

Maximizing or Minimizing Chaos in Nonlinear Maps: A Kalman Filter–Guided Particle Swarm Optimization Approach

Fabio Berberi^{a,*}, Paolo Mercorelli^b

^a*University of Siena, Via Roma 56, 53100 Siena, Italy*

^b*Leuphana University of Lüneburg, Universitätsallee 1, 21335 Lüneburg, Germany*

Abstract

This paper presents a framework to identify hyperparameters that either maximize or minimize chaotic behavior in parameterized nonlinear systems. A Particle Swarm Optimization (PSO) engine explores the n -dimensional search space, while a nonlinear Kalman filter (Extended or Unscented) acts as a chaos detector. For each candidate, the system is simulated and the filter’s innovation statistics, local divergence, and permutation entropy are combined into a composite chaos metric. This metric serves as the PSO fitness, guiding particles toward regions of either maximal or minimal unpredictability, depending on the application. The approach couples the robustness of Kalman-based estimation with PSO’s global search, enabling efficient, filter-driven exploration for objectives ranging from cryptography and secure communications (chaos maximization) to control and stabilization of nonlinear systems (chaos minimization).

Keywords: Chaos, Particle Swarm Optimization, Kalman Filter, Nonlinear Dynamics, Permutation Entropy, Lyapunov Exponent

*Corresponding author. ORCID: 0009-0004-8825-8707

Email addresses: f.berberi@student.unisi.it (Fabio Berberi),
paolo.mercorelli@leuphana.de (Paolo Mercorelli)

¹ORCID (Mercorelli): 0000-0003-3288-5280

1. Introduction

Chaotic systems, with their sensitivity to initial conditions and nonlinear dynamics, are increasingly relevant in secure communications, cryptography, and random number generation. In control engineering, however, chaos is undesirable and must often be suppressed to improve stability.

Tuning system parameters to achieve high or low chaos is crucial, but traditional exhaustive or local search methods are inefficient in high-dimensional spaces. Recent studies have shown that combining *Particle Swarm Optimization* (PSO) with filtering techniques can improve estimation and optimization stability, particularly when Kalman-based methods are employed [1, 2, 3, 4]. In parallel, PSO variants enhanced with chaotic dynamics and hybrid strategies have demonstrated significant improvements in exploration and prevention of premature convergence [5, 6, 7].

Another key aspect is the quantification of chaos, where entropy- and Lyapunov-based measures have been widely used to characterize unpredictability and dynamical complexity [8, 9, 10, 11, 12, 13, 14, 15]. These indicators provide robust tools for distinguishing between regular and chaotic regimes in both discrete-time maps and continuous nonlinear systems.

Here, we propose a different perspective: PSO is coupled with a nonlinear *Kalman Filter* (EKF or UKF) used as a *chaos detector* rather than as a guide for swarm trajectories. By leveraging Kalman-based innovation statistics and classical chaos measures such as permutation entropy and Lyapunov exponents, the approach enables efficient optimization toward either maximization or minimization of chaotic behavior. This flexibility makes the method suitable for diverse applications, ranging from cryptography and secure communications to stabilization and control of nonlinear systems [16].

2. Proposed Approach

2.1. Problem Definition

Let $\mathcal{S}(\boldsymbol{\theta})$ be a parameterized nonlinear system, where $\boldsymbol{\theta} \in \mathbb{R}^n$ denotes the set of n hyperparameters. The objective is to determine the parameter vector $\boldsymbol{\theta}^*$ that either maximizes or minimizes a defined measure of chaotic behavior.

As illustrated in Figure 1, the chaotic behavior of the system is modeled by a function $\mathcal{F}_c$ with coefficient X . This coefficient is transformed through an operator $T(\cdot)$ into a new value X' , which can either:

- **Reduce** the chaotic intensity ($X' < X$), or
- **Increase** the chaotic intensity ($X' > X$),

depending on the optimization goal.

The framework accommodates both discrete-time maps and continuous-time nonlinear dynamical models, as well as any process capable of exhibiting chaotic dynamics. Two complementary optimization objectives are considered:

- **Chaos maximization:**

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

relevant for cryptography, secure communications, and random number generation.

- **Chaos minimization:**

$$\boldsymbol{\theta}^* = \arg \min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

relevant for stabilization in control systems, vibration reduction, or noise suppression.

The optimization relies on Particle Swarm Optimization (PSO) with the *Domain-as-Particle* structure. Kalman-enhanced PSO approaches have been developed to improve swarm guidance and stability. In this work, however, a nonlinear Kalman filter is employed as a *chaos detector*, providing the fitness $J(\boldsymbol{\theta})$ by simulating the system with candidate parameters, filtering outputs, and aggregating innovation variance, permutation entropy, and Lyapunov exponents.

2.2. Chaos Quantification via Kalman Filter

The chaos metric $J(\boldsymbol{\theta})$ is defined as a weighted combination of three indicators derived from the system's simulated response and processed through a nonlinear Kalman filter.

1. **Innovation variance** σ_{ν}^2 :

$$\sigma_{\nu}^2 = \frac{1}{T} \sum_{k=1}^T \nu_k^2, \quad (1)$$

where ν_k denotes the innovation at time step k and T is the simulation length.

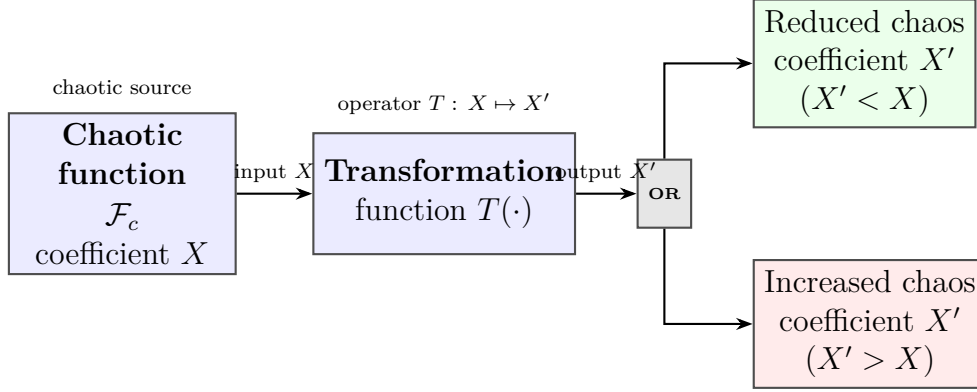


Figure 1: Transformation of a chaotic function $\mathcal{F}_c$ through operator $T(\cdot)$. The output coefficient X' can be reduced ($X' < X$) or increased ($X' > X$) depending on the optimization objective.

2. **Permutation entropy** H_{perm} : Normalized between 0 and 1 via the Bandt–Pompe method over sliding windows.
3. **Maximum Lyapunov exponent** λ_{max} : Estimates the average exponential divergence of nearby trajectories.

The composite chaos metric is:

$$J(\boldsymbol{\theta}) = w_1 \sigma_v^2 + w_2 H_{\text{perm}} + w_3 \lambda_{\text{max}}, \quad (2)$$

with nonnegative weights (w_1, w_2, w_3) . For chaos maximization, PSO increases $J(\boldsymbol{\theta})$; for minimization, it decreases it.

The Kalman filter (EKF or UKF) provides innovations and denoised state estimates for computing both λ_{max} and H_{perm} . Parameter-space exploration uses the *Domain-as-Particle PSO*, partitioning the search space into subdomains acting as independent PSO particles. The weights (w_1, w_2, w_3) are dynamically tuned via a *Meta-PSO*.

3. PSO Search Strategy

3.1. Global Domain Partitioning

We use the *Domain-as-Particle* approach. The search space (global domain $\mathcal{W}$) is divided into subdomains D_j arranged in a grid (Fig. 2). Each subdomain is separated by spacing s , allowing independent local optimization while preserving global movement.

Several studies highlight strategies to balance exploration and exploitation in PSO. For example, some approaches integrate chaotic dynamics. In contrast, the Domain-as-Particle method introduces structural exploration.

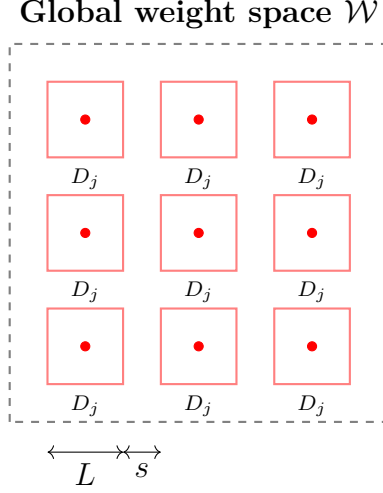


Figure 2: Weight-space partitioning: $\mathcal{W}$ is partitioned into a 3×3 grid of subdomains D_j .

3.2. Local Search Within Subdomains

Each subdomain D_j is treated as a single large particle and is equipped with a local PSO. The outer PSO updates the position of each D_j in $\mathcal{W}$, while the inner PSO searches for the optimal θ inside D_j . Initialization uses a uniform grid of candidate particles (Fig. 3).

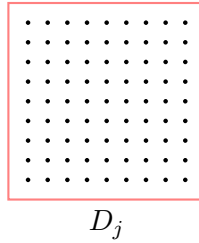


Figure 3: A subdomain D_j containing a uniform interior grid of candidate particles.

3.3. Characteristic Vector and Update Rule

At each global iteration, within D_j the top K particles (by $J(\theta)$) are selected to compute a *characteristic vector* $\mathbf{v}_j$. Domain updates combine

inertia, personal best, global best, and neighbor influence:

$$\Delta_p^j = \phi_p r_p (\mathbf{p}_j - \mathbf{c}_j^{(t-1)}), \quad \Delta_n^j = \phi_n r_n (\mathbf{n}_j - \mathbf{c}_j^{(t-1)}), \quad (3)$$

$$\Delta_g^j = \phi_g r_g (\mathbf{g} - \mathbf{c}_j^{(t-1)}), \quad \mathbf{v}_j^{(t)} = \omega \mathbf{v}_j^{(t-1)} + \Delta_p^j + \Delta_n^j + \Delta_g^j, \quad (4)$$

$$\mathbf{c}_j^{(t)} = \mathbf{c}_j^{(t-1)} + \mathbf{v}_j^{(t)}. \quad (5)$$

When a domain moves into a new region, a fresh grid of particles is generated and a new $\mathbf{v}_j$ is computed. Figure 4 illustrates domain motion and neighbor influence.

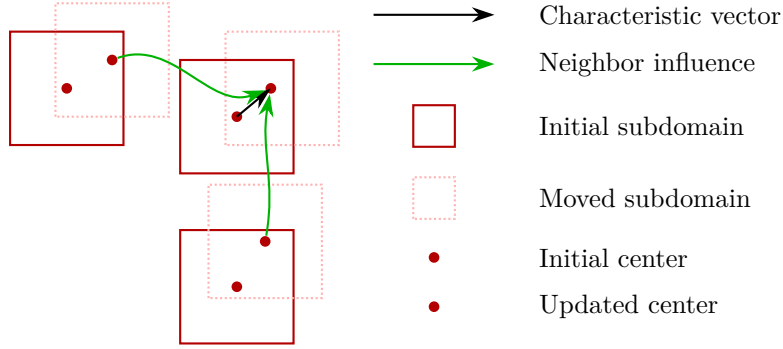


Figure 4: Characteristic vector with neighbor influence: initial subdomains (solid), moved subdomains (dotted), and centers.

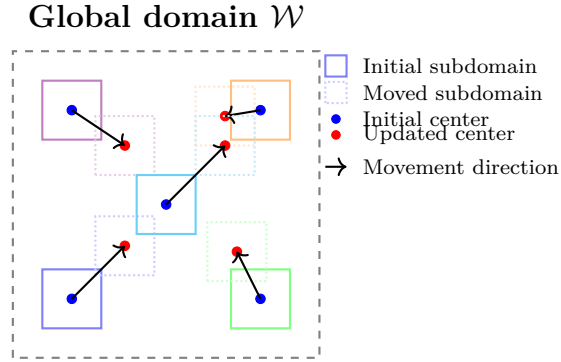


Figure 5: Domain movements inside $\mathcal{W}$ after one iteration.

4. Integration of Kalman and PSO

The framework integrates a Kalman filtering stage into the Domain-as-Particle PSO (Fig. 6).

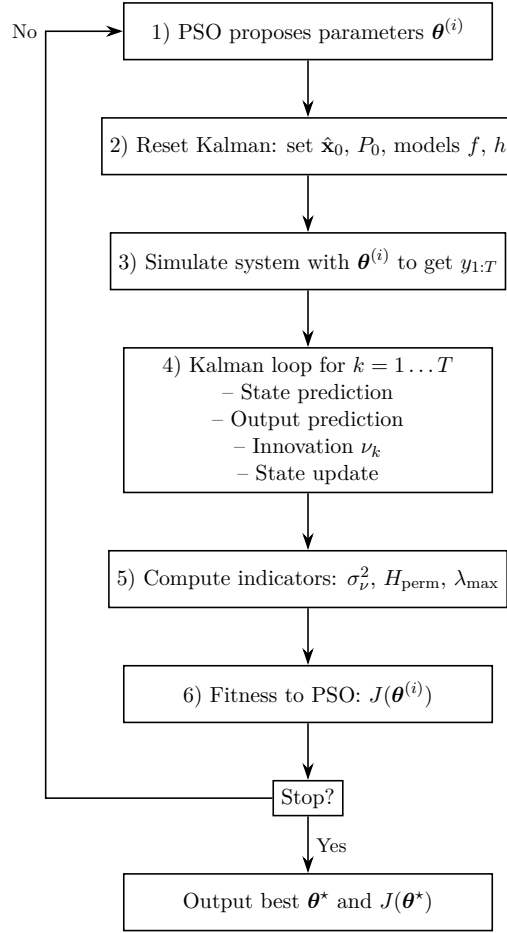


Figure 6: PSO–Kalman loop.

4.1. Stopping Criteria and Convergence

The algorithm stops when one or more criteria are met: (i) maximum global iterations $t_{\max}$; (ii) objective stagnation; (iii) velocity decay; (iv) time limit. Convergence is declared when velocity decay and objective stagnation occur simultaneously.

4.2. Computational Considerations

The Domain-as-Particle PSO uses an outer PSO over N_D domains and an inner PSO in each domain with K particles. With T_g global iterations and T_l local iterations, the complexity is $\mathcal{O}(T_g \cdot N_D \cdot T_l \cdot K \cdot n)$. The approach is highly parallelizable. Communication overhead for neighbor influence is modest relative to objective evaluations.

4.3. Potential Applications

Applications include hyperparameter optimization, feature selection, control tuning (maximization), and engineering design, energy reduction, error minimization (minimization). Advantages over classical PSO include preserved diversity, avoidance of premature convergence, and natural distributed computing.

5. Experimental Setup

To assess the effectiveness of the proposed framework, we conduct experiments on two canonical chaotic systems, representative of one-dimensional and two-dimensional nonlinear dynamics: the Logistic map and the Ikeda map, respectively. The objective is to determine parameter configurations that either amplify or suppress chaotic behavior, as quantified by the composite chaos metric

$$J(\boldsymbol{\theta}) = w_1 \sigma_\nu^2 + w_2 H_{\text{perm}} + w_3 \lambda_{\text{max}}. \quad (6)$$

The swarm optimization procedure employs the following settings: inertia weight $\omega = 0.7$, personal influence $\phi_p = 1.5$, neighborhood influence $\phi_n = 1.0$, and global influence $\phi_g = 1.5$, with random coefficients $r_p, r_n, r_g \sim U(0, 1)$. Each subdomain is initialized with 20 particles; after $T_{\text{local}} = 5$ local iterations, the top $K = 5$ particles are selected to update the characteristic vector. The global search space is partitioned into a 3×3 grid of nine subdomains, thereby ensuring balanced exploration and distributed diversity preservation.

The computational environment consists of Python 3.11, with numerical routines implemented in NumPy and SciPy. Permutation entropy is computed via the Bandt–Pompe approach, while maximum Lyapunov exponents are estimated using variational methods. The filtering stage employs a nonlinear Kalman filter implementation. Visualization is performed with Matplotlib. Hardware specifications are: Intel® i7-13700H CPU, 32 GB RAM, NVIDIA®

RTX 4060 GPU (not explicitly required). Each experimental configuration is repeated 30 times under independent random seeds, and results are reported as mean $\pm$ standard deviation to capture robustness across trials.

5.1. Comparative Results Overview

Figure 7 illustrates representative outcomes for chaos minimization and maximization in both the Logistic and Ikeda systems. Each trajectory highlights the progressive evolution of the chaos index $J(\theta)$, where minimization trends reveal convergence towards regularized regimes, while maximization trajectories accentuate divergence and unpredictability. The distinct profiles confirm that the proposed filter-guided PSO can effectively steer the search toward contrasting dynamical objectives.

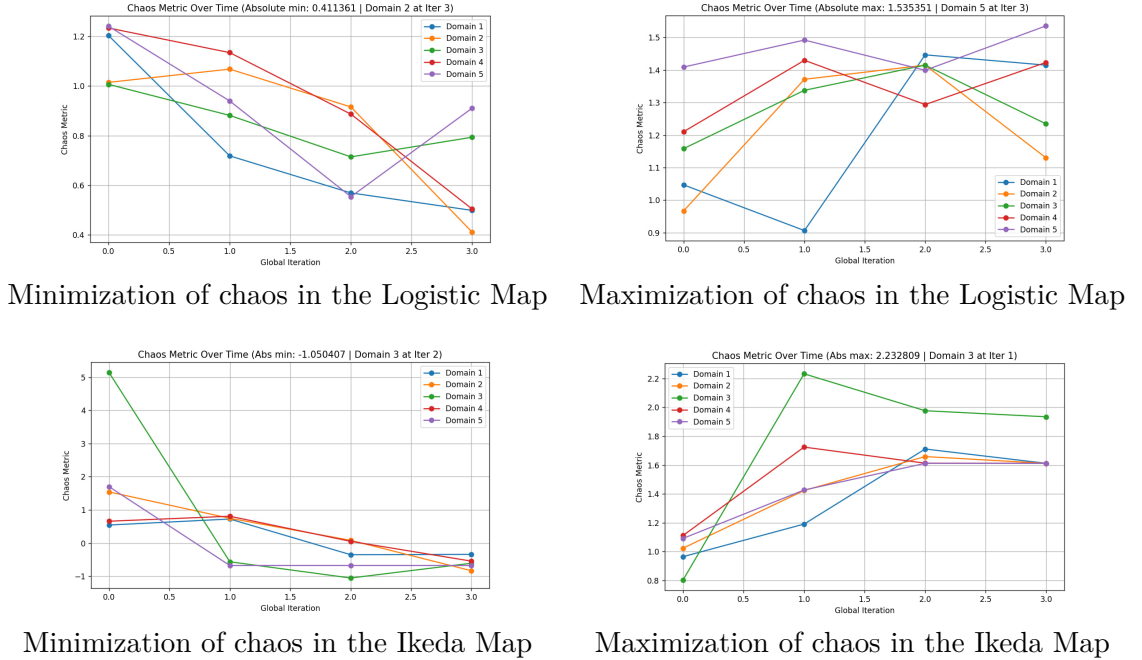


Figure 7: Comparative performance of the proposed framework on Logistic and Ikeda maps for both minimization and maximization objectives. Each panel depicts the best-performing trajectory of $J(\theta)$, showcasing the flexibility of the method in steering nonlinear dynamics toward desired chaotic regimes.

5.2. Convergence Analysis

The convergence behavior of the algorithm reveals heterogeneous dynamics across subdomains. For instance, in the Ikeda minimization task, a subset

of domains rapidly approaches a low-chaos solution, while others stagnate in suboptimal regions. Conversely, in the Logistic maximization scenario, several domains progressively enhance the chaos index, while others plateau prematurely. This divergence across domains underscores the value of the Domain-as-Particle paradigm in maintaining exploration and mitigating premature convergence, a recurring limitation in classical PSO frameworks.

5.3. Classical PSO vs. Domain-as-Particle PSO

To further quantify the benefits of the proposed approach, we compare it against classical PSO in the task of chaos maximization for the Logistic map. As shown in Figure 8, the Domain-as-Particle variant consistently identifies high-performing regions within the first iterations, whereas the standard PSO exhibits slower improvement and early stagnation. The experimental conditions for both algorithms are summarized in Table 1, ensuring fairness in the comparison.

Table 1: Parameter settings for Domain-as-Particle PSO versus Classical PSO in Logistic map experiments.

Domain-as-Particle PSO	Classical PSO
<code>global_bounds</code> = (3.5, 4.0)	<code>bounds</code> = (3.5, 4.0)
<code>num_domains</code> = 5	<code>dim</code> = 5
<code>num_particles</code> = 100	<code>num_particles</code> = 100
<code>top_k</code> = 10	<code>iterations</code> = 20
<code>local_iters</code> = 3	$w_1, w_2, w_3 = 1.0$
<code>global_iters</code> = 3	<code>inertia</code> (ω) = 0.7
$w_1, w_2, w_3 = 1.0$	$c_1, c_2 = 1.5$
—	<code>vmax_frac</code> = 0.1
—	<code>seed</code> = 42

6. Computation Time and Future Directions

Although the proposed Domain-as-Particle PSO demonstrates superior exploration capacity, its computational complexity is inherently higher than that of the classical PSO, owing to the two-level optimization and domain communication overhead. Nevertheless, the algorithm is highly parallelizable, making it compatible with modern multi-core and distributed computing infrastructures.

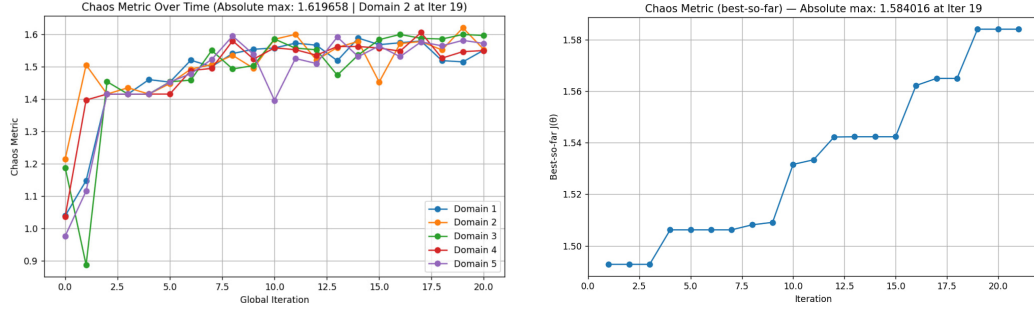


Figure 8: Performance comparison between Domain-as-Particle PSO (left) and Classical PSO (right) for chaos maximization in the Logistic map. The Domain-as-Particle variant exhibits faster convergence and superior exploration capabilities, whereas the classical approach shows slower improvement and premature stagnation.

To alleviate the computational burden, future research will explore surrogate-assisted strategies, whereby lightweight machine learning models (e.g., Random Forest, Gradient Boosting) are trained online to approximate candidate evaluations and pre-filter suboptimal regions. Such surrogate learning has already proven effective in heuristic optimization, agent-based calibration, and PSO variants with forecasting modules. Beyond methodological refinements, an additional avenue of application lies in cryptography and secure communications, where maximization of chaotic dynamics can contribute to enhanced unpredictability and resilience against adversarial attacks.

References

- [1] J. Song, L. Wang, et al., Maximum likelihood-based extended kalman filter for dynamic estimation in COVID-19 spread, *Chaos, Solitons Fractals* 146 (2021) 110922.
- [2] V. E. Papageorgiou, N. Papageorgiou, E. Sideris, An improved epidemiological-unscented kalman filter with dynamic parameter estimation, *Chaos, Solitons Fractals* 166 (2023) 112914.
- [3] D. B. Kara, M. E. Taha, et al., Degradation assessment of an igt with recurrence analysis and kalman filter based data fusion, *Chaos, Solitons Fractals* 186 (2024) 115224.

- [4] W. Yuan, X. He, et al., Stealthy fdi attacks on modified kalman filtering in cyber–physical systems, *Chaos, Solitons Fractals* 179 (2024) 114453.
- [5] B. Alatas, Chaos embedded particle swarm optimization algorithms, *Chaos, Solitons Fractals* 40 (4) (2009) 2004–2016.
- [6] L. dos Santos Coelho, A quantum particle swarm optimizer with chaotic mutation, *Chaos, Solitons Fractals* 37 (5) (2008) 1409–1418.
- [7] Y. Zhang, Q. Xu, et al., A particle swarm optimization algorithm with empirical balance, *Chaos, Solitons Fractals: X* 10 (2023) 100089.
- [8] S. J. Watt, Permutation entropy revisited, *Chaos, Solitons Fractals* 120 (2019) 95–99.
- [9] X. Zhao, Y. Zhang, J. Cao, W. Zhang, Permutation transition entropy: Measuring the dynamical complexity of financial time series, *Chaos, Solitons Fractals* 139 (2020) 109962.
- [10] B. R. R. Boaretto, A. A. F. Loureiro, L. Zunino, O. A. Rosso, Spatial permutation entropy distinguishes resting brain states, *Chaos, Solitons Fractals* 171 (2023) 113453.
- [11] H. Li, J. Cao, et al., Determining the lyapunov exponent spectrum of fractional-order systems, *Chaos, Solitons Fractals* 168 (2023) 113167.
- [12] J. Gancio, G. Mindlin, et al., Lyapunov exponents and extensivity of strongly coupled dynamical systems, *Chaos, Solitons Fractals* 178 (2024) 114392.
- [13] J. H. Argyris, G. Faust, M. Haase, On the influence of noise on the largest lyapunov exponent, *Chaos, Solitons Fractals* 9 (6) (1998) 947–958.

- [14] T. Zhao, H. Chen, Y. Deng, Information fractal dimension of random permutation set, Chaos, Solitons Fractals 174 (2023) 113883.
- [15] T. Zhao, Y. Deng, Linearity in deng entropy, Chaos, Solitons Fractals 178 (2024) 114388.
- [16] L. Minati, M. Frasca, et al., Flatness-based real-time control of experimental analog chua's system, Chaos, Solitons Fractals 177 (2023) 114274.