SoftwareX 26 (2024) 101730

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

Python library for interval-valued fuzzy inference

Krzysztof Dyczkowski [a,*], Piotr Grochowalski [b], Dawid Kosior [b], Dorota Gil [b], Wojciech Kozioł [b], Barbara Pękala [b,c], Uzay Kaymak [d], Caro Fuchs [d], Marco S. Nobile [e]

[a] Adam Mickiewicz University, Poznań, Poland
[b] University of Rzeszów, Rzeszów, Poland
[c] University of Information Technology and Management, Rzeszów, Poland
[d] Eindhoven University of Technology, Eindhoven, Netherlands
[e] Ca' Foscari University of Venice, Italy

ARTICLE INFO

Keywords:
Uncertainty modeling
Interval-valued fuzzy sets
Interval-valued fuzzy inference system (IFIS)
Python
Computational intelligence
Advanced reasoning algorithms

ABSTRACT

Uncertainty is an essential feature of many concepts, data sets, and operations encountered in various applications related to computer science and engineering, and medical diagnostics. This paper introduces an extension to the Simpful library, enhancing it with capabilities for interval-valued fuzzy sets (IVFSs) and generalized fuzzy inference. Addressing the need for comprehensive uncertainty modeling in various applications, this extension offers intuitive definitions and operations for interval-valued fuzzy sets, facilitating advanced reasoning in uncertain environments.

Current code version	v1.0.1
Permanent link to code/repository used for this code version	https://github.com//ElsevierSoftwareX/SOFTX-D-24-00052
Permanent link to Reproducible Capsule	https://codeocean.com/capsule/2679012/tree/v2
Legal Code License	GNU GENERAL PUBLIC LICENSE, Version 3
Code versioning system used	git
Software code languages, tools, and services used	Python >= 3.8 simpful~=2.11.1 numpy~=1.26.2 scipy~=1.12.0 numpydoc~=1.6.0
Compilation requirements, operating environments & dependencies	Simpful library https://github.com/aresio/simpful
If available Link to developer documentation/manual	https://pgrochowalski.github.io/ifis/
Support email for questions	pgrochowalski@ur.edu.pl

1. Motivation and significance

Solving real-world problems often relies on the capability of the language of representation, or logic, to handle imperfect, inaccurate, or ambiguous information. Fuzzy logic and fuzzy sets [1] offer formal inference tools for dealing with imprecise information. By incorporating interval-valued fuzzy sets [2] into the rules model, we can obtain formal tools to enhance reasoning by modeling uncertain information, also known as generalized fuzzy inference systems.

Our work is motivated by the need to apply interval-valued fuzzy inference in various applications, particularly for classifying uncertain data – a task of significant importance. For example, in cases where uncertainty stems from missing data, developing new methods becomes essential to obtain more effective results. In our case, we address uncertain data represented by Interval-Valued Fuzzy Sets (IVFSs), where intervals represent uncertain data from an epistemic perspective.

Our library can be easily integrated into Python based projects via pip installations (Python Package Index (PyPI)) and is publicly

* Corresponding author.
E-mail address: chris@amu.edu.pl (Krzysztof Dyczkowski).

https://doi.org/10.1016/j.softx.2024.101730
Received 27 January 2024; Received in revised form 22 March 2024; Accepted 29 March 2024
Available online 8 April 2024

available under an open source licence on github (see code metadata table).

Hence, we propose an extension of the Simpful library [3], called the Interval-Valued Fuzzy Inference System (IFIS). Simpful is a universal and user-friendly Python library designed for defining, analyzing, and interpreting fuzzy inference systems. Our extended and modified library facilitates programming applications in an uncertain environment. In many decision-making situations encountered during real problem-solving, interval inference processes and systems are necessary. Consequently, the implementation of a comprehensive library supporting these systems becomes imperative, which is the aim of our work.

IFIS overcomes the challenges and limitations associated with modeling uncertainty in fuzzy reasoning. An essential concept offered by the library is the ability to use inputs in the form of interval values, which represent measurements within certain limits, with known measurement error, or subject to fluctuations and uncertainty. Moreover, our library distinguishes itself from others available in the market by offering a wide range of interval operations, including interval aggregation operators or orders.

The area of application of the proposed software is multi-criteria decision-making, in particular in supporting a healthy society [4], medicine [5], economics predictions [6], or sport analytics [7].

1.1. Related work

This section provides an overview of existing tools and libraries related to interval-valued fuzzy logic and inference. Detailed information about software tools dealing with fuzzy logic and reasoning can be found in Simpful and in other sources [8]. However, in this section, we will discuss some key tools and libraries connected specifically to interval-valued fuzzy logic and inference.

Various tools have been developed for interval-valued fuzzy inference, catering to different tasks and programming languages. We provide a brief overview and description of some of these toolkits documented in the literature.

Fuzzycreator [9] is an open-source Python-based toolkit that allows the creation and analysis of conventional and non-conventional type-1, interval type-2, and general type-2 fuzzy sets from data. It includes tools and measures for comparing and evaluating the similarity and distance between fuzzy sets.

SyFSeL [10] (Synthetic Fuzzy Set Library) is an open-source Python-based library that automatically generates synthetic fuzzy sets with specified characteristics and types. Users can adjust parameters to emulate real data, and the generated fuzzy sets can be exported for use in other fuzzy system software.

Perceptual Computing (Per-C) [11] is an open-source Python library that utilizes type-2 fuzzy sets for modeling words in the Computing With Words (CWW) paradigm. The CWW methodology aims to design intelligent systems that make decisions by manipulating linguistic information, similar to human reasoning. Type-2 fuzzy sets enable better representation of uncertainty in linguistic semantics for various problems.

Fuzzy Logic Toolbox for Matlab supports two types of inference systems, Mamdani and Sugeno systems. Mamdani fuzzy inference was first introduced as a method for creating a control system by synthesizing a set of linguistic control rules obtained from experienced human operators. The defuzzification process for a Sugeno system is more computationally efficient than that of a Mamdani system since it uses a weighted average or weighted sum of a few data points rather than a computed centroid of a two-dimensional area [12].

jFuzzyLogic [13] is a Java library that provides a fully functional implementation of a fuzzy inference system according to an associated standard. It offers a programming interface and an Eclipse plugin, making it convenient for writing and testing code for fuzzy control applications.

While several tools exist for implementing fuzzy inference systems in Python, we have not found any open-source tools specifically developed for generalized fuzzy inference. This lack of a universal open system for interval-valued fuzzy inference in Python motivated us to create our complementary library. However, in the context of interval-valued inference systems in Python, some results can be found in [14], where **PyIT2FLS** is described. PyIT2FLS provides tools for fast and easy modeling of fuzzy systems. Nevertheless, it lacks advanced techniques for aggregating and comparing interval values, as well as the capability of handling interval data, which is crucial for uncertainty modeling.

In the landscape of Python-based fuzzy inference systems, there are many options, but none of them specifically address the complex needs of generalized fuzzy inference in an open-source manner. This obvious gap led us to develop our own library, which uniquely focuses on interval-valued fuzzy inference.

2. Key functionalities

Its primary function is to enhance Simpful with the capabilities of handling interval-valued fuzzy sets (IVFSs) and generalized fuzzy inference, thus providing a more comprehensive framework for uncertainty modeling.

Key Features of IFIS:

- Enhanced Fuzzy Set Capabilities: IFIS introduces interval-valued fuzzy sets, allowing for intuitive definitions and operations, vital for advanced reasoning in uncertain environments.
- Flexible Membership Function Definitions: The package enables users to define complex membership functions, including both polygonal (vertex-based) and various functional types such as sigmoidal and Gaussian. This flexibility enables users to model a wide range of fuzzy sets according to their specific requirements.
- Complex Rule Construction: IFIS allows for the construction of complex fuzzy rules using logical operators like AND, OR, and NOT. These operators are adapted for interval-valued fuzzy sets, incorporating operations like minimum and maximum for interval-valued aggregations.
- Interval-Valued Fuzzy Inference: Given values for antecedents as input, IFIS can perform interval-valued fuzzy inference (includes Mamdani and Takagi–Sugeno inference methods).
- Broad capabilities for comparing interval values (orders) and corresponding aggregating operations, which are essential for the inference system.
- Two methods for entering data into the inference system: real values and/or interval data. This flexibility provides better uncertainty modeling at the beginning of the classification process.

3. Software description

We implemented the Interval-Valued Fuzzy Inference System (IFIS) in Python 3, enhancing the Simpful library [3] with dependencies on Numpy and Scipy. This system is adept at performing interval-valued fuzzy inference by processing input values for antecedents and then generating corresponding output values, as depicted in Fig. 1.

Fig. 2 illustrates the class structure within the IFIS module, showcasing the enhancements and new features integrated into the existing Simpful library framework.

The class structure in IFIS builds upon the existing structure of Simpful, incorporating additional classes and functionalities to support interval-valued fuzzy inference. IFIS introduces new elements such as interval-valued membership functions, interval-valued fuzzy rules, and the capability to handle arbitrary interval-valued aggregations. These additions enable the implementation of interval-valued fuzzy inference using Mamdani and Takagi–Sugeno methods.

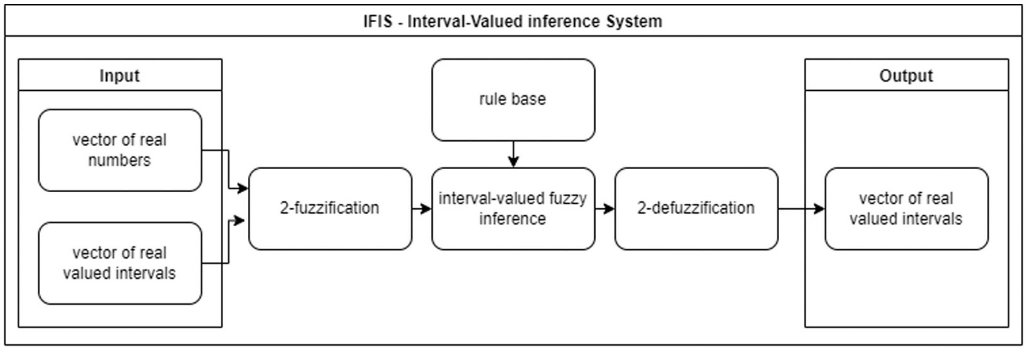


Fig. 1. IFIS (Interval-Valued Fuzzy Inference System).

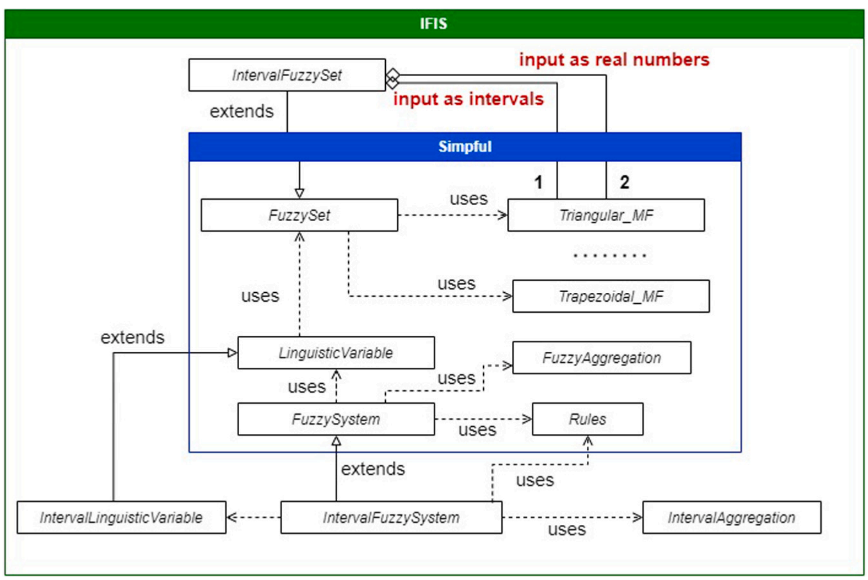


Fig. 2. Overview of IFIS's class structure, showcasing enhancements and new additions to the Simpful framework.

4. Illustrative examples

The tipping problem involves calculating a fair tip as a percentage of the overall bill, considering the quality of service provided by a restaurant.

Listing 1 provides an example of IFIS code defining an interval inference system to calculate the tip amount based on two input variables: food quality and service quality. The code in Listing 1 demonstrates a range inference system designed to determine the tip percentage for the waiter, given an evaluation of the service in a restaurant. The system considers two inputs, namely food quality and service quality. This example showcases how to define an inference system using interval-valued fuzzy sets.

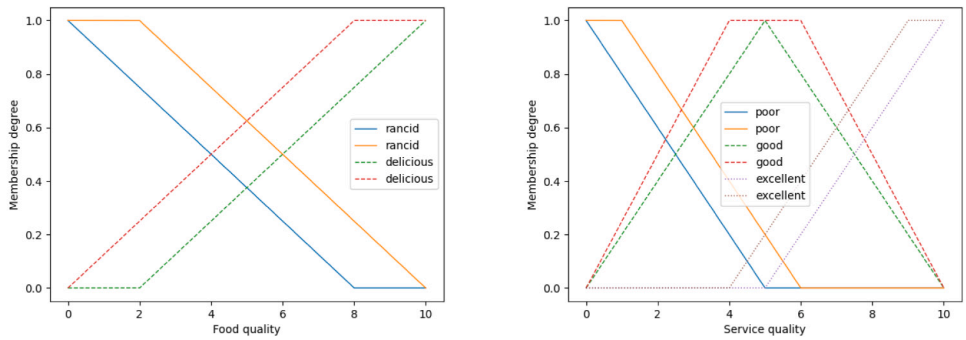
Listing 1: An example implementation of an inference system based on the Takagi–Sugeno method

```
from simpful import *
from interval_simpful.interval_fuzzy_sets import *
from interval_simpful.interval_fuzzy_system import *
from interval_simpful.interval_linguistic_variable import *

# A simple interval fuzzy inference system for the tipping
    problem
# Create interval fuzzy system object
iFS = IntervalFuzzySystem()

# Define interval fuzzy sets and interval linguistic variables
S_1 = IntervalFuzzySet(function_start=Triangular_MF(a=0, b=0, c
    =5), function_end=Trapezoidal_MF(a=1, b=1, c=1, d=6), term=
    'poor')
S_2 = IntervalFuzzySet(function_start=Triangular_MF(a=0, b=5, c
    =10), function_end=Trapezoidal_MF(a=0, b=4, c=6, d=10),
    term='good')
S_3 = IntervalFuzzySet(function_start=Triangular_MF(a=5, b=10, c
    =10), function_end=Trapezoidal_MF(a=4, b=9, c=10, d=10),
    term='excellent')
iFS.add_linguistic_variable("Service", IntervalLinguisticVariable
    ([S_1, S_2, S_3], concept="Service quality",
    universe_of_discourse=[0, 10]))

F_1 = IntervalFuzzySet(function_start=Triangular_MF(a=0, b=0, c
    =8), function_end=Trapezoidal_MF(a=1, b=1, c=2, d=10), term
    ='rancid')
```



(a) Food quality (b) Service quality

Fig. 3. Graphical representation of membership functions for interval-valued fuzzy sets for the linguistic variables (a) food quality and (b) service quality.

```
17 F_2 = IntervalFuzzySet(function_start=Triangular_MF(a=2, b=10, c=10), function_end=Trapezoidal_MF(a=0, b=8, c=10, d=10), term='delicious')
18 iFS.add_linguistic_variable("Food", IntervalLinguisticVariable([F_1, F_2], concept="Food quality", universe_of_discourse=[0, 10]))
19
20 # Define output crisp values
21 iFS.set_crisp_output_value("small", 5)
22 iFS.set_crisp_output_value("average", 15)
23
24 # Define function for generous tip (food score + service score + 5iFS.set_output_function("generous", "Food+Service+5")
25
26 # Define fuzzy rules
27 R1 = "IF (Service IS poor) OR (Food IS rancid) THEN (Tip IS small)"
28 R2 = "IF (Service IS good) THEN (Tip IS average)"
29 R3 = "IF (Service IS excellent) OR (Food IS delicious) THEN (Tip IS generous)"
30 iFS.add_rules([R1, R2, R3])
31
32 # Set antecedents values
33 iFS.set_variable("Service", 4)
34 iFS.set_variable("Food", 8)
35
36 # Perform Sugeno interval inference and print output
37 print(iFS.Sugeno_interval_inference(["Tip"]))
```

The code begins by creating a root object on line 8. This object lays the groundwork for further enhancements with key components for interval-valued fuzzy inference.

Next, between lines 11 and 18, IVFS's are developed and linked with their corresponding linguistic variables. These sets are defined using two membership functions—one for the lower and one for the upper interval value.

The example code highlights two input variables: service and food quality. Service quality is divided into three fuzzy sets: "poor", "good", and "excellent", while food quality is categorized into "rancid" and "delicious". Fig. 3 illustrates these interval-valued fuzzy sets graphically, utilizing both triangular and trapezoidal functions for each linguistic label of the input variables.

Lines 21–25 define the output for the inference system, with variations depending on the chosen inference method. There are three possible outputs: two precise values representing tip percentages of 5% and 15% of the total order value, and one value derived from the relationship stated in line 25. Lines 27–30 set the fuzzy inference rules, and lines 33–34 determine the input variables' values. The inference, initiated on line 38 using Takagi–Sugeno method, produces a tip value interval based on given service and food quality ratings.

In the example, the service quality rating is 4 and the food quality rating is 8 (within the ranges $[0, 10]$), the resulting tip value is represented by the interval [14.1667%, 14.78143%].

An another illustrative example of practical application for our library is its integration into a fall detection algorithm specifically designed for monitoring elderly individuals. This algorithm is openly accessible on GitHub (https://github.com/PGrochowalski/ifis/tree/main/examples/posture_detection), where both the code and data links are provided. The model underwent testing at an elderly care facility in Poland, highlighting the critical role of elderly care in contemporary society. In our study [4], we focus on analyzing a single depth map to identify when a person is lying down. This is in contrast to the UR Fall Detection Dataset, which also offers analysis of character movement. We defined parameters for character posture by clustering 600 images that showcase various forms in different scenarios: everyday activities, falls, and lying down. These images were collected using an inertial motion sensor and Kinect cameras. To respect privacy, we opted to analyze only depth maps instead of video footage, leading to the development of a more precise posture detection model that reduced the error rate by 2.2 percentage points.

5. Impact

New research questions can span across multiple disciplines, leveraging the unique capabilities of the software to handle uncertainty and fuzzy logic. Here are some potential research questions:

- How can interval-valued fuzzy logic be more effectively applied in real-world scenarios, such as medical diagnosis, financial risk assessment, or climate modeling, where uncertainty is a significant factor?
- What new optimization techniques can be developed for interval-valued fuzzy systems to enhance their performance in complex decision-making scenarios?
- How can interval-valued fuzzy logic be used to better model human decision-making processes, particularly in situations where decisions are made under uncertainty or with imprecise information?
- What are the potential applications of interval-valued fuzzy logic in less explored fields, such as social sciences, humanities, or arts, where qualitative data and subjective judgments are prevalent?
- How can interval-valued fuzzy logic enhance the interpretation of sensory data in robotics and IoT devices, especially in environments with high levels of noise or uncertainty?

The exploration of these questions could lead to significant advancements in both theoretical and applied aspects of uncertainty management. Its unique features and capabilities allow for a more nuanced handling of uncertainty and complexity in various research domains. Here's an overview of how and to what extent it improves existing research pursuits:

- In areas like medical diagnostics, and automated control systems, decision-making often relies on interpreting uncertain or imprecise data. The software improves these processes by providing a more sophisticated framework for dealing with such data, leading to potentially more reliable and informed decisions.
- Increase Computational Efficiency by offering a Python-based solution, IFIS makes it easier to integrate advanced fuzzy logic into existing computational frameworks. This integration can lead to more efficient data processing and analysis, particularly beneficial in large-scale simulations or data-intensive research.
- The user-friendly nature of the software lowers the barrier to entry for researchers from various disciplines who might not have deep expertise in fuzzy logic. This accessibility can lead to wider adoption of interval-valued fuzzy techniques in new research areas.
- IFIS provides a practical platform to test and validate theoretical concepts in fuzzy logic and uncertainty quantification, which could lead to refinements and improvements in these areas.
- In areas where predictive modeling is key, the software could improve the handling of uncertainty, leading to more accurate and reliable predictions.
- In an academic setting, educators and students can use the software as a teaching and learning tool for fuzzy logic and uncertainty management, making these concepts more accessible and understandable.

In summary, IFIS enhances the pursuit of existing research questions by providing a more nuanced and efficient approach to handling uncertainty. Its impact is likely to be broad, affecting a wide range of disciplines and potentially leading to significant advancements in both theory and application.

6. Conclusions

The IFIS library significantly advances the modeling and management of uncertainty in data analysis and decision-making. By extending the capabilities of traditional fuzzy inference systems to include interval inference, IFIS offers a more nuanced and robust approach to uncertainty representation. Its integration with the Python programming environment enhances its accessibility and ease of use for a wide range of users. Looking ahead, the focus will be on expanding the library's functionalities to encompass a broader spectrum of uncertainty representation methods and to tailor it for diverse datasets. This will ensure that IFIS remains at the forefront of uncertainty modeling, catering to the evolving needs of various applications. We focus on interval-valued fuzzy rule systems such as Mamdani and Takagi–Sugeno in the article, but it is worth noting that our software will be developed in the future for other rule systems, e.g. neural-fuzzy rule systems.

CRediT authorship contribution statement

Krzysztof Dyczkowski: Validation, Supervision, Software, Resources, Methodology, Conceptualization, Writing – original draft, Writing – review & editing. **Piotr Grochowalski:** Writing – review & editing, Writing – original draft, Validation,Conceptualization, Software. **Dawid Kosior:** Software, Validation, Writing – original draft, Writing – review & editing. **Wojciech Kozioł:** Software, Validation, Writing – original draft, Writing – review & editing. **Barbara Pękala:** Conceptualization, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. **Uzay Kaymak:** Validation, Writing – original draft, Writing – review & editing. **Caro Fuchs:** Validation, Writing – review & editing. **Marco S. Nobile:** Software, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] Zadeh LA. Fuzzy sets. Inf Control 1965;8(3):338–53.
[2] Zadeh LA. The concept of a linguistic variable and its application to approximate reasoning–i. Inform Sci 1975;8(3):199–249.
[3] Spolaor S, Fuchs C, Cazzaniga P, Kaymak U, Besozzi D, Nobile M. Simpful: A user-friendly Python library for fuzzy logic. Int J Comput Intell Syst 2020;13:1687–98.
[4] Mroczek T, Gil D, Pękala B. Fuzzy and rough approach to the problem of missing data in fall detection system. Fuzzy Sets Syst 2024;480:108868.
[5] Sanz JA, Galar M, Jurio A, Brugos A, Pagola M, Bustince H. Medical diagnosis of cardiovascular diseases using an interval-valued fuzzy rule-based classification system. Appl Soft Comput 2014;20:103–11.
[6] Sanz JA, Bernardo D, Herrera F, Bustince H, Hagras H. A compact evolutionary interval–Valued fuzzy rule–based classification system for the modeling and prediction of real-world financial applications with imbalanced data. IEEE Trans Fuzzy Syst 2015;23(4):973–90.
[7] Piłka T, Grzelak B, Sadurska A, Górecki T, Dyczkowski K. Predicting injuries in football based on data collected from GPS-based wearable sensors. Sensors 2023;23:1227. http://dx.doi.org/10.3390/s23031227.
[8] Alcalá-Fdez J, Alonso JM. A survey of fuzzy systems software: Taxonomy, current research trends, and prospects. IEEE Trans Fuzzy Syst 2016;24(1):40–56.
[9] McCulloch J. Fuzzycreator: A Python-based toolkit for automatically generating and analysing data-driven fuzzy sets. In: 2017 IEEE international conference on fuzzy systems. 2017, p. 1–6.
[10] McCulloch J. Syfsel: Generating synthetic fuzzy sets made simple. In: 2018 IEEE international conference on fuzzy systems. 2018, p. 1–6.
[11] Sharma D, Gupta PK, Andreu-Perez J, Mendel JM, López LM. A Python software library for computing with words and perceptions. In: 2021 IEEE international conference on fuzzy systems. 2021, p. 1–6.
[12] Sugeno M. Industrial applications of fuzzy control. USA: Elsevier, Science Inc; 1985.
[13] Cingolani P, Alcala-Fdez J. jfuzzylogic: A Java library to design fuzzy logic controllers according to the standard for fuzzy control programming. Int J Comput Intell Syst 2013;6:61–75.
[14] Haghrah AA, Ghaemi S. PyIT2FLS: A new Python toolkit for interval type 2 fuzzy logic systems. 2019, arXiv:1909.10051.