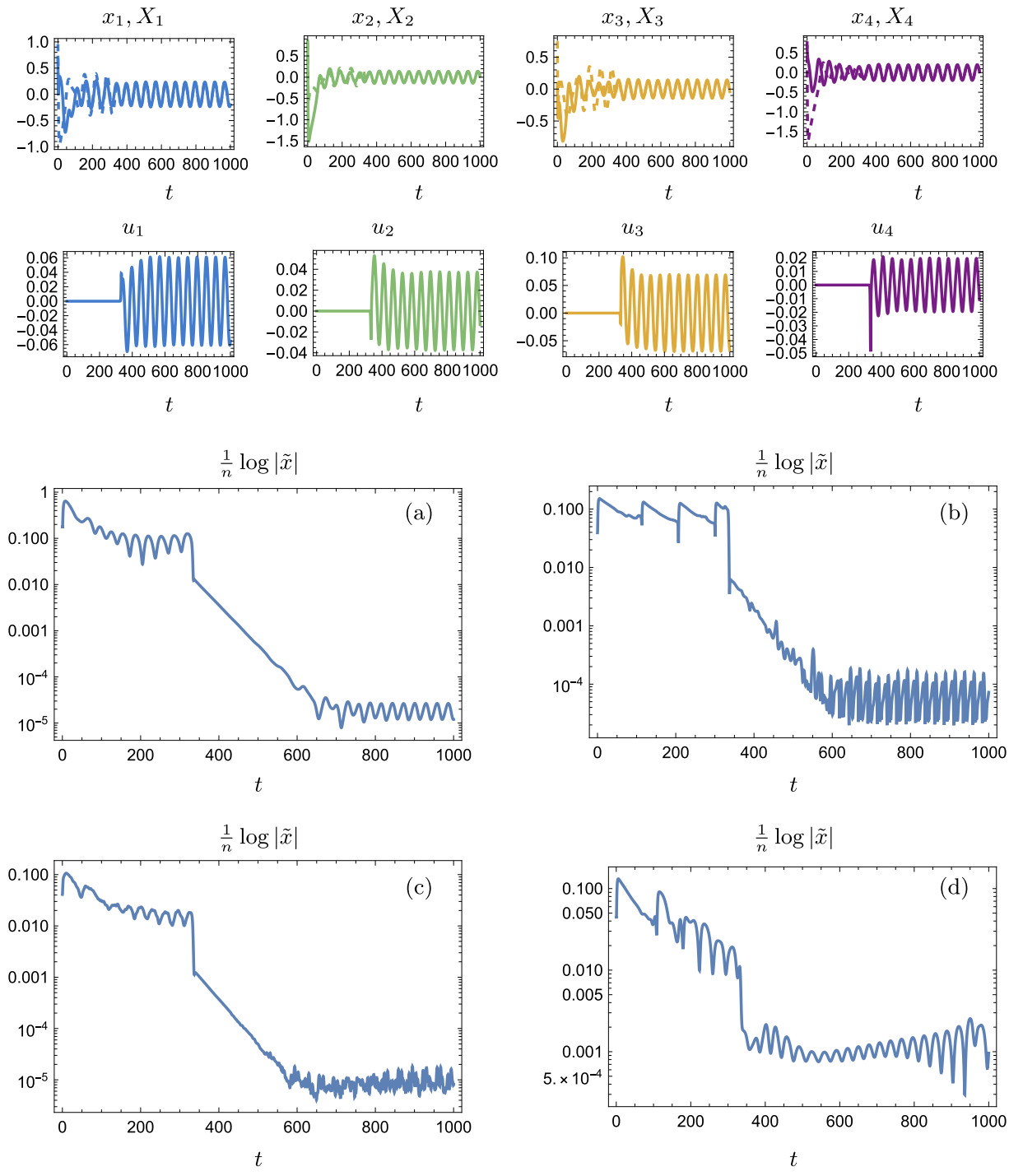

FIG. 2. The first and second rows and (a) correspond to a simulation of (8) with u_i^* given by (6) and for four nodes; x_i (plant) is dashed, and X_i (reference) is solid. (a)–(d) show, in logarithmic scale, the mean norm of the error, $\frac{1}{n}|\tilde{x}|$: (a) for four nodes, (b) for 100 nodes, (c) for 100 nodes with time-varying adjacency matrices, and (d) for 100 nodes with a mismatch in the reference's parameters used by the controller. For $t \in (0, 300)$, the controller is off, and, hence, we see the open-loop response. At $t = 300$, the controller is turned on, and from thereon, we observe the closed-loop response. We notice, naturally, that for $t > 300$, the plant closely follows the reference. See more details in the main text and compare it with Fig. 3.

03 April 2025 11:34:27

© Author(s) 2025

the controller and those of the reference. More precisely, for this simulation, we perturb the parameters α_r, β_r, and A^r_{ij} used by the controller in (6), by some small random number within the interval $(-\frac{1}{20}, \frac{1}{20})$. Such small perturbation is different for each parameter and each entry of A^r. We notice that since the mismatch is relatively small, the performance of the controller is still reasonable, although the error is roughly one order of magnitude larger than the error without the mismatch [Fig. 2(b)]. For all these simulations, we have set $k_i = k = 100$. It follows from Proposition 1 that the larger k, the smaller the error.

C. Neuromodulation inspired controller

As already mentioned, a drawback of the controller presented in Sec. II B is its dependence on the full knowledge of the reference's parameters. To offer an alternative, we propose a second controller whose behavior is inspired by the neuromodulation of synaptic weights in neuroscience and neuromorphic systems.

For clarity, let us recall the general error system,

$$\begin{aligned}\frac{\mathrm{d}\tilde{x}_i}{\mathrm{d}t} &= -\tilde{x}_i - \tilde{y}_i + S\left(\alpha_r X_i + \beta_r \sum_{j=1}^{n} A^r_{ij} X_j\right)\\ &\quad - S\left(\alpha_p x_i + \beta_p \sum_{j=1}^{n} A^p_{ij} x_j + u_i\right),\\ \frac{\mathrm{d}\tilde{y}_i}{\mathrm{d}t} &= \varepsilon(\tilde{x}_i - \tilde{y}_i).\end{aligned} \tag{9}$$

The controller we now propose is given in an integral form by

$$\frac{\mathrm{d}u_i}{\mathrm{d}t} = \tau\left(-\lambda u_i + k\tilde{x}_i\right), \tag{10}$$

with τ, k, and λ being positive constants.

Intuitively, the controller u_i adapts to the rescaled error $\frac{k}{\lambda}\tilde{x}_i$ (quickly if τ is large) and effectively provides a proportional feedback $u_i \approx \frac{k}{\lambda}\tilde{x}_i$. If the error $\tilde{x}_i$ is large, then the controller compensates the second sigmoid function to balance them out. If the error is small, then both sigmoid functions have roughly the same value and cancel out, which leads to the exponentially stable linear error dynamics (5). Besides the well-known advantages of introducing the controller in integral form, the idea just described has its inspiration in some neurological mechanisms, where the dynamic equation for u_i is in "leaky form," similar to the way biological neurons process information. Likewise, the introduced controller can be regarded as an adaptive synaptic weight that modulates the influence of the error. All these interpretations become more evident in Sec. III.

The previous idea is formalized in Proposition 2.

Proposition 2. *Consider error system (9) with the controller given through (10). If $\tau > 0$ and $k > 0$ are sufficiently large, then* $\lim_{t\to\infty} |\tilde{x}(t)| = \mathcal{O}(\frac{1}{k})$.

Proof. First, notice that the nonlinear terms in (9) are bounded, that is, $|S(\cdot)| \le m$. Due to the linear part of (9), its trajectories are globally attracted to a forward invariant set around the origin. Since $\sup_{(a,b)\in\mathbb{R}^2} |S(a) - S(b)| = 2m$, a rough estimate of such a region is a ball of radius $2m$ centered at the origin.

Substituting the steady state value of the controller $u_i = \frac{k}{\lambda}\tilde{x}_i = \sigma\tilde{x}_i$ into the equation of $\frac{\mathrm{d}\tilde{x}_i}{\mathrm{d}t}$ in (9),

$$\begin{aligned}\frac{\mathrm{d}\tilde{x}_i}{\mathrm{d}t} &= -\tilde{x}_i - \tilde{y}_i + S\left(\alpha_r X_i + \beta_r \sum_{j=1}^{n} A^r_{ij} X_j\right)\\ &\quad - S\left(\alpha_p x_i + \beta_p \sum_{j=1}^{n} A^p_{ij} x_j + \sigma\tilde{x}_i\right),\end{aligned}$$

where we recall that (X, Y) and (x, y) are bounded.

For simplicity, let

$$\zeta^r_i(z) = \alpha_r z_i + \beta_r \sum_{j=1}^{n} A^r_{ij} z_j,$$

$$\zeta^p_i(z) = \alpha_p z_i + \beta_p \sum_{j=1}^{n} A^p_{ij} z_j,$$

for $z \in \mathbb{R}^n$. Notice that $\zeta^r_i(z)$ and $\zeta^p_i(z)$, $i = 1, \ldots, n$, are bounded along solutions of (8), i.e., for $z = x$ or $z = X$.

Consider the candidate Lyapunov function $V = \frac{1}{2}\sum_{i=1}^{n}(\tilde{x}_i^2 + \frac{1}{\varepsilon}\tilde{y}_i^2)$. Then,

$$\frac{\mathrm{d}V}{\mathrm{d}t} = \sum_{i=1}^{n}\left(-\tilde{x}_i^2 - \tilde{y}_i^2 + \tilde{x}_i[S(\zeta^r_i(X)) - S(\zeta^p_i(x) + \sigma\tilde{x}_i)]\right).$$

We notice that $\tilde{x}_i S(\zeta^r_i(X)) \le m|\tilde{x}_i|$ and that $S(\zeta^p_i(x) + \sigma\tilde{x}_i) \sim S(\sigma\tilde{x}_i) + \mathcal{O}(\frac{1}{\sigma}) \sim m\,\mathrm{sgn}(\tilde{x}_i) + \mathcal{O}(\frac{1}{\sigma})$ as $\sigma \to \infty$. This leads to

$$\frac{\mathrm{d}V}{\mathrm{d}t} \sim \sum_{i=1}^{n}\left(-\tilde{x}_i^2 - \tilde{y}_i^2 + \mathcal{O}\left(\frac{1}{\sigma}\right)\right),$$

as $\sigma \to \infty$, showing that the region where $\frac{\mathrm{d}V}{\mathrm{d}t} > 0$ is bounded by a ball of radius $\mathcal{O}\left(\frac{1}{\sigma}\right)$. □

Figure 3 shows a simulation of (3) with the controller given by (10); recall the general considerations described in Sec. II A. The figure shows in the first two rows and Fig. 3(a) the time series for $x_i(t)$, $X_i(t)$, $u_i(t)$ and the mean norm of the error $\frac{1}{n}|\tilde{x}|$ in logarithmic scale for $n = 4$ nodes. In Fig. 3(b), we show the mean error for a simulation similar to the previous one but for $n = 100$. Figure 3(c) also shows the mean error for a simulation with $n = 100$ nodes but for time-varying adjacency matrices in the same way as those for Fig. 2. Since this controller does not use the parameters of the plant, there is no equivalent simulation to that of the Fig. 3(d) in Fig. 2. For all the simulations in Fig. 3, $\tau = 1$, $\lambda = 1$, and $k = 100$. It follows from Proposition 2 that the larger the k, the smaller the error.

III. CO-EVOLUTIONARY IMPLEMENTATION OF THE CONTROLLERS AND APPLICATIONS

In Sec. II, we have proposed two distinct controllers that render a reference profile stable for a plant with a fixed network structure. A system of the form (1) has the potential to be exploited in, for example, neuromorphic applications as it is suitable to be implemented

03 April 2025 11:34:27

© Author(s) 2025

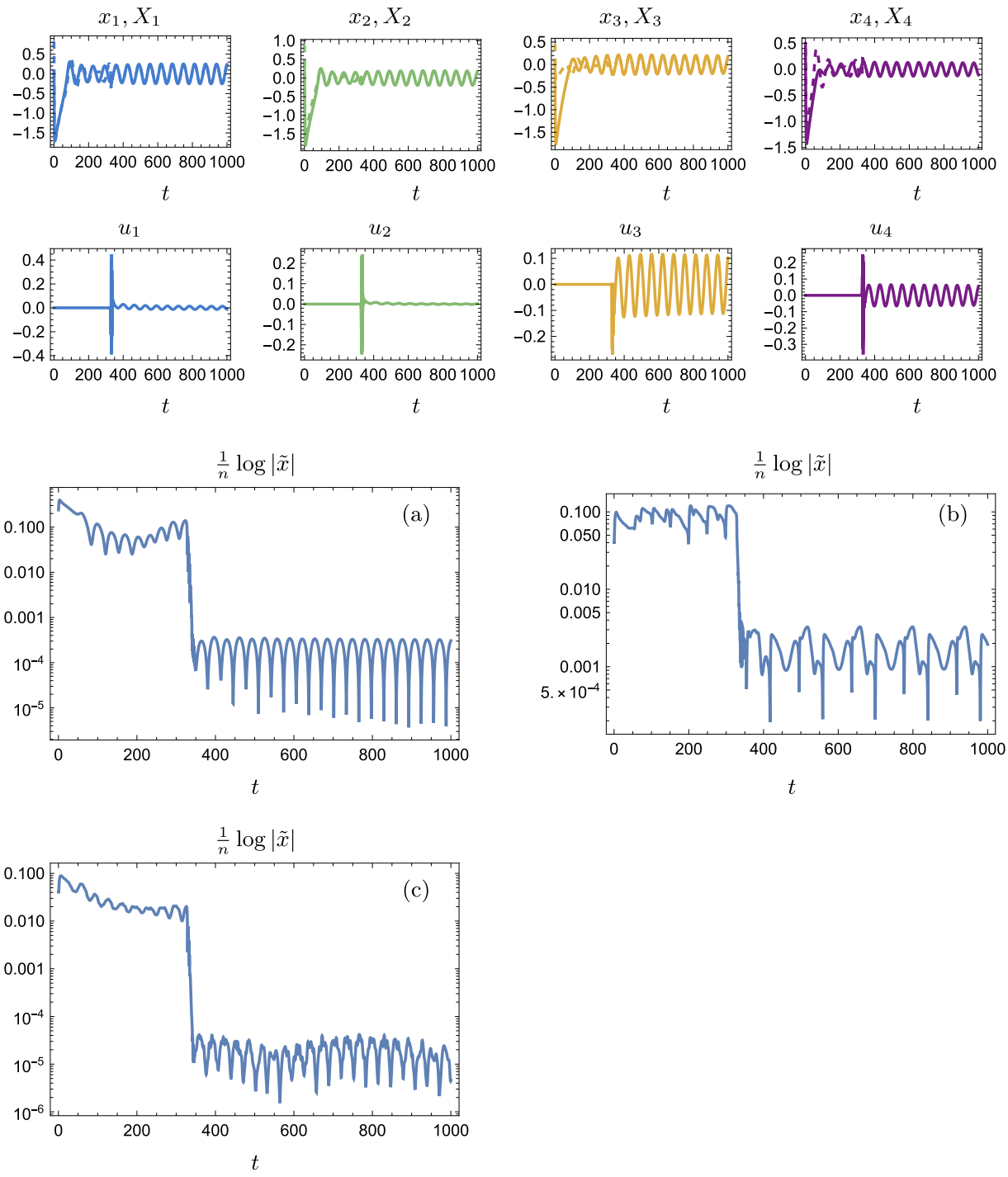

FIG. 3. Simulation of (3) with the neuromorphic inspired controller given by (10). The first and second rows and (a) correspond to a simulation with four nodes; x_i (plant) is dashed, and X_i (reference) is solid. (a)–(c) show, in logarithmic scale, the mean norm of the error $\frac{1}{n}|\tilde{x}|$: (a) for 4 nodes, (b) for 100 nodes, (c) for 100 nodes with time-varying adjacency matrices. For $t \in (0, 300)$, the controller is off, and, hence, we see the open-loop response. At $t = 300$, the controller is turned on, and from thereon, we observe the closed-loop response. We notice, naturally, that for $t > 300$, the plant closely follows the reference. See more details in the main text and compare it with Fig. 2.

03 April 2025 11:34:27

© Author(s) 2025

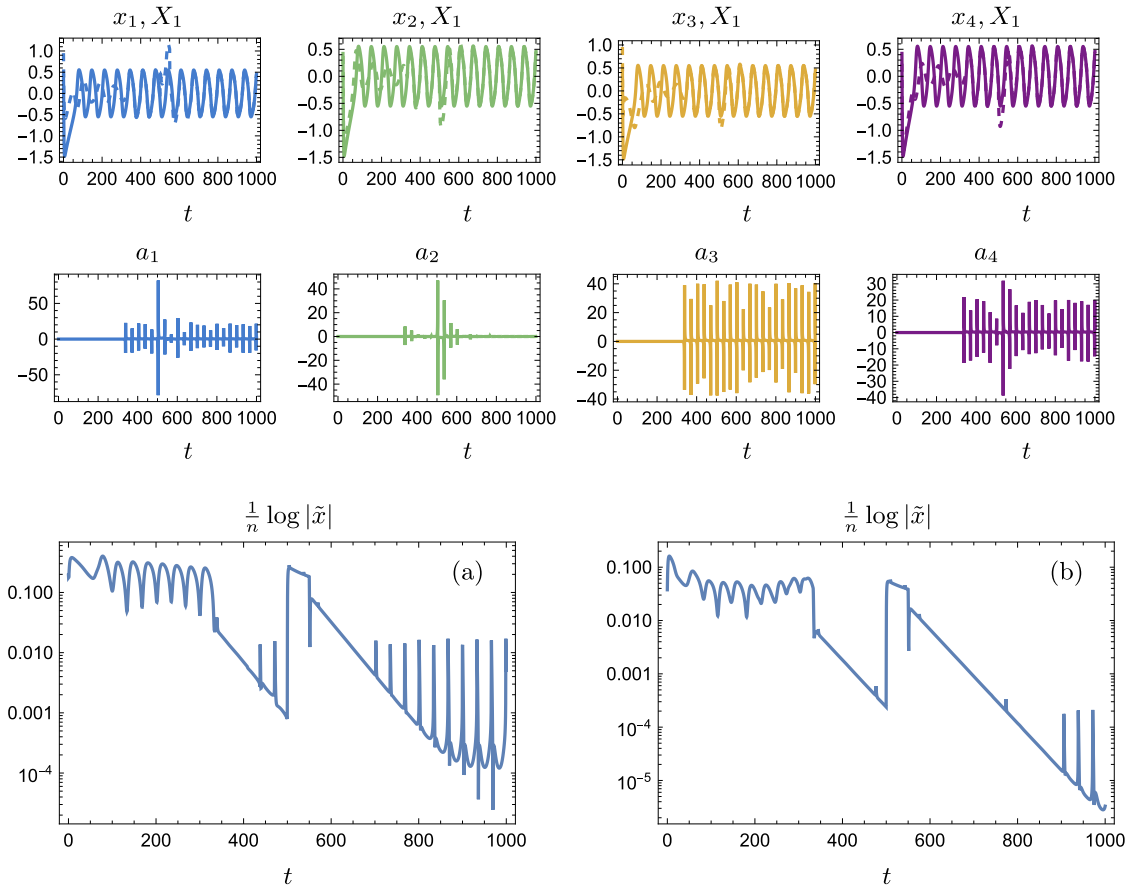

FIG. 4. Synchronization of the plant with respect to a reference node using the adaptation rule (12) and (13). The first and second rows show the time series of X_1 (solid) and x_i (dashed) and the corresponding adaptive weights a_i for $n = 4$ nodes, while (a) shows the corresponding error in logarithmic scale. (b) shows a similar simulation but for $n = 100$ nodes. In all simulations, we first show the system in open-loop for $t \in [0, 300)$. At $t = 300$, the controller is turned on, and we see how the error decreases. For $t \in [500, 550]$, a random constant disturbance is added to each node of the plant, hence the observed increase in the error. After $t = 550$, we see how the error again decreases, showcasing the recovery from the added disturbance.

in hardware. Therefore, it is reasonable to consider possible ways to also implement the controllers. While there are many ways to approach this, in this section, we propose to consider an extra node of type (1) as the controller. Since the dynamics of the controller's node are relatively simple (either converge to the origin or oscillate), the actual control action is to be performed by adapting the weights of the edges connecting the controller's node to the plant. Biological systems, particularly neuronal ones, are the inspiration for this. The controller is regarded as a pre-synaptic neuron and the nodes of the plant as post-synaptic, while the adaptation rule is associated with synaptic plasticity. More precisely, we consider the system,

$$\begin{aligned}
\frac{dX_i}{dt} &= -X_i - Y_i + S\left(\alpha_r X_i + \beta_r \sum_{j=1}^{n} A_{ij}^r X_j\right),\\
\frac{dY_i}{dt} &= \varepsilon(X_i - Y_i),\\
\frac{dx_i}{dt} &= -x_i - y_i + S\left(\alpha_p x_i + \beta_p \sum_{j=1}^{n} A_{ij}^p x_j + a_i x^c\right),\\
\frac{dy_i}{dt} &= \varepsilon(x_i - y_i),\\
\frac{dx^c}{dt} &= -x^c - y^c + S(\alpha_c x^c),\\
\frac{dy^c}{dt} &= \varepsilon(x^c - y^c),\\
\frac{da_i}{dt} &= \tau_c g_i(a_i, x^c, y^c, x, y, \tilde{x}, \tilde{y}).
\end{aligned} \tag{11}$$

where the control is now to be performed by the adaptation rule g_i, and $\tau_c > 0$ sets the adaptation rate. The parameter $\alpha_c > 1$ is chosen so that the controller's node (x_c, y_c) is oscillatory. By "implementation of the controllers," we precisely mean that we will choose an

03 April 2025 11:34:27

© Author(s) 2025

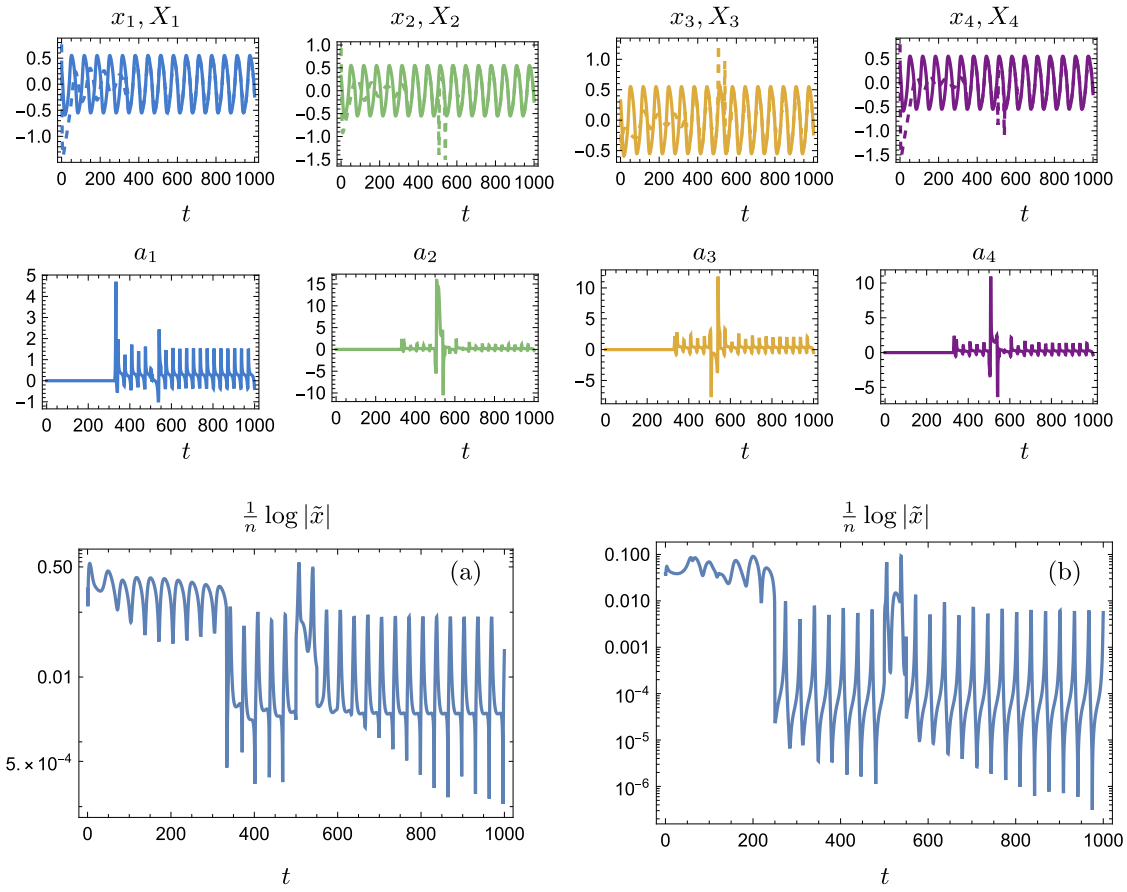

FIG. 5. Synchronization of the plant with respect to a reference node using the Hebbian-like adaptation rule (14). The first and second rows show the time series of X_1 (solid) and x_i (dashed) and the adaptive weights a_i for $n = 4$ nodes while (a) shows the corresponding error in logarithmic scale. (b) shows a similar simulation but for $n = 100$ nodes. In all simulations, we first show the system in open-loop for $t \in [0, 300)$. At $t = 300$, the controller is turned on, and we see how the error decreases. For $t \in [500, 550]$, a random constant disturbance is added to each node of the plant, hence the observed increase in the error. After $t = 550$, we see how the error again decreases, showcasing the recovery from the added disturbance. The most important distinction with Fig. 4 is that we observe periodic increases of the error, which coincide with the crossings of x^c at the origin and due to the multiplicative effect of x^c in the interconnection, namely, the term $a_i x^c$ in each equation of the nodes.

adaptation rule g_i to mimic the control actions of those in Secs. II B and II C.

The first controller, given by (6), is implemented through the adaptation rule,

$$g_i = (a_i^* - a_i), \tag{12}$$

where

$$a_i^* = \begin{cases} \frac{F_i}{x^c}, & |x^c| \geq \delta, \\ \frac{F_i}{\delta}, & 0 \leq x^c < \delta, \\ -\frac{F_i}{\delta}, & -\delta < x^c < 0, \end{cases} \tag{13}$$

with $0 < \delta \ll 1$ and F_i given by (7). Indeed, for $|x^c| > \delta$, we have that $a_i^* x^c = F_i$ and so the effective input to the ith node is exactly as in Proposition 1. Since x^c is oscillatory, and crosses the origin, we must account for it and, hence, propose the piece-wise continuous rule given by (13).

Remark 5.

- *The form of (13) is due to the multiplicative action of x^c. If $x^c = 1$, then the controller would be exactly the same as in (6). However, from (2), the only stable equilibrium point of the controller's node dynamics is the origin.*
- *While the adaptation rule is discontinuous, it is reminiscent of sliding mode control.[33] Improvements to this approach, for example, to avoid discontinuities or to rigorously prove the uniqueness of solutions is not further discussed here but shall be considered in future research.*

On the other hand, the controller given in Proposition 2 is *heuristically* implemented in (11) by

$$g_i = -\lambda a_i + k\tilde{x}_i x^c, \tag{14}$$

and compare with (10), where we notice that they share the same "error modulated behavior."

03 April 2025 11:34:27

© Author(s) 2025

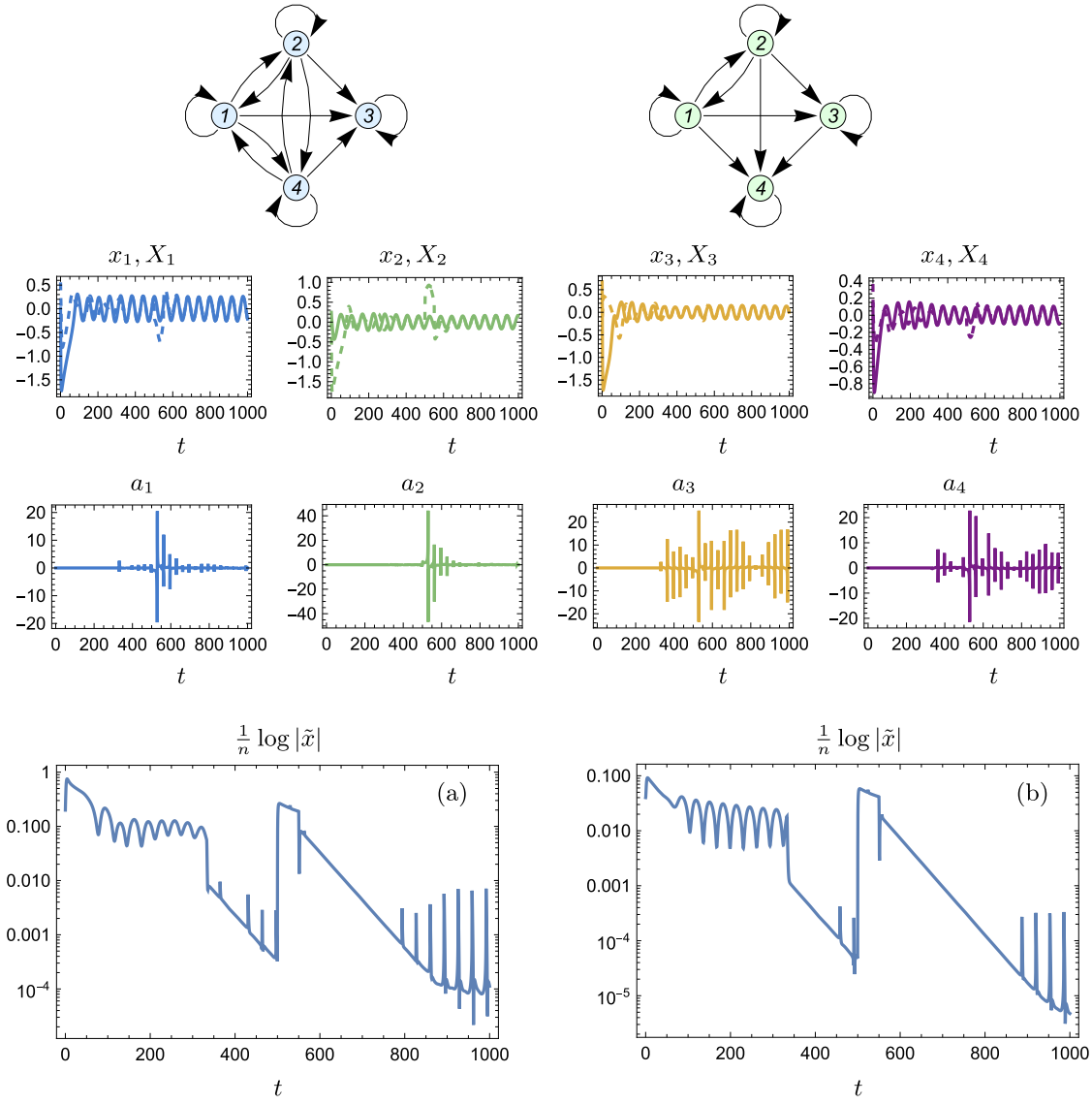

FIG. 6. Simulation of (11) with (12) and (13) where the plant and the reference have different network topologies. At the top of the figure, we show on the left the reference's graph, and on the right the plant's graph for $n = 4$, notice that they are indeed different. The second and third rows show the time series of X_i (solid) and x_i (dashed) and the adaptive weights a_i for $n = 4$ nodes, while (a) shows the corresponding error in logarithmic scale. (b) shows a similar simulation but for $n = 100$ nodes. In all simulations, we first show the system in open-loop for $t \in [0, 300)$. At $t = 300$, the controller is turned on, and we see how the error decreases. For $t \in [500, 550]$, a random constant disturbance is added to each node of the plant, hence the observed increase in the error. After $t = 550$, we see how the error again decreases, showcasing the recovery from the added disturbance.

Here, the biological inspiration is more evident: the adaptation rule (14) has two main components; one is a "pre-synaptic" activity-dependent term $\tilde{x}_i x^c$, which is regulated by the "post-synaptic" error and is reminiscent of a Hebbian-like adaptation (one could say that the learning rate is mediated by how far the plant's node is from the reference's), and an intrinsic decay $-\lambda a_i$ ensuring that the values of a_i remain bounded.

Remark 6. *The adaptation rule (14) is not obtained by letting $u_i = a_i x^c$ in Proposition 2. In such a case, one would obtain*

$$\frac{\mathrm{d}a_i}{\mathrm{d}t} = \frac{1}{x^c}\left(a_i \frac{\mathrm{d}x^c}{\mathrm{d}t} - \tau(-\lambda a_i x^c + k\tilde{x}_i)\right). \tag{15}$$

03 April 2025 11:34:27

© Author(s) 2025

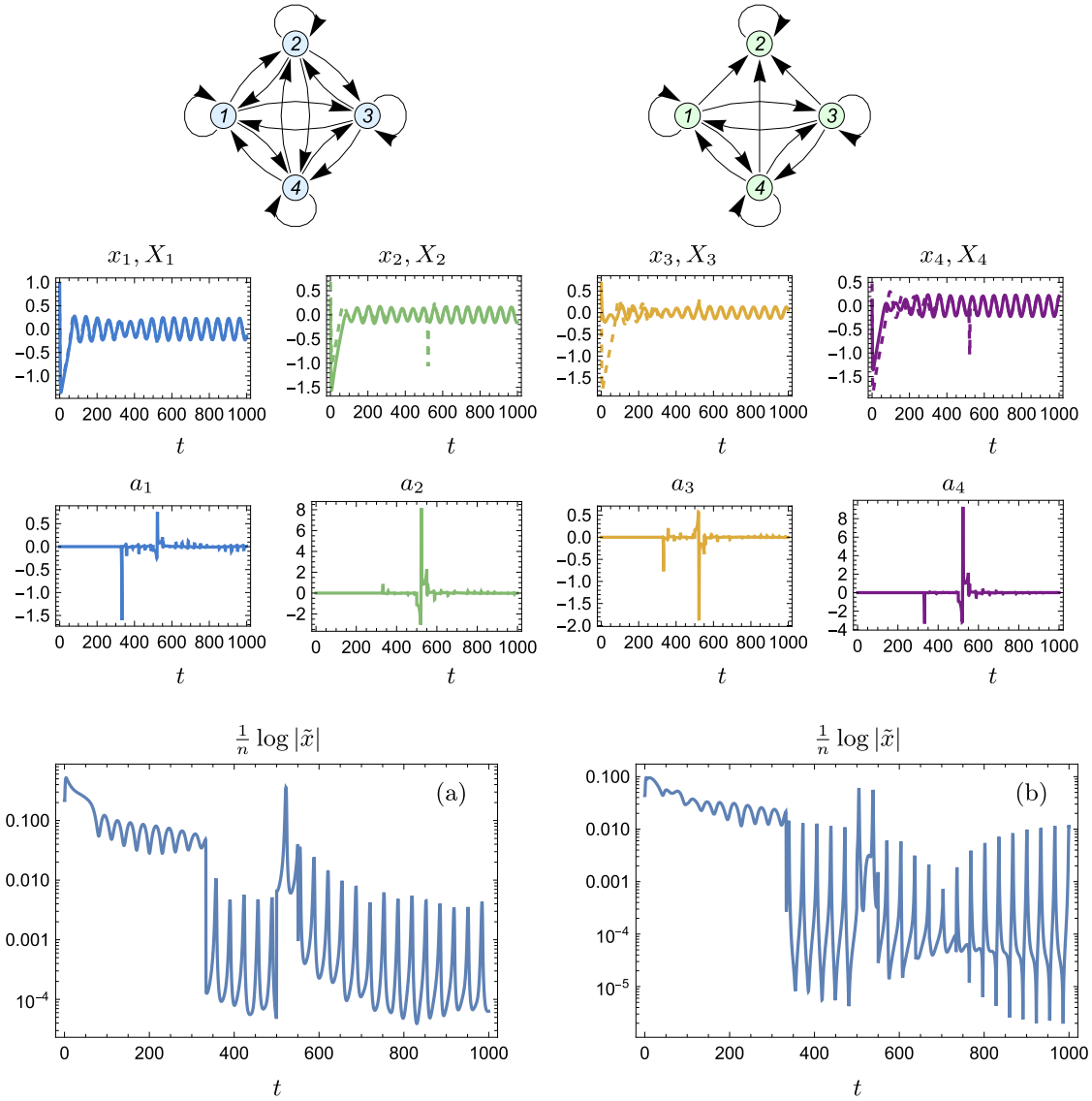

FIG. 7. Simulation of (11) with (14) where the plant and the reference have different network topologies. At the top of the figure, we show on the left the reference's graph, and on the right the plant's graph for $n = 4$, notice that they are indeed different. The second and third rows show the time series of X_1 (solid) and x_i (dashed) and the adaptive weights a_i for $n = 4$ nodes, while (a) shows the corresponding error in logarithmic scale. (b) shows a similar simulation but for $n = 100$ nodes. In all simulations, we first show the system in open-loop for $t \in [0, 300)$. At $t = 300$, the controller is turned on, and we see how the error decreases. For $t \in [500, 550]$, a random constant disturbance is added to each node of the plant, hence the observed increase in the error. After $t = 550$, we see how the error again decreases, showcasing the recovery from the added disturbance.

While the term $a_i \frac{dx^c}{dt}$ in (15) can be interpreted as a "predictive" term, we encounter, again, the nuance of dividing by x^c. Of course, one could opt to propose a similar discontinuous approach as in the previous case, but this would turn out to be difficult to justify implementation-wise. In addition, (15) lacks an intrinsic term guaranteeing boundedness, which is important in the present context.

A more sensible perspective is to recall that (10) effectively induces the feedback $u_i = k\tilde{x}_i$ into the error system. If now u_i is considered $u_i = a_i x^c$, then the ideal value of a_i would be $a_i = \frac{k\tilde{x}_i}{x^c}$, again introducing the division by x^c. In our heuristic implementation, we consider x_c a scaling, or gain, being its sign its most important aspect leading to (14). Naturally, since x^c crosses zero, each time $x^c = 0$, the

03 April 2025 11:34:27

© Author(s) 2025

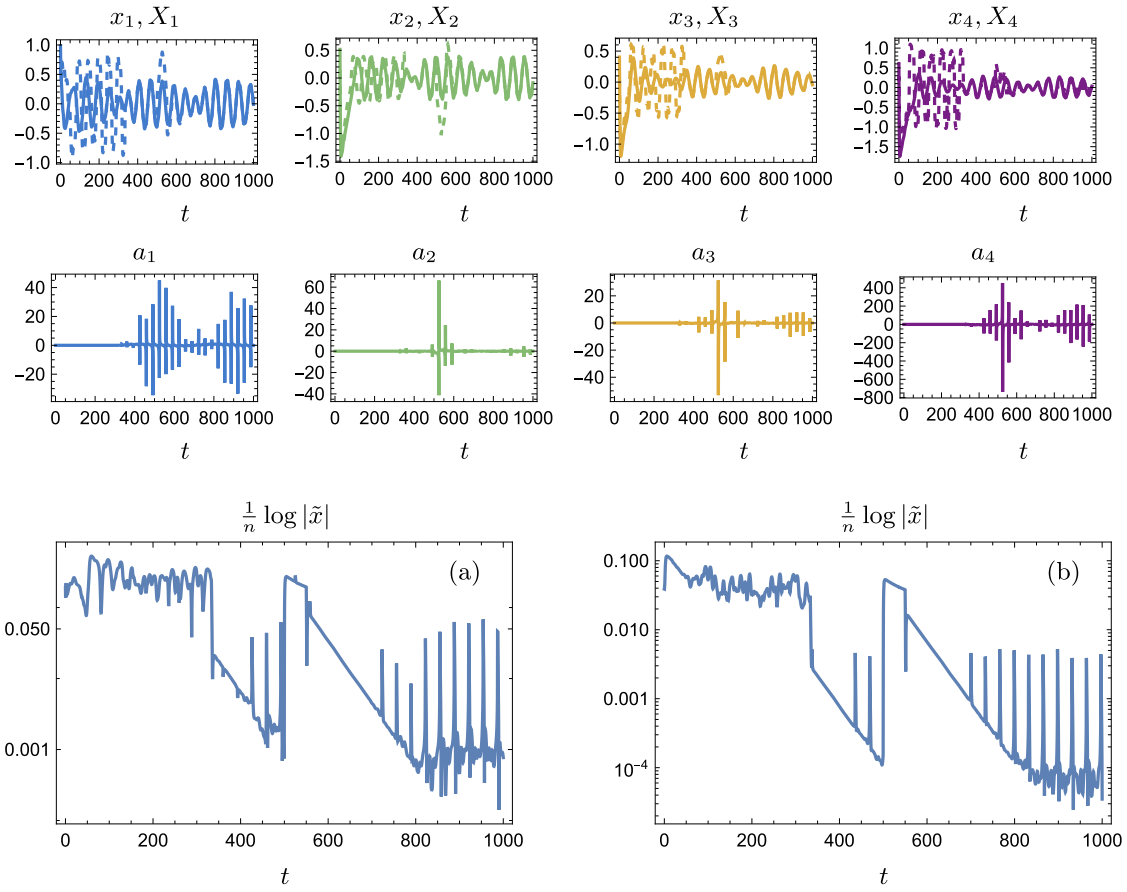

FIG. 8. Simulation of (11) with (12) and (13) where the plant and the reference have different network topologies with time-varying weights. The first and second rows show the time series of X_1 (solid) and x_i (dashed) and the adaptive weights a_i for $n = 4$ nodes, while (a) shows the corresponding error in logarithmic scale. (b) shows a similar simulation but for $n = 100$ nodes. In all simulations, we first show the system in open-loop for $t \in [0, 300)$. At $t = 300$, the controller is turned on, and we see how the error decreases. For $t \in [500, 550]$, a random constant disturbance is added to each node of the plant, hence the observed increase in the error. After $t = 550$, we see how the error again decreases, showcasing the recovery from the added disturbance.

"learned" value of a_i will be forgotten. This is, indeed, visible in the simulations presented below.

In Subsecs. III A–III C, we present the simulations of (11), and we compare through different application scenarios, the adaptation rules given by (12) and (13) and (14). In addition to the general considerations for our numerical simulations described in Sec. II A, we add the following: in the time interval $t \in (500, 550)$, we add a constant random disturbance, taking values in the interval $(-1, 1)$, to each node, different for each node and independent of the node. We do this to numerically test how the controlled plant recovers from a perturbed state.

To avoid repetitions, we mention here once that in our numerical simulations of (11) with the adaptive implementation given by (12) and (13) (specifically, Figs. 4, 6, and 8), we have used $\delta = \frac{1}{100}$ and $\tau_c = 100$. Regarding the adaptive implementation given by (14) (specifically Figs. 5, 7, and 9), we have used $\tau_c = 100$, $k = 100$ and $\lambda = 1$. We have kept these values the same across the different simulations for comparison purposes; of course, the performance of the adaptation rules can be improved if one tunes such parameters adequately.

A. Synchronization

In this section, we consider the problem of synchronization of the plant with respect to a provided reference. For this, the reference network has one node (X_1, Y_1), and the errors are thus defined by

$$\tilde{x}_i = X_1 - x_i, \qquad \tilde{y}_i = Y_1 - y_i.$$

Figure 4 shows a simulation of (11) with the adaptation rule given by (12) and (13). The observed spikes in the weight a_i are due to the discontinuous form of the adaptation and are modulated by δ in (13). In contrast, Fig. 5 shows a simulation of (11) with the adaptation rule given by (14). For this case, we do not see anymore the aforementioned spiking behavior since a_i is smooth. However, we observe that the error tends to increase periodically, and this coincides with x^c crossing zero; as for $x^c = 0$, the system is in open-loop.

03 April 2025 11:34:27

© Author(s) 2025

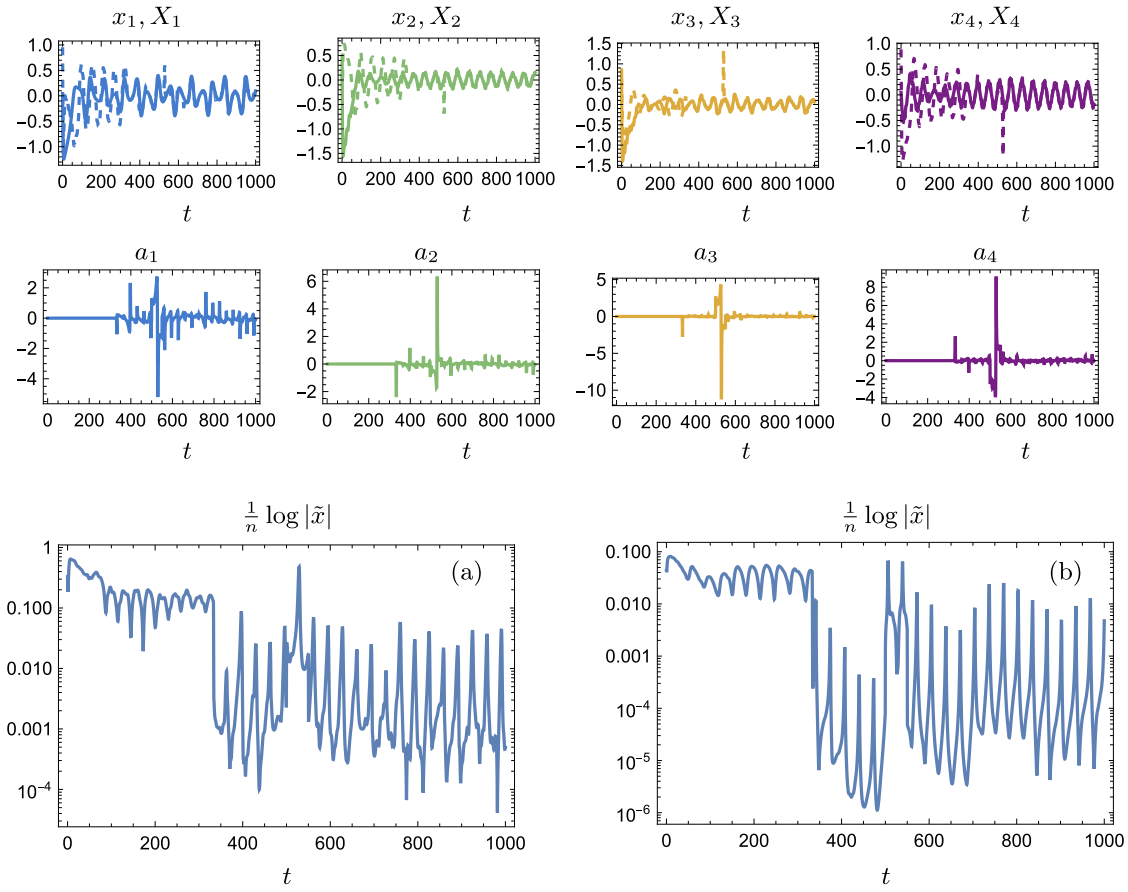

FIG. 9. Simulation of (11) with (14) where the plant and the reference have different network topologies with time-varying weights. The first and second rows show the time series of X_i (solid) and x_i (dashed) and the adaptive weights a_i for $n = 4$ nodes, while (a) shows the corresponding error in logarithmic scale. (b) shows a similar simulation but for $n = 100$ nodes. In all simulations, we first show the system in open-loop for $t \in [0, 300)$. At $t = 300$, the controller is turned on, and we see how the error decreases. For $t \in [500, 550]$, a random constant disturbance is added to each node of the plant, hence the observed increase in the error. After $t = 550$, we see how the error again decreases, showcasing the recovery from the added disturbance.

This issue is barely noticeable in the time series, though. We moreover notice that on both adaptation rules, the closed-loop systems recover well enough after the added disturbance in $t \in [500, 550]$.

Remark 7. *While our main exposition has considered that the reference and the plant have the same number of nodes, it is clear that, as in this section, such a consideration is not necessary. If the plant has $n \geq 1$ nodes and the reference has $m \geq 1$ nodes, one simply needs to define n errors $\tilde{x}_i = X_j - x_i$ for $i = 1, \ldots, n$ and $j \in \{1, \ldots, m\}$.*

B. Arbitrary network topologies

To highlight the fact that our approach does not depend on the particular topologies of the reference's and plant's adjacency matrices, in this section, we present simulations with explicitly different adjacency matrices. Figure 6 shows a simulation of (12) and (13), where for $n = 4$, we have included the graphs of the reference and of the plant, which are different. As in the simulations of Sec. III A, the spikes in the time series of a_i are due to the discontinuous implementation (12) and are modulated by δ in (12). For comparison, Fig. 7 shows a simulation of (11). Both simulations follow the same overall format we have used so far: for $t \in [0, 300]$, the controller is off, and hence, we see the open-loop response; at $t = 300$, the controller is turned on, and we see how the plant follows the reference, while for $t \in [500, 550]$, a constant random disturbance is added to each node. We notice from the time series that the performance of both controllers is reasonably good.

C. Time-varying adjacency matrices

As a final application and comparison setup, we consider here that the adjacency matrices of both the reference and the plant are time-varying. We do this similar to what was presented in Sec. II.

03 April 2025 11:34:27

© Author(s) 2025

More precisely, for the simulations of this section, each entry $A^{\bullet}_{ij}$ (for both the plant and the reference) of the adjacency matrices is of the form $A^{\bullet}_{ij} = A^{\bullet}_{ij}(t) = \bar{A}^{\bullet}_{ij}(1 + \frac{1}{5}\sin(\omega_{ij}t))$, where ω_{ij} is some random frequency within the interval $(0, 1)$ and $\bar{A}^{\bullet}$ is a random rhythmic matrix, meaning that the underlying topologies of each network are also different. While we do not perform formal analysis under this setup, both adaptation approaches seem suitable to handle this scenario. On the one hand, we have that adaptation rules (11) and (12) use the full knowledge of the adjacency matrices. On the other hand, the adaptive rule (11) accounts for the time-varying adjacency matrices from the fact that a_i is regulated by the error. Figure 8 corresponds to (10) with (11) and (12), while Fig. 9 uses (14). The description of the results is similar to Secs. III A and III B. In both cases, we can observe that, despite the adjacency matrices being time-varying, both adaptation rules perform well.

IV. CONCLUSION AND DISCUSSION

This paper has explored co-evolutionary control approaches to render a desired oscillatory pattern stable for a particular class of dynamic networks, keeping their internal properties fixed. We have developed two kinds of controllers: one that uses the full information from the reference and another that only requires local error information. From our analysis, both controllers seem to perform reasonably well. Their main difference is their potential implementation. While the first controller (see Proposition 1) performs observably well, it is hardly implementable not only because it requires full information about the plant but also because of its particular form [see (6)]. In contrast, the second controller (see Proposition 2) requires only local error information. We have also implemented such controllers as co-evolutionary rules in an extended dynamic network and tested their performance in three relevant scenarios. In all cases, again, the controllers perform reasonably well.

An immediate open problem to consider in the future is to extend our approach to more general coupled systems. As a concrete example, one may want to explore ideas similar to those developed here, but for coupled spiking neurons, whose models also have mixed-feedback loops. A challenge for these models is that the nonlinearity is not bounded, in contrast to (2). Another possibility is to consider that the reference and the plant are not of the same type. Hence, even if a system cannot inherently produce and sustain specific oscillations, a controller may impose them.

ACKNOWLEDGMENTS

The work of L.G.V.P. was founded by the Data Science and Systems Complexity (DSSC) Centre of the University of Groningen, as part of a PhD scholarship. H.J.K. would like to acknowledge the financial support of the CogniGron research center and the Ubbo Emmius Funds (University of Groningen). The authors thank the anonymous reviewers for their thorough comments that helped to improve the manuscript.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Luis Guillermo Venegas-Pineda: Conceptualization (equal); Data curation (equal); Formal analysis (equal); Methodology (equal); Software (equal); Visualization (equal); Writing – original draft (equal). **Hildeberto Jardón-Kojakhmetov:** Conceptualization (equal); Formal analysis (equal); Methodology (equal); Project administration (equal); Software (equal); Supervision (equal); Writing – original draft (equal). **Ming Cao:** Conceptualization (equal); Formal analysis (equal); Project administration (equal); Supervision (equal).

DATA AVAILABILITY

The data that support the findings of this study are openly available in CoevolutionaryControl-2025, at https://github.com/hjardonk/CoevolutionaryControl-2025, Ref. 30.

REFERENCES

[1] J. Pantaleone, "Stability of incoherence in an isotropic gas of oscillating neutrinos," Phys. Rev. D **58**, 073002 (1998).
[2] A. Sajadi, R. W. Kenyon, and B. Hodge, "Synchronization in electric power networks with inherent heterogeneity up to 100% inverter-based renewable generation," Nat. Commun. **13**, 2490 (2022).
[3] K. Wiesenfeld, P. Colet, and S. H. Strogatz, "Frequency locking in Josephson arrays: Connection with the Kuramoto model," Phys. Rev. E **57**, 1563–1569 (1998).
[4] D. Călugăru, J. F. Totz, E. A. Martens, and H. Engel, "First-order synchronization transition in a large population of strongly coupled relaxation oscillators," Sci. Adv. **6**, eabb2637 (2020).
[5] A. F. Taylor, M. R. Tinsley, F. Wang, Z. Huand, and K. Showalter, "Dynamical quorum sensing and synchronization in large populations of chemical oscillators," Science **323**, 614 (2009).
[6] M. Shafiei, S. Jafari, F. Parastesh, M. Ozer, T. Kapitaniak, and M. Perc, "Time delayed chemical synapses and synchronization in multilayer neuronal networks with ephatic inter layer coupling," Commun.Nonlinear Sci. Numer. Simul. **84**, 105175 (2020).
[7] P. J. Uhlhaas and W. Singer, "Neural synchrony in brain disorders: Relevance for cognitive dysfunctions and pathophysiology," Neuron **52**, 155–168 (2006).
[8] A. Correa, S. Alguacil, L. F. Ciria, A. Jiménez, and M. Ruíz, "Circadian rhythms and decision-making: A review and new evidence from electroencephalography," Chronobiol. Int. **37**, 520–541 (2020).
[9] C. A. Symanski, J. H. Bladon, E. T. Kullberg, P. Miller, and S. P. Jadhav, "Rhythmic coordination and ensemble dynamics in the hippocampal-prefrontal network during odor-place associative memory and decision making," eLife **11**, e79545 (2022).
[10] C. Bick, M. Goodfellow, C. R. Laing, and E. A. Martens, "Understanding the dynamics of biological and neural oscillator networks through exact mean-field reductions: A review," J. Math. Neurosci. **10**, 9 (2020).
[11] M. J. B. Hauser, "Synchronization of glycolytic activity in yeast cells," Curr. Genet. **68**, 69–81 (2022).
[12] O. Kiehn, "Decoding the organization of spinal circuits that control locomotion," Nat. Rev. Neurosci. **17**, 224–238 (2016).
[13] E. Marder, "Neuromodulation of neuronal circuits: Back to the future," Neuron **76**, 1–11 (2012).
[14] E. Marder and D. Bucher, "Central pattern generators and the control of rhythmic movements," Curr. Biol. **11**, R986–R996 (2001).
[15] J. J. Tyson, R. Albert, A. Goldbeter, P. Ruoff, and J. Sible, "Biological switches and clocks," J. R. Soc. Interface **5**, S1–S8 (2008).
[16] A. J. Ijspeert, "Central pattern generators for locomotion control in animals and robots: A review," Neural Networks **21**, 642–653 (2008).
[17] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," Neural Comput. **25**, 328–373 (2013).

03 April 2025 11:34:27

© Author(s) 2025

[18]L. Potvin-Trottier, N. D. Lord, G. Vinnicombe, and J. Paulsson, "Synchronous long-term oscillations in a synthetic gene circuit," Nature **538**, 514–517 (2016).
[19]B. J. Rosier and T. F. de Greef, "Synthetic biology: How to make an oscillator," eLife **4**, e12260 (2015).
[20]J. Stricker, S. Cookson, M. R. Bennett, W. H. Mather, L. S. Tsimring, and J. Hasty, "A fast, robust and tunable synthetic gene oscillator," Nature **456**, 516–519 (2008).
[21]E. Chicca, F. Stefanini, C. Bartolozzi, and G. Indiveri, "Neuromorphic electronic circuits for building autonomous cognitive systems," Proc. IEEE **102**, 1367–1388 (2014).
[22]E. Angelidis, E. Buchholz, J. Arreguit, A. Rougé, T. C. Stewart, A. V. Arnim, A. Knoll, and A. J. Ijspeert, "A spiking central pattern generator for the control of a simulated lamprey robot running on SpiNNaker and Loihi neuromorphic boards," Neuromorphic Comput. Eng. **1**, 014005 (2021).
[23]D. W. Franklin and D. M. Wolpert, "Computational mechanisms of sensorimotor control," Neuron **72**, 425–442 (2011).
[24]E. P. Frady and F. T. Sommer, "Robust computation with rhythmic spike patterns," Proc. Natl. Acad. Sci. **116**, 18050–18059 (2019).
[25]L. Ribar and R. Sepulchre, "Neuromorphic control: Designing multiscale mixed-feedback systems," IEEE Control Syst. **41**, 34–63 (2021).
[26]R. Sepulchre, G. Drion, and A. Franci, "Control across scales by positive and negative feedback," Annu. Rev. Control Rob. Auton. Syst. **2**, 89–113 (2019).
[27]W. Che and F. Forni, "Dominant mixed feedback design for stable oscillations," IEEE Trans. Autom. Control **69**, 1133–1140 (2023).
[28]T. Tsai, Y. S. Choi, W. Ma, J. R. Pomerening, C. Tang, and J. E. Ferrell, "Robust, tunable biological oscillations from interlinked positive and negative feedback loops," Science **321**, 126–129 (2008).
[29]O. Juárez-Álvarez and A. Franci, "Collective rhythm design in coupled mixed-feedback systems through dominance and bifurcations," arXiv:2401.04324 [math.DS] (2024).
[30]H. Jardón-Kojakhmetov, "Code for 'Co-evolutionary control of a class of coupled mixed-feedback systems'," (2025).
[31]C. Kuehn, *Multiple Time Scale Dynamics* (Springer, 2015), Vol. 191.
[32]P. Kokotović, H. K. Khalil, and J. O'reilly, *Singular Perturbation Methods in Control: Analysis and Design* (SIAM, 1999).
[33]K. D. Young, V. I. Utkin, and U. Ozguner, "A control engineer's guide to sliding mode control," IEEE Trans. Control Syst. Technol. **7**, 328–342 (1999).

03 April 2025 11:34:27

© Author(s) 2025