Pontifícia Universidade Católica
do Rio de Janeiro

David Souza Pinto

Exploring new methods to perform Bagging with Exponential Smoothing

Dissertação de Mestrado

Dissertation presented to the Programa de Pós-graduação em Engenharia de Produção of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia de Produção.

Advisor : Prof. Fernando Luiz Cyrino Oliveira
Co-advisor: Dr. Tiago Mendes Dantas

Rio de Janeiro
September 2020

Pontifícia Universidade Católica
do Rio de Janeiro

David Souza Pinto

Exploring new methods to perform Bagging with Exponential Smoothing

Dissertation presented to the Programa de Pós-graduação em Engenharia de Produção of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia de Produção. Approved by the Examination Committee.

Prof. Fernando Luiz Cyrino Oliveira
Advisor
Departamento de Engenharia Industrial – PUC-Rio

Dr. Tiago Mendes Dantas
Co-advisor
Fundação Instituto Brasileiro de Geografia e Estatística – IBGE

Prof. Paula Medina Maçaira Louro
Departamento de Engenharia Industrial – PUC-Rio

Prof. Ana Paula Barbosa Sobral
Universidade Federal Fluminense – UFF

Rio de Janeiro, September 28th, 2020

David Souza Pinto

Graduado em Engenharia de Produção na Universidade Federal Fluminense (UFF) em 2018.

Graduated in Production Engineering at the Federal Fluminense University in 2018.

Bibliographic data

Pinto, David Souza

Exploring new methods to perform Bagging with Exponential Smoothing / David Souza Pinto; advisor: Fernando Luiz Cyrino Oliveira; co-advisor: Tiago Mendes Dantas. – Rio de janeiro: PUC-Rio, Departamento de Engenharia Industrial, 2020.

v., 75 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Industrial.

Inclui bibliografia

1. Engenharia de Produção – Teses. 2. Séries Temporais;. 3. Bagging;. 4. Clustering.. I. Oliveira, Fernando Luiz Cyrino. II. Dantas, Tiago Mendes. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Industrial. IV. Título.

CDD: 658.5

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) [134451/2018-0].

Abstract

Pinto, David Souza; Oliveira, Fernando Luiz Cyrino (Advisor); Dantas, Tiago Mendes (Co-Advisor). **Exploring new methods to perform Bagging with Exponential Smoothing**. Rio de Janeiro, 2020. 75p. Dissertação de Mestrado – Departamento de Engenharia Industrial, Pontifícia Universidade Católica do Rio de Janeiro.

Exponential smoothing methods are flexible procedures for univariate time series forecasting, developed in the 1960's. Most recent developments based on these models use bagging to improve forecast quality. One of these implementations, **BaggedETS**, developed in 2016, brought improvements in forecast quality and is distributed through the **forecast** package for R. A posterior implementation, **BaggedClusterETS**, adds clustering and validation steps to address the covariance effect associated with bagging. The proposal resulted in further accuracy improvements. This work delves into three extensions of the aforementioned methods: the first studies the effects of the maximum entropy bootstrap on the **BaggedETS**. The second explores different dissimilarity measures to construct the clusters in **BaggedClusterETS**. The third studies a simplified version of **BaggedClusterETS**, where the validation and selection steps are removed, and using only the medoids for bagging. To test these proposals, 21 time series from civil aviation and energy consumption were used.

Keywords

Time Series; Bagging; Clustering.

Resumo

Pinto, David Souza; Oliveira, Fernando Luiz Cyrino; Dantas, Tiago Mendes. **Explorando novos métodos para realizar Bagging com Amortecimento Exponencial**. Rio de Janeiro, 2020. 75p. Dissertação de Mestrado – Departamento de Engenharia Industrial, Pontifícia Universidade Católica do Rio de Janeiro.

Métodos de amortecimento exponencial são formulações versáteis para a previsão de séries temporais univariadas, desenvolvidas na década de 1960. Modelos mais recentes têm feito uso do *bagging* para melhorar a qualidade das previsões. Um destes, o **BaggedETS**, desenvolvido em 2016, trouxe melhorias na qualidade de previsão e está disponível na biblioteca **forecast** para **R**. Uma proposta posterior, **BaggedClusterETS**, adicionou uma etapa de *clustering* e validação para tratar o efeito da covariância associada ao uso do *bagging*, resultando em ganhos adicionais de performance. Este trabalho explora três extensões dos métodos supracitados e seus efeitos: o primeiro estuda os efeitos do *maximum entropy bootstrap* na realização do **BaggedETS**. O segundo explora diferentes medidas de dissimilaridade para construir os clusters do **BaggedClusterETS**. O terceiro emprega uma versão simplificada do **BaggedClusterETS**, removendo as etapas de validação e seleção, empregando apenas os medóides para realizar o *bagging*. Para testar estas propostas, 21 séries temporais da aviação civil e demanda energética foram empregadas.

Palavras-chave

Séries Temporais; Bagging; Clustering.

Table of contents

List of figures

List of tables

List of symbols

ANAC	*Agência Nacional de Aviação Civil*
AR	*Autoregressive*
ARIMA	*Autoregressive Integrated Moving Average*
BITRE	*Bureau of Infrastructure, Transport and Regional Economics*
BTS	*Bureau of Transportation Statistic*
CBB	*Circular block bootstrap*
ETS	*Error, Trend and Seasonality*
IID	*Independently and identically distributed*
LPB	*Linear process bootstrap*
MA	*Moving Average*
MAE	*Mean Absolute Error*
MASE	*Mean Absolute Scaled Error*
MBB	*Moving block bootstrap*
NBB	*Non-overlapping block bootstrap*
MEB	*Maximum entropy bootstrap*
MPE	*Mean percentage error*
MSFE	*Mean squared forecast error*
MSE	*Mean squared error*
OECD	*Organisation for Economic Co-operation and Development*
PAM	*Partitioning around medoids*
SB	*Sieve bootstrap*
SBB	*Stationary block bootstrap*
STL	*Seasonal and Trend decomposition using Loess*
sMAPE	*symmetric Mean Absolute Percentage Error*

No poem is intended for the reader,
no picture for the beholder,
no symphony for the listener.

Walter Benjamin, *Illuminations: Essays and Reflections.*

1
Introduction

1.1
Context and motivation

Time series can be defined as an ordered collection of random variables, indexed by the time t when they were obtained [1]. Such data is ubiquitous, and applications can be found in the most varied fields such as economics, social sciences, industry, and medical sciences to name a few. Forecasting models for time series data can play a critical role in strategic decision-making processes in a variety of business operations, from energy to supply chain and logistics. On that account, these models ought to be accurate and efficient to be serviceable. When forecasting errors are excessive, it can translate into either missed opportunities or unnecessary operational costs – this is critical when considering industries such as aviation or energy, where even small changes can scale costs up or down. Thus, better forecasts can lead to better planning, which has a cascading effect on business operations and their quality [2, 3, 4].

One way of improving forecasting quality is to employ bagging (bootstrap aggregation) as proposed by Breiman almost 25 years ago [5]. The method decreases predictor error through the generation of variations of a dataset via bootstrap. The aggregation comes from the application of a reducing function such as the mean to the ensemble. This method has been applied in a time series setting for a variety of domains – works published from 2004 to 2018 showcase applications to demographic data, financial time series, and wind speed forecasts [6]. An article on forecasts for aviation transporation highlights how bagging led to a significant improvement in forecasting quality [3].

Later, bagging was combined with time series decomposition to improve forecasting quality. The works of Cordeiro and Neves [7] and later Bergmeir, Hyndman and Benítez [8] make use of bagging, albeit their internals differ considerably. The latter, **Bagged.BLD.MBB.ETS** – which employs a combination of Box-Cox transformation, STL decomposition, moving block bootstrap, and exponential smoothing – has displayed the capability to improve forecasting quality when compared to a simple exponential smoothing model, and is readily available through the **forecast** package for the **R** programming language [9]. To answer the question of why such method works, research on the topic highlights that bagging tackles the three sources of uncertainty – namely that which arises from the model, parameter and data itself [10].

Dantas and Oliveira discussed the effect of covariance on forecast quality, and add a clustering step is order to address this issue [6]. This effect arises due to the correlation between each of the bootstrap innovations generated for the ensemble, which contributes to the Mean Squared Forecast Error (MSFE). For the study, the authors used one clustering technique, with one clustering validity index, the average silhouette information criterion. The method displayed an increase in forecast quality for monthly data.

A review of the literature revealed that there is a gap when it comes to studies concerning the intersection of time series forecasting, bagging and clustering. Although there have been some advances in the field [11], work on single series forecasting seems to be lacking – in fact, this literature review has indicated that, so far as it is possible to say, there are no publications that directly continue the research started by Dantas and Oliveira. For this study, variations based on the design proposed by Dantas and Oliveira are considered (see [6]), alongside a different bootstrapping method is used for the model proposed by Bergmeir, Hyndman, and Benítez [8].

1.2 Research objectives and questions

The central research objective for this project is the assembly of new forecasting tools to increase prediction accuracy. Secondary objectives include the introduction of different methods, followed by an appropriate performance evaluation on real-world data.

The following questions are asked:

1. Do any of the proposed methods lead to an improvement in forecasting quality, when compared to current methods and algorithms?
2. Do these methods introduce any additional computational overhead?

Since a clustering step is included in the methods explored in this study, any form of improvement can come either of two sources: 1. How bootstraps are generated; or 2. How bootstrapped series are clustered, the number of clusters created, and how these series are selected. This research takes the work of [6] as a starting point and reference to explore whether there can be any further improvements to the proposed method.

1.3 Research classification

The research can be classified as applied, quantitative, experimental, and bibliographic [12].

Applied research the aim of which is to solve a specific problem, i.e. what methods can be used to improve time series forecasting quality. Time series forecasting and analysis tools are routinely deployed in a variety of fields and applications – e.g. economics, energy planning, healthcare and hydrology studies. As long as data is readily available, researchers and practitioners can either apply available methods or tailor new ones to the characteristics of a given domain.

Quantitative research which involves the effects that different numerical methods have on forecasting quality for different series. There is a keen interest in ranking and evaluating these models in relation to one another with specific, measurable criteria. Two models can be compared with each other using specific error metrics, and for larger experiments, it is possible to test whether the difference between any given number of models is statistically significant.

Experimental research this follows naturally from the previous category since testing is required in order to evaluate whether any of the proposed models work adequately. Given that experiments are conducted with the help of a programming language, research can be reproduced by third parties, given that the scripts, datasets and random seed are available.

Bibliographic research this is the enabling component for the previous categories. A literature review is required to understand most recent research and identify relevant tools and their functions. This step is also vital to identify any research opportunities.

1.4 Structure

In addition to this introduction, this dissertation also includes the following chapters: Chapter 2 reviews the literature on exponential smoothing, bootstrapping, bagging, and clustering. Queries on scientific databases and bibliometrics were combined to understand the evolution of the topic throughout the years and to identify pivotal publications on the topic. Chapter 3 details the method employed for this study. It outlines how the tools for this project were established, and how experiments were built and analysed. Chapter 4 includes a description of the datasets used, the computational environment, and a presentation of the results obtained, as well as the blueprints to reproduce the experiments. The last chapter, Chapter 5, presents the conclusions and the relevant considerations that can be drawn from the previous chapter's experiments, alongside further research recommendations.

2
Literature review

This research uses the work of Dantas and Oliveira as its starting point. The authors combine bagging, clustering, and exponential smoothing, addressing a covariance effect that arises from generating bootstrap replicates [6] — a direct extension of the work by Bergmeir, Hyndman and Benítez [8]. Checking for references for future research, recommendations for expansions include incorporating weighting schemes when generating the aggregated forecast and introducing new decomposition and forecasting methods. To better understand what has been done since their article was published, what the relevant technologies are, and where to best place efforts, this chapter will focus on providing both the theoretical background of the relevant methods in conjunction with a review of the available literature on the topic. This is to ensure that research is within the bounds of what is relevant for a continuation of the aforementioned research.

This chapter is divided into four sections: the first reviews the **state of the art**, and provides an overview of what has been published before and after the work of Dantas and Oliveira [6], publication metrics, and an outline of relevant articles. The **Exponential smoothing** and **Resampling methods** sections analyse the relevant numerical methods that enable forecasting and bagging. **Clustering** is studied in a separate section due to its sprawling nature and how it is categorised in literature [13, 14, 15]. The section presents a definition of the clustering method, details how different methods are classified, what methods are available for time series, and the inner workings of the relevant methods for this research. A subsection on **cluster validity indices** details how to assess the quality of a clustering operation. The last section discusses the relevant **Implementations** found when reviewing the literature.

2.1
State of the art

2.1.1
Query results

In order to structure a strategy to query SCOPUS and Web of Science (WOS), the bibliography listed in [6] was used. The articles helped establish a general understanding of the methods used and what elements to look for in other works.

While some references were clearly available for bootstrap methods, bagging and forecasting, (e.g. [8, 5, 16]), for clustering, a literature review [14] was combined with books and other articles. This preliminary research on the topic aided in constructing an understanding of how to operate with clustering methods and what other options are available. Thus, for the queries, only partitioning methods were considered, as these were the only kind used by Dantas and Oliveira [6]. This classification includes methods such as k-means, k-medians, partitioning around medoids (PAM) or k-medoids, and fuzzy c-means [14, 17, 18].

Considering the three components of the method researched by Dantas and Oliveira [6] — i.e. time series forecasting, bagging and clustering — the following interactions were queries on both SCOPUS and WOS:

1. [Bagging] AND [Clustering],
2. Time series AND [Bagging],
3. Time series AND [Clustering],
4. Time series AND [Bagging] AND [Clustering],
5. Exponential smoothing AND [Bagging],
6. Exponential smoothing AND [Clustering],
7. Exponential smoothing AND [Bagging] AND [Clustering].

The tag [Bagging] contains the term **bagging** itself, alongside the expressions **bootstrap** and **maximum entropy bootstrap**. For [Clustering], the aforementioned partitioning clustering methods were considered: k-means, k-medians, k-medoids, partitioning around medoids, PAM. **Time series** and **Exponential smoothing** were the sole terms for the time series component — ARIMA models were not considered for this study. These queries were then merged into the following four sets, each representing a combination of keywords for a certain aspect of the research. Set 1 contains combinations for Time Series or Exponential Smoothing in relation to [Bagging] terms. Set 2 is the smallest of the sets, considering only the interactions between [Clustering] methods and [Bagging]. Set 3 relates to the works dealing with the combination of [Clustering], time series, and exponential smoothing. Set 4 combines the three previously described components into the queries. The list below provides an overview of the combinations used.

Set 1 (Time series OR Exponential smoothing) AND [Bagging].

Set 2 [Bagging] AND [Clustering].

Set 3 (Time series OR Exponential smoothing) AND [Clustering].

Set 4 (Time series OR Exponential smoothing) AND [Clustering] AND [Bagging].

For both aggregators, title, abstract and keywords were searched. For

WOS, an additional restriction was applied to the queries: only articles, books, book chapters, and proceeding papers were probed. Such filter was not applied to SCOPUS queries. The combination of results yielded around 5325 publications since 1978 for all these topics. Partial combination of the sets reveals a slightly large number of total articles (5518), suggesting there might be a crossover between some of the topics. Table 2.1 displays the breakdown for each of the sets. The first two columns indicate the total number of results for each aggregator, the third column represents the number of articles after merging the results, and the fourth represents the percentage of results in relation to all the sets. Sets 1 and 3 are the largest of the four, making up 88% of the publications on the topics. The combination of the three topics (i.e. the combination of time series/exponential smoothing with bagging and clustering) stands at a meagre 0.7% at the time of writing. This might indicate that the topic has not been researched extensively.

	SCOPUS	WOS	Merged	%
Set 1	1555	2283	2656	48.3
Set 2	201	430	587	10.7
Set 3	1941	1480	2221	40.3
Set 4	10	34	39	0.7
Total	3707	4227	5503	100.0

Table 2.1: Results by set

Set 1 Set 1, the largest in number of publications, covers a wide time frame: until 1994 there were 73 publications on the combined themes, with the number sharply rising each year, peaking at 222 articles and chapters in 2017. In 2020, at the time of writing, 39 items were published. An analysis of the 20 most cited articles in this set includes works on applications of the bootstrap to estimate standard errors, confidence intervals, and other measures. There are also applications for quality control, data snooping, hydrology, and data quality (i.e., missing values). Table 2.2 showcases that there are no applications combining the exponential smoothing and the maximum entropy bootstrap.

Keyword	SCOPUS	WOS
"time series" AND "bootstrap"	1473	2016
"time series" AND bagging	123	306
"time series" AND "maximum entropy bootstrap"	14	16
"exponential smoothing" AND bootstrap	21	25
"exponential smoothing" AND bagging	4	7
"exponential smoothing" AND "maximum entropy bootstrap"	0	0

Table 2.2: Queries for set 1

Set 2 Set 2 has an irregular publication pattern. After two publications in 1997, no works were published until 2000. But research seems to only have gained momentum from 2007 onwards. This is likely due to an increase in computation power, reaching the peak of publications in 2016 — a total of 85 abstracts were located in the queries for this year. Interest on the topics seems to have plateaued afterwards, and in 2020, at the time of writing, only four articles published. The most cited articles reveal applications for text classification, image processing, pattern discovery, and economics. Again, k-means seems to be a popular option (64 results on SCOPUS, 391 on WOS), with fuzzy c-means at a distant second (16 and 28, respectively). k-medians, k-medoids / partitioning around medoids / PAM have few publications available, as highlighted in table 2.3.

Keyword	SCOPUS	WOS
bagging AND k-means	64	391
bagging AND k-medians	0	2
bagging AND "PAM"	3	19
bagging AND "partitioning around medoids"	1	2
bagging AND "k-medoids"	3	9
bagging AND "fuzzy c-means"	16	28

Table 2.3: Queries for set 2

Set 3 Set 3 is the second largest, with 2233 articles on time series/exponential smoothing combined clustering. Again, the topic gained momentum from 2003 onwards, with 616 results — over a fourth of the total production — in 2018 and 2019 alone. As seen in table 2.4, searches indicate that k-means is a popular

clustering method for time series (963 and 753 results for SCOPUS and WOS, respectively), fuzzy c-means comes second (375 and 281 results). Least popular methods include k-medians, k-medoids, partitioning around medoids (PAM). When substituting 'time series' for 'exponential smoothing', no results are found for queries that contain k-medians OR k-medoids.

Keyword	SCOPUS	WOS
"Time series clustering"	644	438
"time series" AND k-means	963	753
"time series" AND k-medians	7	2
"time series" AND "k-medoids"	57	38
"time series" AND "PAM"	45	33
"time series" AND "partitioning around medoids"	26	21
"time series" AND "fuzzy c-means"	375	281
"exponential smoothing" AND k-means	13	9
"exponential smoothing" AND k-medians	0	0
"exponential smoothing" AND "k-medoids"	0	0
"exponential smoothing" AND "PAM"	2	2
"exponential smoothing" AND "partitioning around medoids"	1	1
"exponential smoothing" AND "fuzzy c-means"	10	5

Table 2.4: Queries for set 3

Set 4 Set 4 is the smallest of the sets, covering the intersection of time series/exponential smoothing with bagging and clustering. Research has been inconsistent. Studies were first published in 2001, then in 2004, and resuming in 2007 until 2009. From 2012, there have been yearly publications. This set contains applications on geospatial, medical, agriculture, environmental domains. Table 2.5 summarises the results for the different queries of this set.

Keyword	SCOPUS	WOS
bagging AND "time series" AND clusters	9	35
bagging AND "time series" AND k-means	4	7
bagging AND "time series" AND k-medians	0	0
bagging AND "time series" AND "PAM"	1	1
bagging AND "time series" AND "k-medoids"	0	1

Continues on tbe next page

Continued from the previous page

Keyword	SCOPUS	WOS
bagging AND "time series" AND "partitioning around medoids"	1	1
bagging AND "time series" AND "fuzzy c-means"	0	0
bagging AND "exponential smoothing" AND clusters	1	2
bagging AND "exponential smoothing" AND k-means	0	1
bagging AND "exponential smoothing" AND k-medians	0	0
bagging AND "exponential smoothing" AND "k-medoids"	0	0
bagging AND "exponential smoothing" AND "PAM"	1	1
bagging AND "exponential smoothing" AND "partitioning around medoids"	1	1
bagging AND "exponential smoothing" AND "fuzzy c-means"	0	0
bagging AND "exponential smoothing" AND weighting	0	1

Table 2.5: Queries for set 4

2.1.2 Applications in literature

When mining each of the sets for relevant materials, either theoretical references or examples of applications, a mix of materials for time series forecasting that include either a bagging or clustering component can be identified.

Cordeiro and Neves proposes a combination of exponential smoothing models and the sieve bootstrap to produce forecasts, and tested the model on M3 data [7]; Bergmeir, Hyndmand, and Benítez proposes a different model, a combination of Box-Cox transformation, STL decomposition, moving block bootstrap, and exponential smoothing, again tested on M3 data [8]. This latter model has seen application for aviation demand forecast [3], and energy consumption [4]. All these make a well founded argument for enhancements in forecasting accuracy — especially in sectors sectors where, due to their scale, even small gains in forecasting lead to significant financial impact. [19] take on tourism demand forecasts for Australia, working with a hybrid procedure to model future arrivals. When bagging the model, the authors noticed an increase in predictor accuracy.

Considering applications for the maximum entropy bootstrap developed by Vinod and López-de-Lacalle [20], there is the determination of the onset and withdrawal of the monsoon season [21]. An application for time series data related to the torque friction of rolling bearing, used the bootstrap method to combine the results of five different forecasting technique and were able to solve a prediction problem under conditions where the probability distribution is unknown and with changes to the trend [22]. The method was applied to a weather generator, and managed to achieve better performance in computational efficiency and extrapolation [23]. For a more theoretical application, the method enabled the extraction of descriptive statistics from functional time series, and enabled the estimation of the functional principal components [24].

Clustering was used in combination with time series forecasting methods to improve predictor quality for network traffic loads [25]. The authors noted accuracy gains over simple forecast methods when applying this integrated approach. A problem of daily peak load forecasting was tackled with a model that combines double seasonal Holt-Winters, neural networks, and fuzzy clustering — although complex, the authors noted the approach improved forecast quality [26]. A proposition where a density clustering, based on the forecasts at a horizon h, and where the bootstrap methods employed to produce the forecasts are non-parametric was also identified in literature. These methods are used as an alternative to the sieve bootstrap since they do not share its limitations [27, 28]. Time series clustering was employed to analyse data from European electricity markets, highlighting differences between northern and southern countries [29]. Authors applied hierarchical clustering methods with a convex combination of metrics.

Two reviews on the available clustering literature have been identified [14, 15]. Implementations for **R** were also located. The **cluster** library, whose algorithms are detailed on the book by Kaufman and Rousseeuw [30], the **TSclust** package [31], and the **dtwclust** package [32]. When looking for other, more general references, four books on the topic were identified. Of these, only one does not have a chapter dedicated to time series clustering [33], three include one chapter dedicated to time series clustering [17, 18, 34], and one of the books is completely dedicated to the topic [13]. The work by [33] has an emphasis on how to implement clustering methods with R, and includes code examples and relevant libraries.

At the time of writing, an implementation developed by [11] combines the three aforementioned elements (i.e., time series forecasting, bagging, and clustering), in which clustering and other ensemble methods were used to

improve forecasts for smart meter data. The internals are greatly different from the model proposed by [6].

In the method proposed by [11], first data is z-score normalised and regression coefficients are computed. These coefficients are then used for clustering. The number of clusters is validated by an internal CVI. Series within each cluster can be aggregated based on the consumers themselves or through simple aggregation. These aggregated series are then bootstrapped, to generate training data for forecasting methods. The forecast horizon for the authors is one day ahead. Ensemble learning methods are used to combine the multitude of forecasts produced for a given customer cluster into one single forecast. The final forecast is then computed through aggregation of the data generated in the previous step and compared with real consumption data. Smart meter data was collected from Australia, Ireland, and London [11]

For [6], B replicates of the train series are generated with MBB and forecast for a horizon h. The symmetric mean absolute percentage error (sMAPE) is calculated using the validation and forecast series. Series are then grouped into k clusters using PAM, where the Silhouette criterion is used to obtain the optimum number of clusters. From each cluster, a fraction of the series with the lowest errors is selected and an ETS model is then adjusted. Datasets from the M3 and CIF 2016 competitions were used, with yearly, quarterly, and monthly frequency, totalling 2901 different series [6].

It is important to emphasise that, while all the above-mentioned references are important contributions, none expands on the framework proposed by Dantas and Oliveira [6]. Table 2.6 lists publications (articles in journals, proceedings, books, and book chapters) that are relevant to this research. Materials are presented by their year of publication.

Reference	**Description**
Lahiri (2003) [35]	Reference for different bootstrap methods (Book).
Liao (2005) [14]	Literature review on clustering methods.
Cordeiro and Neves (2006) [36]	Review on bootstrap methods for time series forecasting.
Hyndman and Khandakar (2008) [37]	Implementation of automatic selection of exponential smoothing models with state space models.
Cordeiro and Neves (2009) [7]	Implementation of time series bagging, employs the sieve bootstrap.

Continues on the next page

Continued from the previous page

Reference	Description
Vinod and López-de-Lacalle (2009) [20]	Article for the **meboot** R library, implementing the maximum entropy bootstrap.
Kotsakos et al. (2014) [38]	Book chapter. Contains a section discussing some univariate and multivariate dissimilarity measures. For the clustering algorithms themselves, the authors list partitioning, hierarchical, density-based, and trajectory clustering as shape-based methods.
Montero and Vilar (2014) [31]	Article for the **TSclust** R library, implementing a variety of dissimilarity measures for time series clustering.
Reddy and Vinzamuri (2014) [39]	Book chapter. For partioning methods, the book covers the basic k-Means algorithm and eleven varitons, including Fuzzy c-means. For hiearchical clustering, five agglomerative and three divisive methods are discussed.
Aghabozorgi, Shirkhorshidi, and Wah (2015) [15]	Literature review on clustering methods, published a decade after the work of Liao (2005)[14].
Bergmeir, Hyndman, and Benítez (2016) [8]	Combines exponential smoothing, Loess-based decomposition, bootstrap and bagging to improve forecast quality.
Caiado, Maharaj, and D'Urso (2016) [40]	Book chapter showcasing existing work for model, observation, and feature-based methods, together with examples and applications.
D'Urso (2016) [41]	Book chapter focused on fuzzy clustering, commenting on its matheamtical and computational aspects, how to evaluate partitions, existing variants (prototype, distance, and objective function, and data feature-based implementations).

Continues on the next page

Continued from the previous page

Reference	Description
Dantas, Oliveira, and Repolho (2017) [3]	Application of **Bagged.BLD.MBB.ETS** to aviation data, leading to an improvement of forecasts.
Dantas and Oliveira (2018) [6]	Expands the model propsed by Bergmeir, Hyndman and Benítez (2016) [8]. Resulted in forecasting quality improvements for monthly data.
De Oliveira and Oliveira (2018) [4]	Application of bagging with ARIMA and exponential smoothing to electric consumption series.
Petropoulos, Hyndman, and Bergmeir (2018) [10]	Experiments on why bagging works. The work explore alternatives to bagging, alongside alternatives to the moving block bootstrap procedure for the **Bagged.BLD.MBB.ETS** method.
Hyndman and Athanasopoulos (2019) [16]	Time series forecasting reference (book).
Laurinec et al. (2019) [11]	Application of clustering with other machine learning methods to smart meter data.
Maharaj, D'Urso, and Caiado (2019) [13]	Time series clustering book, includes materials for partitioning, hierarchical, dynamic time warping, fuzzy and other clustering methods, alongside their mathematical formulations.
Martins, Lagarto, and Cardoso (2019) [29]	Application of hierarchical, agglomerative clustering to analyse electricity market prices. Also employs a convex combination of distances.
Sardá-Espinosa (2019) [32]	Article for the **dtwclust** R library, implementing a framework for time series clustering construction and evaluation.

Table 2.6: Selected references from literature

2.2

Exponential smoothing

Exponential smoothing methods were proposed in the late 1950s and early 1960s by Brown, Holt and Winters. These methods use weighted averages of the observed data to produce point forecasts, where more recent data points are more important than older ones. This is reflected in the mathematical modelling of these methods, where these weights decay exponentially [42, 43, 44, 16, 8, 3].

The first, and most simple method, called **single exponential smoothing** (SES) considers only a level l_t and a hyper-parameter α — i.e., the model does not take into consideration any kind of trend or seasonality pattern. The second, **Holt's linear method** models a level l_t and a trend b_t and two hyper-parameters, the α from SES, and a β for the trend [16]. **Holt-Winters' method** is an expansion of Holt's work, where the seasonal cycles of a time series are taken into account [44, 43]. Thus, there are three equations for computing the level l_t, the trend b_t, and seasonality s_t. These three equations also take three hyper-parameters α, β, γ. The models can either be additive or multiplicative. For Holt's linear and Holt-winter's models, a damping of the trend b_t can be also be enabled, by introducing a hyperparameter ϕ [16].

Besides these models, other combinations based on trend and seasonal components can be modelled as shown in table 2.7, for a total of 9 different exponential smoothing methods. Models in bold correspond to the methods described below, a relation between the trend-seasonal components shorthand is available in table 2.8.

	Seasonality		
Trend	None (N)	Additive (A)	Multiplicative (M)
None (N)	**(N, N)**	(N, A)	(N, M)
Additive (A)	**(A, N)**	**(A, A)**	**(A, M)**
Additive, damped (A_d)	**(A_d, N)**	(A_d, A)	**(A_d, M)**

Table 2.7: Exponential smoothing methods (Adapted from [16])

Shorthand	Name of the method
(N, N)	Simple exponential smoothing
(A, N)	Holt's linear method
(A_d, N)	Additive damped trend method
(A, A)	Additive Holt-Winters' method
(A, M)	Multiplicative Holt-Winters' method
(A_d, M)	Holt-Winters' damped method

Table 2.8: Shorthands for the named exponential smoothing methods (Adapted from [16])

The state space models framework extends the exponential smoothing family [37] — it uses the same terms for trend and seasonality use to categorise these methods, but considers that the forecasting error can be either additive or multiplicative, as shown in tables 2.9 and 2.10. While the error does not have an impact on the point forecast, their inclusion enables the generation of prediction intervals for the forecasts. The acronym for these models, **ETS**, stands for the **E**rror, **T**rend, **S**easonal components. [16, 37, 8]. The **ets()** function, available in the **forecast** library includes both options: users can specify the model they want to use, or let an embedded automatic procedure to set the model to the available data [9].

Trend	Seasonality N	A	M
N	$y_t = l_{t-1} + \varepsilon_t$ $l_t = l_{t-1} + \alpha\varepsilon_t$	$y_t = l_{t-1} + s_{t-m} + \varepsilon_t$ $l_t = l_{t-1} + \alpha\varepsilon_t$ $s_t = s_{t-m} + \gamma\varepsilon_t$	$y_t = l_{t-1}s_{t-m} + \varepsilon_t$ $l_t = l_{t-1} + \alpha\varepsilon_t/s_{t-1}$ $s_t = s_{t-m} + \gamma\varepsilon_t/l_{t-1}$
A	$y_t = l_{t-1} + b_{t-1} + \varepsilon_t$ $l_t = l_{t-1} + b_{t-1} + \alpha\varepsilon_t$ $b_t = b_{t-1} + \beta\varepsilon_t$	$y_t = l_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$ $l_t = l_{t-1} + b_{t-1} + \alpha\varepsilon_t$ $b_t = b_{t-1} + \beta\varepsilon_t$ $s_t = s_{t-m} + \gamma\varepsilon_t$	$y_t = (l_{t-1} + b_{t-1})s_{t-m} + \varepsilon_t$ $l_t = l_{t-1} + b_{t-1} + \alpha\varepsilon_t/s_{t-m}$ $b_t = b_{t-1} + \beta\varepsilon_t/s_{t-m}$ $s_t = s_{t-m} + \gamma\varepsilon_t/(l_{t-1} + b_{t-1})$
A_d	$y_t = l_{t-1} + \phi b_{t-1} + \varepsilon_t$ $l_t = l_{t-1} + \phi b_{t-1} + \alpha\varepsilon_t$ $b_t = \phi b_{t-1} + \beta\varepsilon_t$	$y_t = l_{t-1} + \phi b_{t-1} + s_{t-m} + \varepsilon_t$ $l_t = l_{t-1} + \phi b_{t-1} + \alpha\varepsilon_t$ $b_t = \phi b_{t-1} + \beta\varepsilon_t$ $s_t = s_{t-m} + \gamma\varepsilon_t$	$y_t = (l_{t-1} + \phi b_{t-1})s_{t-m} + \varepsilon_t$ $l_t = l_{t-1} + \phi b_{t-1} + \alpha\varepsilon_t/s_{t-m}$ $b_t = \phi b_{t-1} + \beta\varepsilon_t/s_{t-m}$ $s_t = s_{t-m} + \gamma\varepsilon_t/(l_{t-1} + \phi b_{t-1})$

Table 2.9: Additive error models (Adapted from [16])

Trend	Seasonality N	A	M
N	$y_t = l_{t-1}(1+\varepsilon_t)$ $l_t = l_{t-1}(1+\alpha\varepsilon_t)$	$y_t = (l_{t-1}+s_{t-m})(1+\varepsilon_t)$ $l_t = l_{t-1}+\alpha(l_{t-1}+s_{t-m})\varepsilon_t$ $s_t = s_{t-m}+\gamma(l_{t-1}+s_{t-m})\varepsilon_t$	$y_t = l_{t-1}s_{t-m}(1+\varepsilon_t)$ $l_t = l_{t-1}(1+\alpha\varepsilon_t)$ $s_t = s_{t-m}(1+\gamma\varepsilon_t)$
A	$y_t = (l_{t-1}+b_{t-1})(1+\varepsilon_t)$ $l_t = (l_{t-1}+b_{t-1})(1+\alpha\varepsilon_t)$ $b_t = b_{t-1}+\beta(l_{t-1}+b_{t-1})\varepsilon_t$	$y_t = (l_{t-1}+b_{t-1}+s_{t-m})(1+\varepsilon_t)$ $l_t = l_{t-1}+b_{t-1}+\alpha(l_{t-1}+b_{t-1}+s_{t-m})\varepsilon_t$ $b_t = b_{t-1}+\beta(l_{t-1}+b_{t-1}+s_{t-m})\varepsilon_t$ $s_t = s_{t-m}+\gamma(l_{t-1}+b_{t-1}+s_{t-m})\varepsilon_t$	$y_t = (l_{t-1}+b_{t-1})s_{t-m}(1+\varepsilon_t)$ $l_t = (l_{t-1}+b_{t-1})(1+\alpha\varepsilon_t)$ $b_t = b_{t-1}+\beta(l_{t-1}+b_{t-1})\varepsilon_t$ $s_t = s_{t-m}(1+\gamma\varepsilon_t)$
A_d	$y_t = (l_{t-1}+\phi b_{t-1})(1+\varepsilon_t)$ $l_t = (l_{t-1}+\phi b_{t-1})(1+\alpha\varepsilon_t)$ $b_t = \phi b_{t-1}+\beta(l_{t-1}+\phi b_{t-1})\varepsilon_t$	$y_t = (l_{t-1}+\phi b_{t-1}+s_{t-m})(1+\varepsilon_t)$ $l_t = l_{t-1}+\phi b_{t-1}+\alpha(l_{t-1}+\phi b_{t-1}+s_{t-m})\varepsilon_t$ $b_t = \phi b_{t-1}+\beta(l_{t-1}+\phi b_{t-1}+s_{t-m})\varepsilon_t$ $s_t = s_{t-m}+\gamma(l_{t-1}+\phi b_{t-1}+s_{t-m})\varepsilon_t$	$y_t = (l_{t-1}+\phi b_{t-1})s_{t-m}(1+\varepsilon_t)$ $l_t = (l_{t-1}+\phi b_{t-1})(1+\alpha\varepsilon_t)$ $b_t = \phi b_{t-1}+\beta(l_{t-1}+\phi b_{t-1})\varepsilon_t$ $s_t = s_{t-m}(1+\gamma\varepsilon_t)$

Table 2.10: Multiplicative ETS models (Adapted from [16])

2.3 Resampling methods

2.3.1 IID Bootstrap and Bagging

The bootstrap — also known as IID bootstrap –, introduced in 1979 by Efron [45], has been used as a tool to gauge statistical precision, and to measure the uncertainty that is linked to either an estimator or a statistical learning method. The method is used to estimate a parameter $\hat{\theta}$ for a random sample $\mathbf{x}$ with an unknown distribution F [46, 47, 48]. This computer-intensive method does not make assumptions about the underlying structure of the random process that produced the data. A non-exhaustive list of its uses includes: the computation of estimates for the variance, distribution function, critical values, and confidence intervals [35]. With regard to how it can be performed, B artificial data sets are generated by sampling the original data with replacement. Each of these replications has the same dimensions of original dataset [46]. This enables the tool to be flexible enough to be used in a wide variety of learning methods, some of which would be difficult to directly compute measures of variability or the software does not provide an automatic output [47].

Bagging — short for *bootstrap aggregating* — is an ensemble method closely associated with the bootstrap, used to reduce the variance of a statistical learning method. When applied to numeric data, after B bootstrap replicates are generated, data is aggregated by averaging the ensemble. However, it is possible to use other reducing functions such as the median or the trimmed mean [5, 47, 34, 8]. While the method in itself is simple, good performance can be attained with fairly unreliable data [49]. When working with time series data, there is experimental evidence that bagging is capable of handling the uncertainty that arises from the data itself, the forecasting model, and the parameter selection when dealing with time series forecasting [10], although modifications to the IID bootstrap need to be made in order to cater for the serial correlation present. With the relevant adjustments to the bootstrap procedure, application of bagging has resulted in increased forecasting accuracy [19, 3, 6, 4].

2.3.2 Bootstrap for dependent data

As noted in the literature, the IID bootstrap does not work properly for time series data since it does not take into account the dependence structure

of the observations, and when the data re-sampling occurs, this information is lost – i.e. the dependence structure is not preserved, failing to account for lag-covariance terms [35, 36]. A different approach is then required in order to satisfactorily treat dependent data. Research on re-sampling methods that preserve this structure have been conducted since the late 1980s, and an assortment of bootstrapping methods for dependent data has been developed. Examples of this include the moving block bootstrap, the circular block bootstrap, and the sieve bootstrap [35, 50, 36, 35, 51]. These methods employ different sampling strategies to safeguard the previously mentioned dependence structure, but most come with a common restriction — in order to be properly deployed, data must be stationary [35]. Hence, ancillary methods that can transform data from non-stationary to stationary might be required. A non-exhaustive description of re-sampling methods is detailed here, followed by an overview of decomposition methods that can provide the required.

Moving block bootstrap (MBB) Instead of sampling with reposition a single point n times, blocks of size l observations are re-sampled. The setup guarantees that a total of $n - l + 1$ overlapping blocks exist. The procedure makes it possible to preserve the autocorrelation structure within each block [35, 8]. The original IID bootstrap can be seen as a special case of the MBB when the block size l is equal to 1.

Non-overlapping block bootstrap (NBB) While blocks of size l are still used for this method, in the same fashion as MBB, the method does not allow for overlaps, resulting in distinct distributional properties when the samples are not sufficiently large [35].

Circular block bootstrap (CBB) An extension of MBB, where the series data is wrapped around a circle, resulting in additional blocks due to the geometry [36]. The method should theoretically be superior to MBB, since the latter under-samples the $l - 1$ first and $l - 1$ last observations, for a block size l [10]. Another property that arises from this formulation is that "the conditional expectation of the bootstrap sample mean under CBB equals the sample mean of the data [...], a property not shared by MBB or NBB" [35, p. 34].

Stationary Block Bootstrap (SBB) This method uses a random length for the block length l, distinguishing it from previous block-based methods (i.e. MBB, NBB, CBB), where the block size is a fixed value, defined before the method is run. An alternative implementation sees the method behaving as a series of binary trials when selecting values for the sample. A key property of the SBB is that the bootstrap innovations are stationary, conditional on the original dataset [35, 36]. For further details on

the implementation, check [35, 34-36] or [36, p. 1070].

Sieve bootstrap (SB) The procedure consist of adjusting an autoregressive model of order p, – an information criterion such as the AIC can be used to choose the adequate order. From this AR(p) model, the empirical distribution is extracted, and the residuals centred. The bootstrap replicates are constructed by sampling from the empirical distribution and plugging the values into the adjusted AR(p) model [35, 7]. While the model itself is parametric, where the parameters are approximated through the Yule-Walker equations, the residuals from the resulting model are IID [35, 7]. Concerning its properties, while a sieve can be chosen for its more accurate bootstrap estimator, especially when compared to the block-based bootstraps, due to the nature of the method, applicability is restricted to a reduced class of processes [35].

Linear process bootstrap (LPB) One possible alternative to the AR-based sieve would be to deploy a Moving Average (MA) one. This alternative would model a given time series Y_t with MA(q) processes, in the same manner an AR sieve would, by fitting increasingly high order MA(q) processes. There are complications to deliver this kind of implementation due to the computational requirements to adjust MA(q) models for large values of q [50]. An alternative would see the development of an estimator for the covariance matrix. By proposing a way to construct the covariance matrix, the authors have managed to generate MA processes without knowledge of its coefficients, and they note that its performance is similar to that of the "block bootstrap" [50].

As previously mentioned, these methods require stationary data in order to operate properly, otherwise statistical assumptions are violated. When it comes to real-world applications, these rarely produce stationary data [20]. To surmount this limitation, a decomposition procedure can be employed, where the series Y_t is disassembled into independent trend T_t, seasonality S_t and remainder R_t components. These procedures can be either additive or multiplicative, i.e. in order to recreate the series, the individual components are either added or multiplied together, as seen in the equations 2-1 and 2-2. Once the series is taken apart, the aforementioned bootstrapping methods can be applied to the remainder. Given how the remainder is constructed, as what is leftover from removing the trend and, if there is any, seasonality, it might still hold some serial correlation — which would invalidate the use of the IID bootstrap. Examples of decomposition procedures include the ARIMA-based X11-ARIMA, X12-ARIMA, and TRAMO-SEATS, and the Loess-based STL decomposition [52, 53]. STL decomposition has an advantage over X12-ARIMA

and TRAMO-SEATS, as it can be used for time series of any frequency (i.e., yearly, monthly, quarterly, etc.), where the first two can only be deployed for quarterly and monthly data [52, 53].

$$Y_t = T_t + S_t + R_t \tag{2-1}$$

$$Y_t = T_t \times S_t \times R_t \tag{2-2}$$

STL decomposition was designed to be simple and straightforward to use, while still being flexible when considering trend and seasonality. Authors also aimed to make it resilient to missing values and to anomalous behaviour in the data when generating trend and seasonality components [53]. It is important to note that for this method the obtained trend T_t is smoothed [16]. The method has a total of six parameters that have to be inputted in order to be used, and some of these can be tweaked or fine-tuned by a user, including the rate of change for the the seasonal component and the smoothness of the trend-cycle [16, 53]. Some disadvantages include not being able to deal with multiplicative decompositions — unless a log transformation is applied to the data, such as the Box-Cox transformation[1] —, and it does not automatically take into consideration trading day or other calendar variations [16]. While the procedure available in **R** has some automated settings for convenience, for some time series the method parameters may need to be tweaked in order to fit the structure of any given series [16]. This is done by adjusting the smoothing parameter for the seasonal component [53, p. 9].

A procedure combining the Box-Cox transformation, STL decomposition, Moving Block Bootstrap, and Exponential Smoothing was proposed by [8] under the moniker **Bagged.BLD.MBB.ETS**. The model was tested with 2829 times series from the M3 competition data set. Breaking them down by frequency, the authors employed 645 series with yearly frequency, 756 with quarterly, and 1428 with monthly. The proposed model delivered improved forecasts, especially on monthly data. The authors also noted that the size of the time series might have had an impact on the performance of the model [8].

Concerning the block length l for the block-based methods, there are some considerations to be made. While there are discussions on how to calculate the optimal block length l for the block-based methods presented above (MBB, CBB and SBB) [54], an alternative is considered. When working with **Bagged.BLD.MBB.ETS**, in order to specify the length l of the blocks,

[1]The Box-Cox transformation is also used to stabilise the series variance when it changes over time [1, 16], to approximate normality, or improve linearity [1].

the following heuristics was found in literature [8]: l is set to $l = 8$ for both yearly and quarterly data, and $l = 24$ for monthly data, in order to seize any seasonal behaviour in the data. Then, in order to ensure that the values from the original time series can be arranged in any position within the bootstrap replicates, $\lfloor n/l \rfloor + 2$ blocks are drawn; a random amount of points, between 0 and $l - 1$ are removed from the start of the series; after sampling the blocks, a number of points at the end are dropped until the replicate has the same length n of the original series. The CBB and LPB procedures were later considered for **Bagged.BLD.MBB.ETS**, but test results for these alternatives did not identify any significant impact on forecasting quality [10].

Considering that data needs to be stationary in order to use the MBB, and data from real world applications is a mixture of stationary and non-stationary processes, upholding the stationarity assumption can be problematic. The Maximum Entropy Bootstrap (MEB), proposed by Vinod and López-de-Lacalle, can be directly applied to data, without the strategies described previously to obtain a stationary series [20] — i.e. it can be applied to "any arbitrary stochastic process, including those that are non-stationary and heteroscedastic" [55, p. 6].

The procedure first sorts all the data points in increasing order, while keeping count of the original ordering index. Intermediate points are then calculated for the order statistics, and within each interval the maximum entropy density is computed. A sample of size N is drawn from a uniform distribution $[0, 1]$, where N is the same size of the original time series. These samples enable the computation of the sample quantiles, and are sorted using the original ordering index, recovering the dependence structure of the original dataset [21]. Replicates generated through MEB retain the shape of the original series, and their time and frequency domains remain close to the original ones. The whole procedure is non-parametric, which avoids any parametric restrictions or constrictions altogether.

Additionally, the method guarantees that both the central limit and ergodic theorems are upheld. As an additional benefit, unlike the IID bootstrap, the procedure allows limited extrapolation — the sampling range is not restrained to the closed interval $[\min(y_t),\ \max(y_t)]$, [20, 21, 23, 24]. As noted in the previous section, at the time of writing, no studies employing a modified version of the **Bagged.BLD.MBB.ETS**, where the MBB is replaced with the MEB, were identified.

2.4 Clustering

Due to how bootstrap innovations are generated, there might be a high covariance in the ensemble, which can in turn impact the mean squared forecast error (MSFE). Equation 2-3 showcases the three components of the error: the first term represents the fluctuations inherent to the data itself — meaning it cannot be reduced or controlled. The remaining two are associated to the predictor, and therefore to the forecasting method applied to a time series y_t — these two terms, added together, result in the mean square error (MSE). [6, p. 749].

$$\text{MSFE} = \text{Var}(y_{t+1|t}) + \text{bias}(\hat{y}_{t+1|t})^2 + \text{Var}(\hat{y}_{t+1|t}) \tag{2-3}$$

When bagging is used for time series forecasting, the average forecast can be computed as described in 2-4. To compute the bias and the variance, equations 2-5 and 2-6 can be employed. For the former, when a forecast is unbiased, bootstrapping will not generate improvements. For the latter, much of the research on bagging for time series focused on the first term, while ignoring the second term [6]. There is a trade-off, since it is hardly possible to reduce both bias and variance, this leads to opt for a biased estimator as long as it reduces the variance [56, 57].

$$\tilde{y}_{t+1|t} = \frac{1}{B}\sum_{i=1}^{B} \hat{y}^*_{(i)t+1|t} \tag{2-4}$$

$$\text{bias}(\tilde{y}_{t+1|t}) = \frac{1}{B}\sum_{i=1}^{B} \text{bias}(\hat{y}^*_{(i)t+1|t}) \tag{2-5}$$

$$\text{Var}(\tilde{y}_{t+1|t}) = \frac{1}{B}\sum_{i=1}^{B} \text{Var}(\hat{y}^*_{(i)t+1|t}) + \frac{1}{B^2}\sum_{i\neq i'} \text{Cov}[\hat{y}^*_{(i)t+1|t}, \hat{y}^*_{(i')t+1|t}] \tag{2-6}$$

Dantas and Oliveira [6] also highlight that previous works focused on the reduction of variance (the first term of equation 2-6), but no work was done to address the covariance effect when forecasting with bagging — the second component of equation 2-6. Nor there was any formulation to restrict the selection of biased forecasts. To address and reduce this effect, a clustering component was added to **Bagged.BLD.MBB.ETS**, after the bootstrap step of the method. The authors employed the Partitioning Around Medoids (PAM), due to its celerity and resilience to outliers [6].

Given how ubiquitous time series data are in domains such as engineering, environmental science, business finance, energy, health care, and government. Clustering these high dimensional — and often times large in size — datasets,

can be advantageous, leading to the discovery of relevant patterns for applications in the above-mentioned fields [15, 13, 28]. Thus, understanding of the methods available is crucial for application.

2.4.1 Taxonomy

Clustering is an unsupervised machine learning method that aims to identify disjoint subsets (clusters) in a set of data points or objects, without previous knowledge of the groups' make up. For any one cluster, elements within it are highly similar, while they are highly dissimilar when compared to elements belonging to other clusters [58, 15]. How this partitioning is achieved and what the optimal number of cluster should be is not a steadfast concept — the former depends on the data and a practitioner's needs, and the latter can be assessed through criteria that use information extracted from data itself. Clustering can also be the core of a process in itself, or a secondary aspect (as seen in the preprocessing done by Dantas and Oliveira [6]). When it comes to time series data clustering has procedures that are significantly different when compared to the ones for static data [58, 14, 32, 47, 13].

How these methods can be classified varies depending on how data is handled and how the clusters are formed. For static data, where the values do not change over time, there are five categories to put any one method in: partitioning, hierarchical, density-based, grid-based, and model-based [14, 32]. [15, 29-30] includes a sixth category called multi-step clustering, where different clustering methods are applied in sequence to the available data. A description of these items follows [14, 32, 15, 59, 60, 61]:

Partitioning methods n unlabelled objects are split into k partitions, guaranteeing that each group contains at least one object. These can be either crisp, where one object belongs to only one group, or fuzzy, where an object belongs to more than one group to varying degrees.

Hierarchical methods trees of clusters are generated and can be classified by the way the tree structure is created: agglomerative methods start from the bottom, considering each object as a cluster in itself, merging then gradually. Divisive methods do the reverse, starting with all objects into a single cluster, splitting until there is only one object per cluster.

Density-based methods Dense areas are separated by low-density, sparser areas. These circumvent methods that normally expect data originated from a probability distribution of a certain type, such as k-means. These methods are non-parametric and can detect outliers and remove noise, being suitable for datasets that present arbitrary shapes. Two well known

algorithms for this category are DBSCAN and OPTICS. While this family of methods has not found broad applications for time series data due to its high complexity [15], a more recent application implements DBSCAN and OPTICS for energy consumption forecast [11].

Grid-based methods Data is quantised into a finite number of cells, forming a grid upon which clustering operations are performed. [15] remarks that, at the time their article was written, no cluster-based application for time series clustering was identified.

Model-based methods This probabilistic approach attempts to optimise the fit between observed data and some mathematical model — e.g. a mixture of probability models. An assumption of this approach is that, for each cluster, a model is assumed, and the procedure is to find data that best fits that model. An alternative approach uses neural networks instead of probabilistic models.

For time series clustering, how models are classified varies. They can be labelled based on the representation of the available data. This yields three types: observation-based, model-based, and feature-based. Alternatively, considering how clusters are constructed based on the available data, an alternate set of labels can be used: hierarchical, partitioning, and fuzzy clustering. Considering the latter categorisation — i.e. hierarchical, partitioning and fuzzy clustering — the descriptions for the first two are the same as the ones presented at the start of this subsection. Fuzzy clustering creates overlapping groups, contrasting with crisp clustering (i.e. partitioning) where a series can be in more than one group, to varying degrees, making overlapping divisions [32, 15, 14, 13].

Representation-based models are also labelled as **observation-based** or raw-data-based methods due to the use of raw data, or due to the transformation of the observed process. Data can be represented either in the time or frequency domain. These methods lend themselves to a more geometric approach when clustering series. When working with raw data, a large amount of observations introduces noise, ignores the autocorrelation structure present in time series, and can be a high-dimensional task [14, 13, 40].

Feature-based clustering addresses these issues. These methods lead to dimensional reduction, which can improve computational times, and can be applied to series of different sizes. Features can be extracted from the time domain, frequency domain, or wavelet decomposition of a time series. Care needs to be taken, as a given feature might work well for a given implementation, but might not be useful in other contexts. Stationarity also plays a role: features obtained from stationary data are not necessarily the

same as those obtained from non-stationary data [14, 13, 40]. Features from the time domain include autocorrelations, partial autocorrelations, and cross-correlations. Frequency domain features include a series periodogram and its spectral ordinates. From the last one, the wavelet domain, discrete wavelet transform (DWT) can compute wavelet variances and correlations are also found in the literature [13].

Model-based approaches assume that the time series of interest were generated by a certain probability model or by a mixture of probability models. Series are clustered thorough parameter estimation or through the residuals of fitted models. Two challenges in this type of clustering include the handling of heteroskedastic time-series, and the posibility that the analysed time series data includes data that is not associated with the same number of parameter estimates [14, 13, 40].

Authors might use additional labels or other descriptions. [13] goes further and lists supervised feature-based clustering approaches and other methods for time series data. [38] posits two categories: correlation-based on-line clustering, and shape-based off-line clustering. The former is done in real time, where clusters are constructed in real time, based on the correlations between the different series. The latter uses the observations to cluster data of similar shapes through the use of a similarity function. [62] considers three categories for clustering methods: non-overlapping, overlapping, and fuzzy algorithms.

2.4.2 Dissimilarity measures

To enable partitioning, it is necessary to numerically measure the differences between the elements. Distances lend themselves to what is needed for analysis, where conventional distances, such as the Euclidean or Manhattan, can be used to compare profiles. Feature-based measures use concepts such as autocorrelations, spectral ordinates, or others, and in fact lead to a reduction of dimensionality. A third set of measures consider an approximation of the underlying models for the observed data, and evaluation of the dissimilarity is conducted based on what models are fit [31, 13].

Starting with distances based on raw data, a common dissimilarity measure used is the distance class of Minkowski, where the Manhattan Euclidean distances are special cases for this class, when a parameter r is set to $r = 1$ or $r = 2$ respectively. An alternative formulation for the distance can consider a weighing component [13, 38]. Other methods include the Canberra distance, Pearson correlation, and angular separation [63, p. 50]. Two other different

formulations for the a Pearson correlation-based distance include the use of square roots and a parameter β to control how fast the distance decreases [31].

Another alternative includes the Fréchet distance. While the method, at its inception in 1906 by Fréchet, considered only continuous curves, it is also possible to implement the distance for discrete cases. This distance measures the proximity of a curve as a whole, taking into consideration the ordering of the observations, and not just as two sets of points [31]. For details on the implementation, see [31, p. 5]. Other measures include the dynamic time warping distance (DTW) and its derivatives, which aim to identify the optimum warping path between two series under a certain constraint. Global alignment kernel distance (GAK) evaluates two series based on their kernels. Shape-based distance (SBD) is a faster alternative to DTW, and is used for the k-shape clustering method [32]. Regarding the computational cost for a selection of these measures, DTW, GAK, and SBD are considered to have medium, high, and low costs respectively [32]. For the Fréchet distance, the cost is also high, since it creates a set of all possible sequences of pairs that preserve the observation order [31].

Other authors have used features of the time series themselves in order to construct the dissimilarity matrix, including the autocorrelation and partial autocorrelation functions — ACF, PACF respectively – with uniform and geometric weights, periodogram ordinates and normalised periodogram ordinates, and spectral estimators [31]. However, use of ACF for clustering seems to be controversial. [64] mention that the metric is good to classify stationary and non-stationary processes, but performance is not the best when used to cluster ARMA and non-linear processes. [65] proposed an autocorrelation fuzzy c-means, and noted that AR(1) coefficients were used for clustering, yielding better results than k-Means, and comparable to hierarchical methods.

For periodogram-based measures, a selection of distances is available. A first collection of metrics can be built starting with the Euclidean distance between the periodogram ordinates for two series. From this metric, two others, a normalised and a logarithmic normalised versions can be derived. [64].

Given that the spectrum of a series is normally not known, it needs to be estimated. The estimators for the spectra lead to the construction of three different dissimilarity measures. The first metric substitutes the spectra for local linear smoothers of the periodograms generated through least squares. The second applies an exponential transformation to the least squares generated local linear smoothers of the log-periodograms to substitute the spectra. A third metric uses the previously described formulation, but makes use of the maximum local likelihood criterion instead of the least squares [64].

In addition to these metrics, [64] propose and test two additional non-parametric metrics, variations based on the log-spectra. One based on a generalised likelihood ratio distance, which is a significance test between the equality of two log-spectra. The other uses the integrated squared differences between these estimators.

In conjunction with the dissimilarity measures, the computation of prototypes are an important step, and it is directly related to the quality of the clusters. A prototype summarises the characteristics of the series contained within a cluster. Terms for this construct include *average series*, *prototypes* or *centroids* [32, 15].

When using a medoid as a prototype, "the centre of a cluster is defined as a sequence which minimises the sum of squared distances to other objects within the cluster" [15, p. 25]. Since the medoid itself is also a member of the original data set, this preserves the structure of the data and enables the reuse of the distance matrix for each iteration. One significant drawback is when such method is applied to larger datasets — the matrix needs to computed at the start of the procedure [32]. In the alternative, the averaging prototype, the object is constructed by averaging the data [15, p. 25].

2.4.3 Overview of clustering methods

k-Means As indicated in the query results, k-means is one of the most widely available clustering methods available, and it has also been used for time series clustering. The procedure has a random start, where k random objects are selected and allocated to a cluster using a dissimilarity measure. The centres are then recalculated. This is repeated until a convergence criteria is achieved [38]. Given the random start, some implementations enable multiple starts to use the method [66]. The algorithm has $\mathcal{O}(k \cdot n \cdot r \cdot D)$ complexity, where k is the number of clusters (defined by the user, before the algorithm is run), n is the size of the dataset, r is the number of iterations until convergence is achieved, and D refers to the dimensionality of the object space. A modification of the k-means algorithm leads to the k-medoids [38].

Partitioning Around Medoids k-medoids historically has its roots in operations research, the method has been proposed several times, but the primary reference and implementation is the one by Kaufman and Rousseeuw [30], named Partitioning Around Medoids (PAM) [39]. Since medoids are used in this method, PAM is periodically preferred over methods where centroids are

created through the mean or the median — i.e., the centroids come from the data set itself [32]. Another advantage that comes with it is the reuse of the whole distance matrix for the entire clustering process [32]. When it comes to the clustering of large time series, both k-means and k-medoids are preferred as an alternative to other clustering methods due to their complexity, although it should be noted that both are hill-climbing algorithms, which converge on a local optimum [38].

Fuzzy c-Means The Fuzzy c-Means (FcM) clustering method was introduced by two researchers, independently in 1974, and at a later point extended and formalised [41]. Fuzzy clustering can be taken as an extension of crisp clustering, where the constraint of non-overlapping groups is removed, making data points belong to more than one cluster to varying degrees. This degree of membership is constrained in such a manner that its sum equals 1 across all clusters. For k clusters, the membership of n objects can be represented through a matrix u, where all rows sum to 1 [32, 41]. Additonally, there is a fuzziness parameter m that needs to be adjusted before running the method, which can be set to any number in the open interval $(1, \infty)$ and needs to be tailored to the application. Numbers too close to one will result in a quasi-crisp partition, with membership degrees close to either 0 or 1, and exceedingly large values result in excessive overlap. While there are heuristics for the best choice of m, the most acceptable value seems to be $m = 2$ [41].

Fuzzy Analysis Proposed by Kaufman and Rousseeuw [30], this fuzzy clustering implementation does not use prototypes to construct the clusters — instead it aims to minimise the total dispersion of the dataset, which results in an algorithm not as sensitive to outliers, and which is robust to the spherical clustering assumption [30, 63]. The method also requires a parameter r to be set. This membership exponent can be any number on the open interval $(1, \infty)$. While the default is $r = 2$, complete fuzziness might ensue, and it is not possible to determine the best value for this method without prior testing [67, 30].

DBSCAN, OPTICS In DBSCAN, a cluster continues to grow as long as its density exceeds a certain threshold for a fixed-radius neighbourhood. Points are considered connected if they are located within each others' neighbourhoods [59]. Its implementation includes two parameters: a minimum number of points and the size of the neighbourhood [68]. OPTICS functions are an extended version of the DBSCAN algorithm, differing in operations — OPTICS does not

assign cluster memberships, but stores the order in which points are processed, and uses two other pieces of information, the core-distance and the reachability-distance. The latter enables the plot of the clustering structure [59].

2.4.4 Cluster validity indices

The assessment of cluster quality is a key component and enabler of successful clustering applications. This assessment is done through what are called cluster validity indices (CVIs), which take into consideration two factors, compactness and separation. The former is a measure of the proximity of objects within a cluster and the latter measures the degree of separation between clusters [69].

CVIs fall into two classes, internal and external. Internal indices only consider the divisions and the already clustered data, and it is also possible to combine them with significance testing. External indices evaluate the goodness of fit between the output of a given clustering method and a predefined, external structure. External indices are seldomly used, as such information is scarcely obtainable [69, 70, 32].

A third class of indices, called *relative* criteria, has also been put forward — these enable the comparison of different clustering produced by a given method under different parameters, in order to decide on which is the best [70]. There are alternatives to evaluate clustering output, with at least one approach based on cluster ensembles, where instead of evaluating the methods themselves, several clustering results are used to derive consensus [32].

It is not possible to infer which CVI will work best, thus they should be assessed for each particular case [32]. While much research has been devoted to the topic, there is an abundance of indices and approaches, with new indices regularly being created. It is also noted that there are no clear guidelines on choosing an index are readily available [69, 70]. There are also cases where such validation criteria have limitations to fulfil their task. Examples include the absence of external indices and the presence of internal indices that are not robust, the existence of subjective assessments such as case studies, or when there are structural idiosyncrasies in a dataset [69].

CVIs are described in detail in [32, p. 17], where they are identified as either Internal or External, whether a given index should be applied to evaluate crisp or fuzzy clusterings, whether the computed measure should be maximised or minimised, along with further considerations on computation. The relevant information for internal indices is summarised in table 2.11.

Index	Type	Target	Description
Silhouette	Crisp	Maximised	Requires cross-distance matrix.
Dunn	Crisp	Maximised	Requires cross-distance matrix.
COP	Crisp	Minimised	Requires cross-distance matrix.
Davies-Bouldin	Crisp	Minimised	Uses the distances to the computed cluster centroids.
Modified Davies-Bouldin	Crisp	Minimised	Uses the distances to the computed cluster centroids.
Calinski-Harabasz	Crisp	Maximised	Uses the distance to a global centroid.
Score function	Crisp	Maximised	Uses the distance to a global centroid.
MPC	Fuzzy	Maximised	—
K	Fuzzy	Minimised	Computes a global centroid.
T	Fuzzy	Minimised	Computes a global centroid.
SC	Fuzzy	Maximised	—
PBMF	Fuzzy	Maximised	Computes a global centroid.

Table 2.11: Selection of Crispy and Fuzzy cluster validity indices (Adapted from [32, p. 17])

2.4.5 Tools for times series clustering

For this research, the **R** language was the language of choice, given the readily available libraries and implementations of clustering methods. Out of four libraries identified, this research leverages two of them, **cluster** and **TSclust**.

stats Part of the base R program. Contains base partitioning (k-Means) and hierarchical clustering functions [66].

cluster Implementation of partitioning and hierarchical clustering methods, as proposed by Kaufman and Rousseeuw [30]. PAM is also available in this library [67].

dtwclust Implements a variety of clustering methods and cluster validity indices. For full details on the implementation, see [32].

TSclust Library implementing different dissimilarity measures for time series clustering. Methods generate a dissimilarity matrix that can be used for methods in the **cluster** library. Includes feature and observation-based

functions, and a density-based forecast clustering algorithm [31]. The library is compatible with the **cluster** and **dtwclust** libraries.

3
Method

3.1
Proposed algorithms

Given the state of the art – as seen through the queries results – in association with the materials gathered in chapter 2, this research puts forward models that explore variations on the bootstrap component of **Bagged.BLD.MBB.ETS** [8] and on the bootstrap and clustering components of the model proposed by Dantas and Oliveira [6, 71]. These are going to be referred from this point onwards as, respectively, **BaggedETS** and **BaggedClusterETS**.

Considering that most bootstrap procedures need extra steps in order to resample time series data, in order to ensure stationarity in the original method, an initial proposal is to use the Maximum Entropy Bootstrap (MEB) to simplify the procedure. Also, as noted in the previous chapter, the method was successfully used in other problems with dependent data. Since the queries have returned no results for the combination of Exponential Smoothing with MEB, experiments to check the effects of MEB-generated bootstraps have on **BaggedETS** should be checked. The method was implemented as the **BaggedETS.MEB** procedure.

The second proposal is to change the metric distances when computing the dissimilarity matrix required for PAM. Four dissimilarity distances were picked: one based on the Pearson Correlation (COR), another based on the discrete wavelet transform (DWT), a third using the least squares estimation of the log spectra of the series (LLR), and a fourth build on the generalised likelihood (GLK) ratio distance. While in the original method the dissimilarity matrix was built by using the raw data, through the computation of the distance between the series via Euclidean distance (EUCL), this proposal aims to evaluate the impacts on forecast quality when using metrics derived from features during the clustering step. The choice to use feature-based metrics aims to also check, by effectively reducing the dimensionality of the data, whether the noise from the bootstrap replicates affects clustering. These dissimilarity measures are also available in the **TSclust** library [31].

Equations 3-1 to 3-5 detail the workings of each of the distances used. Equation 3-1 is the Euclidean distance [13, 31]. Equation 3-2 is an implementation described in [31], using the square root of one minus the correlation

between two series. Equation 3-3 describes the metric based on the discrete wavelet transform (DWT) for two series, where the algorithm computes the sum of differences between the elements of the approximation coefficients $A_{j^\star}^{(u)}$ and $A_{j^\star}^{(v)}$, where $u, v \in \{1, \ldots, m\}$ and $u \neq v$. The parameter $j^\star$ is a scale parameter obtained by minimising the sum of square errors between the original series and its approximation [31].

Concerning spectra-estimator-based metrics, as seen in equations 3-4 and 3-5: d_{LLR} uses a divergence function W defined as $W(x) = \log(\alpha x + (1+\alpha)) - \alpha \log x$, where α is a value in the open interval $(0, 1)$. Since the spectra f_{X_T}, f_{Y_T} are unknown, the ought to be estimated. For this study, the spectra were obtained via least squares approximations of the exponential transformation of local linear smoothers of the log-periodograms [31]. For this study, α is set to 0.5 (the **TSclust** package default [31]).

The generalised likelihood described in equation 3-5, has the log difference of the periodograms $Z_k = \log(I_X(\lambda_k)) - \log(I_Y(\lambda_k))$. The difference between the log-spectra is defined as $\mu(\lambda_k) = m_X(\lambda_k) - m_Y(\lambda_k)$, where $m_X(\lambda) = \log(f_X(\lambda))$ [31, 64].

$$d_{\text{EUCL}} = \left(\sum_{t=1}^{T} (X_t - Y_t)^2 \right)^{\frac{1}{2}} \tag{3-1}$$

$$d_{\text{COR}} = \sqrt{2(1-r)}\ ,\ r = COR(X_T, Y_T) \tag{3-2}$$

$$d_{\text{DWT}} = \sqrt{\sum_{k} \left(a_{k,j^\star}^{(u)} - a_{k,j^\star}^{(v)} \right)^2} \tag{3-3}$$

$$d_{\text{LLR}} = \frac{1}{4\pi} \int_{-\pi}^{\pi} W \left(\frac{\hat{f}_{X_T}(\lambda)}{\hat{f}_{Y_T}(\lambda)} \right) d\lambda \tag{3-4}$$

$$d_{\text{GLK}} = \sum_{k=1}^{n} \left[Z_k - \hat{\mu}(\lambda_k) - 2\log\left(1 + e^{Z_k - \hat{\mu}(\lambda_k)}\right) \right] - \sum_{k=1}^{n} \left[Z_k - 2\log\left(1 + e^{Z_k}\right) \right] \tag{3-5}$$

And the third and last proposal aims to remove the validation step of the **BaggedClusterETS**. Instead of creating an internal train-validation split inside the algorithm, as in the original model, to adjust a model for each of the B replicates and comput the error, the proposal instead identifies the best clustering and extracts its medoids to do the forecast. For this method, the dissimilarity matrix is built with the raw data, using the dissimilarity measures as in the previous method. This method is referred henceforth as **BaggedMedoidETS**.

For this study, the Partitioning Around Medoids (PAM) clustering

algorithm is kept, alongside the Silhouette criterion to determine the optimal number of clusters, as it also uses the same cross distance matrix input to PAM. The method was kept due to its fast execution and resistance to outliers [6], and due to its simplicity, given that other methods require one or more parameters to be set before the algorithm is executed, as discussed in the previous chapter. The same applies for some distances, which need some sort of parameter to be defined beforehand.

Figure 3.1 is a side by side comparison of the base methods for this research, i.e. **BaggedETS** and **BaggedClusterETS**. Figure 3.2 displays the functioning of the proposed methods for this study. On the left, the **BaggedETS.MEB** model, and on the right, the **BaggedMedoidETS** model. For **BaggedClusterETS** and **BaggedMedoidETS**, changes in the clustering distance when running under PAM are not visually identified, as it is an inherent part of clustering procedure. The **ETS**, **BaggedETS**, and **BaggedClusterETS** methods are also going to be used as a reference when comparing error and computational performance for the proposed variations.

For each proposal, the number B of replicates is set as follows: $B = 100$ for **BaggedETS** and **BaggedETS.MEB**, and $B = 1000$ for both **BaggedClusterETS** and **BaggedMedoidETS**, as indicated in the relevant literature [8, 6].

Table 3.1 offers a highlight of the methods, where changes are applied for each of the experiments. The column indicating the bootstrap method also implies the choice for transformation and decomposition. Thus, for any method that uses MBB, the Box-Cox Transformation and STL/Loess decomposition is applied to the data, as described in figures 3.1 and 3.2. The Forecast column also highlights how the forecasts are produced.

Model	**Bootstrap**	**Clustering**	**Forecast**
BaggedETS	MBB	—	Bagging of the bootstrapped series.
BaggedETS.MEB	MEB	—	Bagging of the bootstrapped series.
BaggedClusterETS	MBB	PAM	Bagging of the selected clustered series.
BaggedMedoidETS	MBB	PAM	Bagging of the prototypes.

Table 3.1: Model overview

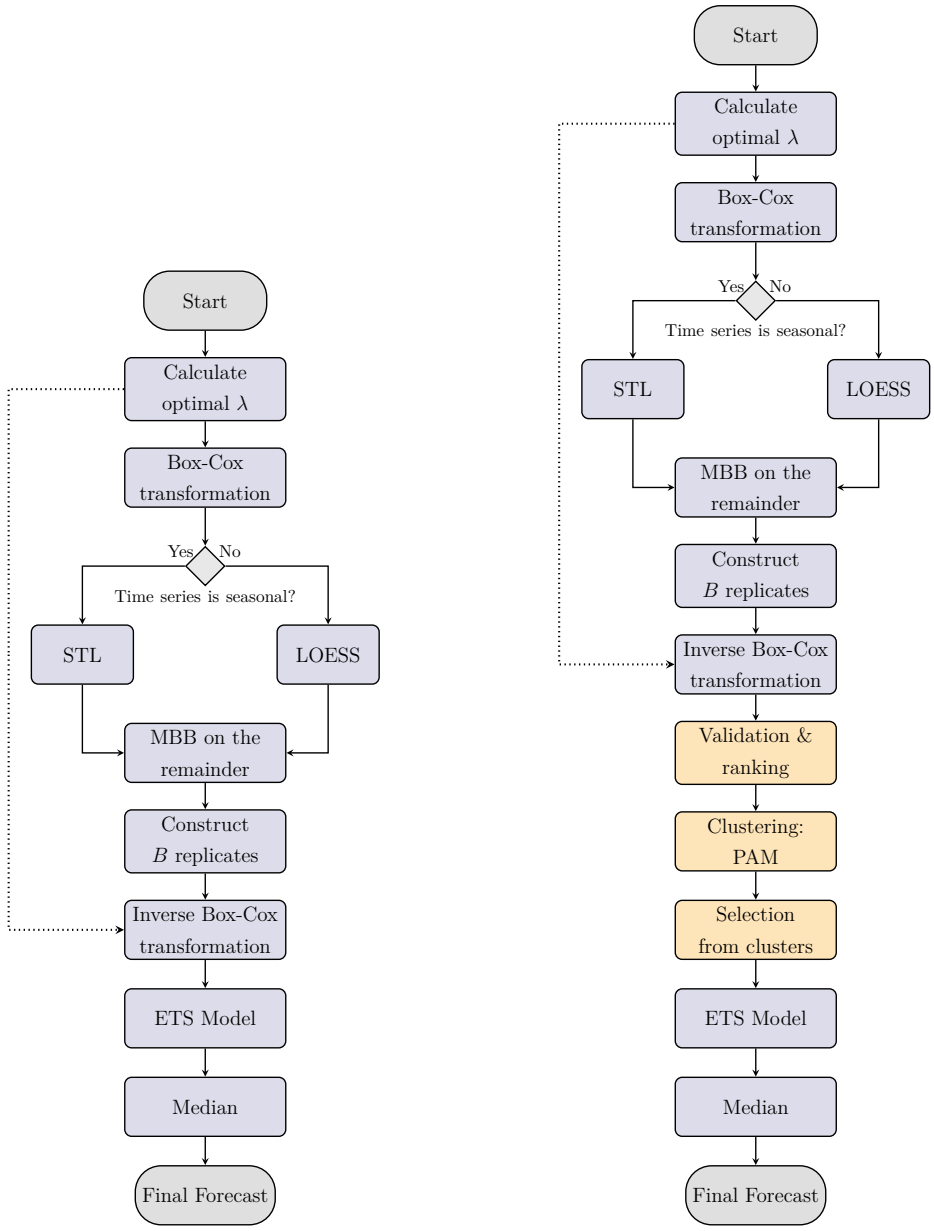

Figure 3.1: Flowcharts for **BaggedETS** (left), **BaggedClusterETS** (right)

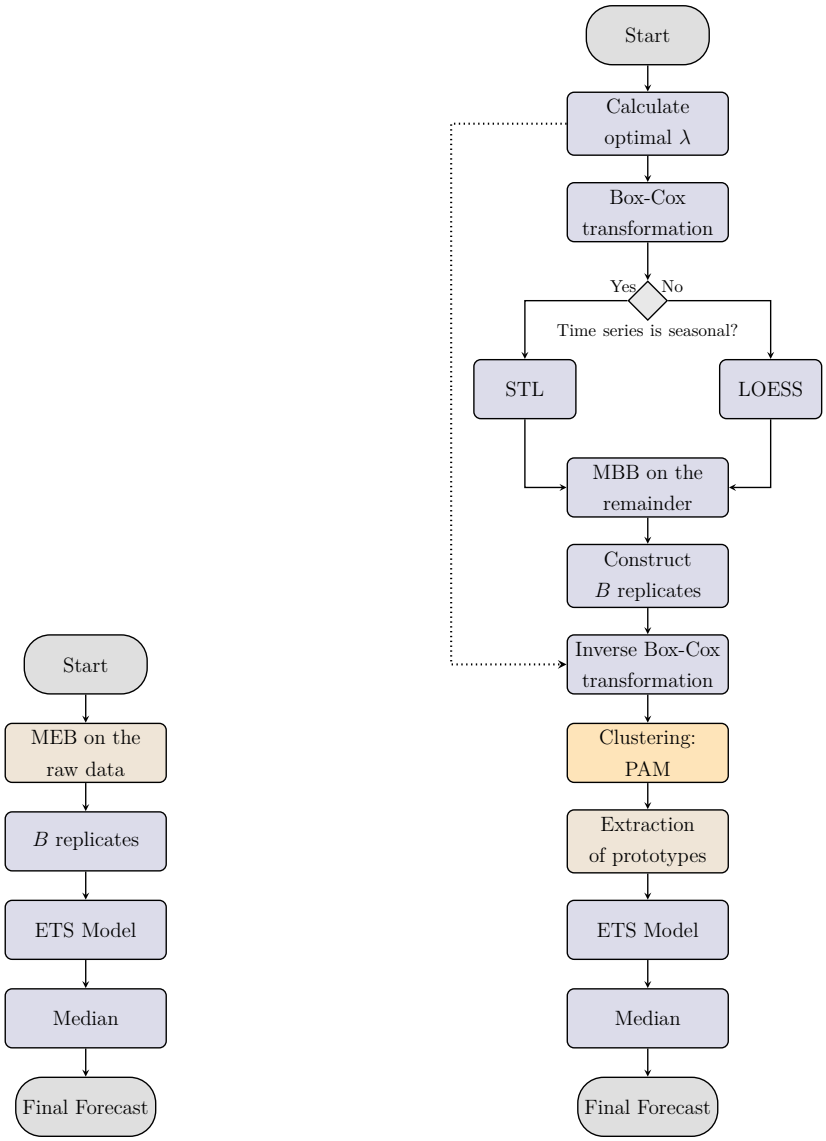

Figure 3.2: Flowcharts for **BaggedETS.MEB** (left) and **BaggedMedoidETS** (right)

3.2 Evaluation

To compare the quality and answer the first question outlined in the introduction – i.e. whether the proposed methods can generate improvement in forecasting quality when compared to the already established algorithms – two error measures are used. The sMAPE and the MASE.

The first, the symmetric mean absolute percentage error (sMAPE), penalises positive and negative values, and is dimensionless, making it possible to compare the results for different time series [72]. Equation 3-6 displays the formula used to compute the error measure, where $\hat{y}_t$ is a forecasted value at time t, y_t is the actual value at time t, and n is the forecast horizon.

This measure was also included due to its wide usage to evaluate forecasting methods [16].

The second, the mean absolute scaled error (MASE), is presented as an alternative that enables the comparison of forecast accuracy for series with different measurements, as shown in equations 3-7 and 3-8. The *training* mean absolute error (MAE) is used to compute the metric, and the computation differs between seasonal and non-seasonal series [16]. This metric (i.e. MASE) is used in addition to the previously defined sMAPE.

$$\text{sMAPE} = \frac{200}{n}\sum_{t=1}^{n}\frac{|y_t - \hat{y}_t|}{|y_t| + |\hat{y}_t|} \tag{3-6}$$

$$\text{MASE} = \frac{y_t - \hat{y}_t}{\frac{1}{T-1}\sum_{t=2}^{T}|y_t - y_{t-1}|} \tag{3-7}$$

$$\text{MASE}_S = \frac{y_t - \hat{y}_t}{\frac{1}{T-m}\sum_{t=m+1}^{T}|y_t - y_{t-m}|} \tag{3-8}$$

3.3 Performance

To answer the second question in the introduction – i.e. whether the methods introduce any computational overhead. The question is a bit nuanced: if any of the proposed methods offer the same forecast quality of a established method with shorter computational times, this can be interpreted as an improvement. The reverse, i.e. same or worse forecast quality with longer computational times, can be understood as a regression.

To enable such analysis, data on the execution time was also collected for the experiments to evaluate how the models behave and how performance is impacted when tweaking the clustering methods, the bootstrap procedure, or when removing the validation step entirely.

While the code for **BaggedClusterETS** [71] requires paralellisation to be deployed in a timely fashion, a finer evaluation on the performance gains for a range of cores is out of scope for this study. Neither benchmarks are produced on a finer level for internal components of the used methods.

4
Results

4.1
Datasets

For this research, data was gathered from different sources. Based on the works of [3] and [4], the same datasets are going to be used albeit with more recent, updated data.

The first set of series represents energy consumption from selected OECD countries and Brazil. For Brazil, data was downloaded from the country's Central Bank data service page [73, 74]. This dataset consists of 7 time series, covering the period between January 2000 and February 2020. The second set consist of 14 aviation data on passenger enplanements from selected European countries, the United States, Brazil, and Australia. For Brazil, only domestic flights were considered, and for Australia only international flights were gathered. Data covers the period from May 2004 to August 2019, with monthly frequency. European data was downloaded through the **eurostat** library, while data for the three other countries was obtained online from the pages of the Bureau of Infrastructure, Transport and Regional Economics (Australia), the National Civil Aviation Agency (Brazil), and the Bureau of Transportation Statistics (United States) [75, 76, 77, 78].

The energy dataset is displayed in figure 4.1 also displays a strong seasonal pattern throughtout the different countries. Canada and France exhibit a more stable behaviour, with almost no trend. Italy and Japan display structural changes in the trend and level, while still stable. Mexico, Turkey and Brazil exhibit a clear, upwards trend. But Brazil displays some dents between 2000-2005, and then again around 2010. Also, the series seems to have a reduced trend after 2015.

The aviation dataset can visualised in figure 4.2. All series display a clear seasonal pattern. For some European countries (e.g. Germany, Denmark), it is possible to observe the adverse effects of the 2010 Eyjafjallajökull eruption in the data, which caused disruptions to the European air traffic, grounding flights [79]. Changes in the level and trend between 2008 and 2010, due to the economic crisis, can be seen in different series for European countries (Netherlands, United Kingdom, Spain, Ireland, Portugal) and the United States. The series for Australia displays a a sharp upwards trend since 2009-2010. Last, but not least, the time series for Brazil displays a more erratic

behaviour, where its slope gets steeper after 2009, but with a structural change around 2014, where the series oscillates with a seasonal pattern around a level.

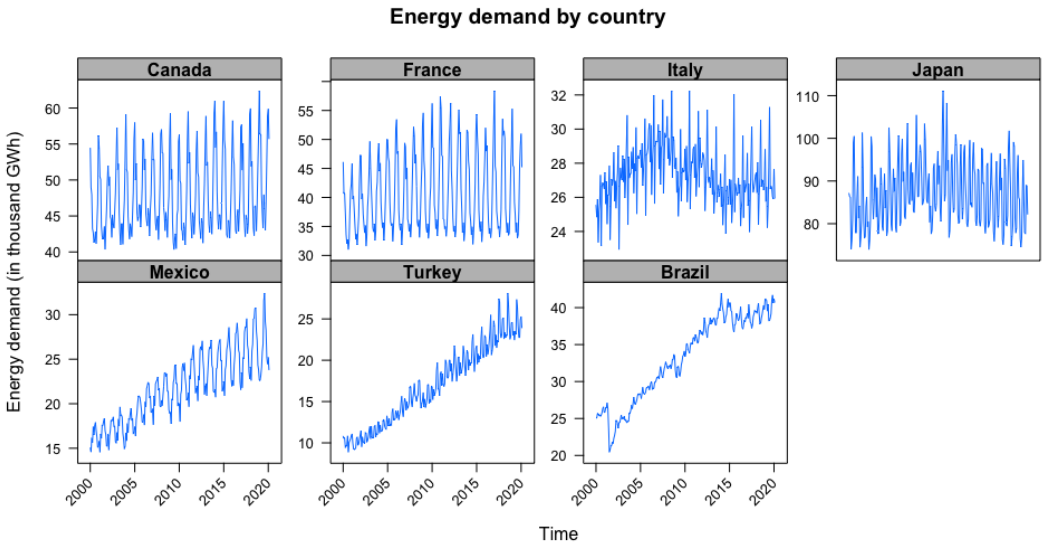

Figure 4.1: Energy time series

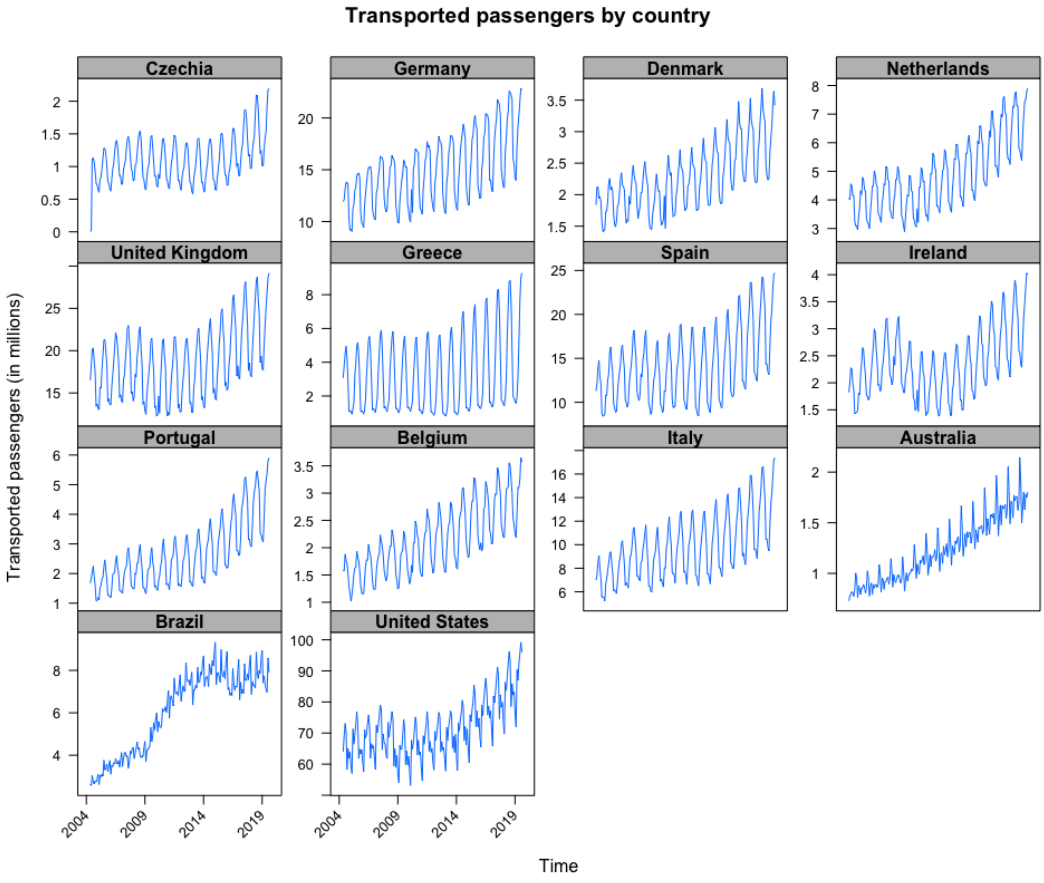

Figure 4.2: Aviation time series

To further verify and confirm the non-stationary nature of the datasets,

the Augmented Dickey–Fuller (ADF) Test to check for unit roots [1] and Autocorrelation Function (ACF) plots [1, 16] were used.

Table 4.1 contains the p-values for the ADF Test. The test was run with the library **tseries**. The null hypothesis H_0 he series has a unit root, and the alternative hypothesis H_1 is that the series is stationary [80]. For each series, $k = 50$ lags were used to run the test, with a level of significance $\alpha = 0.05$. Since none of the p-values fall below α, the null is not rejected for any of the series.

Dataset	**Country**	$p-$**value**
Energy	Canada	0.159
Energy	France	0.777
Energy	Italy	0.489
Energy	Japan	0.786
Energy	Mexico	0.746
Energy	Turkey	0.219
Energy	Brazil	0.961
Aviation	Czechia	0.990
Aviation	Germany	0.990
Aviation	Denmark	0.260
Aviation	Netherlands	0.482
Aviation	United Kingdom	0.104
Aviation	Greece	0.381
Aviation	Spain	0.979
Aviation	Ireland	0.325
Aviation	Portugal	0.986
Aviation	Belgium	0.607
Aviation	Italy	0.990
Aviation	Australia	0.342
Aviation	Brazil	0.825
Aviation	United States	0.541

Table 4.1: p-values for the ADF test by series and dataset

This non-stationary behaviour can also be visualised in the Autocorrelation Function (ACF) plots in figures 4.3 and 4.4. Where the autocorrelations for a given lag k either fall off slowly drop, display a sine-like behaviour, or a combination of both. Both figures use $k = 36$ lags to compute the ACF.

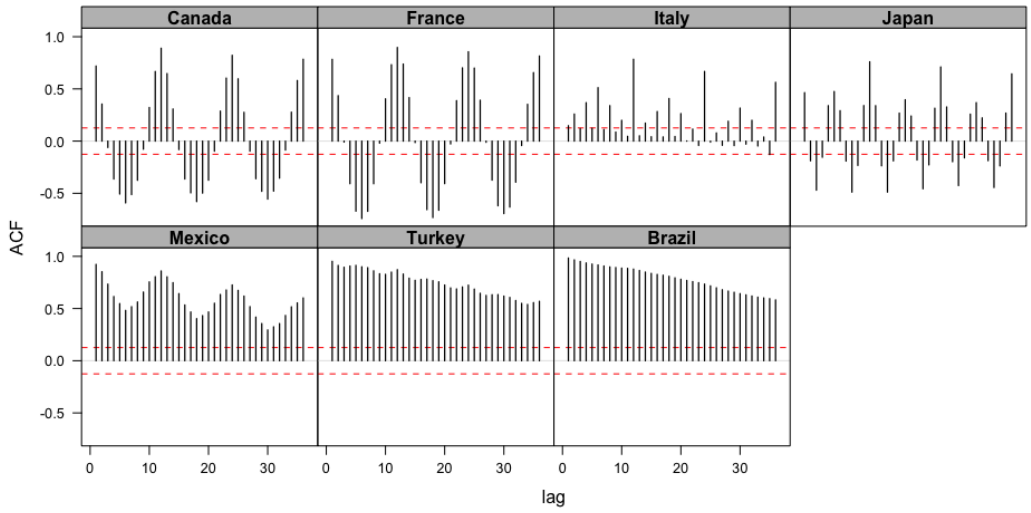

Figure 4.3: ACF plots for the Energy time series

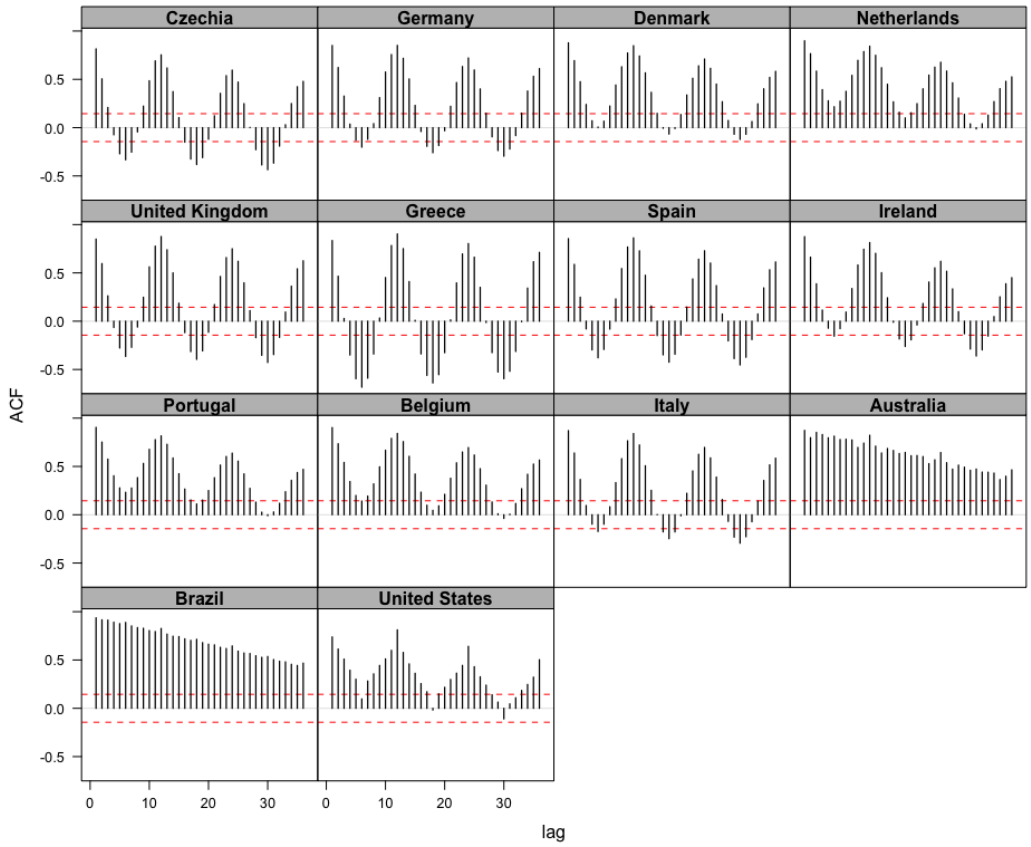

Figure 4.4: ACF plots for the Aviation time series

4.2

Computational environment

Code was run on a machine powered by a Intel Core i5-8500 3GHz, with 8GB RAM, Windows 10 Pro, running R version 3.6.3. The libraries used are detailed in table 4.2, alongside their versions and their uses. For each series, the random seed was set to 17071830. Parallelisation was used to speed up the forecasts for **BaggedClusterETS**, using three cores.

Library	Version	Description
cluster	2.1.0	Implementation of Partitioning Around Medoids.
forecast	8.12	ETS models and moving block bootstrap implementation.
meboot	1.4-7	Maximum entropy bootstrap implementation.
TSclust	1.2.4	Dissimilarity measures.

Table 4.2: Libraries used

4.3 Experiments

Experiment A focuses on evaluating the effects of changing the bootstrap procedure for **BaggedETS**, implements the Maximum Entropy Bootstrap (MEB) from the **meboot** package, with its default settings. Experiment B uses four different dissimilarity measures to conduct **BaggedClusterETS**, using the original method as a baseline. Experiment C removes the validation steps implemented by [6]. Models here use only the euclidean distance. The baseline models vary for each of the experiments: In experiment A, both **ETS** and **BaggedETS** are used, in order to assess where the model stands. In experiments B and C, **BaggedClusterETS** with the clustering distance set to Euclidean is the sole baseline used. For all experiments, execution time was logged in order to evaluate computational performance. Table 4.3 contains a summary of the conducted experiments. Error tables for this chapter highlight the best models in bold.

Forecasts were computed for a horizon of 12 months. For both datasets, data was split into training and validation sets. All models were adjusted to the former, and their forecasts were compared to the latter. For the aviation dataset, data from May 2004 until August 2018 was used in the training set, and from September 2018 to August 2019 for the validation set. For the energy dataset, the tranining set consisted of data from January 2020 to February 2019, and from March 2019 to February 2020 for the validation set.

Experiment	Evaluated model	Details
A	**BaggedETS.MEB**	Bootstraped series generated through MEB are input into the **BaggedETS** function.
B	**BaggedClusterETS**	Applies four different feature-based dissimilarity measures to cluster data.
C	**BaggedMedoidETS**	Removes the validation step for clustering. Uses only the Euclidean distance.

Table 4.3: Experiment overview

4.3.1 Experiment A: Maximum Entropy Bootstrap

Tables 4.4 and 4.5 showcase the error for the three models **ETS**, **BaggedETS** and **BaggedETS.MEB** . The last two are tagged as MBB and MEB in the tables due to the bootstrap method employed in each. When looking at the sMAPE for comparisons, the first model is picked 2 times; the second, 11 times; and the third, 7 times. When using the MASE, the number of times each model is select is 3, 11, and 7, respectively. The metrics give a different choice of model for all series, except for the Dutch, British, Spanish, Portuguese and Australian aviation series.

When **BaggedETS.MEB** is compared to the default **MBB** implementation, the former only outperforms the latter in 8 of the 21 series. One possible explanation for such behaviour might lie in the replicates generated through the MEB. Although the sampling range is increased, the actual range of generated by the MEB replicates is much closer to the original series than the ones generated by the MBB. The co-variance effect, discussed in the literature review, seems to be the cause for the weaker performance across both sMAPE and MASE – the method does not introduce enough variability to reduce the co-variance between the replicates. This can be seen in figures 4.5, 4.6, and 4.7, where it can be seen that the MBB replicates (in grey) covers a wider area than the MEB replicates (in red). The original series is the continuous black line over the replicates.

	sMAPE (%)			MASE		
	ETS	MBB	MEB	ETS	MBB	MEB
Canada	2.172	**1.997**	2.121	0.741	**0.691**	0.774
France	2.841	**2.386**	2.562	0.681	**0.551**	0.638
Italy	2.025	3.228	**1.991**	0.712	1.129	**0.701**
Japan	3.638	**2.615**	3.526	0.868	**0.628**	0.834
Mexico	7.800	**6.860**	7.782	2.589	**2.316**	2.563
Turkey	**1.955**	2.152	1.964	**0.529**	0.556	0.539
Brazil	2.642	**1.671**	3.139	0.732	**0.466**	0.975

Table 4.4: Experiment A error table: Energy series

	sMAPE (%)			MASE		
	ETS	MBB	MEB	ETS	MBB	MEB
Czechia	12.487	**8.031**	9.171	1.884	1.958	**1.456**
Germany	1.656	**1.448**	1.676	0.477	0.409	**0.369**
Denmark	2.610	3.003	**2.423**	**0.581**	0.722	0.587
Netherlands	4.476	3.108	**2.341**	1.007	0.678	**0.445**
United Kingdom	2.244	**1.574**	1.773	0.602	**0.414**	0.649
Greece	5.215	**2.843**	3.639	0.972	0.589	**0.555**
Spain	5.781	**2.566**	4.860	1.141	**0.532**	0.786
Ireland	5.322	2.537	**2.251**	0.800	**0.328**	0.350
Portugal	10.341	**5.284**	6.558	1.948	**1.034**	1.126
Belgium	3.372	**2.688**	3.258	0.630	0.561	**0.534**
Italy	4.343	**2.902**	3.201	1.015	0.686	**0.603**
Australia	**2.004**	2.491	3.295	**0.506**	0.639	0.915
Brazil	4.123	**3.200**	4.240	0.618	**0.467**	0.640
United States	2.167	**0.916**	1.614	0.745	**0.327**	0.635

Table 4.5: Experiment A error table: Aviation series

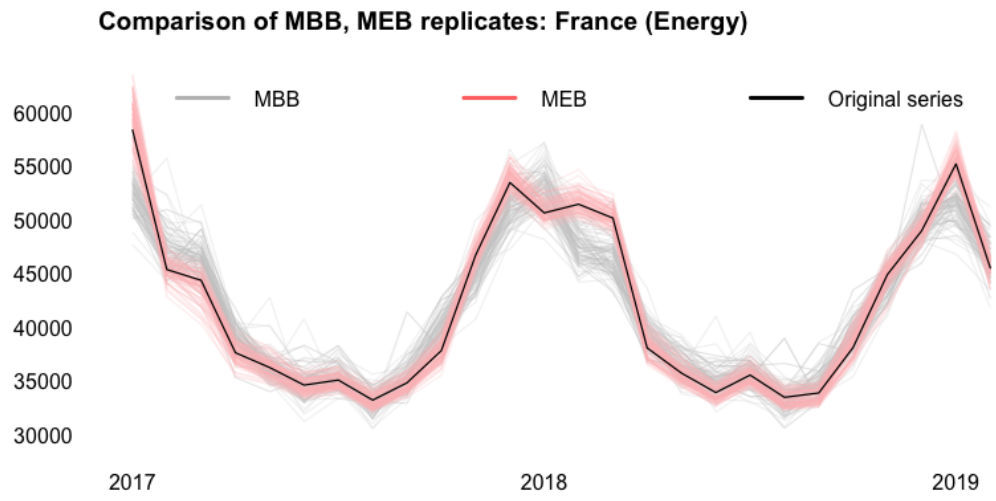

Figure 4.5: Experiment A: Replicates for the French Energy series

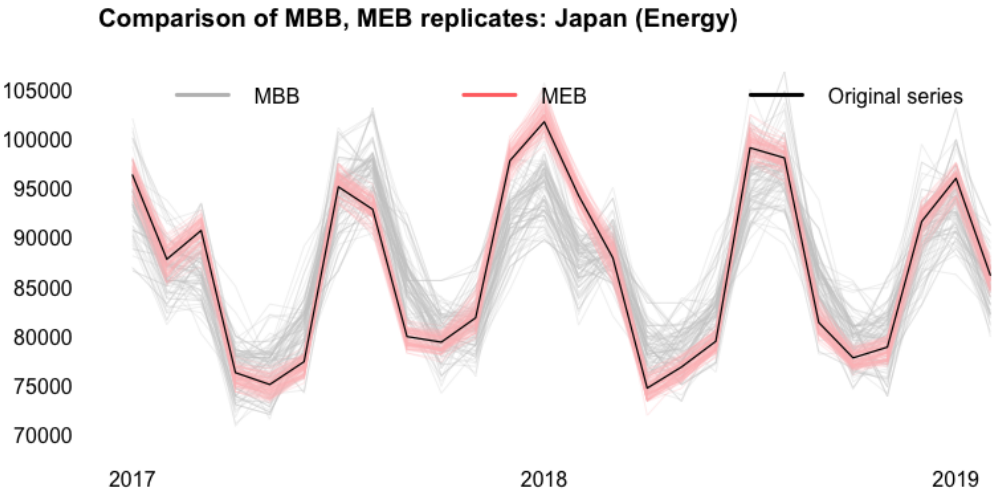

Figure 4.6: Experiment A: Replicates for the Japanese Energy series

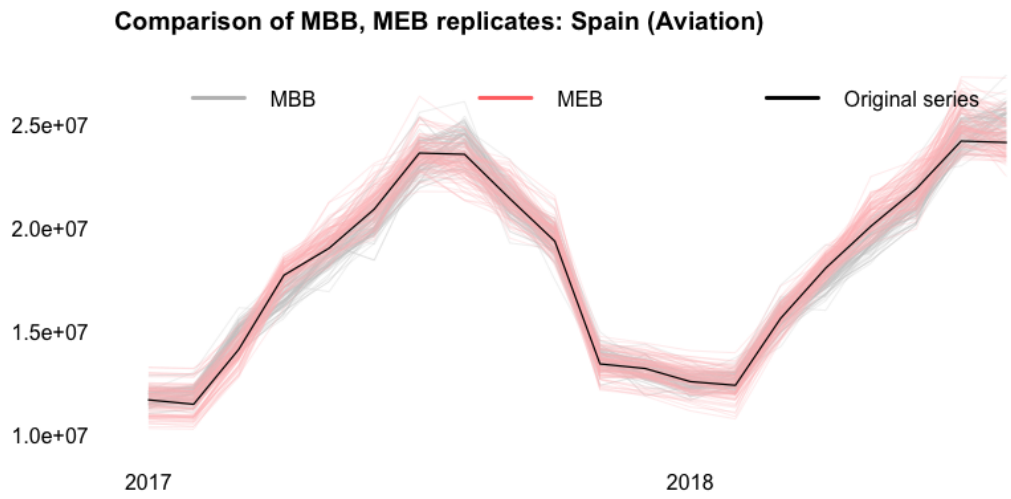

Figure 4.7: Experiment A: Replicates for the Spanish Aviation series

Looking at the errors produced by **BaggedETS.MEB**, these are more evenly spread in comparison, covering roughly the range of both **ETS** and **BaggedETS** models – probably a side-effect of the boostrap method not being able to introduce sufficient variability to tackle the co-variance effect. All distributions are leptokurtic, skewed to the left (with some striking outliers aboe the third quartile), as seen in figure 4.8. Although, performing a partial evaluation of performance, and comparing the results for **BaggedETS.MEB** models against the base **ETS**, the former displays a good performance for almost all series, as expected from the usage of bagging to improve predictor accuracy.

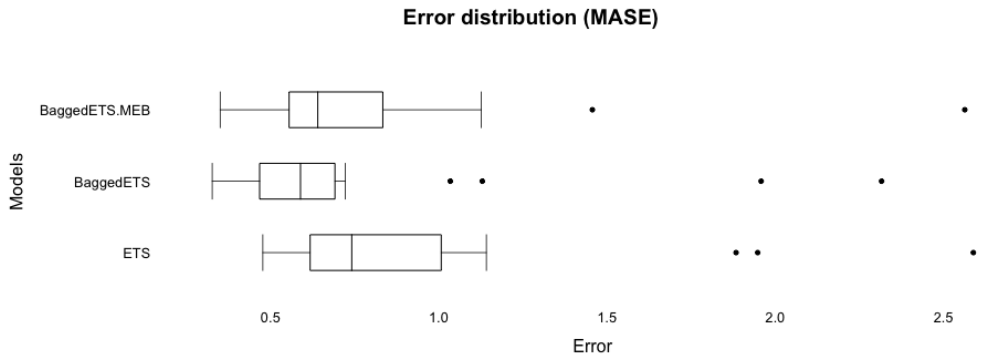

Figure 4.8: Experiment A: MASE distribution

Considering execution times between the bagged Models, there is no clearcut difference. Both implementations have displayed similar performance, as seen in figure 4.9, with a median roughly below 50s. Performance for **ETS** is not shown given that, for each of the 21 series, the models were adjusted and forecasted in less than 1 second.

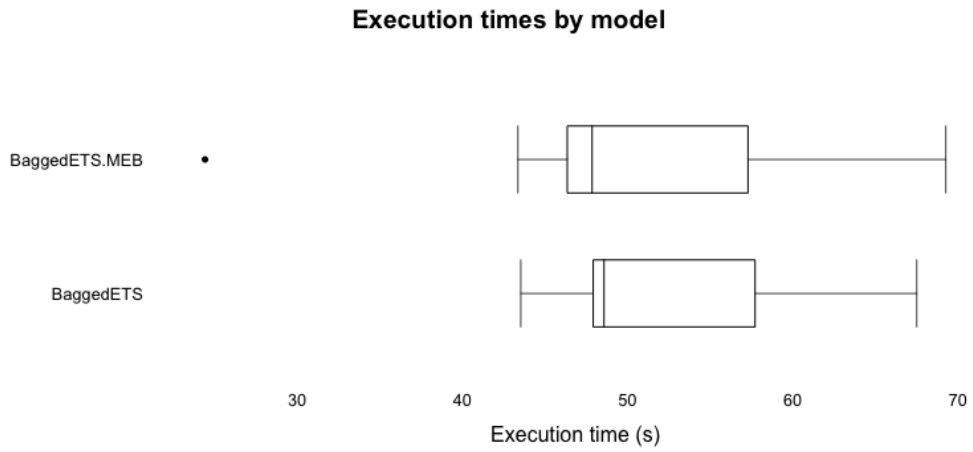

Figure 4.9: Experiment A: Execution times

4.3.2 Experiment B: Dissimilarity matrix construction

As previously discussed, this experiment aims to check whether the choice of clustering distance has any impact in forecast accuracy. Additionally, there is interest in checking how these impact cluster construction. Tables 4.6 and 4.7 showcase the sMAPE and MASE for each of the series and models. **BaggedClusterETS**, which employs the Euclidean distance [6], is listed as EUCL. The other four measures are DWT (based on the wavelet transform), COR (correlation based), LLR (spectral distance, least squares), GLK (generalized likelihood ratio). At a first glance, using sMAPE as a reference, feature-based models performed better in 15 out of the 21 series used. EUCL and DWT display the best performance for 6 series. LLR leads in 5 series. COR and GLK based forecasts had the best performance each for two series.

When looking at MASE evaluation is different. Here, the EUCL model is chosen for 4 of the 21 series. COR displays a better performance, with the best performance in 5 series. DWT rises from 3 to 6, and GLK drops from 4 to 2. Starting at the Energy series, Canada sees a change from DWT to EUCL, France from EUCL to LLR and Brazil from COR to EUCL. For the Aviation dataset, the only models that do not change are Czechia, Greece, Spain, Portugal, Italy and Brazil – for all others, widely different models are chosen under MASE.

	sMAPE (%)					MASE				
	EUCL	COR	DWT	LLR	GLK	EUCL	COR	DWT	LLR	GLK
Canada	1.896	1.899	**1.890**	1.890	1.900	**0.678**	0.680	0.678	0.678	0.679
France	**2.248**	**2.248**	2.272	2.266	2.281	0.529	0.529	**0.525**	**0.525**	0.529
Italy	3.003	2.998	**2.990**	3.026	2.998	1.074	1.072	**1.069**	1.079	1.070
Japan	2.577	2.560	2.577	**2.511**	2.560	0.613	0.611	0.616	**0.596**	0.614
Mexico	6.618	6.618	6.607	6.583	**6.548**	2.239	2.239	2.239	2.232	**2.224**
Turkey	1.948	1.954	1.923	**1.915**	1.924	0.522	0.523	0.506	**0.505**	0.509
Brazil	**1.568**	1.574	1.572	1.568	1.621	0.440	**0.439**	**0.439**	0.444	0.450

Table 4.6: Experiment B error table: Energy series

	sMAPE (%)					MASE				
	EUCL	COR	DWT	LLR	GLK	EUCL	COR	DWT	LLR	GLK
Czechia	7.499	7.823	7.430	**7.301**	7.376	1.655	1.688	1.648	**1.437**	1.460
Germany	1.604	1.588	**1.583**	1.616	1.585	0.448	**0.433**	0.438	0.439	0.435
Denmark	2.680	2.703	2.632	2.611	**2.603**	0.666	0.651	**0.649**	0.655	0.659
Netherlands	**3.045**	3.088	3.055	3.073	3.047	0.705	0.712	0.718	0.705	**0.701**
United Kingdom	1.724	**1.716**	1.775	1.725	1.735	0.497	0.489	0.491	0.484	**0.482**
Greece	**2.612**	2.708	2.757	2.762	2.685	**0.561**	0.590	0.604	0.592	0.590
Spain	**2.582**	2.607	2.694	2.658	2.686	**0.539**	0.540	0.549	0.544	0.546
Ireland	2.397	2.294	2.351	**2.272**	2.287	0.404	**0.382**	0.390	0.383	0.390
Portugal	4.639	4.659	4.648	**4.625**	4.651	0.926	0.937	0.928	**0.923**	0.929
Belgium	**2.528**	2.537	2.609	2.549	2.528	0.522	0.524	0.529	0.526	**0.521**
Italy	2.569	**2.390**	2.764	2.506	2.539	0.643	**0.604**	0.669	0.637	0.634
Australia	2.365	2.370	**2.358**	2.374	2.358	**0.587**	0.590	0.589	0.589	0.589
Brazil	2.957	2.957	**2.834**	3.038	2.939	0.442	0.442	**0.430**	0.445	0.437
United States	0.878	0.885	**0.850**	0.904	0.882	0.302	**0.298**	0.301	0.302	0.299

Table 4.7: Experiment B error table: Aviation series

Overall, the error distributions for all methods are very similar, as depicted in Figure 4.10.

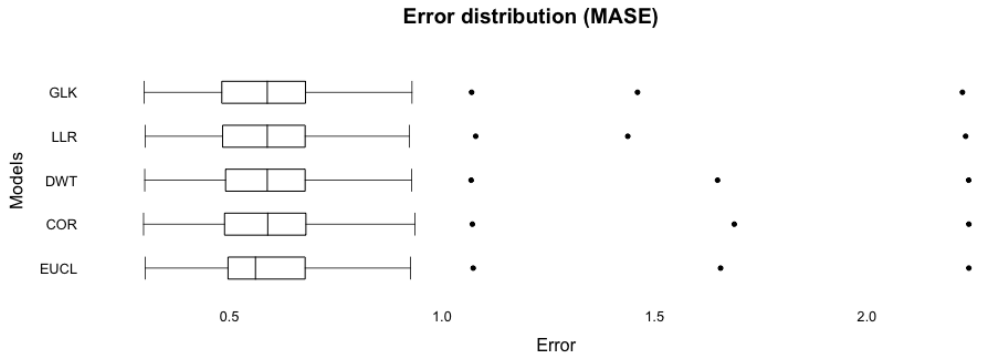

Figure 4.10: Experiment B: MASE distribution

The metrics have an impact on how clusters are formed. For most of the series in the Energy dataset, two clusters seem to be the norm, with some series displaying 3 or 4 clusters. There are extreme cases – Turkey has 75 and 72 clusters for EUCL and COR models, respectively, and Brazil displays 100 clusters under GLK. Tables 4.8 and 4.9 highlights these findings, where bold numbers indicate best model by MASE. While in table 4.8 showcases that only two clusters were picked for each series and modesl, with exceptions for the Turkey series using the EUCL and COR distances, and for Brazil using the GLK distance, table 4.9 depicts a more varied scenario. Models which used the EUCL and COR distances present a greater number of clusters for 9 of the 14 series. DWT for only 6 series. LLR and GLK have lower numbers – GLK does not go above 3 for this dataset.

The low number of clusters for spectral distances — with the exception of the Brazilian energy series — might highlight the dissimilarities picked by other distances, when analysed in the frequency domain, do not alter significantly the spectral properties of the series.

	EUCL	COR	DWT	LLR	GLK
Canada	**2**	2	2	2	2
France	2	2	2	**2**	2
Italy	2	2	**2**	4	2
Japan	2	2	2	**4**	2
Mexico	2	2	2	2	**2**
Turkey	75	72	2	**2**	2
Brazil	2	**2**	2	3	100

Table 4.8: Experiment B: Number of clusters by model (best models for Energy series, according to MASE, in bold)

	EUCL	COR	DWT	LLR	GLK
Czechia	63	89	59	**2**	2
Germany	40	**34**	73	8	2
Denmark	32	73	**91**	2	2
Netherlands	61	89	99	8	**2**
United Kingdom	95	47	42	5	**2**
Greece	**2**	3	2	14	3
Spain	**2**	2	2	3	2
Ireland	45	**41**	2	2	2
Portugal	2	17	2	**2**	2
Belgium	66	74	64	2	**2**
Italy	2	**2**	2	2	2
Australia	**57**	57	2	2	2
Brazil	2	2	**2**	2	2
United States	100	**2**	2	3	2

Table 4.9: Experiment B: Number of clusters by model (best models for Aviation series, according to MASE, in bold)

Execution times are roughly the same for EUCL, COR and DWT implementations. The higher computational cost is apparent for the spectral-based methods. LLR displays a slightly higher variance when compared to the first three methods, and GLK has its lower end on the same level as the median for LLR. These times can be visualised in figure 4.11.

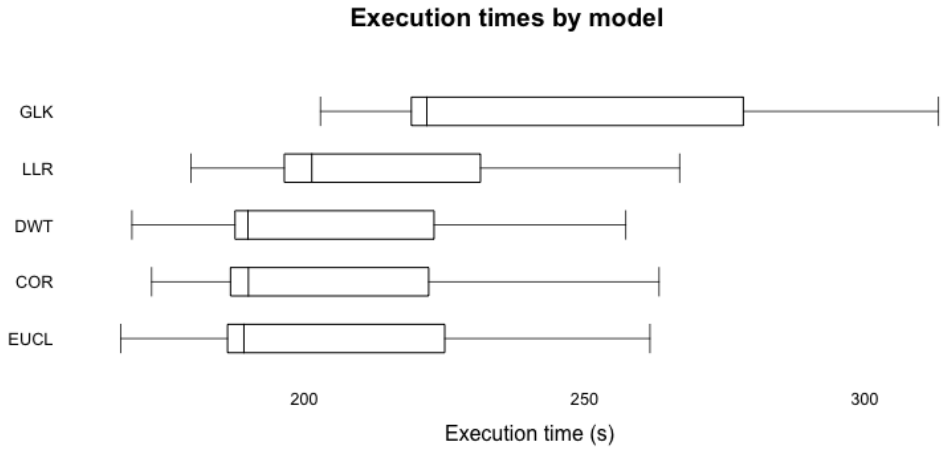

Figure 4.11: Experiment B: Execution times

4.3.3 Experiment C: Clustering without validation

Tables 4.10 and 4.11 display the sMAPE and MASE for the models. In the tables, the names CLUSTER and MEDOID refer to the **BaggedClusterETS** and **BaggedMedoidETS** models, respectively. Considering sMAPE, the latter is

picked three times for the French and Japanese energy series, and the Spanish aviation series. When using MASE to evaluate performance, **BaggedMedoidETS** is the chosen model for 7 out of the 21 series – series that change models are Mexico (Energy), and Denmark, Greece, and Italy (Aviation). The removal of the Validation and Ranking step (see Figure 3.1) has an impact in the quality of the forecast but it is not clear what can influence such change in behaviour. Evaluating performance for each of the methods, the differences are sometimes small (e.g. the MASE for Mexico and Turkey in the Energy series; the MASE for the United Kingdom, Australia in the Aviation series), sometimes these are larger (e.g. the sMAPE for Czechia and Brazil in the Aviation series).

Extending the comparison to other models, it displays a similar error dispersion to **ETS**, but performs better in 15 series out of 21. When compared to **BaggedETS**, **BaggedMedoidETS** offers better performance in 9 series out of 21 series. Since **BaggedMedoidETS** does not include the original series in the process, only the medoids obtained through clustering, it seems to not introduce enough variability to diminish the effects of the co-variance than both **BaggedETS** and **BaggedClusterETS**. The accuracy improvements seen when comparing it against **ETS** are due to the effects of bagging.

	sMAPE (%)		MASE	
	CLUSTER	MEDOID	CLUSTER	MEDOID
Canada	**1.896**	2.257	**0.678**	0.767
France	2.248	**2.068**	0.529	**0.462**
Italy	**3.003**	3.354	**1.074**	1.177
Japan	2.577	**2.246**	0.613	**0.531**
Mexico	**6.618**	6.627	2.239	**2.231**
Turkey	**1.948**	2.093	**0.522**	0.562
Brazil	**1.568**	1.679	**0.440**	0.462

Table 4.10: Experiment C error table: Energy series

	sMAPE (%)		MASE	
	CLUSTER	MEDOID	CLUSTER	MEDOID
Czechia	**7.499**	10.690	**1.655**	2.513
Germany	**1.604**	1.682	**0.448**	0.506
Denmark	**2.680**	2.809	0.666	**0.599**
Netherlands	**3.045**	3.354	**0.705**	0.804
United Kingdom	**1.724**	1.756	**0.497**	0.505

Continues on the next page

Continued from the previous page

	sMAPE (%)		MASE	
	CLUSTER	MEDOID	CLUSTER	MEDOID
Greece	**2.612**	2.615	0.561	**0.501**
Spain	2.582	**2.443**	0.539	**0.528**
Ireland	**2.397**	3.140	**0.404**	0.525
Portugal	**4.639**	5.741	**0.926**	1.106
Belgium	**2.528**	2.821	**0.522**	0.552
Italy	**2.569**	2.693	0.643	**0.617**
Australia	**2.365**	2.437	**0.587**	0.619
Brazil	**2.957**	3.900	**0.442**	0.584
United States	**0.878**	1.120	**0.302**	0.405

Table 4.11: Experiment C error table: Aviation series

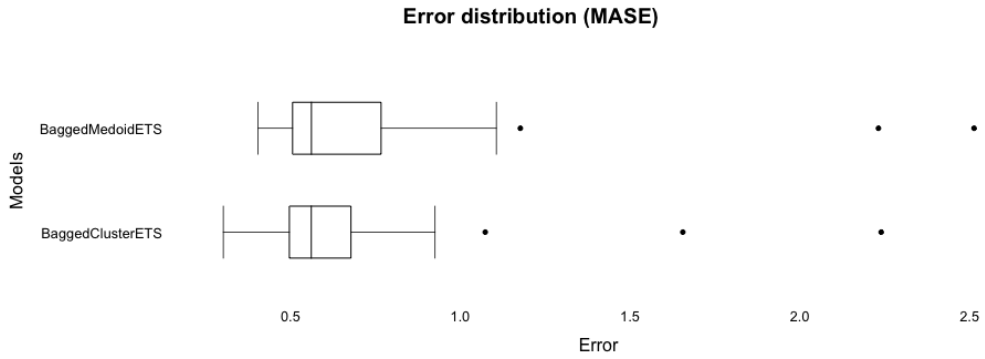

Figure 4.12: Experiment C: MASE distribution

Looking at the number of clusters picked by both methods, in Tables 4.12 and 4.13, **BaggedMedoidETS** picks a larger number of clusters than **BaggedClusterETS**, excluding the cases where both methods only pick two clusters, and the aberrant behaviour for the Australian and American aviation series.

	CLUSTER	MEDOID
Canada	**2**	2
France	2	**2**
Italy	**2**	2
Japan	2	**2**
Mexico	2	**2**
Turkey	**75**	92
Brazil	**2**	2

Table 4.12: Experiment C: Number of clusters by model (best models for Energy series, according to MASE, in bold)

	CLUSTER	MEDOID
Czechia	**63**	83
Germany	**40**	92
Denmark	32	**74**
Netherlands	**61**	86
United Kingdom	**95**	97
Greece	2	**2**
Spain	2	**2**
Ireland	**45**	96
Portugal	**2**	2
Belgium	**66**	98
Italy	2	**2**
Australia	**57**	2
Brazil	**2**	2
United States	**100**	2

Table 4.13: Experiment C: Number of clusters by model (best models for Aviation series, according to MASE, in bold)

When it comes to performance (Figure 4.13), **BaggedMedoidETS** was implemented without parallelisation, thus processing times scale when a large number of medoids is present. **BaggedClusterETS** was parellelised for this experiment, but its execution times on a single core are signficantly higher. In the same manner, improvements can be made by using more cores.

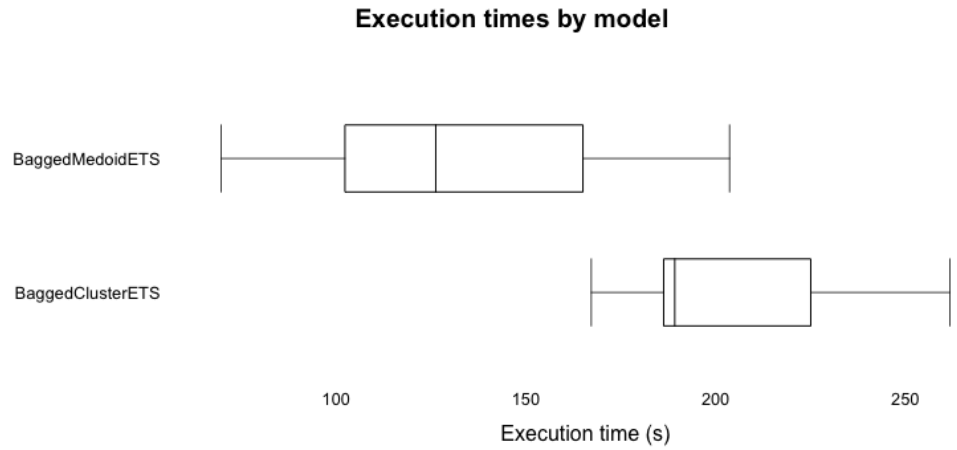

Figure 4.13: Experiment C: Execution times

5
Conclusion

This work presented extensions based on the works of Bergmeir, Hyndman, and Benítez [8] and Dantas and Oliveira [6], where different bagging and clustering procedures were combined with exponential smoothing, aiming to improve forecast accuracy. This covers a gap in the literature, by studying the application of a different bootstrap method and experimenting with different clustering measures. Three strategies were tested with real world data from the energy and aviation domains, where small improvements still have significant orders of magnitude for decision making.

The first experiment roughly expanded the ideas in Petropoulos, Hyndman, and Bermeir [10], by employing the framework proposed by Bergmeir, Hyndman, and Benítez [8], the **BaggedETS**. The results for the 21 series seem to give a slight advantage to the original proposal, which employs the MBB, over the one where the MEB was applied to the series. This slight advantage might not be significant, but the MEB seems to not introduce sufficient variability in the replicates to tackle the co-variance effect, which is reflected in the forecasts. Whether there is some structural pattern that might influence such behaviour and help identify when it is better to use this or that bootstrap method, it is not possible to tell from this study.

The second experiment, where feature-based metrics – i.e. the clusters built using the correlation, discrete wavelet transform of the series – were used to construct the dissimilarity matrix, saw an improvement in forecast accuracy, especially when evaluating the models with the MASE. Picking a different metric does not alter the results, and feature-based models still maintain the best performance under sMAPE.It seems possible that these measures reduce the effect of the noise during the clustering, but such reduction is not sufficient to improve the quality of the process. This is can be observed in some of the error measures where there is minimal or no difference, which might be an evidence of the weak influence of this noise-reducing effect. While the accuracy gains are minor, given the scale of the applications (energy demand in thousands of GWh, millions of monthly passengers carried), these might have an impact during decision making processes. The same cannot be said for smaller applications, where such gains might not make a difference. The change of metrics did not lead to improvements in computational times, and spectral measures, given how they are computed, actually increased execution times.

The third and last experiment employed only the medoids obtained through PAM to conduct the forecasts and compared it to the original **BaggedClusterETS** proposal. The usage of medoids led to a reduction of computation times, but at a cost of decreasing forecast quality. It could be argued that, for smaller applications, a regression in accuracy might be acceptable, especially since further gains in performance can be made by adding parallelisation to adjust the methods. But there is not a consistent drop in accuracy across the board to consider it, especially when the method does not consistently outperforms **BaggedETS**.

A limitation for this study was the low ammount of series used. While the findings do shed light on the workings of these models, they do not enable broader analysis, such as the ones done by Bergmeir, Hyndman, and Benítez [8] and Petropoulos, Hyndman, and Bermeir [10]. A recommendation would be to re-run the models here with M3 and M4 competition data, in order to also include series with yearly and quarterly frequencies. The volume of data would also enable a better, statistics-based evaluation and profiling of the procedures herein discussed. Neither this study occupied itself with an in-depth verification about which structural properties of the series lead to performance differences. While one might expect similar performance from similar shape series using the same methods, this was not the case.

On clustering methods, this study only made use the Partitioning Around Medoids algorithm, but other methods could be explored, especially fuzzy methods such as Fuzzy c-means and density-based methods such as DBSCAN – since these methods require parameters to be set before the clustering, a thorough exploration on methods or heuristics to set appropriate paramters is required. K-means has not been applied under this model, so it might warrant testing. There is also the possibility to study other decomposition methods, and verify the effects that TRAMO-SEATS and X-12 ARIMA, for example, have when decomposing series, even though these methods can only be applied to monthly and quarterly series. Yet another possibility is to explore the effects of different CVIs to choose the proper number of clusters.

6
References

[1] SHUMWAY, R. H.; STOFFER, D. S.. **Time series analysis and its applications: with R examples**. Springer, 4 edition, 2017.

[2] ZHANG, Y.; QU, H.; WANG, W. ; ZHAO, J.. **A novel fuzzy time series forecasting model based on multiple linear regression and time series clustering**. Mathematical Problems in Engineering, 2020, 2020.

[3] DANTAS, T. M.; OLIVEIRA, F. L. C. ; REPOLHO, H. M. V.. **Air transportation demand forecast through bagging holt winters methods**. Journal of Air Transport Management, 59:116–123, 2017.

[4] DE OLIVEIRA, E. M.; OLIVEIRA, F. L. C.. **Forecasting mid-long term electric energy consumption through bagging ARIMA and exponential smoothing methods**. Energy, 144:776–788, 2018.

[5] BREIMAN, L.. **Bagging predictors**. Machine Learning, 24(2):123–140, 1996.

[6] DANTAS, T. M.; OLIVEIRA, F. L. C.. **Improving time series forecasting: An approach combining bootstrap aggregation, clusters and exponential smoothing**. International Journal of Forecasting, 34(4):748–761, 2018.

[7] CORDEIRO, C.; NEVES, M. M.. **Forecasting time series with BOOT.EXPOS procedure**. REVSTAT-Statistical Journal, 7(2):135–149, 2009.

[8] BERGMEIR, C.; HYNDMAN, R. J. ; BENÍTEZ, J. M.. **Bagging exponential smoothing methods using STL decomposition and box–cox transformation**. International Journal of Forecasting, 32(2):303–312, 2016.

[9] HYNDMAN, R.; ATHANASOPOULOS, G.; BERGMEIR, C.; CACERES, G.; CHHAY, L.; O'HARA-WILD, M.; PETROPOULOS, F.; RAZBASH, S.; WANG, E. ; YASMEEN, F.. **forecast: Forecasting functions for time series and linear models**, 2019. R package version 8.9.

[10] PETROPOULOS, F.; HYNDMAN, R. J. ; BERGMEIR, C.. **Exploring the sources of uncertainty: Why does bagging for time series forecasting work?** European Journal of Operational Research, 268(2):545–554, 2018.

[11] LAURINEC, P.; LÓDERER, M.; LUCKÁ, M. ; ROZINAJOVÁ, V.. **Density-based unsupervised ensemble learning methods for time series forecasting of aggregated or clustered electricity consumption**. Journal of Intelligent Information Systems, 53(2):219–239, 2019.

[12] GEHARDT, T. E.; SILVEIRA, D. T.. **Métodos de pesquisas**. Editora da UFRGS, 2009.

[13] MAHARAJ, E. A.; D'URSO, P. ; CAIADO, J.. **Time series clustering and classification**. Chapman and Hall/CRC, 2019. ISBN (eBook): 978-0-429-05826-4.

[14] LIAO, T. W.. **Clustering of time series data – a survey**. Pattern recognition, 38(11):1857–1874, 2005.

[15] AGHABOZORGI, S.; SHIRKHORSHIDI, A. S. ; WAH, T. Y.. **Time-series clustering – a decade review**. Information Systems, 53:16–38, 2015.

[16] HYNDMAN, R. J.; ATHANASOPOULOS, G.. **Forecasting: Principles and Practice**. OTexts, 3 edition, 2019.

[17] AGGARWAL, C. C.; REDDY, C. K.. **Data clustering: algorithms and applications**. CRC press, 2014.

[18] HENNIG, C.; MEILA, M.; MURTAGH, F. ; ROCCI, R.. **Handbook of cluster analysis**. CRC Press, 2016.

[19] ATHANASOPOULOS, G.; SONG, H. ; SUN, J. A.. **Bagging in tourism demand modeling and forecasting**. Journal of Travel Research, 57(1):52–68, 2017.

[20] VINOD, H. D.; LÓPEZ-DE-LACALLE, J.. **Maximum entropy bootstrap for time series: The `meboot` R package**. Journal of Statistical Software, 29(5):1–19, 2009.

[21] COOK, B. I.; BUCKLEY, B. M.. **Objective determination of monsoon season onset, withdrawal, and length**. Journal of Geophysical Research: Atmospheres, 114(D23):12, 2009.

[22] XIA, X.; CHANG, Z.; LI, Y.; YE, L. ; QIU, M.. **Analysis and prediction for time series on torque friction of rolling bearings**. Journal of Testing and Evaluation, 46(3):1022–1041, 2017.

[23] SRIVASTAV, R. K.; SIMONOVIC, S. P.. **Multi-site, multivariate weather generator using maximum entropy bootstrap**. Climate Dynamics, 44(11-12):3431–3448, 2015.

[24] SHANG, H. L.. **Bootstrap methods for stationary functional time series**. Statistics and Computing, 28(1):1–10, 2018.

[25] ALDHYANI, T. H.; JOSHI, M. R.. **Integration of time series models with soft clustering to enhance network traffic forecasting**. In: 2016 SECOND INTERNATIONAL CONFERENCE ON RESEARCH IN COMPUTATIONAL INTELLIGENCE AND COMMUNICATION NETWORKS (ICRCICN), p. 212–214, 2016.

[26] LAOUAFI, A.; MORDJAOUI, M.; LAOUAFI, F. ; BOUKELIA, T. E.. **Daily peak electricity demand forecasting based on an adaptive hybrid two-stage methodology**. International Journal of Electrical Power & Energy Systems, 77:136–144, 2016.

[27] VILAR, J. A.; ALONSO, A. M. ; VILAR, J. M.. **Non-linear time series clustering based on non-parametric forecast densities**. Computational Statistics & Data Analysis, 54(11):2850–2865, 2010.

[28] VILAR, J. A.; VILAR, J. M.. **Time series clustering based on non-parametric multidimensional forecast densities**. Electronic Journal of Statistics, 7:1019–1046, 2013.

[29] MARTINS, A.; LAGARTO, J. ; CARDOSO, G. M.. **Electricity market price analysis using time series clustering**. In: 2019 16TH INTERNATIONAL CONFERENCE ON THE EUROPEAN ENERGY MARKET (EEM), p. 1–6. IEEE, 2019.

[30] KAUFMAN, L.; ROUSSEEUW, P. J.. **Finding groups in Data**. Wiley-Interscience, 1990.

[31] MONTERO, P.; VILAR, J. A.. **TSclust: An R package for time series clustering**. Journal of Statistical Software, 62(1):1–43, 2014.

[32] SARDÁ-ESPINOSA, A.. **Time-series clustering in R using the dtwclust package**. The R Journal, p. 1–22, 2019.

[33] KASSAMBARA, A.. **Practical guide to cluster analysis in R: Unsupervised machine learning**, volumen 1. STHDA, 2017.

[34] AGGARWAL, C. C.. **Outlier Analysis**. Springer, 2 edition, 2017.

[35] LAHIRI, S. N.. **Resampling methods for dependent data**. Springer, 2003.

[36] CORDEIRO, C.; NEVES, M.. **The bootstrap methodology in time series forecasting**. Proceedings of CompStat2006, p. 1067–1073, 2006.

[37] HYNDMAN, R. J.; KHANDAKAR, Y.. **Automatic time series for forecasting: the forecast package for r**. Journal of Statistical Software, 27:1–22, 2008.

[38] KOTSAKOS, D.; TRAJCEVSKI, G.; GUNOPULOS, D. ; AGGARWAL, C. C.. **Data clustering: algorithms and applications**, chapter 15. Time-Series Data Clustering, p. 357–380. CRC press, 2014.

[39] REDDY, C. K.; VINZAMURI, B.. **Data clustering: algorithms and applications**, chapter 4. A Survey of Partitional and Hierarchial Clustering Algorithms, p. 87–110. CRC press, 2014.

[40] CAIADO, J.; MAHARAJ, E. A. ; D'URSO, P.. **Handbook of cluster analysis**, chapter 12. Time Series Clustering, p. 241–263. CRC Press, 2016.

[41] D'URSO, P.. **Handbook of cluster analysis**, chapter 24. Fuzzy Clustering, p. 545–574. CRC Press, 2016.

[42] BROWN, R. G.. **Statistical forecasting for inventory control**. McGraw/Hill, 1959.

[43] HOLT, C. C.. **Forecasting seasonals and trends by exponentially weighted moving averages**. International journal of forecasting, 20(1):5–10, 2004.

[44] WINTERS, P. R.. **Forecasting sales by exponentially weighted moving averages**. Management science, 6(3):324–342, 1960.

[45] EFRON, B.. **Bootstrap method: another look at the jackknife**. The Annals of Statistics, 7:1–26, 1979.

[46] HASTIE, T.; TIBSHIRANI, R. ; FRIEDMAN, J.. **The elements of statistical learning: data mining, inference and prediction**. Springer, 2 edition, 2009.

[47] JAMES, G.; WITTEN, D.; HASTIE, T. ; TIBSHIRANI, R.. **An introduction to statistical learning**. Springer, 2013.

[48] EFRON, B.; TIBSHIRANI, R. J.. **An introduction to the bootstrap**. CRC Press, 1994.

[49] LANTZ, B.. **Machine learning with R**. Packt Publishing Ltd, 2 edition, 2015.

[50] MCMURRY, T. L.; POLITIS, D. N.. **Banded and tapered estimates for autocovariance matrices and the linear process bootstrap**. Journal of Time Series Analysis, 31(6):471–482, 2010.

[51] LAHIRI, S. N.. **Theoretical comparisons of block bootstrap methods**. Annals of Statistics, p. 386–404, 1999.

[52] DAGUM, E. B.; BIANCONCINI, S.. **Seasonal Adjustment Methods and Real Time Trend-Cycle Estimation**. Spring, 2016.

[53] CLEVELAND, R. B.; CLEVELAND, W. S.; MCRAE, J. E. ; TERPENNING, I.. **STL: a seasonal-trend decomposition**. Journal of official statistics, 6(1):3–33, 1990.

[54] POLITIS, D. N.; WHITE, H.. **Automatic block-length selection for the dependent bootstrap**. Econometric Reviews, 23(1):53–70, 2004.

[55] COOK, E. R.; PALMER, J. G.; AHMED, M.; WOODHOUSE, C. A.; FENWICK, P.; ZAFAR, M. U.; WAHAB, M. ; KHAN, N.. **Five centuries of Upper Indus River flow from tree rings**. Journal of Hydrology, 486:365–375, 2013.

[56] GEMAN, S.; BIENENSTOCK, E. ; DOURSAT, R.. **Neural networks and the bias/variance dilemma**. Neural Computation, 4(1):1–58, 1992.

[57] MURPHY, K. P.. **Machine learning: a probabilistic perspective**. MIT press, 2012.

[58] HENNIG, C.; MEILA, M.. **Handbook of cluster analysis**, chapter 1. Cluster Analysis: An Overview, p. 1–20. CRC Press, 2016.

[59] ESTER, M.. **Data clustering: algorithms and applications**, chapter 5. Density-based Clustering, p. 111–126. CRC press, 2014.

[60] CHENG, W.; WANG, W. ; BATISTA, S.. **Data clustering: algorithms and applications**, chapter 6. Grid-based Clustering, p. 127–148. CRC press, 2014.

[61] DENG, H.; HAN, J.. **Data clustering: algorithms and applications**, chapter 3. Probabilistic Models for Clustering, p. 61–86. CRC press, 2014.

[62] D'URSO, P.; DISEGNA, M.; MASSARI, R. ; PRAYAG, G.. **Bagged fuzzy clustering for fuzzy data: An application to a tourism market**. Knowledge-Based Systems, 73:335–346, 2015.

[63] EVERITT, B. S.; LANDAU, S.; LEESE, M. ; STAHL, D.. **Cluster analysis**. John Wiley & Sons, 2011.

[64] DÍAZ, S. P.; VILAR, J. A.. **Comparing several parametric and nonparametric approaches to time series clustering: a simulation study**. Journal of classification, 27(3):333–362, 2010.

[65] D'URSO, P.; MAHARAJ, E. A.. **Autocorrelation-based fuzzy clustering of time series**. Fuzzy Sets and Systems, 160(24):3565–3589, 2009.

[66] R CORE TEAM. **R: A Language and Environment for Statistical Computing**. R Foundation for Statistical Computing, Vienna, Austria, 2019.

[67] MAECHLER, M.; ROUSSEEUW, P.; STRUYF, A.; HUBERT, M. ; HORNIK, K.. **cluster: Cluster Analysis Basics and Extensions**, 2019. R package version 2.1.0 — For new features, see the 'Changelog' file (in the package source).

[68] HAHSLER, M.; PIEKENBROCK, M. ; DORAN, D.. **dbscan: Fast density-based clustering with R**. Journal of Statistical Software, 91(1):1–30, 2019.

[69] XIONG, H.; LI, Z.. **Data clustering: algorithms and applications**, chapter 23. Clustering Validation Measures, p. 571–602. CRC press, 2014.

[70] HALKIDI, M.; VAZIRGIANNIS, M. ; HENNIG, C.. **Handbook of cluster analysis**, chapter 26. Method-Independent Indices for Cluster Validation and Estimating the Number of Clusters, p. 595–617. CRC Press, 2016.

[71] DANTAS, T. M.. **Bagged.Cluster.ETS**, 2018. `https://github.com/tiagomendesdantas/Bagged.Cluster.ETS`. Accessed on 2020-06-02.

[72] MAKRIDAKIS, S.. **Accuracy measures: theoretical and practical concerns**. International Journal of Forecasting, 9(4):527–529, 1993.

[73] IEA. **Monthly electricity statistics**, 2020. `https://www.iea.org/reports/monthly-electricity-statistics`. Accessed on 2020-06-02.

[74] ELETROBRAS. **Consumo de energia elétrica - brasil - total**, 2020. `https://www3.bcb.gov.br/sgspub`. Accessed on 2020-06-02.

[75] LAHTI, L.; HUOVARI, J.; KAINU, M. ; BIECEK, P.. **eurostat R package**. R Journal, 9(1):385–392, 2017. Version 3.6.1.

[76] ANAC. **Relatório demanda e oferta do transporte aéreo**, 2020. `https://www.anac.gov.br/assuntos/setor-regulado/empresas/envio-de-informacoes/relatorio-demanda-e-oferta-do-transporte-aereo-empresas-brasileiras`. Accessed on 2020-06-02.

[77] BTS. **Passengers, all carriers - all airports**, 2020. `https://www.transtats.bts.gov/Data_Elements.aspx`. Accessed on 2020-06-02.

[78] BITRE. **International airline activity–time series**, 2020. `https://www.bitre.gov.au/publications/ongoing/international_airline_activity-time_series`. Accessed on 2020-06-02.

[79] GUDMUNDSSON, M. T.; THORDARSON, T.; HÖSKULDSSON, Á.; LARSEN, G.; BJÖRNSSON, H.; PRATA, F. J.; ODDSSON, B.; MAGNÚSSON, E.; HÖGNADÓTTIR, T.; PETERSEN, G. N. ; OTHERS. **Ash generation and distribution from the april-may 2010 eruption of Eyjafjallajökull, Iceland**. Scientific reports, 2(572), 2012.

[80] TRAPLETTI, A.; HORNIK, K.. **tseries: Time Series Analysis and Computational Finance**, 2019. R package version 0.10-47.