University of Tehran Press

Industrial Management Journal

Online ISSN: 3115 - 7386

Homepage: https://imj.ut.ac.ir

Presenting a Global Optimization-based Dynamic Fuzzy Inference System (GODFIS) for online data stream prediction

Farzad Bahrami [1] ✉, **and Mohammad Hossein Alavidoost** [2]

1. Corresponding author, Assistant Prof., Department of Industrial Management, Faculty of Management, University of Arak, Arak, Iran. E-mail: f-bahrami@araku.ac.ir

2. Ph.D., Department of Industrial Management, Faculty of Management, University of Tehran, Tehran, Iran. E-mail: mostafa.alavidoost@gmail.com

Article Info

Article type:
Research Article

Article history:
Received June 07, 2026
Received in revised form July 14, 2026
Accepted September 19, 2026
Available online October 01, 2026

Keywords:
Evolving fuzzy systems, optimality, prediction, online data stream.

ABSTRACT

Objective: Real-world data streams exhibit nonlinear and non-stationary characteristics that create significant challenges for effective predictive modeling. While evolving fuzzy systems (EFS) can dynamically update their structure and parameters to address these challenges, existing methods primarily focus on system identification rather than the optimality of learned parameters. This paper aims to bridge this gap by proposing a novel Global Optimization-based Dynamic Fuzzy Inference System (GODFIS) designed to enhance prediction accuracy through optimized parameter estimation.

Methodology: The proposed GODFIS framework leverages two key concepts: recursive center-of-gravity and global parameter estimation. Under this framework, we introduce a new Global Optimization-based Evolving Clustering Algorithm (GOECA) to optimize clustering in the response space and accurately determine initial parameters. Furthermore, a novel noise elimination principle is incorporated to maintain response quality and mitigate the negative impact of noise on the system's knowledge base.

Results: To evaluate the performance of GODFIS, extensive experiments were conducted on standard benchmarking datasets. The empirical results demonstrate that the proposed model achieves significantly higher prediction accuracy in solving benchmark problems compared to state-of-the-art and existing evolving fuzzy approaches.

Conclusion: By integrating global parameter optimization with robust noise elimination, GODFIS effectively addresses the limitations of conventional evolving fuzzy systems in non-stationary environments. The findings confirm that optimizing learned parameters and refining the clustering process lead to superior and more reliable predictive performance in dynamic, real-world data streams.

Cite this article: Bahrami, F., & Alavidoost, M., (2026). Presenting a global optimization-based dynamic fuzzy inference system (GODFIS) for online data stream prediction, ***Industrial Management Journal***, 18(4), 10-47. https://doi.org/10.22059/imj.2026.412275.1008301

Publisher: University of Tehran Press.

DOI: https://doi.org/10.22059/imj.2026.412275.1008301

Introduction

In the contemporary era of big data and real-time analytics, the rapid growth of non-stationary and high-velocity data streams has created a strong demand for adaptive modeling frameworks (Angelov & Buswell, 2002; Xiaowei Gu et al., 2023; Lughofer, 2011b). At the same time, data-driven artificial intelligence approaches have increasingly outperformed traditional models and shown promising results across a wide range of forecasting applications(Faezirad et al., 2021; Faghihi Nezhad & Minaei, 2018; Kazemi et al., 2011). In such dynamic environments, predictive models must be capable of continuous learning, adapting to evolving patterns, and updating both their structure and parameters without requiring complete retraining. Evolving Fuzzy Systems (EFSs) have emerged as an effective solution to this challenge by combining online learning capability with the interpretability of fuzzy rule-based systems (Hagras, 2004; Pratama et al., 2017).

The foundational concepts of EFS, particularly recursive rule generation and local model adaptation, were pioneered by seminal works such as (Angelov & Filev, 2004) and (Lughofer, 2011b), who laid the groundwork for one-pass learning. More recently, researchers have focused on enhancing the robustness and structural compactness of these systems to handle complex challenges such as noise, concept drift, and non-stationarity (Lughofer et al., 2015; Pratama, Anavatti, Angelov, et al., 2014). Despite these significant advancements, establishing a balance between structural flexibility and global parameter optimality remains a challenging task. Methods proposed by authors such as (Pratama, Anavatti, & Lughofer, 2014) and (Angelov et al., 2018) have notably improved the stability of evolving fuzzy models; Nevertheless, the development of a unified framework capable of simultaneously achieving efficient online evolution and accurate global parameter optimization remains a significant open challenge in this field (Gu & Angelov, 2019; Gu et al., 2021; Hu et al., 2025). In particular, global parameter optimality during online learning has not yet received sufficient attention, which may adversely affect model robustness, especially in noisy and highly dynamic environments (Angelov et al., 2018).

To bridge these gaps, this paper introduces a novel Global Optimization-based Dynamic Fuzzy Inference System (GODFIS), in which each core component is systematically engineered to address the specific limitations of conventional EFS architectures. First, to overcome the sub-optimality associated with local learning in standard evolving models, the Global Optimization-based Evolving Clustering Algorithm (GOECA) is proposed to optimize the antecedent parameters. GOECA employs a recursive center-of-gravity update that adaptively reshapes cluster boundaries, thereby establishing a direct mapping between the structural deficiencies of existing methods and the optimization capabilities of the proposed framework. Second, GODFIS incorporates a global parameter estimation mechanism to optimize the consequent parameters more

effectively at the global level. Finally, to reduce the sensitivity to noise and outliers in streaming environments—which often causes rule explosion and weakens interpretability in algorithms such as ECM—the Noise Elimination Principle (NEP) is introduced as a dynamic pruning mechanism.

The remainder of this paper is organized as follows. Related Work Section reviews related work on evolving fuzzy systems. Materials and Methods Section presents the proposed GODFIS methodology. Results and Discussion Section reports the experimental results and comparative evaluations. Finally, the Conclusion section concludes the paper.

Literature Background

The development of evolving fuzzy systems began with early studies on online identification and incremental rule learning for adaptive fuzzy modeling. Among the pioneering contributions, EFuNN (Kasabov, 2001) employed local learning for fast online processing, while (Angelov & Buswell, 2002) introduced a non-iterative incremental approach for evolving fuzzy system identification. Around the same time, DENFIS (Kasabov & Qun, 2002) was proposed as a data-stream-oriented framework based on recursive clustering and weighted recursive least squares for updating both rule structures and consequent parameters. A major milestone in the field was the introduction of online Takagi–Sugeno modeling strategies. (Angelov & Filev, 2004) proposed an online identification method for Takagi–Sugeno fuzzy models, which laid the groundwork for subsequent single-pass and streaming-based approaches. This model laid the foundation for subsequent developments such as Simpl_eTS (Angelov & Filev, 2005), eXtended eTS (xTS) (Angelov & Zhou, 2006), eTS+ (Angelov, 2010), an evolving classifier (Angelov & Zhou, 2008) based on the eTS model (Angelov & Zhou, 2008), and the DEC model (Baruah & Angelov, 2014). All these models use potential/density and dispersion criteria for input space clustering. In addition to these methods, other notable approaches in this field include:

FLEXFIS (Lughofer, 2008), SOFMLS (Rubio, 2009), eMG (Lemos et al., 2011a), McFIS (Subramanian & Suresh, 2012), SAFIS (Rong et al., 2006), ESAFIS (Rong et al., 2011), PANFIS (Pratama, Anavatti, Angelov, et al., 2014), GENEFIS (Pratama, Anavatti, & Lughofer, 2014), GSETSK (Nguyen et al., 2015), Gen-Smart-EFS (Lughofer et al., 2015), rFCM (Dovžan & Škrjanc, 2011b), rGK (Dovžan & Škrjanc, 2011a), eFuMo (Dovžan et al., 2012), ePL (Lima et al., 2010), ePL+ (Leandro Maciel et al., 2014), rPFM (L. Maciel et al., 2014), ePFM (Maciel et al., 2017). In recent years, researchers have developed more advanced models. (Pratama et al., 2017) introduced a Type-2 Recurrent Fuzzy Neural Network (eT2RFNN) to enhance model robustness in handling uncertain data. The Correntropy-based Evolving Fuzzy Neural System (CEFNS) introduced by (Bao et al., 2018) and its improved version, the Recursive Maximum Correntropy-based Evolving Fuzzy System (RMCEFS) (Rong et al., 2020), were proposed, utilizing additional error criteria. (Samanta et al., 2019) presented the Spatio-Temporal Fuzzy Inference System

(SPATFIS), which employs the maximum membership degree of input features for fuzzy rule generation and a similarity index for rule merging. A Self-Evolving Fuzzy System (SEFS) that uses online training errors to adjust a dynamic threshold for rule generation was introduced by (Ge & Zeng, 2019). In 2020, the same researchers proposed a conservative rule-adding strategy in the EFS-SLAT system (Ge & Zeng, 2020). Other notable recent studies include works by (Gu, 2021; X. Gu, P. Angelov, et al., 2023; Gu & Shen, 2021; Hu et al., 2024; Ožbot et al., 2023; Rodrigues & de Oliveira Serra, 2022; Santoso et al., 2020; Wang & Fei, 2022; Yu et al., 2022). For a comprehensive review of evolving methods, refer to key resources such as (de Campos Souza, 2020; Fernandez et al., 2019; Xiaowei Gu et al., 2023; Leite et al., 2020; Lughofer, 2011b; Škrjanc et al., 2019).

Comprehensive reviews (de Campos Souza, 2020; Xiaowei Gu et al., 2023; Lughofer, 2011b; Škrjanc et al., 2019) highlight that while EFSs excel in online adaptability and interpretability, many existing methods suffer from low global prediction accuracy due to the "unlearning effect" and suboptimal parameters resulting from purely exploratory, local adaptation mechanisms (Ge & Zeng, 2018). The lack of global parameter optimization often leads to information loss and reduced robustness. To address this, studies on system optimality and parameter stability have been considered (Rong et al., 2018). For instance, (Gu & Angelov, 2019) proposed separate algorithms for antecedent and consequent optimization, though performance gains remained limited to specific scenarios. More recently, evolutionary optimization techniques like Particle Swarm Optimization (PSO) have been integrated into EFSs to optimize parameters iteratively (e.g., PSO-ALMMo by Gu et al., 2021 (Gu et al., 2021), and MIIPSO-EFS by (Hu et al., 2025)). However, these metaheuristic approaches are computationally expensive and dependent on problem dimensionality, restricting their applicability to offline environments.

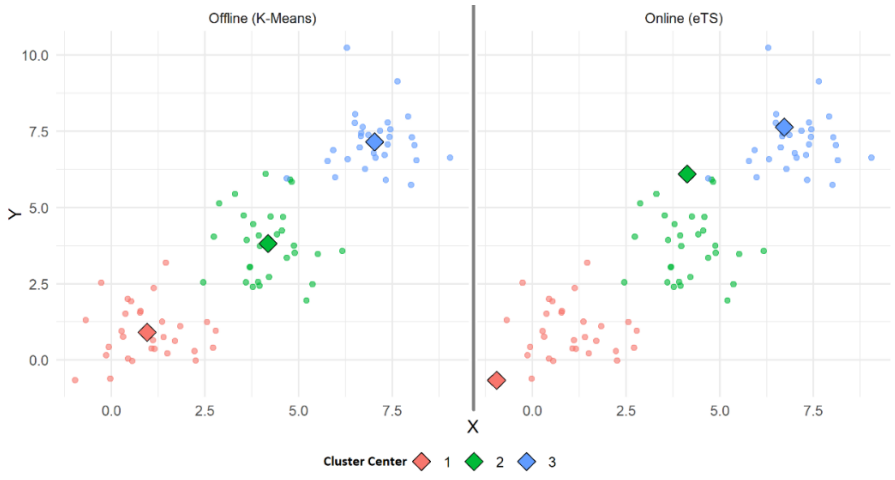

Figure 1. Comparison between cluster centers in offline and online methods

In summary, over the past two decades, EFSs have evolved from early local-learning models to advanced single-pass architectures robust to noise and concept drift (categorized into four generations in Table 1). Nevertheless, simultaneous global parameter optimization in fully online, real-time environments remains an open challenge.

Table 1. Developments in the Field of EFS over the Past Two Decades and Representative Models in Literature

Generation	Period	Primary Objective	Key Features	Limitations	Representative Models
Gen 1: Local learning and recursive clustering algorithms	Early 2000s	Development of the first online learning structures for stream data	• Local learning and incremental rule generation • Distance-based recursive clustering • Non-iterative training	• Sensitivity to noise • Challenges with imbalanced/complex data • Simple distance metrics	EFuNN (K. Kasabov, 2001), Non-iterative EFS (Angelov & Buswell, 2002), DENFIS (Kasabov & Qun, 2002)
Gen 2: eTS family models and structural developments	2004–2010	Integration of structure and parameter learning in a single pass	• Unsupervised structure learning and supervised parameter learning • Density/Distance-based rule generation • Fast updates without data storage • Evolutionary structure	• Dependency on density/distance criteria • Occasional excessive rule generation (rule explosion)	eTS (Angelov & Filev, 2004), Simpl_eTS (Angelov & Filev, 2005), xTS (Angelov & Zhou, 2006), eTS+ (Angelov, 2010), DeTS (Baruah & Angelov, 2014)
Gen 3: More robust and flexible models	2006–2017	Enhancing accuracy, stability, and noise robustness	• Stronger rule merging/pruning mechanisms • Better noise management • More stable recursive models • Increased structural flexibility	• Higher model complexity • Lack of global optimization • Potential for overfitting in complex data	FLEXFIS (Lughofer, 2008), SOFMLS (Rubio, 2009), eMG (Lemos et al., 2011b), PANFIS (Pratama, Anavatti, Angelov, et al., 2014), ePL+ (Leandro Maciel et al., 2014), ePFM (Maciel et al., 2017)
Gen 4: Models robust to noise, uncertain data, and complex structures	2017–Present	Managing high uncertainty, Type-2 fuzzy data, spatio-temporal structures, and concept drift	• Support for uncertain data • Spatio-temporal structures • Advanced similarity and entropy-based indices • Dynamic thresholds for rule generation/merging • Advanced stability analysis	• Increased computational complexity • Late convergence in some structures • Lack of precise joint optimization between structure and parameters	eT2RFNN (Pratama et al., 2017), CEFNS (Bao et al., 2018), SPATFIS (Samanta et al., 2019), EFS-SLAT (Ge & Zeng, 2020); (X. Gu, M. Li, et al., 2023; Hu et al., 2024)

To address this gap, this study proposes GODFIS (Global Optimization-based Dynamic Fuzzy Inference System), an evolving fuzzy framework designed to optimize both antecedent and consequent parameters in a fully online setting. Its main component, the Global Optimization-based Evolving Clustering Algorithm (GOECA), reformulates the classical Center-of-Gravity (COG) concept in a recursive manner to update antecedent parameters adaptively and without the need for historical data storage. At the same time, consequent parameters are estimated through a global Recursive Least Squares (RLS) procedure. In addition, a pruning mechanism referred to as the Noise Elimination Principle (NEP) is incorporated to remove noisy or weakly activated rules according to cluster quality and sample density. As a result, the proposed framework enhances prediction performance while preserving structural compactness and model stability (summarized in Table 2).

Table 2. Summary comparison between existing EFSs and the proposed GODFIS algorithm

Feature	Classical EFS (e.g., eTS, ePL, ePFM)	Offline-optimized EFSs (e.g., PSO-ALMMo)	Proposed GODFIS
Antecedent optimization	Exploratory clustering without precise optimization	Offline optimization requiring complete historical data	Online optimization without historical data
Consequent optimization	Local updating (wRLS) → moderate accuracy	Evolutionary optimization only in offline mode with very high computational cost	Online global optimization using RLS, coordinated with optimized antecedents
Rule pruning	- Not available (e.g., eTS) - Available (e.g., eTS+)	Usually lacks a rule pruning mechanism	Noise Elimination Principle: removal of noisy or rarely activated rules
Online applicability	Yes, but with lower accuracy and stability	No, limited to offline scenarios	Yes, with simultaneous preservation of accuracy, stability, and structural flexibility
Need for complete historical data.	No / Yes	Yes	No
Computational complexity	Low, but at the cost of reduced accuracy	High, due to evolutionary computations	Efficient and lightweight, owing to recursive formulations in both antecedent and consequent parts

Materials and Methods

As shown in Figure 2, A standard fuzzy rule-based system consists of four main components (Fuzzifier, Inference Engine, Knowledge Base, and Defuzzifier) (Fernandez et al., 2019). This study employs a Type-1 TS fuzzy model, where the i-th fuzzy rule structure is defined by Eq. (1).

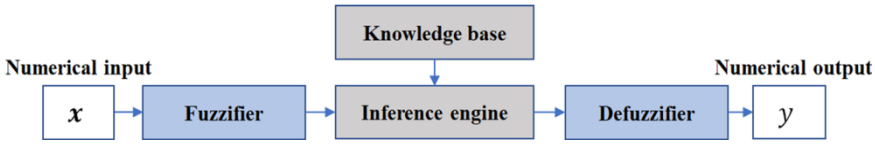

Figure 2. A standard system based on fuzzy rules

$$\mathcal{R}_i\text{: IF } \mathbf{x} \;\; is \;\; \tilde{\mathcal{A}}_i \;\; \text{THEN} \quad y_i = \theta_i(\mathbf{x}) = \mathbf{x}_e^T\theta_i = \theta_{i0} + \theta_{i1}x_1 + \cdots + \theta_{in}x_n \tag{1}$$

Where, $\mathcal{R}_i$ represents the i-th fuzzy rule ($i = 1,2,\dots,r$), r is the total number of fuzzy rules in the system, $\mathbf{x} = [x_1, x_2, \dots, x_n]^T \in \Re^n$ is the system input vector, $\tilde{\mathcal{A}}_i = [\tilde{\mathcal{A}}_{i1}, \tilde{\mathcal{A}}_{i2}, \dots, \tilde{\mathcal{A}}_{in}]$ is the fuzzy set of the antecedent part of the i-th rule with membership functions $\mu_{ij}(x_j)$, and $y_i \in \Re$ is the output of the i-th rule. θ_{i0} and θ_{ij} $(j = 1, \dots, n)$ are the consequent parameters of the i-th rule.

The degree of firing strength (normalized DOF) of each fuzzy rule (γ_i) is calculated by Eq. *(2)*, and the final output is obtained through weighted aggregation of all rule outputs (Eq. *(3)*).

$$\gamma_i(\mathbf{x}) = \prod_{j=1}^{n} \mu_{ij}(x_j) \tag{2}$$

$$y = \sum_{i=1}^{r} \lambda_i * y_i = \sum_{i=1}^{r} \lambda_i * \mathbf{x}_e^T\theta_i\,; \qquad \lambda_i = \frac{\gamma_i(\mathbf{x})}{\sum_{j=1}^{r}\gamma_j(\mathbf{x})};\ \mathbf{x}_e^T = [1\ \mathbf{x}^{\mathrm{T}}] \tag{3}$$

The antecedent parameters are determined through input space clustering. Figure 3 shows a three-dimensional example (with two inputs x1 and x2 and one output y), where three clusters projected onto the input axes (x1 and x2) form the input space and rule antecedents (Lughofer, 2008).

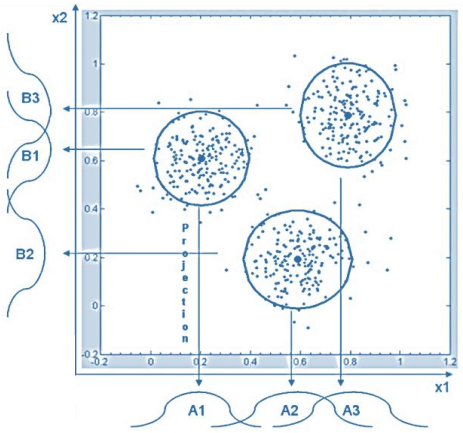

Figure 3. Horizontal projection of three clusters on the input axes x1 and x2

As previously mentioned, the TS model uses parameterized fuzzy regions, each associated with a local sub-model. The model's nonlinear behavior emerges from the intelligent, weighted combination of multiple local sub-models. It is worth noting that each local model's contribution to the final output directly depends on its Degree of Firing (DOF). For a better understanding of this mechanism, Figure 3 illustrates the function approximation approach of this model.

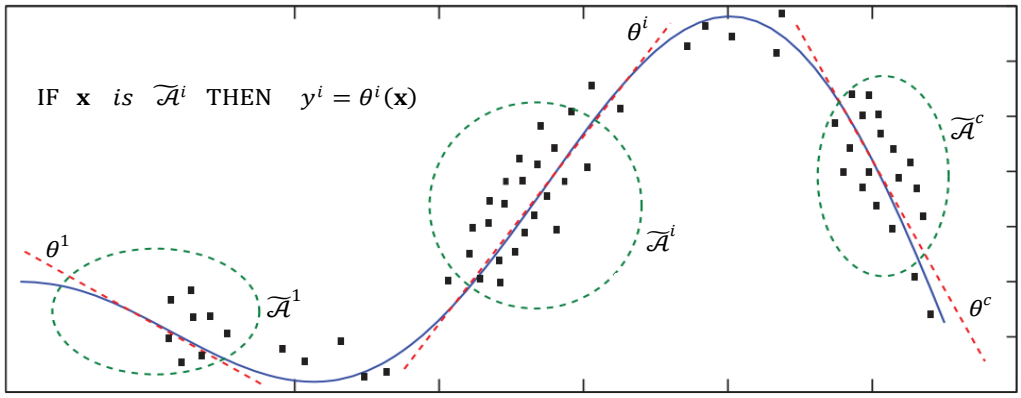

Figure 4. TS modeling for nonlinear function approximation (Maciel et al., 2017**)**

In fact, the core problem involves identifying the number of fuzzy rules r, the antecedent parameters, including cluster centers. $Cc_i = (Cc_{i,1}, Cc_{i,2}, \dots, Cc_{i,r})$ and radius parameters (spread measure) $\sigma_i = (\sigma_{i,1}, \sigma_{i,2}, \dots, \sigma_{i,r})$, and also the consequent parameters $\theta_i = (\theta_{i0}, \theta_{i1}, \theta_{i2}, \dots, \theta_{in})$ of the TS fuzzy system for predicting y. Therefore, the main research activities that scholars typically focus on implementing include:

1. Identification of Initial structure (determining number of clusters/rules) and antecedent parameter learning (for both offline and online models)
2. Estimation of consequent parameters (for both offline and online models)
3. Structure modification and parameter updates when needed (for online models only)

The following sections will examine the evolving learning method for both antecedent and consequent parts, and provide detailed explanations of the GODFIS learning algorithm.

Evolving clustering

For designing a fuzzy inference system, it is necessary to extract the rules of the system under study. The rules of fuzzy systems are extracted by "domain experts" or "clustering methods" or a combination of them. The first method is qualitative, meaning the rules may vary from one expert to another, and access to experts is not always possible. In contrast, the methods used in this research are quantitative, extracted based on system data, and do not require expert involvement.

However, the GODFIS model, due to its use of TS modeling features, allows experts to delete, add, or modify rules. Among evolving clustering algorithms, the ECM algorithm, introduced by (Kasabov & Qun, 2002) for single-pass and online clustering, has gained researchers' attention due to its high speed, simplicity, Also easy to understand, and high interpretability in various engineering fields (Hong & White, 2009; Lughofer, 2011a; Ravi et al., 2007; Soltic et al., 2004; Talei et al., 2013).

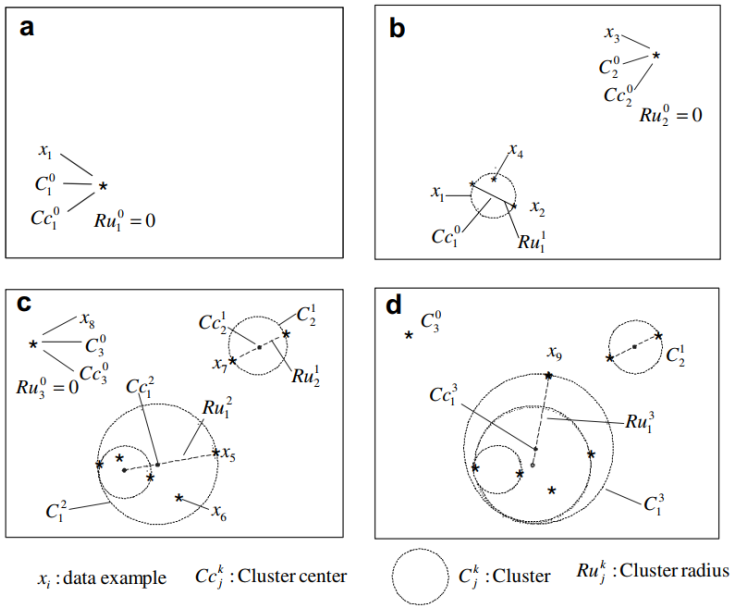

Figure 5. A dynamic clustering process using the online ECM algorithm in a two-dimensional space

ECM is a fast online clustering method that does not require predefined cluster numbers, as clusters automatically evolve based on their features from data streams. The following steps detail the ECM clustering process in a two-dimensional space (as shown in Figure 5):

- Step 1 – Creating the first cluster: The first cluster (C_1^0) is created using the first data sample, and its center (Cc_1^0) and radius ($Ru_1^0 = 0$) are determined (Figure *5-a*).
- Step 2 – Checking a new sample: If all samples have been checked, the clustering process ends. Otherwise, the distance between the current sample and existing cluster centers is calculated.
- Step 3 – Assigning the sample to an existing cluster: If the current sample falls within the range of an existing cluster, no new cluster is created, and existing clusters are not updated (e.g., points x_4 and x_6 in Figure *5*). The algorithm then returns to Step 2. Otherwise, it proceeds to Step 4.

- Step 4 – The value S_{ij} (distance of the new point from the cluster j-th + radius of the cluster j-th) is calculated for all clusters and compared with a predefined threshold (D_{thr}):
  - Creating a new cluster: If $S_{ia} = \min_j(S_{ij}) > 2D_{thr}$, a new cluster is created (e.g., points x_3 and x_8 in Figure *5*). Then, the algorithm returns to Step 2.
  - Assigning to the nearest cluster: If $S_{ia} = \min_j(S_{ij}) < 2D_{thr}$, the center of cluster a is shifted, and its radius is increased by $S_{ia}/2$ (e.g., points x_2, x_5, x_7 and x_9 in Figure *5*). Then, the algorithm returns to Step 2.

As observed, although the algorithm does not store information about previous samples, the maximum distance between a cluster center and its farthest sample does not exceed the threshold. This threshold is a parameter that affects the estimated number of clusters.

Although ECM is efficient for online clustering, it has three main limitations. First, its greedy and local update strategy can lead to sub-optimal cluster centers and reduced prediction accuracy. Second, its assumption of a uniform radius across dimensions forces clusters into hyper-spherical shapes, which is unsuitable for non-spherical data and can reduce interpretability. Third, ECM lacks an effective pruning mechanism, so obsolete or weak clusters may accumulate over time, increasing complexity and sensitivity to noise. To address these issues, this paper proposes GOECA. The algorithm performs online optimization of cluster parameters, allows dimension-wise variable radii to form hyper-ellipsoidal clusters, and includes a pruning mechanism to remove obsolete clusters and control rule growth.

GOECA Clustering Algorithm

In most offline optimization algorithms like K-Means or FCM, the cluster center is defined as the weighted mean of all points within the cluster. In fact, this center is the point that is considered the center of gravity of the cluster. Note that the definition of "weight" may vary across these algorithms. In contrast, many online clustering algorithms (such as eTS, xTS, eTS+, ePL, ePL+, and ePFM) typically consider the data points themselves as cluster centers. This approach differs from the concept of the center of gravity in optimization, because the optimal cluster center point is usually not located exactly on the input data. In addition, online methods obtain cluster centers exploratively and do not have a separate step for cluster optimization, which reduces the overall accuracy of online algorithms. While some researchers have attempted to compute optimal clusters using classical methods (e.g., rFCM, rPFM), these algorithms lack evolving mechanisms and underperform in non-stationary and dynamic environments. Thus, a method is needed that preserves cluster optimality while remaining suitable for online applications.

Membership functions describe the degree of association of rules with a particular instance. Like many studies in this field, GODFIS employs Gaussian membership functions due to their generalizability and ability to cover the entire feature space (Angelov & Zhou, 2008; Xiaowei Gu et al., 2023). The membership function is defined by Eq. (4):

$$\mu_{ij}(x_j) = \exp^{-\frac{(x_j - Cc_{ij})^2}{2\sigma_{ij}{}^2}} \quad (4)$$

Here, Cc_{ij} represents the cluster center, and σ_{ij} denotes the radius of the membership function (influence region or extent of membership function). Given its resemblance to the normal distribution, the radius can also be interpreted via standard deviation.

Updating clusters involves adjusting their centers and radius. To evaluate a new point's impact on the knowledge base, the effect of this point must be evaluated using specific criteria. In GOECA (like ECA), cluster structures are distance-based, but GOECA eliminates dimensionality effects from the distance criteria, making thresholds dimension-independent (Eq. (5)):

$$D_i^k = \frac{\left\| Cc_i^k - \mathbf{x}^k \right\|_d}{n}, \quad \forall i \quad (5)$$

Where, $\| . \|_d$ denotes the distance norm.

Three scenarios exist for cluster updates:

1) Point within an existing cluster: Only the center updates; the radius remains unchanged.
2) Point outside clusters but assigned to the nearest: If $\mathrm{Min}(D_i^k) < D_{thr}$, both center and radius update.
3) Point far from all clusters: If $\mathrm{Min}(D_i^k) > D_{thr}$, a new cluster forms with the new point as its center and an initial radius of zero.

Offline algorithms typically compute centers via weighted averages (Eq. (6)):

$$Cc_{ij}^k = \frac{\sum_{t=1, \mathbf{x}^t \in C_i}^{k} x_j^t}{nPop_i^k}, \quad \forall i, j \quad (6)$$

This requires access to all historical data—infeasible for online/big-data applications. To address this, Eqs. (7)-(8) reformulate the calculation recursively:

$$Cc_{ij}^{k+1} = \frac{\sum_{t=1,\mathbf{x}^t\in C_i}^{k+1} x_j^t}{nPop_i^{k+1}} = \frac{\sum_{t=1,\mathbf{x}^t\in C_i}^{k} x_j^t + x_j^{k+1}}{nPop_i^k + 1} = \frac{\sum_{t=1,\mathbf{x}^t\in C_i}^{k} x_j^t}{nPop_i^k + 1} + \frac{x_j^{k+1}}{nPop_i^k + 1}$$
$$= \frac{\sum_{t=1,\mathbf{x}^t\in C_i}^{k} x_j^t}{nPop_i^k} \times \frac{nPop_i^k}{nPop_i^k + 1} + \frac{x_j^{k+1}}{nPop_i^k + 1} \quad (7)$$
$$= Cc_{ij}^k \times \frac{nPop_i^k}{nPop_i^k + 1} + \frac{x_j^{k+1}}{nPop_i^k + 1}$$

$$Cc_{ij}^{k+1} = Cc_{ij}^k \times \frac{nPop_i^k}{nPop_i^k + 1} + \frac{x_j^{k+1}}{nPop_i^k + 1} \quad (8)$$

New clusters initialize their centers with the first assigned point. Subsequent updates follow Eq. (8). When a new cluster is formed, the cluster center is the first point that created the cluster. From then on, whenever a new point is added to that cluster, the cluster center is updated according to the recursive Eq. (8), otherwise, the cluster center remains unchanged ($Cc_{ij}^{k+1} = Cc_{ij}^{k}$). This equation ensures the center will converge to the true COG while maintaining online capability.

When a new point falls outside the current radius boundary and causes the cluster center to update, the radius is adaptively adjusted according to the new conditions. In the offline ECM model proposed by (Kasabov & Qun, 2002), the cluster radius is defined as the distance between the cluster center and its farthest contained point. This same principle applies to the online scenario, where the radius is recursively calculated using Eq. (9)

$$S_{ia} = \min_j\left(S_{ij} = d_{ij} + Ru_j\right), \qquad Ru_a = S_{ia}/2 \quad (9)$$

The main limitation of this equation in radius calculation is that it assigns the same radius across all dimensions, producing only spherical clusters. This is often unsuitable for real-world data, where clusters rarely have perfectly uniform shapes. Furthermore, the cluster radius must be updated in accordance with the R-COG method. In the proposed approach, the distance from the cluster center to the farthest point in each dimension is considered as the radius. For simplicity, we first calculate the cluster center using Eq. (8), and then formulate the radius calculation method as recursive Eq. (10) based on recursive rules. It should be noted that the initial radius value for all dimensions is set to zero.

$$\sigma_{aj}^{k+1} = \mathrm{Max}\left(\sigma_{aj}^k, \left\|Cc_{aj}^{k+1} - x_j^{k+1}\right\|_d\right), \qquad \forall j \quad (10)$$

This approach enables continuous adaptation via centroid dynamics while overcoming spherical-cluster constraints and eliminating full data-history storage requirements.

Estimation of the consequent part in evolving models

Consequent parameters are typically estimated using either local or global methods. Local estimation updates the parameters of each rule independently, whereas global estimation updates all consequent parameters simultaneously. Empirical studies show Weighted Recursive Least Squares (wRLS)—a local method—is more prevalent than its global counterpart (standard RLS) in online learning (Ge & Zeng, 2018). While (Yen et al., 1998) demonstrated global methods' superior accuracy in offline settings, this advantage isn't consistently observed in online implementations. Some studies even report better local method performance in streaming scenarios (Angelov, 2010; Lughofer, 2008). The reason for this could be due to the knowledge base update process being done online and in a single-pass manner. (Ge & Zeng, 2018) warn that neglecting the interdependence between antecedent and consequent learning in online methods can lead to a "unlearning effect". This phenomenon can be interpreted as follows: suboptimal clustering generates compounded prediction errors. Since online methods typically update antecedents first followed by consequents, any inaccuracy in antecedent parameter identification propagates to consequent estimation, thereby amplifying the overall system error (Inaccurate antecedent identification → flawed consequent estimation → compounded prediction errors).

The main difference between the global and local parameter estimation is that in the global method, all consequent parameters are updated in an integrated manner, whereas in the local method, only the parameters related to the active cluster are adjusted, and the linear functions are optimized separately without considering the interaction between different rules. This feature causes that in the global method, the error caused by the incorrect detection of the antecedent part to affect the entire model structure and significantly reduce the prediction accuracy. Therefore, one of the main reasons why online methods prefer using local updates for the consequent part is precisely this issue of incorrect and suboptimal antecedent parameter identification.

Our experimental results indicate that the GODFIS algorithm, due to its precise cluster identification based on the concept of the COG and its effort toward cluster optimality, is significantly compatible with the global consequent update method. This finding aligns with the results presented by Yen et al. (1998), which were proposed for optimal conditions and offline learning. In other words, by employing the global consequent update approach, GODFIS delivers better outcomes compared to the local update method. Although GODFIS is fundamentally designed for global consequent updating with the aim of achieving an optimal model, it is also capable of utilizing the local consequent update method. In the following sections, both global and local consequent update models, as introduced by (Angelov & Filev, 2004), will be examined. These findings suggest that, when clusters are accurately identified and antecedent parameters are precisely calculated, the global update method can outperform the local update method. This

highlights the importance of precise clustering algorithms in fuzzy inference systems. For models with fixed antecedent parameters, the problem of estimating the parameters of the linear consequents can be formulated as a least squares error minimization problem (Angelov & Filev, 2004). This is achieved by replacing the summation operation in Eq. (3) with an equivalent vector representation of the outputs y, as shown in Eq. (11).

$$y = \psi^T \theta \tag{11}$$

Where, $\theta = [\theta_1^{\mathrm{T}}, \theta_2^{\mathrm{T}}, \dots, \theta_{\mathrm{r}}^{\mathrm{T}}]^{\mathrm{T}}$ is the vector of linear model parameters, and $\psi = [\lambda_1 x_{\mathrm{e}}^{\mathrm{T}}, \lambda_2 x_{\mathrm{e}}^{\mathrm{T}}, \dots, \lambda_{\mathrm{r}} x_{\mathrm{e}}^{\mathrm{T}}]^{\mathrm{T}}$ represents the weighted input vector with normalized rule DOFs.

For consequent updating, we minimize the error function. Given input-output data pairs $(x_{\mathrm{k}}^{\mathrm{T}}, y_{\mathrm{k}}), \mathrm{k} = [1, \mathrm{t}]$, the optimal parameter vector θ that minimizes the objective function is calculated using Eq. (12):

$$E_G = \sum_{k=1}^{t} (y_k - \psi_k^T \theta)^2 \tag{12}$$

Where $\psi_{\mathrm{k}} = \left[\lambda_1(x_{\mathrm{k}}) x_{\mathrm{ek}}^{\mathrm{T}}, \lambda_2(x_{\mathrm{k}}) x_{\mathrm{ek}}^{\mathrm{T}}, \dots, \lambda_{\mathrm{r}}(x_{\mathrm{k}}) x_{\mathrm{ek}}^{\mathrm{T}}\right]^{\mathrm{T}}$; $x_{\mathrm{ek}} = \left[1, x_{\mathrm{k}}^{\mathrm{T}}\right]^{\mathrm{T}}$. Alternatively, the vectorized form of Eq. (12) can be reformulated as Eq. (13)

$$E_G = (Y - \Psi^T \theta)^T (Y - \Psi^T \theta) \tag{13}$$

Where, Y is an $(\mathrm{m} \times \mathrm{t})$ dimensional matrix composed of y_{k} values, Ψ is an $\mathrm{r} \times (\mathrm{n}+1) \times \mathrm{t}$ dimensional matrix formed by $\psi_{\mathrm{k}}^{\mathrm{T}}$, and θ is an $\mathrm{r} \times (\mathrm{n}+1) \times \mathrm{m} \times \mathrm{t}$ dimensional matrix generated from instantaneous values. The parameter vector θ that minimizes Eq. (13) is computed via the pseudo-inverse matrix (Eq. (14)).

$$\theta = (\Psi^T \Psi)^{-1} \Psi^T Y \tag{14}$$

In recursive parameter estimation (upon arrival of each new data point), the consequent coefficients are updated through the following recursive Eqs. (15)-(16):

$$\hat{\theta}_k = \hat{\theta}_{k-1} + C_k \psi_k \left(y_k - \psi_k^T \hat{\theta}_{k-1}\right) \tag{15}$$

$$C_k = C_{k-1} + \frac{C_{k-1} \psi_k \psi_k^T C_{k-1}}{1 + \psi_k^T C_{k-1} \psi_k}, \quad k = [1, t] \tag{16}$$

Eq. (15) and (16) are computed recursively at each arrival of a new data point. This algorithm starts with the initial conditions $\hat{\theta}_0 = 0$ and $C_0 = \Omega I$, where Ω is a large positive number (usually $\Omega = 1000$) and I is the identity matrix. C_k is a covariance matrix with dimensions $r(n+1) \times r(n+1)$, and $\hat{\theta}_k$ represents the estimated coefficients of the consequent functions based on the first k data samples. The recursive Kalman filter algorithm can efficiently learn from new data in an online manner. This method updates the model parameters without requiring iterative calculations, re-optimization processes, or storage of historical data. However, this learning process assumes a fixed number of rules. As the number of rules increases, both the number of equations and the size of the covariance matrix grow. Therefore, the following adjustments must also be applied to the Kalman filter computations. The coefficients of the linear function for a new rule are obtained from the weighted average of the coefficients of the linear functions from the existing rules. For weighting these coefficients, the normalized DOF λ_i of each rule is used. In this case, we have the following (Eq. (17)):

$$\hat{\theta}_k = \theta_i = \left(\hat{\theta}_{1,k-1}, \hat{\theta}_{2,k-1}, \dots, \hat{\theta}_{r,k-1}, \hat{\theta}_{r+1,k}\right), \qquad \hat{\theta}_{r+1,k} = \sum_{i=1}^{r} \lambda_i \hat{\theta}_{i,k-1} \tag{17}$$

When a new rule is added at the k-th time, the covariance matrix C_k is modified according to Eq. (18).

$$C_k = \begin{bmatrix} \rho C_{k-1} & 0 \\ 0 & \Omega I \end{bmatrix}; \qquad \rho C_{k-1} = \begin{bmatrix} \rho\varsigma_{11} & \cdots & \rho\varsigma_{1,r(n+1)} \\ \cdots & \cdots & \cdots \\ \rho\varsigma_{r(n+1),1} & \cdots & \rho\varsigma_{r(n+1),r(n+1)} \end{bmatrix} \Rightarrow$$

$$\Rightarrow \qquad C_k = \begin{bmatrix} \rho\varsigma_{11} & \cdots & \rho\varsigma_{1,r(n+1)} & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \rho\varsigma_{r(n+1),1} & \cdots & \rho\varsigma_{r(n+1),r(n+1)} & 0 & \cdots & 0 \\ 0 & 0 & 0 & \Omega & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 0 & 0 & \Omega \end{bmatrix} \tag{18}$$

In this matrix, ς_{ij} represents the elements of the covariance matrix ($i = [1, r \times (n+1)]; j = [1, r \times (n+1)]$), and $\rho = (r^2 + 1)/r^2$ is a constant coefficient for these elements (for proof of these relations, refer to (Angelov & Filev, 2004)). In the modified matrix, the part related to the

$(r+1)$-th rule —specifically, the last $n+1$ rows and columns— is initialized similarly to the initial covariance matrix C_0. The large value Ω is placed along the portion of the main diagonal, starting from row and column $(r(n+1)+1) \times (r(n+1)+1)$ up to the last row and column of the matrix.

With this adjustment, the covariance matrix is approximated as if it had originally been formed with $r+1$ rules from the beginning.

The objective functions in Eq. (12) and (13) are globally optimal; however, meaningful local sub-models can be identified using a locally weighted objective function (Angelov, 2010; Angelov & Filev, 2004). An approximate solution that minimizes the cost function is given by Eq. (19), where, by assuming independence or weak interaction levels between the linear subsystems, the error function can be decomposed into the sum of individual cost functions.

$$E_L = \sum_{i=1}^{r} E_{Li}, \qquad E_{Li} = (Y - X^T\theta_i)^T \Lambda_i (Y - X^T\theta_i) \tag{19}$$

In this Equation, matrix X is composed of row vectors. x_{ek}^T ($X \in R^{t\times(n+1)}$). The matrix Λ_i is a diagonal matrix in which $\lambda_i(x_k)$ values appear as the elements on its main diagonal.

The solutions θ_i that minimize the weighted least squares problems defined by the objective functions E_{Li} can be obtained by applying a weighted pseudo-inverse (as shown in Eq. (20)).

$$\theta_i = (X^T\Lambda_i X)^{-1} X^T \Lambda_i Y, \qquad i = [1, r] \tag{20}$$

Alternatively, a set of solutions for each cost function E_{Li} (the θ_i vectors) can be recursively computed using the wRLS algorithm, as given by Eq. (21) and (22), with the initial conditions $\hat{\theta}_0 = 0$ and $c_{i0} = \Omega I$.

$$\hat{\theta}_{ik} = \hat{\theta}_{ik-1} + c_{ik} x_{ek} \lambda_i(x_k)\left(y_k - x_{ek}^T \hat{\theta}_{ik-1}\right) \tag{21}$$

$$c_{ik} = c_{ik-1} + \frac{\lambda_i(x_k) c_{ik-1} x_{ek} x_{ek}^T c_{ik-1}}{1 + \lambda_i(x_k) x_{ek}^T c_{ik-1} x_{ek}}, \quad k = [1, t] \tag{22}$$

As shown in Eq. (21) and (22), the wRLS update depends on the normalized DOF. When $(\lambda_{i^*}(x_k) = 1)$, the algorithm reduces to standard RLS and updates the parameters using only rule i^*. When $\lambda_{i^o}(x_k) = 0$, neither the parameters nor the covariance matrix are updated. For $0 <$

$\lambda_i(x_k) < 1$, both are updated proportionally to the weight of the corresponding rule. This flexible mechanism enables the algorithm to intelligently differentiate between various rules in the fuzzy system and apply updates proportionally to each rule's contribution to the final output.

It is important to note that directly applying standard RLS may lead to non-convergence, as the original RLS algorithm is designed under the assumption of a fixed system structure (Angelov, 2010). However, the approach proposed in this study incorporates distinctive features that address this challenge.

Unlike many previous methods where a fuzzy rule may be replaced by another or where rule centers undergo noticeable changes with new incoming data (such as in eTS, xTS, eTS+, ePL, ePL+, ePFM, etc.), in the proposed approach, the structure gradually converges to its optimal and stable points as new data arrive. This results in significantly fewer structural changes relative to the sampling rate, which in turn allows for the direct use of RLS. This distinction is clearly reflected in the results, particularly in the comparison between local and global consequent update methods.

Cluster Quality Measurement

An evolving fuzzy system has a dynamic and open structure, meaning that its rule base can expand by adding new fuzzy rules or shrink by removing some. In other words, to avoid the problem of overfitting, several methods in the literature employ rule pruning techniques. In the GODFIS algorithm, NEP, as an online monitoring mechanism, is introduced, which is responsible for evaluating the quality of the clustering structure. This online monitoring method is an improved version of the Utility Criterion presented in (Angelov, 2010). The Utility criterion represents the accumulated relative DOF of a rule (Eq. (23)):

$$U_{ik} = \frac{\sum_{k=1}^{t} \theta_k}{k - I_i^*}, \quad k = [1, t] \tag{23}$$

Where I_i^* represents the time step at which an input data point becomes the focal point of the i-th fuzzy rule—that is, the time when the i-th fuzzy rule was generated.

The utility criterion provides insight into how useful a fuzzy rule is. The main goal of using this quality criterion is to avoid retaining unused clusters within the clustering structure—that is, to eliminate fuzzy rules that consistently exhibit low activation levels. To achieve this, the following rule elimination principle is adopted (Eq. (24)):

$$IF \quad U_{ik} < \epsilon \quad THEN \quad Prune \ \mathcal{R}_i, \quad r_k \leftarrow r_k - 1 \tag{24}$$

Where, $\epsilon \in [0.03,0.1]$ is a threshold used to control the utility of each cluster, and r_k denotes the number of rules in the rule base at time step k.

The utility principle focuses on retaining high-quality rules and eliminating those that have become obsolete (rules that were active for a short period but have not been used much afterward). However, this approach can sometimes conflict with the global optimality of the model. On the other hand, noisy data points may form new clusters, which not only increases the complexity of the GODFIS algorithm's structure but also leads to overfitting and adverse effects of noisy points on the model's structure. Therefore, to achieve global optimality, the utility principle needs to be revised and refined. In other words, rule elimination should be performed more cautiously to avoid deviating from global optimality. To address the issue of noisy points, this study proposes the NEP, which adds a new condition to the utility criterion, and the utility rule is revised according to Eq. (25) as the NEP.

$$IF \quad (U_{ik} < \epsilon) \quad AND \quad \left(nPop_{ik} < \varepsilon \times \frac{k}{r}\right) \quad THEN \quad Prune \ \mathcal{R}_i, \quad r_k \leftarrow r_k - 1 \qquad (25)$$

A cluster is considered noisy when both its utility is low and the number of data points assigned to it is below the threshold ($\varepsilon \times \frac{k}{r}$). In this regard, $\frac{k}{r}$ represents each cluster's share of total data (assuming that the data are uniformly distributed among the clusters). If the amount of data in a cluster is significantly less than this value, the probability that the cluster is created by noisy data increases. The value of $\varepsilon = 0.01$ is considered a suggested value, which is chosen based on the concept of three standard deviations (3σ) in the normal distribution. Of course, the choice of ε directly impacts the strictness of the Noise Elimination Process (NEP). In such a way that the smaller ε is, the stricter the noise elimination becomes and may lead to the generation of more rules. Conversely, the larger ε, the easier the noise elimination is, and the more rules are removed. The choice of ε directly impacts the strictness of the Noise Elimination Process (NEP). A smaller ε results in stricter noise elimination, potentially generating more rules. Conversely, a larger ε makes noise elimination more lenient, leading to the elimination of more rules.

Conceptual Integration of the Noise Elimination Principle with GOECA and RLS

In the proposed GODFIS framework, the Noise Elimination Principle is formulated as an integral component that operates coherently with both the antecedent and consequent adaptation mechanisms, with the primary objective of preserving structural consistency, numerical stability, and overall system optimality. In evolving fuzzy systems, each fuzzy rule simultaneously represents a data-driven cluster in the input space and a local parametric model in the output space; therefore, any rule pruning strategy must explicitly account for this dual representation. From the

antecedent perspective, the recursive GOECA clustering algorithm continuously updates cluster parameters in an online manner based on the recursive center-of-gravity concept, without requiring historical data. Within this process, the utility measure U_{ik} quantitatively reflects the representativeness and reliability of the cluster associated with the i-th fuzzy rule. Persistently low utility values indicate clusters that are either poorly formed or predominantly generated by noisy data fluctuations, whose retention may degrade the quality of the evolving antecedent structure. From the consequent perspective, rule parameters are estimated using a globally optimized RLS scheme that is fully synchronized with the evolving antecedent configuration. Given the sensitivity of RLS-based estimation to weakly supported models, fuzzy rules with insufficient activation frequency ($nPop_{ik}$) may introduce numerical instability and increase estimation variance. Accordingly, data population is incorporated as a complementary criterion to assess the statistical significance of each rule in the consequent learning process. By jointly enforcing the conditions of low cluster utility and insufficient data support, the Noise Elimination Principle selectively removes rules that are simultaneously insignificant from both structural (GOECA) and parametric (RLS) viewpoints. This dual-criterion pruning strategy effectively prevents premature elimination of informative rules during early learning stages, while suppressing the long-term accumulation of noisy or weakly contributing rules. Consequently, the proposed pruning mechanism acts as a stabilizing control layer between recursive clustering in the antecedent part and global RLS optimization in the consequent part. This coordinated interaction leads to a compact and well-conditioned rule base, improved numerical stability, and enhanced predictive performance in online and nonstationary environments. The conceptual diagram of the relationship between GOECA, RLS, and NEP is presented in Figure 6.

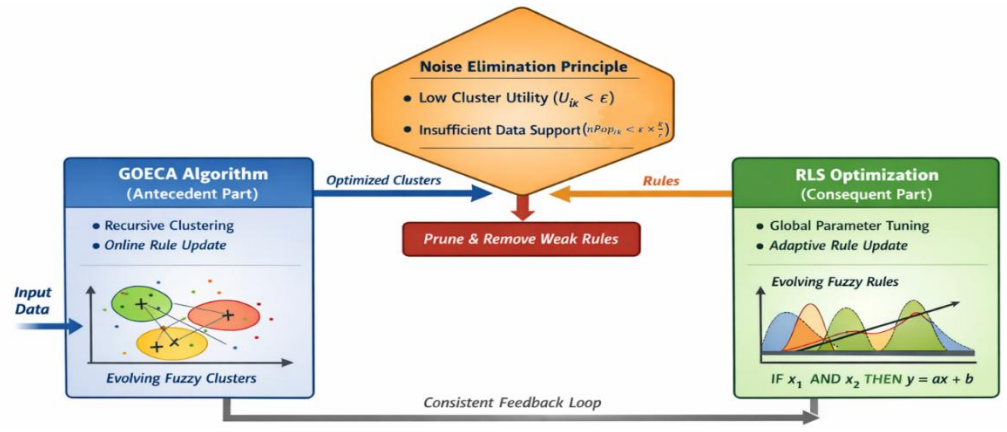

Figure 6. Conceptual form of the relationship between GOECA, RLS, and NEP

Also, the operational flowchart of the NEP is presented in the table below.

Noise Elimination Principle (Operational Procedure)

Input:

- Set of fuzzy rules at time k
- Utility threshold $\epsilon \in [0.03,0.1]$
- Elimination coefficient $\varepsilon = 0.01$
- Current number of rules r_k

Output:

- Updated fuzzy rule base
- Updated number of rules r_k

Begin

1. **for** each fuzzy rule $i = 1,2, \dots, r_k$ **do**
2. Compute the utility value. U_{ik}
3. **if** $(U_{ik} < \epsilon)$ **and** $\left(nPop_{ik} < \varepsilon \times \frac{k}{r}\right)$ **then**
4. Remove fuzzy rule i $(\mathcal{R}_i)$
5. Update the number of rules: $r_k \leftarrow r_k - 1$
6. **end if**
7. **end for**

End

GODFIS Algorithm

Based on the previous discussions, this section outlines the detailed steps of the GODFIS algorithm. GODFIS has an open and flexible structure that allows it to adapt and expand as the data patterns evolve. Its recursive nature also makes it computationally efficient. Rules are generated and updated in parallel with the clustering of the input space using the GOECA algorithm. Model training begins with an initial model estimation—that is, extracting r initial rules and estimating the parameters. θ_0 and C_0 as follows. Finally, the process continues until no new data are available.

Step 1: Initial Preparation: the training data are preprocessed through normalization and feature selection.

Step 2: Initial Structure Extraction: the initial rule base is extracted offline using GOECA, which defines the antecedent parameters, cluster centers, radii, and membership counts, followed by estimation of the TSK consequent parameters using either global or local learning.

Step 3: Online Learning: In online learning, each new sample is used to update the antecedent structure according to its relation to the existing clusters; it may update an existing center, update both center and radius, or create a new cluster. The consequent parameters are then updated using either RLS for global learning or wRLS for local learning.

Step 4: Complexity Reduction and Noise Robustness: The Noise Elimination Principle evaluates cluster utility and removes noisy clusters when necessary.

Stopping Condition: If no new data is available, the process ends; otherwise, return to Step 3.

Overall, GODFIS is characterized by adaptive rule-based evolution (growth and pruning), efficient recursive learning without full retraining, robustness to noise via noisy-cluster detection and removal, and synchronized optimization of antecedent and consequent parameters toward improved global performance. The next section reports comparative computational results against related methods, and Figure 7 illustrates the workflow of the proposed GODFIS.

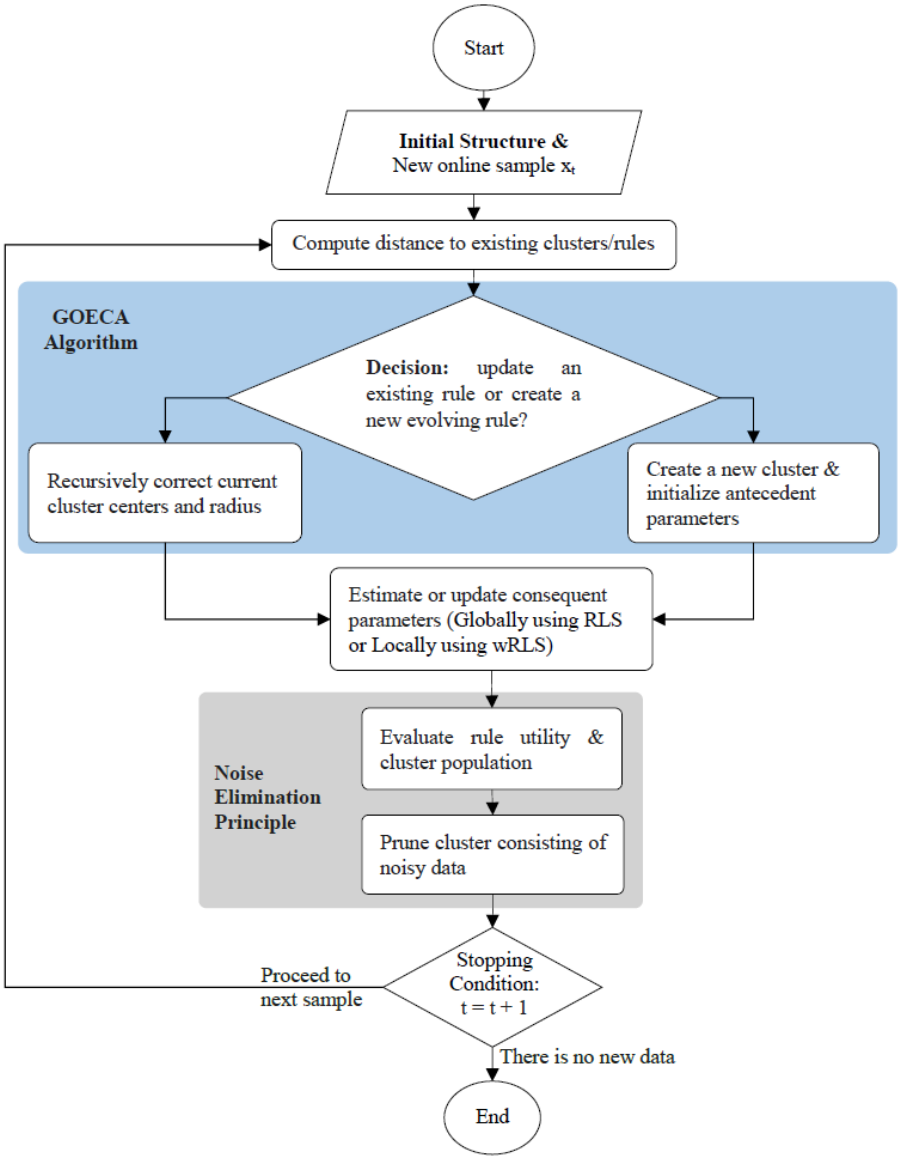

Figure 7. Flowchart of the proposed GODFIS algorithm

Results

This section evaluates the performance of the proposed GODFIS algorithm from three perspectives. First, GODFIS is compared with existing methods on benchmark datasets, including the Mackey–Glass time series, the Delta Ailerons dataset, and S&P 500 closing price data, in order to assess its predictive capability on nonlinear and widely used problems. Second, the effects of local and global consequent parameter updates are examined to determine their influence on overall system performance. Third, the effectiveness of the Noise Elimination Principle (NEP) is evaluated in terms of its ability to handle noisy data and its impact on model output.

The evaluation is based on four quantitative criteria: Root Mean Square Error (RMSE) – Eq. 0, Non-Dimensional Error Index (NDEI) – Eq. 0, the number of generated rules, and execution time. These metrics respectively measure prediction accuracy, cross-dataset comparability, structural complexity, and computational efficiency.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{26}$$

$$NDEI = \frac{RMSE}{std(y)} \tag{27}$$

Performance Evaluation on the MG Time Series

In this section, the performance of the GODFIS algorithm is evaluated against previously proposed algorithms in the literature using the MG (Mackey & Glass, 1977) dataset, which is a well-known chaotic time series prediction problem. The MG problem is considered a classical and widely used benchmark in the field of chaotic time series forecasting and serves as a standard reference for evaluating prediction models. Numerous studies have employed this dataset to assess the performance of evolving fuzzy systems (Ge & Zeng, 2018; Xiaowei Gu et al., 2023).

The MG time series is generated by solving the following delayed differential equation (Eq. 0):

$$\frac{dx(t)}{dt} = \frac{ax(t-\tau)}{1 + x^{10}(t-\tau)} - bx(t) \tag{28}$$

Where, $a = 0.2, b = 0.1, \tau \geq 17$. In this study, following common practice in prior research, the value $\tau = 17$ has been selected. A total of 3,500 input-output pairs are generated using the

fourth-order Runge-Kutta numerical method with an initial condition of $x(0) = 1.2$. The first 3,000 samples are taken from the time range $t = 201$ to 3200 (as a training dataset), and the remaining 500 samples are taken from the time range $t = 5001$ to 5500 (as test data). The prediction model is defined as follows (Eq. 0):

$$\hat{x}(t+85) = f\big(x(t), x(t-6), x(t-12), x(t-18)\big) \tag{29}$$

To compare the prediction performance of the models, three metrics are used (NDEI, Number of generated rules (#(Rule)), and Execution Time (t_{exe})). The test results of the proposed model are presented in Table 3, along with the results of previous models. Since the performance of evolving fuzzy systems is often sensitive to parameter settings, for a fair comparison, the results are taken directly from the literature (Ge & Zeng, 2018; Ge & Zeng, 2019; Xiaowei Gu et al., 2023; Rong et al., 2020; Samanta et al., 2019).

Table 3. Comparison of GODFIS with other methods in the literature on MG time series

Model	Model Type	NDEI	#(Rule)	t_{exe}
GODFIS	T-S	**0.106**	42	9.3
LEOA (Ge & Zeng, 2018)	T-S	0.248	42	144.8
ALMMo (Angelov et al., 2018)	T-S	0.402	9	0.5
PSO-ALMMo (Gu et al., 2021)	T-S	0.191	8	314.3
CEFNS (Bao et al., 2018)	T-S	0.264	5	0.4
DENFIS (Kasabov & Qun, 2002)	T-S	0.278	58	–
EFuMo (Dovžan et al., 2015)	T-S	0.139	41	–
EFS-SLAT (Ge & Zeng, 2020)	T-S	0.114	8	–
ESAFIS (Rong et al., 2011)	T-S	0.296	6	5.8
eTS (Angelov & Filev, 2004)	T-S	0.381	9	3.1
eTS+ (Angelov, 2010)	T-S	0.392	10	–
Simpl_eTS (Angelov & Filev, 2005)	T-S	0.394	11	4.4
Simpl_eTS+ (Angelov, 2011)	T-S	0.375	15	–
GENEFIS (Pratama, Anavatti, & Lughofer, 2014)	T-S	0.28	19	4.5
GSETSK (Nguyen et al., 2015)	T-S	0.347	19	–
McFIS (Subramanian & Suresh, 2012)	T-S	0.25	10	8.2
PANFIS (Pratama, Anavatti, Angelov, et al., 2014)	T-S	0.301	19	4.5
RMCEFS (Rong et al., 2020)	T-S	0.117	5	**0.3**
SAFIS (Rong et al., 2006)	RBF	0.376	6	–
SEFS (Ge & Zeng, 2019)	T-S	0.129	4	0.4
eT2RFNN (Pratama et al., 2017)	T-S	0.32	**3**	–
SPATFIS (Samanta et al., 2019)	T-S	0.279	30	–
DeTS (Baruah & Angelov, 2014)	T-S	0.44	**3**	–

As shown in Table 3, the GODFIS algorithm demonstrates the best performance in terms of the NDEI index compared to other models. This indicates that GODFIS is highly effective in accurately predicting chaotic time series data. However, GODFIS generates more fuzzy rules than some other algorithms and exhibits a longer execution time, although this time remains within an acceptable range. This behavior stems from the algorithm's architecture, which is designed to retain more historical information to ensure global optimization while maintaining high accuracy.

Considering that LEOA is the only other online model focused on the global optimization of EFS, GODFIS significantly outperforms it. Unlike LEOA, GODFIS achieves better accuracy (NDEI) and faster execution time while maintaining the same number of rules. In addition to LEOA, another relevant work is PSO-ALMMo, which also addresses global optimization in EFS, although it is not an online method and relies on a metaheuristic approach. Despite this difference, GODFIS achieves better prediction accuracy (NDEI) and lower computational time than PSO-ALMMo, although it produces more fuzzy rules in comparison.

Figure 8 and Figure 9 provide a visual comparison between the predicted and actual test data values. Figure 8 displays the same comparison using a scatter plot along the 45-degree line, where points closer to the 45-degree line indicate higher prediction accuracy. This visualization confirms the high precision and reliability of the GODFIS model in forecasting the MG dataset. Figure 9 presents the comparison between 500 test data points and the model's predictions. As shown, GODFIS accurately predicts most data points, with only a few minor deviations.

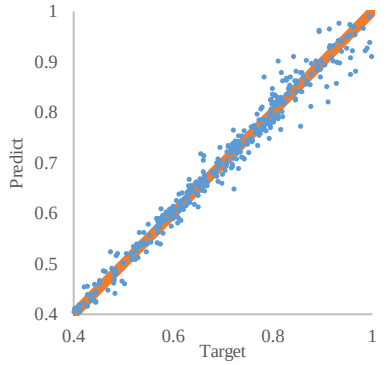

Figure 8. Comparison of predicted and actual values of the MG dataset on a 45-degree line

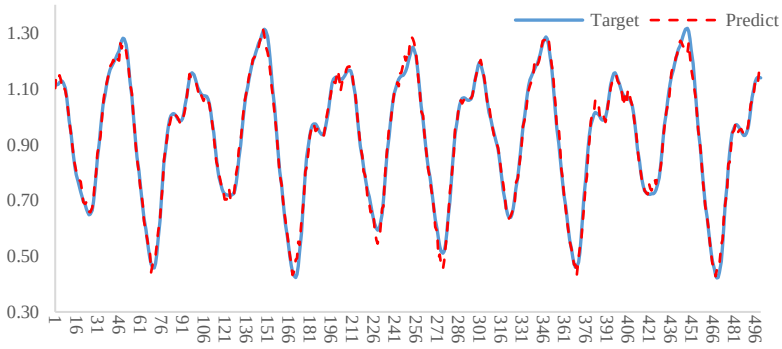

Figure 9. Comparison between predicted values and actual values for the first 500 data points on the MG dataset

Figure 10 presents the prediction error histogram for the test data, showing an approximately normal distribution centered around zero. This lack of systematic bias confirms the model's stability and prediction accuracy. Additionally, Figure 11 tracks the prediction error over time,

demonstrating that the absolute error remains below 0.09 throughout execution, with most test samples yielding errors below 0.06 or 0.04. These results indicate a substantial improvement in accuracy compared to the LEOA algorithm, which has been reported to produce absolute errors up to 0.3.

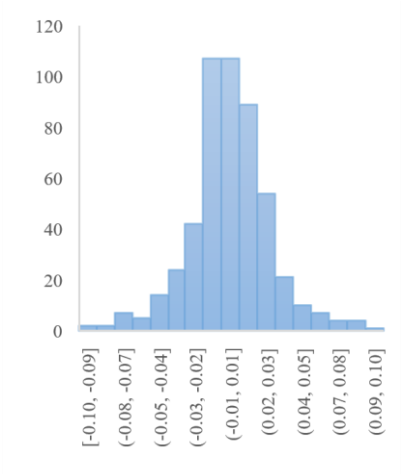

Figure 10. Histogram of prediction errors on MG test data

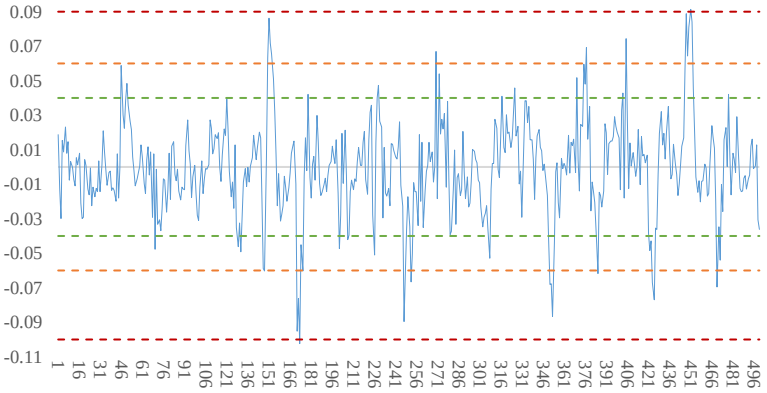

Figure 11. Prediction error trace on MG test data

Performance Evaluation on the Delta Ailerons dataset

To further evaluate the performance of the GODFIS algorithm, the Delta Ailerons dataset is used. This dataset is sourced from the KEEL[1] data repository and is related to the control surfaces of the F-16 aircraft (Alcalá-fdez et al., 2011). The model predicts changes in the aircraft's control surface (Sa) based on five input variables: RollRate, PitchRate, currPitch, currRoll, and diffRollRate.

Table 4 presents a comparative analysis of GODFIS with other EFSs reported in the literature on the Delta Ailerons dataset (Ge & Zeng, 2020; Xiaowei Gu et al., 2023; Gu et al., 2021; Rong et al., 2020). The comparison focuses on three key performance metrics: RMSE, Number of fuzzy rules, and Execution Time.

[1] Available at: https://sci2s.ugr.es/keel/datasets.php

Table 4. Comparison of GODFIS with other methods in the literature on the Delta Ailerons dataset

Model	Model Type	RMSE	#(Rule)	t_{exe}
GODFIS	T-S	0.0449	2	11.7
ALMMo (Angelov et al., 2018)	T-S	0.0513	10	0.4
CEFNS (Bao et al., 2018)	T-S	0.0502	3	0.3
DENFIS (Kasabov & Qun, 2002)	T-S	0.0497	11	–
EFS-SLAT (Ge & Zeng, 2020)	T-S	0.0412	8	–
ESAFIS (Rong et al., 2011)	T-S	0.0506	13	12.4
eTS (Angelov & Filev, 2004)	T-S	0.0513	4	2.1
McFIS (Subramanian & Suresh, 2012)	T-S	0.0509	15	–
RMCEFS (Rong et al., 2020)	T-S	0.0498	2	0.2
SAFIS (Rong et al., 2006)	RBF	0.0549	14	6.8
SEFS (Ge & Zeng, 2019)	T-S	0.0476	7	0.4
Simpl_eTS (Angelov & Filev, 2005)	T-S	0.0512	4	2

As shown in Table 4, the GODFIS algorithm demonstrates a competitive performance compared to other existing methods. Specifically, in terms of prediction accuracy (RMSE), GODFIS ranks second, following the EFS-SLAT algorithm. Regarding the number of fuzzy rules, GODFIS shares the top rank with RMCEFS. These results indicate that, despite its dynamic structure and online learning capability, GODFIS has been able to produce a compact and simple model while maintaining a satisfactory level of accuracy. Due to the global optimization focus embedded within GODFIS, a slightly longer runtime compared to some methods is expected and justifiable.

On the other hand, the PSO-ALMMo algorithm, which is an evolutionary version of ALMMo and limited to offline scenarios, has focused on achieving global optimization in EFS (the results of this algorithm are taken from (Gu et al., 2021). Although PSO-ALMMo is not an online model, the solution quality of GODFIS demonstrates a competitive performance compared to PSO-ALMMo (as shown in Table 5). While PSO-ALMMo achieves better accuracy in terms of the NDEI metric, placing GODFIS in second place, GODFIS significantly outperforms both competing algorithms in terms of the number of generated rules, offering a much more compact structure. Moreover, GODFIS operates considerably faster and more efficiently than PSO-ALMMo in terms of execution time. These results indicate that GODFIS is able to strike an effective balance between structural simplicity, execution speed, and model quality, while also preserving the key advantage of online learning and real-time adaptability to data changes—an advantage that offline algorithms such as PSO-ALMMo inherently lack.

Table 5. Comparison of GODFIS, ALMMo, and PSO-ALMMo on the Delta Ailerons dataset.

Model	Model Type	NDEI	#(Rule)	t_{exe}
GODFIS	T-S	0.5437	2	11.7
ALMMo (Angelov et al., 2018)	T-S	0.5499	10	0.5
PSO-ALMMo (Gu et al., 2021)	T-S	0.5351	10	362.4

Figure 12 shows a comparison between predicted values and actual test data values, with points plotted along the 45-degree line. Figure 13 also presents a comparison between predicted test data values and their actual values on the y-axis. For better clarity, the first 100 test data points were selected. As can be seen, except for a few points, the algorithm has predicted the points with high accuracy.

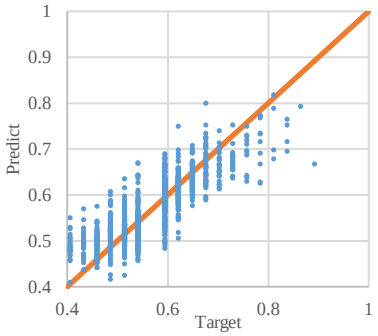

Figure 12. Comparison of predicted and actual values of Delta Ailerons test data on a 45-degree line

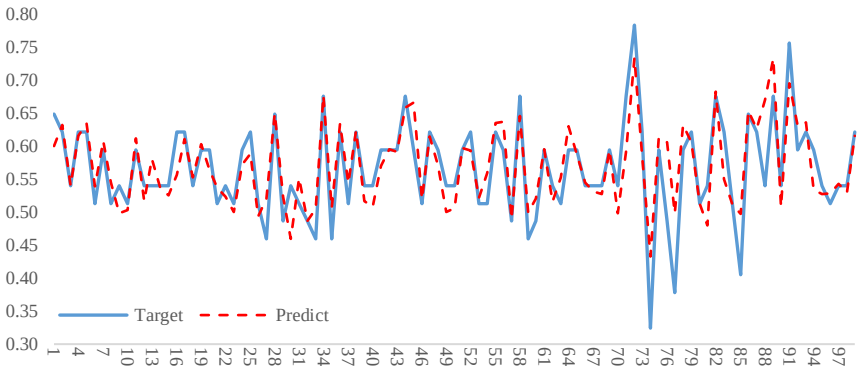

Figure 13. Comparison between predicted values and actual values on the Delta Ailerons dataset (first 100 data points).

Figure 14 presents the histogram of prediction errors for the test points. As observed, the error distribution is approximately normal. Figure 15 illustrates the prediction error trace of the GODFIS algorithm over the first 100 test points of the Delta Ailerons dataset. As shown, the absolute prediction error does not exceed 0.13 at any point. Moreover, in most instances, the error remains below 0.06, and in a significant portion, even below 0.04. This indicates the high stability and accuracy of the GODFIS algorithm in the prediction process, especially when dealing with real-world data under operational conditions. Overall, Figure 14 and Figure 15 confirm that in addition to precision, GODFIS offers reliable statistical prediction quality and well-distributed error behavior.

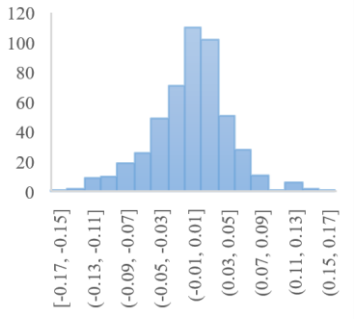

Figure 14. Histogram of prediction errors on the Delta Ailerons test data

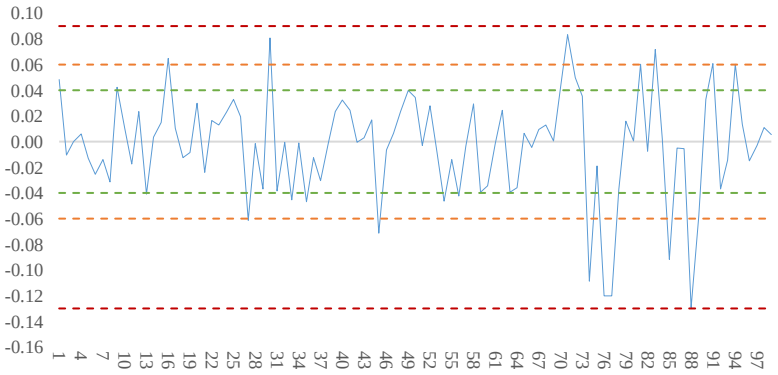

Figure 15. Prediction error trace on the first 100 Delta Aileron test data

Performance Evaluation on S&P 500 close price prediction

In the following, to experimentally compare the performance of the GODFIS algorithm, the problem of predicting the closing price of the S&P 500 index is used (Pratama, Anavatti, Angelov, et al., 2014). This dataset is considered one of the most challenging time series prediction problems due to its nonlinear behavior, irregular fluctuations, and significant temporal variations, and it is often used to rigorously evaluate the capability of learning models. The dataset was obtained from the Yahoo! Finance website[2] and includes 14,893 samples spanning from January 3, 1950 to March 12, 2009 (a period of 60 years). For the prediction task, the GODFIS algorithm is applied for online prediction modeling on both the original time series and its reversed version. As a result, the total number of data points used is 29,786. The first half of the data is used for training, while the remaining half is used for testing. The relationship between the system's input and output is governed by Eq. 0.

$$\hat{x}(t+1) = f\big(x(t), x(t-1), x(t-2), x(t-3), x(t-4)\big) \tag{30}$$

The performance comparison between the proposed algorithm and various other EFSs on this problem is presented in Table 6, based on NDEI, number of rules, and execution time. The reported results are directly taken from (Angelov et al., 2018; Ge & Zeng, 2018; Pratama, Anavatti, Angelov, et al., 2014).

Table 6. Comparison of GODFIS with other methods in the literature on the S&P 500 dataset

Model	Model Type	NDEI	#(Rule)	t_{exe}
GODFIS	T-S	0.0153	14	16.7
LEOA (Ge & Zeng, 2018)	T-S	0.123	52	-
ALMMo (Angelov et al., 2018)	T-S	0.0147	6	3.6
PSO-ALMMo (Gu et al., 2021)	T-S	0.0146	6	2783.3
DENFIS (Kasabov & Qun, 2002)	T-S	0.02	6	-
EFuNN (Kasabov, 2001)	T-S	0.154	114	-
EFS-SLAT (Ge & Zeng, 2020)	T-S	0.016	23	-
eTS (Angelov & Filev, 2004)	T-S	0.015	14	-
GENEFIS (Pratama, Anavatti, & Lughofer, 2014)	T-S	0.07	2	-
PANFIS (Pratama, Anavatti, Angelov, et al., 2014)	T-S	0.09	4	-
SEFS (Ge & Zeng, 2019)	T-S	0.018	2	-
Simpl_eTS (Angelov & Filev, 2005)	T-S	0.045	7	-

As shown in Table 6, among the online algorithms (i.e., all except PSO-ALMMo, which is designed for offline scenarios), the GODFIS algorithm ranks third in terms of the NDEI index, following the ALMMo and eTS algorithms, with only a slight performance gap. Although ALMMo performs better in terms of the number of fuzzy rules, GODFIS and eTS both produce 14 rules,

[2] Available at: https://uk.finance.yahoo.com/

indicating a good balance between accuracy and complexity in GODFIS. Among the EFS algorithms specifically designed to achieve global optimization, GODFIS significantly outperforms LEOA in both NDEI and the number of generated rules. This confirms the superiority of the optimization strategy adopted in GODFIS. Compared to the PSO-ALMMo algorithm (despite its lack of online learning capability), GODFIS demonstrates a considerable advantage in terms of execution time. This difference mainly stems from PSO-ALMMo's offline structure and its reliance on metaheuristic methods for global optimization.

Figure 16 presents a comparison between the predicted and actual values along the 45-degree line, while Figure 17 shows the predicted and actual values plotted over time. Both figures confirm the high accuracy of GODFIS and the overall reliability of its prediction performance.

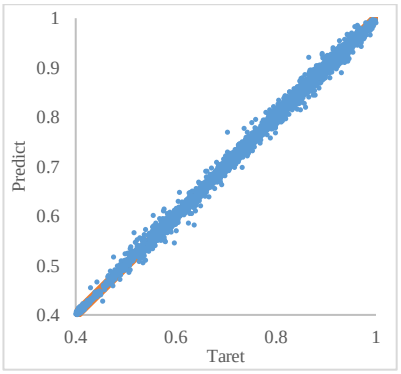

Figure 16. Comparison of predicted and actual test data values for S&P 500 on the 45-degree line

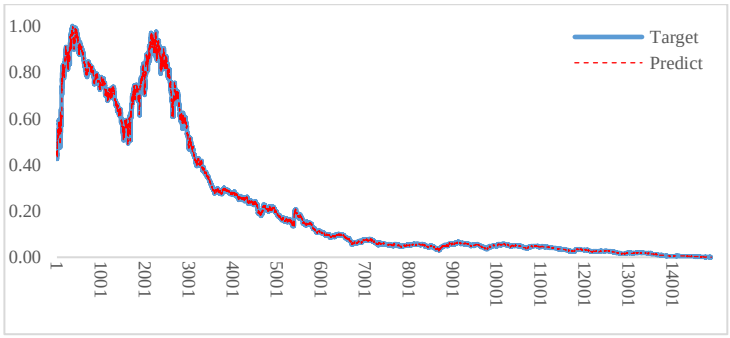

Figure 17. Comparison between predicted and actual values of S&P 500 test data

Figure 18 shows the histogram of the prediction errors on the test points, revealing an approximately normal distribution. Figure 19 illustrates the prediction error trace of the GODFIS algorithm during its execution on the S&P 500 test data. As observed, the absolute prediction error rarely exceeds 0.06 and, in most cases, remains even below 0.04. These results indicate the acceptable performance and stability of the algorithm in forecasting the data.

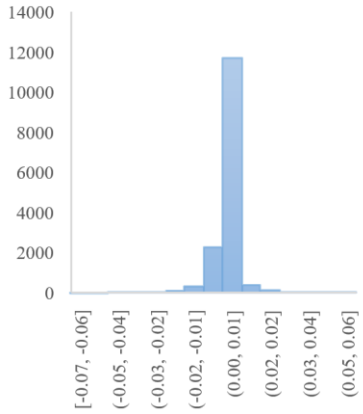

Figure 18. Histogram of prediction errors on the S&P 500 test data

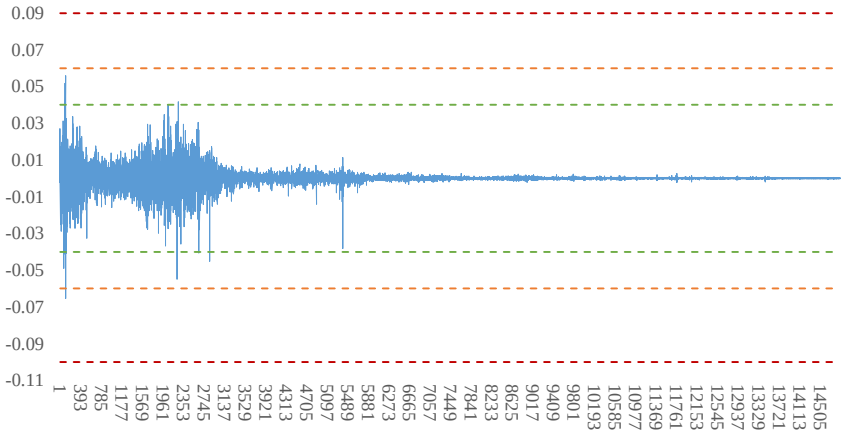

Figure 19. Prediction error trace on the S&P 500 test data

Performance Evaluation of NEP

To quantitatively and practically evaluate the performance of the NEP, several experiments were conducted on three benchmark datasets: MG (Mackey & Glass, 1977), the Delta Ailerons (Alcalá-fdez et al., 2011), and S&P 500 (Pratama, Anavatti, Angelov, et al., 2014). In all experiments, significant Gaussian noise with zero mean and variance ranging from 0.01 to 0.1 was added to the data to simulate non-ideal operating conditions.

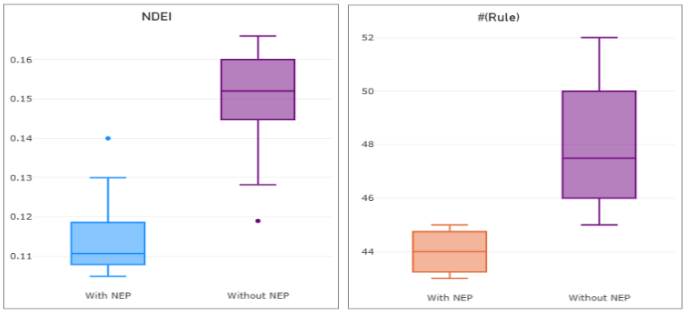

Figure 20. Comparison of NDEI on MG data with and without NEP applied

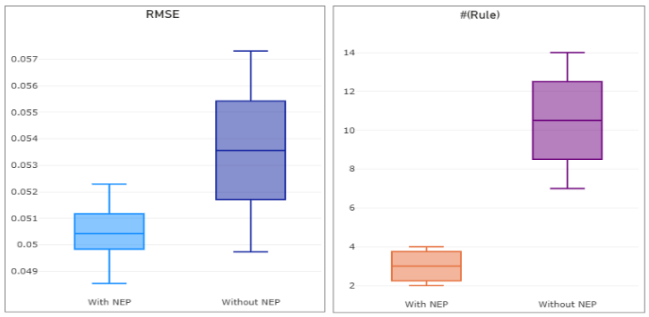

Figure 21. Comparison of RMSE on Delta Ailerons data with and without NEP applied

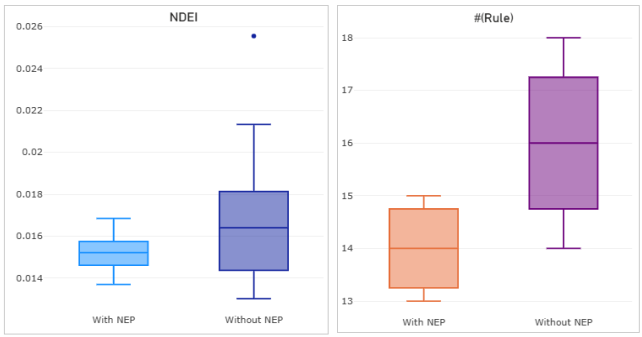

Figure 22. Comparison of RMSE on S&P 500 data with and without NEP applied

The results demonstrate that NEP improves prediction accuracy while reducing model complexity across all benchmarks. For the MG dataset, it decreased NDEI by approximately 23% and the number of rules by 10% (Figure 20). On Delta Ailerons, RMSE was reduced by about 11%, and the rule count by 67% (Figure 21). For the S&P 500 dataset, NEP lowered RMSE by approximately 9% and the number of rules by 12.5% (Figure 22).

Overall, these results show that NEP reduces the influence of noise while controlling model complexity, thereby improving prediction accuracy and stability across the evaluated benchmarks. Its effectiveness on chaotic and nonlinear datasets also supports its practical value for real-world applications.

Comparison of Local and Global Update of Consequent Parameters

As mentioned, the GODFIS algorithm is designed for global updating of consequent parameters. This global approach results in higher-quality outcomes, as it considers all relationships and interactions among the parameters in an integrated manner. To further investigate this aspect, all benchmark problems were also solved using the local consequent parameter update method. A comparison of the results obtained from both approaches (global update vs. local update) is presented in Table 7.

Table 7. Comparison of Global vs. Local consequent parameter update in the GODFIS algorithm

Dataset	Model	RMSE	NDEI	#(Rule)	t_{exe}
Mackey-Glass	GODFIS (Global)	0.0270	0.1062	42	9.3
	GODFIS (Local)	0.0843	0.3321	42	8.4
Delta Ailerons	GODFIS (Global)	0.0449	0.5437	2	11.7
	GODFIS (Local)	0.0456	0.5492	2	11.3
S&P 500	GODFIS (Global)	0.0044	0.0153	14	34.3
	GODFIS (Local)	0.0045	0.0157	14	26.3

As shown in Table 7, the number of rules generated is identical for both approaches. This is expected, as the initial rule detection phase is the same in both methods. Regarding the RMSE and NDEI indices, the GODFIS algorithm demonstrates higher prediction accuracy in the global consequent parameter update approach. This improvement stems from the accurate and optimal identification of antecedent parameters using the COG concept. These results are consistent with the findings of (Yen et al., 1998), which, although developed under offline and optimal conditions, also confirm that global update strategies generally offer better prediction accuracy than local ones, provided that the antecedent structure is correctly identified. Overall, the performance of GODFIS can be attributed to the integration of three components: the globally optimized evolving clustering algorithm (GOECA), global consequent-parameter updates, and the NEP pruning mechanism. Their combined effect improves prediction accuracy and structural efficiency, providing an advantage over the evaluated existing methods.

Conclusion

In this research, we presented GODFIS, a novel online learning algorithm for evolving fuzzy system identification. The proposed framework was designed to address an important limitation observed in much of the existing literature, namely the limited emphasis on joint optimization and global adaptation in online evolving fuzzy models. To overcome this limitation, GODFIS integrates three complementary components: GOECA, global consequent parameter updating, and the Noise Elimination Principle (NEP). First, this study introduces a new clustering method called GOECA, which operates based on recursive COG computation. This design enables near-optimal clustering of the input space while preserving online operation. Second, unlike local update strategies, the global updating of consequent parameters allows the model to consider the interdependence among rules more effectively and thus improves predictive performance. Third, the NEP mechanism helps reduce the impact of noisy observations and supports the maintenance of a reliable knowledge base during learning.

The experimental results on widely used benchmark datasets showed that GODFIS achieves high prediction accuracy, fast adaptation to new data, and competitive computational efficiency. In particular, the comparison with established EFS methods indicated that the proposed architecture provides a strong balance between accuracy and adaptability. In addition, the comparison between global and local consequent parameter updates confirmed that the global strategy yields better performance in terms of RMSE and NDEI, while producing the same number of rules. This suggests that the observed improvement is mainly due to more effective parameter optimization rather than an increase in model complexity.

Beyond its mathematical and algorithmic contributions, the proposed GODFIS framework offers significant practical value for decision-makers and industrial managers. In today's dynamic business environments, characterized by high-velocity and non-stationary data streams, traditional static forecasting models often fail to provide timely insights. By leveraging the online learning capability and structural adaptability of GODFIS, managers can implement real-time decision-making systems. Specifically, in areas such as energy management, stock market analysis, and supply chain demand forecasting, the model enables organizations to proactively allocate resources, minimize operational waste, and mitigate risks associated with sudden market shifts. Furthermore, the high interpretability of the fuzzy rule base ensures that managers can easily comprehend the underlying logic of the predictions, thereby fostering trust and transparency in AI-driven decision support systems.

Ethical considerations

The authors avoided data fabrication, falsification, and plagiarism, and any form of misconduct.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflict of interest

The authors declare no conflict of interest.

References

Alcalá-fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., & Herrera, F. (2011). KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. *Journal of Multiple-Valued Logic & Soft Computing*, *17*.

Angelov, P. (2010). Evolving Takagi-Sugeno Fuzzy Systems from Streaming Data (eTS+). In *Evolving Intelligent Systems* (pp. 21–50). https://doi.org/https://doi.org/10.1002/9780470569962.ch2

Angelov, P. (2011). Fuzzily Connected Multimodel Systems Evolving Autonomously From Data Streams. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *41*(4), 898–910. https://doi.org/10.1109/TSMCB.2010.2098866

Angelov, P., & Buswell, R. (2002). Identification of evolving fuzzy rule-based models. *IEEE Transactions on Fuzzy Systems*, *10*(5), 667–677. https://doi.org/10.1109/TFUZZ.2002.803499

Angelov, P., & Filev, D. (2005, 25–25 May 2005). Simpl_eTS: a simplified method for learning evolving Takagi-Sugeno fuzzy models. The 14th IEEE International Conference on Fuzzy Systems, 2005. FUZZ '05.,

Angelov, P., & Zhou, X. (2006, 7–9 Sept. 2006). Evolving Fuzzy Systems from Data Streams in Real-Time. 2006 International Symposium on Evolving Fuzzy Systems,

Angelov, P. P., & Filev, D. P. (2004). An approach to online identification of Takagi-Sugeno fuzzy models. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *34*(1), 484–498. https://doi.org/10.1109/TSMCB.2003.817053

Angelov, P. P., Gu, X., & Príncipe, J. C. (2018). Autonomous Learning Multimodel Systems From Data Streams. *IEEE Transactions on Fuzzy Systems*, *26*(4), 2213–2224. https://doi.org/10.1109/TFUZZ.2017.2769039

Angelov, P. P., & Zhou, X. (2008). Evolving Fuzzy-Rule-Based Classifiers From Data Streams. *IEEE Transactions on Fuzzy Systems*, *16*(6), 1462–1475. https://doi.org/10.1109/TFUZZ.2008.925904

Bao, R. J., Rong, H. J., Angelov, P. P., Chen, B., & Wong, P. K. (2018). Correntropy-Based Evolving Fuzzy Neural System. *IEEE Transactions on Fuzzy Systems*, *26*(3), 1324–1338. https://doi.org/10.1109/TFUZZ.2017.2719619

Baruah, R. D., & Angelov, P. (2014). DEC: Dynamically Evolving Clustering and Its Application to Structure Identification of Evolving Fuzzy Models. *IEEE Transactions on Cybernetics*, *44*(9), 1619–1631. https://doi.org/10.1109/TCYB.2013.2291234

de Campos Souza, P. V. (2020). Fuzzy neural networks and neuro-fuzzy networks: A review the main techniques and applications used in the literature. *Applied Soft Computing*, *92*, 106275. https://doi.org/https://doi.org/10.1016/j.asoc.2020.106275

Dovžan, D., Loga, V., & Škrjanc, I. (2012). Solving the sales prediction with fuzzy evolving models. WCCI 2012 IEEE World Congress on Computational Intelligence, une, Brisbane, Australia,

Dovžan, D., Logar, V., & Škrjanc, I. (2015). Implementation of an Evolving Fuzzy Model (eFuMo) in a Monitoring System for a Waste-Water Treatment Process. *IEEE Transactions on Fuzzy Systems*, *23*(5), 1761–1776. https://doi.org/10.1109/TFUZZ.2014.2379252

Dovžan, D., & Škrjanc, I. (2011a). Recursive clustering based on a Gustafson–Kessel algorithm. *Evolving Systems*, *2*(1), 15–24. https://doi.org/10.1007/s12530-010-9025-7

Dovžan, D., & Škrjanc, I. (2011b). Recursive fuzzy c-means clustering for recursive fuzzy identification of time-varying processes. *ISA Transactions*, *50*(2), 159–169. https://doi.org/https://doi.org/10.1016/j.isatra.2011.01.004

Faezirad, M., Pooya, A., Naji-Azimi, Z., & Amir Haeri, M. (2021). Demand Prediction in University Booking Systems to Reduce Food Waste Using Neural Networks Including Weighted Error Function. *Industrial Management Journal*, *13*(2), 193–170. https://doi.org/10.22059/imj.2021.318760.1007821

Faghihi Nezhad, M. T., & Minaei, B. (2018). Prediction of Stock Market Behavior Based on Artificial Neural Networks through Intelligent Ensemble Learning Approach. *Industrial Management Journal*, *10*(2), 315–334. https://doi.org/10.22059/imj.2018.255079.1007412

Fernandez, A., Herrera, F., Cordon, O., Jesus, M. J. d., & Marcelloni, F. (2019). Evolutionary Fuzzy Systems for Explainable Artificial Intelligence: Why, When, What for, and Where to? *IEEE Computational Intelligence Magazine*, *14*(1), 69–81. https://doi.org/10.1109/MCI.2018.2881645

Ge, D., & Zeng, X.-J. (2018). Learning evolving T–S fuzzy systems with both local and global accuracy – A local online optimization approach. *Applied Soft Computing*, *68*, 795–810. https://doi.org/https://doi.org/10.1016/j.asoc.2017.05.046

Ge, D., & Zeng, X.-J. (2020). Learning data streams online — An evolving fuzzy system approach with self-learning/adaptive thresholds. *Information Sciences*, *507*, 172–184. https://doi.org/https://doi.org/10.1016/j.ins.2019.08.036

Ge, D., & Zeng, X. J. (2019). A Self-Evolving Fuzzy System Which Learns Dynamic Threshold Parameter by Itself. *IEEE Transactions on Fuzzy Systems*, *27*(8), 1625–1637. https://doi.org/10.1109/TFUZZ.2018.2886154

Gu, X. (2021). Multilayer Ensemble Evolving Fuzzy Inference System. *IEEE Transactions on Fuzzy Systems*, *29*(8), 2425–2431. https://doi.org/10.1109/TFUZZ.2020.2988846

Gu, X., & Angelov, P. (2019). Self-boosting first-order autonomous learning neuro-fuzzy systems. *Applied Soft Computing*, *77*, 118–134. https://doi.org/https://doi.org/10.1016/j.asoc.2019.01.005

Gu, X., Angelov, P., Han, J., & Shen, Q. (2023). Multilayer Evolving Fuzzy Neural Networks. *IEEE Transactions on Fuzzy Systems*, *31*(12), 4158–4169. https://doi.org/10.1109/TFUZZ.2023.3276263

Gu, X., Han, J., Shen, Q., & Angelov, P. P. (2023). Autonomous learning for fuzzy systems: a review. *Artificial Intelligence Review*, *56*(8), 7549–7595. https://doi.org/10.1007/s10462-022-10355-6

Gu, X., Li, M., Shen, L., Tang, G., Ni, Q., Peng, T., & Shen, Q. (2023). Multiobjective Evolutionary Optimization for Prototype-Based Fuzzy Classifiers. *IEEE Transactions on Fuzzy Systems*, *31*(5), 1703–1715. https://doi.org/10.1109/TFUZZ.2022.3214241

Gu, X., & Shen, Q. (2021). A self-adaptive fuzzy learning system for streaming data prediction. *Information Sciences*, *579*, 623–647. https://doi.org/https://doi.org/10.1016/j.ins.2021.08.023

Gu, X., Shen, Q., & Angelov, P. P. (2021). Particle Swarm Optimized Autonomous Learning Fuzzy System. *IEEE Transactions on Cybernetics*, *51*(11), 5352–5363. https://doi.org/10.1109/TCYB.2020.2967462

Hagras, H. A. (2004). A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots. *IEEE Transactions on Fuzzy Systems*, *12*(4), 524–539. https://doi.org/10.1109/TFUZZ.2004.832538

Hong, Y.-S. T., & White, P. A. (2009). Hydrological modeling using a dynamic neuro-fuzzy system with on-line and local learning algorithm. *Advances in Water Resources*, *32*(1), 110–119. https://doi.org/https://doi.org/10.1016/j.advwatres.2008.10.006

Hu, L., Xu, X., Liu, J., Yan, X., & Han, M. (2025). MIIPSO-EFS: Learning system with self-optimized parameters for chaotic time series online prediction. *Knowledge-Based Systems*, *310*, 112878. https://doi.org/https://doi.org/10.1016/j.knosys.2024.112878

Hu, L., Xu, X., Ren, W., & Han, M. (2024). Hierarchical Evolving Fuzzy System: A Method for Multidimensional Chaotic Time Series Online Prediction. *IEEE Transactions on Fuzzy Systems*, *32*(6), 3329–3341. https://doi.org/10.1109/TFUZZ.2023.3348847

K. Kasabov, N. (2001). On-line learning, reasoning, rule extraction and aggregation in locally optimized evolving fuzzy neural networks. *Neurocomputing*, *41*(1), 25–45. https://doi.org/https://doi.org/10.1016/S0925-2312(00)00346-5

Kasabov, N. (2001). Evolving fuzzy neural networks for supervised/unsupervised online knowledge-based learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *31*(6), 902–918. https://doi.org/10.1109/3477.969494

Kasabov, N. K., & Qun, S. (2002). DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction. *IEEE Transactions on Fuzzy Systems*, *10*(2), 144–154. https://doi.org/10.1109/91.995117

Kazemi, A., Modarres, M., & Mehregan, M. R. (2011). Transport energy Demand Forecasting Using Markov Chain Grey Model: Case Study of Iran. *Industrial Management Journal*, *3*(2), 117–132.

Leite, D., Škrjanc, I., & Gomide, F. (2020). An overview on evolving systems and learning from stream data. *Evolving Systems*, *11*(2), 181–198. https://doi.org/10.1007/s12530-020-09334-5

Lemos, A., Caminhas, W., & Gomide, F. (2011a). Multivariable Gaussian Evolving Fuzzy Modeling System. *Trans. Fuz Sys.*, *19*(1), 91–104. https://doi.org/10.1109/tfuzz.2010.2087381

Lemos, A., Caminhas, W., & Gomide, F. (2011b). Multivariable Gaussian Evolving Fuzzy Modeling System. *IEEE Transactions on Fuzzy Systems*, *19*(1), 91–104. https://doi.org/10.1109/TFUZZ.2010.2087381

Lima, E., Hell, M., Ballini, R., & Gomide, F. (2010). Evolving Fuzzy Modeling Using Participatory Learning. In *Evolving Intelligent Systems* (pp. 67–86). https://doi.org/https://doi.org/10.1002/9780470569962.ch4

Lughofer, E. (2011a). EFS Approaches for Regression and Classification. In *Evolving Fuzzy Systems – Methodologies, Advanced Concepts and Applications* (pp. 93–164). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-18087-3_3

Lughofer, E. (2011b). *Evolving fuzzy systems-methodologies, advanced concepts and applications* (Vol. 53). Springer.

Lughofer, E., Cernuda, C., Kindermann, S., & Pratama, M. (2015). Generalized smart evolving fuzzy systems. *Evolving Systems*, *6*(4), 269–292. https://doi.org/10.1007/s12530-015-9132-6

Lughofer, E. D. (2008). FLEXFIS: A Robust Incremental Learning Approach for Evolving Takagi–Sugeno Fuzzy Models. *IEEE Transactions on Fuzzy Systems*, *16*(6), 1393–1410. https://doi.org/10.1109/TFUZZ.2008.925908

Maciel, L., Ballini, R., & Gomide, F. (2017). Evolving possibilistic fuzzy modelling. *Journal of Statistical Computation and Simulation*, *87*(7), 1446–1466. https://doi.org/10.1080/00949655.2016.1270281

Maciel, L., Gomide, F., & Ballini, R. (2014). Enhanced evolving participatory learning fuzzy modeling: an application for asset returns volatility forecasting. *Evolving Systems*, *5*(2), 75–88. https://doi.org/10.1007/s12530-013-9099-0

Maciel, L., Gomide, F., & Ballini, R. (2014, 9–12 Dec. 2014). Recursive possibilistic fuzzy modeling. 2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS),

Mackey, M. C., & Glass, L. (1977). Oscillation and Chaos in Physiological Control Systems. *Science*, *197*(4300), 287–289.

Nguyen, N. N., Zhou, W. J., & Quek, C. (2015). GSETSK: a generic self-evolving TSK fuzzy neural network with a novel Hebbian-based rule reduction approach. *Applied Soft Computing*, *35*, 29–42. https://doi.org/https://doi.org/10.1016/j.asoc.2015.06.008

Ožbot, M., Lughofer, E., & Škrjanc, I. (2023). Evolving Neuro-Fuzzy Systems-Based Design of Experiments in Process Identification. *IEEE Transactions on Fuzzy Systems*, *31*(6), 1995–2005. https://doi.org/10.1109/TFUZZ.2022.3216992

Pratama, M., Anavatti, S. G., Angelov, P. P., & Lughofer, E. (2014). PANFIS: A Novel Incremental Learning Machine. *IEEE Transactions on Neural Networks and Learning Systems*, *25*(1), 55–68. https://doi.org/10.1109/TNNLS.2013.2271933

Pratama, M., Anavatti, S. G., & Lughofer, E. (2014). GENEFIS: Toward an Effective Localist Network. *IEEE Transactions on Fuzzy Systems*, *22*(3), 547–562. https://doi.org/10.1109/TFUZZ.2013.2264938

Pratama, M., Lu, J., Lughofer, E., Zhang, G., & Er, M. J. (2017). An Incremental Learning of Concept Drifts Using Evolving Type-2 Recurrent Fuzzy Neural Networks. *IEEE Transactions on Fuzzy Systems*, *25*(5), 1175–1192. https://doi.org/10.1109/TFUZZ.2016.2599855

Ravi, V., Srinivas, E. R., & Kasabov, N. K. (2007, 13–15 Dec. 2007). On-Line Evolving Fuzzy Clustering. International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007),

Rodrigues, S. E., & de Oliveira Serra, G. L. (2022). An approach for evolving neuro-fuzzy forecasting of time series based on parallel recursive singular spectrum analysis. *Fuzzy Sets and Systems*, *443*, 1–29. https://doi.org/https://doi.org/10.1016/j.fss.2021.09.009

Rong, H.-J., Sundararajan, N., Huang, G.-B., & Saratchandran, P. (2006). Sequential Adaptive Fuzzy Inference System (SAFIS) for nonlinear system identification and prediction. *Fuzzy Sets and Systems*, *157*(9), 1260–1275. https://doi.org/https://doi.org/10.1016/j.fss.2005.12.011

Rong, H.-J., Sundararajan, N., Huang, G.-B., & Zhao, G.-S. (2011). Extended sequential adaptive fuzzy inference system for classification problems. *Evolving Systems*, *2*(2), 71–82. https://doi.org/10.1007/s12530-010-9023-9

Rong, H. J., Angelov, P. P., Gu, X., & Bai, J. M. (2018). Stability of Evolving Fuzzy Systems Based on Data Clouds. *IEEE Transactions on Fuzzy Systems*, *26*(5), 2774–2784. https://doi.org/10.1109/TFUZZ.2018.2793258

Rong, H. J., Yang, Z. X., & Wong, P. K. (2020). Robust and Noise-Insensitive Recursive Maximum Correntropy-Based Evolving Fuzzy System. *IEEE Transactions on Fuzzy Systems*, *28*(9), 2277–2284. https://doi.org/10.1109/TFUZZ.2019.2931871

Rubio, J. d. J. (2009). SOFMLS: Online Self-Organizing Fuzzy Modified Least-Squares Network. *IEEE Transactions on Fuzzy Systems*, *17*(6), 1296–1309. https://doi.org/10.1109/TFUZZ.2009.2029569

Samanta, S., Pratama, M., & Sundaram, S. (2019). A novel Spatio-Temporal Fuzzy Inference System (SPATFIS) and its stability analysis. *Information Sciences*, *505*, 84–99. https://doi.org/https://doi.org/10.1016/j.ins.2019.07.056

Santoso, F., Garratt, M. A., & Anavatti, S. G. (2020). T2-ETS-IE: A Type-2 Evolutionary Takagi–Sugeno Fuzzy Inference System With the Information Entropy-Based Pruning Technique. *IEEE Transactions on Fuzzy Systems*, *28*(10), 2665–2672. https://doi.org/10.1109/TFUZZ.2019.2943813

Škrjanc, I., Iglesias, J. A., Sanchis, A., Leite, D., Lughofer, E., & Gomide, F. (2019). Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: A Survey. *Information Sciences*, *490*, 344–368. https://doi.org/https://doi.org/10.1016/j.ins.2019.03.060

Soltic, S., Pang, S., Kasabov, N., Worner, S., & Peackok, L. (2004). Dynamic Neuro-fuzzy Inference and Statistical Models for Risk Analysis of Pest Insect Establishment. In N. R. Pal, N. Kasabov, R. K. Mudi, S. Pal, & S. K. Parui, *Neural Information Processing* Berlin, Heidelberg.

Subramanian, K., & Suresh, S. (2012). A meta-cognitive sequential learning algorithm for neuro-fuzzy inference system. *Applied Soft Computing*, *12*(11), 3603–3614. https://doi.org/https://doi.org/10.1016/j.asoc.2012.06.012

Talei, A., Chua, L. H. C., Quek, C., & Jansson, P.-E. (2013). Runoff forecasting using a Takagi–Sugeno neuro-fuzzy model with online learning. *Journal of Hydrology*, *488*, 17–32. https://doi.org/https://doi.org/10.1016/j.jhydrol.2013.02.022

Wang, Z., & Fei, J. (2022). Fractional-Order Terminal Sliding-Mode Control Using Self-Evolving Recurrent Chebyshev Fuzzy Neural Network for MEMS Gyroscope. *IEEE Transactions on Fuzzy Systems*, *30*(7), 2747–2758. https://doi.org/10.1109/TFUZZ.2021.3094717

Yen, J., Liang, W., & Gillespie, C. W. (1998). Improving the interpretability of TSK fuzzy models by combining global learning and local learning. *IEEE Transactions on Fuzzy Systems*, *6*(4), 530–537. https://doi.org/10.1109/91.728447

Yu, H., Lu, J., & Zhang, G. (2022). Topology Learning-Based Fuzzy Random Neural Networks for Streaming Data Regression. *IEEE Transactions on Fuzzy Systems*, *30*(2), 412–425. https://doi.org/10.1109/TFUZZ.2020.3039681