UNIVERSIDADE FEDERAL DE JUIZ DE FORA

FACULDADE DE ENGENHARIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Arthur Caio Vargas e Pinto

Data-driven Partitioning Approaches for Type-2 Fuzzy Set Based Time Series

Juiz de Fora

2025

Arthur Caio Vargas e Pinto

Data-driven Partitioning Approaches for Type-2 Fuzzy Set Based Time Series

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Doutor em Engenharia Elétrica. Área de concentração: Sistemas Eletrônicos.

Orientador: Prof. Dr. Eduardo Pestana de Aguiar

Coorientadores: Prof. Dr. Frederico Gadelha Guimarães e Prof. Dr. Christian Wagner

Juiz de Fora

2025

Ficha catalográfica elaborada através do programa de geração
automática da Biblioteca Universitária da UFJF,
com os dados fornecidos pelo(a) autor(a)

Vargas e Pinto, Arthur Caio .
Data-driven Partitioning Approaches for Type-2 Fuzzy Set Based Time Series / Arthur Caio Vargas e Pinto. -- 2025.
99 p. : il.

Orientador: Eduardo Pestana de Aguiar
Coorientadores: Frederico Gadelha Guimarães, Christian Wagner
Tese (doutorado) - Universidade Federal de Juiz de Fora, Faculdade de Engenharia. Programa de Pós-Graduação em Engenharia Elétrica, 2025.

1. Previsão de Séries Temporais. 2. Sistemas Fuzzy do Tipo-2. 3. Particionamento baseado em dados. I. Pestana de Aguiar, Eduardo, orient. II. Gadelha Guimarães, Frederico, coorient. III. Wagner, Christian, coorient. IV. Título.

FEDERAL UNIVERSITY OF JUIZ DE FORA
RESEARCH AND GRADUATE PROGRAMS OFFICE

Arthur Caio Vargas e Pinto

Data-driven Partitioning Approaches for Type-2 Fuzzy Set Based Time Series

Thesis submitted to the Graduate Program in Electrical Engineering of the Federal University of Juiz de Fora as a partial requirement for obtaining a Doctor's degree in Electrical Engineering.
Concentration area: Electronic Systems

Approved on 26 of June of 2025.

EXAMINING BOARD

Prof. Dr. Eduardo Pestana de Aguiar – Academic Advisor
Federal University of Juiz de Fora

Prof. Dr. Frederico Gadelha Guimarães – Academic Co-Advisor
Federal University of Minas Gerais

Prof. Dr. Christian Wagner – Academic Co-Advisor
University of Nottingham

Prof. Dr. Alexandre Gonçalves Evsukoff
Federal University of Rio de Janeiro

Prof. Dr. Guilherme de Alencar Barreto
Federal University of Ceara

Prof. Dr. Moisés Vidal Ribeiro
Federal University of Juiz de Fora

Prof. Dr. André Luis Marques Marcato
Federal University of Juiz de Fora

Juiz de Fora, 06/26/2025.

Documento assinado eletronicamente por **Moises Vidal Ribeiro**, **Professor(a)**, em 26/06/2025, às 11:35, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Eduardo Pestana de Aguiar**, **Professor(a)**, em 26/06/2025, às 11:37, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Andre Luis Marques Marcato**, **Professor(a)**, em 26/06/2025, às 11:38, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **GUILHERME DE ALENCAR BARRETO**, **Usuário Externo**, em 27/06/2025, às 11:32, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **ALEXANDRE GONCALVES EVSUKOFF**, **Usuário Externo**, em 28/06/2025, às 10:46, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Frederico Gadelha Guimarães**, **Usuário Externo**, em 01/07/2025, às 18:41, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

To my family. The main reason of everything!
Dedico este trabalho à minha família. Razão principal de tudo!

ACKLOWLEDGMENT

I would like to thank my parents, Lecia and Ademar, for their unwavering love, sacrifices and teaching me the importance of education. Their belief in me, endless support, and the values they instilled have been the foundation of my achievements. This accomplishment is as much theirs as it is mine. You are my role models and will always have my respect and love!

To my brothers, Thomás and Matheus, my best friends in the world and the ones I can always count on. No matter the challenges, knowing I could always count on you both made all the difference.

To my wife Lourena and my son Miguel, your love and presence have been my greatest source of strength and motivation throughout this journey. No words can truly express the depth of my gratitude and love for you both. Lourena, your support, patience and encouragement kept me going through every challenge. Your love and sacrifices made this achievement possible, and I am forever grateful. Miguel, your presence filled my days with joy and purpose. Your laughter, curiosity, and pure joy have brightened even the hardest days, reminding me why I push forward. This journey would not have been possible without you both by my side.

To my family in law, for the continuing incentive and support.

My deepest gratitude to my supervisor, Prof. Dr. Eduardo Pestana de Aguiar, for his unwavering support, patience and guidance throughout my studies at UFJF. His expertise and encouragement were instrumental in shaping this work and my growth as a researcher. Moreover, I am grateful to my co-supervisors, Prof. Dr. Frederico Gadelha Guimarães and Prof. Dr. Christian Wagner, for all their support and contributions. I am incredibly proud and feel truly fortunate to have had the opportunity to work with them.

To Prof. Dr. Petrônio Cândido de Lima e Silva for his invaluable help. His generosity in sharing knowledge and willingness to help will never be forgotten.

To all colleagues and teammates at LAIIC - Industrial Automation and Computational Intelligence Lab, always willing to help and contribute.

To my friends and relatives, for the moments of distraction and rest.

To UFJF - Universidade Federal de Juiz de Fora and CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico for the structure and financial support, so important along this journey. Especially, thanks to IFMG - Instituto Federal de Minas Gerais for the opportunity of full dedication for the conclusion of this work. Their contributions were fundamental in enabling the development and completion of this research.

Thank you God, for my health and for putting me on this path that I love so much!

ABSTRACT

Fuzzy Set Based Time Series (FTS) prediction techniques offer potential advantages in efficient and intuitive data partitioning and the effective handling of uncertainty in the data. However, such prediction models are commonly challenging to design, requiring careful and application-specific tuning of hyperparameters to provide competitive forecasting performance. Conventional FTS models often rely on predefined partitioning schemes and user-specified hyperparameters, which may introduce subjectivity and limit their adaptability to complex datasets. This thesis presents novel data-driven interval type-2 fuzzy set based time series models — SODA-T2FTS, ADP-T2FTS, and ADP-T2LIMG — that integrate interval type-2 fuzzy logic with advanced partitioning techniques, namely the Self-Organized Direction Aware Data (SODA) and Autonomous Data Partitioning (ADP) algorithms. These approaches handle epistemic uncertainty and improve predictive performance while reducing reliance on user intervention. The models were evaluated on financial, benchmark, and energy time series datasets, and evaluation was overall performed in terms of the average number of rules (c.f. interpretability), error metrics, execution time, model complexity and noise sensitivity. Results showed that the proposed models presented superior accuracy, lower forecasting errors, and competitive interpretability compared to state-of-the-art forecasting techniques, also highlighting that data-driven approaches significantly enhance fuzzy set based time series forecasting by improving partitioning accuracy, reducing subjectivity, and increasing adaptability to different datasets.

Keywords: Type-2 Fuzzy Systems. Forecasting. Time Series Analysis. Type-2 Fuzzy Set Based Time Series. Data-driven partitioning.

RESUMO

As técnicas de previsão de séries temporais fuzzy (FTS) oferecem vantagens potenciais na realização de um particionamento de dados eficiente e intuitivo, além do tratamento eficaz da incerteza presente nos dados. No entanto, esses modelos de previsão geralmente apresentam desafios em seu desenvolvimento, exigindo um ajuste cuidadoso e específico de hiperparâmetros para garantir um desempenho competitivo na previsão. Os modelos convencionais de FTS frequentemente dependem de esquemas de particionamento predefinidos e parâmetros especificados pelo usuário, o que pode introduzir subjetividade e limitar sua adaptabilidade a conjuntos de dados complexos. Esta tese apresenta novos modelos de séries temporais fuzzy intervalares do tipo-2 baseados em particionamento orientado por dados — SODA-T2FTS, ADP-T2FTS e ADP-T2LIMG — que integram a lógica fuzzy intervalar do tipo-2 com técnicas avançadas de particionamento, especificamente os algoritmos Self-Organized Direction Aware Data (SODA) e Autonomous Data Partitioning (ADP). Essas abordagens capturam a incerteza epistêmica e melhoram o desempenho preditivo, reduzindo a necessidade de intervenção do usuário. Os modelos foram avaliados em séries temporais financeiras, de referência e de energia, e a comparação foi realizada, de forma geral, em termos do número médio de regras (interpretabilidade), métricas de erro, tempo de execução, complexidade do modelo e sensibilidade ao ruído. Os resultados mostraram que os modelos propostos apresentaram maior precisão, menores erros de previsão e interpretabilidade competitiva em comparação com as técnicas de previsão mais avançadas da literatura. Além disso, os resultados destacam que abordagens baseadas em dados melhoram significativamente a previsão de séries temporais fuzzy (FTS) ao aprimorar a precisão do particionamento, reduzir a subjetividade e aumentar a adaptabilidade a diferentes conjuntos de dados.

Palavras-chave: Sistemas Fuzzy do Tipo-2. Previsão. Análise de Séries Temporais. Séries Temporais Baseadas em Lógica Fuzzy do Tipo-2. Particionamento baseado em dados.

Lista de Figuras

Lista de Tabelas

LISTA DE ABREVIATURAS E SIGLAS

ACF	*Autocorrelation Function*
ADP	*Autonomous Data Partitioning algorithm*
AEP	*American Electric Power dataset*
ARIMA	*AutoRegressive Integrated Moving Average*
CFTS	*Conventional Fuzzy Set Based Time Series*
CNN	*Convolutional Neural Networks*
DEOK	*Ohio/Kentucky's Duke Energy dataset*
EBOP	*European Brent Oil Price Dataset*
EIASC	*Enhanced Iterative Algorithm with Stop Condition*
EWFTS	*Exponentially Weighted Fuzzy Set Based Time Series*
FLR	*Fuzzy Logical Relationship*
FLRG	*Fuzzy Logical Relationship Group*
FOU	*Footprint of Uncertainty*
FRA	*Friedman Aligned Ranks test*
FTS	*Fuzzy Set Based Time Series*
HOFTS	*High Order Fuzzy Set Based Time Series*
IT2FS	*Interval Type-2 Fuzzy Set*
IWFTS	*Improved Weighted Fuzzy Set Based Time Series*
LMF	*Lower Membership Function*
LSTM	*Long Short-Term Memory*
MWFTS	*Markov Weighted FTS*
NASDAQ	*National Association of Securities Dealers Automated Quotations*
NREL	*National Renewable Energy Laboratory*
PWFTS	*Probabilistic Weighted Fuzzy Set Based Time Series*
R-HFCM	*Randomized High-Order Fuzzy Cognitive Maps*
RMSE	*Root Mean Squared Error*
S&P 500	*Standard & Poor's 500 Index*
SMAPE	*Symmetric Mean Absolute Percentage Error*
SNR	*Signal-to-Noise-Ratio*
SONDA	*Brazilian National System of Environmental Data Organization dataset*
SODA	*Self-Organized Direction Aware Data Partitioning*
T1FS	*Type-1 Fuzzy Set*
T2FS	*Type-2 Fuzzy Set*
T2FTS	*Type-2 Fuzzy Set Based Time Series*
TAIEX	*Taiwan Stock Exchange Capitalization Weighted Stock Index*
TWFTS	*Trend Weighted Fuzzy Set Based Time Series*
UMF	*Upper Membership Function*
WFTS	*Weighted Fuzzy Set Based Time Series*

LISTA DE SÍMBOLOS

AD^G	ADP's Global density
ALD	ADP's Local density
$\bar{\mu}_{\tilde{A}}(y_i)$	Upper membership function (UMF)
c_{L_f}	Lower limit of type-reduced fuzzy set
c_{R_g}	Upper limit of type-reduced fuzzy set
D	SODA's Local density
D^G	SODA's Global density
d_A	Angular component
d_M	Magnitude component
$f(t)$	Fuzzified time series function at time t
L_f	Left switch point
$\mathcal{M}$	Learned fuzzy model after training procedure
$\max()$	Maximum operator (s-norm) used in consequent processing
$\min()$	Minimum operator (t-norm) used in antecedent processing
$\mu_{\tilde{A}}$	Membership function of a fuzzy set $\tilde{A}_i$
$\pi_n^M(\boldsymbol{x}_i)$	Cumulative Proximity of x_i
R_g	Right switch point
$\theta_{\boldsymbol{x}_i,\boldsymbol{x}_j}$	Angle between vectors x_i and x_j in cosine similarity
$\tilde{A}$	Linguistic variable
$\tilde{A}_i$	Type-2 Fuzzy set
$\underline{\mu}_{\tilde{A}}(y_i)$	Lower membership function (LMF)
UoD	Universe of Discourse
UoD_{sub}	Internal Universe of Discourse used in ADP-T2LIMG
$Y(t)$	Time Series
$y(t)$	Instance of a time series at time t
$\dot{y}_i$	Actual observed values for comparison with forecasts

CONTENTS

1 Introduction

Time series forecasting is a technique widely studied in academic research, which consists of analysing data and the sequence of time in order to predict future events. Shumway and Stoffer (2017) describe the twofold purpose of time series prediction, which is to understand or model the mechanism that gives rise to an observed time series and to predict or forecast its future values based on historic values. However, uncertainty, challenging behaviors, and patterns within specific time series data might make it challenging to examine and forecast. Dealing with the uncertainty in time series data, in particular, is a recurrent problem in forecasting and decision-making, especially in complex systems where data is often imprecise, incomplete, or affected by noise.

In real-world applications, uncertainties may arise due to various factors, including measurement errors, missing data, and the intrinsic variability of the observed phenomenon (Box et al., 2015), and can be categorized as aleatoric and epistemic uncertainty (Kiureghian; Ditlevsen, 2009). Aleatoric uncertainty (also known as irreducible uncertainty), originates from the inherent randomness in the data generation itself and represents variability that cannot be reduced, even with more data collection or improved models. In financial markets, for instance, price fluctuations due to investor sentiment, geopolitical events, or unforeseen economic changes introduce aleatoric uncertainty that no model can eliminate entirely, making precise prediction inherently difficult. Similarly, in climate science, turbulence in atmospheric dynamics introduces natural variability that limits forecast precision. On the other hand, Epistemic uncertainty arises due to a lack of knowledge about the true data-generating process. This type of uncertainty is associated with the limitations of the chosen model, insufficient training data, or incorrect assumptions about the system being modeled. Unlike aleatoric uncertainty, epistemic uncertainty can be reduced by improving the model, incorporating more representative data, or refining feature selection techniques. For instance, an energy time series may summarize all daily energy production into energy per second, minute, or hour, introducing the epistemic uncertainty on data.

Addressing these uncertainties requires forecasting models to be capable of accounting for vagueness, imprecision, and randomness in time series data. Due to this fact, several models have been used in the literature to predict variables, such as traditional time series regression, exponential smoothing, ARIMA, ensemble learning techniques, such as random forests and boosting, and more recently, deep learning models employed to generate probabilistic forecasts, capturing both aleatoric (data-related) and epistemic (model-related) uncertainties.

Fuzzy Set Based Time Series (FTS) models have emerged as an effective forecasting approach to handling uncertainty by representing time series data in linguistic terms and incorporating fuzzy logic principles. Unlike conventional statistical models, which rely on precise numerical values, FTS models define time series observations using type-1 fuzzy sets, allowing for a more flexible representation of uncertainty (Singh, 2017). It combines performance and low computational costs and has been used to solve numerous forecasting problems such as environment-related issues (Koo et al., 2020; Dec et al., 2021; Wang; Li; Lu, 2018a), stock market (Shafii et al., 2019; Wu et al., 2021; Musikasuwan; Septiarini, 2020; Cheng; Chen; Jian, 2016), solar energy (Orang et al., 2020), enrollments (Hieu; Ho; Lan, 2020) and even COVID cases (Tinh, 2020; Kumar; Susan, 2021). The main reason for FTS popularity is that linguistic values or fuzzy sets are used to generate the desired output rather than crisp numbers (Zhang et al., 2020).

Fuzzy logic was first proposed in 1965 by Zadeh (1996). Song and Chissom (1993) introduced the FTS method by implementing it on the data of student enrollments at the University of Alabama. Later, Chen et al. (1996) introduced a method using simplified arithmetic operations instead of the complex max-min composition operations proposed by Song and Chissom. This model is considered as the milestone in this research area (Bose; Mali, 2019). FTS was proposed to overcome the limitations of statistical time series models, such as the autoregressive (AR), moving average (MA), and autoregressive moving average (ARMA) models, that have limitations in dealing with uncertainty in time series (Dincer; Akkuş, 2018). On the other hand, FTS achieves realistic results with higher accuracy rates in forecast output (Gupta; Kumar, 2019). In terms of research advancements, FTS early works mainly focused on improving the basic FTS model, but over the years, the research and application of FTS have expanded significantly. FTS theory has been combined with machine learning techniques to create forecasting models that aim to leverage the strengths of both methodologies: fuzzy logic for interpretability and uncertainty handling and machine learning for pattern recognition and predictive accuracy (Zhan et al., 2024; Panigrahi; Behera, 2020; PhamToan; VoThiHang; PhamThi, 2024).

Epistemic uncertainty in FTS models is primarily introduced by the modeling assumptions, particularly in the data partitioning step, which defines how the time series data is divided into intervals for the creation of fuzzy sets. For this, some hyperparameters must be defined mainly the partitioning method, the number of partitions and model order (Gao; Duru, 2020). Thus, one major source of epistemic uncertainty in FTS partitioning is the subjective determination of the number and shape of fuzzy intervals. Classical approaches, such as equal-length and equal-frequency partitioning, assume a fixed number of intervals based on heuristic rules, which may not always capture the true data distribution. Since these methods do not adapt to the underlying variability of the data, they may introduce uncertainty due to misrepresentation of time series dynamics. Besides that, partitioning or clustering techniques that require user interference may introduce subjective bias and may not adapt well to dynamic datasets. To mitigate this, researchers have proposed data-driven partitioning strategies, such as entropy-based, clustering-based, and optimization-based methods, to reduce epistemic uncertainty by adapting the partitioning structure to the actual statistical properties of the data. These approaches dynamically adjust the number of partitions based on measures such as information entropy, data density, or prediction errors, ensuring that the fuzzy sets better represent the variability in the time series.

Since FTS models rely on linguistic representation through fuzzy sets, the choices regarding the hyperparameters for this partitioning procedure significantly impact model accuracy and robustness (Huarng, 2001). If the partitioning is not optimally determined, it introduces epistemic uncertainty due to incorrect assumptions about the structure of the data. For example, using an overly coarse partitioning may lead to a loss of important details, whereas an excessively fine partitioning can lead to overfitting and an increase in computational complexity (Chen; Tanuwijaya, 2011). Another important consideration in reducing epistemic uncertainty is the shape of the membership functions used in FTS data partitioning. Improper selection of membership functions can introduce additional uncertainty, as different shapes influence the degree of membership in which input values are assigned to fuzzy sets (Dixit; Jain, 2023). As epistemic uncertainty is reducible, by refining both the partitioning process and membership function design, researchers can systematically reduce epistemic uncertainty and improve the forecasting performance of FTS models.

Although many improvements have been made in FTS forecasting models, there are limitations of type-1 fuzzy sets to model and minimize the impact of uncertainties expected in some applications. Therefore, type-2 fuzzy sets were presented by Zadeh (1975) as an extension of the concept of a conventional fuzzy set, providing an additional degree of freedom to model uncertainty and imprecision in a better way (Castillo et al., 2007; Pinto et al., 2021). The key advantage of type-2 fuzzy sets (T2FS) over type-1 fuzzy sets (T1FS) is their ability to model uncertainty within the membership functions themselves. In a T1FS, each input value is assigned a crisp membership degree, whereas in a T2FS, the membership function is itself fuzzy, meaning that each input has a range of possible membership values (Abhishekh; Gautam; Singh, 2018). This capability is particularly useful in FTS models, where epistemic uncertainty arises from imperfect knowledge of the system, subjective partitioning choices, and unknown relationships between past and future values. Thus, type-2 fuzzy sets allow for a more adaptive modeling approach by incorporating an additional level of uncertainty in the decision-making process, enabling better handling of noisy, incomplete, or sparse data, as well as improved robustness when forecasting time series with sudden fluctuations (Mendel, 2007).

Despite significant improvements published in the type-2 FTS literature in the past years, much still needs to be studied. There are several hyperparameters to be defined in the type-2 FTS design; however, one of the most recurrent challenges in the FTS literature is defining the number and length of intervals of the universe of discourse

- *UoD* in the model as these are parameters that directly impact in model's accuracy (Singh, 2017), also considering that most of the partitioning/clustering methods used for this task often rely on user-defined parameters for partitioning, introducing subjectivity and limiting adaptability to complex datasets. Therefore, to address the difficulty in the FTS universe of discourse partitioning and uncertainty handling, this research aims to investigate FTS forecasting model design by understanding the importance of key design hyperparameters, to propose new accurate and effective type-2 FTS methods and to validate their performance in different appropriate case studies, as illustrated in Figure 1.

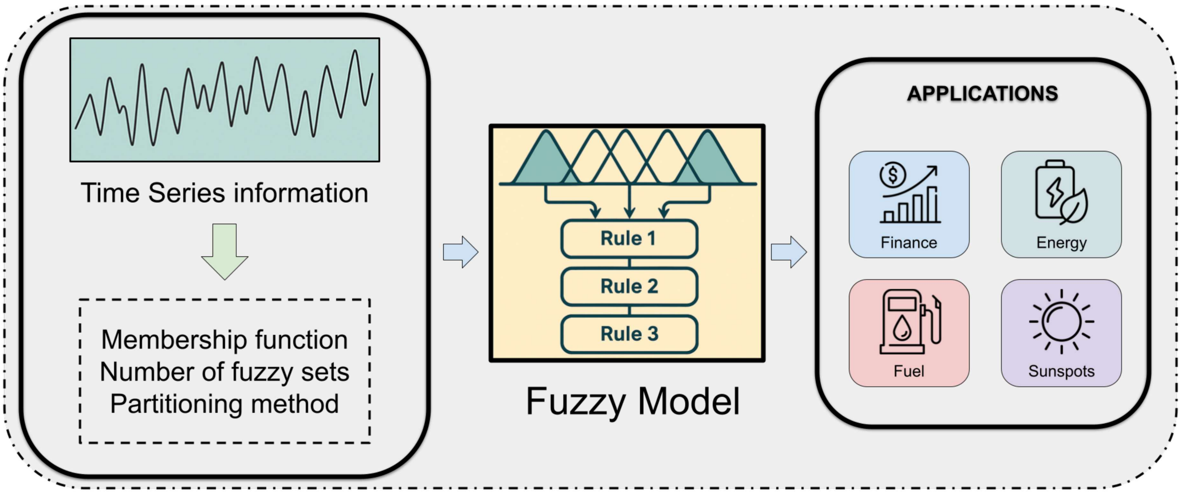


Figure 1 – Overall forecasting process in this research.

1.1 Main Contributions

This thesis presents contributions to the Time Series Forecasting research field, summarized below:

- Introduction of fully data-driven partitioning algorithms (SODA and ADP) that remove reliance on manual heuristics, thereby addressing a key source of epistemic uncertainty in type-2 FTS model design.
- Development of the SODA-T2FTS and ADP-T2FTS models to address the limitations of user-dependent partitioning methods in conventional FTS, enabling more adaptive, accurate, and uncertainty-aware forecasting for univariate time series.
- Extension of ADP-T2FTS into ADP-T2LIMG to capture deeper structural information from the data, improving fuzzy set definition and enhancing forecast accuracy in complex time series.

- Comprehensive evaluation of the proposed models on complex financial, energy and benchmark time series, also analyzing their performance in terms of the average number of rules (c.f. interpretability), error metrics, execution time, model complexity and noise sensitivity.

- Empirical benchmarking against diverse state-of-the-art FTS and machine learning methods to validate the effectiveness of the proposed data-driven type-2 FTS models across different modeling strategies.

1.2 Publications

The following papers have been published by the author throughout the course. The items marked with a (*) indicate the publications not related to the thesis itself.

A.1 Journal Publications

1. VARGAS PINTO, A. C., SILVA, L. C. C. SILVA, P. C. L., GUIMARÃES, F. G., WAGNER, C., PESTANA DE AGUIAR, E. Autonomous data partitioning for type-2 fuzzy set based time series. *Evolving Systems*, v. 15, n. 2, p. 575-590, 2024.

2. *DIAS, U. R. F., VARGAS PINTO, A. C., MONTEIRO, H. L. M., PESTANA DE AGUIAR, E. New perspectives for the intelligent rolling stock classification in railways: an artificial neural networks-based approach. Journal of the Brazilian Society of Mechanical Sciences and Engineering, v. 46, n. 4, p. 230, 2024.

3. VARGAS PINTO, A. C., FERNANDES, T. E., SILVA, P. C. L., GUIMARÃES, F. G., WAGNER, C., PESTANA DE AGUIAR, E. Interval Type-2 Fuzzy Set Based Time Series Forecasting Using a Data-driven Partitioning Approach. *Evolving Systems*, v. 13, n. 5, p. 703-721, 2022.

4. *OLIVEIRA, J. V. P., COELHO, A. L. F., SILVA, L. C. C., VIANA, L. A., VARGAS PINTO, A. C., PINTO, F. A. C., Filho, D. O. Using image pre-mapping for applications of monitoring electrical switchboards. *Automation in Construction*, 112, 103091. 2020.

A.2 Conference Publications

1. VARGAS PINTO, A. C., ORANG, O., GUIMARÃES, F. G., PESTANA DE AGUIAR, E. A new data driven forecasting approach for time series forecasting. *In:* IEEE INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS (FUZZIEEE), 2025. - Accepted (To be presented in July/2025).

2. VARGAS PINTO, A. C., SILVA, P. C. L., GUIMARÃES, F. G., WAGNER, C., PESTANA DE AGUIAR, E. Self-Organised Direction Aware Data Partitioning for

Type-2 Fuzzy Time Series Prediction. *In:* IEEE INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS (FUZZIEEE), Luxembourg. p. 1-6, 2021.

3. VARGAS PINTO, A. C.,SILVA, P. C. L., GUIMARÃES, F. G., PESTANA DE AGUIAR, E. Séries Temporais Fuzzy do Tipo-2. *In:* ANAIS DO CONGRESSO BRASILEIRO DE AUTOMÁTICA (CBA), v. 2, n. 1, 2020.
4. VARGAS PINTO, A. C., SANTA MARIA, T. H., OLIVEIRA, J. G., MARCATO, A. L. M., PESTANA DE AGUIAR, E. Controlador MPPT baseado em Lógica Fuzzy Aplicado a um Sistema Fotovoltaico Conectado à uma Rede. *In:* ANAIS DO 14º SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE (SBAI), Ouro Preto, 2019.

1.3 Work Structure

This thesis is organized as follows:

- Chapter 2 – Theoretical Foundations: explores the fundamental concepts of Fuzzy Set Based Time Series forecasting, also discussing type-2 fuzzy sets and detailing partitioning methodologies in FTS, providing the necessary background for the proposed methodologies.
- Chapter 3 – The SODA-T2FTS Approach: presents the SODA-T2FTS forecasting model, explaining its training and forecasting procedures. It describes how SODA is used to partition the UoD and details the experimental results obtained for multiple financial time series datasets, comparing its performance to existing state-of-the-art FTS models.
- Chapter 4 – The ADP-T2FTS Approach: introduces ADP-T2FTS and describes the model training procedure, fuzzy rule extraction, and forecasting steps. A detailed comparative analysis with other partitioning methods shows that ADP-T2FTS achieves lower RMSE values and superior robustness against noise, making it a highly effective data-driven FTS model.
- Chapter 5 – The ADP-T2LIMG Approach: presents ADP-T2LIMG, an improved version of ADP-T2FTS that extracts additional structural information from ADP-generated data clouds. This enhanced method allows for more accurate fuzzy set design, further improving forecasting precision. The chapter outlines the training and forecasting procedures of ADP-T2LIMG, as well as comprehensive computational experiments that validate its superior performance compared to state-of-the-art machine learning and deep learning models.

- Chapter 6 – Conclusion and Future Works: This final chapter summarizes the key findings of this research, emphasizing the benefits of data-driven partitioning and type-2 fuzzy logic in time series forecasting. It also discusses potential research future directions.

2 Theoretical Foundations

This chapter presents the theoretical background necessary to support the development of the models proposed in this research. It begins by introducing the fundamentals of type-2 fuzzy sets, with a focus on interval type-2 fuzzy sets (IT2FS), highlighting their advantages in managing epistemic uncertainty. Then there is a review of Fuzzy Set Based Time Series Prediction (FTS) forecasting, outlining the key concepts, procedures, and motivations behind its use in modeling uncertain and imprecise temporal data. Next, it explores the importance of partitioning the Universe of Discourse (UoD), discussing how partitioning choices directly influence model performance and uncertainty handling. Finally, it reviews the data-driven partitioning approaches to be used in the models proposed in subsequent chapters.

2.1 An overview of Interval Type-2 Fuzzy Sets

The very nature of time series causes its analysis and forecast to be affected by numerous variables that add uncertainties to the model. Factors such as trend, seasonality and cyclical movements end up making it difficult to build an accurate forecasting model. In this sense, the operating structure of type-2 fuzzy sets, first proposed by Zadeh (1975), can contribute to the construction of a more robust and tolerant system to noise and uncertainties in the data.

Type-2 fuzzy logic extends traditional (Type-1) fuzzy logic by incorporating an additional level of uncertainty, making it particularly useful for handling imprecise and highly uncertain data. Formally, a type-2 fuzzy set $\tilde{A}$ in a domain X is characterized by a type-2 membership function $\mu_{\tilde{A}}(x, u)$, which is defined as shown in Equation (2.1). This function maps each element x to a fuzzy set in the interval $[0, 1]$.

$$\tilde{A} = \left\{ ((x, u), \mu_{\tilde{A}}(x, u)) \mid x \in X, u \in J_x \subseteq [0, 1] \right\}, \tag{2.1}$$

where:

- x is an element in the universe X;
- u is a secondary variable that belongs to the primary membership domain $J_x \subseteq [0, 1]$;
- $\mu_{\tilde{A}}(x, u)$ represents the secondary membership function, which determines the degree of membership of u for a given x.

Researchers have proposed adjustments to improve the level of accuracy or performance of fuzzy systems. There is a substantial computational bottleneck since the primary membership of type-2 FTS may be infinite in number because it is represented by a subinterval in [0, 1]. Hence, in order to solve this issue, Liang and Mendel (2000) proposed a theory of a type-2 fuzzy set as interval type-2 fuzzy set (IT2FS) by considering the secondary membership function to be 1. Subsequently, this concept emerged as the most broadly used because of its successful utilization in several decision-making problems (Biswas; De, 2018).

As seen, Interval Type-2 Fuzzy Sets (IT2FS) are then a special case of general Type-2 Fuzzy Sets (T2FS), where the secondary membership functions are always equal to 1, i.e. $\mu_{\tilde{A}}(x,u) = 1, \quad \forall u \in J_x$. This simplification reduces computational complexity while still capturing uncertainty more effectively than traditional Type-1 Fuzzy Sets. Thus, an interval type-2 fuzzy set can defined by an upper membership function (UMF) and lower membership function (LMF), both equivalent to a traditional type-1 membership function. The lower membership function is less than or equal to the upper membership function for all possible input values, as shown in Figure 2. The blue region between the UMF and LMF is the footprint of uncertainty (FOU), which can be expressed as Equation (2.2), where x is a variable in X, u is the primary membership of x and $\underline{\mu}_{\widetilde{A}}$ and $\bar{\mu}_{\widetilde{A}}$ represent the lower and upper membership functions of $\tilde{A}$, respectively (McCulloch; Wagner, 2020).

$$FOU(\tilde{A}) = \{(x,u) \mid x \in X, u \in [\ \underline{\mu}_{\widetilde{A}}(x), \bar{\mu}_{\widetilde{A}}(x)]\} \tag{2.2}$$

The FOU is what provides interpretability to type-2 fuzzy sets when dealing with uncertainties, especially when defining the end-points for the membership functions, providing a direct way to model and quantify epistemic uncertainty, which arises due to incomplete or imprecise knowledge.

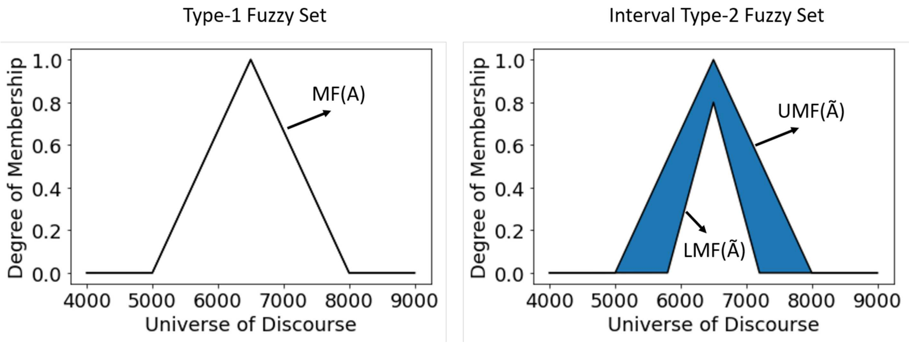


Figure 2 – Type-1 and type-2 triangular fuzzy sets (third dimension not shown for IT2 case).

While in type-1 fuzzy sets the degree of membership is characterized by a single crisp value; in type-2 fuzzy sets, the degree of membership is considered a fuzzy set, which allows incorporating the treatment of uncertainties in the data and in the construction of the fuzzy sets, making it more suitable to deal with variations in the noise at the inputs (Shukla; Muhuri, 2019).

Regarding the use of type-2 fuzzy logic in FTS models, Huarng and Yu (2005) proposed its use to enrich the fuzzy relationships obtained from type-1 models and then to improve the forecasting performance of the Taiwan stock index. Bajestani and Zare (2011) proposed a new method to forecast the Taiwan stock index based on optimized high-order type-2 FTS. A weighted type-2 FTS forecasting method to enhance the model's performance was proposed by Abhishekh, Gautam and Singh (2018). Furthermore, Mittal et al. (2020) discussed the essential work in the field of type-2 FTS, including its theoretical and practical implications.

2.1.1 Fuzzy logic systems

Fuzzy logic systems - FLS use type-1 or type-2 fuzzy sets to produce output crisp values from input crisp values. As shown in Figure 3, a type-2 FLS is similar to a type-1 FLS. The main difference between type-2 FLS and type-1 FLS is the type-reducer, which converts the output type-2 fuzzy set for later defuzzification. The complete structure of a type-2 FLS includes a fuzzifier, the fuzzy rule base, the fuzzy inference engine, and the output processing block, which is composed of a type-reducing mechanism followed by a defuzzifier. The output processing block is responsible for performing the operations that will convert the resulting fuzzy set into a crisp value that can be used.

In a Fuzzy Logic System, rules serve as the foundation for decision-making in the inference engine. These rules can either be derived from expert knowledge or extracted directly from numerical data. In both cases, they are usually represented as a series of IF–THEN statements that consists of two main components: the antecedent (the IF-part), which defines the conditions for the rule activation, and the consequent (the THEN-part), which specifies the outcome of the given rule. Fuzzy sets (FSs) are associated with the linguistic terms used in these rules, influencing both the inputs and outputs of the FLS. By structuring decision-making through these fuzzy rules, a FLS can handle uncertainty and approximate reasoning effectively.

The two most widely used types of fuzzy inference systems are the Mamdani and Sugeno, which differ primarily in their rule structure, output calculation, and defuzzification methods. The Mamdani inference system operates in four main stages: fuzzification, rule evaluation, aggregation, and defuzzification. In fuzzification crisp inputs are converted into fuzzy sets using predefined membership functions. Then, fuzzy rules, often expressed in an "IF-THEN"format, are applied. Once all the rules are evaluated, their outputs are aggregated into a single fuzzy set. Defuzzification is then required to convert this fuzzy set into a crisp value. The Sugeno inference system differs from the Mamdani model primarily in its output structure. Instead of fuzzy sets, the Sugeno model generates outputs in the

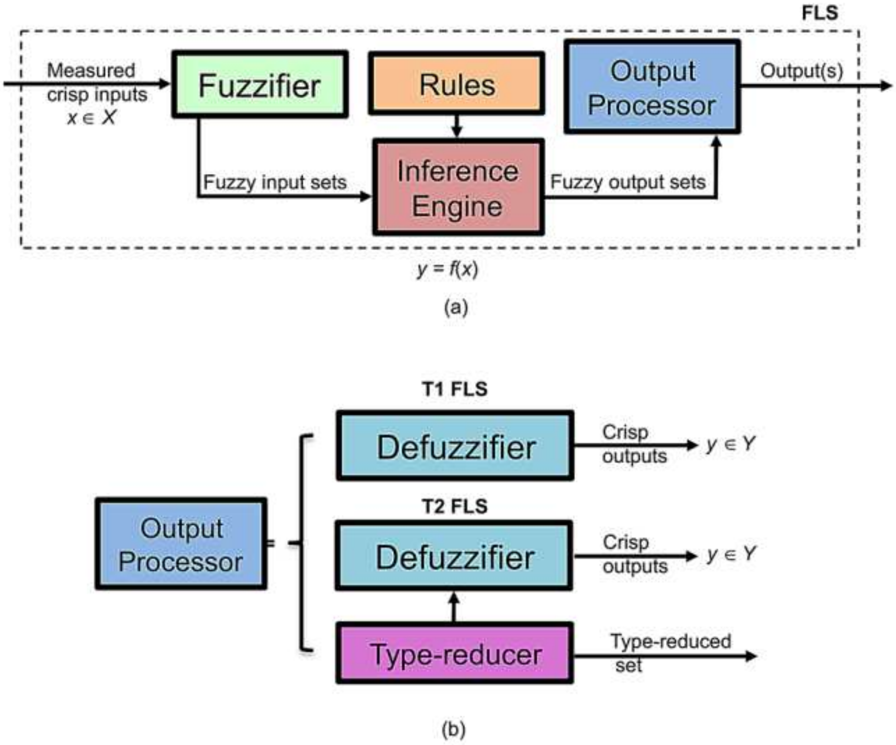


Figure 3 – (a) Components of a FLS, and (b) nature of the output processor for T1 and T2 FLSs (Aguiar et al., 2021).

form of mathematical functions, eliminating the need for a separate defuzzification step, making the Sugeno approach computationally more efficient but less interpretable than Mamdani models.

Processing the input values in the Inference Engine involves several computational steps, that rely on rules that consist of the antecedent and the consequent. After the input data is fuzzified, the antecedent is processed using fuzzy logical operators. If the rule contains multiple antecedents connected by AND, the system typically applies the *minimum* operator, selecting the lowest membership value among the conditions. If the antecedents are connected by OR, the *maximum* operator is used, choosing the highest membership value. After evaluating the antecedent, the consequent processing begins. The rule strength obtained from the antecedent is used to modify the fuzzy set in the consequent. This step, known as implication, adjusts the membership function of the consequent based on the degree of membership of the antecedent. The implication method clips (*min* implication) or scales (*prod* implication) the UMF and LMF of the output type-2 membership function using the rule firing range limits. This process produces an output fuzzy set for each rule. When multiple rules are active, their consequents must be aggregated to derive a single type-2 fuzzy set from the rule output fuzzy sets. This process combines the truncated fuzzy sets using the *maximum* operator, ensuring that the final fuzzy output encompasses contributions from all relevant rules.

2.2 Fuzzy Set Based Time Series Prediction

The values of a fuzzy set based time series are represented by fuzzy sets and there is a relationship between current and previous observations. There are different fuzzy methods proposed to solve time series forecasting problems, from the simplest to the most advanced ones, which involve weighting mechanisms, probabilistic, among others, that improve model performance.

If a conventional time series is composed of observations represented by real numbers, fuzzy set based time series are composed of fuzzy sets, which are all within the universe of discourse - UoD, defined by the range of the values in the original time series. According to Lee et al. (2006) consider that $Y(t)$ $(t = 0, 1, 2, ...)$ represents the crisp (numerical) values of the original time series. After fuzzification, $Y(t)$ is converted into $F(t)$, which is a collection of $\tilde{A}_i(t)$ $(i = 1, 2, ...)$ fuzzy sets. Then, $F(t)$ is called a fuzzy set based time series on $Y(t)$ $(t = 0, 1, 2, ...)$. The group of fuzzy sets $\tilde{A}_i(t)$ $(i = 1, 2, ...)$ can also be understood as a linguistic variable $\tilde{A}$ and each fuzzy set $\tilde{A}_i$ is a linguistic term of that linguistic variable.

The initial model proposed by Song and Chissom (1993) is computationally costly due to its complex matrix max-min operations. Chen et al. (1996) later created a simpler model that is considered to be the milestone of FTS prediction. Chen's method helped defining FTS forecasting models as a sequence of steps, usually followed by researchers in the field, whose main objective is to create a knowledge representation of the time series temporal dynamics. FTS forecasting models procedure can be divided into training and forecasting procedures. The training procedure aims to create the linguistic variable $\tilde{A}$ and a knowledge representation of the time series dynamics, both represented by the learned model $\mathcal{M}$, and involves the following steps: (1) Data transformation (2) Definition of the UoD, (3) Partition of the UoD into intervals, (4) Fuzzification of the dataset, and (5) Establishment of fuzzy logical relationships (FLRs) and fuzzy logical relationship groups (FLRGs). The forecasting procedure is (1) Input value fuzzification, (2) Finding compatible rules, and (3) Defuzzification. The partitioning of the dataset, the identification of fuzzy logical relationships, and the defuzzification are essential in the forecasting performance of the model (Bose; Mali, 2019; Singh, 2017). Each step is described below:

- **Step 1: Data transformation**

  Raw time series data is usually non-stationary, where the statistical properties, such as mean and variance, change over time. They often exhibit trends, seasonality, and noise, which can hide underlying patterns and make it difficult to establish meaningful fuzzy relationships. Many real-world datasets, such as stock prices, temperature variations, and economic indicators, exhibit non-stationarity, which can lead to inaccurate predictions if left unaddressed. Transformations such as normalization, smoothing, and differentiation help mitigate these issues by making the data more suitable for fuzzy modeling.

Differentiation, in particular, is widely used when dealing with non-stationary time series. It refers to computing the difference between consecutive values to capture the rate of change rather than absolute values, reducing non-stationary effects and making it easier to define fuzzy relationships that remain valid across different time periods. Differentiation is applied in pre-processing as $\Delta y(t) = y(t-1) - y(t)$ and in post-processing as $y(t) = y(t-1) + \Delta y(t)$, considering a time series Y.

- **Step 2: Definition of the** UoD

  The UoD defines the boundaries within which the time series data is analyzed and processed. Thus, the UoD refers to the range of all possible values that the time series can assume, providing a structured framework for fuzzification. The UoD is defined using the minimum $(min(Y(t)))$ and maximum $(max(Y(t)))$ values observed in the historical time series data, with an additional margin added to accommodate potential fluctuations, usually defined to extend from 10% to 20% the original UoD.

  Considering the time series represented by $Y(t)$, the UoD is defined in Equation (2.3):

$$UoD = [L_{bound}, U_{bound}] \tag{2.3}$$

  The lower bound is calculated as $L_{bound} = 0.9\ min(Y(t))$ if $min(Y(t))$ is larger than zero and $L_{bound} = 1.1\ min(Y(t))$ if it is not. The upper bound is calculated as $U_{bound} = 1.1\ max(Y(t))$ if $max(Y(t))$ is larger than zero and $U_{bound} = 0.9\ max(Y(t))$ if it is not.

- **Step 3:** UoD **partitioning**

  The UoD is partitioned into k intervals and to each interval a fuzzy set $\tilde{A}_i$ $(i = 0, 1, ..., k)$ is assigned, creating the linguistic variable $\tilde{A}$ used to describe Y. The correct number of intervals (hence fuzzy sets) for FTS models is one of the most important parameters to be defined as it directly affects the model's accuracy. Thus it has been a hot research topic for many researchers that use partitioning and clustering algorithms to fulfill this task. Clustering is the task of dividing data samples into several groups. Data samples in the same groups are more similar to other data samples in the same group and dissimilar to those in other groups. Clustering is widely used in many fields of study, including natural language processing, pattern recognition, image analysis, among others; hence being a relevant topic in the data analytics field. Established clustering techniques require proper parameter settings that can vary significantly across datasets, and their performance heavily depends on prior knowledge and assumptions. Nevertheless, uncertainties about the problem are high in real situations, prior knowledge is usually very limited, and assumptions are rarely met. Thus, for time series forecasting, it is important to find partitioning or clustering algorithms independent of user interference that can analyze the data and partition it according to its structure. This topic will be addressed in details in Subsection 2.2.

- **Step 4: Fuzzification of time series data**

  The fuzzification process is the method to convert the crisp input values of the time series into a linguistic representation, where each $f(t) \in F$ is the fuzzification of $y(t) \in Y$. For every instance in the time series training data, the membership values are calculated considering all fuzzy sets $\tilde{A}_i$ $(i = 1, 2, ..., k)$ generated in the previous step. A fuzzy set $\tilde{A}_i$ in the UoD is defined by Equation (2.4).

  $$\tilde{A}_i = \frac{\mu_{\tilde{A}_i}(u_1)}{u_1} + \frac{\mu_{\tilde{A}_i}(u_2)}{u_2} + \cdots + \frac{\mu_{\tilde{A}_i}(u_n)}{u_n} \tag{2.4}$$

  where $\mu_{\tilde{A}_i}$ is the membership function of a fuzzy set $\tilde{A}_i$ and $\mu_{\tilde{A}_i}(u_j)$ indicates the membership degree of u_j in $\tilde{A}_i$.

  The fuzzy sets may be designed using different membership functions. Although the membership function has a minor impact on model forecasting accuracy, it is still an important parameter to be defined. In general, membership functions are usually triangular, trapezoidal, or gaussian.

- **Step 5: Extraction of temporal patterns**

  After the fuzzification process, fuzzy logical relationships (FLR) can be established. FRLs are used to represent the temporal patterns between lags in F and relate previous and current states. Suppose a first ordel model and that $\tilde{A}_i$ is the fuzzified value of period t and $\tilde{A}_k$ is the fuzzified value of period $t + 1$. In that case, the fuzzy logical relationship FLR is defined as $\tilde{A}_j \rightarrow \tilde{A}_k$, where, $\tilde{A}_j$ (left hand-side) is the current state and $\tilde{A}_k$ (right hand-side) is the next state of time series historical data. A simple and generic understanding of the FTS process is represented in Figure 4, in which the instances of a time series are fuzzified into fuzzy sets and then the temporal patterns from one instance to the next are identified. After all FLRs have been identified for all the time series instances, the knowledge rule-base can be generated by the creation of the fuzzy logical relationship groups (FLRG). All the rules which share the same antecedent (left hand-side) can be grouped in a FLRG that can be interpreted as the possible future states (right hand-side) for a given antecedent, as exemplified in Figure 5. For example, The FLRG $\tilde{A}_{\mathrm{j}} \rightarrow \tilde{A}_{\mathrm{k}}, \tilde{A}_{\mathrm{t}}, \tilde{A}_{\mathrm{g}}$ implies that IF $f(t) \in F$ is $\tilde{A}_{\mathrm{j}}$ THEN $f(t+1)$ is $\tilde{A}_{\mathrm{k}}$ OR $\tilde{A}_{\mathrm{t}}$ OR $\tilde{A}_{\mathrm{g}}$.

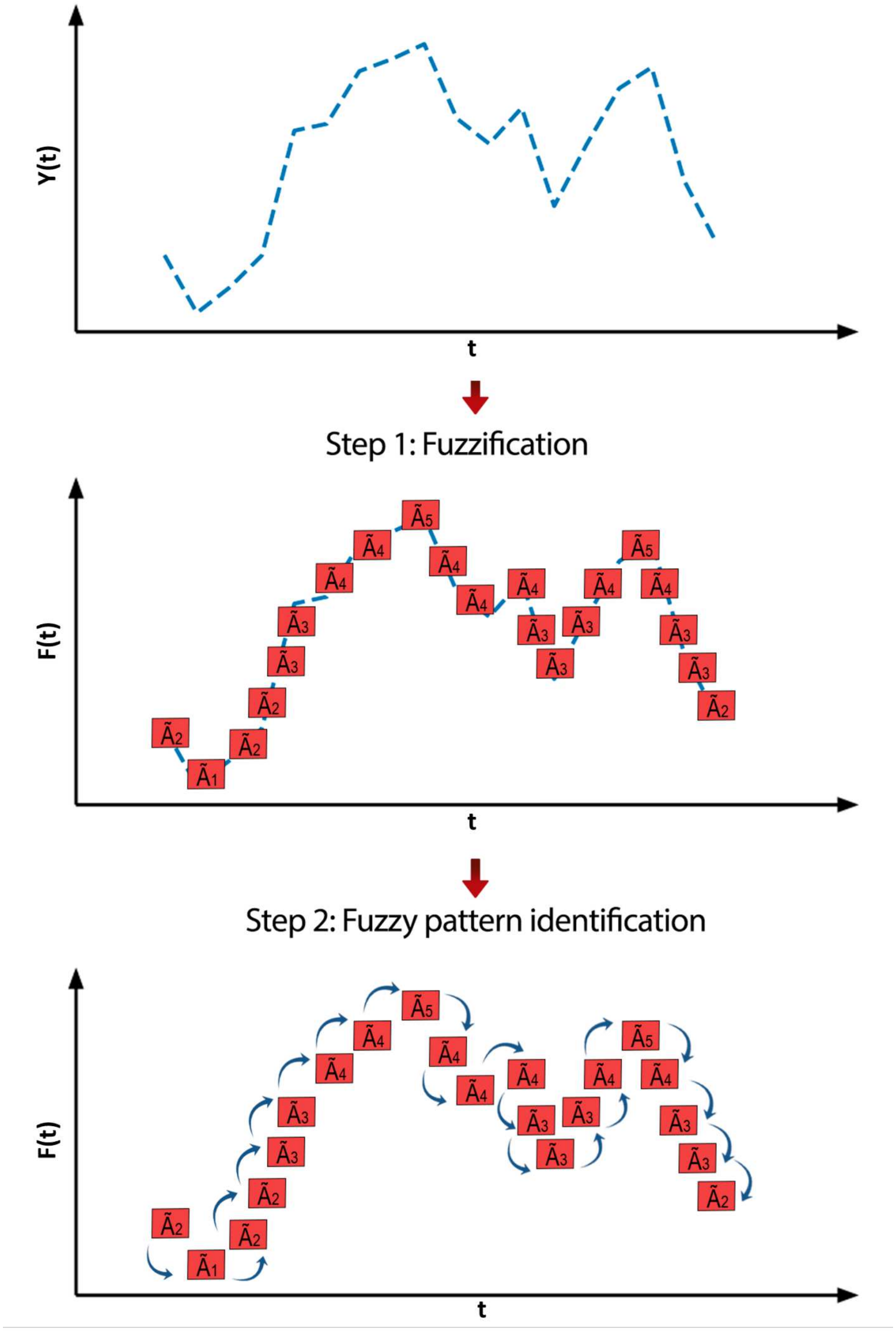


Figure 4 – Generic example of FTS process.

After the training procedure is over, the forecasting procedure happens as follows:

- **Step 1: Input value fuzzification**

  The learned model is then used in the training procedure to forecast the next values of the time series using the values in the test dataset as input. An input value $y(t)$ is converted into fuzzy values of the linguistic variable $\tilde{A}$.

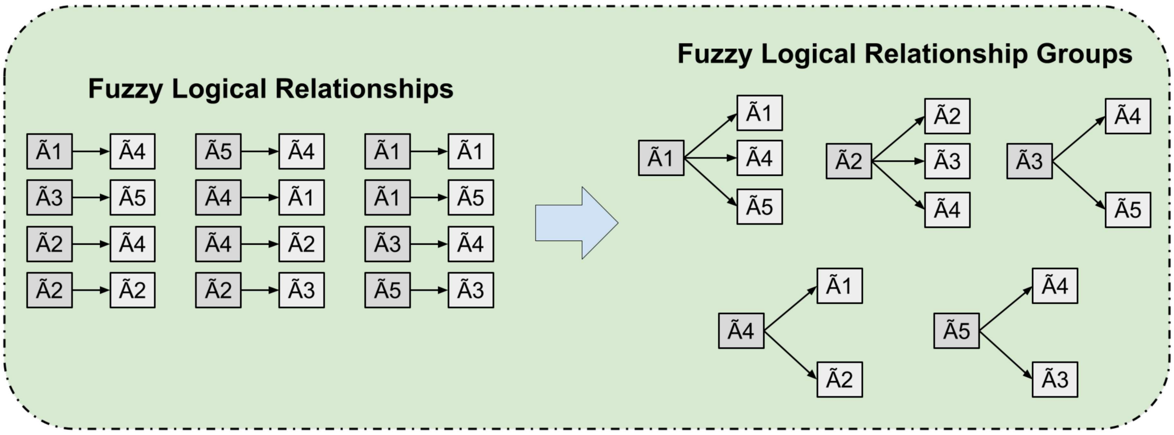


Figure 5 – Generic example of FLRGs definition.

- **Step 2: Finding compatible rules** Knowing which fuzzy sets were activated by the test sample, one can find the rules that are associated with these fuzzy sets and have them as their antecedent. The consequent of these rules will be the possible forecast values $f(t+1)$, considering that the rule structure is the representation of the temporal patterns of consequent instances $\tilde{A}_i \rightarrow \tilde{A}_j$, where $\tilde{A}_i$ is on the left-hand side of time t and $\tilde{A}_j$ is on the right-hand side of time $t+1$.

- **Step 3: Defuzzification**

  The *min* operator is used for antecedent processing (t-norm), which is applied to select the *minimum* values of the Lower Membership Functions - LMFs and Upper Membership Functions - UMFs of all antecedents for every rule, and the output result of antecedent processing is applied to consequent LMFs and UMFs. When the fuzzy sets are triangular membership functions, and when the *min* operator is used, the result is a trapezoidal consequent FOU. Consequent processing (s-norm) is performed using the *max* operator. Aggregation is then performed, which is a process that aims to generate a single interval type-2 fuzzy set from the rule output fuzzy sets.

  Defuzzification is the last step in type-2 FTS models and obtains a crisp number from the output of the aggregated fuzzy set. In type-2 FTS models, defuzzification involves two phases. The first one is the type reduction, which is a conversion of the T2FS, resulting from inference, into a T1FS, and then a conventional defuzzification method is applied. There are several methods of defuzzification, such as the centroid method, bisector method, and mean of maxima (MOM) method, to name a few. In this present research, type-reduction was performed using the Enhanced Iterative Algorithm with Stop Condition - EIASC method. According to Wu and Nie (2011) and Liu and Lin (2021), the EIASC is the most efficient method to use in practice. The approach to be chosen depends on the need of the application (Islam; Hossain; Haque, 2021). The aggregated T2FS is reduced to a T1FS, which is a range with lower limit c_{Lf} and upper limit c_{Rg}, which are calculated using Equations (2.5) and (2.6), where N is the number of samples taken across the output variable range, y_{i} is the *ith* output value sample, $\underline{\mu}_{\tilde{A}}(y_i)$ is the LMF, $\bar{\mu}_{\tilde{A}}(y_i)$ is the UMF, and Lf and Rg are estimated switch points. When defuzzification is performed using the Centroid method (Mendel; Liu, 2007; Mendel et al., 2014; Liu; Wan, 2019), it returns the center of gravity of the type-1 fuzzy set along the x-axis, and is calculated using Equation (2.7).

$$c_{Lf} = \frac{\sum\limits_{i=1}^{Lf} y_i \bar{\mu}_{\tilde{A}}(y_i) + \sum\limits_{i=Lf+1}^{N} y_i \underline{\mu}_{\tilde{A}}(y_i)}{\sum\limits_{i=1}^{Lf} \bar{\mu}_{\tilde{A}}(y_i) + \sum\limits_{i=Lf+1}^{N} \underline{\mu}_{\tilde{A}}(y_i)} \tag{2.5}$$

$$c_{Rg} = \frac{\sum\limits_{i=1}^{Rg} y_i \bar{\mu}_{\tilde{A}}(y_i) + \sum\limits_{i=Rg+1}^{N} y_i \underline{\mu}_{\tilde{A}}(y_i)}{\sum\limits_{i=1}^{Rg} \bar{\mu}_{\tilde{A}}(y_i) + \sum\limits_{i=Rg+1}^{N} \underline{\mu}_{\tilde{A}}(y_i)} \tag{2.6}$$

$$y(x) = \frac{1}{2}[c_{Lf} + c_{Rg}] \tag{2.7}$$

The FTS forecasting model described above will be reviewed and explored in Chapters 3, 4 and 5, with numerical examples and case studies.

Existing FTS methods basically vary by their order and amount of input variables used. Regarding their order, FTS models can be of $h-th$ order, divided into first and high order models. Model order represents the number of past lags that are used in the forecasting process. The number of lags indicate how much past information is available to the model to recognize the possible temporal patterns and make a forecast. Considering the fuzzy set based time series $F(t)$, in first order models only one past lag from the time series is used to forecast the future values and the rule structure is $F(t-1) \rightarrow F(t)$. In a second order model, two past instances are used in the pattern recognition procedure and the rule structure is $F(t-2), F(t-1) \rightarrow F(t)$. For higher order models, h past instances are used to predict $F(t)$ as: $F(t-h), ..., F(t-2), F(t-1) \rightarrow F(t)$. It is important to note that computational costs are usually higher for high order models. Representing $F(t-1)$ and $F(t)$ by $\mathrm{A_i}$ and $\mathrm{A_j}$, respectively, a first order model can be expressed as $A_{\mathrm{i}} \rightarrow A_{\mathrm{j}}$ (Bose; Mali, 2019). Considering their number of input variables, a time series with a single time-dependent variable is called a univariate time series. In multivariate time series more than one variable is varying over time and its forecasting is based on the assumption that each variable is dependent on its past instances as well as the behaviour of other variables.

FTS literature focuses on addressing its disadvantages, such as high computation when the fuzzy rule matrix is large, and the lack of persuasiveness in determining the UoD and the length of intervals (Tinh, 2020; Liu et al., 2010). Hence, researchers have proposed, from different aspects, diverse models based on the FTS theory to enhance forecasting accuracy. Some adjustments are proposed to improve the interval lengths, the order of the fuzzy logical relationships, and defuzzification methods in order to achieve an accurately forecasted output (Dincer; Akkuş, 2018; Jiang et al., 2017).

Specifically, during the fuzzification process, the algorithm needs to perform a division for the UoD, and consequently, construct fuzzy sets. These interval lengths directly influence the forecasting performance. Firstly, Huarng (2001) proposed a distribution-based and average-based to improve forecasting results significantly. Huarng and Yu (2006b) introduced ratio-based lengths of intervals instead of equal lengths of intervals. Afterwards, Yolcu et al. (2009) improved this method by implementing optimization to determine the ratio. Later on, dynamic interval length has also been studied in other research. For example, a hybrid forecasting model which combines particle swarm optimization (PSO) with fuzzy set based time series was proposed by Kuo et al. (2009) to forecast enrollments of students. Kumar and Susan (2021) also proposed a method based on PSO, namely nested FTS-PSO and exhaustive search FTS-PSO for defining the optimal interval length to forecast COVID-19. A model based on a tree partitioning method (TPM) was used for determining the optimal partitions of intervals by Alyousifi, Othman and Almohammedi (2021). This method simplifies the partitioning of the UoD and minimizes the undesirable influences of abnormal data points. It showed the capability to deal with forecasting problems to achieve higher model accuracy successfully.

Concerning the process of establishing fuzzy relationships, Yu (2005) argued that recurrent fuzzy relationships, which were simply ignored in previous studies, should be considered in forecasting. They suggested that different weights should be attributed to different fuzzy relationships. A method based on feed forward neural networks to define fuzzy relationships in high order FTS models was proposed by Aladag et al. (2009), and this proposed method produced better forecasts than the other methods compared. Later, Chen and Wang (2010) introduced a method to forecast the Taiwan Capitalization Weighted Stock Index (TAIEX) based on a fuzzy forecasting method based on fuzzy-trend logical relationship groups. Fuzzy logical relationships were divided into fuzzy logical relationship groups based on the trend of adjacent fuzzy sets emerging in the precursors of fuzzy logical relationships.

Regarding the defuzzification method, Chen et al. (1996) discussed that the mid-points values of the intervals corresponding to the fuzzy sets of the forecasted state are essential information that should be studied to determine the predicted value. Many studies have proved the benefits of this method and its applications (Jiang et al., 2017; Yolcu et al., 2009). Patki et al. (2015) studied the centroid, bisector, and mean of maxima (MOM) methods for defuzzification to predict water quality index.

2.3 Universe of Discourse Partitioning

Partitioning can be performed using equal-width intervals, unequal-width intervals, or autonomous methods like clustering-based approaches. In the context of FTS, where appropriate partitioning is essential for model accuracy and performance, data-driven methods for dividing data into distinct intervals without requiring manual intervention are particularly useful as they work directly from the analysis of data distribution. For a given time series $Y(t)$, the linguistic variable $\tilde{A}$ is used to describe $Y(t)$ by assigning a type-2 fuzzy set $\tilde{A}_i$ $(i = 0, 1, ..., k)$ to each interval of the UoD. These intervals may be generated dynamically based on the distribution of the data, ensuring that the partitions capture meaningful patterns or structures in the dataset while minimizing information loss.

Once the UoD has been defined, the creation of the linguistic variable $\tilde{A}$ is governed by three hyperparameters: the number of partitions (k), the partitioning method and the membership function (μ).

2.3.1 The number of partitions

The number of partitions k in fuzzy set based time series plays a crucial role in determining the effectiveness of the forecasting model. Partitioning refers to dividing the UoD into intervals, with each interval corresponding to a fuzzy set. The choice of the number of partitions directly influences the precision, interpretability, and computational efficiency of the FTS model.

Selecting an appropriate value for k is essential because it affects how well the fuzzy model captures patterns and relationships in the data. There is a trade-off between model performance and k, which can be summarised as:

- **If k is too small**
    - Broad fuzzy sets, leading to loss of information due to excessive generalization.
    - Limits the model's ability to distinguish subtle fluctuations in the data.
    - Higher ambiguity in fuzzy relationships.
    - Leads to generalized forecasts with limited accuracy.
- **If k is too large**
    - The model may become overly sensitive to minor fluctuations in the data.

- Can lead to overfitting, where the model captures noise rather than true patterns.
- Increases computational complexity and reduces interpretability.
- Narrow membership functions, increasing sensitivity but reducing robustness to small variations.

Studies (Guo; Pedrycz; Liu, 2019; Chen; Phuong, 2017; Vamitha, 2021; Askari; Montazerin; Zarandi, 2020) have proved that efficient partitioning significantly impacts the predictive accuracy of FTS models. As seen above, if partitions are too coarse, the model cannot capture fine-grained variations. Conversely, excessive partitions may lead to unnecessary complexity without substantial accuracy improvements.The number of partitions of the UoD must be correctly defined along with another very important hyperparameter, the partitioning method, which will define how the UoD will be partitioned in terms of the length of the partitions, as will be seen in the next subsection.

2.3.2 The partitioning method

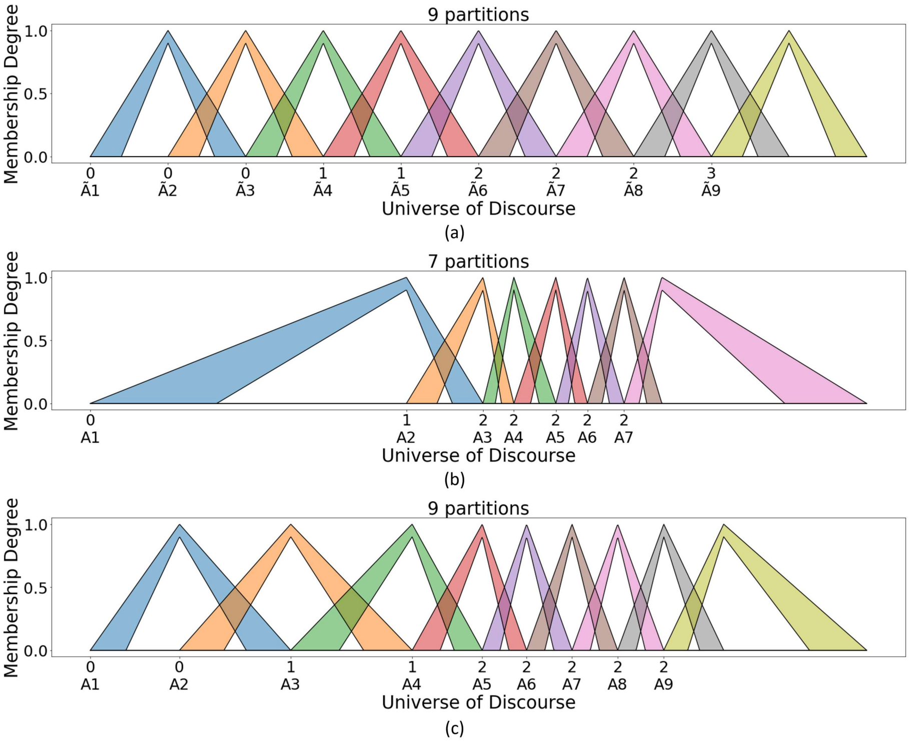
The partitioning method in FTS models will define how the UoD will be divided into intervals and consequently, the length and midpoints for each fuzzy set created. The simplest way of performing this task is using equal-length intervals, a method called Grid Partitioning, where the UoD is divided into k intervals of uniform size (Gao; Duru, 2020). Then, k overlapping fuzzy sets are created on each of these intervals. This method is easy to implement and computationally efficient, being a popular choice for early FTS models. Cheng et al. (2008) later introduced equal-frequency partitioning, where each interval contains approximately the same number of data points, ensuring a more balanced distribution of data across partitions.

Partitioning the UoD in uneven intervals has also been proposed in many works in the literature. Clustering algorithms, such as Fuzzy c-Means (FCM) (Song et al., 2019; Askari; Montazerin, 2015; Sun et al., 2015) and K-Means (Cheng; Chen; Jian, 2016; Mukminin et al., 2021), determine optimal partitioning based on the intrinsic patterns of the dataset (Huarng; Yu, 2005). These techniques allow for a more adaptive partitioning strategy, where intervals are shaped according to the density and distribution of data points. This approach has demonstrated improved accuracy in applications where data contains multiple modes or irregular distributions. However, clustering-based partitioning methods require parameter tuning, such as selecting the number of clusters, which can introduce additional computational complexity.

Entropy (Wang; Li; Lu, 2018b; Chen; Chen, 2015) has also been an option for UoD partitioning, which leverages information theory to identify partitions that maximize information gain and minimize uncertainty. By selecting intervals based on entropy measures, this method enhances the model's ability to capture significant changes in the time series while avoiding redundant or irrelevant partitions.

Metaheuristic optimization techniques (Izakian; Pedrycz; Jamal, 2015; Ewbank et al., 2020) have also been been successfully implemented for partitioning in FTS models, with genetic algorithm (GA)-based (Ye et al., 2016; Aladag et al., 2014) and particle swarm optimization (PSO)-based methods (Kumar; Susan, 2021; Askari; Montazerin; Zarandi, 2020; Chen; Phuong, 2017) gaining popularity. Genetic algorithms optimize the number and width of partitions by evolving candidate solutions over multiple generations, leading to an optimal partitioning strategy (Cai et al., 2013). Similarly, PSO-based partitioning optimizes interval boundaries using swarm intelligence, enabling dynamic adaptation to complex time series patterns. Markov weighted fuzzy time-series model based on the optimum partition method was proposed by Alyousifi et al. (2020) to partition the UoD. In addition, an adaptive model is introduced to calculate the final forecasted result in the defuzzification phase. In short, model order and the UoD partitioning method are two major aspects influencing the forecasting accuracy. By analysing existing methodologies, it can be concluded that an accurate UoD partitioning can yield better forecasting accuracy.

Figure 6 shows how different partitioning methodologies split the UoD for the same data.



(a)

(b)

(c)

Figure 6 – Partitioning results for (a) Grid, (b) c-Means and (c) Entropy partitioning methods.

2.3.3 The membership function (μ)

In the fuzzification process the numerical time series data is transformed into fuzzy values. This transformation relies on membership functions (μ), which define the degree to which a numerical value belongs to a particular fuzzy set. They are mathematical representations that map crisp input values to a fuzzy scale, typically within the range [0,1]. A value closer to 1 indicates a higher degree of membership to a fuzzy set, while a value closer to 0 suggests minimal or no membership. The selection of a membership function affects the way data is fuzzified into fuzzy sets and, consequently, influences both the fuzzification and defuzzification processes in a FTS framework. Thus, the choice of an appropriate membership function may influence in accuracy, interpretability, and effectiveness of the FTS model.

There are several types of membership functions, but the most commonly used membership functions in FTS model design include (Zadeh, 1996):

- **Triangular membership functions:** Simple, computationally efficient, and widely used due to their linear structure. Defined by Equation (2.8), where a and c are the lower and upper bounds, and b is the peak value.

$$\mu(x) = \begin{cases} 0, & x \leq a \text{ or } x \geq c \\ \frac{x-a}{b-a}, & a \leq x \leq b \\ \frac{c-x}{c-b}, & b \leq x \leq c \end{cases} \tag{2.8}$$

- **Trapezoidal membership functions:** A generalization of the triangular function that allows for a plateau region, providing more flexibility. It is similar to the triangular membership function but with an additional flat-top region, defined by four parameters (a, b, c, d), as given by Equation (2.9).

$$\mu(x) = \begin{cases} 0, & x \leq a \text{ or } x \geq d \\ \frac{x-a}{b-a}, & a \leq x \leq b \\ 1, & b \leq x \leq c \\ \frac{d-x}{d-c}, & c \leq x \leq d \end{cases} \tag{2.9}$$

- **Gaussian membership functions:** Smooth and continuous, often preferred when handling real-world time series data with gradual transitions and controlled by two parameters: the mean m and the standard deviation δ, as shown in Equation (2.10).

$$\mu(x) = \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right) \tag{2.10}$$

Figure 7 shows an example of UoD partitioning, where eight evenly distributed triangular, trapezoidal and gaussian type-2 fuzzy sets are created. In the next section, the structure and definition of type-2 fuzzy sets will be explored.

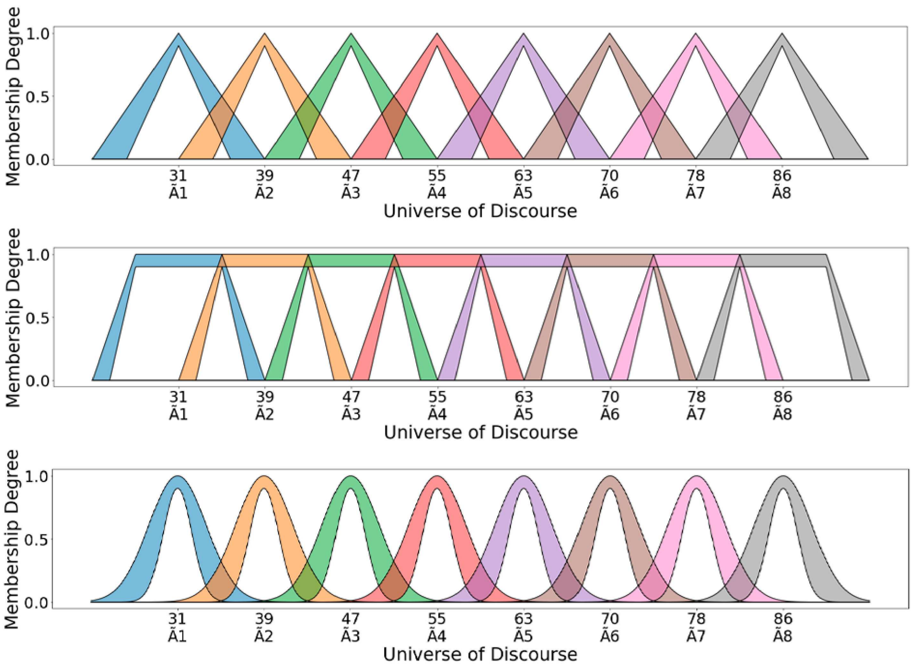


Figure 7 – Example of UoD partitioning using different membership functions.

2.4 Data-driven Partitioning Approaches

The exposed scenario showed that the appropriate number of fuzzy sets for FTS prediction models has been investigated by many researchers (Gao; Duru, 2020; Hua et al., 2019; Silva et al., 2020b) and is an important parameter that needs to be defined as it directly affects the accuracy of the model. Data-driven methods are preferred over the ones that require user interference, as those may introduce subjective bias in model design. In this work, two data-driven partitioning algorithms are explored for UoD partitioning: the Self-Organised Direction Aware algorithm - SODA (Gu et al., 2018) and the Autonomous Data Partitioning - ADP (Gu; Angelov; Príncipe, 2018), to be defined in the following sections.

2.4.1 Self-organised Direction Aware Data Partitioning Algorithm - SODA

Recent research has focused on data-driven models in an effort to make machine learning more autonomous and less reliant on human intervention. SODA is a data partitioning algorithm that relies on Empirical Data Analytics (EDA) operators to identify input data distribution peaks/modes, then uses them as focal points to associate with other points, forming data clouds that resemble Voronoi tessellation. Data clouds can be understood as a special class of clusters but with a few key differences. They are non-parametric and their shape is not predefined by the type of distance metric used. Data clouds directly represent the properties of the observed input data samples.

SODA approach employs:

- A distance component based on a distance metric: The Minkowski, Euclidean (which is a special case of the Minkowsky distance) and Mahalanobis distances, traditional distance metrics broadly used in the literature that measure the magnitude difference between vectors.
- A directional/angular component based on the cosine similarity: The cosine similarity focuses on the directional similarity.

Thus, SODA is able to take advantage of the information extracted within a metric space and within a pseudo-metric, similarity oriented one, namely, the spatial and directional divergences, resulting in a deeper understanding of the properties of the dataset.

Assuming the real data space $\boldsymbol{R}^m$ and considering a dataset $\{\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, ...\}$, where $\boldsymbol{x_i} = [x_{i,1}, x_{i,2}, ..., x_{i,m}]^T$ is a m dimensional vector, $i = 1, 2, 3, ...$; indicate the time instances and m is the dimensionality. Then, the magnitude component d_M can be expressed as shown in Equation (2.11), considering the Euclidean metric:

$$d_M(\boldsymbol{x_i}, \boldsymbol{x_j}) = ||\boldsymbol{x_i} - \boldsymbol{x_j}|| = \sqrt{\sum_{l=1}^{m}(x_{i,l} - x_{j,l})^2}; \tag{2.11}$$

$$i = 1, 2, ..., n$$

The angular component d_A, based on the cosine similarity, can be expressed as (2.12):

$$d_A(\boldsymbol{x_i}, \boldsymbol{x_j}) = \sqrt{1 - \cos(\theta_{\boldsymbol{x_i},\boldsymbol{x_j}}}); i, j = 1, 2, ..., n \tag{2.12}$$

where $\cos(\theta_{\boldsymbol{x_i},\boldsymbol{x_j}}) = \frac{\langle \boldsymbol{x_i}, \boldsymbol{x_j} \rangle}{||\boldsymbol{x_i}||\,||\boldsymbol{x_j}||}$, $\theta_{\boldsymbol{x_i},\boldsymbol{x_j}}$ is the angle between $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$. In the Euclidean space, $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle = \sum_{l=1}^{m} x_{i,l} x_{j,l}$ and $\|\boldsymbol{x}_i\| = \sqrt{\langle \boldsymbol{x}_i, \boldsymbol{x}_i \rangle}$, so the directional component $d_A(\boldsymbol{x}_i, \boldsymbol{x}_j)$ can be rewritten as follows:

$$d_A(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{\frac{1}{2}\sum_{l=1}^{m}\left(\frac{x_{i,l}}{\|\boldsymbol{x}_i\|} - \frac{x_{j,l}}{\|\boldsymbol{x}_j\|}\right)^2} = \frac{1}{\sqrt{2}}\left\|\frac{\boldsymbol{x}_i}{\|\boldsymbol{x}_i\|} - \frac{\boldsymbol{x}_j}{\|\boldsymbol{x}_j\|}\right\|, \quad i, j = 1, 2, \ldots, n \tag{2.13}$$

Empirical Data Analytics (EDA) is a non-parametric and assumption-free technique that consists of clustering algorithms focused on data. It is conducted based on the empirical observation of the data alone, with no previous knowledge or definitions required. EDA resembles statistical learning in its nature but is free from the range of assumptions made in traditional probability theory and statistical learning approaches. One of the main EDA operators is the Global Density, which is defined for unique data samples together with their respective number of repeats $\{f_1, f_2, ..., f_{n_u}\}$ in the dataset. The Global Density is expressed using two other EDA operators: Cumulative Proximity and Local Density. The Cumulative Proximity π of $\boldsymbol{x_i}$ $(i = 1, 2, ..., n)$ is initially defined by Equation (2.14), where $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ are different instances from the time series and $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ denotes the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ (Gu et al., 2018).

$$\pi_n^M(\boldsymbol{x_i}) = \sum_{j=1}^{n} d_M^2(\boldsymbol{x_i}, \boldsymbol{x_j}) \tag{2.14}$$

With the Euclidean component d_M, the Cumulative Proximity can be calculated recursively as:

$$\pi_n^M(\boldsymbol{x}_i) = \sum_{j=1}^{n} d_M^2(\boldsymbol{x}_i, \boldsymbol{x}_j) = n\left(\left\|\boldsymbol{x}_i - \boldsymbol{\mu}_n^M\right\|^2 + X_n^M - \left\|\boldsymbol{\mu}_n^M\right\|^2\right) \tag{2.15}$$

where $\boldsymbol{\mu}_n^M$ is the mean of $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ and X_n^M is the mean of $\{\|\boldsymbol{x}_1\|^2, \|\boldsymbol{x}_2\|^2, \ldots, \|\boldsymbol{x}_n\|^2\}$; they can be updated recursively as:

$$\boldsymbol{\mu}_n^M = \frac{n-1}{n}\boldsymbol{\mu}_{n-1}^M + \frac{1}{n}\boldsymbol{x}_n; \quad \boldsymbol{\mu}_1^M = \boldsymbol{x}_1 \tag{2.16}$$

$$X_n^M = \frac{n-1}{n}X_{n-1}^M + \frac{1}{n}\|\boldsymbol{x}_n\|^2; \quad X_1^M = \|\boldsymbol{x}_1\|^2 \tag{2.17}$$

Using the angular component, the Cumulative Proximity can be rewritten as:

$$\pi_n^A(\boldsymbol{x}_i) = \sum_{j=1}^{n} d_A^2(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{n}{2}\left(\left\|\frac{\boldsymbol{x}_i}{\|\boldsymbol{x}_i\|} - \boldsymbol{\mu}_n^A\right\|^2 + X_n^A - \left\|\boldsymbol{\mu}_n^A\right\|^2\right) \tag{2.18}$$

where $\boldsymbol{\mu}_n^M$ is the mean of $\left\{\frac{\boldsymbol{x}_1}{\|\boldsymbol{x}_1\|}, \frac{\boldsymbol{x}_2}{\|\boldsymbol{x}_2\|}, \ldots, \frac{\boldsymbol{x}_n}{\|\boldsymbol{x}_n\|}\right\}$ and $X_i^A = 1$ for $i = 1, 2, \ldots, n$, and similarly:

$$\boldsymbol{\mu}_n^A = \frac{n-1}{n}\boldsymbol{\mu}_{n-1}^A + \frac{1}{n}\frac{\boldsymbol{x}_n}{\|\boldsymbol{x}_n\|}; \quad \boldsymbol{\mu}_1^A = \frac{\boldsymbol{x}_1}{\|\boldsymbol{x}_1\|} \tag{2.19}$$

The Local Density D of $\boldsymbol{x_i}$ $(i = 1, 2, ..., n;\ n_u > 1)$ (Equation (2.20)), is defined as the inverse of the normalized cumulative proximity and directly indicates the main pattern of the observed data.

$$D_n(\boldsymbol{x_i}) = \frac{\sum_{j=1}^{n} \pi_n(\boldsymbol{x_j})}{2n\pi_n(\boldsymbol{x_i})} \tag{2.20}$$

For the case of Euclidean distance, the Local Density can be reduced to the following expression:

$$D_n^M(\boldsymbol{x}_i) = \frac{1}{1 + \frac{\|\boldsymbol{x}_i - \boldsymbol{\mu}_n^M\|^2}{X_n^M - \|\boldsymbol{\mu}_n^M\|^2}} \tag{2.21}$$

And for the angular component, the Local Density has a similar form:

$$D_n^A(\boldsymbol{x}_i) = \frac{1}{1 + \frac{\|\boldsymbol{x}_i - \boldsymbol{\mu}_n^A\|^2}{1 - \|\boldsymbol{\mu}_n^A\|^2}} \tag{2.22}$$

Since both magnitude and angular components are equally important, the Local Density of $\boldsymbol{x}_i$ $(i = 1, 2, \ldots, n;\ n_u > 1)$ is defined as the sum of the metric/Euclidean-based Local Density $(D_n^M(\boldsymbol{x}_i))$ and the angular-based Local Density $(D_n^A(\boldsymbol{x}_i))$:

$$D_n(\boldsymbol{x}_i) = D_n^M(\boldsymbol{x}_i) + D_n^A(\boldsymbol{x}_i) = \frac{1}{1 + \frac{\|\boldsymbol{x}_i - \boldsymbol{\mu}_n^M\|^2}{X_n^M - \|\boldsymbol{\mu}_n^M\|^2}} + \frac{1}{1 + \frac{\|\boldsymbol{x}_i - \boldsymbol{\mu}_n^A\|^2}{1 - \|\boldsymbol{\mu}_n^A\|^2}} \tag{2.23}$$

After that, the Global Density of a particular unique data sample, $\boldsymbol{u}_i$ $(i = 1, 2, \ldots, n_u;\ n_u > 1)$ can be expressed as the product of its Local Density and its number of repeats considered as a weighting factor, as denoted in Equation (2.24):

$$D_n^G(\boldsymbol{u_i}) = f_i D_n(\boldsymbol{u_i}) \tag{2.24}$$

These characteristics also make SODA suitable for streaming data, being capable of continuously processing data streams based on the offline processing of an initial dataset. According to Gu et al. (2018), it has to be stressed that the concept of “non-parametric” means SODA algorithms is free from user or problem specific parameters and presumed models imposed for the data generation, but this does not mean that SODA algorithms do not have hyperparameters to achieve data processing.

The performance of SODA partitioning algorithm was assessed by Gu et al. (2018) in a number of benchmark clustering problems and then compared to "state-of-art"clustering algorithms such as the DBScan, Density Peak and Random Swap clustering algorithms. They showed that SODA is able to perform clustering with very high computation efficiency and produces high quality clustering results in various benchmark problems. SODA results are generated based entirely on the ensemble properties of the input data.

2.4.2 Autonomous Data Partitioning Algorithm - ADP

The second method used in this work as a tool for finding the optimal number of partitions for the time series is the Autonomous Data Partitioning algorithm - ADP (Gu; Angelov; Príncipe, 2018), which is a fully autonomous local-modes-based data partitioning algorithm. Local modes are akin of the peaks of the local data density. The method is free from the user and problem-specific parameters and prior assumptions. It also utilizes parameter-free operators to disclose the underlying data distribution and ensemble properties of the empirically observed data using the natural distance metric. According to these operators, the local modes representing the local maxima of the data density can be automatically determined. The proposed approach uses these local modes to partition the data space in shape-free data clouds (Angelov; Yager, 2012) (a kind of Voronoi tessellation). As seen, data clouds can be understood as a particular type of cluster but with several distinctive differences. They are non-parametric, and the type of distance metric they use has no bearing on their shape.

Data clouds directly represent the properties of the local set of observed input data samples. The strong performance of the ADP algorithm comes from a fundamentally different data processing approach based on rank operators. Rank operators are usually avoided in clustering because of their non-linearity, and so most clustering algorithms use the linear mean operator (Gu; Angelov; Príncipe, 2018). The ADP algorithm can identify prototypes from the data samples based on their ranks in terms of the data densities and mutual distances instead of the commonly used means and variances. These prototypes are then used to aggregate data samples around them, forming Voronoi tessellations.

Numerical experiments performed by Gu, Angelov and Príncipe (2018) based on benchmark clustering problems were used to evaluate the performance of the ADP clustering algorithm, and the results were compared to well-known clustering algorithms such as the Self-organizing map (Kohonen, 1998), Non-parametric mixture model-based clustering (Blei; Jordan, 2006) and DBSCAN (Ester et al., 1996). They showed that ADP could achieve excellent clustering performance very fast without prior knowledge about the dataset. Moreover, in Angelov and Gu (2019) the authors further compared ADP in four different datasets using advanced classification approaches: i) Support vector machine (SVM) classifier with Gaussian kernel; ii) k-nearest neighbor (KNN) classifier with $k = 10$; ii) Neural network (NN) with three hidden layers of size 20; iv) Decision tree classifier. The results indicated that the ADP algorithm is able to provide highly accurate classification metrics with much less supervision compared with other approaches. This is of paramount importance to real applications since the labels of training data are often very expensive to obtain. ADP has been used by researchers in a number of recent papers, as in Souza, Ponce and Lughofer (2020), Gu and Angelov (2020), Huang et al. (2022), Souza and Lughofer (2022) and Svetlakov and Hodashinsky (2021).

Like SODA, the ADP algorithm is based on the Empirical Data Analytics (EDA) method (Angelov, 2014; Angelov; Gu; Kangin, 2017). EDA measures play an instrumental role in the ADP approach for extracting the ensemble properties from the observed data and frees the ADP algorithm from the requirement of making prior assumptions on the data generation model parameters. Most importantly, they ensure the objectiveness of the partitioning results. The main EDA operators for ADP are defined by Gu, Angelov and Príncipe (2018) as the Local Density ALD and Global Density AD^G.

Local Density is a measure within the EDA framework identifying the main local mode of the data distribution and is derived empirically from all the observed data samples in a parameter-free way. The Local Density, ALD of the data sample $\boldsymbol{x}_k$ is expressed as follows ($k = 1, 2, \ldots, K; L_K > 1$):

$$ALD_K(\boldsymbol{x}_k) = \frac{\sum_{j=1}^{K}\sum_{l=1}^{K} d^2(\boldsymbol{x}_j, \boldsymbol{x}_l)}{2K\sum_{l=1}^{K} d^2(\boldsymbol{x}_k, \boldsymbol{x}_l)} \tag{2.25}$$

where $d(\boldsymbol{x}_k, \boldsymbol{x}_l)$ is the distance between data samples $\boldsymbol{x}_k$ and $\boldsymbol{x}_l$; the coefficient 2 is used in the denominator for normalization (because each distance is counted twice in the numerator).

For the case of Euclidean distance, the calculation of $\sum_{l=1}^{K} d^2(\boldsymbol{x}_k, \boldsymbol{x}_l) = \sum_{l=1}^{K} \|\boldsymbol{x}_k - \boldsymbol{x}_l\|^2$ and $\sum_{j=1}^{K}\sum_{l=1}^{K} d^2(\boldsymbol{x}_j, \boldsymbol{x}_l) = \sum_{j=1}^{K}\sum_{l=1}^{K} \|\boldsymbol{x}_j - \boldsymbol{x}_l\|^2$ can be simplified by using the mean of $\{\boldsymbol{x}\}_K$, $\boldsymbol{\mu}_K$ and the average scalar product, X_K, as:

$$\sum_{l=1}^{K} \|\boldsymbol{x}_k - \boldsymbol{x}_l\|^2 = K\left(\|\boldsymbol{x}_k - \boldsymbol{\mu}_K\|^2 + X_K - \|\boldsymbol{\mu}_K\|^2\right) \tag{2.26}$$

$$\sum_{j=1}^{K}\sum_{l=1}^{K} \|\boldsymbol{x}_j - \boldsymbol{x}_l\|^2 = 2K^2\left(X_K - \|\boldsymbol{\mu}_K\|^2\right) \tag{2.27}$$

$\boldsymbol{\mu}_K$ and X_K can be updated recursively as:

$$\boldsymbol{\mu}_k = \frac{k-1}{k}\boldsymbol{\mu}_{k-1} + \frac{1}{k}\boldsymbol{x}_k; \quad \boldsymbol{\mu}_1 = \boldsymbol{x}_1; \quad k = 1, 2, \ldots, K \tag{2.28}$$

$$X_k = \frac{k-1}{k}X_{k-1} + \frac{1}{k}\|\boldsymbol{x}_k\|^2; \quad X_1 = \|\boldsymbol{x}_1\|^2; \quad k = 1, 2, \ldots, K \tag{2.29}$$

The recursive calculation of $\boldsymbol{\mu}_K$ and X_K allows for “one pass” calculation, thus ensuring computational efficiency because only the key aggregated/summarized information has to be kept in memory. It is possible then to re-formulate the Local Density - ALD in a recursive form:

$$ALD_K(\boldsymbol{x}_k) = \frac{1}{1 + \frac{\|\boldsymbol{x}_k - \boldsymbol{\mu}_K\|^2}{X_K - \|\boldsymbol{\mu}_K\|^2}} \tag{2.30}$$

The global density, AD^G estimated at the unique data sample $\boldsymbol{u}_k$ is a weighted sum of its local density by the corresponding occurrence, f_k $(k = 1, 2, \ldots, L_K; L_K > 1)$, expressed as follows:

$$AD^G_K(\boldsymbol{u}_k) = \frac{f_k}{\sum_{j=1}^{L_K} f_j} D_K(\boldsymbol{u}_k) = \frac{f_k}{1 + \frac{\|\boldsymbol{u}_k - \boldsymbol{\mu}_K\|^2}{X_K - \|\boldsymbol{\mu}_K\|^2}} \tag{2.31}$$

3 The SODA-T2FTS Approach

As shown in Chapter 2, conventional FTS models typically rely on static or heuristic partitioning strategies, which may not align well with the underlying structure of the data and often require user intervention. These limitations may introduce epistemic uncertainty and hinder generalization across datasets. To address this, in this chapter SODA-T2FTS is introduced, a data-driven interval type-2 FTS model that leverages the Self-Organized Direction Aware (SODA) algorithm to automatically partition the UoD. This method seeks to reduce subjectivity in the model design, improve the representation of uncertainty, and enhance forecasting accuracy when compared to state-of-the-art partitioning-based FTS approaches.

SODA-T2FTS receives as input the time series data and then all the steps necessary for training and testing the model are performed in order to obtain error metrics and forecasting results. The summarized process is shown in Figure 8. For illustration purposes of the following procedures, this chapter will be analysed using the Taiwan Stock Exchange Capitalization Weighted Stock Index - TAIEX time series, which is composed of 5200 samples and will be later discussed along with other financial time series.

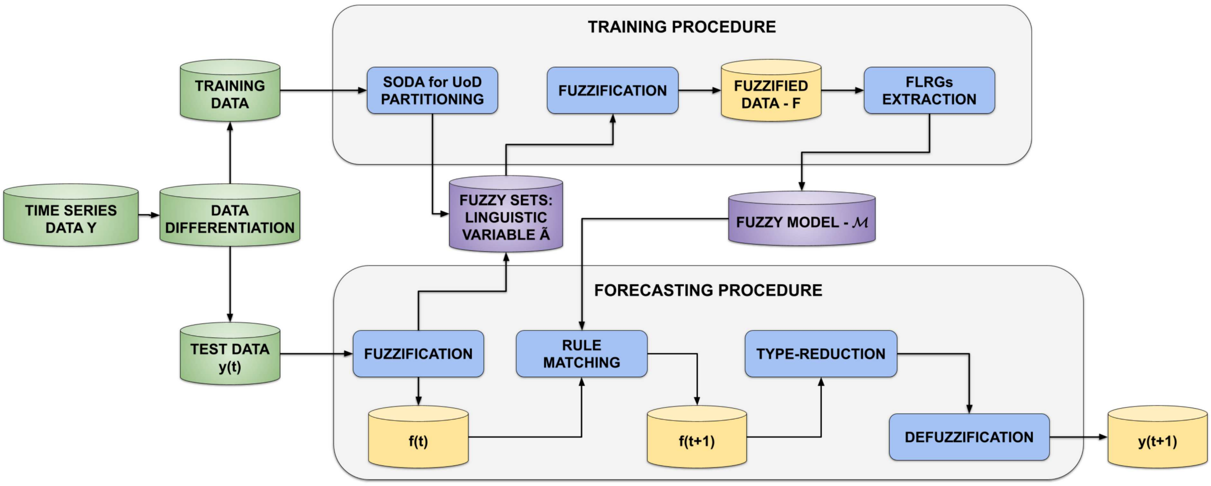


Figure 8 – SODA-T2FTS training and forecasting procedure.

3.1 Training Procedure

The training procedure aims to learn the temporal patterns of the time series training data thus building the model that will be used in the forecasting phase. The steps for the training procedure are listed in the following subsections.

3.1.1 Definition of the UoD

Assuming that the time series is represented by $Y(t)$, the UoD is defined as $UoD = [L_{bound}, U_{bound}]$, as demonstrated in Section 2.2. The lower bound is calculated as $L_{bound} = 0.9\ min(Y(t))$ if $min(Y(t))$ is larger than zero and $L_{bound} = 1.1\ min(Y(t))$ if it is not. The upper bound is calculated as $U_{bound} = 1.1\ max(Y(t))$ if $max(Y(t))$ is larger than zero and $U_{bound} = 0.9\ max(Y(t))$ if it is not.

3.1.2 SODA Algorithm for Data Partitioning

The main steps of the SODA algorithm include: firstly, form a number of Direction-Aware (DA) planes from the observed data samples using both, the magnitude-based and angular-based densities; secondly, identify focal points; finally, use the focal points to partition the data space into data clouds and then find the optimal number of partitions for the input data. The detailed procedure of the proposed SODA partitioning algorithm is as follows (Gu et al., 2018):

- **Stage 1: Preparation**. The process begins with the calculation of the square Euclidean components, d_M, and square angular components, d_A, between every pair of data samples, $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ using Equations (2.11) and (2.12):

$$\bar{d}_M^2 = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} d_M^2(\boldsymbol{x}_i, \boldsymbol{x}_j)}{n^2} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{n^2} = 2\left(X_n^M - \|\boldsymbol{\mu}_n^M\|^2\right) \quad (3.1)$$

$$\bar{d}_A^2 = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} d_A^2(\boldsymbol{x}_i, \boldsymbol{x}_j)}{n^2} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} \left\|\frac{\boldsymbol{x}_i}{\|\boldsymbol{x}_i\|} - \frac{\boldsymbol{x}_j}{\|\boldsymbol{x}_j\|}\right\|^2}{2n^2} = 1 - \|\boldsymbol{\mu}_n^A\|^2 \quad (3.2)$$

  Using both d_M and d_A the data samples can be visualized in 2D planes, called DA planes. The *x-axis* of the DA plane represents the magnitude component and the *y-axis* the angular component. Figure 9 shows an illustrative example where the data sample x_1 is selected as the origin and the other data samples x_2, x_3 and x_4 are projected based on their magnitude and angular components.

  Then, it is possible to obtain the global density $D_n^G(\boldsymbol{u_i})(i = 1, 2, ..., n_u)$ of the unique data samples $\{\boldsymbol{u_1}, \boldsymbol{u_2}, ..., \boldsymbol{u}_{n_u}\}$ using Equation (2.24). After the global densities of all the unique data samples are calculated, they are ranked in a descending order and renamed as $\{\hat{\boldsymbol{u}}_1, \hat{\boldsymbol{u}}_2, ..., \hat{\boldsymbol{u}}_{n_u}\}$.

- **Stage 2: DA Plane Projection**. The DA projection operation begins with the $\hat{\boldsymbol{u}}_1$, which is the unique data sample that has the highest global density. It is initially set to be the first reference, $\boldsymbol{\mu}_1 \leftarrow \hat{\boldsymbol{u}}_1$, which is also the origin point of the first DA plane, denoted by $\boldsymbol{P}_1$. Figure 10 illustrates an example of independent DA planes

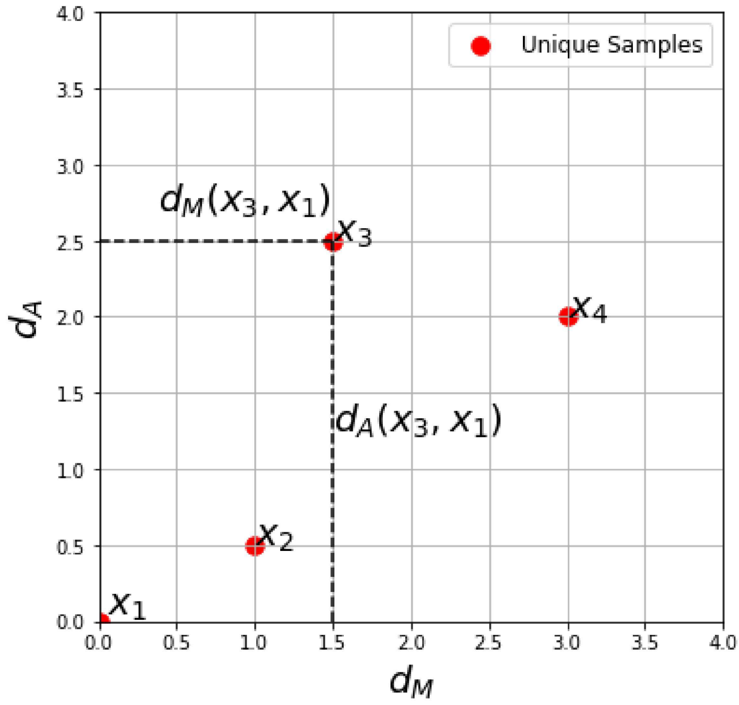


Figure 9 – Illustrative example of a DA plane (Adapted from Gu et al. (2018)).

dividing the whole data space, where the black dots stand for each data sample and the coloured squares represent the DA planes.

Some data samples are located in several DA planes at the same time, and their affiliations are decided by the distances between them and the origin points of the nearby DA planes. In this stage there is a hyperparameter called gridsize which is set to decide the granularity of the clustering results and relates to the Chebyshev inequality (Angelov; Gu; Príncipe, 2018). After all the unique data samples are projected onto the DA planes, the next stage can start.

- **Stage 3: Identifying the Focal Points**. In this stage, find the neighbouring DA planes denoted by $\{\boldsymbol{P}\}_e^n(l = 1, 2, ..., L, l \neq e)$ for each DA plane $\boldsymbol{P}_e$ by evaluating the condition in Equation (3.3), which takes into account the magnitude and angular components and the gridsize value as part of the distance threshold. By examination of all existing DA planes, it is possible to find all the modes/peaks of the data density.

$$\begin{gathered}\boldsymbol{Condition\ 1:}\\ IF\left(\frac{d_M(\boldsymbol{\mu}_e, \boldsymbol{\mu}_l)}{\bar{d}_M} \leq \frac{2}{\gamma}\right)\ AND\ \left(\frac{d_A(\boldsymbol{\mu}_e, \boldsymbol{\mu}_l)}{\bar{d}_A} \leq \frac{2}{\gamma}\right)\\ THEN\left(\{\boldsymbol{P}\}_e^n \leftarrow \{\boldsymbol{P}\}_e^n \cup \boldsymbol{P}_l\right)\end{gathered} \tag{3.3}$$

Where γ corresponds to the gridsize and $\boldsymbol{P}_l$ $(l = 1, 2, ..., L)$ represents all the possible

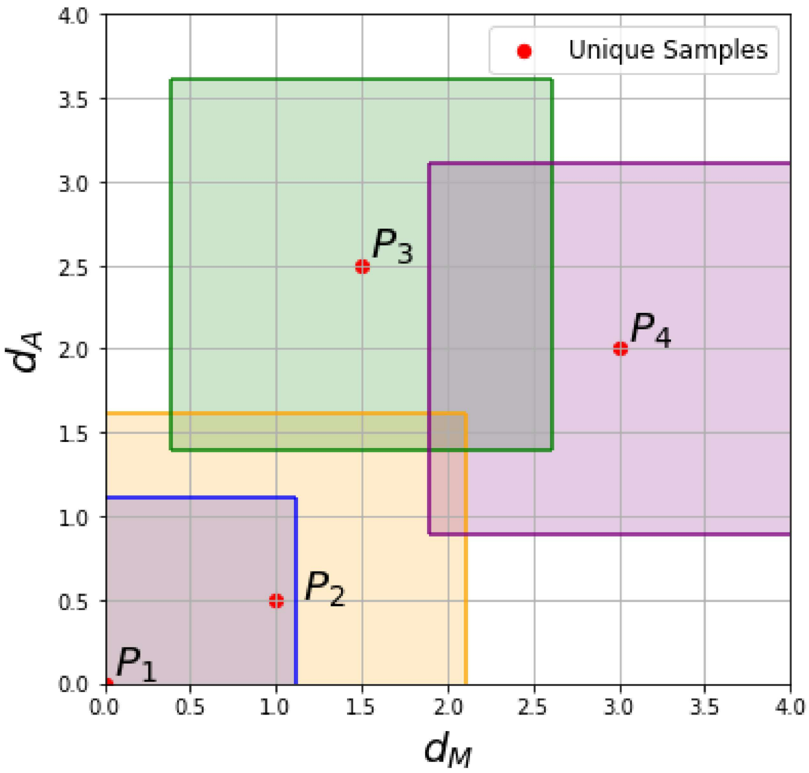


Figure 10 – Example of DA planes (Adapted from Gu et al. (2018)).

neighbouring DA planes, with L being the total number of existing DA planes in the data space.

- **Stage 4: Forming Data Clouds**. After all the DA planes standing for the modes/peaks of the data density are identified, consider their origin points, denoted by $\boldsymbol{\mu}^o$, as the focal points and use them to form data clouds as a Voronoi tessellation (Okabe et al., 1993). It is worth to mention that the concept of data clouds is quite similar to the concept of clusters, but differs in the following aspects: i) data clouds are non-parametric; ii) data clouds do not have a specific shape; iii) data clouds represent the real data distribution.

SODA-T2FTS uses the number of data clouds that SODA generated from the time series to define the number of partitions for the UoD. The final number of data clouds is directly related to the Gridsize hyperparameter γ as presented in Equation (3.3). Thus it should be optimally chosen for each dataset to achieve better forecasting results.

Let C be the number of data clouds formed by SODA from the time series. The number of partitions is used to define the linguistic variable $\tilde{A}$. For a C number of partitions, create C overlapping fuzzy sets $\tilde{A}_i$, for $i = 1, ..., C$. Each fuzzy set $\tilde{A}_i$ is a linguistic term of the linguistic variable $\tilde{A}$. Figure 11 shows examples of UoD partitioning using SODA outputs generated from different Gridsizes. Considering that SODA output for a certain input data is the number seventeen, then the UoD is partitioned into seventeen equal-length intervals and seventeen overlapping interval type-2 triangular fuzzy sets are created from these intervals: $\tilde{A}_1, \tilde{A}_2, \tilde{A}_3, ..., \tilde{A}_{17}$.

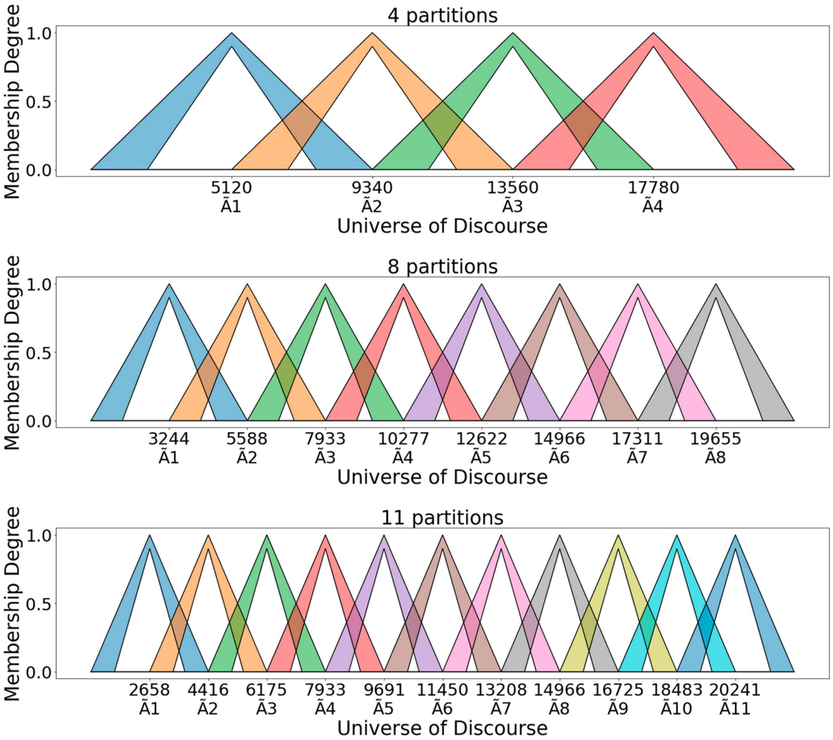


Figure 11 – Examples of UoD partitioning using SODA output.

3.1.3 Fuzzification and Identification of Fuzzy Logical Relationship Groups

Each sample from the $Y(t)$ time series is inserted into the model and the membership values are calculated for all the fuzzy sets in the fuzzification procedure. Thus, for each input, an interval-valued degree of membership is obtained: $[\underline{\mu}_{\widetilde{A}}(x), \bar{\mu}_{\widetilde{A}}(x)]$, crisp values that correspond to the degrees of membership for both lower and upper membership functions for each fuzzy set, respectively. In Table 1 are listed some TAIEX samples and their respective fuzzy sets.

As a time series is a sequence of samples, the relationship between one sample and the next can be analysed and identified as a fuzzy rule, which also represents the temporal patterns between sequential samples. For example: if a random sample $y(t)$ from $Y(t)$ is fuzzified as $\tilde{A}_\mathrm{i}$ and $y(t+1)$ is fuzzified as $\tilde{A}_\mathrm{j}$, then a fuzzy rule can be created: $\tilde{A}_\mathrm{i} \rightarrow \tilde{A}_\mathrm{j}$, where $\tilde{A}_\mathrm{i}$ is called the antecedent and $\tilde{A}_\mathrm{j}$ is the called the consequent. This fuzzy rule can also be read as IF $y(t)$ is $\tilde{A}_\mathrm{i}$ THEN $y(t+1)$ is $\tilde{A}_\mathrm{j}$. The rule extraction can be done for every sample in the fuzzified time series. Referring to the samples from 708 to 711 in Table 1 for example, Table 2 lists the fuzzified TAIEX instances and the respective fuzzy rules generated from them.

Table 1 – TAIEX samples and correspondent fuzzy sets.

N	Date	TAIEX	Fuzzy Sets
...	...	...	...
703	06/16/1997	8621.02	$\tilde{A}_8$,$\tilde{A}_9$
704	06/17/1997	8715.85	$\tilde{A}_8$,$\tilde{A}_9$
705	06/18/1997	8705.69	$\tilde{A}_8$,$\tilde{A}_9$
706	06/19/1997	8730.22	$\tilde{A}_8$,$\tilde{A}_9$
707	06/20/1997	8820.11	$\tilde{A}_8$,$\tilde{A}_9$
708	06/21/1997	8913.20	$\tilde{A}_8$,$\tilde{A}_9$
709	06/23/1997	8894.43	$\tilde{A}_8$,$\tilde{A}_9$
710	06/24/1997	8955.63	$\tilde{A}_9, \tilde{A}_{10}$
711	06/25/1997	8950.07	$\tilde{A}_9, \tilde{A}_{10}$
712	06/26/1997	8943.88	$\tilde{A}_9, \tilde{A}_{10}$
713	06/27/1997	8942.24	$\tilde{A}_9, \tilde{A}_{10}$
714	06/28/1997	9043.55	$\tilde{A}_9, \tilde{A}_{10}$
715	06/30/1997	9075.59	$\tilde{A}_9, \tilde{A}_{10}$
716	07/02/1997	9056.22	$\tilde{A}_9, \tilde{A}_{10}$
717	07/03/1997	9029.56	$\tilde{A}_9, \tilde{A}_{10}$
718	07/04/1997	9168.73	$\tilde{A}_9, \tilde{A}_{10}$
...	...	...	...

Table 2 – Fuzzification and rules generated for the TAIEX time series.

Date	TAIEX	Fuzzy sets	Rules
06/21/1997	8913.2	$\tilde{A}_8, \tilde{A}_9$	-
06/23/1997	8894.43	$\tilde{A}_8, \tilde{A}_9$	$\tilde{A}_8 \to \tilde{A}_8, \tilde{A}_8 \to \tilde{A}_9$ $\tilde{A}_9 \to \tilde{A}_8, \tilde{A}_9 \to \tilde{A}_9$
06/24/1997	8955.63	$\tilde{A}_9, \tilde{A}_{10}$	$\tilde{A}_8 \to \tilde{A}_9, \tilde{A}_8 \to \tilde{A}_{10}$ $\tilde{A}_9 \to \tilde{A}_9, \tilde{A}_9 \to \tilde{A}_{10}$
06/25/1997	8950.07	$\tilde{A}_9, \tilde{A}_{10}$	$\tilde{A}_9 \to \tilde{A}_9, \tilde{A}_9 \to \tilde{A}_{10}$ $\tilde{A}_{10} \to \tilde{A}_9, \tilde{A}_{10} \to \tilde{A}_{10}$

It is important to mention that as the fuzzy sets overlap, each sample from each day may activate several fuzzy sets at the same time. Repeated rules are considered unique for simplicity and to reduce computational costs.

The next step is to group all the rules with the same antecedent, and create a Fuzzy Logic Relationship Group (FLRG) for every fuzzy set. FLRGs can be interpreted as the possible future states for the consequent when a given antecedent is identified. For example, The FLRG $\tilde{A}_\text{i} \to \tilde{A}_\text{j}, \tilde{A}_\text{k}, \tilde{A}_\text{g}$ implies that IF $y(t)$ is $\tilde{A}_\text{i}$ THEN $y(t+1)$ is $\tilde{A}_\text{j}$ *OR* $\tilde{A}_\text{k}$ *OR* $\tilde{A}_\text{g}$.

3.2 Forecasting Procedure

Once the linguistic variable $\tilde{A}$ is defined and the FTS model $\mathcal{M}$ is learned, the test samples can be presented to produce a forecast. The forecasting procedure takes as input a sample from the time series test dataset $y(t)$ and the output is generated using the model learned in the training procedure. The complete forecasting procedure happens to each sample of the time series and is presented below:

- **Step 1: Fuzzification**

  For a given sample from the time series $y(t) \in Y$, calculate the degrees of membership for both lower and upper membership functions for each fuzzy set.

- **Step 2: Rule Matching**

  The rule matching procedure consists in finding the rules that are activated by each of the fuzzified samples and their respective FLRGs and is strongly related to the learned model. The aim in this procedure is to produce $f(t+1)$ to represent the future crisp value $y(t+1)$. In this procedure, there is a combination of rules and a mapping from input to output type-2 fuzzy sets. Antecedent processing is executed using the *min* operator, which is applied to select the minimum values of the UMFs and LMFs of all antecedents for each rule. The result of antecedent processing is applied to consequent LMF and UMF functions. As explained in Section 2.2, for the case of triangular membership functions and when the *min* operator is used, the result is a trapezoidal consequent FOU. Consequent processing is performed using the *max* operator and then after aggregation, a single interval type-2 fuzzy set from the rule output fuzzy sets is generated.

- **Step 3: Type-reduction**

  To complete the forecasting process, a final crisp output value must be obtained for each test value. This is accomplished by the output processor. The aggregated type-2 fuzzy set is first reduced to a type-1 fuzzy set. There exist several type-reduction methods in literature. In this work, the Enhanced Iterative Algorithm with Stop Condition - EIASC (Wu; Nie, 2011) was used to perform type-reduction because of its reduced computational costs in comparison to other methods such as the Karnik-Mendel - KM (Karnik; Mendel, 2001) and the Iterative Algorithm with Stop Condition - IASC (Duran; Bernal; Melgarejo, 2008).

- **Step 4: Defuzzification**

  The defuzzification procedure is the last part of the forecasting process, where the values obtained by the type reduction algorithm are defuzzified and crisp values are generated. Defuzzification is performed using the Centroid method, which returns the center of gravity of the type-1 fuzzy set along the x-axis, and is calculated using Equations (2.5), (2.6) and (2.7), explained in Section 2.2.

3.3 Sliding window method

In this chapter, a sliding window method was applied as a comparison procedure, working in terms of the obtained forecasting error metrics. In this method, a dataset Y of size T is sliced into overlapping sequences of W (called data windows) of fixed size and with each shifted from the previous one by a constant D. Each data window has its training and testing divisions and then the model is trained and tested, as exemplified in Figure 12. When defining the best values for data window size W and window offset D, it is advisable that for a dataset Y of size T, the data window, and the data shift result in at least 10 experiments, well distributed over the total dataset. Most importantly, it is crucial that each window contains enough information about the time series temporal patterns and covers different behaviors such as falling and rising patterns, so that the training procedure has enough information to build the knowledge base. It was defined a window size of $W = 1000$ instances that was shifted by $D = 200$ instances along the time series data. The data for each window was divided into two sets; the training set which contained 80% of the data and the test set, with the remaining 20% of the data. Considering a time series composed of 5200 instances, the sliding window method would create 22 experiments (windows) and for each window, training and testing are performed out of sample and then error metrics are calculated.

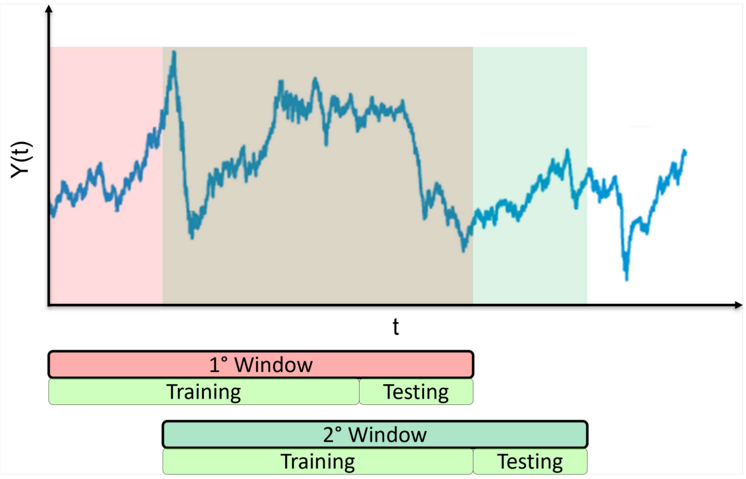


Figure 12 – Sliding window method example.

SODA-T2FTS overall training and forecasting processes are summarized in Algorithm 1, which take as input the time series data and then error metrics are calculated by following the sliding window procedure.

Algorithm 1: SODA-T2FTS training and forecasting method

```
while Not at the end of time series data $Y$ do
    Remove chunk from time series data according to window size $W$;
    Divide time series data window into training and forecasting sets;
    for The training data $Y_{tn}$ do
        Calculate $UoD$;
        Input $Y_{\mathrm{tn}}$ into SODA and get as result the number $C$ of data clouds
          identified;
        Use $C$ as the number of partitions for the $UoD$ to be split;
        To each partition assign an interval type-2 fuzzy set $\tilde{A}_{\mathrm{i}}$ $(i = 1, 2, ...C)$;
        Fuzzify each instance $y(t)$ into $f(t)$;
        Identify temporal patterns (FLRs) between consequent instances from $Y_{\mathrm{tn}}$ ;
        Group FLRs into FLRGs and build the model $\mathcal{M}$;
    end
    for The forecasting data $Y_{fs}$ do
        Fuzzify each instance $y(t)$ into $f(t)$;
        for $FLRG \in \mathcal{M}$ do
            if $f(t)$ matches the LHS then
                activated_rules ← FLRG;
            end
        end
        Use activated_rules in the fuzzy inference procedure;
        Compute $f(t + 1)$;
        Apply type-reduction and deffuzification to obtain $y(t + 1)$;
        Calculate and store RMSE metrics;
    end
    Slide the window along time series data by a number of $D$ instances;
end
```

In the next section the proposed SODA-T2FTS model is evaluated by forecasting financial time series and the results obtained are compared to "state-of-art"FTS forecasting methods. The models are compared in terms of error metrics, execution time, model complexity and noise sensitivity.

3.4 Experimental Results

3.4.1 Case Studies and Experimental Settings

The National Association of Securities Dealers Automated Quotations - NASDAQ, the Standard & Poor's 500 - S&P500 and the Taiwan Stock Exchange Capitalization Weighted Stock Index - TAIEX, three time series usually used in forecasting literature (Tak, 2020; Zhang; Wang; Li, 2016; Tsai; Cheng; Tsai, 2019; Zhao; Gao; Guan, 2020; Lahmiri, 2020) were chosen to assess the proposed model. Financial datasets are important for understanding market behavior, and accurate forecasts can help businesses and individuals anticipate market movements, allocate resources wisely, and respond proactively to economic shifts. The three datasets are shown in Figure 13 and Table 3 presents a preliminary statistical analysis.

The NASDAQ time series is composed of 4000 instances, sampled from 2001 to 2016 and is non-stationary with an ADF test statistic (Dickey; Fuller, 1979) of 0.47. After differentiation, the ADF test statistic was -17.35. The S&P500 is a non-stationary time series with an ADF test of -1.74 and is composed of 4500 instances, sampled from 1995 to 2013. After differentiation, the ADF test statistic was -11.81. The TAIEX dataset is a non stationary time series with an ADF test of -2.70, composed of 5200 instances, sampled from 1995 to 2015. After differentiation, the ADF test statistic was -20.93. All three datasets were trained and tested with original and first order differentiated data, in order to make the time series stationary (or at least closer to stationary). The models trained with original data obtained higher error metrics than the ones trained with differentiated data, therefore they have not been mentioned to avoid redundancy and for better visualization and understanding.

Table 3 – Descriptive statistics for TAIEX, NASDAQ, and S&P500 datasets

Parameter	**TAIEX**	**NASDAQ**	**SP500**
Data points	5200	4000	4500
Mean	7071.34	2685.56	1158.51
Std. deviation	1475.13	1046.29	241.00
Minimum	3473.81	1123.98	576.5
1st quartile	5920.04	1989.8	1000.68
Median	7200.24	2344.14	1181.97
3rd quartile	8233.76	3036.00	1338.19
Maximum	10227.91	5226.75	1706.98

Evaluation of machine learning models usually requires a training set and a test set. In general, 80% of the data is randomly chosen to compose the training set and the remaining 20% are used for testing. But when evaluating a time series forecasting model, one must take into account that there is a relationship between the observations, which means that they are time-dependent so cannot be randomly divided into groups. Thus, data must be split respecting the temporal order in which values were observed to provide statistically robust model evaluation and best simulate real-life scenarios.

All experiments have been carried out in Python 3.7 in an Intel i5-8265U 1.6 GHz with 8 GB DDR4. All the data and source codes are available on: https://github.com/arthurcaio92/SODA-T2FTS.

3.4.2 SODA partitioning

The *UoD* is partitioned using the SODA algorithm. SODA output defines the optimal number of fuzzy sets to be created for partitioning. As mentioned earlier, there is a hyperparameter named gridsize that can be changed in SODA to increase or decrease the granularity of the clustering results. Table 4 shows how increasing the granularity

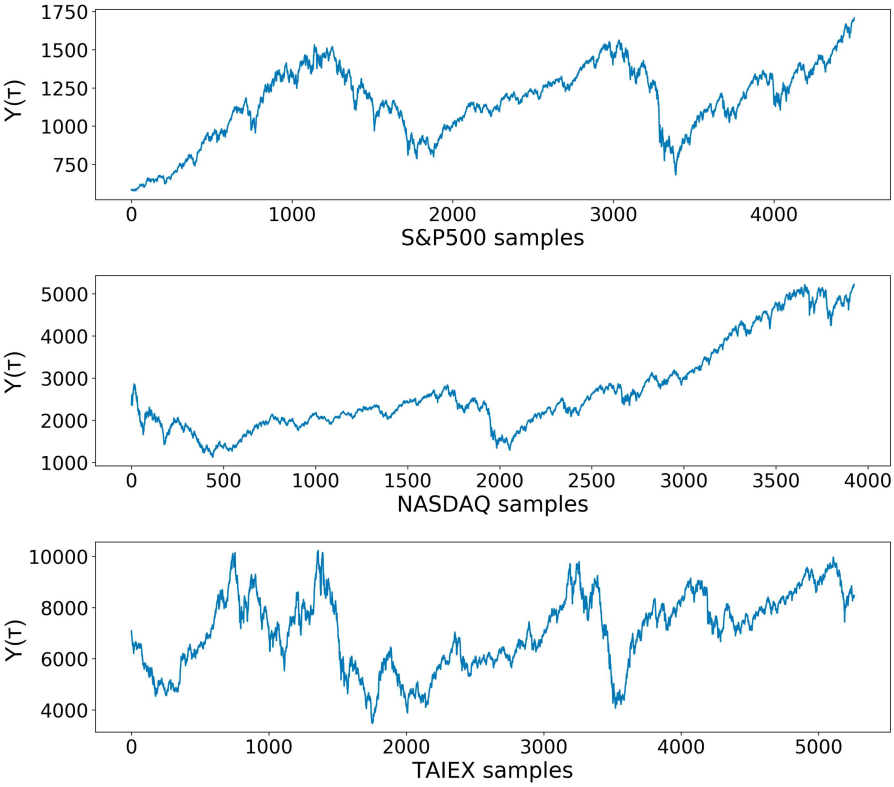


Figure 13 – Datasets for SODA-T2FTS.

of the SODA algorithm affects the partitioning scheme of the TAIEX time series sliding windows.

It can be observed that considering the same data, the higher the gridsize, more focal points are identified therefore more partitions are generated for the same input data. Increasing the number of partitions of the UoD translates into a higher number of membership functions in the model, which may affect its accuracy and precision, but also increases the computational costs associated to the forecasting process. The "Average"values in Table 4 correspond to the average number of partitions (rounded) considering all windows, and will be the values used in further analysis. In order to assess model performance, the comparison metric chosen is the Root Mean Squared Error - RMSE value of the current model, which is the standard accuracy metric used to evaluate the performance of time series forecasting methods and is described in Equation (3.4).

$$RMSE = \sqrt{\frac{\sum(actual\ value - forecast\ value)^2}{number\ of\ observations}} \tag{3.4}$$

The results SODA-T2FTS obtained for the TAIEX time series are presented in Table 5. The RMSE values obtained for each window in the sliding method vary according to the gridsize value. The greater the gridsize, the lower the RMSE value considering the same data samples (each window).

Table 4 – TAIEX time series UoD partitions according to SODA gridsize.

Window	Gridsize									
	1	2	3	4	5	6	7	8	9	10
1° ([0:1000])	1	4	8	11	14	18	23	33	42	49
2° ([200:1200])	1	2	6	9	17	21	28	36	45	54
3° ([400:1400])	1	5	7	10	18	20	26	38	43	52
4° ([600:1600])	1	4	8	10	16	21	28	40	49	59
5° ([800:1800])	1	3	6	12	14	18	34	41	53	62
6° ([1000:2000])	1	3	7	11	16	25	28	41	49	60
7° ([1200:2200])	1	2	6	10	15	22	27	31	42	52
8° ([1400:2400])	1	3	5	10	14	17	25	30	37	45
9° ([1600:2600])	1	3	5	12	14	18	23	32	39	51
10° ([1800:2800])	1	2	6	8	11	15	18	25	29	39
11° ([2000:3000])	1	4	8	10	12	18	26	26	32	39
12° ([2200:3200])	2	2	5	12	16	20	25	33	36	44
13° ([2400:3400])	1	5	6	10	16	19	23	27	34	41
14° ([2600:3600])	2	4	7	8	15	21	30	39	44	57
15° ([2800:3800])	1	2	4	9	14	18	28	36	43	55
16° ([3000:4000])	1	2	7	10	16	25	30	40	50	58
17° ([3200:4200])	1	3	5	8	12	19	25	39	47	55
18° ([3400:4400])	1	3	6	9	14	18	24	33	41	47
19° ([3600:4600])	1	4	6	8	17	20	28	36	49	54
20° ([3800:4800])	2	3	6	10	14	19	21	33	38	43
21° ([4000:5000])	1	3	7	11	13	17	25	32	41	48
22° ([4200:5200])	1	4	5	11	11	18	20	26	31	36
Average	1	3	6	10	15	19	26	34	42	50

Tables 4 and 5 show that there is a strong relationship between the RMSE forecasting values and the number of partitions of the UoD. There is a trade off between model performance and complexity related to the number of membership functions. Fewer partitions may lead to poor performance and under-fitting, whereas increasing the number of partitions increases model's accuracy until it over-fits. Thus, it is possible to observe that a higher number of partitions translates into a higher number of membership functions, making SODA-T2FTS perform better as the RMSE values decrease until a certain point (over-fitting).

3.4.3 Time series forecasting

The best fit for model parameters was found by the implementation of a gridsearch algorithm, a method that tests every possible combination and finds the optimal parameters for the model. Each model was trained and tested using the Gridsize on the [1:10] range for both original and differentiated data and model order varied on the [1:3] range. The results of this gridsearch procedure for the TAIEX, NASDAQ, and S&P500 time series are shown in Figure 14 and Table 6, in which the best performing models are highlighted in bold.

Table 5 – RMSE values for each window according to SODA gridsize.

Window	Gridsize									
	1	2	3	4	5	6	7	8	9	10
1° ([0:1000])	118.67	118.67	110.52	123.38	125.29	115.53	119.01	114.22	111.47	104.61
2° ([200:1200])	127.19	127.19	119.25	106.09	128.37	117.33	111.04	101.21	101.92	107.27
3° ([400:1400])	145.08	132.76	129.94	135.38	146.96	148.93	146.25	144.43	135.02	133.42
4° ([600:1600])	139.18	139.18	129.66	116.41	113.70	115.68	110.62	112.96	113.70	109.56
5° ([800:1800])	92.00	92.00	92.00	90.60	82.27	74.07	75.84	69.89	65.85	68.38
6° ([1000:2000])	98.69	98.69	97.68	70.91	65.55	61.99	58.03	63.78	56.50	57.22
7° ([1200:2200])	134.14	134.14	134.14	118.47	94.56	71.30	36.13	70.39	64.27	56.81
8° ([1400:2400])	68.60	68.60	68.60	53.17	50.05	52.52	51.70	56.50	58.48	66.50
9° ([1600:2600])	124.57	124.57	124.57	83.91	79.39	69.31	61.58	57.49	55.84	52.47
10° ([1800:2800])	114.63	114.63	114.63	66.52	79.50	50.70	43.53	47.38	44.84	39.23
11° ([2000:3000])	125.91	125.91	84.83	77.14	64.74	64.44	65.09	65.09	58.47	60.66
12° ([2200:3200])	136.42	136.42	136.42	91.33	76.97	80.84	81.46	61.47	66.90	67.19
13° ([2400:3400])	163.12	148.21	153.65	128.68	121.65	122.70	117.21	115.55	120.21	117.65
14° ([2600:3600])	118.98	118.98	112.02	120.42	134.28	133.73	125.03	122.88	118.86	117.12
15° ([2800:3800])	96.27	96.27	96.27	93.79	127.42	123.02	112.08	117.86	112.10	100.19
16° ([3000:4000])	86.86	86.86	86.86	99.77	107.78	105.11	109.04	96.26	92.55	93.93
17° ([3200:4200])	90.46	90.46	90.46	98.75	120.51	100.12	101.16	102.46	97.64	94.22
18° ([3400:4400])	114.34	114.34	114.95	103.48	99.56	91.73	86.95	94.05	94.24	84.54
19° ([3600:4600])	95.71	95.71	87.71	77.05	53.40	58.44	57.68	51.33	57.69	49.65
20° ([3800:4800])	107.85	107.85	94.95	84.31	64.74	57.17	75.87	55.23	57.05	53.65
21° ([4000:5000])	108.23	108.23	103.85	63.36	77.69	64.52	65.65	55.49	48.01	48.22
22° ([4200: 5200])	79.56	79.56	75.69	69.63	69.63	65.08	67.57	68.15	67.92	67.26

This way, it was possible to analyze the performance of different parameter combinations and find the best one according to the calculated RMSE value. For the TAIEX, using Gridsize = 10 (which created 50 partitions) and the number of lags = 1 generated the lowest RMSE value. For NASDAQ, the combination was Gridsize = 8 (which created 27 partitions) and number of lags = 1 and for S&P500, it was Gridsize = 9 (which created 28 partitions) and number of lags = 1. In Table 6, the columns "Partitions (avg.)"and "RMSE (avg.)"corresponds to the average of the number of partitions and RMSE values obtained for all windows in the sliding window method, respectively, and "Time"indicates the execution time of all windows in seconds.

For the three datasets, first order models outperformed second and third order ones. This can be understood considering the random walk theory and analysing the autocorrelation function (ACF) of the TAIEX time series, shown in Figure 15, which finds the correlation coefficient between a time series and its lagged version at each lag, describing how the present value of a time series is related with its past values (Box et al., 2015). The $x - axis$ contains the values for the lag, the $y - axis$ contains the values for the Pearson's correlation coefficient at each lag and the blue shaded region is a confidence interval. If the height of the bars is outside this region, it means the correlation is statistically significant. The random walk theory suggests that stock prices in financial time series are similar to random walk stochastic processes, acting as a sum of small normally-distributed random steps. Despite its long-term memory, for short term

Table 6 – Number of partitions, RMSE values and execution times obtained in the Gridsearch procedure.

		TAIEX			NASDAQ			S&P500		
Gridsize	Order	Partitions (avg.)	RMSE (avg.)	Time(s)	Partitions (avg.)	RMSE (avg.)	Time(s)	Partitions (avg.)	RMSE (avg.)	Time(s)
1	1	1	113.02	30.38	3	27.69	22.61	4	10.67	26.82
	2		151.38	31.84		41.87	25.39		16.86	32.02
	3		182.33	33.40		51.81	30.62		21.09	42.88
2	1	3	111.78	33.30	6	27.01	25.60	8	9.44	30.88
	2		150.19	38.33		41.12	31.55		16.46	39.20
	3		181.57	48.64		51.67	43.93		20.39	56.44
3	1	6	107.21	38.58	8	26.68	28.13	9	8.47	32.74
	2		140.00	47.43		39.96	35.85		16.02	41.11
	3		171.38	66.27		50.81	50.89		20.29	60.01
4	1	10	94.21	44.40	11	26.16	31.14	12	8.79	35.10
	2		134.04	55.08		39.99	38.57		16.45	43.89
	3		164.99	78.16		50.20	54.35		20.31	64.23
5	1	15	94.73	49.91	15	25.79	33.15	15	8.49	38.25
	2		132.93	60.67		40.18	40.06		16.33	47.00
	3		160.59	87.85		50.18	58.24		19.88	69.44
6	1	19	88.38	54.57	19	25.09	36.25	18	8.70	39.87
	2		126.10	66.06		38.60	43.41		16.10	49.07
	3		155.24	97.26		48.03	62.26		19.74	74.32
7	1	26	85.39	61.22	23	24.51	39.05	21	8.73	42.63
	2		123.06	73.44		38.29	48.78		15.83	52.36
	3		151.38	107.96		47.43	69.02		19.25	79.65
8	**1**	34	83.82	69.24	**27**	**24.20**	**43.40**	24	8.21	44.88
	2		121.93	82.08		37.94	50.47		15.71	55.37
	3		149.00	116.26		47.57	72.80		18.96	84.54
9	**1**	42	81.80	76.99	32	24.91	46.14	**28**	**8.13**	**48.01**
	2		119.95	88.91		38.21	54.47		15.62	58.81
	3		147.82	122.93		46.95	76.07		18.85	88.30
10	**1**	**50**	**79.53**	**85.40**	37	24.39	49.23	32	8.14	51.79
	2		119.05	96.79		38.13	57.76		15.39	62.51
	3		146.01	129.74		46.09	77.23		18.39	92.39

forecasting the last lag is enough. The ACF plot also shows that each lag from the time series has a strong correlation to the previous one, as commonly observed in financial time series. This indicates that a first order model would perform better in a forecasting method and also explains why SODA-T2FTS first order models outperformed higher order models.

SODA-T2FTS results are then compared to other models from literature: Naïve (last value repeated), ARIMA (Chatfield, 2001), Hwang (Hwang; Chen; Lee, 1998), traditional FTS - FTS (Song; Chissom, 1993), Conventional FTS - CFTS (Chen et al., 1996), Weighted FTS - WFTS (Yu, 2005), Improved Weighted FTS - IWFTS (Ismail; Efendi, 2011), Exponentially Weighted FTS - EWFTS (Sadaei et al., 2014), High Order FTS - HOFTS (Severiano et al., 2017), Trend Weighted FTS - TWFTS (Cheng; Chen; Chiang, 2006) and the recently introduced Probabilistic Weighted FTS - PWFTS (Silva et al., 2019). Considering all models, the comparison was performed on the same design approach, using as gridsearch parameters model order on the [1:3] range and number of

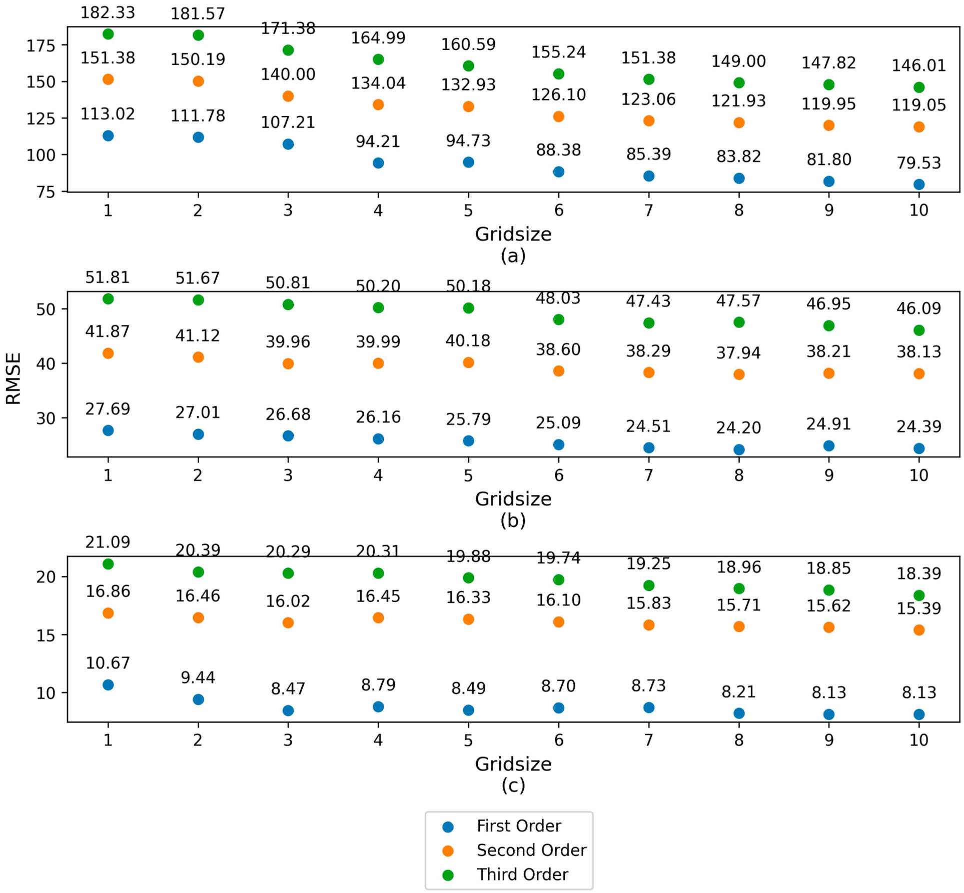


Figure 14 – RMSE values for the TAIEX (a), NASDAQ (b) and S&P500 (c) time series.

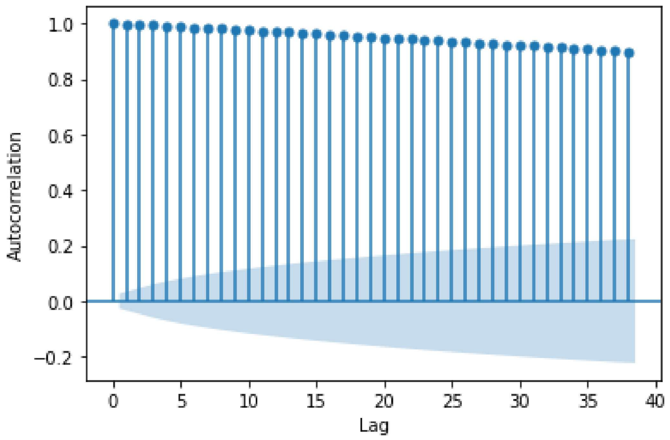


Figure 15 – ACF plot for the TAIEX time series.

partitions on the [1:50] range. All models were trained and tested following the same approach for the gridsearch, using the sliding window method and ranked according to their RMSE mean values, for all methods in the comparison in Table 7. Aiming a better visualization and understanding, instead of showing all the gridsearch results as done in Table 6, only the best performing results are reported in Table 7.

Table 7 shows that SODA-T2FTS is computationally heavier than the other models as it presents longer executions times but despite that, partitioning the UoD using SODA improved accuracy of the model which made SODA-T2FTS be ranked as number 1 for all test datasets in Table 7. The Friedman Aligned Ranks test - FRA (Silva et al., 2020a) was used to evaluate the RMSE values obtained by each model for each dataset. The FRA test is a nonparametric statistical test in which the default assumption, or null hypothesis H_0, is that the multiple paired samples have the same distribution, meaning that the models are indistinguishable. A rejection of the null hypothesis indicates that one or more of the paired samples has a different distribution, meaning that the models can be differentiated from one another (Mahmoudi; Mahmoudi; Pak, 2019).

To carry out pairwise comparisons, a *post-hoc* procedure is applied. The Finner method was used with SODA-T2FTS as the control method. The null hypothesis H_0 implies that the control and test models are equal and the hypothesis H_1 denotes that control and test models are statistically different (Ng; Lee; Lee, 2020). The significance level α is a measure of the strength of the evidence that must be present in the sample for the null hypothesis to be rejected and was chosen to be $\alpha = 0.05$. Results from the Friedman Aligned Ranks test rejected H_0 with a statistic of 25.427 and *p-value* of 0.004592, and results from the Finner *post-hoc* test in Table 8 show that SODA-T2FTS is statiscally different from Naive, Hwang and CFTS methods.

3.4.4 Model Complexity

One of the challenges in machine learning is building forecasting algorithms with high accuracy while keeping them "simple"enough to make them interpretable. Although complex models usually achieve better accuracy, interpreting them is more difficult. When working on any forecasting problem it is important to understand the trade-off between model interpretability and model performance. Table 9 shows the average number of rules generated by each model in the training procedure and the corresponding average RMSE value obtained, considering the same number of partitions for the TAIEX dataset. Observe that SODA-T2FTS presents a slightly higher number of rules than the other methods, showing an even closer number to PWFTS number of rules. As the number of partitions increases, SODA-T2FTS's RMSE values decrease and the model outperforms the other methods. This shows an important advantage of SODA-T2FTS, which is obtaining better performance requiring almost the same number of rules as type-1 FTS methods, indicating that SODA-T2FTS can obtain lower error metrics without the need of increasing model complexity.

Table 7 – Calculated RMSE values.

Dataset	Model	Order	Part.	RMSE		Rank	Time(s)
				AVG	STD		
TAIEX	ARIMA	1,0,0	-	87.32	28.15	5	1.11
	CFTS	1	31	95.08	31.11	10	9.17
	EWFTS	1	7	88.70	30.80	8	2.97
	FTS	1	10	88.68	28.38	7	4.42
	HOFTS	3	5	91.06	30.55	9	8.20
	Hwang	3	5	105.54	42.75	11	0.87
	IWFTS	1	4	87.42	28.04	6	1.73
	Naive	1	-	127.24	40.68	12	0.04
	PWFTS	3	3	83.72	27.63	2	32.65
	TWFTS	1	10	85.14	28.60	3	3.95
	SODA-T2FTS	**1**	**50**[a]	**79.53**	**26.80**	**1**	**85.40**
	WFTS	1	4	87.21	27.71	4	2.44
NASDAQ	ARIMA	1,0,0	-	28.53	11.07	6	0.59
	CFTS	1	10	31.89	11.02	10	2.73
	EWFTS	1	10	29.01	11.02	7	1.59
	FTS	1	15	29.50	9.67	8	4.81
	HOFTS	2	10	30.26	10.94	9	5.68
	Hwang	2	10	36.11	14.79	11	0.81
	IWFTS	1	5	28.31	10.68	5	1.45
	Naive	1	-	41.24	15.15	12	0.03
	PWFTS	3	3	27.38	10.43	2	19.68
	TWFTS	1	8	27.64	10.65	3	2.00
	SODA-T2FTS	**1**	**27**[b]	**24.20**	**8.35**	**1**	**43.40**
	WFTS	1	4	28.22	10.80	4	1.60
S&P500	ARIMA	1,0,0	-	11.79	4.00	10	0.57
	CFTS	1	7	11.65	3.55	9	2.33
	EWFTS	1	2	10.32	3.68	7	1.09
	FTS	1	18	9.54	3.03	2	5.02
	HOFTS	2	6	11.18	3.03	8	3.37
	Hwang	3	29	12.11	4.26	11	3.77
	IWFTS	1	2	10.26	3.65	6	0.65
	Naive	1		15.31	5.20	12	0.02
	PWFTS	1	3	9.60	3.21	3	3.88
	TWFTS	1	9	10.14	3.56	4	1.94
	SODA-T2FTS	**1**	**28**[c]	**8.13**	**3.56**	**1**	**48.01**
	WFTS	1	2	10.19	3.65	5	1.10

[a] For Gridsize = 10.

[b] For Gridsize = 8.

[c] For Gridsize = 9.

3.4.5 Footprint of uncertainty in results

To investigate the actual importance of the footprint of uncertainty (FOU) size when constructing the interval type-2 fuzzy sets in the model, nine FOU levels were

Table 8 – RMSE *post-hoc* tests using SODA-T2FTS as control method.

Comparison	Statistic	*p-value*	Hypothesis
SODA-T2FTS vs Naive	3.40993	0.00712	Rejected H_0
SODA-T2FTS vs Hwang	3.17744	0.00814	Rejected H_0
SODA-T2FTS vs CFTS	2.75119	0.02160	Rejected H_0
SODA-T2FTS vs HOFTS	1.97621	0.12685	Accepted H_0
SODA-T2FTS vs FTS	1.93746	0.12685	Accepted H_0
SODA-T2FTS vs EWFTS	1.27872	0.33726	Accepted H_0
SODA-T2FTS vs IWFTS	1.16248	0.35707	Accepted H_0
SODA-T2FTS vs ARIMA	1.12373	0.35707	Accepted H_0
SODA-T2FTS vs WFTS	1.08498	0.35707	Accepted H_0
SODA-T2FTS vs TWFTS	0.73624	0.49391	Accepted H_0
SODA-T2FTS vs PWFTS	0.65874	0.51006	Accepted H_0

Table 9 – Average number of rules and RMSE obtained by each model.

Partitions	**TWFTS**		**WFTS**		**IWFTS**		**EWFTS**		**CFTS**		**PWFTS**		**SODA-T2FTS**	
	Rules	**RMSE**	**Rules**	**RMSE**	**Rules**	**RMSE**	**Rules**	**RMSE**	**Rules**	**RMSE**	**Rules**	**RMSE**	**Rules**	**RMSE**
1	1	410.6	1	410.62	1	410.6	1	410.6	1	410.6	1	410.62	1.14	113.02
3	3	106.6	3	92.41	3	92.54	3	95.43	3	126.1	3	83.94	3.18	111.78
6	6	86.36	6	87.30	6	87.46	6	90	6	102.5	6	85.10	6.18	107.21
10	9.65	85.15	9.65	88.98	9.65	88.78	9.65	91.04	9.65	101.7	10.00	86.97	9.86	94.21
15	13.13	86.13	13.13	89.71	13.13	89.86	13.13	91.38	13.13	99.69	14.70	87.93	14.09	94.73
19	15.83	86.66	15.83	89.93	15.83	89.87	15.83	91.53	15.83	96.17	18.22	88.28	18.32	88.38
26	20.65	88.19	20.65	91.25	20.65	91.04	20.65	91.78	20.65	94.95	23.91	88.92	23.77	85.39
34	25.57	89.37	25.57	91.98	25.57	91.65	25.57	92.84	25.57	94.89	29.39	88.95	29.55	83.82
42	30.52	91.38	30.52	93.89	30.52	93.41	30.52	94.11	30.52	95.01	34.83	89.29	34.77	81.80
50	34.74	92.09	34.74	94.42	34.74	93.98	34.74	94.68	34.74	94.8	40.26	89.36	40.50	79.53

defined: from 0.1 to 0.9. Table 10 shows the standard deviation of RMSE values for all gridsize tested using different FOU levels for the TAIEX time series. Due to the low standard deviation values, it can be observed that for this dataset, changing the FOU size of the membership functions does not play a significant role in the forecasting output. Type-2 itself already presents great improvement compared to type-1 models, but changing the FOU parameter within the interval type-2 fuzzy logic did not change the output significantly.

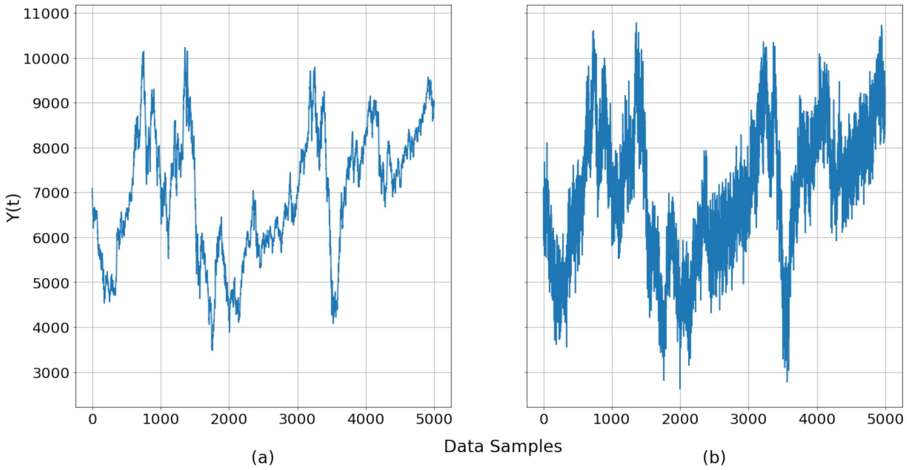
3.4.6 Noise Response

White gaussian noise was added to the TAIEX time series in order to challenge the proposed model and analyse how the forecasting models would perform when dealing with increasing noise rates in the original data. Gaussian noise represents a close approximation of noise in real world scenarios and consequently is one of the common noise adding procedures in the literature. The variance, or equivalently the standard deviation, completely characterizes the noise. The standard deviation may be thought of as a measure of the expected “amplitude” of the noise; its square captures the expected power. Noise was added to the original time series according to the Signal-to-Noise Ratio (SNR), which can be defined as the ratio of signal power to noise power, and allows for artificially adding

Table 10 – RMSE values and their standard deviation for TAIEX time series forecasting using different FOU levels.

Gridsize	FOU									STD
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	
1	113.02	113.02	113.02	113.02	113.02	113.02	113.02	113.02	113.02	0.00
2	111.87	111.83	111.78	111.78	111.87	111.83	111.78	111.83	111.78	0.04
3	107.79	107.59	107.21	107.21	107.79	10.,59	107.21	107.59	107.21	0.25
4	95.56	95.01	94.21	94.21	95.56	95.01	94.21	95.01	94.21	0.58
5	95.91	95.39	94.73	94.73	95.91	95.39	94.73	95.39	94.73	0.50
6	89.16	88.75	88.38	88.38	89.16	88.75	88.38	88.75	88.38	0.33
7	85.74	85.54	85.39	85.39	85.74	85.54	85.39	85.54	85.39	0.15
8	84.02	83.90	83.82	83.82	84.02	83.90	83.82	83.90	83.82	0.08
9	82.07	81.95	81.80	81.80	82.07	81.95	81.80	81.95	81.80	0.12
10	79.79	79.65	79.53	79.53	79.79	79.65	79.53	79.65	79.53	0.11

noise to a clean signal in a controlled manner. To generate noise based on a desired SNR, the power of the original signal is first calculated, and then noise is added with a power level determined by the target SNR. This ensures that the resulting noisy signal has a specific ratio of signal power to noise power, often expressed in decibels (dB). A high SNR corresponds to a clear signal (low noise) and a low SNR corresponds to a noisy signal (high amounts of noise). Figure 16 shows the original TAIEX time series in comparison to itself after white noise was added (SNR = 20%). Table 11 shows the RMSE values obtained when SODA-T2FTS, PWFTS and HOFTS were used to forecast the TAIEX time series with different noise ratios. It can be observed that SODA-T2FTS handles the increasing noise in data better than PWFTS and HOFTS for SNR values ranging from 80% to 50%. From 45% to 35%, PWFTS performs better and from 30% to 20%, the highest added noise rates, SODA-T2FTS obtained the lowest RMSE values.



(a) (b)

Figure 16 – TAIEX original data samples (a) and after noise was added (b).

In conclusion, this chapter discussed a univariate interval type-2 FTS prediction model, SODA-T2FTS, that combines the capacity of interval type-2 fuzzy sets to effectively

Table 11 – RMSE values and their standard deviation for TAIEX time series with added noise.

SNR (%)	PWFTS				SODA-T2FTS				HOFTS			
	Part.	Order	RMSE		Gridsize	Order	RMSE		Part.	Order	RMSE	
			AVG	STD			AVG	STD			AVG	STD
80	3	3	84,44	28,80	10	1	79,84	28,03	5	3	92.29	31.73
75	3	1	84,55	28,67	10	1	80,17	28,33	5	3	92.42	31.20
70	3	3	84,47	28,80	10	1	80,37	26,66	5	3	92.43	31.58
65	3	3	84,66	28,87	9	1	79,90	28,22	5	3	92.49	31.59
60	3	3	84,69	28,66	10	1	79,41	27,67	5	3	93.11	31.58
55	3	3	85,52	28,63	10	1	80,59	26,96	7	3	94.11	31.16
50	3	3	87,97	28,20	10	1	83,47	26,79	5	3	95.92	31.93
45	3	3	93,96	27,06	10	1	99,15	27,78	5	3	99.94	30.96
40	3	3	111,95	23,65	4	1	119,24	20,05	5	3	116.22	23.99
35	3	1	155,19	12,39	1	1	166,27	18,77	7	3	164.97	21.10
30	3	1	254,12	21,43	1	1	251,73	17,94	7	3	268.60	25.96
25	3	3	428,15	29,50	2	1	417,81	16,58	33	3	466.47	21.03
20	3	1	772.32	41.08	2	1	749.35	55.26	33	3	800.81	41.81

model variations in uncertainty, to a data-driven partitioning algorithm called SODA. SODA is used to define the number of membership functions in the model, ensuring that this important hyperparameter is selected based on the ensemble properties of the input data rather than user definitions. Results revealed that SODA-T2FTS obtained the lowest errors in all experiments, including those in which noise was added to the original time series, indicating that the proposed forecasting model can predict complex time series with high accuracy using a data-driven approach independent of user interference. In addition to outperforming in accuracy, the proposed approach presented a similar number of fuzzy rules compared to other FTS methods, thus, delivering better performance for comparable interpretability.

4 The ADP-T2FTS Approach

While Chapter 3 introduced SODA-T2FTS as a data-driven solution to reduce subjectivity in FTS partitioning, it represents just one pathway to address the broader challenge of automatic and adaptive fuzzy set generation. As discussed in Chapter 2, conventional partitioning methods are often limited by fixed heuristics or require user-defined parameters, leading to models that may struggle to generalize across diverse datasets. In this regard, in this chapter an alternative approach named ADP-T2FTS is explored, which employs the Autonomous Data Partitioning (ADP) algorithm to define the UoD based on local and global data density. Moreover, it presents a numerical illustration of the model procedure, for clarity.

ADP-T2FTS receives as input the time series data and then all the steps necessary for training and testing the model are performed in order to obtain error metrics and forecasting results. For illustration purposes of the following procedures, this chapter is analysed using the EBOP dataset, which contains information about daily historical Brent Oil Prices in Europe and will be deeply discussed in Section 4.3.1, along with other datasets.

4.1 Training Procedure

The objective of the forecasting method is to learn the patterns from the time series training data, thus generating the model $\mathcal{M}$ to predict future data. The exemplified steps for training are listed as follows.

4.1.1 Universe of Discourse

First, the highest and the lowest prices of the Brent Oil Prices training dataset are determined. According to Table 13, which describes the data, $min(Y(t)) = 26.01$ and $max(Y(t)) = 86.07$. Considering Equation (2.3) and an expansion of 10%, the UoD is $UoD = [23.409, 94.677]$.

4.1.2 ADP Algorithm for UoD Partitioning

Considering the time series data as input, the summarized procedure of the ADP partitioning algorithm is as follows (Gu; Angelov; Príncipe, 2018):

- **Stage 1: Rank order the samples with regards to the distance to the global mode**

  The process begins by organizing the unique data samples $\{\boldsymbol{u}\}_K$ into an indexing list, denoted by $\{\boldsymbol{u}^*\}_K$ based on their mutual distances and global density values

AD^G. Firstly, calculate the global densities of the unique data samples $AD_K^G(\boldsymbol{u}_i)(i = 1, 2, ..., L_K)$ using the following Equation defined in Chapter 2:

$$AD_K^G(\mu^j) = \frac{f_K}{1 + \frac{||u_K - \mu_K||^2}{X_K - ||\mu_K||^2}} \tag{4.1}$$

The unique data sample with the highest global density is then selected as the first element of the indexing list $\{\boldsymbol{u}^*\}_{L_K}$. $\boldsymbol{u}^{*1}$ is set as the first reference point: $\boldsymbol{u}^{*r} \leftarrow \boldsymbol{u}^{*1}$ and u^{*1} is removed from $\{\boldsymbol{u}\}_{L_K}$. Then, by selecting out the unique data sample which is nearest to $\boldsymbol{u}^{*r}$, the second element of $\{\boldsymbol{u}^*\}_{L_K}$ denoted by $\boldsymbol{u}^{*2}$ is identified and it is set as the new reference point $\boldsymbol{u}^{*r} \leftarrow \boldsymbol{u}^{*2}$ and is also removed from $\{\boldsymbol{u}\}_{L_K}$. This procedure is repeated until $\{\boldsymbol{u}\}_{L_K}$ is empty, and the rank ordered list $\{\boldsymbol{u}^*\}_{L_K}$ is finally derived. Based on this list, the global density of the unique data samples can be ranked as $\{AD_K^G(\boldsymbol{u}^*)\} = \{AD_K^G(\boldsymbol{u}^{*1}), AD_K^G(\boldsymbol{u}^{*2}), ..., AD_K^G(\boldsymbol{u}^{L_K})\}$.

- **Stage 2: Detecting local maxima (local modes)**

  Based on the list $\{\boldsymbol{u}^*\}_{L_K}$ and the ranked global density $\{AD_K^G(\boldsymbol{u}^*)\}$, one can identify all the data samples with the local maxima of AD^G. The collection of data samples with the local maxima of AD^G is denoted as $\{\boldsymbol{u}^{**}\}_{L_K^*} = \{\boldsymbol{u}^{**j} \mid j = 1, 2, ..., L_K^*\}(L_K^* < L_K)$.

- **Stage 3: Forming data clouds**

  The local peaks found and stored in $\{\boldsymbol{u}^{**}\}_{L_K^*}$ are then used to attract the data samples $\{\boldsymbol{x}\}_K$ that are closer to them utilizing a *min* operator as shown in Equation (4.2). As all the data samples are assigned to the closest local maxima, a number of Voronoi tessellations are naturally formed, and data clouds are built around the local maxima. It is important to note that this process is free from any threshold.

$$\textit{winning cloud} = \arg \min_{j=1,2,...,L_k^*} \left(\|\boldsymbol{x}_i - \boldsymbol{u}^{**j}\| \right), \quad i = 1, 2, \ldots, K; \quad L_k^* > 1 \tag{4.2}$$

  After the data clouds are formed, one can easily calculate the actual center (mean) $\boldsymbol{\mu}^j$, the standard deviation $\sigma^j(j = 1, 2, ..., L_K^*)$ and the support S^j, which represents the number of data samples within the data cloud, for every data cloud. This procedure is determined from the data without any prior assumption, except the selection of the metric.

- **Stage 4: Filtering local modes**

  In this stage, the initial Voronoi tessellations are filtered and combined into larger, more meaningful data clouds, as the data clouds formed in Stage 3 may contain some less representative ones. The global densities at the data clouds centers $\{\boldsymbol{\mu}\}_{L_K^*}$ are firstly calculated as follows ($j = 1, 2, ..., L_K^*$):

$$AD_K^G(\boldsymbol{\mu}^j) = \frac{S^j}{1 + \frac{\|\boldsymbol{\mu}^j - \boldsymbol{\mu}_K\|^2}{X_K - \|\boldsymbol{\mu}_K\|^2}} \tag{4.3}$$

In order to identify the centers with the local maxima of the global density, the following three objectively derived quantifiers of the data pattern are introduced in Equations (4.4), (4.5) and (4.6).

$$\eta_K^c = \frac{2\sum_{p=1}^{L_K^*-1}\sum_{q=p+1}^{L_K^*} ||\mu^p - \mu^q||}{L_K^*(L_K^* - 1)} \tag{4.4}$$

$$\gamma_K^c = \frac{x, y \in \{\mu\}_{L_K^*}, x \neq y, ||x - y|| \leq \eta_K^c}{M_\eta} \tag{4.5}$$

$$\lambda_K^c = \frac{x, y \in \{\mu\}_{L_K^*}, x \neq y, ||x - y|| \leq \gamma_K^c}{M_\gamma} \tag{4.6}$$

η_K^c is the average Euclidean distance between any pair of existing local modes. γ_k^c is the average Euclidean distance between any pair of existing local modes within a distance less than η_K^c, and M_η in Equation (4.5) is the number of such local mode pairs. λ_K^c is the average Euclidean distance between any pair of existing local modes within a distance less than γ_k^c, M_γ in Equation (4.6) is the number of such local mode pairs. The quantifier λ_K^c can be viewed as the estimation of the distances between the strongly connected data clouds condensing the local information from the whole dataset. Moreover, instead of relying on a fixed threshold, which may frequently fail, η_K^c, γ_k^c, and λ_K^c are derived from the dataset objectively and according to the Granularity hyperparameter G. Each center μ^j ($j = 1, 2, ..., L_K^*$) is compared with the centers of neighboring data clouds in terms of the global density, and the following condition is evaluated using Equation (4.7).

$$\begin{gathered} IF\left[||\mu^i - \mu^j|| \leq \frac{\lambda_K^c}{2}\right] \\ THEN(\mu^i \ is \ neighbouring \ \mu^j) \end{gathered} \tag{4.7}$$

λ_K^c is the average distance between the centers of any two potentially relevant data clouds. Therefore, when the condition in Equation (4.7) is satisfied, both μ^i and μ^j are highly influencing each other and, the data samples within the two corresponding data clouds are strongly connected. Therefore, the two data clouds are considered neighbors. This criterion also guarantees that only small-size (less critical) data clouds that significantly overlap with large-size (more important) ones will be removed during the filtering operation.

After the filtering operation (Stage 4), the data cloud centres with local maximum global densities denoted by $\{\boldsymbol{\mu}^*\}_{L_K^{**}} = \mu^{*j} \mid j = 1, 2, ..., L_K^{**}, L_K^{**} \leq L_K^*\}$ are obtained. After that, $\{\boldsymbol{\mu}^*\}_{L_K^{**}}$ are used as local modes for forming data clouds in stage 3 and are filtered in stage 4. Stages 3 and 4 are repeated until all the distances between the existing local modes exceed $\frac{\lambda_K^c}{2}$. Finally, it is possible to obtain the remaining centers with the local maxima of AD^G, denoted by $\{\boldsymbol{\mu}^o\}$, and use them as the local modes to form data clouds. After the data clouds are formed by all the identified local modes, the algorithm identifies the ones with support equal to 1, meaning that there is no sample associated with these data clouds except for the local modes. This kind of local modes are considered to be outliers so their data clouds are removed and these local modes are assigned to the nearest normal data cloud. Once this process is over and all data clouds have been identified, their corresponding centers, supports, standard deviations, members, and other parameters can be extracted.

The UoD needs to be partitioned for the the fuzzy sets to be created, and ADP is used for that. The number of data clouds that ADP generated from the input data, which is the time series training data, represents the optimal number of partitions for the UoD and is the number that is used for this parameter in the model. The number of partitions is used to define the linguistic variable $\tilde{A}$ to describe the time series Y. For a k number of partitions, create k evenly distributed overlapping fuzzy sets $\tilde{A}_i$ ($i = 1, ..., k$). Each fuzzy set $\tilde{A}_i$ is a linguistic term of the linguistic variable $\tilde{A}$. Using the default Granularity value of $G = 3$, ADP splits the EBOP dataset's previously defined $UoD = [23.409, 94.677]$ into $k = 23$ partitions. Therefore, there were twenty-three equal-sized intervals: $u_1 = [23.409, 29.348]$, $u_2 = [26.379, 32.318]$, $u_3 = [29.349, 35.287]$, $u_4 = [32.318, 38.257]$, ... , $u_{22} = [85.769, 91.708]$ and $u_{23} = [88.738, 94.677]$. From the intervals u_i $(i = 0, 1, ..., 23)$ obtained, to each interval a fuzzy set $\tilde{A}_i$ $(i = 1, 2, ..., 23)$ is assigned, as shown in Figure 17. For this example, it is assumed that the model's membership functions are triangular.

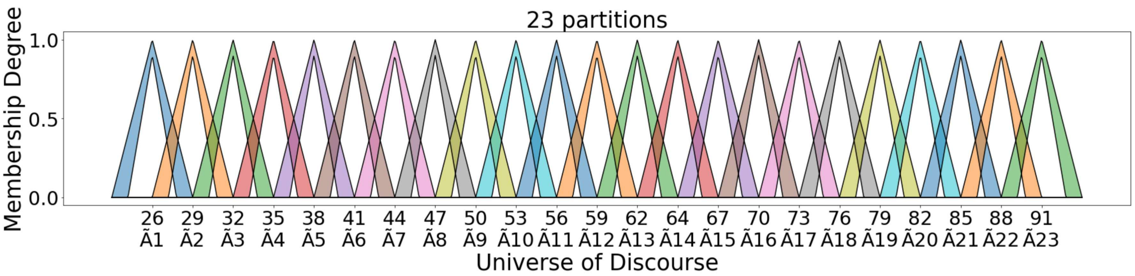


Figure 17 – Fuzzy sets creation from intervals

4.1.3 Fuzzification of time series data

Every sample from the historical training data is fuzzified, and membership values are calculated for all the fuzzy sets. The fuzzification of the Brent oil Prices time series values from July 2nd, 2019 to July 31st, 2019 are shown in Table 12.

Table 12 – Fuzzification and FLR extraction of Brent oil prices for July, 2019

Date	**Brent Oil Prices**	**Fuzzy Set**	**FLR**
07/02/2019	62.72	$\tilde{A}_{14}, \tilde{A}_{15}$	–
07/03/2019	63.53	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}, \tilde{A}_{15} \rightarrow \tilde{A}_{13}, \tilde{A}_{15} \rightarrow \tilde{A}_{14}$
07/04/2019	63.62	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}$
07/05/2019	64.23	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}$
07/08/2019	64.89	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}$
07/09/2019	64.3	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}$
07/10/2019	66.41	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}$
07/11/2019	67.64	$\tilde{A}_{14}, \tilde{A}_{15}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{14}, \tilde{A}_{13} \rightarrow \tilde{A}_{15}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{15}$
07/12/2019	66.65	$\tilde{A}_{14}, \tilde{A}_{15}$	$\tilde{A}_{14} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{15}, \tilde{A}_{15} \rightarrow \tilde{A}_{14}, \tilde{A}_{15} \rightarrow \tilde{A}_{15}$
07/15/2019	66.86	$\tilde{A}_{14}, \tilde{A}_{15}$	$\tilde{A}_{14} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{15}, \tilde{A}_{15} \rightarrow \tilde{A}_{14}, \tilde{A}_{15} \rightarrow \tilde{A}_{15}$
07/16/2019	65.87	$\tilde{A}_{14}, \tilde{A}_{15}$	$\tilde{A}_{14} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{15}, \tilde{A}_{15} \rightarrow \tilde{A}_{14}, \tilde{A}_{15} \rightarrow \tilde{A}_{15}$
07/17/2019	63.67	$\tilde{A}_{14}, \tilde{A}_{15}$	$\tilde{A}_{14} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{15}, \tilde{A}_{15} \rightarrow \tilde{A}_{14}, \tilde{A}_{15} \rightarrow \tilde{A}_{15}$
07/18/2019	60.7	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}, \tilde{A}_{15} \rightarrow \tilde{A}_{13}, \tilde{A}_{15} \rightarrow \tilde{A}_{14}$
07/19/2019	61.04	$\tilde{A}_{12}, \tilde{A}_{13}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{12}, \tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{12}, \tilde{A}_{14} \rightarrow \tilde{A}_{13}$
07/22/2019	61.96	$\tilde{A}_{12}, \tilde{A}_{13}$	$\tilde{A}_{12} \rightarrow \tilde{A}_{12}, \tilde{A}_{12} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{12}, \tilde{A}_{13} \rightarrow \tilde{A}_{13}$
07/23/2019	62.28	$\tilde{A}_{12}, \tilde{A}_{13}$	$\tilde{A}_{12} \rightarrow \tilde{A}_{12}, \tilde{A}_{12} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{12}, \tilde{A}_{13} \rightarrow \tilde{A}_{13}$
07/24/2019	63.83	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{12} \rightarrow \tilde{A}_{13}, \tilde{A}_{12} \rightarrow \tilde{A}_{14}, \tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{14}$
07/25/2019	63.47	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}$
07/26/2019	62.46	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}$
07/29/2019	62.29	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}$
07/30/2019	62.55	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}$
07/31/2019	64.07	$\tilde{A}_{13}, \tilde{A}_{14}$	$\tilde{A}_{13} \rightarrow \tilde{A}_{13}, \tilde{A}_{13} \rightarrow \tilde{A}_{14}, \tilde{A}_{14} \rightarrow \tilde{A}_{13}, \tilde{A}_{14} \rightarrow \tilde{A}_{14}$

4.1.4 Fuzzy relationships

Fuzzy Logical Relationships (FLRs) are then used to relate previous and current states of the time series instances. Considering for example Table 12 where the oil price for 07/17/2019 is fuzzified as $\tilde{A}_{14}$ and $\tilde{A}_{15}$ and the next instance relative to 07/18/2019 is fuzzified as $\tilde{A}_{13}$ and $\tilde{A}_{14}$. One can then establish the four fuzzy relationships between these consequent instances:

$$\begin{gathered}\tilde{A}_{14} \rightarrow \tilde{A}_{13}\\ \tilde{A}_{14} \rightarrow \tilde{A}_{14}\\ \tilde{A}_{15} \rightarrow \tilde{A}_{13}\\ \tilde{A}_{15} \rightarrow \tilde{A}_{14}\end{gathered} \tag{4.8}$$

Table 12 also shows all the FLRs identified from the time series values of July, 2019. For this time frame, the following FLRGs are generated:

$$
\begin{aligned}
\tilde{A}_{12} &\rightarrow \tilde{A}_{12}, \tilde{A}_{13}, \tilde{A}_{14} \\
\tilde{A}_{13} &\rightarrow \tilde{A}_{13}, \tilde{A}_{14}, \tilde{A}_{15} \\
\tilde{A}_{14} &\rightarrow \tilde{A}_{12}, \tilde{A}_{13}, \tilde{A}_{14}, \tilde{A}_{15} \\
\tilde{A}_{15} &\rightarrow \tilde{A}_{13}\tilde{A}_{14}, \tilde{A}_{15}
\end{aligned}
\tag{4.9}
$$

4.2 Forecasting procedure

4.2.1 Input value fuzzification

After the rule base is built, the model learned is used to forecast future values of input test values. Consider, for example, that there is a need to forecast the EBOP value for 07/08/2019 in Table 12. An input value $y(t)$ needs to be converted into fuzzy values of the linguistic variable $\tilde{A}$. For that, the previous instance $y(t) = 64.23$, relative to the date of 07/05/2019, will be used, which is fuzzified into the fuzzy sets $\tilde{A}_{13}$ and $\tilde{A}_{14}$.

4.2.2 Finding compatible rules

The input value was fuzzified into the fuzzy sets $\tilde{A}_{13}$ and $\tilde{A}_{14}$, so now it is necessary to map the rules that have these two sets as antecedent. For the activated rules, the antecedent represents the fuzzy set that the input value was fuzzified into and consequent will be the possible forecast values $f(t+1)$. The procedure explained in "Subsection 3.1.4: Fuzzy relationship"was performed for the entire time series and then FLRs were generated for all the fuzzy sets. After all FLRs have been identified for all the time series instances, the knowledge rule-base can be generated by the creation of the fuzzy logical relationship groups (FLRG). All the rules which share the same antecedent (left hand side) can be grouped in a FLRG that can be interpreted as the possible future states (right hand side) for a given antecedent. The following rules could be identified after the rule extraction procedure using the entire training dataset:

$$\tilde{A}_{13} \rightarrow \tilde{A}_{11}, \tilde{A}_{12}, \tilde{A}_{13}, \tilde{A}_{14}, \tilde{A}_{15} \ and$$

$$\tilde{A}_{14} \rightarrow \tilde{A}_{12}, \tilde{A}_{13}, \tilde{A}_{14}, \tilde{A}_{15}, \tilde{A}_{16}$$

Now that the associated rules are defined, one can use their corresponding degrees of membership and fuzzy operators to obtain a crisp value.

4.2.3 Defuzzification

For the initial test value $y(t) = 64.23$, fuzzified into the $\tilde{A}_{13}$ and $\tilde{A}_{14}$ fuzzy sets, antecedent processing is performed using the *min* operator and then consequent processing is done using the *max* operator. Aggregation is then performed, generating a single interval type-2 fuzzy set from the rule output fuzzy sets. Using Equations (2.5) and (2.6) it is possible to find the lower and upper limits c_{Lf} and c_{Rg} for the type-reduced fuzzy set, which are calculated to be $c_{\mathrm{Lf}}(64.23) = 63.71$ and $c_{\mathrm{Rg}}(64.23) = 65.38$. The final forecast value $y(t+1)$ is then the centroid of this fuzzy set, calculated by Equation (2.7). Then, the forecast value for 07/08/2019 is $y(t+1) = 64.54$, as seen below

$$y(x) = \frac{1}{2}[c_{Lf}(y(t)) + c_{Rg}(y(t))] = \frac{1}{2}[63.71 + 65.38] = 64.54 \tag{4.10}$$

ADP-T2FTS overall training and forecasting procedures are summarized in Algorithm 2.

Algorithm 2: ADP-T2FTS training and forecasting procedures

```
Split time series data $Y$ into training and test sets;
for Training data $Y_{trg}$ do
    Compute the $UoD$;
    Use ADP to partition $Y_{\mathrm{trg}}$ into $dc$ distinct data clouds;
    Split this $UoD$ into $dc$ partitions;
    Assign an IT2FS $\tilde{A}_{\mathrm{i}}$ $(i = 1, 2, ...dc)$ to each partition;
    Fuzzify each instance from $Y_{\mathrm{trg}}$ by computing membership values across all fuzzy
      sets;
    Detect temporal patterns (FLRs) between consecutive instances from $Y_{\mathrm{trg}}$;
    Organize FLRs into FLRGs;
    Build the model $\mathcal{M}$;
end
for The test data $Y_{test}$ do
    Fuzzify each instance $y(t)$ from $Y_{\mathrm{test}}$ by computing membership values across all
      fuzzy sets;
    for $FLRG \in \mathcal{M}$ do
        if $f(t)$ matches the LHS then
            rules_matched ← FLRG;
        end
    end
    Use rules_matched in the fuzzy inference procedure and compute $f(t+1)$;
    Obtain $y(t+1)$ after type-reduction and defuzzification;
    Compute error metrics;
end
```

4.3 Experimental Results

4.3.1 Case Studies

The EBOP[1] dataset contains daily historical Brent Oil Prices in Europe and is composed of 1277 instances sampled from January/2016 to December/2020. Table 13 presents preliminary statistical analysis, and the behavior of the time series historical values can be observed in Figure 18. Data was divided into a training set containing the first 70% of the data and a test set with the remaining 30% of the data.

[1] https://www.eia.gov/dnav/pet/hist/RBRTED.htm. Access in 16/08/2022

Table 13 – EBOP Statistical analysis

Parameter	Complete data	Train set	Test set
Data	1277	893	384
Mean	55.076	57.715	48.94
Standard deviation	13.651	12.657	13.921
Minimum	9.120	26.01	9.12
First quartile	45.700	48.42	40.892
Median	54.800	55.92	48.825
Third quartile	65.030	68.01	61.068
Maximum	86.070	86.07	70.25

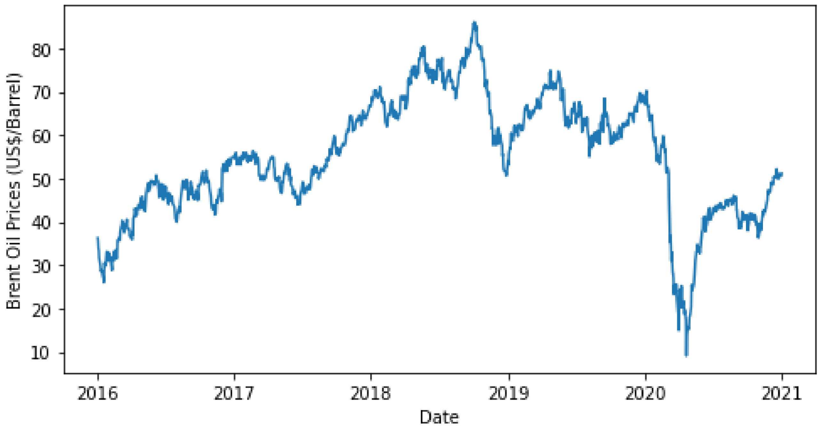


Figure 18 – EBOP dataset

Initially, the EBOP dataset was used to evaluate the forecasting performance of the proposed method, comparing it to existing forecasting models of different orders and design mechanisms. The experiments were run using a laptop with an Intel i5-8265U 1.6 GHz and 8 GB DDR4. All the data and source codes are available on: https://github.com/arthurcaio92/ADP-T2FTS.

The comparison metrics applied to evaluate the performance of time series prediction methods were the root mean-square error (RMSE), defined in Chapter 3 as Equation (3.4), the mean absolute percentage error (MAPE) and the symmetric mean absolute percentage error (SMAPE), expressed by Equations (4.11) and (4.12) respectively, where y_{i} represents the forecast values and $\dot{y}_{\mathrm{i}}$ corresponds to the actual values.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_{\mathrm{i}} - \dot{y}_{\mathrm{i}}|}{\dot{y}_{\mathrm{i}}} \times 100 \tag{4.11}$$

$$SMAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{2\cdot(|y_{\mathrm{i}} - \dot{y}_{\mathrm{i}}|)}{|y_{\mathrm{i}}| + |\dot{y}_{\mathrm{i}}|} \tag{4.12}$$

4.3.2 Parameter Optimization

In the process of building the best-performing forecasting model possible, three key parameters must be defined in ADP-T2FTS before the model can be compared to other methods: the Granularity hyperparameter G, the membership function (MF) that will be used, and model order. The G hyperparameter is set to regulate the level of detail of the clustering results of the ADP algorithm, directly interfering in the partitioning output. Additionally, the MF also affects the model's accuracy by affecting the way the fuzzy sets cover the UoD, and model order dictates how many lags (instances) of the time series are used in the forecasting process. Several tests were performed to find their best values for all parameters based on the RMSE forecasting error metrics obtained from the models generated. The G hyperparameter ranged from 1 to 5, the MF for triangular, trapezoidal, and gaussian MFs, and model order ranged from 1 to 3. Thus, it was possible to find the best values of G and model order and the best MF to be implemented, shown in bold in Table 14. It is also possible to understand how G influences the partitioning ADP output by noting that as G increases, more fuzzy sets are generated, meaning that UoD is being partitioned into more intervals. Another information learned from Table 14 is that the MF has a minor impact on model forecasting accuracy, as the RMSE and MAPE values for triangular, trapezoidal, and gaussian MFs are similar.

4.3.3 ADP Partitioner

The use of ADP for the UoD partitioning is evaluated by comparing the forecasting performance of the ADP generated models to other models that use other partitioning methodologies from the literature. In this present study, the partitioning methods used were SODA (Gu et al., 2018), Entropy (Cheng; Chang; Yeh, 2006), FCM (Li; Cheng; Lin, 2008), c-Means (Zhang; Zhu et al., 2012) and DBSCAN (Ester et al., 1996). Table 15 shows the results for all the models generated using the different partitioning methodologies. Results show that partitioning the UoD with ADP generates the lowest RMSE and MAPE values, confirming that it is an efficient tool for autonomous data partitioning and is completely capable of being used with FTS models.

4.3.4 Model Performance Comparison

ADP-T2FTS model performance was compared to type-1 fuzzy forecasting models in the literature. All models were used to predict the Brent Oil dataset and the forecasting occurred under the same circumstances for all of them. For the comparison, the following models were used: Hwang (Hwang; Chen; Lee, 1998), traditional FTS (FTS), Conventional FTS (CFTS) (Chen et al., 1996), High Order FTS (HOFTS) (Severiano et al., 2017), Weighted High Order FTS (WHOFTS) (Severiano et al., 2017), Weighted FTS (WFTS) (Yu, 2005), Exponentially Weighted FTS (EWFTS) (Sadaei, 2013), Improved Weighted

Table 14 – Parameter optimization.

G	**Partitions**	**Order**	**MF**	**RMSE**	**MAPE**
1	**6**	**1**	**TRI**	**1.5561**	**2.9060**
			TRAP	1.5582	2.9087
			GAUS	1.6222	2.9965
		2	TRI	2.2118	4.0753
			TRAP	2.2039	4.0486
			GAUS	2.2715	4.1181
		3	TRI	2.5857	4.8241
			TRAP	2.5954	4.8496
			GAUS	2.8522	5.2461
2	11	1	TRI	1.7352	3.2770
			TRAP	1.7103	3.2353
			GAUS	1.6918	3.1039
		2	TRI	2.2856	4.2553
			TRAP	2.2935	4.2635
			GAUS	2.3715	4.3293
		3	TRI	2.6906	5.0204
			TRAP	2.6892	4.9711
			GAUS	2.9421	5.3320
3	23	1	TRI	1.8213	3.3663
			TRAP	1.8445	3.4780
			GAUS	1.7686	3.0744
		2	TRI	2.2315	4.1807
			TRAP	2.3297	4.4175
			GAUS	2.5467	4.5195
		3	TRI	2.6021	4.9784
			TRAP	2.6225	5.0932
			GAUS	3.0283	5.8287
4	33	1	TRI	1.7967	3.3390
			TRAP	1.8204	3.3861
			GAUS	1.7736	3.2526
		2	TRI	2.2174	4.1588
			TRAP	2.2290	4.2283
			GAUS	2.4523	4.4368
		3	TRI	2.5399	4.8714
			TRAP	2.5364	4.8820
			GAUS	2.8001	5.4160
5	36	1	TRI	1.7984	3.4242
			TRAP	1.7958	3.4028
			GAUS	1.7614	3.2472
		2	TRI	2.0938	3.9236
			TRAP	2.1268	3.9678
			GAUS	2.4054	4.4666
		3	TRI	2.5014	4.7429
			TRAP	2.5438	4.8402
			GAUS	2.6756	5.2456

Table 15 – Performance comparison of partitioners

Partitioner	**MF**	***G***	**Order**	**Part.**	**RMSE**	**MAPE**
SODA	Triangular	1	1	5	1.5896	2.9541
FCM	Trapezoidal	-	1	7	1.5874	2.9385
ENTROPY	Triangular	-	1	5	1.5881	2.9400
CMEANS	Triangular	-	1	5	1.5879	2.9455
DBSCAN	Triangular	-	1	2	1.5896	2.9540
ADP	**Triangular**	**1**	**1**	**6**	**1.5561**	**2.9060**

FTS (IWFTS) (Ismail; Efendi, 2011), Trend Weighted FTS (TWFTS) (Cheng; Chen; Chiang, 2006), Probabilistic Weighted FTS (PWFTS) (Silva et al., 2019) and Markov Weighted FTS (MWFTS) (Alyousifi et al., 2020). According to Table 16, the results confirm that the ADP-T2FTS achieves better performance and lower error metrics in terms of RMSE and MAPE, for the Brent Oil dataset.

Table 16 – Model Performance Comparison

Model	**Order**	**Partitions**	**RMSE**	**MAPE**
MWFTS	1	36	2.16	4.42
ADP-T2FTS	**1**	**6**	**1.56**	**2.91**
PWFTS	3	7	4.17	9.08
WHOFTS	3	7	4.29	8.9
HOFTS	3	8	5.25	10.92
IWFTS	1	36	2.16	4.42
TWFTS	1	33	2.17	4.46
WFTS	1	34	2.18	4.53
EWFTS	1	36	2.19	4.52
CFTS	1	35	2.19	4.61
FTS	1	7	9.33	20.84
Hwang	2	5	12.94	31.18

Two more time series were used to evaluate model's performance: the University of Alabama enrollments time series ((Chen et al., 1996)) and the Yearly Sunspot number from 1700 to 1987 dataset ((Kim; Chung, 2005)). In Table 17 ADP-T2FTS forecasting metrics were compared to results presented by Pattanayak, Behera and Panigrahi (2021), where a probabilistic intuitionistic fuzzy set based model for high order fuzzy set based time series forecasting was proposed and compared to advanced probabilistic FTS models. In this model, PIFTSF, statistical and non-statistical uncertainty are handled using probability and intuitionistic fuzzy elements, which are then aggregated to obtain the probabilistic intuitionistic fuzzy element. It was also included in this comparison the model proposed in Chapter 3: SODA-T2FTS and for the sake of fairness, the experiment settings were defined to be the same from Pattanayak, Behera and Panigrahi (2021): the first 70% of time series data is considered for training, next 15% is considered for validation, and the recent 15% is considered for testing. Table 17 shows that ADP-T2FTS obtained the lowest SMAPE metric among all models for both Enrollments and Sunspot time series.

Table 17 – SMAPE results for Sunspot and Enrollments datasets

Model	**Sunspot**	**Enrollments**
Aladag et al. (2009)	99.89	199.90
Aladag (2013)	68.72	14.69
Kumar and Gangwar (2015)	67.08	2.80
Bisht and Kumar (2016)	72.61	3.03
Bas et al. (2018)	77.77	14.69
Gupta and Kumar (2019a,b,c)	70.96	6.76
Panigrahi and Behera (2020)	31.42	6.23
PIFTSF	30.71	2.18
SODA-T2FTS	25.97	2.25
ADP-T2FTS	**25.27**	**1.23**

ADP-T2FTS was later used to forecast the well-known Taiwan Stock Exchange Capitalization Weighted Stock Index - TAIEX time series and was compared to state-of-the-art forecasting models designed using type-2 fuzzy logic, neural networks and regression algorithms, studied in Jiang et al. (2018). The models used in the comparison were the univariate conventional regression model (U_R model) (Yu; Huarng, 2008), the univariate neural network model (U_NN model) (Yu; Huarng, 2008), the univariate neural network-based FTS model (U_NN_FTS model) (Huarng; Yu, 2006a; Yu; Huarng, 2008), the univariate neural network-based FTS model with substitutes (U_NN_FTS_S model) (Huarng; Yu, 2006a; Yu; Huarng, 2008), the bivariate conventional regression model (B_R model) (Yu; Huarng, 2008), the bivariate neural network model (B_NN model) (Yu; Huarng, 2008), the bivariate neural network-based FTS model (B_NN_FTS model) (Huarng; Yu, 2006a; Yu; Huarng, 2008), the univariate neural network-based FTS model with substitutes (U_NN_FTS_S model) (Huarng; Yu, 2006a), the type-2 neurofuzzy modeling method (T2NFS) (Liu; Yeh; Lee, 2012), the complex neurofuzzy system and autoregressive integrated moving average models (CNFS-ARIMA) (Li; Chiang, 2012), the direct and iterative local modeling based on the neuro-fuzzy forecasting model (LMNF-D/I) (Peng et al., 2015), the new FTS model combined with ant colony optimization and auto-regression proposed by Cai et al. (2015) and the interval type-2 fuzzy logic model based on a Fuzzy Logical Relationship Map proposed by Jiang et al. (2018).

All models were used to forecast the TAIEX "close" index from 1999 to 2004. For every year, the data from January to October were used for training and from November to December for testing. The RMSE was the comparison metric chosen for this experiment, and the results for all models can be seen in Table 18, with the best-performing ones highlighted in bold. ADP-T2FTS outperforms other models for the years 1999, 2000, 2001, and 2004. For 2002, the T2FNS (Liu; Yeh; Lee, 2012) obtains the lowest RMSE values and for 2003, LMNF-D/I (Peng et al., 2015) is the best performing model.

Table 18 – RMSE values for TAIEX forecasting

Methods	**1999**	**2000**	**2001**	**2002**	**2003**	**2004**
U_R model	164.00	420.00	1070.00	116.00	329.00	146.00
U_NN model	107.00	309.00	259.00	78.00	57.00	60.00
U_NN_FTS model	109.00	255.00	130.00	84.00	56.00	116.00
U_NN_FTS_S model	109.00	152.00	130.00	84.00	56.00	116.00
B_R model	103.00	154.00	120.00	77.00	54.00	85.00
B_NN model	112.00	274.00	131.00	69.00	52.00	61.00
B_NN_FTS model	108.00	259.00	133.00	85.00	58.00	67.00
B_NN_FTS_S model	112.00	131.00	130.00	80.00	58.00	67.00
Cai *et al.*'s method	102.22	131.53	112.59	60.33	51.54	50.33
CNFS-ARIMA	100.01	122.58	115.82	64.34	57.69	55.56
LMNF-D/I	92.19	123.33	116.73	63.66	**50.90**	53.63
T2NFS	97.30	120.90	103.84	**58.10**	51.00	51.73
Jiang's *et al.*'s model	97.61	119.73	113.26	67.39	54.95	56.21
ADP-T2FTS	**91.37**	**118.81**	**100.91**	65.91	57.81	**45.88**

Results reported in this chapter showed that ADP is entirely suitable for studying the universe of discourse partitioning, besides being an autonomous partitioning method that does not depend on user interference. In the experiments type-2 FTS models outperformed type-1 FTS and other advanced forecasting models, showing that type-2 fuzzy logic may be able to model epistemic uncertainty better and minimize its influence in the forecasting process and model design, hence providing higher accuracy.

5 The ADP-T2LIMG Approach

The previous models proposed so far, SODA-T2FTS and especially ADP-T2FTS already present advancements in FTS model design, but the partitioning method can still be more explored in order to better define the UoD intervals. The results reported in Chapter 4 indicate that ADP-T2FTS outperformed other state-of-the-art forecasting models in the literature, including those based on different mechanisms such as type-1 and type-2 fuzzy logic, neural networks, and regression algorithms. However, it is worth observing that only the number of data clouds identified by the ADP partitioning algorithm was employed to partition the UoD and generate the fuzzy sets. Essential information on shape, endpoints, minimum and maximum values, and other characteristics of the data clouds remained unused, leaving valuable information untapped. To bridge this gap, an updated version of the previous model, termed ADP-T2LIMG, is introduced in this chapter. This new model leverages the parameters from ADP's output, utilizing this enriched information to more accurately design and define the fuzzy sets in the forecasting model, aligning them more closely with the data distribution.

The architecture of ADP-T2LIMG consists of different stages including: data cloud identification, partitioning, fuzzification, extraction of fuzzy patterns, Fuzzy Logical Relationships matching, defuzzification, and accuracy measurement. The complete training and forecasting procedures for ADP-T2LIMG are summarised in Algorithm 3. All source codes and data are available at: https://github.com/arthurcaio92/ADP-T2LIMG.

5.1 Training

5.1.1 Data clouds identification

Input the time series $Y(t)$ into ADP for its data to be analyzed and partitioned. ADP works by identifying clusters based on local density maxima and partitioning the data into "data clouds"without requiring user-defined parameters. The procedure, which involves the four primary stages: calculating global densities, identifying local modes, forming data clouds, and filtering clusters, is performed according to definitions set in Section 2.4.2 and 4.1.2.

The output of the ADP algorithm is a set of data clouds, each represented by a center (μ_K), its standard deviation (σ_j), and its support (S^j), which is the number of data points assigned to the cluster. These data clouds objectively reflect the natural data distribution without user intervention or reliance on prior assumptions. Figure 19 shows an example of ADP partitioning of a time series.

Algorithm 3: ADP-T2LIMG training and forecasting method

```
while Not at the end of time series data $Y$ do
    Extract a segment from $Y$ according to a window size $W$;
    Split time series data window into training and test sets;
    for Training data $Y_{trg}$ do
        Compute the $UoD$;
        Use ADP to partition $Y_{\text{trg}}$ into $dc$ distinct data clouds;
        for Each data cloud do
            Create an $UoD_{sub}$ using the minimum and maximum values of the data
              cloud;
            Split this $UoD_{sub}$ into $sub_{dc} = dc/2$ partitions;
            Assign an IT2FS $\tilde{A}_{\text{i}}$ $(i = 1, 2, ...dc)$ to each partition ;
        end
        Match all $UoD_{sub}$ and their respective fuzzy sets into the original $UoD$;
        Filter the fuzzy sets and remove all that are fully overlapped by another set;
        Fuzzify each instance from $Y_{\text{trg}}$ by computing membership values across all
          fuzzy sets;
        Detect temporal patterns (FLRs) between consecutive instances from $Y_{\text{trg}}$ ;
        Organize FLRs into FLRGs;
        Build the model $\mathcal{M}$;
    end
    for The test data $Y_{test}$ do
        Fuzzify each instance $y(t)$ from $Y_{\text{test}}$ by computing membership values across
          all fuzzy sets;
        for $FLRG \in \mathcal{M}$ do
            if $f(t)$ matches the LHS then
                rules_matched ← FLRG;
            end
        end
        Use rules_matched in the fuzzy inference procedure and compute $f(t+1)$;
        Obtain $y(t+1)$ after type-reduction and defuzzification;
        Compute and save RMSE metrics;
    end
    Slide the window across $Y$ by $D$ instances;
end
```

5.1.2 Partitioning

First, define the original UoD using the highest and lowest values in the time series Y, considering an expansion of 10%. ADP outputs the dataset partitioned into a dc number of data clouds, along with their respective information regarding shape, endpoints, minimum and maximum values, and other important characteristics. Unlike what happens for ADP-T2FTS, for ADP-T2LIMG the properties of all data clouds are explored by taking the minimum and maximum values of each data cloud identified by ADP and creating an Internal Universe of Discourse UoD_{sub} for each of them. Then, this UoD_{sub} is split into $sub_{dc} = dc/2$ intervals, and to each partition an interval type-2 fuzzy set (IT2FS) $\tilde{A}_{\text{i}}$ $(i = 1, 2, ...dc)$ is assigned. After that, all UoD_{sub} and their associated

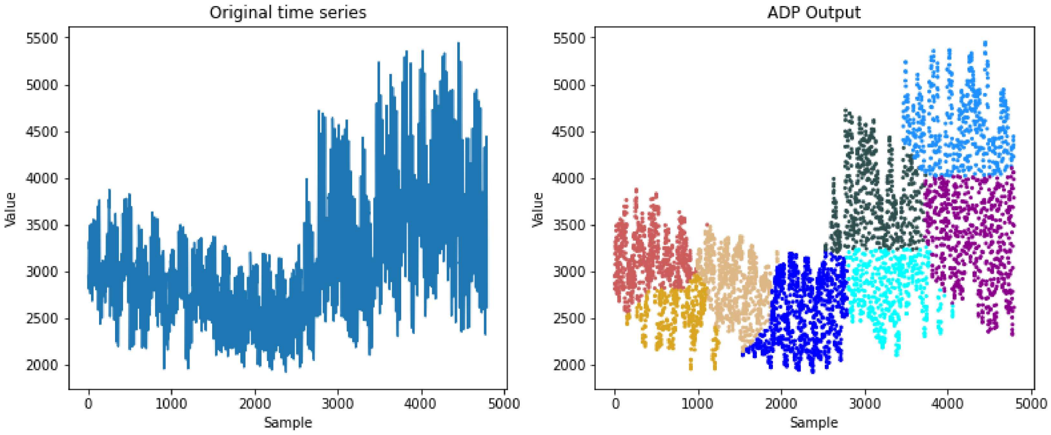


Figure 19 – ADP data cloud partitioning for the DEOK dataset.

fuzzy sets are matched into the original UoD. For some datasets, one can observe that at this point there might be highly overlapping fuzzy sets, so to avoid any harm to the model's interpretability, a filter function is applied to remove fuzzy sets that are fully overlapped by another set, according to the rule presented in Equation (5.1), where $\tilde{A}_2$ is a fuzzy set that is compared to all other fuzzy sets in the model. When the domain of the fuzzy set fully belongs to the domain of another fuzzy set, it means that the former is fully overlapped and can be removed from the model. This way, the UoD is designed with fuzzy sets that better represent the time series data.

$$\text{If domain}(\tilde{A}_2) \in \text{domain}(\tilde{A}_n), \text{ then remove } \tilde{A}_2 \tag{5.1}$$

Figure 20 illustrates the difference in executing the partitioning step of the training procedure (partitioning the UoD) between ADP-T2FTS and ADP-T2LIMG. For a visual understanding of the partitioning procedure, Figure 21 shows a given time series (after differentiation) and the respective ADP output. Then, considering inverted axes in the ADP output plot, Figure 22 exemplifies the creation of an UoD_{sub} for each data cloud and the creation of the fuzzy sets. After all UoD_{sub} are matched into the original UoD and the filter function is applied, the final partitioning is obtained, as shown in figure 23.

5.1.3 Fuzzification and extraction of fuzzy patterns

Each interval is associated with a fuzzy set represented by its membership function. The membership function of each fuzzy set assigns a degree of membership to every value in the UoD). The entire training set from the historical time series data is converted into fuzzy values using the defined fuzzy sets in the UoD. For a given crisp input value from the time series Y, an interval-valued membership degree is obtained for each input, ranging from $\underline{\mu}_{\tilde{A}}(x)$ (the membership degree for the lower membership function) to $\overline{\mu}_{\tilde{A}}(x)$ (the membership degree for the upper membership function).

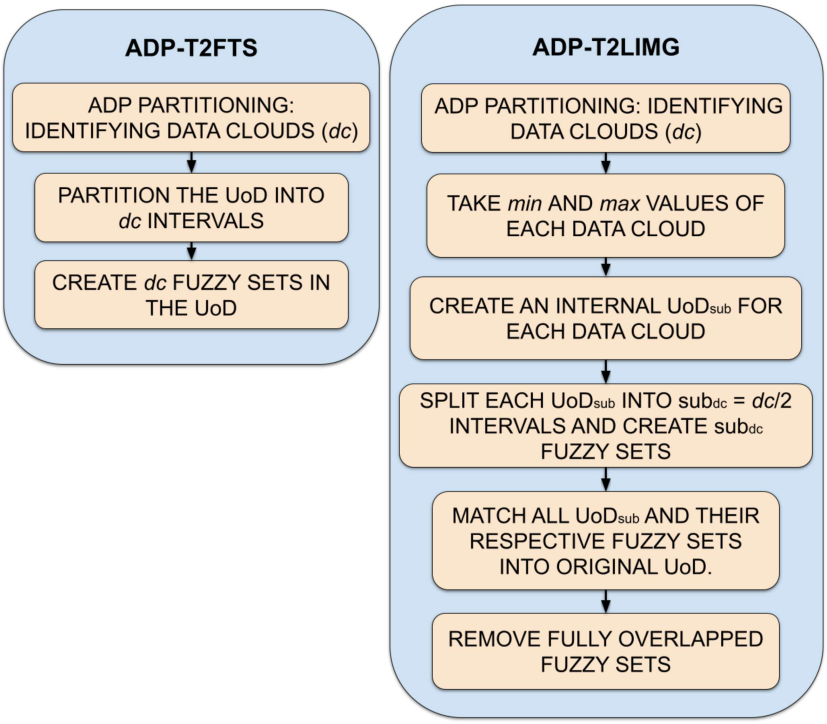

ADP-T2FTS
ADP PARTITIONING: IDENTIFYING DATA CLOUDS (*dc*)
PARTITION THE UoD INTO *dc* INTERVALS
CREATE *dc* FUZZY SETS IN THE UoD
ADP-T2LIMG
ADP PARTITIONING: IDENTIFYING DATA CLOUDS (*dc*)
TAKE *min* AND *max* VALUES OF EACH DATA CLOUD
CREATE AN INTERNAL UoDsub FOR EACH DATA CLOUD
SPLIT EACH UoDsub INTO subdc = *dc*/2 INTERVALS AND CREATE subdc FUZZY SETS
MATCH ALL UoDsub AND THEIR RESPECTIVE FUZZY SETS INTO ORIGINAL UoD.
REMOVE FULLY OVERLAPPED FUZZY SETS

Figure 20 – Procedures for partitioning of the UoD in the previous and currently proposed model

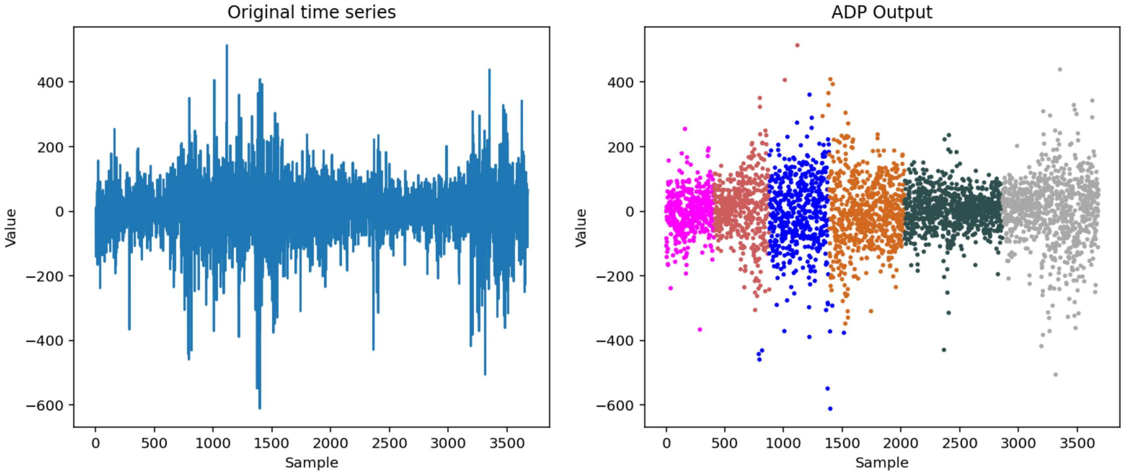


Figure 21 – Original time series and the respective ADP output.

Once the training set is fuzzified, temporal relationships between consecutive fuzzy values are analyzed to establish Fuzzy Logical Relationships (FLRs). For instance, if a data point at time $t-1$ is fuzzified to the fuzzy set $\widetilde{A}_4$ and another data point at time t is fuzzified to $\widetilde{A}_7$, the following logical relationship is formed: $\widetilde{A}_4 \rightarrow \widetilde{A}_7$. These relationships are then grouped into Fuzzy Logical Relationship Groups (FLRGs), simplifying the forecasting process by consolidating all possible future states for a given current state. For example, the FLRG $\widetilde{A}_4 \rightarrow \widetilde{A}_2, \widetilde{A}_4, \widetilde{A}_7$ indicates that the current state $\widetilde{A}_4$ has historically been followed by the states $\widetilde{A}_2$, $\widetilde{A}_4$, and $\widetilde{A}_7$ in the dataset.

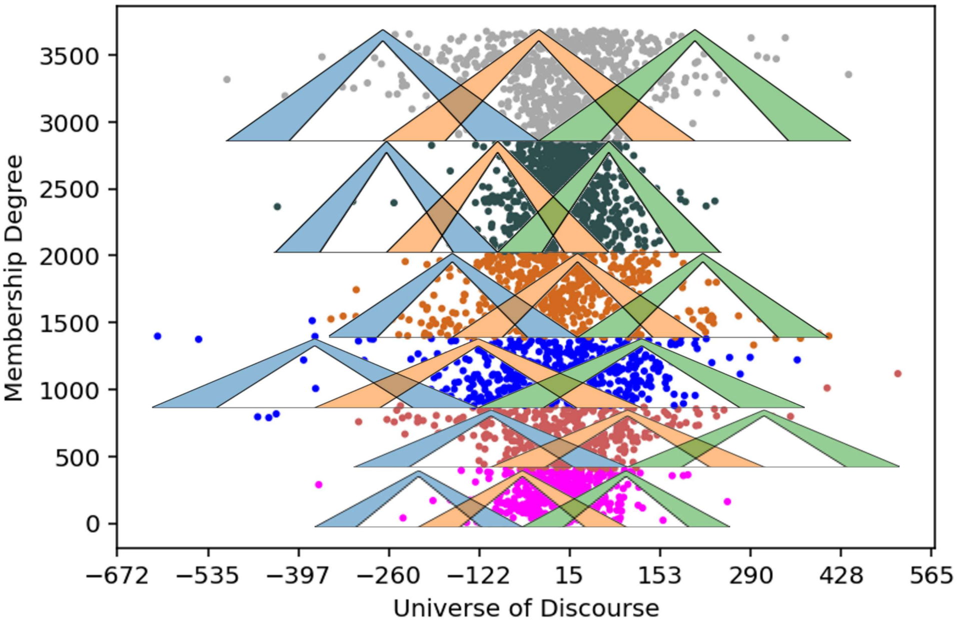


Figure 22 – Internal UoD and fuzzy sets for each data cloud.

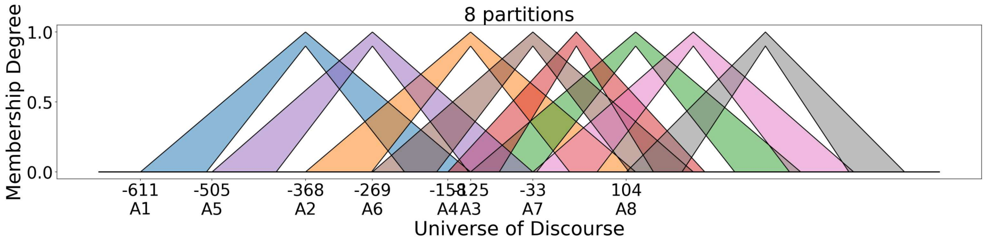


Figure 23 – Final ADP-T2LIMG partitioning.

5.2 Forecasting

The forecasting phase in FTS procedure involves predicting future values for all data points in the test dataset.

5.2.1 Fuzzification

All data points $y(t) \in Y, t-1, ..., n$ from the time series test dataset are fuzzified and their fuzzy values are calculated in respect to the linguistic variable $\tilde{A}$.

5.2.2 Fuzzy Logical Relationships matching

After all test samples are fuzzified, their corresponding fuzzy logical relationship group (FLRG) can be retrieved based on their fuzzy state (i.e. fuzzy set). An FLRG contains all possible future states for a given current state of the test sample, learned from the training historical data during the training phase.

5.2.3 Type-reduction and defuzzification

The FLRG is used to compute the forecast fuzzy set. As the fuzzy sets overlap, antecedent and consequent processing are carried out using the *min* and *max* operators. After that, a single IT2FS is generated through the process of Aggregation. This IT2FS is then reduced to a T1FS in the type-reduction process, which is executed using the EIASC procedure (Wu; Nie, 2011). Centroid defuzzification is then performed to convert the forecast type-reduced fuzzy set into a crisp numerical value.

5.2.4 Evaluating Forecast Accuracy

In this step, the model performance is evaluated by comparing the predicted values with the actual observations.

5.3 Computational Experiments

ADP-T2LIMG is compared to the other two proposed models in this thesis: SODA-T2FTS and ADP-T2FTS. The datasets used were the TAIEX time series and the solar time series from SONDA[1] - Sistema de Organização Nacional de Dados Ambientais (Brazilian National System of Environmental Data Organization). Results shown in Table 19 highlight ADP-T2LIMG's performance compared to the other two methods for both financial and energy datasets, also showing ADP-T2LIMG's longer execution time.

Table 19 – RMSE values for TAIEX and SONDA datasets.

Dataset	Model	Gridsize	Order	Part.	RMSE	Time(s)
TAIEX	ADP-T2FTS	4	1	615	62.63	43.19
	SODA-T2FTS	10	1	50	79.53	85.4
	ADP-T2LIMG	5	1	232	**59.86**	580.16
SONDA	ADP-T2FTS	5	1	132	104.23	132.07
	SODA-T2FTS	3	1	6	140.11	56.03
	ADP-T2LIMG	3	1	543	**99.76**	1112.87

Five univariate time series from two datasets, the PJM hourly energy consumption dataset, and the Global Energy Forecasting Competition 2012 (GEFCom 2012) dataset were then used to verify the performance of the proposed model against advanced FTS methods. For the PJM dataset[2], 8,000 hourly energy consumption samples were selected from Ohio/Kentucky's Duke Energy (DEOK) and American Electric Power (AEP). The GEFCom 2012 dataset, a well-known source available on Kaggle[3], includes data from 20 zones, with hourly energy consumption recorded between January 1, 2004, and July 7,

[1] <http://sonda.ccst.inpe.br/>

[2] <https://www.kaggle.com/datasets/robikscube/hourly-energy-consumption>

[3] https://www.kaggle.com/c/global-energy-forecasting-competition-2012-load-forecasting

2008. 8,000 samples from zones 1, 2, and 3 were utilized in this investigation. Table 20 presents a preliminary statistical analysis of the datasets. Energy datasets are relevant in forecasting literature at the moment as accurate forecasts may help balance supply and demand, prevent energy shortages, and support the efficient integration of renewable energy into existing energy systems, ultimately contributing to a greener and more reliable energy scenario.

Table 20 – Descriptive statistics for PJM and GEFCom datasets

Parameter	**AEP**	**DEOK**	**Zone 1**	**Zone 2**	**Zone 3**
Data points	8000	8000	8000	8000	8000
Mean	15567.67	3056.21	17477.74	163801.62	176747.22
Std. deviation	2594.37	640.06	5189.69	36241.20	36123.22
Minimum	9823	1870	8688	8672	9060
1st quartile	13703	2261	13644.25	135660.00	147385.50
Median	15494	3021	16445	161617.50	170498.55
3rd quartile	17339.50	3663	19364.25	194861.50	206905.15
Maximum	24015	5445	39584	270013	291344

Forecast accuracy is evaluated using the sliding-window cross-validation technique. For this case, the method employs a window size of 2000 instances, shifted by 200 instances (10% of the window size) along the time series. Thus, each dataset is divided into 30 windows such that 80% of each window is used for training and the remaining 20% for testing. Root Mean Square Error (RMSE) is applied to calculate the accuracy of the model, which is calculated as the average of performance metrics across all 30 windows.

Table 21 showcases the forecasting accuracy of the proposed ADP-T2LIMG model alongside several baseline models, including ARIMA, Long Short-Term Memory (LSTM), Probabilistic Weighted FTS (PWFTS), Randomized High-Order Fuzzy Cognitive Maps (R-HFCM), Convolutional Neural Networks (CNN), and CNN-LSTM, as detailed in Orang et al. (2024). The best-performing model in each case is highlighted in bold for clarity.

Table 21 – Model performance comparison (RMSE)

Method	Zone 1	Zone 2	Zone 3	DEOK	AEP
R-HFCM	716.364	5531.027	5837.610	136.783	624.151
PWFTS	939.671	6826.440	7359.919	134.861	662.849
LSTM	4932.353	32610.370	36592.800	2862.968	679.392
CNN	611.859	4685.848	5193.894	94.662	377.457
CNN-LSTM	650.442	5847.702	5935.384	95.021	421.745
ARIMA	5274.916	42207.760	39253.070	603.279	3005.837
ADP-T2LIMG	**487.466**	**3654.046**	**4056.288**	**71.865**	**316.445**

Based on this table, ADP-T2LIMG consistently achieves the lowest RMSE values across all regions, outperforming all other models under the same sliding window method and settings. This highlights its robustness in handling complex time series data compared to R-HFCM and PWFTS, which show moderate performance, and models like LSTM, CNN, and CNN-LSTM, which perform better but still lack ADP-T2LIMG's precision. The poor performance of ARIMA further underscores the limitations of traditional statistical methods. The superior results of ADP-T2LIMG can be attributed to its deeper exploration of ADP data clouds and their properties, which significantly enhances forecasting accuracy.

To statistically compare the performance of the models, a Kruskal-Wallis test was conducted with a confidence level of $\alpha = 0.05$, considering the average RMSE. In this analysis, the null hypothesis (H_0) asserts that the average RMSE errors are equal across all methods, while the alternative hypothesis (H_1) indicates that at least one of the averages is different. If H_0 is rejected, a post hoc test is applied to compare the differences between each pair of means, utilizing the Wilcoxon test in this study.

Table 22 – Statistical Ranking of The Forecasting Methods

Dataset	R-HFCM	PWFTS	LSTM	CNN	CNN-LSTM	ARIMA	ADP-T2LIMG
AEP	4	5	6	2	3	6	**1**
DEOK	4	5	7	2	2	6	**1**
Zone1	4	5	6	2	3	6	**1**
Zone2	3	5	6	2	4	6	**1**
Zone3	3	5	6	2	3	6	**1**
Avg. Rank	3.6	5	6.2	2	3	6	**1**

Since H_0 is rejected, the Wilcoxon test was conducted, with the results presented in Table 22. This table showcases the statistical ranking of the proposed ADP-T2LIMG against other competing methods across five datasets. ADP-T2LIMG consistently outperforms the other models, achieving the top rank (rank 1) across all datasets. CNN also shows strong performance, securing the second-best rank in most cases, followed closely by CNN-LSTM. Meanwhile, R-HFCM generally ranks in the middle, with a slightly better performance than PWFTS, LSTM, and ARIMA, which typically occupy the lower ranks. The average ranking highlights ADP-T2LIMG as the top performer with an average rank of 1, while CNN and CNN-LSTM follow with average ranks of 2 and 3, respectively. In contrast, LSTM and ARIMA rank lower, reflecting less competitive performance across the datasets.

As a final comparison, ADP-T2LIMG was used to forecast the TAIEX time series and was compared in Table 23 to state-of-the-art forecasting models. The models used in the comparison were some of the ones already presented and used in Section 4.3.4: Cai *et al.*'s model (Cai et al., 2015), CNFS-ARIMA (Li; Chiang, 2012), LMNF-D/I (Peng et al., 2015), T2NFS (Liu; Yeh; Lee, 2012), Jiang *et al.*'s model (Jiang et al., 2018) and ADP-T2FTS. The RMSE was the comparison metric chosen for this experiment, and the results for all models can be seen in Table 23, with the best-performing ones highlighted in bold.

Table 23 – Forecasting performance for the TAIEX time series (per year)

Methods	**1999**	**2000**	**2001**	**2002**	**2003**	**2004**
Cai *et al.*'s method	102.22	131.53	112.59	60.33	51.54	50.33
CNFS-ARIMA	100.01	122.58	115.82	64.34	57.69	55.56
LMNF-D/I	92.19	123.33	116.73	63.66	50.90	53.63
T2NFS	97.30	120.90	103.84	58.10	51	51.73
Jiang's *et al.*'s model	97.61	119.73	113.26	67.39	54.95	56.21
ADP-T2FTS	91.37	**118.81**	100.91	65.91	57.81	**45.88**
ADP-T2LIMG	**76.37**	127.49	**67.47**	**49.78**	**33.07**	45.94

5.3.1 Noise Response

This time, non-gaussian noise was added to datasets for testing the resilience of the proposed models when dealing with increasing noise rates in the original data. Laplacian noise is a type of non-Gaussian noise that follows a double-exponential distribution, introducing abrupt variations that simulate real-world uncertainties, such as sudden market shifts or unexpected external influences. Similar to carried out in Section 3.4.6, noise was added to the original time series according to the Signal-to-Noise Ratio (SNR). A high SNR corresponds to a clear signal (low noise) and a low SNR corresponds to a noisy signal (high amounts of noise). Datasets used in this experiment were the Yearly Sunspot number from 1700 to 1987 dataset ((Kim; Chung, 2005)) also used in Section 4.3.4 and the NREL[4] solar energy time series, obtained from the United States National Renewable Energy Laboratory, with added SNR values ranging from 50% to 20%. Tables 24 and 25 show the RMSE values obtained when SODA-T2FTS, ADP-T2FTS and ADP-T2LIMG were used to forecast these time series with different noise ratios. It can be observed that ADP-T2LIMG obtained the lowest RMSE values in most SNR scenarios, handling well the increasing noise rate.

Table 24 – RMSE values and their standard deviation for Sunspots time series with added laplacian noise.

SNR (%)	ADP-T2FTS					SODA-T2FTS					ADP-T2LIMG				
	Grid.	Order	Part.	RMSE		Grid.	Order	Part.	RMSE		Grid.	Order	Part.	RMSE	
				AVG	STD				AVG	STD				AVG	STD
50	1	1	5	22.73	2.71	1	1	4	22.45	2.87	1	1	5	**21.02**	**3.29**
45	1	1	5	22.74	2.75	1	1	4	22.61	2.87	1	1	8	**21.16**	**3.17**
40	1	1	5	22.50	2.21	1	1	4	**22.40**	**2.25**	5	1	867	24.42	2.18
35	4	1	75	23.94	2.65	1	1	4	23.52	3.59	1	1	7	**21.01**	**3.28**
30	1	1	5	22.30	3.08	1	1	4	22.24	2.97	1	1	5	**21.05**	**2.57**
25	1	1	5	23.81	2.25	1	1	4	23.73	2.36	1	1	5	**21.98**	**1.26**
20	1	1	4	25.59	0.57	1	1	4	25.59	0.57	1	1	4	**22.83**	**2.13**

[4] https://midcdmz.nrel.gov/apps/html.pl?ite=oahugrid;page=instrumentsGH10

Table 25 – RMSE values and their standard deviation for NREL time series with added laplacian noise.

SNR (%)	ADP-T2FTS					SODA-T2FTS					ADP-T2LIMG				
	Grid.	Order	Part.	RMSE		Grid.	Order	Part.	RMSE		Grid.	Order	Part.	RMSE	
				AVG	STD				AVG	STD				AVG	STD
50	5	1	163	98,44	3,19	1	1	2	149,42	2,90	3	1	760	**90,99**	**2,44**
45	5	1	166	99,14	5,76	2	1	4	149,33	2,42	3	1	608	**98,06**	**3,87**
40	4	1	124	98,51	3,43	2	1	3	147,37	2,20	3	1	653	**93,12**	**4,88**
35	4	1	121	98,03	3,51	5	1	13	151,11	3,03	3	1	668	**93,53**	**3,80**
30	4	1	123	**98,97**	**5,95**	5	1	12	150,57	7,12	3	1	610	99,92	5,71
25	5	1	196	99,75	4,03	5	1	12	146,33	4,00	3	1	597	**99,62**	**4,74**
20	5	1	205	117,35	5,62	1	1	1	151,21	0,94	3	1	643	**115,79**	**5,98**

In conclusion, ADP-T2LIMG was proposed as an update to ADP-T2FTS, where more information from ADP's output would be utilized to more accurately create the fuzzy sets in the forecasting model, aligning them more closely with the data distribution. The results obtained indicate that ADP was being sub-utilized, as there was information unused that could improve the model's performance. In this approach, a lot more information provided by ADP was used to split the Universe of Discourse and create fuzzy sets respecting data distribution. ADP-T2LIMG outperformed SODA-T2FTS, ADP-T2FTS AND advanced FTS models from the literature in datasets from different domains, including in noise response experiments, showing it could be a viable option for time series forecasting.

6 Conclusions

This research introduced three novel forecasting models — SODA-T2FTS, ADP-T2FTS, and ADP-T2LIMG — each designed to improve fuzzy set based time series (FTS) forecasting through advanced autonomous data partitioning algorithms and type-2 fuzzy logic. These models enhance traditional forecasting approaches by leveraging data-driven partitioning techniques, reducing reliance on predefined parameters, and improving the handling of epistemic uncertainty in time series data and model design.

In Chapter 3 SODA-T2FTS was presented, which utilizes the Self-Organized Direction Aware (SODA) partitioning algorithm,to determine the optimal partitions. Experimental results showed that SODA-T2FTS improves forecast accuracy while maintaining model interpretability, also outperforming conventional type-1 FTS models and statistical approaches in forecasting accuracy, interpretability and noise resistance.

The Autonomous Data Partitioning algorithm (ADP) improves SODA's ability to partition datasets autonomously, and hence was used in Chapter 4 for the creation of ADP-T2FTS, a forecasting model that demonstrates superior performance compared SODA-T2FTS and to state-of-the-art type-1 and type-2 FTS models, as well as neural networks and regression-based approaches.

ADP-T2LIMG further refines the ADP-T2FTS approach in Chapter 5 by extracting deeper structural information from ADP-generated data clouds, improving the definition of fuzzy sets. This enhanced partitioning method enables the model to more accurately capture data patterns, leading to the lowest RMSE values across multiple datasets, outperforming traditional machine learning and statistical models such as LSTM, CNN, and ARIMA.

Results have shown that data-driven approaches significantly enhance FTS forecasting by improving partitioning accuracy, reducing subjectivity, and increasing adaptability to different datasets. Traditional FTS models rely on user-defined partitioning strategies, such as equal-length and clustering-based methods, which introduce epistemic uncertainty due to subjective parameter selection. By contrast, the proposed models—SODA-T2FTS, ADP-T2FTS, and ADP-T2LIMG—eliminate the need for manual tuning by employing autonomous, data-driven partitioning algorithms, making them more robust and adaptive to complex time series data.

Among the data-driven methods explored, the Self-Organized Direction Aware (SODA) and Autonomous Data Partitioning (ADP) algorithms provided a powerful alternative to traditional approaches. SODA's self-organizing structure enabled effective data-aware partitioning, while ADP further improved partitioning flexibility by adapting to data cloud structures using rank operators. The models developed using these methods consistently outperformed conventional FTS models in terms of forecast accuracy, interpretability, and resilience to noise, proving the superiority of data-driven partitioning over heuristic methods.

Additionally, the integration of interval type-2 fuzzy logic greatly contributed to enhancing forecasting performance. Traditional type-1 FTS models suffer from limited uncertainty representation, making them less effective when dealing with noisy or highly variable time series. In contrast, type-2 fuzzy sets provide an additional degree of freedom, allowing for better uncertainty modeling through the footprint of uncertainty (FOU). This capability enabled the proposed models to capture variations more effectively, resulting in improved prediction accuracy without a significant increase in model complexity. The experiments confirmed that type-2 fuzzy logic consistently produced more stable and reliable forecasts for the financial and energy datasets applied.

As a direct outcome of this research, an open-source Python library, pyT2FTS[1], was developed to provide a flexible and accessible framework for fuzzy set based time series forecasting. This library integrates the data-driven partitioning techniques (SODA and ADP) and interval type-2 fuzzy logic methodologies explored in this study, offering users the ability to easily modify key parameters and build forecasting models. By automating complex tasks such as data partitioning, fuzzification, and rule extraction, pyT2FTS simplifies the development of accurate and robust forecasting models while maintaining a high degree of interpretability.

6.1 Summary of methods limitations

This research focused exclusively on rule-based, time-invariant models, which limits the applicability of the proposed methods primarily to stationary or well-behaved time series, particularly those that have undergone appropriate pre-processing. Besides that, all proposed models are designed for univariate time series forecasting. This may limit their effectiveness in domains where variables exhibit strong interdependencies, and a multivariate framework would yield better performance

Regarding model complexity, although ADP-T2LIMG achieved the best forecasting accuracy among all tested models, it also presented significantly higher execution time compared to SODA-T2FTS and ADP-T2FTS. This increased computational cost may limit its applicability in real-time or resource-constrained environments.

Moreover, SODA-T2FTS, ADP-T2FTS, and ADP-T2LIMG, are designed for batch processing, where the entire dataset must be available before training and testing can occur. This limits their applicability in real-time or streaming contexts where data arrives continuously and models must adapt incrementally.

[1] <https://github.com/arthurcaio92/pyT2FTS>

6.2 Future Research Directions

SODA-T2FTS, ADP-T2FTS, and ADP-T2LIMG demonstrate clear advantages in data-driven partitioning and type-2 FTS forecasting. To further build upon these contributions, future research could explore:

- **Extension to multivariate fuzzy set based time series:** The current models focus on univariate time series forecasting, where predictions are made based on a single variable. However, many real-world applications, such as economic forecasting, climate modeling, and stock market analysis, require the consideration of multiple interdependent variables. A natural next step would be to extend the proposed models to multivariate FTS models, incorporating intervariable dependencies to enhance forecasting accuracy.
- **General type-2 fuzzy sets for Enhanced Uncertainty Modeling:** While interval type-2 fuzzy sets have been proven effective in handling uncertainty, they still rely on fixed lower and upper membership function bounds, which may not fully capture higher-order uncertainty variations. Future research could explore general type-2 fuzzy sets (GT2FS), which allow for more flexible and adaptive uncertainty representation. Implementing GT2FS in FTS models could provide even greater robustness in noisy, imprecise, and highly uncertain environments.
- **Exploring hybrid approaches:** Recent advancements in artificial intelligence have highlighted the power of hybrid models that integrate fuzzy logic with machine learning techniques, thus, future research could investigate adapting the proposed models to these machine learning techniques: (a) Deep learning-assisted type-2 FTS models, where neural networks (e.g., LSTM, CNN, or Transformers) help refine fuzzy membership functions or optimize rule extraction; (b) Neuro-fuzzy architectures that combine data-driven partitioning with adaptive learning mechanisms to improve forecasting accuracy. (c) Fuzzy ensemble methods, leveraging multiple FTS models to improve prediction reliability.
- **Adaptive and real-time fuzzy set based time series Forecasting:** Most existing FTS models, including those proposed in this study, rely on static training data. However, many real-world forecasting problems require real-time learning and adaptation. Future research could focus on online learning techniques that allow FTS models to continuously update based on new data, dynamically adjusting the rule base as patterns change over time.

REFERENCES

ABHISHEKH; GAUTAM, S. S.; SINGH, S. A refined weighted method for forecasting based on type 2 fuzzy time series. *International Journal of Modelling and Simulation*, Taylor & Francis, v. 38, n. 3, p. 180–188, 2018.

AGUIAR, E. P. de et al. A new model to distinguish railhead defects based on set-membership type-2 fuzzy logic system. *International Journal of Fuzzy Systems*, Springer, v. 23, p. 1057–1069, 2021.

ALADAG, C. H. et al. Forecasting in high order fuzzy times series by using neural networks to define fuzzy relations. *Expert Systems with Applications*, Elsevier, v. 36, n. 3, p. 4228–4231, 2009.

ALADAG, C. H. et al. Fuzzy lagged variable selection in fuzzy time series with genetic algorithms. *Applied Soft Computing*, Elsevier, v. 22, p. 465–473, 2014.

ALYOUSIFI, Y.; OTHMAN, M.; ALMOHAMMEDI, A. A. A novel stochastic fuzzy time series forecasting model based on a new partition method. *IEEE Access*, IEEE, 2021.

ALYOUSIFI, Y. et al. Markov weighted fuzzy time-series model based on an optimum partition method for forecasting air pollution. *International Journal of Fuzzy Systems*, Springer, v. 22, p. 1468–1486, 2020.

ANGELOV, P. Outside the box: an alternative data analytics framework. *Journal of Automation Mobile Robotics and Intelligent Systems*, v. 8, n. 2, p. 29–35, 2014.

ANGELOV, P.; GU, X.; KANGIN, D. Empirical data analytics. *International Journal of Intelligent Systems*, v. 32, n. 12, p. 1261–1284, 2017.

ANGELOV, P.; YAGER, R. A new type of simplified fuzzy rule-based system. *International Journal of General Systems*, Taylor & Francis, v. 41, n. 2, p. 163–185, 2012.

ANGELOV, P. P.; GU, X. Applications of autonomous data partitioning. *Empirical Approach to Machine Learning*, Springer, p. 261–276, 2019.

ANGELOV, P. P.; GU, X.; PRíNCIPE, J. C. A generalized methodology for data analysis. *IEEE Transactions on Cybernetics*, v. 48, n. 10, p. 2981–2993, 2018.

ASKARI, S.; MONTAZERIN, N. A high-order multi-variable fuzzy time series forecasting algorithm based on fuzzy clustering. *Expert Systems with Applications*, Elsevier, v. 42, n. 4, p. 2121–2135, 2015.

ASKARI, S.; MONTAZERIN, N.; ZARANDI, M. F. Modeling energy flow in natural gas networks using time series disaggregation and fuzzy systems tuned by particle swarm optimization. *Applied Soft Computing*, Elsevier, v. 92, p. 106332, 2020.

BAJESTANI, N. S.; ZARE, A. Forecasting taiex using improved type 2 fuzzy time series. *Expert Systems with Applications*, Elsevier, v. 38, n. 5, p. 5816–5821, 2011.

BISWAS, A.; DE, A. K. A unified method of defuzzification for type-2 fuzzy numbers with its application to multiobjective decision making. *Granular Computing*, Springer, v. 3, n. 4, p. 301–318, 2018.

BLEI, D. M.; JORDAN, M. I. Variational inference for dirichlet process mixtures. *Bayesian analysis*, International Society for Bayesian Analysis, v. 1, n. 1, p. 121–143, 2006.

BOSE, M.; MALI, K. Designing fuzzy time series forecasting models: A survey. *International Journal of Approximate Reasoning*, Elsevier, v. 111, p. 78–99, 2019.

BOX, G. E. et al. *Time series analysis: forecasting and control.* [S.l.]: John Wiley & Sons, 2015.

CAI, Q. et al. A novel stock forecasting model based on fuzzy time series and genetic algorithm. *Procedia Computer Science*, Elsevier, v. 18, p. 1155–1162, 2013.

CAI, Q. et al. A new fuzzy time series forecasting model combined with ant colony optimization and auto-regression. *Knowledge-Based Systems*, Elsevier, v. 74, p. 61–68, 2015.

CASTILLO, O. et al. Type-2 fuzzy logic: theory and applications. In: IEEE. *2007 IEEE international conference on granular computing (GRC 2007).* [S.l.], 2007. p. 145–145.

CHATFIELD, C. Prediction intervals for time-series forecasting. In: *Principles of forecasting.* [S.l.]: Springer, 2001. p. 475–494.

CHEN, M.-Y.; CHEN, B.-T. A hybrid fuzzy time series model based on granular computing for stock price forecasting. *Information Sciences*, Elsevier, v. 294, p. 227–241, 2015.

CHEN, S.-M. et al. Forecasting enrollments based on fuzzy time series. *Fuzzy sets and systems*, v. 81, n. 3, p. 311–319, 1996.

CHEN, S.-M.; PHUONG, B. D. H. Fuzzy time series forecasting based on optimal partitions of intervals and optimal weighting vectors. *Knowledge-Based Systems*, Elsevier, v. 118, p. 204–216, 2017.

CHEN, S.-M.; TANUWIJAYA, K. Fuzzy forecasting based on high-order fuzzy logical relationships and automatic clustering techniques. *Expert Systems with Applications*, Elsevier, v. 38, n. 12, p. 15425–15437, 2011.

CHEN, S.-M.; WANG, N.-Y. Fuzzy forecasting based on fuzzy-trend logical relationship groups. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, IEEE, v. 40, n. 5, p. 1343–1358, 2010.

CHENG, C.-H.; CHANG, J.-R.; YEH, C.-A. Entropy-based and trapezoid fuzzification-based fuzzy time series approaches for forecasting it project cost. *Technological Forecasting and Social Change*, Elsevier, v. 73, n. 5, p. 524–542, 2006.

CHENG, C.-H.; CHEN, T.-L.; CHIANG, C.-H. Trend-weighted fuzzy time-series model for taiex forecasting. In: NEURIPS. *Proceedings of the International Conference on Neural Information Processing.* Vancouver, 2006. p. 469–477.

CHENG, C.-H. et al. Fuzzy time-series based on adaptive expectation model for taiex forecasting. *Expert systems with applications*, Elsevier, v. 34, n. 2, p. 1126–1132, 2008.

CHENG, S.-H.; CHEN, S.-M.; JIAN, W.-S. Fuzzy time series forecasting based on fuzzy logical relationships and similarity measures. *Information Sciences*, Elsevier, v. 327, p. 272–287, 2016.

DEC, G. et al. Forecasting models of daily energy generation by pv panels using fuzzy logic. *Energies*, Multidisciplinary Digital Publishing Institute, v. 14, n. 6, p. 1676, 2021.

DICKEY, D. A.; FULLER, W. A. Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American statistical association*, Taylor & Francis, v. 74, n. 366a, p. 427–431, 1979.

DINCER, N. G.; AKKUŞ, Ö. A new fuzzy time series model based on robust clustering for forecasting of air pollution. *Ecological Informatics*, Elsevier, v. 43, p. 157–164, 2018.

DIXIT, A.; JAIN, S. Intuitionistic fuzzy time series forecasting method for non-stationary time series data with suitable number of clusters and different window size for fuzzy rule generation. *Information Sciences*, Elsevier, v. 623, p. 132–145, 2023.

DURAN, K.; BERNAL, H.; MELGAREJO, M. Improved iterative algorithm for computing the generalized centroid of an interval type-2 fuzzy set. In: NAFIPS. *Proceedings of the Annual Meeting of the North American Fuzzy Information Processing Society*. Piscataway, 2008. p. 1–5.

ESTER, M. et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: *kdd*. [S.l.: s.n.], 1996. v. 96, n. 34, p. 226–231.

EWBANK, H. et al. Sustainable resource management in a supply chain: a methodological proposal combining zero-inflated fuzzy time series and clustering techniques. *Journal of Enterprise Information Management*, Emerald Publishing Limited, v. 33, n. 5, p. 1059–1076, 2020.

GAO, R.; DURU, O. Parsimonious fuzzy time series modelling. *Expert Systems with Applications*, Elsevier, v. 156, p. 113447, 2020.

GU, X.; ANGELOV, P. P. Highly interpretable hierarchical deep rule-based classifier. *Applied Soft Computing*, Elsevier, v. 92, p. 106310, 2020.

GU, X. et al. Self-organised direction aware data partitioning algorithm. *Information Sciences*, v. 423, p. 80–95, 2018.

GU, X.; ANGELOV, P. P.; PRÍNCIPE, J. C. A method for autonomous data partitioning. *Information Sciences*, Elsevier, v. 460, p. 65–82, 2018.

GUO, H.; PEDRYCZ, W.; LIU, X. Fuzzy time series forecasting based on axiomatic fuzzy set theory. *Neural Computing and Applications*, Springer, v. 31, n. 8, p. 3921–3932, 2019.

GUPTA, K. K.; KUMAR, S. Hesitant probabilistic fuzzy set based time series forecasting method. *Granular Computing*, Springer, v. 4, n. 4, p. 739–758, 2019.

HIEU, N. D.; HO, N. C.; LAN, V. N. Enrollment forecasting based on linguistic time series. *Journal of Computer Science and Cybernetics*, v. 36, n. 2, p. 119–137, 2020.

HUA, Q. et al. A robust fuzzy time series forecasting method based on multi-partition and outlier detection. *Chinese Journal of Electronics*, IET, v. 28, n. 5, p. 899–905, 2019.

HUANG, S. et al. Trapezoidal type-2 fuzzy inference system with tensor unfolding structure learning method. *Neurocomputing*, Elsevier, v. 473, p. 54–67, 2022.

HUARNG, K. Effective lengths of intervals to improve forecasting in fuzzy time series. *Fuzzy sets and systems*, Elsevier, v. 123, n. 3, p. 387–394, 2001.

HUARNG, K.; YU, H.-K. A type 2 fuzzy time series model for stock index forecasting. *Physica A: Statistical Mechanics and its Applications*, Elsevier, v. 353, p. 445–462, 2005.

HUARNG, K.; YU, T. H.-K. The application of neural networks to forecast fuzzy time series. *Physica A: Statistical mechanics and its applications*, Elsevier, v. 363, n. 2, p. 481–491, 2006.

HUARNG, K.; YU, T. H.-K. Ratio-based lengths of intervals to improve fuzzy time series forecasting. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, IEEE, v. 36, n. 2, p. 328–340, 2006.

HWANG, J.-R.; CHEN, S.-M.; LEE, C.-H. Handling forecasting problems using fuzzy time series. *Fuzzy sets and systems*, Elsevier, v. 100, n. 1-3, p. 217–228, 1998.

ISLAM, M.; HOSSAIN, M.; HAQUE, I. Mathematical comparison of defuzzification of fuzzy logic controller for intelligence air conditioning system. *Int. J. Sci. Res. in Mathematical and Statistical Sciences Vol*, v. 8, n. 2, 2021.

ISMAIL, Z.; EFENDI, R. Enrollment forecasting based on modified weight fuzzy time series. *Journal of Artificial Intelligence*, v. 4, n. 1, p. 110–118, 2011.

IZAKIAN, H.; PEDRYCZ, W.; JAMAL, I. Fuzzy clustering of time series data using dynamic time warping distance. *Engineering Applications of Artificial Intelligence*, v. 39, p. 235–244, 2015. ISSN 0952-1976.

JIANG, J.-A. et al. An interval type-2 fuzzy logic system for stock index forecasting based on fuzzy time series and a fuzzy logical relationship map. *IEEE Access*, IEEE, v. 6, p. 69107–69119, 2018.

JIANG, P. et al. A novel high-order weighted fuzzy time series model and its application in nonlinear time series prediction. *Applied Soft Computing*, Elsevier, v. 55, p. 44–62, 2017.

KARNIK, N. N.; MENDEL, J. M. Centroid of a type-2 fuzzy set. *information SCiences*, Elsevier, v. 132, n. 1-4, p. 195–220, 2001.

KIM, M.-S.; CHUNG, C.-S. Sunspot time series prediction using parallel-structure fuzzy system. In: SPRINGER. *Fuzzy Systems and Knowledge Discovery: Second International Conference, FSKD 2005, Changsha, China, August 27-29, 2005, Proceedings, Part II 2*. [S.l.], 2005. p. 731–741.

KIUREGHIAN, A. D.; DITLEVSEN, O. Aleatory or epistemic? does it matter? *Structural safety*, Elsevier, v. 31, n. 2, p. 105–112, 2009.

KOHONEN, T. The self-organizing map. *Neurocomputing*, Elsevier, v. 21, n. 1-3, p. 1–6, 1998.

KOO, J. W. et al. Prediction of air pollution index in kuala lumpur using fuzzy time series and statistical models. *Air Quality, Atmosphere & Health*, Springer, v. 13, n. 1, p. 77–88, 2020.

KUMAR, N.; SUSAN, S. Particle swarm optimization of partitions and fuzzy order for fuzzy time series forecasting of covid-19. *Applied Soft Computing*, Elsevier, v. 110, p. 107611, 2021.

KUO, I.-H. et al. An improved method for forecasting enrollments based on fuzzy time series and particle swarm optimization. *Expert Systems with applications*, Elsevier, v. 36, n. 3, p. 6108–6117, 2009.

LAHMIRI, S. A predictive system integrating intrinsic mode functions, artificial neural networks, and genetic algorithms for forecasting s&p500 intra-day data. *Intelligent Systems in Accounting, Finance and Management*, Wiley Online Library, v. 27, n. 2, p. 55–65, 2020.

LEE, L.-W. et al. Handling forecasting problems based on two-factors high-order fuzzy time series. *IEEE Transactions on fuzzy Systems*, IEEE, v. 14, n. 3, p. 468–477, 2006.

LI, C.; CHIANG, T.-W. Complex neurofuzzy arima forecasting—a new approach using complex fuzzy sets. *IEEE Transactions on Fuzzy Systems*, IEEE, v. 21, n. 3, p. 567–584, 2012.

LI, S.-T.; CHENG, Y.-C.; LIN, S.-Y. A fcm-based deterministic forecasting model for fuzzy time series. *Computers & Mathematics with Applications*, Elsevier, v. 56, n. 12, p. 3052–3063, 2008.

LIANG, Q.; MENDEL, J. M. Interval type-2 fuzzy logic systems: theory and design. *IEEE Transactions on Fuzzy systems*, IEEE, v. 8, n. 5, p. 535–550, 2000.

LIU, C.-F.; YEH, C.-Y.; LEE, S.-J. Application of type-2 neuro-fuzzy modeling in stock price prediction. *Applied Soft Computing*, Elsevier, v. 12, n. 4, p. 1348–1358, 2012.

LIU, J.-W. et al. Adaptive-expectation based multi-attribute fts model for forecasting taiex. *Computers & mathematics with applications*, Elsevier, v. 59, n. 2, p. 795–802, 2010.

LIU, X.; LIN, Y. New efficient algorithms for the centroid of an interval type-2 fuzzy set. *Information Sciences*, Elsevier, v. 570, p. 468–486, 2021.

LIU, X.; WAN, S. Combinatorial iterative algorithms for computing the centroid of an interval type-2 fuzzy set. *IEEE Transactions on Fuzzy Systems*, IEEE, v. 28, n. 4, p. 607–617, 2019.

MAHMOUDI, M. R.; MAHMOUDI, M.; PAK, A. On comparing, classifying and clustering several dependent regression models. *Journal of Statistical Computation and Simulation*, Taylor & Francis, v. 89, n. 12, p. 2280–2292, 2019.

MCCULLOCH, J.; WAGNER, C. On the choice of similarity measures for type-2 fuzzy sets. *Information Sciences*, Elsevier, v. 510, p. 135–154, 2020.

MENDEL, J. et al. *Introduction to type-2 fuzzy logic control: theory and applications.* [S.l.]: John Wiley & Sons, 2014.

Mendel, J. M. Type-2 fuzzy sets and systems: an overview. *IEEE Computational Intelligence Magazine*, v. 2, n. 1, p. 20–29, 2007.

MENDEL, J. M.; LIU, F. Super-exponential convergence of the karnik–mendel algorithms for computing the centroid of an interval type-2 fuzzy set. *IEEE Transactions on Fuzzy Systems*, IEEE, v. 15, n. 2, p. 309–320, 2007.

MITTAL, K. et al. A comprehensive review on type 2 fuzzy logic applications: Past, present and future. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 95, p. 103916, 2020.

MUKMININ, U. S. et al. Fuzzy time series based on frequency density-based partitioning and k-means clustering for forecasting exchange rate. *Journal of Physics: Conference Series*, v. 1943, n. 1, p. 012119, 2021.

MUSIKASUWAN, S.; SEPTIARINI, T. W. Forecasting indonesia stock exchange (idx) composite using fuzzy time series methods. *International Journal of Innovative Science and Research Technology*, v. 5, n. 3, p. 1349–1356, 2020.

NG, C. T.; LEE, W.; LEE, Y. Logical and test consistency in pairwise multiple comparisons. *Journal of Statistical Planning and Inference*, Elsevier, v. 206, p. 145–162, 2020.

OKABE, A. et al. Spatial tessellations: concepts and applications of voronoi diagrams. *Gale OneFile: Health and Medicine*, v. 260, n. 5111, p. 1170–1173, 1993.

ORANG, O. et al. A large reservoir computing forecasting method based on randomized fuzzy cognitive maps. In: *2024 IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS)*. [S.l.: s.n.], 2024. p. 1–8.

ORANG, O. et al. Solar energy forecasting with fuzzy time series using high-order fuzzy cognitive maps. In: IEEE. *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. [S.l.], 2020. p. 1–8.

PANIGRAHI, S.; BEHERA, H. A study on leading machine learning techniques for high order fuzzy time series forecasting. *Engineering Applications of Artificial Intelligence*, v. 87, p. 103245, 2020. ISSN 0952-1976. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S095219761930226X>>.

PATKI, V. K. et al. Fuzzy system modeling for forecasting water quality index in municipal distribution system. *Urban Water Journal*, Taylor & Francis, v. 12, n. 2, p. 89–110, 2015.

PATTANAYAK, R. M.; BEHERA, H. S.; PANIGRAHI, S. A novel probabilistic intuitionistic fuzzy set based model for high order fuzzy time series forecasting. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 99, p. 104136, 2021.

PENG, H.-W. et al. Time series forecasting with a neuro-fuzzy modeling scheme. *Applied Soft Computing*, Elsevier, v. 32, p. 481–493, 2015.

PHAMTOAN, D.; VOTHIHANG, N.; PHAMTHI, B. Improving forecasting model for fuzzy time series using the self-updating clustering and bi-directional long short term memory algorithm. *Expert Systems with Applications*, v. 241, p. 122767, 2024. ISSN 0957-4174. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0957417423032694>>.

PINTO, A. C. V. et al. Self-organised direction aware data partitioning for type-2 fuzzy time series prediction. In: IEEE. *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. [S.l.], 2021. p. 1–6.

SADAEI, H. J. *Improved models in Fuzzy Time Series for forecasting*. Tese (Doutorado) — Universiti Teknologi Malaysia, 2013.

SADAEI, H. J. et al. Short-term load forecasting using a hybrid model with a refined exponentially weighted fuzzy time series and an improved harmony search. *International Journal of Electrical Power & Energy Systems*, Elsevier, v. 62, p. 118–129, 2014.

SEVERIANO, C. A. et al. Very short-term solar forecasting using fuzzy time series. In: FUZZ-IEEE. *Proceedings of the IEEE international conference on fuzzy systems*. Naples, 2017. p. 1–6.

SHAFII, N. H. et al. Fuzzy time series and geometric brownian motion in forecasting stock prices in bursa malaysia. *Jurnal Intelek*, v. 14, n. 2, p. 240–250, 2019.

SHUKLA, A. K.; MUHURI, P. K. Big-data clustering with interval type-2 fuzzy uncertainty modeling in gene expression datasets. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 77, p. 268–282, 2019.

SHUMWAY, R. H.; STOFFER, D. S. *Time series analysis and its applications: with R examples*. [S.l.]: Springer, 2017.

SILVA, G. R. et al. A fuzzy data reduction cluster method based on boundary information for large datasets. *Neural Computing and Applications*, Springer, v. 32, n. 24, p. 18059–18068, 2020.

SILVA, P. C. et al. Distributed evolutionary hyperparameter optimization for fuzzy time series. *IEEE Transactions on Network and Service Management*, IEEE, v. 17, n. 3, p. 1309–1321, 2020.

SILVA, P. C. d. L. et al. Probabilistic forecasting with fuzzy time series. *IEEE Transactions on Fuzzy Systems*, IEEE, v. 28, n. 8, p. 1771–1784, 2019.

SINGH, P. A brief review of modeling approaches based on fuzzy time series. *International Journal of Machine Learning and Cybernetics*, Springer, v. 8, n. 2, p. 397–420, 2017.

SONG, Q.; CHISSOM, B. S. Fuzzy time series and its models. *Fuzzy sets and systems*, v. 54, n. 3, p. 269–277, 1993.

SONG, X. et al. A new fuzzy c-means clustering-based time series segmentation approach and its application on tunnel boring machine analysis. *Mechanical Systems and Signal Processing*, Elsevier, v. 133, p. 106279, 2019.

SOUZA, P.; PONCE, H.; LUGHOFER, E. Evolving fuzzy neural hydrocarbon networks: A model based on organic compounds. *Knowledge-Based Systems*, Elsevier, v. 203, p. 106099, 2020.

SOUZA, P. V. C.; LUGHOFER, E. Online active learning for an evolving fuzzy neural classifier based on data density and specificity. *Neurocomputing*, Elsevier, v. 512, p. 269–286, 2022.

SUN, B. et al. Prediction of stock index futures prices based on fuzzy sets and multivariate fuzzy time series. *Neurocomputing*, Elsevier, v. 151, p. 1528–1536, 2015.

SVETLAKOV, M.; HODASHINSKY, I. Clustering-based rule generation methods for fuzzy classifier using autonomous data partitioning algorithm. In: IOP PUBLISHING. *Journal of Physics: Conference Series*. [S.l.], 2021. v. 1989, n. 1, p. 012032.

TAK, N. Type-1 possibilistic fuzzy forecasting functions. *Journal of Computational and Applied Mathematics*, Elsevier, v. 370, p. 112653, 2020.

TINH, N. V. Forecasting of covid-19 confirmed cases in vietnam using fuzzy time series model combined with particle swarm optimization. *Comput. Res. Progr. Appl. Sci. Eng.*, v. 6, n. 2, p. 114–120, 2020.

TSAI, M.-C.; CHENG, C.-H.; TSAI, M.-I. A multifactor fuzzy time-series fitting model for forecasting the stock index. *Symmetry*, Multidisciplinary Digital Publishing Institute, v. 11, n. 12, p. 1474, 2019.

VAMITHA, V. A different approach on fuzzy time series forecasting model. *Materials Today: Proceedings*, Elsevier, v. 37, p. 125–128, 2021.

WANG, J.; LI, H.; LU, H. Application of a novel early warning system based on fuzzy time series in urban air quality forecasting in china. *Applied Soft Computing*, Elsevier, v. 71, p. 783–799, 2018.

WANG, J.; LI, H.; LU, H. Application of a novel early warning system based on fuzzy time series in urban air quality forecasting in china. *Applied Soft Computing*, v. 71, p. 783–799, 2018. ISSN 1568-4946.

WU, D.; NIE, M. Comparison and practical implementation of type-reduction algorithms for type-2 fuzzy sets and systems. In: FUZZ-IEEE. *Proceedings of the IEEE International Conference on Fuzzy Systems*. Taipei, 2011. p. 2131–2138.

WU, H. et al. Stock index forecasting: A new fuzzy time series forecasting method. *Journal of Forecasting*, Wiley Online Library, v. 40, n. 4, p. 653–666, 2021.

YE, F. et al. A novel forecasting method based on multi-order fuzzy time series and technical analysis. *Information Sciences*, Elsevier, v. 367, p. 41–57, 2016.

YOLCU, U. et al. A new approach for determining the length of intervals for fuzzy time series. *Applied Soft Computing*, Elsevier, v. 9, n. 2, p. 647–651, 2009.

YU, H.-K. Weighted fuzzy time series models for taiex forecasting. *Physica A: Statistical Mechanics and its Applications*, Elsevier, v. 349, n. 3-4, p. 609–624, 2005.

YU, T. H.-K.; HUARNG, K.-H. A bivariate fuzzy time series model to forecast the taiex. *Expert Systems with Applications*, Elsevier, v. 34, n. 4, p. 2945–2952, 2008.

ZADEH, L. A. The concept of a linguistic variable and its application to approximate reasoning-iii. *Information sciences*, Elsevier, v. 9, n. 1, p. 43–80, 1975.

ZADEH, L. A. Fuzzy sets. In: *Fuzzy sets, fuzzy logic, and fuzzy systems: selected papers by Lotfi A Zadeh*. [S.l.]: World Scientific, 1996. p. 394–432.

ZHAN, T. et al. Differential convolutional fuzzy time series forecasting. *IEEE Transactions on Fuzzy Systems*, v. 32, n. 3, p. 831–845, 2024.

ZHANG, E.; WANG, D.; LI, H. A comprehensive high order type 2 fuzzy time series forecasting model. In: CCDC. *Proceedings of the Chinese Control and Decision Conference*. Yinchuan, 2016. p. 6681–6686.

ZHANG, Y. et al. A novel fuzzy time series forecasting model based on multiple linear regression and time series clustering. *Mathematical Problems in Engineering*, Hindawi, v. 2020, 2020.

ZHANG, Z.; ZHU, Q. et al. Fuzzy time series forecasting based on k-means clustering. *Open Journal of Applied Sciences*, v. 2, n. 4, p. 100–103, 2012.

ZHAO, A.; GAO, J.; GUAN, H. Forecasting model for stock market based on probabilistic linguistic logical relationship and distance measurement. *Symmetry*, Multidisciplinary Digital Publishing Institute, v. 12, n. 6, p. 954, 2020.