Digite algo...

Thesis presented to the Instituto Tecnológico de Aeronáutica, in partial fulfillment of the requirements for the degree of Doctor of Science in the Program of Electronic and Computing Engineering, Field of Control Systems.

Filipe Rodrigues de Souza Moreira

MAXIMUM VISIBILITY: A NOVEL APPROACH FOR TIME SERIES FORECASTING BASED ON COMPLEX NETWORK THEORY

Thesis approved in its final version the signatories below:

Prof. Dr. Takashi Yoneyama
Advisor

Prof. Dr. Filipe Alves Neto Verri
Co-advisor

Profa. Dra. Emilia Villani
Pro-Rector of Graduate Courses

Campo Montenegro
São José dos Campos, SP – Brazil
2022

Cataloging-in-Publication Data
Documentation and Information Division

Moreira, Filipe Rodrigues de Souza
Maximum Visibility: A Novel Approach for Time Series Forecasting Based on Complex Network Theory / Filipe Rodrigues de Souza Moreira.
São José dos Campos, 2022.
206f.

Thesis of Doctor of Science – Course of Electronic and Computing Engineering. Field of Control Systems - Instituto Tecnológico de Aeronáutica, 2022. Advisor: Prof Dr. Takashi Yoneyama. Co-Advisor: Prof. Dr. Filipe Alves Neto Verri.

1. Time Series Forecasting. 2. Natural Visibility Graph. 3. Forecasting Model. 4. Complex Network. I. Instituto Tecnológico de Aeronáutica. II. Maximum Visibility: A Novel Approach for Time Series Forecasting Based on Complex Network Theory

BIBLIOGRAPHIC REFERENCE

MOREIRA, Filipe Rodrigues de Souza. **Maximum Visibility: A Novel Approach for Time Series Forecasting Based on Complex Network Theory**. 2022. 206f. Thesis of Doctor of Science in Control Systems – Instituto Tecnológico de Aeronáutica, São José dos Campos.

CESSION OF RIGHTS

AUTOR NAME: Filipe Rodrigues de Souza Moreira
PUBLICATION TITLE: Maximum Visibility: A Novel Approach for Time Series Forecasting Based on Complex Network Theory
PUBLICATION KIND/YEAR: Tese / 2022

_____

Filipe Rodrigues de Souza Moreira
H27E, 115, Campus do CTA,
CEP: 12228-550, São José dos Campos - SP

MAXIMUM VISIBILITY: A NOVEL APPROACH FOR TIME SERIES FORECASTING BASED ON COMPLEX NETWORK THEORY

Filipe Rodrigues de Souza Moreira

Thesis Committee Composition:

Prof. Dr.	Rodrigo Arnaldo Scarpel	Chairperson	-	ITA
Prof. Dr.	Takashi Yoneyama	Advisor	-	ITA
Prof. Dr.	Filipe Alves Neto Verri	Co-advisor	-	ITA
Prof. Dr.	Marco Antônio Leonel Caetano		-	INSPER
Profa. Dra.	Mariá Cristina Vasconcelos Nascimento Rosset		-	UNIFESP
Profa. Dra.	Chang Chiann		-	USP
Prof. Dr.	Mauri Aparecido de Oliveira		-	ITA

ITA

Dedico esse trabalho a Deus, que me presenteou com capacidade intelectual, com orientadores dedicados e muito capazes, com uma família apoiadora e com amigos que me impulsionam a ser uma pessoa melhor todos os dias.

"*Porque sou eu que conheço os planos que tenho para vocês, diz o Senhor. Planos de fazê-los prosperar e não de causar dano, planos de dar a vocês esperança e um futuro*".
Jeremias 29:11.

Abstract

This work presents two time series forecasting methods. The first is the Maximum Visibility Approach (MVA), a new time series forecasting method based on the Complex Network theory. The second is the Hybrid Means of Multiple Approaches (HMMA) which is a hybrid methodology formed by the combination of two forecasting models. The MVA initially maps time series data into a complex network using the visibility graph method. Then, based on the similarity measures between the nodes in the network, MVA calculates the one-step-ahead forecasts. MVA does not use all past terms in the forecasting process, but only the most significant observations, which are indicated as a result of the autocorrelation function. The HMMA combines two forecasting methods using four types of means: quadratic, arithmetic, geometric and harmonic. Using the means inequalities, four new one-step-ahead estimators are created so, it is possible that one of these new estimators be closer to the true next term, and in this case, the combination will be more accurate than the two original estimators. These methods were applied to five different groups of data, most of them showing trend characteristics, seasonal variations and/or non-stationary behavior. We estimated error measures to evaluate the performances of MVA and HMMA. The results of the statistical tests and error measures revealed that both techniques has good performance compared to the accuracy obtained by the comparative methods considered in this work. In all cases, MVA and HMMA surpassed or achieved accuracy similar to the other forecasting methods in Literature, which confirms that this work will contribute to the field of time series forecasting not only in the theoretical aspect, but also in practice.

Resumo

Este trabalho apresenta dois métodos de previsão de séries temporais. O primeiro é o Maximum Visibility Approach (MVA), um novo método de previsão de séries temporais baseado na teoria de Redes Complexas. O segundo é o Hybrid Means of Multiple Approaches (HMMA), que é uma metodologia híbrida formada pela combinação de dois modelos de previsão. O MVA inicialmente transforma os dados da série temporal em uma rede complexa usando o método Natural Visibility Graph (NVG). Em seguida, com base nas medidas de similaridade entre os nós da rede, o MVA calcula as previsões um passo à frente. Nem todos os termos anteriores são utilizados no processo de previsão, mas apenas as observações mais significativas, que são indicadas como resultado da função de autocorrelação. O HMMA combina dois métodos de previsão usando quatro tipos de médias: quadrática, aritmética, geométrica e harmônica. Usando as desigualdades das médias, quatro novas estimativas um passo à frente são criadas, portanto, tendo em vista a aleatoriedade das séries temporais reais, é possível que um desses novos estimadores esteja mais próximo do termo verdadeiro seguinte e nesse caso, essa combinação terá uma acurácia maior que as dos estimadores originais. Esses métodos foram aplicados a cinco diferentes grupos de dados, a maioria deles apresentando características de tendência, variações sazonais e/ou não-estacionariedade. Foram estimadas medidas de erro e realizados testes estatísticos para avaliar os desempenhos do MVA e HMMA. Os resultados obtidos revelaram que ambas as técnicas apresentam bom desempenho em relação à precisão obtida pelos métodos de comparação considerados neste trabalho. Em todos os casos, o MVA e o HMMA ou superaram ou obtiveram acurácia similar às obtidas pelos outros métodos de previsão da Literatura, o que confirma que este trabalho irá contribuir para o campo da previsão de séries temporais não apenas no aspecto teórico, mas também na prática.

Summary

List of Figures

List of Tables

Agradecimentos

Aproveito essa oportunidade para manifestar toda minha gratidão ao meu bom Deus, a quem chamo de Pai amado, que sempre esteve comigo, celebrando em momentos de alegria e me amparando em momentos de tristeza. Louvo somente a Ele por ter me concedido a oportunidade de chegar até esse ponto da minha carreira.

Agradeço a minha esposa Vanessa Brian que sempre me deu todo o suporte para completar esse desafio do doutorado e, as minhas filhas Isabela e Elisa, que tiveram muita paciência e compreensão, sobretudo nos momentos em que estive ausente para desenvolver esse trabalho.

Agradeço a toda a minha família, dando um destaque à minha mãe, pai, minhas irmãs, minha tia Mirian, minha sogra, cunhadas, meus avós do coração e também aos meus pastores e pastoras da Igreja da Cidade em São José dos Campos, meus líderes de Célula e meus amigos mais chegados, que sempre estiveram comigo dando todo o suporte em oração e palavras de ânimo para que eu completasse esse trabalho.

1 Introduction

Among the strands in the field of Time Series Analysis is the creation of methods to study time-correlated data and make predictions (SHUMWAY; STOFFER, 2011). This is a dynamic research area that has attracted the attention of the scientific community over the past few decades (ADHIKARI; AGRAWAL, 2013). Applications of time series modeling and analysis encompass several fields of science, including: medicine (IMHOFF *et al.*, 1998), robotics (RADHAKRISHNAN *et al.*, 2015), cyber defense (ABDULLAH; PILLAI; CAI, 2015), defense strategy (EMMANOUILIDIS; KARPETIS, 2020), army mission analysis (MARUSICH; BUCHLER, 2016), finance (WEN, 2018), social sciences (REHMAN *et al.*, 2019), economics (KUNST; WAGNER, 2020), seismology (FILATOV; LYUBUSHIN, 2020) and criminology (GREENBERG, 2001). One way to estimate future terms of a time series is assuming the hypothesis that each observed data is somehow correlated with past data. Thus, it is natural to establish mathematical models that try to explain such correlations and describe the observed data as a function of time.

Autoregressive integrated moving average (ARIMA) is one of the most popular models employed in forecasting methods (BOX; JENKINS; REINSEL, 2013), (BOX, 2015), (ADHIKARI; AGRAWAL, 2013). This model is attributed to George Box and Gwilym Jenkins (BOX, 2015) and is essentially a linear regression model that describes movements of a univariate time series (BOX; JENKINS, 1970; HU, 2013; COCHRANE, 1997). ARIMA is a combination of the autoregressive (AR) (BOX; JENKINS, 1970) and the moving average (MA) (BOX; JENKINS, 1970; HU, 2013; HIPEL; MCLEOD, 1994) models applied to a integrated time series. If the time series is not integrated, together both produce the autoregressive moving average (ARMA) (BENJAMIN; RIGBY; STASINOPOULOS, 2003; BOX; JENKINS, 1970). In the case of a time series with seasonality, SARIMA can be applied (CARMONA-BENíTEZ; NIETO, 2020). There are also some extensions that are applicable to nonlinear (BAI *et al.*, 2020) or non-Gaussian (BENJAMIN; RIGBY; STASINOPOULOS, 2003) time series, however, in terms of implementation, those other are not comparable to the simplicity of ARIMA models.

Data mining is a relatively new field that has grown since the 1990s but was, in fact, recognized as a field of its own, in the first years of the twenty-first century (NISBET; MINER; ELDER, 2009). According to Zaki (2014) data mining is defined as the "process of automatic extraction of knowledge from large databases, using machine learning, pattern recognition, database techniques, and statistics". According to Makridakis, Spiliotis and Assimakopoulos (2018) the six pure Machine Learning methods submitted in the M4-competition performed poorly, none of them outperforming the combination approaches and only one achieved accuracy better than Naive2. Under the influence of forecasting competitions, many variations of combined methods have been created in the quest to achieve better forecasting accuracy (MAKRIDAKIS; SPILIOTIS; ASSIMAKOPOULOS, 2018). See these examples: in Egrioglu and Fildes (2020), a hybrid technique based on bagging approach is proposed to achieve more accurate forecasts;

Chindanur and B (2015) shows a combination between ARIMA and artificial neural network (ANN) to predict internet traffic data; Kaushik *et al.* (2020) calibrated an ensemble structure based on ARIMA, multilayer perceptron (MLP) and long short-term memory (LSTM) with the objective of predict healthcare expenditures; and Purohit *et al.* (2021) which presents some hybrid approaches, including mixing the exponential smoothing model and ANN in order to forecast the price of agricultural products.

A wealth of content can be found in the literature on time series data mining, some of which are based on complex networks (KRAMER *et al.*, 2009), (MAO; XIAO, 2019), (GAO *et al.*, 2016), (FERREIRA, 2017). In Mao and Xiao (2019), one can find an approach for time series forecasting based on complex network analysis. Specifically, this method is based on node similarities (LIU; Lü, 2010) extracted from a network constructed using the visibility graph method (LACASA *et al.*, 2008). In this work, it is referred to as the Mao-Xiao approach (MXA). Its accuracy is better than other methods for a variety of applications including methods using fuzzy logic (CHEN, 1996), (YU, 2005), bandwidth interval-based forecasting methods (PATHAK; SINGH, 2011), and hybrid fuzzy logic and visibility graph approach (ZHANG; ASHURI; DENG, 2017).

This study presents two new time series forecasting techniques. The MVA, Maximum Visibility Approach, is a new time series forecasting technique based on the complex network theory. The second is called HMMA, which is a hybrid methodology, that combines MVA and other forecasting method. This presentation follows the subsequent steps. Initially, some basic concepts needed to understand the proposed methods were revisited. Then, both methods, MVA and HMMA, are described. The results of the tests carried out with five different datasets are then presented, with the objective of comparing the performances obtained by the MVA, HMMA and the comparative methods: the state-of-the-art Mao-Xiao approach (MXA) (MAO; XIAO, 2019), dynamic auto-ARIMA, support vector regression (SVR) (SAMSUDIN; SHABRI; SAAD, 2010), long short-term memory (LSTM) (HOCHREITER; SCHMIDHUBER, 1997), the multilayer perceptron (MLP) (KAUSHIK *et al.*, 2020), hybrid additive ARIMA-ANN (CHINDANUR; B, 2015), hybrid additive ETS-ANN (PANIGRAHI; BEHERA, 2017) and naive estimation.

1.1 Motivations

In 2018, ITA hosted the first Workshop of Artificial Intelligence Applied to Finances and the article "Time series trend classification and forecasting using complex network analysis" was presented. This article was our first reading related to time series forecasting based on complex networks. According to Anghinoni *et al.* (2018), a real time series was mapped into a complex network and through its topological characteristics and community research an algorithm of trend detection could be proposed. Another important reading was Ferreira (2017) which provided theoretical background to better understanding the field of time series

Data Mining using Complex Network theory. Lastly, we found Mao and Xiao (2019) which presented a state-of-the-art approach to provide one-step-ahead time series forecasting also based on Complex Network Analysis, more specifically using link-prediction theory. According to Mao and Xiao (2019) their proposed method was able to generate forecasts more accurate than others in a variety of applications, including methods using fuzzy logic (CHEN, 1996), (YU, 2005), bandwidth interval-based forecasting methods on (PATHAK; SINGH, 2011), and hybrid fuzzy logic and visibility graph approach (ZHANG; ASHURI; DENG, 2017).

After studying the content of Mao and Xiao (2019), an opportunity arose to create other time series forecasting methods based on what was presented in the reference literature. In order to produce one-step-ahead forecast, Mao and Xiao (2019) uses the influence of the previous observation with the highest node similarity with the current observation. However, for most time series, it was observed that there are other nodes with the same level of similarity with the node associated to the current observation e all those information was not taking into account by Mao and Xiao (2019). Therefore, it was proposed a time series forecasting method named maximum visibility approach (MVA) which consider the influence of all previous observations that are statistically significant to the last observation. In addition, it was observed that MVA could be yet improved if it were combined with another well-accurate method, for example, one which is able to capture linear relations in order to obtain the one-step-ahead forecast. Therefore this author proposed the Hybrid Means of Multiple Approaches (HMMA) which combines the forecasting results obtained by two methods in order to find a new estimate.

1.2 Thesis Contributions

This work generated some contributions that deserve to be highlighted. Are they:

1. **Maximum Visibility Approach**
   Maximum Visibility Approach (MVA) is new forecasting technique, proposed by this author, based on the Complex Network theory and described in the section 4.1. This forecasting method proved to be applicable in different types of time series, especially for being able to capture non-linear time series patterns and use this information to produce one-step-ahead forecast.

2. **HMMA**
   Hybrid Means of Multiple Approaches (HMMA) is a new forecasting technique, proposed by this author, based on a hybrid formulation of one or more forecasting models. In this work HMMA is described in the section 4.2 and it was performed combining Maximum Visibility Approach with the estimate obtained by the second more accurate method for each time series analyzed in this work. HMMA proved to be suitable to produce forecasts considering time series with trend characteristics, seasonal variations, and non-stationary behaviors. It is important to point out that HMMA outperformed most of the comparative

methods considering in this research, including MXA (MAO; XIAO, 2019), which is a state-of-the-art method based on the Complex Network Theory and other hybrid methods.

1.3 Thesis Outlines

In this section we present how our text, composed by nine sections, is organized. In the section 1 we contextualized this work bringing facts about the field of Time Series, the ARIMA model, forecasting methods and data mining using complex networks theory. Then we presented this thesis objectives and the motivations for this research. In the section 2 we discuss some basic concepts regarding the theories of Time Series, Graphs, Complex Networks, means inequalities and the Natural Visibility Graph, which is a method of mapping time series into complex network. The section 3 contains brief presentations about methods of time series forecasting. In the sequence, comes section 4 where we present our proposed time series forecasting methods, followed by the section 5 where the results of this research are presented and discussed. Next, we state the conclusions of this thesis in the section 6, followed by the appendix section where most results directly presented in the body of the thesis are properly proofed.

2 Theoretical Background

This section provides a brief presentation of the basic concepts and techniques required to describe the proposed methods MVA and HMMA.

2.1 Time Series Theory

In this section we present some key definitions and concepts regarding the Time Series Theory, starting with its own definition. According to Hipel and McLeod (1994):

Definition 2.1 (Time Series)**.** *Time series is a set of observations, being each one, organized along the time.*

Let T be an index set and let $(\Omega, \mathcal{F}, P)$ be the space of probability , where each fundamental part is defined as follows (FULLER, 1996):

1. Ω represents the *sure event*, that is, the set of all possible outcomes of a given experiment. A real value of a time series is defined as $Y(t, \omega) \in T \times \Omega$, where $\omega \in \Omega$ is fixed and known as any *elementary event*. Therefore, if ω is unknown, $Y(t, \omega) = Y(t) = Y_t$ can be understood as one realization from the random process $\{Y_t : t \in \mathbb{Z}\}$, but it is also called as sample function, or a random variable.

2. $\mathcal{F}$ is called sigma-algebra and it is equal to $\mathcal{P}(\Omega)$ which represents the set of all subsets of Ω for those the probability can be evaluated.

3. P is the probability measure, or the expected long-run frequency, associated to each element which composes $\mathcal{F}$.

It is important to point out that Y_t is a set of random variables, because for each $t \in T$, one random variable is defined. The equation 2.1 shows a realization of a infinite *random process*.

$$\{Y_t : t \in \mathbb{Z}\}_{t \to -\infty}^{+\infty} = \{\cdots, Y_{-1} = y_{-1}, Y_0 = y_0, Y_1 = y_1, \dots\}. \tag{2.1}$$

Consider a n-element finite set of random variables $\mathbf{Y_t} = \{Y_1, Y_2, \cdots, Y_n, \}$. The joint distribution function of probability associated to $\mathbf{Y_t}$ is given by:

$$F_{Y_1, Y_2, \cdots, Y_n}(y_1, y_2, \cdots, y_n) = P(Y_1 \leq y_1, \cdots, Y_n \leq y_n,). \tag{2.2}$$

2.1.1 Expected Value and Variance

Let $Y : \Omega \to \mathbb{R}$ be the random variable defined on a space of probability $(\Omega, \mathcal{F}, P)$. Y is associated to a probability density function $f_Y(y)$, through whom the expected value of Y, $E[Y] = \mu_Y$, can be defined. In this work, μ_Y will be called the mean of Y . Generally, in such space of probability, using Lebesgue Integral notation, $E[Y]$ is defined, according to Löffler and Kruschwitz (2019), as following:

$$E[Y] = \int_{\Omega} Y(\omega) dP(\omega). \tag{2.3}$$

As a consequence of equation 2.3, one can observe that $E[\cdot]$ is a linear operator. In fact, considering any two real numbers α and β, and any two random variables, Y and X, from the same space of probability $(\Omega, \mathcal{F}, P)$, the next result yields:

$$E[\alpha X + \beta Y] = \alpha E[X] + \beta E[Y]. \tag{2.4}$$

The proof of the equation 2.4 is presented in appendix A.1. Similarly to what was done for the definition of expected value of Y, the concept of variance, $Var(Y)$, here denoted by σ_Y^2, can be defined as follows (LöFFLER; KRUSCHWITZ, 2019):

$$Var(Y) = \int_{\Omega} (Y(\omega) - E[Y])^2 dP(\omega). \tag{2.5}$$

Considering definition given by equation 2.5 and taking into account the result obtained from equation 2.4, the properties of the linear operator $E[\cdot]$ applies. Therefore:

$$Var(Y) = E\left[Y^2\right] - \mu_Y^2. \tag{2.6}$$

The proof of the equation 2.6 is presented in appendix A.2

2.1.2 Autocovariance and Autocorrelation

Let $\{Y_t\}_{t \to -\infty}^{+\infty}$ be a realization of a random process. Define now the vector $\mathbf{Y_t}$:

$$\mathbf{Y_t} = \begin{bmatrix} y_t \\ y_{t-1} \\ y_{t-2} \\ \vdots \\ y_{t-k} \end{bmatrix} \tag{2.7}$$

The vector $\mathbf{Y_t}$, defined by equation 2.7, is composed by the $t–th$ observation and its k previous ones. Naturally $\mathbf{Y_t}$ is a $(k+1)$-variate random variable and its distribution is a *joint*

distribution related to each single random variables $\{Y_t, Y_{t-1}, \cdots, Y_{t-k}\}$, given by

$$f_{Y_t,Y_{t-1},\cdots,Y_{t-k}}(y_t, y_{t-1}, \cdots, y_{t-k}). \tag{2.8}$$

After defining the multivariate distribution, presented by equation 2.8, define $\gamma_{t,k}$ as the *autocovariance of order* k, or the *autocovariance at lag* k (COCHRANE, 1997).

$$\gamma_{t,k} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\cdots\int_{-\infty}^{+\infty} (y_t - \mu_t)(y_{t-k} - \mu_t) f_{Y_t,\cdots,Y_{t-k}}(y_t, \cdots, y_{t-k}) \times dy_t dy_{t-1}\cdots dy_{t-k} = E\left[(Y_t - \mu_t)(Y_{t-k} - \mu_t)\right]. \tag{2.9}$$

Definition 2.2. *The autocovariance function (ACF) is the sequence of covariances of* Y_t *and its own lagged values* (HAMILTON, 1994).

The *autocovariance at lag* k can be re-written as:

$$\gamma_{t,k} = COV(Y_t, Y_{t-k}) = E\left[(Y_t - \mu_t)\cdot(Y_{t-k} - \mu_t)\right] \tag{2.10}$$

With the purpose of make $\gamma_{t,k}$ non-dimensional, it can be divided by $\sigma_t^2 = \gamma_{t,0}$, that is, the time series variance related to the time t. This procedure results in the *autocorrelation coefficient*, $\rho_{t,k}$ (COCHRANE, 1997) , which is limited to the interval $[-1, 1]$.

$$\rho_{t,k} = \frac{\gamma_{t,k}}{\sigma_t^2} \tag{2.11}$$

According to Box and Jenkins (1976), $\rho_{t,k}$ and $\gamma_{t,k}$ are defined as theoretical Autocorrelation Function (ACF) and theoretical Autocovariance Function (ACVF), respectively. Considering Hipel and McLeod (1994), the more appropriate estimator for $\hat{\gamma}_{t,k}$, related to a time series $\{y_1, y_2, \cdots, y_n\}$ is:

$$\hat{\gamma}_{t,k} = \frac{1}{n}\sum_{i=1}^{n-k}(y_i - \bar{y})(y_{i+k} - \bar{y}). \tag{2.12}$$

Considering $\hat{\gamma}_{t,k}$, defined by equation 2.12, after dividing it by $\hat{\gamma}_{t,0}$, thus comes the estimator for ACF.

$$\hat{\rho}_{t,k} = \frac{\hat{\gamma}_{t,k}}{\hat{\gamma}_{t,0}} \tag{2.13}$$

Given that $\hat{\rho}_{t,k}$ is a measure for correlation betweeen two random variables, comes:

$$|\hat{\rho}_{t,k}| \leq 1. \tag{2.14}$$

See proof for equation 2.14 in appendix A.3. The other statistical measure which quantifies how correlated $y_{t,n}$ is to each previous observation is the Partial Autocorrelation Function (PACF),

that is defined as the partial correlation of the random process $\{Y_t : t \in \mathbb{Z}\}$ at lag k (BOX *et al.*, 2016). PACF can be understood as the correlation between variables Y_n and Y_{n-k}, excluding the effect of the inner variables $Y_{n-1}, Y_{n-2}, \cdots, Y_{n-k+1}$.

Before the derivation of the PACF formula, it is necessary to discuss about *Linear Projection*, *Ordinary Least Square Regression* and how they are related.

2.1.3 Time Series Characteristics

2.1.3.1 Stationarity As stated on sections 2.1.1 and 2.1.2, a stochastic process can be written as a random variable at each point in time. According to discussed in Hipel and McLeod (1994), when the joint distribution of any possible combinations of random variables of the time series process is not impacted by shifting this particular set backwards or forwards in time, i.e., the joint distribution does not depends on time, then the stochastic process is called ***strictly stationary***.

$$F_{Y_1,Y_2,\cdots,Y_n}(y_1, y_2, \cdots, y_n) = F_{Y_{t+1},Y_{t+2},\cdots,Y_{t+n}}(y_{t+1}, y_{t+2}, \cdots, y_{t+n}), \forall t. \tag{2.15}$$

In the case of strictly stationary time series the expressions for expected value and covariance, defined on sections 2.1.1 and 2.1.2, can be considered time independent, hence $\mu_t = \mu$ and $\gamma_{t,k} = \gamma_k$, $\forall t$ and any k. However, in practice, not always the hypothesis of strict stationarity are needed, therefore, if the next pair of conditions apply, the time series is called *weak stationary*.

1. The expected value of Y_t is a constant for all $t \in T$.
2. $(Y_{t_1}, Y_{t_2}, \cdots, Y_{t_n})$ and $(Y_{t_1+h}, Y_{t_2+h}, \cdots, Y_{t_n+h})$ have the same covariance matrices, for all $h \in \mathbb{N}$ and all $t_1, t_2, \cdots, t_n, t_1 + h, t_2 + h, \cdots, t_n + h$ in the set T.

Hipel and McLeod (1994) also stated that a process is ***order-k-weak-stationary*** when the statistical moments of a given time series till order k is only dependent on time differences and not dependent on the time of occurrence of the data for which the moments were estimated.

Notice that in this case, the covariance between Y_n and Y_{n-k} depends only on the lag between both random variables, hence, covariance is shaped as a even-function, because, as Y_{n-k} and Y_{n+k} have the same distance to Y_n, as consequence, $\gamma_k = \gamma_{-k}$ (HAMILTON, 1994). Considering the concept of weak stationarity, equation 2.9 can be re-visited and the autocovariance at lag k for a stationary stochastic process can be given by:

$$\gamma_k = E\left[(Y_n - \mu)\left(Y_{n-k} - \mu\right)\right]. \tag{2.16}$$

The autocorrelation at lag k for a stationary stochastic process can be calculated as:

$$\rho_k = \frac{E\left[\left(Y_n - \mu\right)\left(Y_{n-k} - \mu\right)\right]}{\sigma^2}. \tag{2.17}$$

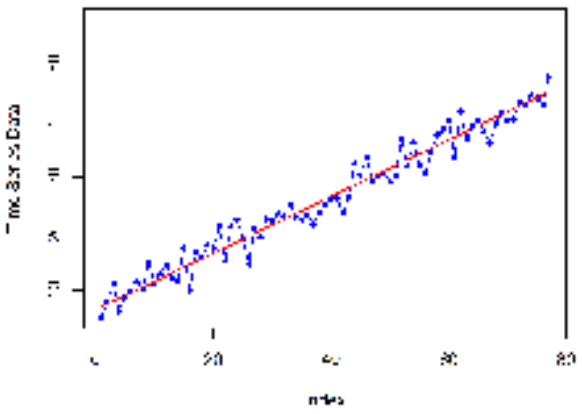
For both equations 2.16 and 2.17, given the stochastic process is stationary, $E\left[Y_k\right] = \mu$ and Var $\left[Y_k\right] = \sigma^2$. Also, for this type of stochastic process, the equations 2.12 and 2.13 can be re-written as:

$$\hat{\gamma}_k = \frac{1}{n}\sum_{i=1}^{n-k}\left(y_i - \bar{y}\right)\left(y_{i+k} - \bar{y}\right) \tag{2.18}$$

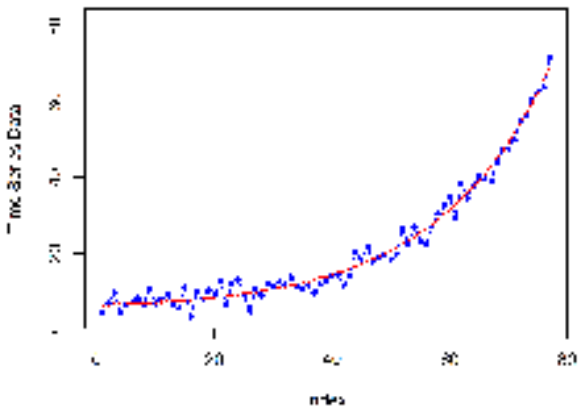
and

$$\hat{\rho}_k = \frac{\hat{\gamma}_k}{\hat{\gamma}_0}. \tag{2.19}$$

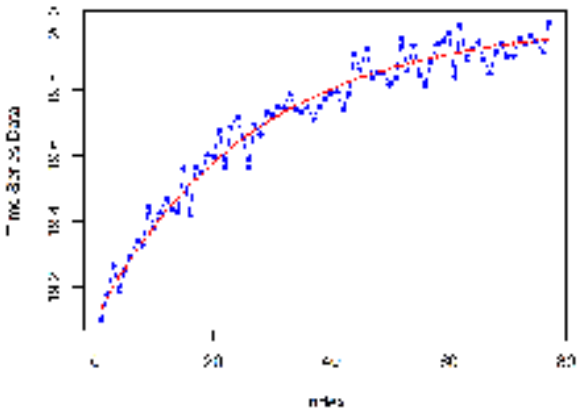
2.1.3.2 Trend Behavior A trend exists when there is a long-term increase or decrease in the data (HYNDMAN; ATHANASOPOULOS, 2018), that is, the general tendency of a time series to increase, decrease or stagnate over a long period of time is termed as *secular trend* or simply *trend* (ADHIKARI; AGRAWAL, 2013). There are various kinds of trends, for example, linear (Figure2.1a), exponential (Figure2.1b) and damped (Figure2.1c). We see upward trend in series informing Brazilian population from 1950 to 2021 (WORLDPOPULATIONREVIEW, 2021), number of cars manufactured per year in Brazil from 1998-Dec-01st to 2008-Aug-01st (TRADINGECONOMICS, 2021), or the number of passengers in a London underground station from 6am to 8am in a business day (IELTS, 2020). Examples of downward trend can be found in series relating infant mortality rate in South Korea from 1950 to 2020 (MACROTRENDS, 2021), or the IBOVESPA closing prices from 2020-Feb-19th to 2020-Mar-23rd (INVESTING, 2021).



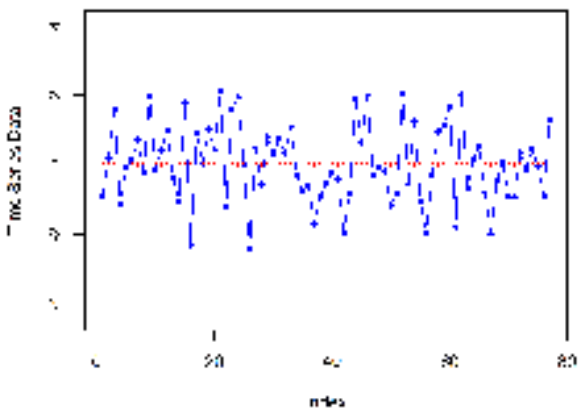
(a) Time series with linear trend



(b) Time series with exponential trend



(c) Time series with damped trend



(d) Time series without trend

Figure 2.1: Time series with different types of trends. Source: Drawn by the author.

2.1.3.3 Seasonality A seasonal pattern can be observed when the time series presents regular and predictable changes that recur periodically, for example, in a specific month of the year or hour of the day. Any predictable movement or pattern that recurs or repeats in the same frequency is said to be seasonal. Seasonality is characterized by a fixed and known frequency which produces linear or, mostly often, nonlinear components (see Hylleberg (1992) and Box *et al.* (2015)) that changes with the time and does repeat. As an example, we show the time series related to the monthly totals of accidental deaths in the USA from 1973 to 1978, got from R-dataset package, but also presented by Brockwell and Davis (1991). This is a time series with 72 observations, and visually stationary, no-trend and seasonal.

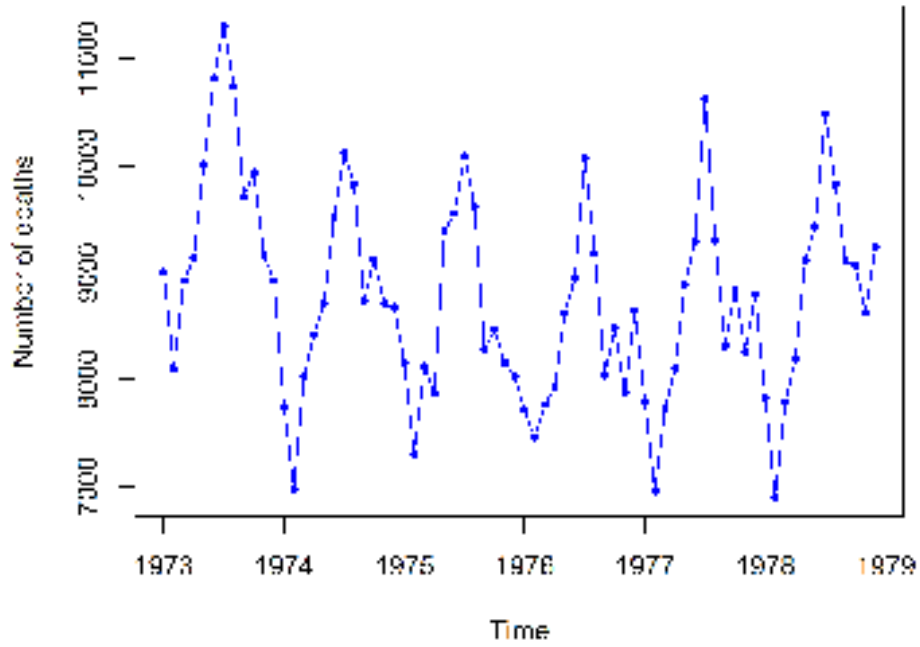


Figure 2.2: Monthly accidental deaths in the USA from 1973 to 1978. Data Source: Brockwell and Davis (1991). Figure Source: Done by the author.

One can observe from Figure 2.2 that the number of accidental deaths in the USA reaches the peak and valley in every July and February, respectively. This indicates a seasonal time series. The seasonality can also be confirmed through the autocorrelation function related to the time series. See in the Figure 2.3 that the peaks occurs at every lag multiple of 12. Given this result, we can state that this time series is seasonal with period 12.

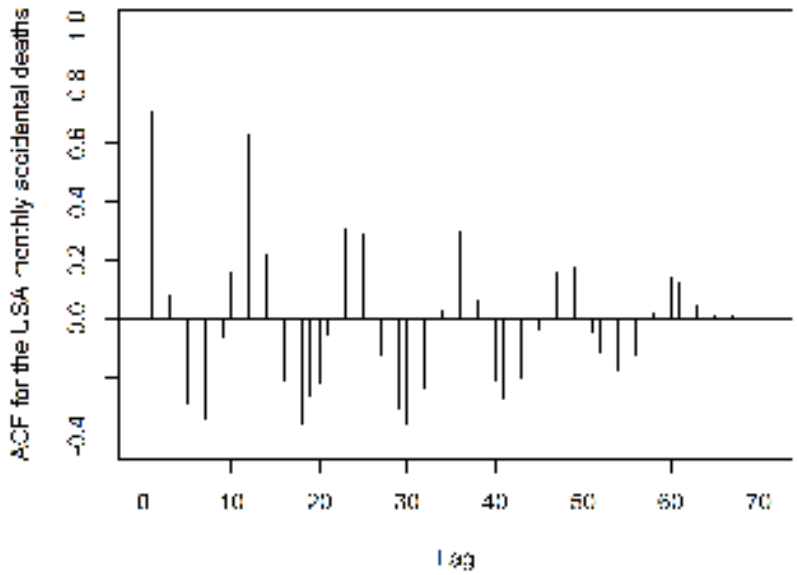


Figure 2.3: Autocorrelation function for the time series of the monthly accidental deaths in the USA from 1973 to 1978. Data Source: Brockwell and Davis (1991). Figure Source: Done by the author.

2.1.4 Principles of Forecasting

Consider that one is interested in forecast the one-step-ahead term, Y_{n+1}, of a given time series, based on Y_n and in its first $(k-1)$ lags. For that, define the k-vector $\mathbf{Y_{n,k}}$ as:

$$\mathbf{Y_{n,k}} = \left(Y_n, Y_{n-1}, \cdots, Y_{n-k+1}\right)^{\mathrm{T}}, \tag{2.20}$$

where the subscript $\mathbf{T}$ means the transpose matrix operation. Let $\hat{Y}_{n+1|k}$ be the estimator for Y_{n+1} based on $\mathbf{Y_{n,k}}$. In order to evaluate the quality of the forecast it is necessary consider a *loss function*. According to Hamilton (1994), a very convenient choice is the quadratic *loss function*, known as *Mean Squared Error* (MSE). The idea is to find $\hat{Y}_{n+1|k}$ which minimizes

$$\text{MSE}\left(\hat{Y}_{n+1|k}\right) = E\left[\left(Y_{n+1} - \hat{Y}_{n+1|k}\right)^2\right]. \tag{2.21}$$

Given that the forecast is based on $\mathbf{Y_{n,k}}$, so $\hat{Y}_{n+1|k}$ must be a function of $\mathbf{Y_{n,k}}$. Let $g\left(\mathbf{Y_{n,k}}\right)$ be the function which minimizes MSE $\left(\hat{Y}_{n+1|k}\right)$. According to Hamilton (1994) a promising candidate for forecasting rule is

$$g\left(\mathbf{Y_{n,k}}\right) = E\left[Y_{n+1}|\mathbf{Y_{n,k}}\right]. \tag{2.22}$$

The proof for the result presented in equation 2.22 is shown in appendix A.5.

2.1.5 Forecasts based on Linear Projection

Consider the linear model to estimate Y_{n+1} given by:

$$\hat{Y}_{n+1|k} = \sum_{i=1}^{n-k} \beta_i Y_{n+1-i} = \boldsymbol{\beta}^{\mathbf{T}} \mathbf{Y}_{\boldsymbol{n,k}}. \tag{2.23}$$

Minimizing the mean squared error (MSE),

$$E\left[\left(\hat{Y}_{n+1|k} - \boldsymbol{\beta}^{\mathbf{T}} \mathbf{Y}_{\boldsymbol{n,k}}\right)^2\right] \tag{2.24}$$

the vector $\boldsymbol{\beta}$ can be calculated.

$$\hat{\boldsymbol{\beta}} = E\left[\mathbf{Y}_{\boldsymbol{n,k}} \mathbf{Y}_{\boldsymbol{n,k}}^{\mathbf{T}}\right]^{-1} E\left[\hat{Y}_{n+1|k} \cdot \mathbf{Y}_{\boldsymbol{n,k}}\right]. \tag{2.25}$$

See appendix A.6 and follow the derivation of $\hat{\boldsymbol{\beta}}$. According to Rao (2021) the result of $\hat{\boldsymbol{\beta}}$ can also be written as:

$$\hat{\boldsymbol{\beta}} = Var\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)^{-1} COV\left[\hat{Y}_{n+1|k} \cdot \mathbf{Y}_{\boldsymbol{n,k}}\right]. \tag{2.26}$$

Let $\hat{\boldsymbol{\beta}}_{\boldsymbol{LP}}$ be the estimator shown in equation 2.25. As one can observe, $\hat{\boldsymbol{\beta}}_{\boldsymbol{LP}}$ is obtained considering the characteristics of the distribution of $\mathbf{Y}_{\boldsymbol{n,k}}$. In practice, a sample of the past values of a time series is all the known information about this time series, thus consider the model:

$$\begin{pmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_{n-k+1} \end{pmatrix} = \begin{pmatrix} y_{n-1} & y_{n-2} & \cdots & y_{n-k} \\ y_{n-2} & y_{n-3} & \cdots & y_{n-k-1} \\ \vdots & \vdots & \cdots & \vdots \\ y_k & y_{k-1} & \cdots & y_1 \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_k \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_{n-k} \end{pmatrix}$$

or

$$\mathbf{Y}_{\boldsymbol{n,k}} = \mathbf{Y}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{2.27}$$

where $n > 2k$. Let $SSE\left(\hat{\boldsymbol{\beta}}\right)$ be the sum of squared error (SSE) obtained considering $\mathbf{Y}_{\boldsymbol{n,k}}$ and the estimate $\hat{\boldsymbol{Y}}_{\boldsymbol{n,k}} = \mathbf{Y}\hat{\boldsymbol{\beta}}$.

$$SSE = \sum_{i=1}^{n-k} \epsilon_i^2 = \sum_{i=1}^{n-k} \left(y_i - \mathbf{Y}_i^{\mathrm{T}} \hat{\boldsymbol{\beta}}\right)^2 = \left(\mathbf{Y}_{\boldsymbol{n,k}} - \mathbf{Y}\hat{\boldsymbol{\beta}}\right)^{\mathbf{T}} \cdot \left(\mathbf{Y}_{\boldsymbol{n,k}} - \mathbf{Y}\hat{\boldsymbol{\beta}}\right). \tag{2.28}$$

According to Rencher (2002), applying the technique of the *Ordinary Least Squared*, the estimate of $\hat{\boldsymbol{\beta}}$ which minimizes the sum of squared error is given by

$$\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}} = \left(\mathbf{Y}^{\mathbf{T}}\mathbf{Y}\right)^{-1}\mathbf{Y}^{\mathbf{T}}\mathbf{Y}_{\boldsymbol{n,k}}. \tag{2.29}$$

It can be found in the pages 325 and 326 from Rencher (2002) a demonstration that no other estimator $\hat{\boldsymbol{\beta}}$ can achieve a smaller SSE than that one obtained from the result expressed by equation 2.29.

It is important to compare the values of $\hat{\boldsymbol{\beta}}$ shown in equations 2.29 and 2.25. As one can observe, $\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}}$ is calculated based on sample measurements. According to Hamilton (1994), there is a formal mathematical sense in which the two estimators are the same. More specifically, if the stochastic process $\{\mathbf{Y}_{\boldsymbol{n,k}}, Y_{n+1}\}$ is covariance-stationary and ergotic for the second moments, therefore the sample moments will converge to the population moments when $n \to \infty$. Hence:

$$\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}} \to \hat{\boldsymbol{\beta}}_{\boldsymbol{LP}} \tag{2.30}$$

The derivation for the optimal sum of squared errors, is shown in appendix A.7. Applying the calculated $\hat{\boldsymbol{\beta}}$ in the equation 2.28, comes:

$$SSE\left(\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}}\right) = \mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\mathbf{Y}_{\boldsymbol{n,k}} - \mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\mathbf{Y}\left(\mathbf{Y}^{\mathbf{T}}\mathbf{Y}\right)^{-1}\mathbf{Y}^{\mathbf{T}}\mathbf{Y}_{\boldsymbol{n,k}}. \tag{2.31}$$

According to Shumway and Stoffer (2011), the ordinary least squares estimators are unbiased, that is, $E\left[\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}}\right] = \boldsymbol{\beta}_{\boldsymbol{OLS}}$, and have the smallest variance within the class of linear unbiased estimators. In the case of $\boldsymbol{\epsilon_i} \sim N(0, \sigma^2_\epsilon)$ for all $i \in \{1, 2, \cdots, n-k\}$, therefore $\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}}$ is also the maximum likelihood estimator for $\boldsymbol{\beta}_{\boldsymbol{OLS}}$ and is normally distributed with

$$COV\left(\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}}\right) = \sigma^2_\epsilon\left(\mathbf{Y}^{\mathbf{T}}\mathbf{Y}\right)^{-1}. \tag{2.32}$$

As calculated in Shumway and Stoffer (2011), an unbiased estimator, s^2_ϵ, for the variance σ^2_ϵ, is:

$$s^2_\epsilon = MSE\left(\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}}\right) = \frac{SSE\left(\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}}\right)}{n-2k}. \tag{2.33}$$

2.1.6 Partial Autocorrelation Function in Time Series

Definition 2.3. *Suppose that $\{Y_n\}_t$ is a time series. The partial correlation between Y_n and Y_{n-k} is defined as the correlation between Y_n and Y_{n-k} excluding the influence of the in-between set $\mathbf{Y}_{ex}$ composed by $\mathbf{Y}_{ex} = (Y_{n-1}, Y_{n-2}, \cdots, Y_{n-k+1})$* (RAO, 2021).

Be $P_{\mathbf{Y}_{ex}}(Y_n)$ and $P_{\mathbf{Y}_{ex}}(Y_{n-k})$ the linear predictors of Y_n and Y_{n-k}, respectively, through $\mathbf{Y}_{ex}$. The $P_{\mathbf{Y}_{ex}}(Y_i)$ is the projection of the random variable Y_i onto the space spanned by $\mathbf{Y}_{ex}$ and it is the linear combination expressed by equation 2.23, which minimizes the MSE, shown

in equation 2.24 (RAO, 2021), (BOX *et al.*, 2016). Thus, $P_{\mathbf{Y}_{ex}}(Y_n)$ and $P_{\mathbf{Y}_{ex}}(Y_n)$ can be given by:

$$P_{\mathbf{Y}_{ex}}(Y_n) = \left[Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_n, \mathbf{Y}_{ex}\right)\right]^{\mathrm{T}} \cdot \mathbf{Y}_{ex}, \tag{2.34}$$

and

$$P_{\mathbf{Y}_{ex}}(Y_{n-k}) = \left[Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_{n-k}, \mathbf{Y}_{ex}\right)\right]^{\mathrm{T}} \cdot \mathbf{Y}_{ex}. \tag{2.35}$$

Define the residual of Y_i, the difference $\epsilon_i = Y_i - P_{\mathbf{Y}_{ex}}(Y_i)$. The partial correlation between variables Y_n and Y_{n-k} is the covariance between its residuals, normalized by the products of its standards deviations (HAMILTON, 1994), (RAO, 2021).

$$\rho_{Y_n Y_{n-k} \backslash \mathbf{Y}_{ex}} = \frac{COV\left(\left(Y_n - P_{\mathbf{Y}_{ex}}(Y_n)\right), \left(Y_{n-k} - P_{\mathbf{Y}_{ex}}(Y_{n-k})\right)\right)}{\sqrt{Var(Y_n - P_{\mathbf{Y}_{ex}}(Y_n))}\sqrt{Var(Y_{n-k} - P_{\mathbf{Y}_{ex}}(Y_{n-k}))}} \tag{2.36}$$

The appendices A.8 and A.9, respectively, show the derivations of the covariance in the numerator of equation 2.36 and the variances in the denominator of this same equation. The result for $COV\left(\left(Y_n - P_{\mathbf{Y}_{ex}}(Y_n)\right), \left(Y_{n-k} - P_{\mathbf{Y}_{ex}}(Y_{n-k})\right)\right)$ is:

$$\gamma_{Y_n Y_{n-k} \backslash \mathbf{Y}_{ex}} = COV\left(Y_n, Y_{n-k}\right) - COV\left(Y_n, \mathbf{Y}_{ex}\right)^{\mathrm{T}} Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_{n-k}, \mathbf{Y}_{ex}\right). \tag{2.37}$$

The variances of $\left(Y_n - P_{\mathbf{Y}_{ex}}(Y_n)\right)$ and $\left(Y_{n-k} - P_{\mathbf{Y}_{ex}}(Y_{n-k})\right)$ are given by:

$$Var\left(\epsilon_n\right) = Var\left(Y_n\right) - COV\left(Y_n, \mathbf{Y}_{ex}\right)^{\mathrm{T}} Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_n, \mathbf{Y}_{ex}\right), \tag{2.38}$$

and,

$$\begin{aligned} Var\left(\epsilon_{n-k}\right) &= \\ &= Var\left(Y_{n-k}\right) - \left[COV\left(Y_{n-k}, \mathbf{Y}_{ex}\right)\right]^T Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_{n-k}, \mathbf{Y}_{ex}\right). \end{aligned} \tag{2.39}$$

The partial correlation between variables Y_n and Y_{n-k}, $\rho_{Y_n Y_{n-k} \backslash \mathbf{Y}_{ex}}$ is calculated by replacing the last three results, expressed by equations 2.37, 2.38 and 2.39, in the equation 2.36.

2.1.7 Ergodic Process

Let $\bar{y}$ be a sample mean of a n-element covariance-stationary random process $\{Y_t\}$.

Definition 2.4. *A random process is said to be ergodic for the mean, if* $\bar{y} \to E\left[Y_t\right]$ *and the autocovariances tends to zero, sufficient quickly as* $n \to \infty$ (HAMILTON, 1994).

Naturally, a consequence of a covariance-stationary random process be ergodic is that

$$\frac{1}{n-k}\sum_{i=k+1}^{n}\left(Y_i-\mu\right)\left(Y_{i-k}-\mu\right)\rightarrow\gamma_k, \tag{2.40}$$

for all k.

2.1.8 White Noise

Definition 2.5. *The white noise process is the sequence* $\{\epsilon_t\}_{t=-\infty}^{+\infty}$ *of uncorrelated random variables with zero mean and constant variance* σ^2*, therefore* $E\left[\epsilon_t\right]=0$*,* $E\left[\epsilon_t^2\right]=\sigma^2$ *and* $E\left[\epsilon_t\cdot\epsilon_\tau\right]=0$ *for* $t\neq\tau$*. If the random variables comes from a normal distribution, so the white noise is called Gaussian White Noise and* $\epsilon_t\sim N(0,\sigma^2)$ (SHUMWAY; STOFFER, 2011),(HAMILTON, 1994).

The Figure 2.4 illustrates the example of one realization of a Gaussian white-noise process which is normally distributed with zero mean and variance 1.

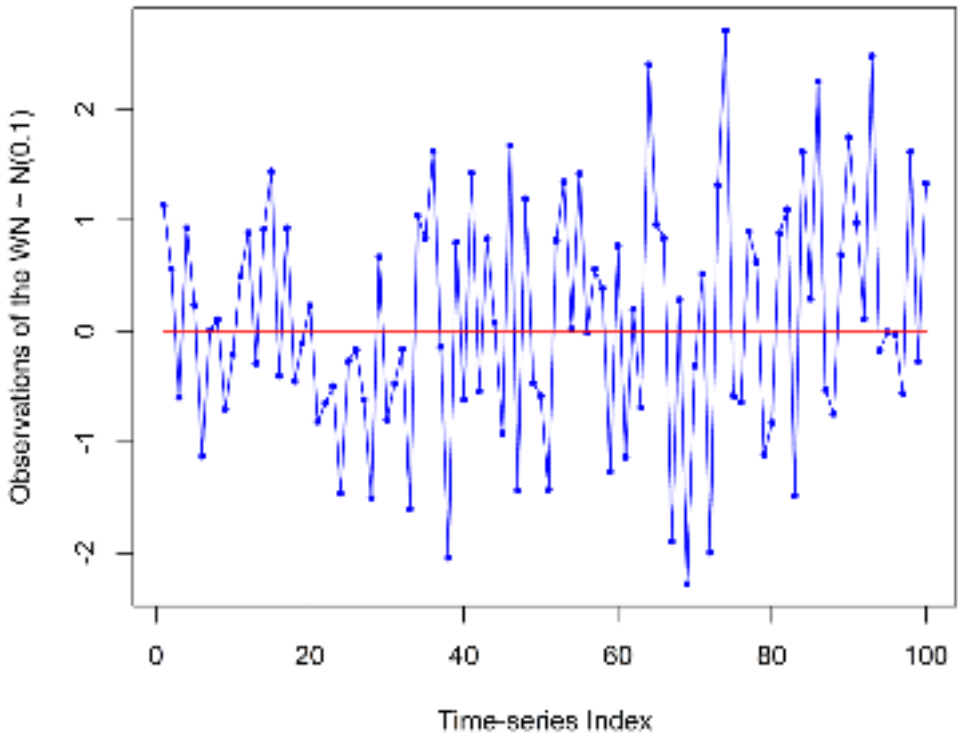


Figure 2.4: One realization of the Gaussian white-noise process N(0,1)

2.1.9 Random Walk Process

The random walk process is an stochastic sequence $\{S_n\}$, starting with $S_0=0$ and defined by the following law (RAO, 2021), (SHUMWAY; STOFFER, 2011):

$$y_t=y_{t-1}+\delta+\epsilon_t,\qquad t\in\{1,2,3,...\} \tag{2.41}$$

where δ is a real constant and ϵ_t is part of a white-noise process, with zero mean and constant variance. In the case of $\delta = 0$, this process is called a simple one-dimensional random walk process (SHUMWAY; STOFFER, 2011), but in the other hand, whether $\delta \neq 0$, the process is a random walk with drift, or a trend-random-walk-process. By summing y_t using its recurrence equation, y_t can be re-written in the form:

$$y_t = \delta t + \sum_{k=1}^{t} \epsilon_k. \tag{2.42}$$

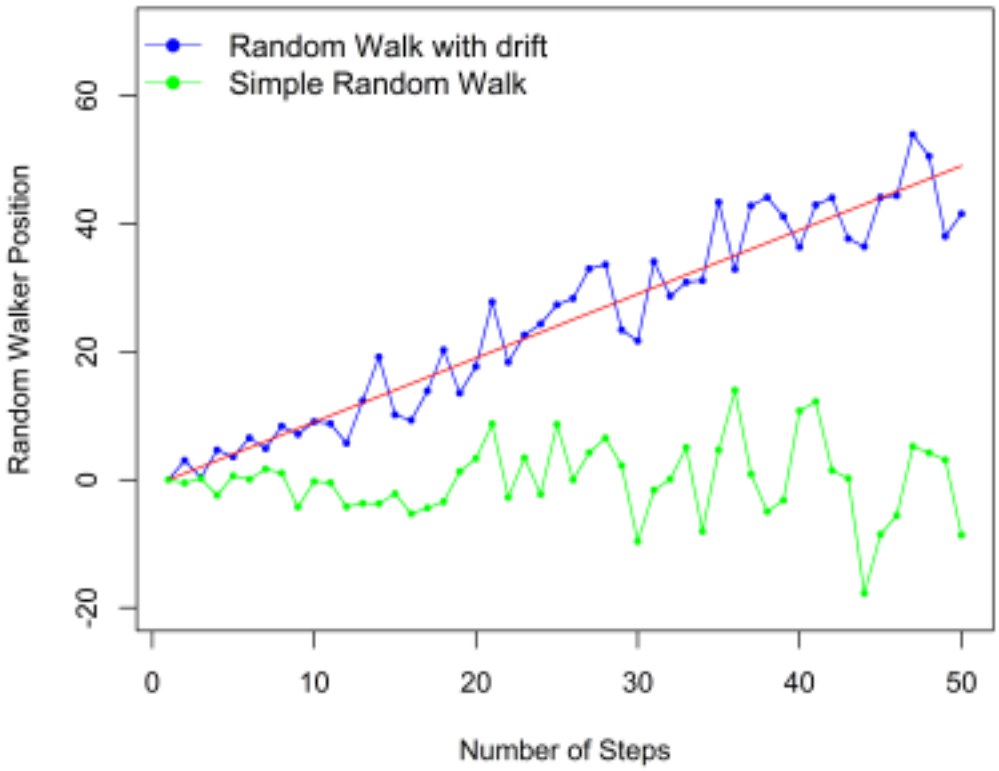


Figure 2.5: One realization of the random walk process with drift ($\delta = 1$) and $\epsilon_t \sim N(0,1), \forall t \in \{1, 2, \cdots, 50\}$, and of the simple random walk considering the same ϵ distribution.

One can observe in Figure 2.5 that for both random walkers, the variances are increasing as t increases, indeed, because given the equation 2.42, the following comes:

$$Var\left(y_t\right) = t \overbrace{\sigma_\epsilon^2}^{=1} = t, \tag{2.43}$$

which makes the random walk process is classified as a non-stationary process, since its variance is not constant with t. Also, the expected value is not constant.

$$E\left[y_t\right] = \delta t. \tag{2.44}$$

The autocovariance of the random walk process is given by:

$$\gamma_k = COV\left(y_n, y_{n-k}\right) = COV\left(\sum_{k=1}^{n}\epsilon_k, \sum_{k=1}^{n-k}\epsilon_k\right) = (n-k)\sigma_\epsilon^2, \tag{2.45}$$

To read the derivations of the results expressed in equations 2.43, 2.44 and 2.45, see appendix A.10.

2.1.10 Hypothesis Tests in Time Series Analysis

Normally, forecasting models and methods to evaluate the quality of the forecasting results can only be applied under specific time series conditions, for example, stationarity, seasonal behavior and white noise characteristics. The confirmation about whether the time series behaves as necessary is done based on the results of proper hypothesis tests. In the next four sections four useful tests are described. The augmented Dickey-Fuller test, for time series stationarity; the Kwiatkowski-Phillips-Schmidt-Shin test which confirms level or trend stationarity; the Ljung-Box test which verify independence between data; and Diebold-Mariano test which indicate if two forecasting methods produce the same level of accuracy.

2.1.10.1 Augmented Dickey–Fuller Test - ADF: The Augmented Dickey-Fuller (ADF) test is related to stationarity, it is applied to verify whether a time series is stationary or not. The ADF test assumes the presence of a unit root as the null hypothesis (FULLER, 1996). As alternative hypothesis, data are level or trend stationary (FULLER, 1996). Equation (2.46) represents the model assumed in the ADF test.

$$\begin{aligned} y_t &= \delta y_{t-1} + \mu + \gamma t + \epsilon_t + \\ &+ (\phi_1 \Delta y_{t-1} + \phi_2 \Delta y_{t-2} + \cdots + \phi_k \Delta y_{t-k}) \end{aligned} \tag{2.46}$$

The Dickey-Fuller test only contains the first lag as the differentiating term, that is, $\phi_1 \Delta y_{t-1}$. The Augmented Dickey-Fuller test model, in turn, considers the first k lags and provides more thoroughness to the test. According to Fuller (1996), the statistic of the test is given by:

$$ADF = \frac{\hat{\delta}}{\hat{\sigma}_{\hat{\delta}}}, \tag{2.47}$$

where $\hat{\delta}$ is the estimator of the coefficient δ presented in (2.46) and $\hat{\sigma}_{\hat{\delta}}$ is the corresponding estimation of the standard deviation of $\hat{\delta}$. The null and alternative hypotheses are (FULLER, 1996):

H_0**:** $\delta = 1$, for presence of unit root,

H_A**:** $\delta < 1$, for trend stationarity.

The test points to a stationary time series. If the test statistic is smaller than the critical value, the null hypothesis must be rejected and the time series is assumed to be stationary.

2.1.10.2 Kwiatkowski-Phillips-Schmidt-Shin (KPSS) Test: The Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test verifies the trend or level stationarity (KWIATKOWSKI *et al.*, 1992). Some time series do not present stationarity around zero, but can present stationarity around a deterministic trend, and KPSS is able to point out if this behavior occurs. Given a time series $y_1, y_2, \cdots, y_n$, Kwiatkowski *et al.* (1992) says that these data can be broken into three terms: trend, random walk and stationary residual.

$$y_t = (\mu + \gamma t) + (\delta y_{t-1} + u_t) + \epsilon_t \tag{2.48}$$

In (2.48), taking γ as a real number, γt is the trend term, u_t is an independent and identically distributed random variable with zero mean and variance σ_u^2, and ϵ_t is the stationary residual term, which is also a white noise distribution with zero mean and variance σ_ϵ^2. The null and alternative hypotheses are defined as follows (KWIATKOWSKI *et al.*, 1992):

H_0**:** $\sigma_u^2 = 0$, for trend stationary. Or:

H_0**:** $\sigma_u^2 = 0$ and $\mu = 0$, for level stationary. Against

H_A**:** $\sigma_u^2 > 0$, presence or unit root, that is, non-stationarity.

Consider $\hat{\sigma}_\epsilon^2$ as a consistent estimator of the long-run variance σ_ϵ^2 of the residuals (KOKOSZKA; YOUNG, 2016). For second-order stationary processes, the long-run variance is defined as the sum of all autocovariances (MüLLER, 2007). Assuming $S_n = \sum_{i=1}^{n} \epsilon_i$, the statistic of the KPSS test can be given by Kwiatkowski *et al.* (1992)

$$KPSS_n = \frac{1}{n^2 \hat{\sigma}_\epsilon^2} \sum_{k=1}^{n} S_n^2. \tag{2.49}$$

We can observe in Kokoszka and Young (2016) discussions around the asymptotic distribution of $KPSS_n$.

2.1.10.3 Ljung-Box Test: The Ljung-Box test is performed to assess if a given dataset is independently distributed. The null hypothesis considers the independence of data, and the alternative hypothesis is that there is some dependency among the given data. Equation (2.50), given by Ljung and Box (1978), shows the statistic of the Ljung-Box test.

$$Q = n(n+2) \sum_{k=1}^{h} \frac{\hat{\rho}_k^2}{n-k} \tag{2.50}$$

where

$$\hat{\rho}_k = \frac{\sum_{i=k+1}^{n} y_i y_{i-k}}{\sum_{i=1}^{n} y_i^2}. \tag{2.51}$$

In (2.50), n is the data length, $\hat{\rho}_k$ is the sample autocorrelation at lag k, and h is the maximum lag considered in the test. According to Ljung and Box (1978), the statistic Q follows a χ^2_h distribution: therefore, for a significance level α, the null hypothesis is rejected if $Q > \chi^2_{1-\alpha,h}$.

The Ljung-Box test can be applied to the residuals obtained by an ARIMA(p, d, q) model, defined in section 3.9, to test the quality of the forecasting results. It is done because, in order to confirm that an ARIMA model gave its best forecasting results, its residuals must behave as a white noise process, therefore, the residuals must be independent. According to Ljung and Box (1978), in this case, the chi-squared degrees of freedom, η, has to be updated from $\eta = h$ to $\eta = h - p - q$.

2.1.10.4 Diebold-Mariano Test: Consider two sets of estimates $\{\hat{y_{it}}\}_{t=1}^{n}$ and $\{\hat{y_{jt}}\}_{t=1}^{n}$ for a time series $\{y_t\}_{t=1}^{n}$. Let $\{e_t\}_{t=1}^{n}$ be the associate forecast errors. Also, let the loss associated with forecast *i* be a function of the forecast error, e_{it}, and denoted by $g\left(e_{it}\right)$. This loss function, $g(\cdot)$, is such as $g(0) = 0$, it is never negative and the higher is the magnitude of the error, the greater is $g(\cdot)$. Normally $g\left(e_{it}\right) = e_{it}^2$ or $g\left(e_{it}\right) = |e_{it}|$.

Let $d_t = g\left(e_{it}\right) - g\left(e_{jt}\right)$ be the loss-differential between the two forecasts. According to Diebold and Mariano (1995), the Diebold-Mariano test points out the null hypothesis $E\left[d_t\right] = 0$, which is equivalent of "equal accuracy" between the two sets of estimators. If the null hypothesis is not rejected, so it is said that both estimators produce the same level of accuracy statistically speaking.

Consider the sample $\{d_t\}_{t=1}^{n}$ of a loss-differential series. If $\{d_t\}_{t=1}^{n}$ is covariance stationary and presents short memory, hence (DIEBOLD; MARIANO, 1995):

$$\sqrt{n}\left(\bar{d} - \mu\right) \sim N\left(0, 2\pi f_d(0)\right) \tag{2.52}$$

where

$$\bar{d} = \frac{1}{n}\sum_{t=1}^{n} d_t \tag{2.53}$$

is the sample mean of the loss differential series, $E\left[d_t\right] = \mu$, and

$$f_d(0) = \frac{1}{2\pi}\sum_{i=-\infty}^{\infty} \gamma_d(k) \tag{2.54}$$

is the autocovariance at lag k of the loss differential series. Given equation 2.52, comes:

$$\frac{\bar{d}-\mu}{\sqrt{\frac{2\pi f_d(0)}{n}}} \sim N(0,1) \tag{2.55}$$

but considering a sufficient large sample,

$$\frac{\bar{d}-\mu}{\sqrt{\frac{2\pi \hat{f}_d(0)}{n}}} \sim N(0,1), \tag{2.56}$$

where $\hat{f}_d(0)$ is a consistent estimator for $f_d(0)$. Therefore, the statistic to test $H_0 : \mu = 0$ is given by

$$S = \frac{\bar{d}}{\sqrt{\frac{2\pi \hat{f}_d(0)}{n}}} \sim N(0,1). \tag{2.57}$$

According to Diebold and Mariano (1995) a proper estimator for $f_d(0)$ is:

$$\hat{f}_d(0) = \frac{1}{2\pi} \sum_{i=-k+1}^{k-1} \Theta\left(\frac{k}{h-1}\right) \hat{\gamma}_d(k) \tag{2.58}$$

where $\Theta\left(k(h-1)^{-1}\right) = 1$ if $|k(h-1)^{-1}| \leq 1$ or $\Theta\left(k(h-1)^{-1}\right) = 0$, otherwise. Also, h is the forecasting-ahead horizon and

$$\hat{\gamma}_d(k) = \frac{1}{n} \sum_{i=|k|+1}^{n} \left(d_i - \bar{d}\right)\left(d_{i-|k|} - \bar{d}\right). \tag{2.59}$$

In this thesis, we are providing one-step-ahead forecasting only, therefore, for our purposes, $h = 1$, hence, in our case all Diebold-Mariano test were calculated considering

$$\hat{f}_d(0) = \frac{\hat{\gamma}_d(0)}{2\pi} = \frac{1}{2\pi n} \sum_{i=1}^{n} \left(d_i - \bar{d}\right)^2. \tag{2.60}$$

2.1.10.5 Webel-Ollech Test: With the objective of identifying the data seasonality Webel and Ollech (2018) presented a technique, which is basically a combination of the methods modified QS (LYTRAS, 2015) and Friedmann (POHLERT, 2014) tests. In Webel and Ollech (2018), the authors suggested a generic scheme to construct an overall seasonality test from a given set of candidate tests. A broad set of simulated ARIMA processes were used in order to identify the most informative tests. The resulting classification rule presented low type I and type II misclassification rates, with high accuracy and without showing excessive complexity. The residual-based variants of the QS and Friedman tests were identified as being

most informative. For this reason, both tests were considered to compose the Webel-Ollech test (WEBEL; OLLECH, 2018). In this thesis, the Webel-Ollech test is denominated as WO test.

2.2 Graph Theory

According registers in the literature, the first work regarding Graph Theory dates from 1736, when *Leonard Euler* (1707 - 1783) solved, the well known *Königsberg bridge problem* (JUNGNICKEL, 2013). The city of Königsberg, which until 1945, integrated Prussia territory, nowadays is called Kaliningrad and also is part of Russia. The Pregolya river, which got two islands that were part of Königsberg territory, used to cross through all the city at that time, XVIII century, additionally, seven bridges were used in order to make connected all the city, according to Figure 2.6.

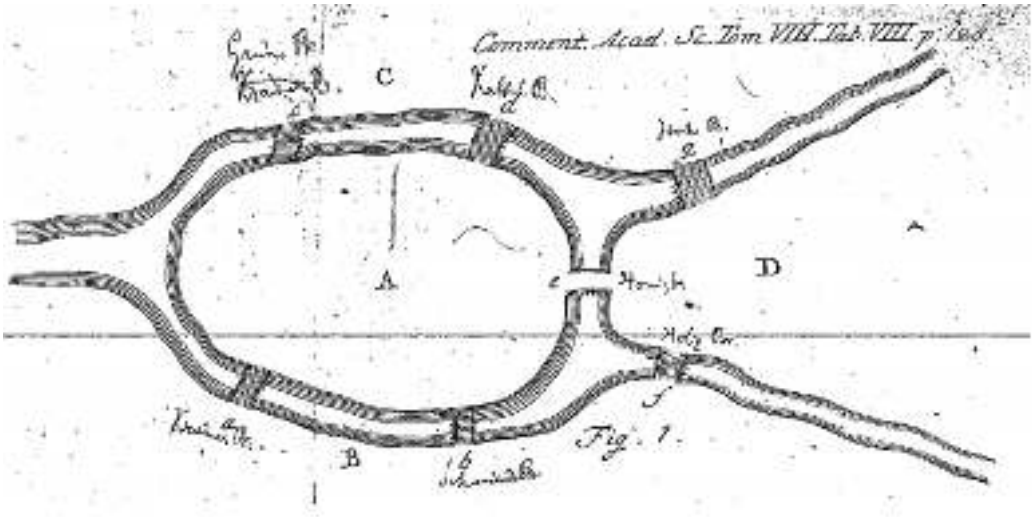


Figure 2.6: Illustrative map from the seven Königsberg bridges (EULER, 1741)

Observing the Figure 2.6, there were four connected territories: two lands, let say B and C, separated by Pregolya river, and two islands A and B. The seven bridges connecting all those four territories were: *a* and *b* connecting A to B; *c* and *d* connecting A to C; *e* connecting both islands; *f* providing connection between B and D; and, lastly, *g* connecting D to C. Consider associate A, B, C or D, to points, or vertices, and each of those respective connection as bonds or lines connecting these points. The result of this arrangement can be illustrated by the Figure 2.7. At that time, the question to be answered was: could be possible walk around through all the city of Königsberg, crossing over all seven bridges just once? According to the official website, Pacific (2020), which is a digital library dedicated to research and preservation of works by several relevant authors in the past, Leonhard Euler presented, in early 1735, his work to The Saint Petersburg Academy, with the formal proof, that there were no solution to the Königsberg bridge problem (KBP) (EULER, 1741). Moreover, his work was published only at 1741 (PACIFIC, 2020).

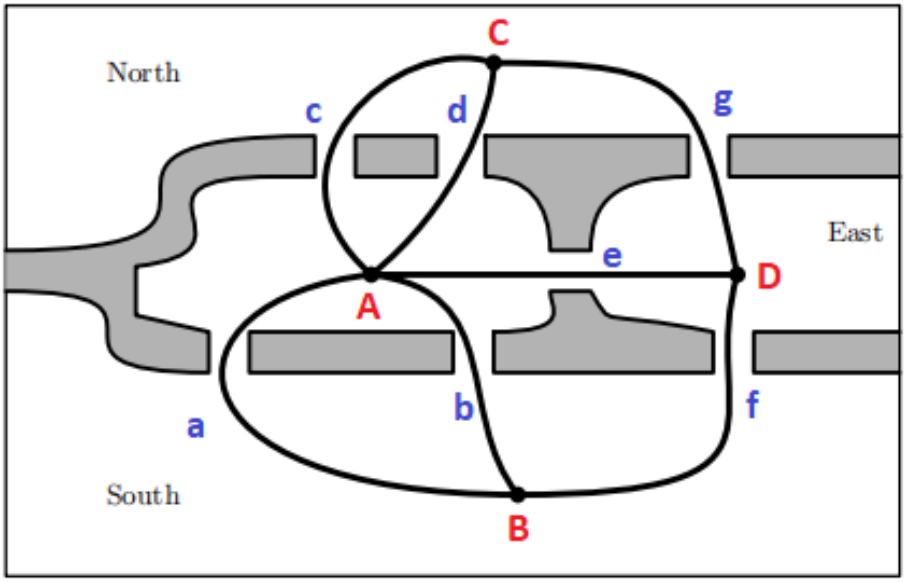


Figure 2.7: Königsberg connections (JUNGNICKEL, 2005) (Adapted by the author)

According to Lopes and Táboas (2015), Euler realized that the exact shape of the two islands and the other two parts of the land did not matter: the problem can be solved just observing the connections properties. Let a city walk begin, starting on the vertex C, for example: when the first return to vertex C occurs, two of the three bridges connected to C, will have already been used. If one leaves C again, trying to walk through the others bridges, one will not be able to return to C without cross over a unused bridge. The same idea can be used if the city walk started from another vertex, besides, it is now shown that this problem is indeed with no solution.

In this section we show some definitions and conceptions from graph theory, which are important to give a better understanding of the contents in the section 2.3.

2.2.1 Concepts and definitions on Graph Theory

Based on West (2000), Steen (2010) and Biggs (1993) follow the definitions of **graph** and **directed graph**.

Definition 2.6 (Graph). *A* ***graph*** $G = (V, E)$ *is a triple which contains a vertex set,* $\boldsymbol{V(G)}$*, a edge set,* $\boldsymbol{E(G)}$ *and a relation that associates each edge with each two vertices (not necessarily distinct) called* ***endpoints****. If* $v \in \boldsymbol{V(G)}$ *is connected to* $w \in \boldsymbol{V(G)}$ *both vertices are said to be* ***adjacent****.*

Definition 2.7 (Directed Graph). *A* ***directed graph*** $\boldsymbol{G}$ *or a* ***digraph*** D *is given by a set of vertices* $\boldsymbol{V(G)}$*, and a set of edges* $\boldsymbol{E(G)}$ *and can be written as* $D = (\boldsymbol{V(G)}, \boldsymbol{E(G)})$*, or only,* $D = (V, E)$*. Every single vertex* $v \in \boldsymbol{V(G)}$ *is jointed to another vertex* $w \in \boldsymbol{V(G)}$*, and here there is not necessary having* $v \neq w$*, by a edge* $e = \langle \overrightarrow{v, w} \rangle$*, in this order, once,*

orientation matters. Vertices v and w are called ***tail****, and* ***head****, respectively, from the edge $e = \langle \overrightarrow{v, w} \rangle$.*

Definition 2.8 (Undirected Graph)**.** *A graph is called* ***undirected graph*** *if, for every edge $e = \langle v, w \rangle \in$ **V(G)**, orientation does not matter* (STEEN, 2010), (JUNGNICKEL, 2005).

Definition 2.9 (Vertex Degree)**.** *Consider a graph $G = (V, E)$. For each vertex $v \in V$, its degree, represented by $\rho(v)$, is defined by the number of edges containing heads or tails leaning on v.*

Definition 2.10 (Regular Graph)**.** *Let $G = (V, E)$ be a given graph. G is called a k-regular, if and only if, $\forall\ v \in$ **V(G)**, $\rho(v) = k$* (JUNGNICKEL, 2013).

Definition 2.11 (Vertex indegree)**.** *Consider a digraph $D = (V, E)$. For each vertex $v \in$ **V(D)**, its* ***indegree****, $\rho_{in}(v)$, is measured by the number of edges with heads coinciding to v.*

Definition 2.12 (Vertex outdegree)**.** *Consider a digraph $D = (V, E)$. For each vertex $v \in$ **V(D)**, its* ***outdegree****, $\rho_{out}(v)$, is measured by the number of edges with tails coinciding to v.*

Theorem 2.1. *In a given digraph $D = (V, E)$, the sum of all indegrees, as well as the sum of all outdegrees, is equal to the number of edges of D:*

$$\sum_{v \in \mathbf{V(D)}} \rho_{in}(v) = \sum_{v \in \mathbf{V(D)}} \rho_{out}(v) = |A(D)| \tag{2.61}$$

Proof. Taking into account that each edge is composed by one head and one tail hence the number of heads is exactly the number of edges. Considering the definition of indegree, it can be concluded that the sum of all indegrees is equal the number of edges. The same argument can be applied to the case of summing all outdegrees. □

In the theorem 2.1 notation $|A(D)|$ refers to the digraph adjacency matrix and this concept is presented in the definition 2.19. In accordance to the theorem 2.1, follows the corollary 2.1.

Corollary 2.1 (from theorem 2.1)**.** *In a network $G = (V, E)$, $\forall\ v_i \in$ **V(G)**, the following propriety satisfies:*

$$\sum_{i=1}^{n} \rho(v_i) = 2 \cdot n\,(\ \mathbf{E(G)}) \tag{2.62}$$

where $n($ ***E(G)***$)$ *means the number of elements on set* ***E(G)***.

Proof. The proof comes immediately by summing both: sum of indegrees to the sum of outdegrees. Once both of them are equal the total number of edges of G, hence, this sum results in 2 times the total number of edges. □

Definition 2.13 (Connected Graph)**.** *An undirected graph G is considered connected if there is a path which connects each pair of its vertices. If there is any single vertex, $v \in$* ***V(G)****,non reachable from any other vertex $w \in$* ***V(G)****, the graph G is called disconnected.*

As reported by Bollobas (2002), by definition, multiple edges or loops are not allowed in graphs. If any of those occur, G is not a graph, but a **multigraph**.

Definition 2.14 (Multiple Edges)**.** *For any graph $G = (V, E)$, if there are two or more edges $e_i \in$* ***E(G)****, with $i \geq 2$, which connect the same two vertices $v \in$* ***V(G)*** *and $w \in$* ***V(G)****, thus all $e_i = \langle v, w \rangle$ are called* ***multiple edges****, also* ***multi-edge*** *or* ***parallel edge****.*

Definition 2.15 (Loops)**.** *A graph $G = (V, E)$, contains a loop if, there is a edge $e \in$* ***E(G)****, whose head and tail are incident to the same vertex $v \in$* ***V(G)****, i.e., $\exists\ e = \langle v, v \rangle$.*

The next definition presents the concept of simple graphs and it is based on Gibbons (1985),West (2000) and Bronshtein *et al.* (2004).

Definition 2.16 (Simple Graph)**.** *A simple graph is an unweighted and undirected graph which contains no loops or multiple edges. Simple graphs are also called a strict graph.*

Definition 2.17 (Neighbor Set $N(v)$)**.** *For any graph $G = (V, E)$ and vertex $v \in$* ***V(G)****, the* ***neighbor set N(v)*** *of v is the collection of vertices (distinct of v) adjacent to v, i.e.*

$$N(v) \stackrel{def}{=} \{w \in \mathbf{V(G)} / v \neq w, \exists e \in \mathbf{E(G)} : e = \langle w, v \rangle\}. \tag{2.63}$$

Definition 2.18 (Complete Graph)**.** *For any graph $G = (V, E)$ such $|$* ***V(G)****$| = n$ and $|$* ***E(G)****$| = m$, if all possible two-elements subsets of* ***V(G)*** *are found as edges of G, this graph is called a* ***complete graph*** *and is represented by K_n.*

If $G = (V, E)$ is not a multigraph, i.e., admit no multiple edges and no loops, so $0 \leq |\ \mathbf{E(G)}| \leq \binom{n}{2}$. Regarding the contents of definition 2.18, trough combinatory arguments, the number of edges existing in K_n is $m = \binom{n}{2}$.

There are different ways of representing a graph (STEEN, 2010). Both, Steen (2010) and West (2000) present, according shown in definition 2.19, an alternative way to describe a graph, which is known by adjacency matrix.

Definition 2.19 (Adjacency Matrix)**.** *Considering a graph $G = (V, E)$, where $n(V) = n$ and $n(E) = m$, its* ***adjacency matrix*** *is a $n \times n$ matrix such elements $a_{v,w}$ represent the number of edges $\langle v, w\rangle \in$ **E(G)** bonding $v \in$ **V(G)** and $w \in$ **V(G)**.*

As an example, the adjacency matrix associated to the graph expressed in the Figure 2.7 is given by:

$$A(G) = \begin{bmatrix} 0 & 2 & 2 & 1 \\ 2 & 0 & 0 & 1 \\ 2 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \tag{2.64}$$

About the adjacency matrix, consider the following considerations.

1. If $G = (V, E)$ is a undirected graph, i.e.,$\langle v, w\rangle = \langle w, v\rangle$, $\forall\langle v, w\rangle \in$ **E(G)**, hence $A(G)$ will be symmetric;
2. $G = (V, E)$ will be a simple graph, if and only if, $a_{i,j} \leq 1$ and $a_{i,i} = 0$, $\forall i, j$; and,
3. $\rho(v_i) = \sum_{j=1}^{n} a_{i,j}$, i.e., the vertex v_i degree is equal the sum of all elements on the *i-th* row of $A(G)$.

Some authors, such as Biggs (1993) and Feofiloff, Kohayakawa and Wakabayashi (2009), define adjacency matrix differently from that was shown in definition 2.19. For those authors, $a_{i,j} = 1$ if, and only if, vertices i and j are connected, otherwise,$a_{i,j} = 0$. In the case of loops, $a_{i,i} = 1$. In this definition, does not matter, how much connections there are between two vertices $v \in$ **V(G)** and $w \in$ **V(G)**, but only if both are connected or not.

In a different way, a graph can also be described through the Incidence Matrix. As stated in Skiena (1990), the fist definition of incidence matrix came from the physicist Kirchhoff (1847) and follows:

Definition 2.20 (Incidence Matrix)**.** *Considering a graph $G = (V, E)$, with $n(V) = n$ and $n(E) = m$, its* ***incidence matrix*** *is a $n \times m$ matrix where each row determines a*

vertex and each column a edge and its elements m_{v_i,e_j} *represent the number of incidences occurred by the edge* $e_j \in$ ***E(G)***, *with* $j \in 1, 2, \cdots, m$ *upon the vertex* $v_i \in 1, 2, \cdots, n$. *The possible values to* m_{v_i,e_j} *are 0, 1 or 2: the first case when* e_j *is not connected to* v_i*; the second happens if one, and only one* e_j *endpoint is attached to* v_i*; and, lastly if the value is 2, hence there is a loop, i.e., both* e_j *endpoints are connected to* v_i.

Just for illustrating, the incidence matrix related to the Königsberg bridges problem, shown in the Figure 2.7, is specified by:

	AB	**AC**	**AD**	**BD**	**CD**
A	1	1	1	0	0
B	1	0	0	1	0
C	0	1	0	0	1
D	0	0	1	1	1

$$M(G) = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}. \tag{2.65}$$

2.3 Complex Networks

Complex network analysis (CNA) is a discipline of exploring quantitative relationships in the graphs with non-trivial and irregular structure (ZINOVIEV, 2017). Despite the researches in complex networks have started with graph theory, only recently it became a focus of attention, given that real networks have characteristics which are not explained by uniformly random connectivity (COSTA *et al.*, 2007). Therefore, for that and others reasons the study of complex networks has been the important branch of many scientific fields (DONG *et al.*, 2013). It can be considered a multidisciplinary area because, in terms of concept, Complex Networks can be understood as an intersection of Graph Theory and Statistical Mechanics (COSTA *et al.*, 2007). Over the years, Graph Theory has providing solid theoretical background in the study of the networks. However, due to the constant computers development, massive databases were collected, stored and processed (FERREIRA, 2017). The use of traditional graph theory approaches is unsuitable for the task of processing large-scale networks given the high computational cost required for seeking optimal solutions (FERREIRA, 2017).

A complex network has a non-trivial topography. It is not a grid, not a tree, not a ring, but it is not entirely random, either (ZINOVIEV, 2017). Each complex network presents specific topological features which characterize its connectivity and highly influence the dynamics of the processes executed on the network (COSTA *et al.*, 2007). One of the most common mechanisms is the preferential connection process, whereby nodes highly connected get even more edges, forming huge hubs in the core, surrounded by the poorly connected periphery (ZINOVIEV, 2017). This mechanism helps the inference process since studying the relationship between pairs of nodes, hidden information can be useful to indicate which node, or groups of nodes, are more likely to form new connections. Characterizing complicated dynamics from time series is a fundamental problem of interest in many fields (GAO; JIN, 2009). In order to analyze these dynamics some different measures has been proposed, for example, Lyapunov exponent (SáNDOR *et al.*, 2021), entropy (ZENIL; KIANI; TEGNÉR, 2018), and fractal scale (WEI; WANG, 2016). Complex networks have established themselves in recent years as being particularly suitable and flexible for representing and modeling many complex natural and artificial systems. Investigation of nonlinear time series in terms of complex networks has begun only recently (GAO; JIN, 2009).

In order to provide a better understanding about the results presented in this work, some basic concepts on Complex Network theory is presented. All contents presented in this subsection are mostly based on Steen (2010).

2.3.1 Network Vertex Degree

A network vertex degree follows the same statement of meaning as that one expressed by definition 2.9 in section 2.2.1. Complementing the explanation given in that section, some authors, such as Skiena (1990), also calls vertex degree as local degree or valency. When the

point of interest is network analysis an important concern, how explained in Steen (2010), is how vertices degrees behave. Just for an example, in a social network when each node is associated with a person, it is possible to find out the main players of this social network by analyzing the vertex with the higher degrees (TOIVONEN *et al.*, 2006).

By looking to the vertex degree, it is also possible to get information regarding the network structure: if a network is composed by vertices which contains approximately the same degree, so there is a high chance that it presents a regular shape, given the definition 2.10, thus in this case, its vertices play holes with similar importance; in contrary, if the network is featured by a high skewed vertex sequence, i.e., few high connected vertices, which work such as hubs, and many other unpopular vertices, so, if one of these densely-connected vertices is removed, the entire network can be broken into disconnected smaller structures (STEEN, 2010).

One of complex networks characteristics is the dynamism of their topology by introduction of new nodes and vertices randomly. Fatally it impacts the measurement of its structural characteristics. Social networks, for instance, is a kind of random graph which are constantly changing (DONG *et al.*, 2013), hence, it is expected find out no deterministic results for topological quantifying, but probabilistic. Any measurements for, e.g., vertex degree, are done in terms of the picture of the network in the actual moment, so, in order to get results that describe the network in global perspective, it is necessary propose a modeling able to provide information regarding the network evolution along its existence.

In the case of understanding how vertices degrees are changing, one should gather information about the sequence of these vertices degrees. A **degree sequence** is nothing less than listing the vertex degrees of a network in a monotonic order, including repetition as needed (DIESTEL, 2005). Usually these degree sequence are in a descending order (STEEN, 2010).

Definition 2.21 (Degree Distribution)**.** *The fraction of vertices with degree k, is calculated by*

$$p_k = P\left(\rho(v) = k\right) = \frac{g(k)}{n}. \tag{2.66}$$

*When p_k is taken as an approximation to the probability of the vertex $v \in$ **V(G)** register degree k. If this measure is plotted as function of k, the resultant plot is called **degree distribution**.*

According to Latora, Nicosia and Russo (2017), vertex degree change in a random networks can be performed by utilizing a histogram to register the degrees sequence. A simple and connected graph $G = (V, E)$ with $n($ **V(G)**$) = n$ and $n($ **E(G)**$) = m$ was used to produce $g(k) = |\{v \in$ **V(G)** $/\rho(v) = k\}|$ and plot it. The values of $g(k)$ mean the number of vertices which present degree k (STEEN, 2010).

Indeed, p_k can be defined as a probability mass function, since, $p_k \geq 0$, and

$$\sum_{k=0}^{n-1} p_k = \sum_{k=0}^{n-1} \frac{g(k)}{n} = \frac{1}{n} \sum_{k=0}^{n-1} g(k) = \frac{n}{n} = 1. \tag{2.67}$$

Definition 2.22 (Average Vertex Degree)**.** *The average vertex degree of a undirected network $G = (V, E)$ with $n(\ \boldsymbol{V(G)}) = n$ and $n(\ \boldsymbol{E(G)}) = m$ can be calculated through the expected value of degree distribution. In this case, when the distribution is discrete, the expected value is calculated by:*

$$\overline{k} = E(K) = \sum_{i=1}^{n} k_i \cdot p_k. \tag{2.68}$$

Naturally, considering the first simplification for $p_k = \frac{1}{n}$, and corollary 2.1, the result expressed by equation 2.68 is resumed to

$$\overline{k} = \sum_{i=1}^{n} k_i \cdot \left(\frac{1}{n}\right) = \frac{1}{n} \sum_{i=1}^{n} k_i = \frac{2m}{n}. \tag{2.69}$$

The way of describing the dynamic of the vertex degree can change according to the type of the random network. Other works dedicated to study such models consider the following types of random networks: social networks (MUCHNIK *et al.*, 2013), social networks after disasters (KIM; HASTAK, 2018) and complex networks using Information Theory (FREITAS *et al.*, 2019).

2.3.2 Degree Correlation

There are some real complex networks whose nodes are more likely to get linked to other similar nodes in terms of degree (MUCHNIK *et al.*, 2013), in the other hand, is it possible find the opposite behavior between nodes of other real complex network (XENARIOS *et al.*, 2000). The basic structural metric able to indicate how likely nodes bond to nodes of similar or dissimilar vertex degree is called **degree correlation**. See definition 2.23 about degree correlation (TAKEMOTO; AKUTSU, 2016).

Definition 2.23. *Degree-degree correlation indicates the relationship between the node degrees, and it is often defined as the Pearson correlation coefficient of degrees between a connected node pair. This measurement is also known as assortative coefficient.*

2.3.3 Network Link Prediction

The link prediction is an important research field in data mining. At present, most link prediction algorithms are based on the similarity between two entities (DONG *et al.*, 2013). Two nodes of a network are more likely to be connected if they are more similar and for sustaining this hypothesis a strong assumption is that the link itself indicates a similarity between the two nodes and this similarity can be transferred through the links (LU; ZHOU,

2010). The objective of this field is to estimate how likely is the existence of a connection between two nodes. For example, according to Martinez *et al.* (1999), discovering connections in food webs structures is highly cost in the field, thus it is difficult, expensive and it can take years to gather relevant information hence, the knowledge of those networks normally is incomplete. Therefore, in order to diminish the research cost, instead of keep the search for all the possible interactions, forecasting the new bonds based on the observed links seams a better idea if the results are enough accurate (CLAUSET; MOORE; NEWMAN, 2008). The structural equivalence contained in networks can be quantified through similarity indices (LU; ZHOU, 2010).

Two nodes in a graph are said to be similar if they share the same predefined characteristics. Given a complex network at a moment $G = (V, E)$ with $v_i \in$ **V(G)** and $v_j \in$ **V(G)**, link prediction is to predict the probability of the link between the nodes v_i and v_j (DONG *et al.*, 2013). Some techniques provide similarity measures between the vertices in a graph.

Since the forecasting method proposed by Mao and Xiao (2019), one of the benchmark of this research, applied a specific link prediction technique to provide time series forecasting, a brief presentation of this similarity index is being done in this next section.

2.3.3.1 Similarity Based on Random Walk Process: The contents of the present section is based on the methodology proposed by Lu and Zhou (2010). Consider an undirected sample network $G = (V, E)$, as described in *definition* 2.8 excluding the possibilities of multiple links and self connections. Given a pair of nodes $v_i, v_j \in$ **V(G)**, Lu and Zhou (2010) defines s_{ij} as a score of similarity between the nodes.

Consider the transition probability matrix, $P = (p_{i,j})_{N^2}$, where $N = |G|$, defined as:

$$P(p_{i,j}) := \begin{cases} a_{i,j}/k_i, & \text{if } (v_i, v_j) \in E \\ 0, & \text{otherwise} \end{cases} \tag{2.70}$$

where $a_{i,j}$ is an element of the incidence matrix as described in definition 2.20 and k_i is the degree of the node v_i. The probability that a random walker, starting from the node v_i, reaches v_j in exactly t steps is given by $\vec{\pi}_{ij}(t)$. Note that a random walker moves according to described in section 2.1.9.

According to Lu and Zhou (2010) the probability of a random walker starts from node v_i is $\vec{\pi}_i$. Additionally, considering leaving v_i after exactly t steps, is defined by the following recurrent equation:

$$\vec{\pi}_i(t) = P^T \vec{\pi}_i(t-1), \tag{2.71}$$

where the initial state, $\vec{\pi}_i(t)$, is a $N \times 1$ vector where the i-th element is 1 and the others are 0. Lu and Zhou (2010) defines the node similarity as:

$$s_{ij}^{LRW}(t) = \frac{k_i}{2|E|} \cdot \pi_{i,j}(t) + \frac{k_j}{2|E|} \cdot \pi_{j,i}(t), \tag{2.72}$$

where $|E|$ is the number of links in the network. According to is argued by Lu and Zhou (2010), in a random walk from v_i to v_j, the walker has a significant probability to hold off from both nodes even they being close to each other. The consequence for this is a poor forecasting accuracy since in most real networks nodes tend to bond with the nodes which are closer instead of those ones located far away.

Lu and Zhou (2010) proposes the superposed random walk, which is a kind of cumulative function with the purpose of releasing, at each step, a random walker in the starting point, superposing the contribution of each walker. This procedure increases the similarity between the initial node and the neighbor nodes. Follows the similarity index:

$$s_{ij}^{SRW}(t) = \sum_{k=1}^{t} s_{ij}^{LRW}(k) \tag{2.73}$$

All the possibilities for the next node are ranked by the SRW. The greater the new-nodes values of SRW, the greater the probability of they come to existence.

2.3.3.2 Dice Approach: The researcher *Lee R. Dice*, in 1945, discussed about the need of a quantitative manner of measuring how two different species are related in nature (DICE, 1945). With this motivation he introduced the Dice Coefficient which is able to measure similarity between two vectors. The Dice score index, or simply Dice coefficient, can be applied to many tasks, and for example, it is commonly used metrics for the evaluation of segmentation tasks in medical imaging (BERTELS *et al.*, 2019). Consider the example where the sets A and B represents the features present in two molecules, respectively. Now and define $n(\Theta)$ as the cardinal of the set Θ. The Dice index is the number of features which are in common to both molecules divided by the average size of the total number of features present in each molecule, that is

$$s_{ij}^{DICE} = \frac{2 \cdot n\left(A \cap B\right)}{n(A) + n(B)}. \tag{2.74}$$

In the context of a network graph, given any two vertices of a network, the Dice similarity (ANDREWS; HAMARNEH, 2015) is calculated taking twice the number of common neighbors over the sum of the degrees of both vertices.

$$s_{ij}^{DICE} = \frac{2 \cdot n\left(N\left(v_i\right) \cap N\left(v_j\right)\right)}{\rho\left(v_i\right) + \rho\left(v_j\right)}. \tag{2.75}$$

It is not difficult to conclude that the Dice index range is from zero to 1. The result is zero if both vertices does not share any neighbor and it is 1 if both neighbors sets are equal.

2.3.3.3 Jaccard Approach: Given the contents of Arnaboldi *et al.* (2015) the Jaccard coefficient measures similarity between finite sample sets. Consider a network $G = (V, E)$ with $v_i \in$ **V(G)** and $v_j \in$ **V(G)**. The Jaccard coefficient is calculated by the number of common

neighbors between v_i and v_j over the cardinal of the union of its neighbors at all.

$$s_{ij}^{JAC} = \frac{n\left(N\left(v_i\right) \cap N\left(v_j\right)\right)}{n\left(N\left(v_i\right)\right) + n\left(N\left(v_j\right)\right) - n\left(N\left(v_i\right) \cap N\left(v_j\right)\right)}. \tag{2.76}$$

Note that, considering the Jaccard index definition, $0 \leq s_{ij}^{JAC} \leq 1$. Indeed, whether $N\left(v_i\right) = N\left(v_j\right)$, thus $s_{ij}^{JAC} = 1$, but in the case of $N\left(v_i\right) \cap N\left(v_j\right) = \emptyset$, consequently $s_{ij}^{JAC} = 0$.

2.3.3.4 Inverse Log-Weighted Approach: The inverse log-weighted similarity between any two vertices is defined by the number of their common neighbors, weighted by the inverse logarithm of their degrees (ADAMIC; ADAR, 2003).

$$s_{ij}^{ILW} = \sum_{\forall v \in (N(v_i) \cap N(v_j))} \frac{1}{\ln\left(\rho\left(v\right)\right)}. \tag{2.77}$$

According to Adamic and Adar (2003) the inverse log-weighted similarity is based on the assumption that two vertices should be considered more similar if they share a low-degree common neighbor, since high-degree common neighbors are more likely to appear even by pure chance.

2.4 Means Inequality

Let $\{x_1, x_2, \cdots, x_n\}$ be a set of positive real numbers. Taking $w = w_1 + \cdots + w_n$, with $w_k \geq 0$ and $w > 0$ hence:

$$\frac{\sum_{k=1}^{n} w_k x_k}{w} \geq \sqrt[w]{\prod_{k=1}^{n} x_k^{w_k}}. \tag{2.78}$$

The result given by equation (2.78) is known as the arithmetic and geometric means inequality (GAO; SITHARAM; ROITBERG, 2019),(SCHAUMBERGER, 1988). We also recall the power mean inequality (SCHAUMBERGER, 1988), which states that the quadratic mean of a set of positive numbers is greater than the respective arithmetic mean. In the particular case where $n = 2$, that is, when only two positive numbers are involved, consider the following: let a and b, positive numbers. Hence,

$$\left(\sqrt{b} - \sqrt{a}\right)^2 \geq 0 \Leftrightarrow \frac{a+b}{2} \geq \sqrt{ab}. \tag{2.79}$$

Given that a and b are not equal to zero, and applying $\frac{1}{a}$ and $\frac{1}{b}$ in equation (2.79), we obtain

$$\frac{\frac{1}{a}+\frac{1}{b}}{2} \geq \sqrt{\frac{1}{ab}} \Leftrightarrow \sqrt{ab} \geq \frac{2ab}{a+b} = \frac{2}{\frac{1}{a}+\frac{1}{b}}. \tag{2.80}$$

Applying the positive numbers a^2 and b^2 in the result presented by equation (2.79), we find $a^2 + b^2 \geq 2ab$. Adding $a^2 + b^2$ to both sides of this inequality and then dividing both results by four, we obtain:

$$\frac{a^2+b^2}{2} \geq \frac{(a+b)^2}{4} \Leftrightarrow \sqrt{\frac{a^2+b^2}{2}} \geq \frac{a+b}{2}. \tag{2.81}$$

Taking into account the results from equations (2.79), (2.80) and (2.81), considering two positive numbers, the relation between the quadratic mean (QM), arithmetic mean (AM), geometric mean (GM), and harmonic mean (HM) is given by: $QM \geq AM \geq GM \geq HM$.

2.5 Mapping Time Series into Complex Networks

There is a recent group of time series forecasting approaches: those based on Graph Theory analysis. The time series are initially mapped into a complex network and analyzing the interactions between the nodes of the network it is possible to describe the dynamics which govern such interactions (STROGATZ, 2001). Therefore, information which is not captured by traditional methods, specially nonlinear characteristics, can be extracted and forecasts can be more accurate (FERREIRA, 2017).

It is possible to find in the literature some works regarding time series analysis based on complex network theory. For example, in Baggio and Sainaghi (2015) where the authors mapped time series into networks in order to assess the complex dynamics of tourism systems in Livigno, Italy; in Scarsoglio, Ridolfi and Lacobello (2016) the authors identified spatial patterns in turbulence using node degree centrality; and Jiang *et al.* (2017) and Telesca and Lovallo (2012) show successful studies about the seismic activity in Italy.

In order to produce analysis on complex networks, created from time series data, firstly it is necessary map the time series into a complex network. According to Zou *et al.* (2019) there are multiple types of methods which solve this task: the natural visibility graph (LACASA *et al.*, 2008), the horizontal visibility graph (LUQUE *et al.*, 2010), the Multi-scale Limited Penetrable Horizontal Visibility Graph (GAO *et al.*, 2016), the cyclical networks approach (ZHANG; SMALL, 2006), the correlation networks approach (YANG; YANG, 2008), and the recurrence networks (MARWAN *et al.*, 2009). In the next section we show the natural visibility graph approach, which was used in this research to transform the analyzed time series into complex networks. In order to provide reference, we describe the horizontal visibility graph in the appendix B.

2.6 Natural Visibility Graph Networks

According to Lacasa *et al.* (2008), the natural visibility graph, or only visibility graph, transforms a time series into a complex network. Each time series observation is associated to a vertex in a complex network. Let (t_i, y_i) and (t_k, y_k), be two distinct points in a time series, and take any other point (t_u, y_u) between them. The points (t_i, y_i) and (t_k, y_k) are considered visually related if the following rule is satisfied:

$$y_u < y_k + \left(\frac{t_u - t_k}{t_k - t_i}\right)(y_k - y_i)\,. \tag{2.82}$$

In Figure2.8, the $i\text{-}th$ bar relates to the $i\text{-}th$ network node. The rule represented by equation (2.82) is illustrated in the Figure2.8. One can observe that each connection between two bars corresponds to a link between equivalent nodes in the complex network.

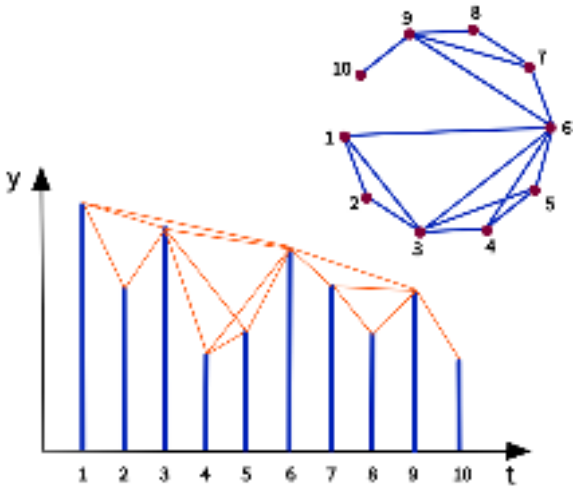


Figure 2.8: Scheme for visibility graph methodology: A 10-elements time series is mapped into a graph where each node represents a bar, and each edge represents the connection between two bars. Source: Drawn by the author

The visibility graph theory has established itself as an efficient approach for probing the dynamics underlying real complex systems from time series (GAO *et al.*, 2016). According to Lacasa *et al.* (2008), the network obtained from this particular mapping inherits several properties of the time series, and its study reveals nontrivial information about the series itself. This mapping makes the structure of the time series be conserved in the graph topology: periodic series convert into regular graphs, random series into random graphs, and fractal series into scale-free graphs. Other particularities achieved from this mapping can be found in Lacasa *et al.* (2008).

3 Time Series Data Mining and Forecasting Process

Data mining is a relatively new field, which has grown since the 1990s, but in the first years of the twenty-first century, it was, in fact, recognized as a field of its own (NISBET; MINER; ELDER, 2009). Yet, according to Nisbet, Miner and Elder (2009), data mining can be defined in several ways, which differ primarily in their focus on different aspects. One of the first definitions of this subject comes from Piatetsky-Shapiro and Frawley (1991): "The non-trivial extraction of implicit, previously unknown, and potentially useful information from data". Also, Kotu and Deshpande (2015) says that data mining is the field of knowledge that finds useful patterns in the data, but, a more complete definition can be found through Zaki (2014) when it is said that data mining is the "process of automatic extraction of knowledge, from large databases, using machine learning, pattern recognition, database techniques and statistics".

It can be found in literature works related to time series data mining based on complex networks, for example, network inference from multivariate time series (KRAMER *et al.*, 2009), time series forecasting using complex network analysis (MAO; XIAO, 2019), methodology for nonlinear time series (GAO *et al.*, 2016) and data mining based on complex network analysis (FERREIRA, 2017).

3.1 Residual Evaluation

Consider one observation of a time series, y_k and its respective estimator $\hat{y}_k$. The residual of this estimation is given by:

$$e_k = y_k - \hat{y}_k. \tag{3.1}$$

The set of residuals obtained from a forecasting assignment is used to evaluate the quality of the forecast. For example, in the Diebold-Mariano test, see section 2.1.10, the residuals obtained from two different methods applied to the same time series, are used to test whether the accuracies obtained from both sets of estimators are statistically the same.

3.2 Forecast Accuracy: Performance Indicators

Given a set of a time series observations $\{y_1, y_2, \ldots, y_m\}$ and those respective predicted values $\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m\}$, the performance indicators to be used in this work are:

1. Mean Absolute Error (MAE),

$$\text{MAE} = \frac{1}{m}\sum_{t=1}^{m} \left|\hat{y}_t - y_t\right|, \tag{3.2}$$

According to presented in the equation 3.2, the MAE quantifies the average magnitude

of the errors produced in a forecasting process, without taking into account whether they are positive or negative. Observing the definition of the MAE we can note that all the individual errors are equally weighted in the average, therefore, it is a linear score which does not indicate whether large errors are being committed. This means that two sets of forecasts, with different variances can present similar levels of MAE. The MAE is a scale-dependent accuracy measure therefore it should not be considered in the task of making comparisons between series of different scales (HYNDMAN; ATHANASOPOULOS, 2018).

2. Mean Absolute Percentage Error (MAPE),

$$MAPE = \frac{1}{m}\sum_{t=1}^{m}\left|\frac{\hat{y}_t - y_t}{y_t}\right|, \text{ and} \tag{3.3}$$

Given that MAPE is calculated dividing measurements with same dimensions, so the series scale does not matter: it is a scale-free measure. Obviously if there is any zeros within the dataset, this measure can not be applied.

3. Root Mean Squared Error (RMSE),

$$RMSE = \sqrt{\frac{1}{m}\sum_{t=1}^{m}(\hat{y}_t - y_t)^2}. \tag{3.4}$$

The RMSE is a commonly applied in order to evaluate accuracy and it is a quadratic scoring rule which measures the average magnitude of the error. Similarly to MAE, the RMSE makes no distinction among the direction of the errors. RMSE is also scale dependent, therefore it can not be used as a performance indicator between different time series (HYNDMAN; KOEHLER, 2006). Observe that the errors are squared prior to the average calculation, thus, the RMSE intensifies the weights given to the large errors. This means whether large errors are unwanted, it is more appropriate using the RMSE to measure accuracy. It is important to comment that RMSE is not larger whether the variance of the errors are larger, but, RMSE is greater whether the variance of the error magnitudes become more intense.

3.3 Sliding Window

Both the autocorrelation function (ACF) and the partial autocorrelation function (PACF) quantify the influence of past observations on the current value of the time series. Additionally, ACF or PACF can indicate whether previous values cause, or not, a significant impact on the current observation. Considering that some past values may not be statistically significant to

predict the next term of the time series, it is reasonable to disregard such data in the forecasting process. The set of past observations considered in the forecast is defined as a window.

Given the set $\mathbf{Y} = \{y_1, y_2, \ldots, y_n\}$, a sliding window, $\mathbf{Z}_i$, over $\mathbf{Y}$ is a subset with fixed number of elements, k, which, during the simulation, changes at each iteration i, according to the following rule:

$$\begin{aligned} \mathbf{Z}_1 &= \{y_1, y_2, \ldots, y_k\}, \\ \mathbf{Z}_2 &= \{y_2, y_3, \ldots, y_{k+1}\}, \\ &\ldots, \\ \mathbf{Z}_{n-k+1} &= \{y_{n-k+1}, y_{n-k+2}, \ldots, y_n\}. \end{aligned} \tag{3.5}$$

The sliding window process is based on the methodology walk-forward validation presented by Kaastra and Boyd (1996). The strategy exposed in the equation (3.5) is illustrated in the Figure 3.1.

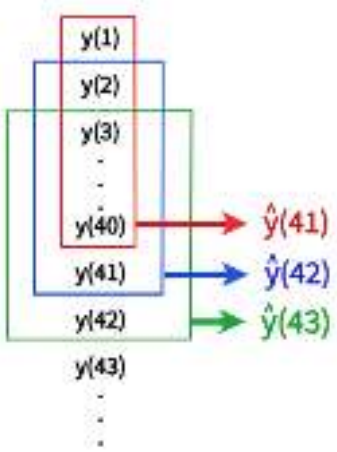


Figure 3.1: Description of a sliding window process with size 40. Source: Drawn by the authors.

3.4 Naive Method

The naive estimation is one of the simplest and high cost-benefit forecasting method (HYNDMAN; ATHANASOPOULOS, 2018), among other reasons, also because this method outperforms other sophisticate forecasting approaches for many economic and financial time series (HYNDMAN; ATHANASOPOULOS, 2018). The naive method is given by

$$\hat{y}_{t+1} = y_t. \tag{3.6}$$

3.5 Exponential Smoothing Models

Exponential smoothing is a forecasting method for univariate time series proposed in the late 1950s (BROWN, 1959) and widely used for short-term forecasts (BIAZZI, 2019). The forecasts obtained through the exponential smoothing approaches are weighted averages of past observations, with the weights decreasing exponentially as the observations get older (HYNDMAN; ATHANASOPOULOS, 2018), hence, the smaller is the lag from the current observation the higher the impact of the related weight.

3.5.1 Single Exponential Smoothing

The single exponential smoothing is one of the simplest methods among the exponential smoothing models. The one-step-ahead estimation for time $t+1$ is given by:

$$\hat{y}_{t+1} = \sum_{i=0}^{M} \alpha \left(1-\alpha\right)^{i} y_{t-i}. \tag{3.7}$$

The result presented in the equation (3.7) gives more importance to the older observations if α is closer to 0. In the extreme case of $\alpha = 1$, the SES forecasts are equal to the naive estimation.

3.5.2 Weighted Average

The weighted average combines, using weighted arithmetic mean, the current observation, y_t, with $\hat{y}_t$ (HYNDMAN; ATHANASOPOULOS, 2018). See equation (3.8).

$$\hat{y}_{t+1} = \alpha y_t + \left(1-\alpha\right)\hat{y}_t. \tag{3.8}$$

3.5.3 Level Component

An alternative to SES is the level component form, represented by l_t. weighted average combines, using weighted arithmetic mean, the current observation, y_t, with $\hat{y}_t$ (HYNDMAN; ATHANASOPOULOS, 2018). See equation (3.8).

$$\begin{aligned} \text{Forecast Equation :} \quad & \hat{y}_{t+1} = & & l_t \\ \text{Level Equation :} \quad & l_t = & & \alpha y_t + \left(1-\alpha\right) l_{t-1}. \end{aligned} \tag{3.9}$$

3.5.4 Trend Component

According to Holt (2004) it is possible adjusting the simple exponential smoothing in order to produce forecasts for a trend time series. The Holt's Linear Trend Approach can provide

one-step-ahead estimators by combining forecasting equation and two smoothing equations.

$$\begin{aligned} \text{Forecast Equation :} \quad \hat{y}_{t+1} &= l_t + b_t \\ \text{Level Equation :} \quad l_t &= \alpha y_t + (1-\alpha)(l_{t-1} + b_{t-1}) \\ \text{Trend Equation :} \quad b_t &= \beta^* (l_t - l_{t-1}) + (1-\beta^*) b_{t-1}, \end{aligned} \tag{3.10}$$

where l_t is the level component of the time series at time t, the b_t is the estimate of the trend at time t, and, as well as α, β^* is a constant which satisfies $0 \leq \beta^* \leq 1$. Note that the term $(l_t - l_{t-1})$ is an estimate of the trend in the time t, therefore, the trend equation is combining the first past trend to the estimate of the current trend.

3.5.5 Seasonal Component

In order to capture the seasonal variations of the time series, Holt (2004) and Winters (1960) adjusted the Holt's original approach. According to Hyndman and Athanasopoulos (2018) the Winter-Holt seasonal method is composed by the combination of three smoothing equations and one forecast equation, but this ensemble is done in two forms: additive and multiplicative. If the seasonal variations are sufficiently constant along the time series, so the additive form is more likely to produce better accuracy. In the case of the seasonal variations fluctuate in time, so the multiplicative form can be a better choice.

$$\begin{aligned} \text{Forecast Equation :} \quad \hat{y}_{t+1} &= l_t + b_t + s_{t+1-T} \\ \text{Level Equation :} \quad l_t &= \alpha (y_t - s_{t-T}) + (1-\alpha)(l_{t-1} + b_{t-1}) \\ \text{Trend Equation :} \quad b_t &= \beta^* (l_t - l_{t-1}) + (1-\beta^*) b_{t-1} \\ \text{Seasonal Equation :} \quad s_t &= \gamma (y_t - l_{t-1} - b_{t-1}) + (1-\gamma) s_{t-T} \end{aligned} \tag{3.11}$$

where T is the period of the seasonal variations and γ is a constant which satisfies $0 \leq \gamma \leq 1$.

3.5.6 Error-Trend-Seasonal (ETS) Models

The Error-Trend-Seasonal (ETS) approach is a forecasting method applied to univariate time series (JOFIPASI; MIFTAHUDDIN; HIZIR, 2018). The method aims to decompose the original data into trend and seasonal components and combine the obtained new data to produce forecasts (PANIGRAHI; BEHERA, 2017).

Each exponential smoothing (ES) method is classified through pair of letters (T, S) denoting trend and seasonal, respectively. The trend component can be classified to $N, A,$ and A_d, which means No-Trend, Additive trend and Damped-additive trend, respectively. In the case of the seasonal component, it can be classified as N, A and M which means No-seasonal, Additive and Multiplicative, respectively. Some of the combinations have specific names, for example: model (N, N) is the simple exponential smoothing; (A, N) is the Holt's

linear approach; (A_d, N) additive damped trend method; and (A, A) additive Holt-Winter's method.

The ETS models consider also the influence of the estimation errors in the forecasting process, so a third letter is included in the ES notation. The errors can be included in the additive and multiplicative form, so the models passes to be labeled as (E, T, S) with $E = \{A, M\}$, $T = \{N, A, A_d\}$ and $S = \{N, A, M\}$. The details of each ETS combination method can be observed in Hyndman and Athanasopoulos (2018) and Hyndman *et al.* (2008).

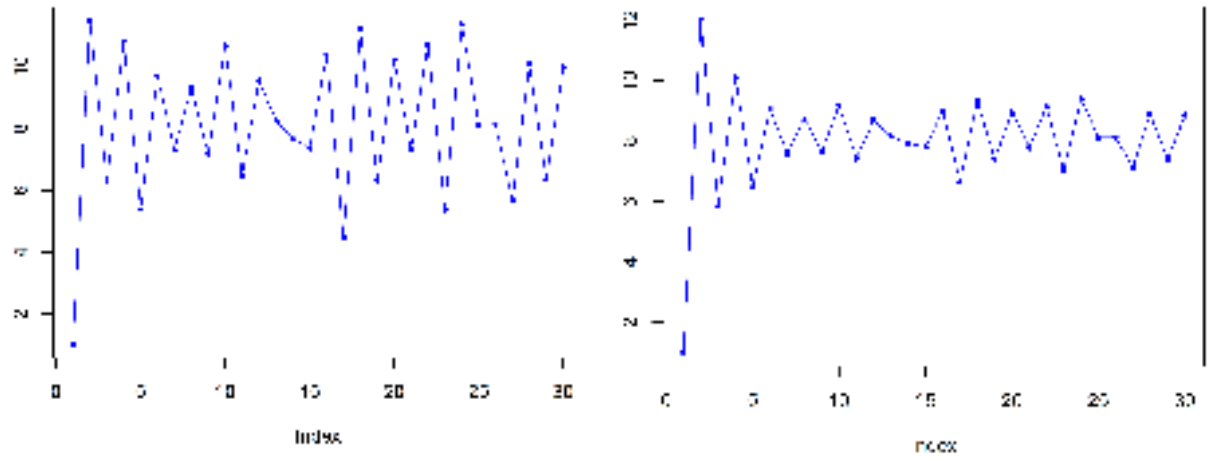
In this work, all forecasting results are one-step-ahead estimates, so, we are not presenting the h-step-ahead ETS models, however, they are described in Hyndman and Athanasopoulos (2018).

3.6 Autoregressive Model (AR)

Widely used in Statistics, Econometrics and Signal Processing, the Autoregressive model (AR) is a mathematical formulation of a random process and a linear predictive modeling technique (DAI; LIU; ZHANG, 2015). The term *autoregressive* relates a regression of the variable versus itself, thus, in an autoregressive model, the current observation is predicted by using a linear combination of past data (SHUMWAY; STOFFER, 2017). The AR(p) model, autoregressive of order p, is described by:

$$y_n = c + \sum_{i=1}^{p} \phi_i y_{t-i} + \epsilon_t. \tag{3.12}$$

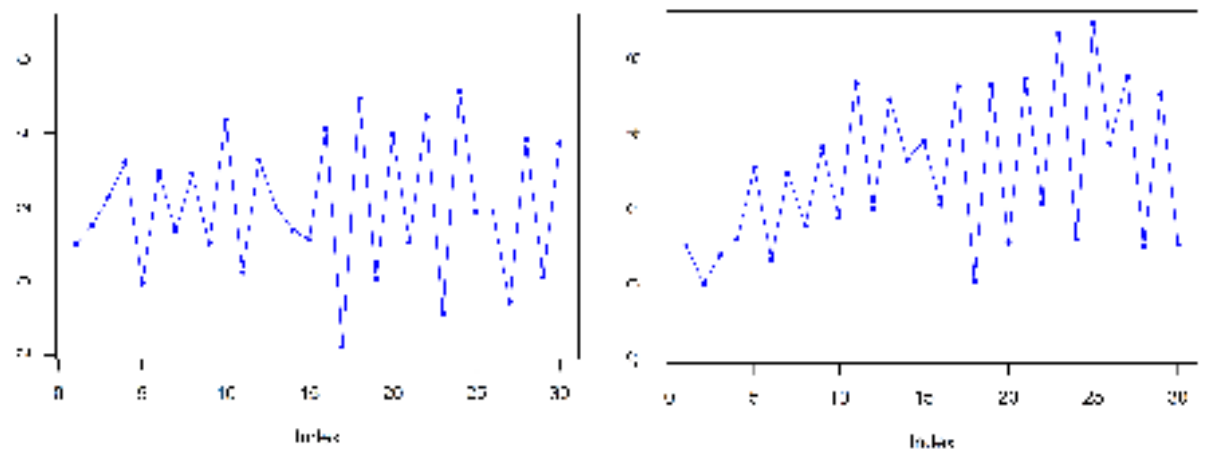
where ϕ_i is the $i\text{-}th$ coefficient related to $i\text{-}th$ last time series observation, c is a fixed constant and ϵ_t is a white noise component. The Figures 3.2a and 3.2b show the difference in the scale of two AR(1) models due to a change in the variance of the white noise term, without changing the coefficients. Not that the time series patterns keeps the same.



(a) (b)

Figure 3.2: Examples of AR(1) models built based on the same coefficients: $y_t = 13 - 0.6y_{t-1} + \epsilon_t$. But: (a) $\epsilon_t \sim N(0, 1)$, and: (b) $\epsilon_t \sim N(0, 0.4)$. Source: Drawn by the author.

By changing the coefficients ϕ_i, the patterns of the time series changes, therefore, it is possible fit the proper model to a large variety of time series showing different patterns. It can be observed in the Figures 3.3a and 3.3b. In both cases, the white noise term has variance 1.



(a) (b)

Figure 3.3: Examples of AR(1) and AR(2) models, respectively, built based on the same white noise parameters $\epsilon_t \sim N(0, 1)$. But: (a) $y_t = 3 - 0.6y_{t-1} + \epsilon_t$, with $y_1 = 1$, and: (b) $y_t = 1 + 0.65y_{t-2} + \epsilon_t$, with $y_1 = 1$ and $y_2 = 0$. Source: Drawn by the author.

The patterns obtained for the two time series in the Figure 3.3 are considered different, given the KPSS test results: for the model in the Figure 3.3a, the KPSS test reveals an level-stationary time series, whilst for the other model, the same test rejects the null hypothesis of level-stationarity. Considering a generic AR(1) model,

$$y_t = c + \phi_1 y_{t-1} + \epsilon_t, \tag{3.13}$$

therefore the following can be concluded:

1. If $\phi_1 = 0$, so y_t is a white noise centered in c.

2. If $|\phi_1| < 1$, so y_t is stationary (BOX *et al.*, 2015). In this case, $E[y_t] = E[y_{t-1}] = \mu$, and consequently, $\mu = c + \phi_1\mu + \overbrace{E[\epsilon_t]}^{=0} \Leftrightarrow \mu = \dfrac{c}{1-\phi_1}$. Hence, y_t will oscillate around the expected value, μ.

3. If $\phi_1 = 1$, so y_t is a random walk process, according to discussed in the section 2.1.9.

We consider important to present a brief discussion of how to find the parameter p which produces the best AR fit over a stationary time series. Let's start defining $w_t = y_t - \mu_y$, where $\mu_y = E[y_t]$. Without any loss of generality, take the p-order stationary autoregressive model:

$$w_t = \phi_1 w_{t-1} + \phi_2 w_{t-2} + \cdots + \phi_p w_{t-p} + \epsilon_t. \tag{3.14}$$

Multiplying throughout in equation (3.14) by w_{t-k}, for $k \geq 0$ and next taking the expected value of the resultant expression comes:

$$E[w_t w_{t-k}] = \phi_1 E[w_{t-1}w_{t-k}] + \phi_2 E[w_{t-2}w_{t-k}] + \cdots + \phi_p E[w_{t-p}w_{t-k}] + E[w_{t-k}\epsilon_t]. \tag{3.15}$$

The last term of the equation (3.15), $E[w_{t-k}\epsilon_t]$ is zero, because, w_{t-k} is related to the white noise terms up to ϵ_{t-k}, which are uncorrelated to ϵ_t. From the equation (2.16), $E[w_t w_{t-k}]$ is the autocovariance, γ_k, thus, the equation (3.15) can be re-written as a difference equation in the autocovariances of the given time series.

$$\gamma_k = \phi_1\gamma_{k-1} + \phi_2\gamma_{k-2} + \cdots + \phi_p\gamma_{k-p}. \tag{3.16}$$

Dividing each term of the equation (3.16) by the variance of the time series, that is, γ_0 (see equation 2.17), the equation (3.16) becomes

$$\rho_k = \phi_1\rho_{k-1} + \phi_2\rho_{k-2} + \cdots + \phi_p\rho_{k-p}. \tag{3.17}$$

By using the z-transform theory for difference equations presented in the formal basis by Ogata (1995), or for practical use by Moreira (2006), the values of the autocorrelations, ρ_i, can be written by

$$\rho_k = A_1 r_1^k + A_2 r_2^k + \cdots + A_p r_p^k, \tag{3.18}$$

where $r_1, r_2, \cdots, r_p$ are the roots of the characteristic equation of order p: $x^p - \phi_1 x^{p-1} - \phi_2 x^{p-2} - \cdots - \phi_{p-1}x - \phi_p = 0$. We must revisit the necessity of $|\phi_i| < 1$ for stationarity, and tt is quite reasonable assume distinct roots for the characteristic equation, that is, $r_i \neq r_j$, for all $i \neq j$. Hence, two situations can occur:

(a) Real roots: so, the term $A_i r_i^k$ contributes to a convergent exponential response, known as damped exponential.
(b) Complex-conjugate roots: so, the term $A_i r_i^k$ contributes to a oscillatory response with a convergent exponential amplitude, given by $|r_i|^k$ and a frequency of $2\pi f = \text{sen}^{-1}\left(\frac{\text{Im}\,(r_i)}{|r_i|}\right)$. These terms are known as damped sine waves.

Normally, the values of ρ_k are composed by the influence of the two damped functions: exponentials and sine waves. This discussion was done, just in order to show the close relation which exists between the autocorrelation function to the coefficients of the autoregressive models. In the practical aspect, all we know are the data, so it is possible estimate the autocorrelation function, as shown in the equation (2.17). Generally the coefficients of the autoregressive model are unknown, so, it is necessary finding an approach which is able to give such coefficients as function of the estimated autocorrelation functions. One popular approach for such task is the one based on the *Yule-Walker* equations, which are presented in Box *et al.* (2015), Hyndman and Athanasopoulos (2018), Rao (2021) and Hamilton (1994).

According to Rao (2021) the parameter p of the AR(p) model is fully related to the results shown in the Partial Autocorrelation Function (PACF). In the section 2.1.2 we presented the general calculations in order to obtain the partial correlation between variables Y_n and Y_{n-k}, which is the covariance between its residuals, normalized by the products of its standards deviations. The equation (2.36), when applied to a stationary time series, can be simplified. Whether the time series is at least second order stationary, the subscribe Y_{n-k} makes no difference, but the only significant term is the lag k, therefore the equations 2.37, 2.38 and 2.39 become:

$$\gamma_{Y_n Y_{n-k} \backslash \mathbf{Y}_{ex}} = \gamma_{k \backslash \mathbf{Y}_{ex}} = \gamma_k - COV\left(Y_k, \mathbf{Y}_{ex}\right)^{\mathrm{T}} Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_0, \mathbf{Y}_{ex}\right), \tag{3.19}$$

$$Var\left(\epsilon_n\right) = Var\left(Y_k\right) - COV\left(Y_k, \mathbf{Y}_{ex}\right)^{\mathrm{T}} Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_k, \mathbf{Y}_{ex}\right), \tag{3.20}$$

and,

$$\begin{aligned} Var\left(\epsilon_{n-k}\right) &= \\ &= Var\left(Y_0\right) - \left[COV\left(Y_0, \mathbf{Y}_{ex}\right)\right]^T Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_0, \mathbf{Y}_{ex}\right). \end{aligned} \tag{3.21}$$

The sections 6.2.2, 6.2.3 and 6.2.4 from Rao (2021) are dedicated to present the calculations which lead the use of the equations 3.19, 3.20 and 3.21 to derive the autocorrelations of a generic AR(p) model. In the page 167, of Rao (2021) the reader can see the proof of the following identity:

$$\rho_{p \backslash p} = \phi_{p+1,p+1}, \tag{3.22}$$

that is, the partial correlation at lag p-1 is the last coefficient of the AR(p) model. The conclusion

is that for AR(p) models, the partial correlation of order greater than p will be zero (RAO, 2021). Hence, the plot ot the PACF can be helpful in order to finding the value of p in a AR(p) model. The Figure 3.4 shows the PACF plot for the time series presented in the Figure 3.3b. See that after the second value, the other autocorrelations keep into the insignificance region. This indicates that $p = 2$, indeed, because the PACF graph was obtained from a AR(2) model.

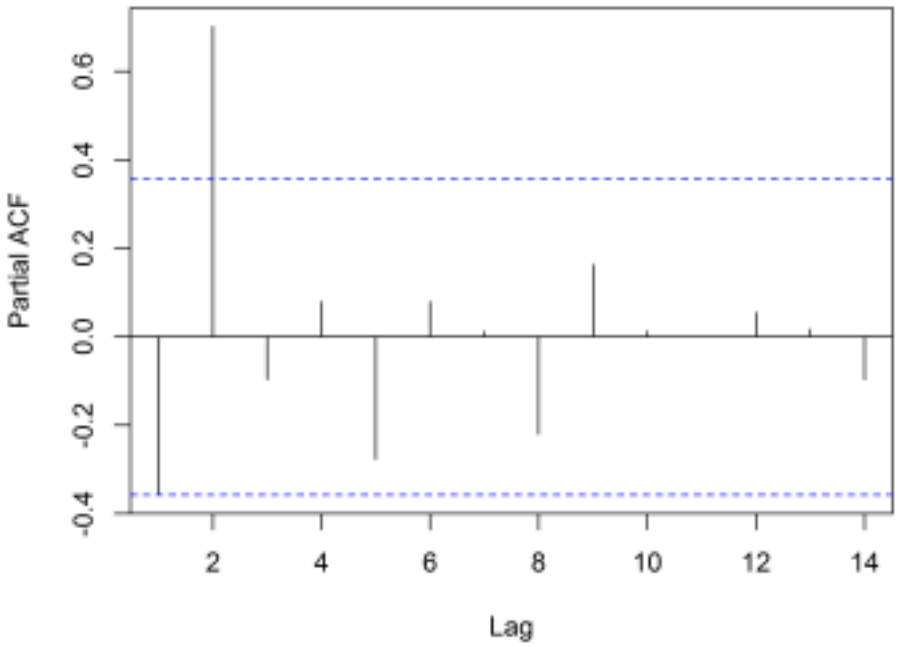


Figure 3.4: The partial autocorrelation function of a AR(2) model built based on white noise parameters $\epsilon_t \sim N(0, 1)$ and equation $y_t = 1 + 0.65y_{t-2} + \epsilon_t$, with $y_1 = 1$ and $y_2 = 0$. Source: Drawn by the author.

3.7 Moving Average Model (MA)

The moving average model (MA), or moving-average process is commonly used to model univariate time series (RAO, 2021). According to Petris (2009) the current observation of a time series is linearly dependent on a sequence of white noise terms, that is, the moving average models a time series which is a weighted sum of independent random variables (RAO, 2021). As a consequence, such models by nature, always show weak stationarity. A moving average model of order q is generally written as MA(q). Its mathematical formulation is given by:

$$y_n = \mu + \epsilon_t + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}. \tag{3.23}$$

where θ_i is the $i\text{-}th$ coefficient related to $i\text{-}th$ last time series error, μ is a fixed constant and ϵ_t is a white noise component relative to the current time series term. As discussed in the section 3.6, the coefficients $\theta_1, \theta_2, \cdots, \theta_q$ are directly related to the patterns of the time series (BOX

et al., 2015), that is, by changing one of these coefficients, the time series patterns changes. The current white noise term just changes the scale of the time series. In order to understand how is the process to estimate the order of a MA(q) model, we are going to start studying the particularities of a MA(1) model. Consider that $\epsilon_t \sim N(0, \sigma^2)$, for all t. Adapting the equation 3.23 to $q = 1$ we have

$$y_t = \mu + \epsilon_t + \theta_1 \epsilon_{t-1}. \tag{3.24}$$

Taking $w_t = y_t - \mu$, the model becomes

$$w_t = \epsilon_t + \theta_1 \epsilon_{t-1}. \tag{3.25}$$

Given the properties of the white noise terms, $E[w_t] = 0$, $\gamma_0 = Var[w_t] = (1 + \theta_1^2)\sigma^2$, $E[\epsilon_t \epsilon_{t-k}] = 0$ for all $k \neq 0$, and the autocovariance, γ_1, is

$$\begin{aligned}
\gamma_1 &= E[w_t \cdot w_{t-1}] = E[(\epsilon_t + \theta_1 \epsilon_{t-1}) \cdot (\epsilon_{t-1} + \theta_1 \epsilon_{t-2})] = \\
&= \overbrace{E[\epsilon_t \epsilon_{t-1}]}^{=0} + \theta_1 \overbrace{E[\epsilon_t \epsilon_{t-2}]}^{=0} + \theta_1 E\left[\epsilon_{t-1}^2\right] + \theta_1^2 \overbrace{E[\epsilon_{t-1} \epsilon_{t-2}]}^{=0} = \\
&= \theta_1 \sigma^2.
\end{aligned} \tag{3.26}$$

Generalizing the result obtained by the equation (3.26), let's calculate $\gamma_n = E[w_t \cdot w_{t-n}]$.

$$\begin{aligned}
\gamma_1 &= E[w_t \cdot w_{t-n}] = E[(\epsilon_t + \theta_1 \epsilon_{t-1}) \cdot (\epsilon_{t-n} + \theta_1 \epsilon_{t-n-1})] = \\
&= \overbrace{E[\epsilon_t \epsilon_{t-n}]}^{=0} + \theta_1 \overbrace{E[\epsilon_t \epsilon_{t-n-1}]}^{=0} + \theta_1 \overbrace{E[\epsilon_{t-1} \epsilon_{t-n}]}^{=0} + \theta_1^2 \overbrace{E[\epsilon_{t-1} \epsilon_{t-n-1}]}^{=0} = 0.
\end{aligned} \tag{3.27}$$

Thus, for all $n > 1$, $\gamma_n = 0$. Dividing γ_1 by the variance, we obtain, $\rho_1 = \dfrac{\theta_1}{1 + \theta_1^2}$ and the autocorrelation function is composed by two non-zero values: $\rho_0 = 1$ and $\rho_1 = \dfrac{\theta_1}{1 + \theta_1^2}$. The other, $\rho_k = 0$ for all $k \geq 2$. Therefore, a practical procedure to define the order of the moving average model is taking a look in the autocorrelation plot. For example, whether the third value is placed in a region of statistical insignificance, then probably we are dealing to a MA(1) model.

In the case of a MA(2) model, comes

$$w_t = \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2}, \tag{3.28}$$

then, $\gamma_0 = Var[w_t] = (1 + \theta_1^2 + \theta_2^2)\sigma^2$, $\gamma_1 = E[w_t w_{t-1}] = (\theta_1 + \theta_1 \theta_2)\sigma^2$, $\gamma_2 = E[w_t w_{t-2}] = \theta_2 \sigma^2$, and $\gamma_k = 0$ for all $k \geq 3$, see appendix **??**. Hence, the autocorrelation function will show three significant values and all other in the insignificance region. Thus, from the practical aspect, we can conclude that evaluating the autocorrelation function (ACF) might be an alternative to define the maximum lag of a moving average process, because if the first $q + 1$ values of the

ACF are in the region of statistical significance, the time series might be modeled through a MA(q) model.

According to Box *et al.* (2015), an AR(p) model can always be written as a MA(∞) time series, however the inverse is only done under proper conditions (COCHRANE, 1997) and (RAO, 2021). Just for an example, in the appendix A.14, we use Mathematical Induction to proof that a stationary AR(1) model $y_t = c + \phi_1 y_{t-1} + \epsilon_t$ can be written as a MA(∞) model given by

$$y_t = \frac{c}{1-\phi_1} + \sum_{i=0}^{\infty} \phi_1^i \epsilon_{t-i}. \tag{3.29}$$

3.8 Auto-Regressive Moving Average Model (ARMA)

In the statistical analysis of time series, the model known by autoregressive–moving-average (ARMA) gives a simple description of a stationary, or weakly stationary stochastic process based on two linear combinations: the first for the autoregression (AR) and the other for the moving average (MA). The ARMA model was firstly presented in the thesis of Peter Whittle, Hypothesis testing in time series analysis, in 1951, and it became more accepted through the book by George E. P. Box and Gwilym Jenkins, in 1970.

A model ARMA(p,q) is defined according to (3.30), where p and q are the number of autoregressive and moving average terms, respectively (HAMILTON, 1994).

$$y_n = c + \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} \tag{3.30}$$

where ϕ_i is the i-th coefficient of the autoregressive term and θ_i is the i-th coefficient of the moving average term. There are an extensive theory dedicated to analyze the ARMA(p,q) models, such in theoretical aspects as in the practical one: see Rao (2021), Hamilton (1994), Cochrane (1997), Box *et al.* (2015), Shumway and Stoffer (2011), Box, Jenkins and Reinsel (2013) and Morettin and Toloi (2006). We indicate these examples of references where the reader will be able to know the particularities of an ARMA(p,q) model. In this work, we want only to show one way to estimate the values of p and q which will provide the simpler model ARMA(p,q) able to fit as better as possible, a stationary dataset.

Taking into account the proposed by Box *et al.* (1994), the univariate time series modeling process using AR(p), MA(q) and ARMA(p,q) models comprises the following steps: (a) identification of the model orders p and q; (b) parameters estimation; (c) check whether the estimated residuals have the white noise characteristics: if yes, proceed to the forecasting task; if no, go back to the first step.

According to discussed in the section 3.6, the coefficients of the autoregressive model are directly related to the values of the autocorrelation function. Hence, the approach to find the values of p and q of an ARMA model passes through looking the plots of the autocorrelation

function, see section 2.1.2, and the partial correlation function, see section 2.1.6. However, when both models are mixed to provide an ARMA(p,q) the identification of the model orders p and q by just looking the autocorrelation and the partial autocorrelation functions can lead to sub-optimal models. Therefore it is necessary using some information criteria to guarantee that the choice for the values of p and q are the best ones.

The first method to evaluate the quality of the model is know as Akaike Information Criteria (AIC), which is a metric that measures the quality of a statistical model considering also its simplicity (STOICA; SELÉN, 2004). Therefore, the AIC provides a metric for comparing and selecting models, in which the lower AIC values the greater the quality and simplicity (TADDY, 2019). AIC aims to estimate how much information is lost by a given model: the less information lost, the greater the model quality and the lower the AIC score (AKAIKE, 1974). The AIC deals with the balance between the quality and parsimony of a model, that is, it works, at the same time, with overfitting and underfitting. Its formula is given by

$$AIC = 2k - 2\ln(\hat{L}) \tag{3.31}$$

where $\hat{L}$ is the maximum value of the model likelihood function, and k is the number of estimated parameters (AKAIKE, 1974). In the case of a small sample, AIC tends to select models that have too many parameters, that is, AIC can overfit (MCQUARRIE; TSAI, 1998). Therefore, it was proposed the AICc, which is a correction in the original AIC expression (MCQUARRIE; TSAI, 1998).

There are other measures to evaluate loss of information. The second common metric is the Bayesian information criterion (BIC), which is similar to the formula for AIC, but the number of parameters is penalized with $k\ln(n)$, instead of $2k$ as is done in the AIC (WIT; HEUVEL; ROMEIJN, 2012).

Both metrics AIC and BIC can be calculated in order to select the best model ARMA(p,q). The values of p and q are obtained from the model with minimum scores on these two metrics.

3.9 Autoregressive Integrated Moving Average Model

Considering Hamilton (1994) the dataset must be stationary to an ARMA model be properly applied. Therefore in the case of non-stationary time series, it may be necessary to take differences in the original data in order to make it stationary. Let d be the number of differences in the given dataset until it become stationary. If ARMA(p,q) is applied to the d times differentiated data, the model becomes ARIMA(p,d,q) (BOX *et al.*, 2015).

3.10 State-of-the-art Method: Mao-Xiao Approach

The work presented by Mao and Xiao (2019), here in this work, called Mao-Xiao Approach, brings a alternative method to provide time series forecasting based on complex

networks. Essentially, a time series with n observations, i.e., points (t_k, y_k), is transformed to a network through visibility graph method (LACASA *et al.*, 2008); secondly the Node Similarity Matrix (NSM), i.e, the matrix which contains a similarity measurement between every two pairs of nodes that generate the complex network created in the first step, is calculated by method exposed in Liu and Lü (2010); the NSM's last row gives the similarity between the last node and each one of the fist $n-1$ nodes, so according to the presented methodology, the node k, with the greater similarity value, is taken and a linear equation is calculated considering both nodes: $N_n(t_n, y_n)$ and $N_k(t_k, y_k)$; the first estimation of the $(n+1)th$ point is calculated by the equation 3.32:

$$\tilde{y}_{n+1} = y_n + \frac{(y_n - y_k)}{(t_n - t_k)}(t_{n+1} - t_n); \tag{3.32}$$

After the first estimation is done, a refining is proposed according to equation 3.33:

$$\hat{y}_{n+1} = w_n \cdot y_n + w_{n+1} \cdot \tilde{y}_{n+1}. \tag{3.33}$$

The weights w_n and w_{n+1} are, respectively, calculated using:

$$w_{n+1} = \frac{d_{k\to n}}{d_{k\to n+1}} \tag{3.34}$$

and

$$w_n = \frac{d_{n\to n+1}}{d_{k\to n+1}}. \tag{3.35}$$

In the equations 3.35 and 3.34, the notation $d_{k\to n}$ refers to a distance measurement, fully explained by Zhang, Ashuri and Deng (2017), but essentially, this distance, $d_{k\to n}$, is given in the meaning of horizontal distance between observations n and k, therefore, considering points (t_k, y_k) and (t_n, y_n), hence $d_{k\to n} = t_n - t_k$.

Despite the MXA being an innovative method, it is necessary to comment on the fact that the estimator generated by this method is not necessarily connected with the past value that generated it. The natural visibility graph maps a time series into a complex network based on the direct visibility criteria, that is, two points in the time series becomes a linked pair of vertices in the network, if and only if, there is an unobstructed straight line connecting both points in the time series plot.

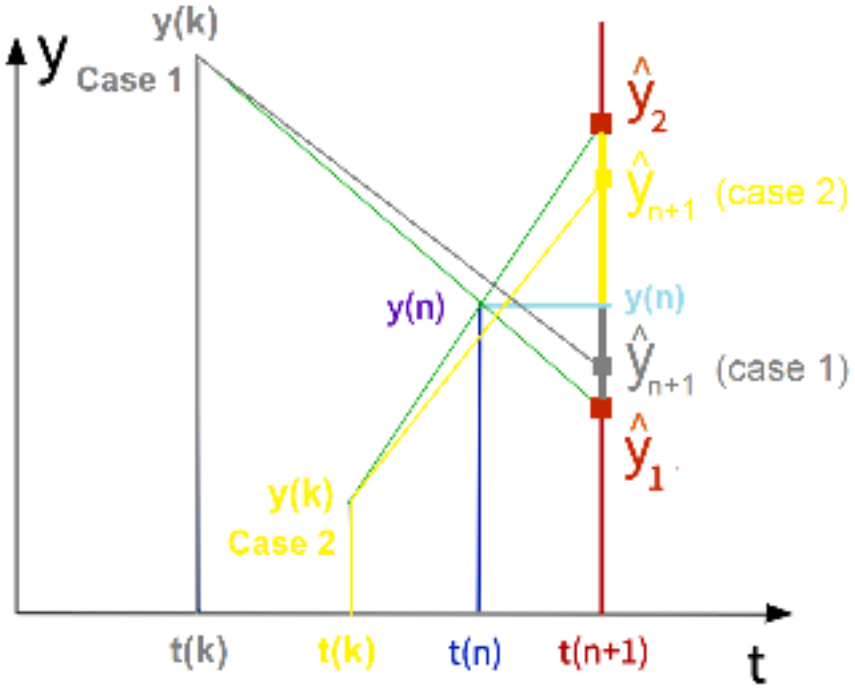


Figure 3.5: Analysis of the MXA forecasts in the visibility graph approach. Source: Drawn by the authors

Consider the Case 1, illustrated in the Figure 3.5. In this case, y_k is greater than y_n and the MXA-estimator is $\hat{y}_{n+1}$, in gray. Note that $\hat{y}_{n+1}$ is visually connected to y_k and y_n, thus, in the correspondent network produced by the visibility graph, $\hat{y}_{n+1}$ is linked to y_k and y_n . Now observe the Case 2, where $y_k < y_n$. The MXA-estimator is $\hat{y}_{n+1}$, in yellow, but in this case, note that the yellow line connecting $\hat{y}_{n+1}$ and y_k is obstructed by the y_n bar , so, this estimator is not linked to y_k in the network domain.

3.11 Dynamic ARIMA

Automatic ARIMA is a method used to calculate the best one-step-ahead ARIMA(p,d,q) estimate considering a given dataset. In this work, we use the version of Auto-ARIMA given by a variation of the Hyndman-Khandakar algorithm (HYNDMAN; KHANDAKAR, 2008), available in R. The best ARIMA(p,d,q) model is found after combining the unit root tests, minimization of the second-order Akaike information criterion (AICc), and maximum likelihood estimation (MLE). All steps of this algorithm are detailed in Hyndman and Athanasopoulos (2018)

The Dynamic ARIMA (DA) method provides one-step-ahead forecasting considering a sliding window and using the ARIMA method. The term *Dynamic* is due to the application of the sliding window process, which constantly changes the dataset used to provide an one-step-ahead forecasting. Consider a n-element time series, and a test set composed by the last

m observations, $y_{n-m+1}, \cdots, y_n$. We calculate $\hat{y}_{n-m+1}$ based on all observations from y_1 to y_{n-m}. Next, we apply the sliding window, so we used all data, from estimate y_1 to y_{n-m+1} to calculate $\hat{y}_{n-m+2}$. We repeat this procedure until calculate $\hat{y}_n$ based in all observations from y_1 to y_{n-1}. As it can be seen, we do not use one single model ARIMA to calculate all m estimates of the test set, instead, we find m ARIMA models, each one taking all the last observations to calculate the next one.

3.12 Support Vector Regression

The Support Vector Machines (SVM) are a common machine learning method dedicated to classification and regression (CHANG; LIN, 2011). The SVMs variant for regression is called Support Vector Regression (SVR). The SVR maps the original data into a high-dimensional feature space using a nonlinear mapping and next calculate a linear regression in the feature space (ZHOU *et al.*, 2009). According to described in Chang and Lin (2011), consider a training set where each k-element window $\mathbf{Z}_i$ is associated to a target point y_{k+1}. The function approximation is performed as follows

$$y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\phi(\mathbf{Z}) + b \tag{3.36}$$

where $\phi(\mathbf{Z}_i)$ is a nonlinear function and $\mathbf{w}$ is the k-element weight vector. According to Samsudin, Shabri and Saad (2010), taking the parameters $C > 0$ and $\epsilon > 0$, the values of b and $\mathbf{w}$ are calculated by the minimization of the function

$$R(C) = 0.5\mathbf{w}^{\mathrm{T}}\mathbf{w} + \frac{C}{k}\sum_{i=1}^{k} L_t\left(d_i, y_i\right) \tag{3.37}$$

where d_i is the desired value and $L_t\left(d_i, y_i\right) = |d_i - y_i| - \epsilon$ if $|d_i - y_i| > \epsilon$ and zero, otherwise. The equation (3.37) is transformed into its primal function by the introduction of two positive slack variables ξ_i and ξ_i^*. The standard representation of support vector regression in the form of a quadratic programming task

$$\begin{aligned} \min_{\mathbf{w},b,\xi,\xi^*} \quad & 0.5\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\sum_{i=1}^{k}\xi_i + C\sum_{i=1}^{k}\xi_i^* \\ \text{subject to} \quad & \mathbf{w}^{\mathrm{T}}\phi(\mathbf{Z}_i) + b - y_i \leq \epsilon + \xi_i, \\ & y_i - \mathbf{w}^{\mathrm{T}}\phi(\mathbf{Z}_i) - b \leq \epsilon + \xi_i^*, \\ & \xi_i, \xi_i^* \geq 0, i = 1, \cdots, k. \end{aligned} \tag{3.38}$$

The parameter ϵ is called as the tube size of the SVR and is equal to the approximation accuracy obtained on the training data points. Given Chang and Lin (2011), the dual problem related to

the equation (3.38) is

$$\begin{aligned} \min_{\alpha,\alpha^*} \quad & 0.5\left(\alpha-\alpha^*\right)^{\mathrm{T}} Q\left(\alpha-\alpha^*\right)+\epsilon \sum_{i=1}^{k}\left(\alpha_i+\alpha_i^*\right)+\sum_{i=1}^{k} y_i\left(\alpha_i+\alpha_i^*\right) \\ \text{subject to} \quad & \mathbf{U}^{\mathrm{T}}\left(\alpha-\alpha^*\right)=0, \\ & 0 \leq \alpha_i, \alpha_i^* \leq C, i=1, \cdots, k. \end{aligned} \tag{3.39}$$

where $\mathbf{U}$ is a k-element vector of one's, $Q_{i,j} = K\left(\mathbf{Z}_i, \mathbf{Z}_j\right) = \phi(\mathbf{Z}_i)^{\mathrm{T}}\phi(\mathbf{Z}_j)$ with $K\left(\mathbf{Z}_i, \mathbf{Z}_j\right)$ known as the kernel function, and α and α^* are Lagrange multipliers. According to Zhou *et al.* (2009), Chang and Lin (2011) and Ma and Guo (2014) there are some commonly used kernel functions, to be known:

1. Linear: $K\left(\mathbf{Z}_i, \mathbf{Z}_j\right) = \mathbf{Z}_i^{\mathrm{T}}\mathbf{Z}_j$.
2. Polynomial: $K\left(\mathbf{Z}_i, \mathbf{Z}_j\right) = \left(\mathbf{Z}_i^{\mathrm{T}}\mathbf{Z}_j + r\right)^d$, with $d \in \{2, 3, \cdots\}$.
3. Sigmoid: $K\left(\mathbf{Z}_i, \mathbf{Z}_j\right) = tanh\left(\gamma \mathbf{Z}_i^{\mathrm{T}}\mathbf{Z}_j + r\right), \gamma > 0$.
4. Radial Basis Function (RBF): $K\left(\mathbf{Z}_i, \mathbf{Z}_j\right) = exp\left(-\gamma\left(\mathbf{Z}_i - \mathbf{Z}_j\right)^{\mathrm{T}}\left(\mathbf{Z}_i - \mathbf{Z}_j\right) + r\right), \gamma > 0$.

Also, Rohmah *et al.* (2021) presents a comparison among four SVR kernels, including linear, polynomial and RBF, with the objective to provide predictions to an econometric time series.

The approximation function is calculated by

$$y\left(\mathbf{Z}_i\right) = \sum_{i=1}^{k}\left(\alpha_i + \alpha_i^*\right) K\left(\mathbf{Z}_i, \mathbf{Z}_j\right) + b. \tag{3.40}$$

3.13 Multilayer Perceptron Model

According to Kaushik *et al.* (2020), a Multilayer Perceptron (MLP) is a variant of the original model proposed in Rosenblatt (1961).The Figure 3.6 illustrates the original perceptron proposed in Rosenblatt (1961).

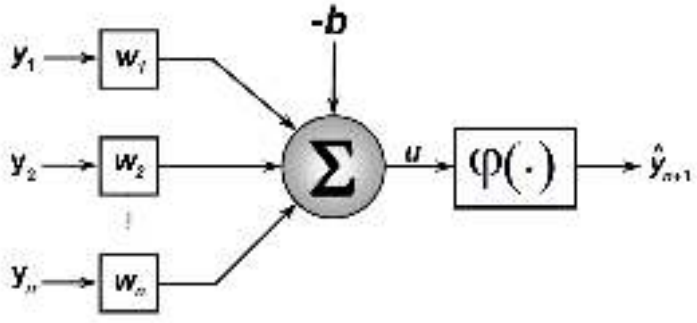


Figure 3.6: The original perceptron proposed in Rosenblatt (1961). Figure Source: Drawn by the author.

The neuron functioning occurs according to described in equation (3.41):

$$\hat{y}_{n+1} = \phi\left(\sum_{i=1}^{n} y_i w_i - b\right), \tag{3.41}$$

where $\phi(\tau)$ is the activation function, y_i is the i-th input, n is the number of inputs, w_i is the weigh related to the input i, and b is the bias associated to the neuron (KAUSHIK *et al.*, 2020). Describing equation (3.41), a neuron calculates a weighted sum of the inputs and this result is processed by a nonlinear activation function. The neural network architecture is considered as the universal function approximation due the presence of the activation functions (KAUSHIK *et al.*, 2020). The mappings from inputs to outputs are produced with influence of an activation function and this makes the neural network able to learn complex data representations (CHUNG; LEE; PARK, 2016). According to Chung, Lee and Park (2016) and krizhevsky, Sutskever and Hinton (2012), the most common activation functions presented in the literature are sigmoid, hyperbolic tangent (tanh) and ReLu functions. We can observe in krizhevsky, Sutskever and Hinton (2012) discussions about the sigmoid and tanh activation functions suffer from the vanishing gradient problem and about ReLu activation function overcome this problem. Based on krizhevsky, Sutskever and Hinton (2012), ReLu also presents faster convergence and is computationally efficient to do calculations.

The MLP mathematical and structural model is presented by Kaushik *et al.* (2020). The training assignment is done based on the backpropagation approach, which is a supervised learning method (SHIBLEE; KALRA; CHANDRA, 2008). The MLP model is composed by three blocks: the input layer, the hidden layer and the output layer. The Figure 3.7 shows the structure of a multiple input and single output (MISO) MLP model, composed by one hidden layer.

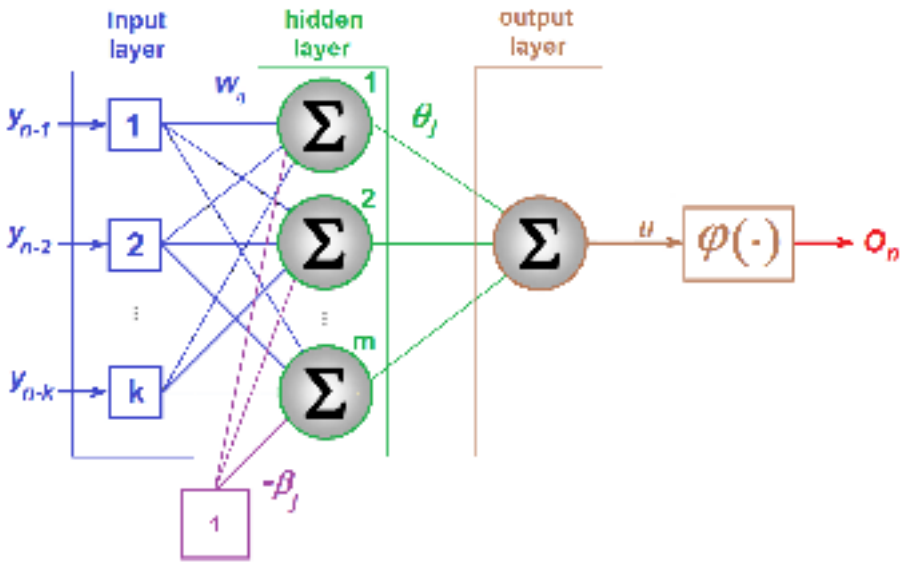


Figure 3.7: The structure of the MLP model. Figure Source: Drawn by the author.

The MLP model calculates the output according to described in the equation (3.42):

$$O_n = \phi\left(\sum_{j=1}^{m}\theta_j \sum_{i=1}^{k} y_{n-i}w_{ij} - \beta_i\right), \tag{3.42}$$

where m is the number of neurons in the hidden layer. A typical MLP model presents multiple hidden layers, each one with multiple neurons and every neuron in the h-th hidden layer is fully connected to the every neuron in the $(h+1)$-th hidden layer.

3.14 Long Short Term Memory Model

Deep learning methods such as recurrent neural network and long short-term memory have recently being widely used to solve problems in computer vision, time series forecasting, natural language processing, finance and other fields (NGUYEN; PHAN; ZELINKA, 2021). Since the LSTM (Long Short-Term Memory) is an architecture inspired in the Recurrent Neural Network (RNN) (RUMELHART; HINTON; WILLIAMS, 1986), before starting the discussion about LSTM, we will present some basic information about the RNN structure. The Recurrent Neural Networks (RNN) are specifically designed to work on data in sequence including time series forecasting. According to Rumelhart, Hinton and Williams (1986), the traditional feed forward neural networks are not able to deal with the time dependence naturally present in the real time series and the RNNs are designed to solve this problem.

Standard basic RNNs face the vanishing gradient problem, where the gradient decreases as rises the number of layers. For example, the deep RNNs with a high number of layers, presents almost null gradient what compromise the network learning ability (NGUYEN; PHAN;

ZELINKA, 2021). According to Hochreiter and Schmidhuber (1997), this problem observed in the RNNs is the explanation of why these networks have a short-term memory and tend to struggle when dealing to the necessity of memorizing all the information contained in long sequence of data. The LSTM recurrent network comes to solve the vanishing gradient problem (HOCHREITER; SCHMIDHUBER, 1997).

LTSM is composed by three gates which work together in order to keep longstanding important information and forget the irrelevant ones (TORRES *et al.*, 2020). Also, according to Torres *et al.* (2020) These gates are Γ^f the forget gate, Γ^u the update gate and Γ^o the output gate. Γ^f is responsible to choose what information has to be discard or kept. Γ^f near to zero, means forgetting the information, whilst values close to 1 means the information must be kept. The gate Γ^u works deciding if the new information $\tilde{c}_t$ will be used to update the c_t memory state. Whether c_t has to be updated it happens through Γ^u and Γ^f. Lastly, Γ^o makes decision about which output value be placed as input of the next hidden unit. The equations related to every step of the LSTM work are:

$$\tilde{c}_t = \tanh\left(W_c \cdot [a_{t-1}, y_t] + b_c\right) \tag{3.43}$$

$$\Gamma^u = \phi\left(W_u \cdot [a_{t-1}, y_t] + b_u\right) \tag{3.44}$$

$$\Gamma^f = \phi\left(W_f \cdot [a_{t-1}, y_t] + b_f\right) \tag{3.45}$$

$$\Gamma^o = \phi\left(W_o \cdot [a_{t-1}, y_t] + b_o\right) \tag{3.46}$$

$$c_t = \Gamma^u \cdot \tilde{c}_t + \Gamma^f \cdot c_{t-1} \tag{3.47}$$

$$a_t = \Gamma^u \cdot \tanh\left(c_t\right) \tag{3.48}$$

where W_u, W_f and W_o, and b_u, b_f and b_o are the weights and biases related to the update, forget and output gates, respectively, and W_c and b_c are the weights and bias with respect to the $\tilde{c}_t$, that is, the memory cell candidate (TORRES *et al.*, 2020).

3.15 Hybrid ARIMA-ANN

In agreement to was discussed in the section 3.9, the ARIMA model adjusts a linear combination of the past observations considering a stationary data. In the case of the data is non-stationary, differences has to be performed till the data becomes stationary. Unlike ARIMA, the Artificial Neural Network (ANN) is a non-linear approach, and as well as the MLP structure, the ANN model is composed by three blocks: the input layer, the hidden layer and the output layer (KANG, 1992). Each layer is formed by the inclusion of one or more neurons. In order to solve the time series forecasting problem the output has one single neuron. The Figure 3.8 shows the structure of a multiple input and single output (MISO) ANN model, composed by one hidden layer.

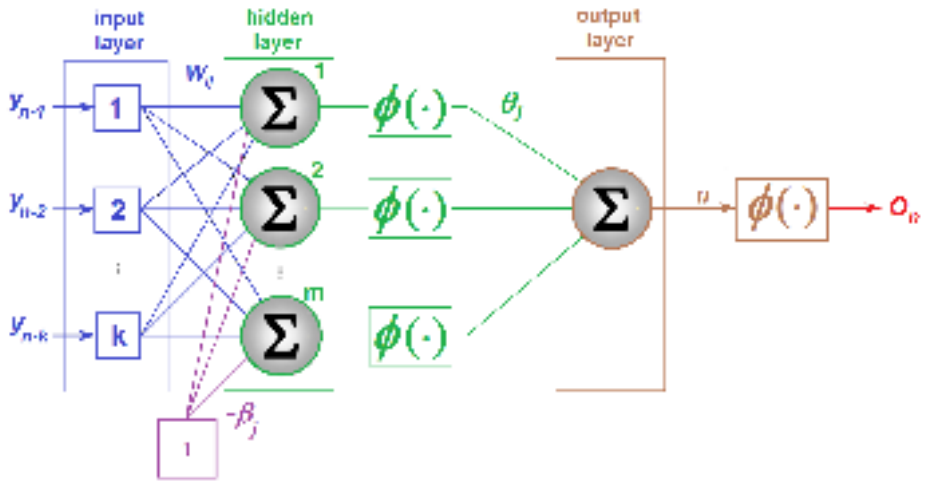


Figure 3.8: The structure of the ANN model. Figure Source: Drawn by the author.

We are presenting an architecture composed by one hidden layer, but the ANN structure can have others. The hidden layer are composed from any number of nodes, whose outputs are considered inputs for the next layer. The input layer are fed with one or more neurons depending on the size of the window used in the forecasting task. According to Chindanur and B (2015), three-layer ANNs are widely used for time series forecasting.

The real time series generally present linear and non-linear characteristics, so standalone techniques purely linear or non-linear can present weak accuracy when they deal with these datasets to produce forecasts (CHINDANUR; B, 2015). The idea of using a hybrid approach combining ARIMA and ANN is to benefit from the main characteristic of both methods: linearity, in the case of ARIMA and non-linearity from ANN.

According to Khashei and Bijari (2011) a hybrid ARIMA-ANN model was proposed to provide time series forecasting. The technique consider that any real time series is composed by linear and non-linear components: $y_t = L_t + N_t$. An ARIMA model is used to capture the linear component, L_t. Next, the past original values, the prediction to the current term, and past error data are used as inputs to an ANN model. After training and validation, the obtained model is used to provide the one-step-ahead forecasting (KHASHEI; BIJARI, 2011).

$$\hat{y}_{t+1} = f\left(\hat{L}_t, e_1, e_2, \cdots, e_k, y_{t-1}, y_{t-2}, \cdots y_{t-k}\right) \tag{3.49}$$

where $e_i = y_{t-i} - \hat{y}_{t-i}$. The Figure 3.9 illustrates the Khashei-Bijari algorithm to produce the combination between ARIMA and ANN.

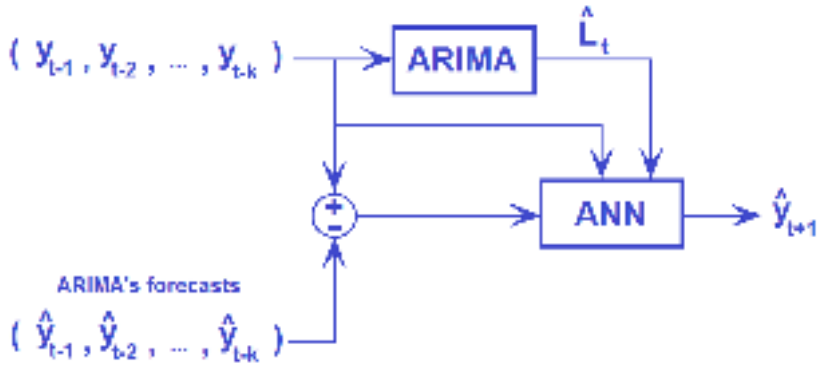


Figure 3.9: The Khashei-Bijari algorithm for the hybrid ARIMA-ANN model. Figure Source: Drawn by the author.

It is important to clarify that the Dynamic ARIMA (DA) method provides one-step-ahead forecasting considering a sliding window and using the ARIMA method. The ARIMA is applied multiple times, every interaction a new model ARIMA(p, d, q) is used, considering a different dataset, that are, the windows. Consider a n-element time series, and a test set composed by the last m observations, $y_{n-m+1}, \cdots, y_n$. The element $\hat{y}_{n-m+1}$ is calculated based on all observations from y_1 to y_{n-m}. After that, the sliding window is applied, changing the next dataset by including the element y_{n-m+1}, so we used all data, from y_1 to y_{n-m+1} to calculate $\hat{y}_{n-m+2}$, which represents a estimate for the real term y_{n-m+2}. We repeat this procedure until calculate $\hat{y}_n$ based in all observations from y_1 to y_{n-1}. This was the procedure adopted to allow ARIMA gives its best results, because we observed that the h-step-ahead using ARIMA normally led to very imprecise results when applied to real time series. Ratifying, as it can be seen, we do not use one single model ARIMA to calculate all m estimates of the test set, instead, we find m ARIMA models, each one taking all the last observations to calculate the next one.

3.16 Hybrid ETS-ANN

According discussed in the section 3.5.6, the ETS models are composed by a family of time series models with a basal state space model containing an error term (E) and a level, trend (T) or seasonal (S) components. Despite the ETS models are not always considered an linear approach, in the hybrid version with ANN, the ETS produces the linear component of the real time series. According to Purohit *et al.* (2021) the difference between the time series observations and the ETS forecasts feed the ANN model. The ANN output are then summed to the current ETS forecast producing the hybrid one-step-ahead forecast. The Figure 3.10 illustrates the algorithm to produce the additive combination between ETS and ANN according to proposed by Purohit *et al.* (2021).

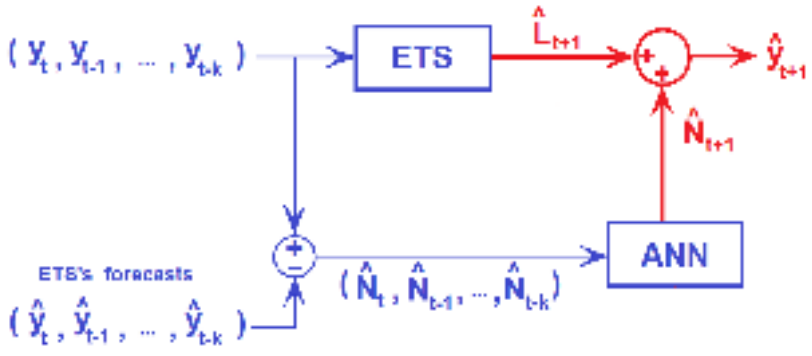


Figure 3.10: Algorithm for the hybrid Additive-ETS-ANN model presented by Purohit *et al.* (2021). Figure Source: Drawn by the author.

The hybrid additive approach combining ETS and ANN was implemented according to described in (PUROHIT *et al.*, 2021). The ETS-models were defined based on the R function "ets" from the package forecast. This function is able to calculate the best ETS model which fits with the given time series. As well as done with ARIMA, we also used the sliding window to create multiple ETS models to estimate the next term: each model was obtained based on the input data, that is, the window, in each interaction of the simulation.

Consider a n-element time series, and a test set composed by the last m observations, $y_{n-m+1}, \cdots, y_n$. The element $\hat{y}_{n-m+1}$ is calculated based on all observations from y_1 to y_{n-m}. After that, the sliding window is applied, changing the next dataset by including the element y_{n-m+1}, so we used all data, from y_1 to y_{n-m+1} to calculate $\hat{y}_{n-m+2}$. We repeat this procedure until calculate $\hat{y}_n$ based in all observations from y_1 to y_{n-1}. This was the procedure adopted to allow ETS gives its best results. We do not use one single model ETS to calculate all m estimates of the test set, instead, we find m ETS models, each one taking all the last observations to calculate the next one. Initially, the set with the pure ETS estimates were created and next the combination with ANN was provided according to illustrated in the Figure 3.10.

The Table 5.11 shows the best number or nodes in the hidden layer for each time series. These parameters were defined from the accuracy obtained on the training and validation sets.

4 New Forecasting Methods

4.1 Method 1: Maximum Visibility Approach

The Maximum Visibility approach (MVA) is a new method based on the complex network theory, inspired by the technique proposed in Mao and Xiao (2019), which, in this work, is called the Mao-Xiao approach (MXA). The forecasting process is conducted based on a time series that presents temporally correlated data; Thus, it is reasonable to research which past terms may contain relevant information to assist in the process of forecasting the next term in the time series (OZAKI, 2012).

The first step of the MVA considers using the autocorrelation function (ACF), related to the given time series, to find the window size. The autocorrelation function (ACF) defines how data points in a time series are related, on average, to the preceding data points (BOX *et al.*, 1994). We calculate the sequence of autocorrelations between the last observation and each previous one: $\{\rho_{n,n}, \rho_{n,n-1}, \cdots, \rho_{n,1}\}$. Next each pair $(k, \rho_{n,n-k})$ is plotted and in the same graph we plot the 95% confidence bounds for strict white noise. In order to do the forecasting task, we only use the observations that precede the first entry of the region of statistical insignificance. This can configure a loss of information in the forecasting process, but empirically, it was observed that when all last observations were used in the MVA's forecasting task the increase of time was not compensated by the gain in accuracy considering the training and validation sets.

Figure 4.1 illustrates the autocorrelation function of the time series composed by the monthly number of passengers for international airlines, in thousands of people, from 1949 to 1960, presented in Box and Jenkins (1976). We can observe in Figure 4.1, that the first observation immediately before the first entry into the region of statistical insignificance is y_{n-40}. In this case, it becomes acceptable to consider the impact of up to the first 40 previous data prior to y_n in the modeling task and the window size is 40.

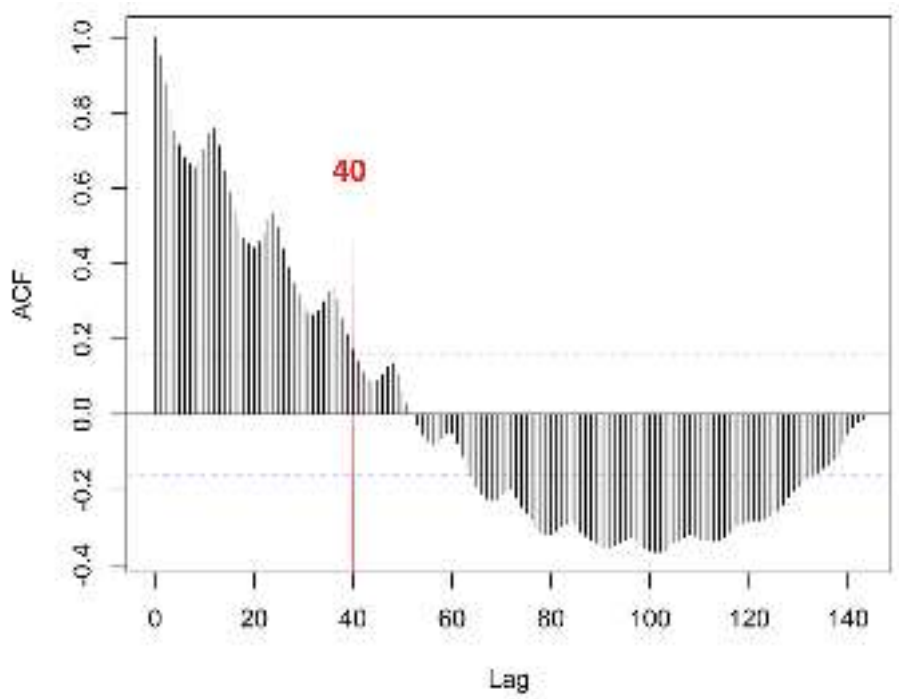


Figure 4.1: Autocorrelation Function from the time series of the number of monthly total of international airline passengers, in thousands of people, from 1949 to 1960. Figure Source: Drawn by the authors.

It is important to comment that, in order to find the window size, the Partial Autocorrelation Function (PACF) could be used instead. However, as each term of this sequence, $\rho_{y_n,y_{n-k}\setminus\mathbf{Y}_{ex}}$, do not consider the influence of the terms between y_n and y_{n-k}, the sequence converges faster to the statistical insignificance region, which would give a smaller window size, hence the loss of information could be yet higher.

The next step is to transform the given time series into a complex network. The literature presents some methods which maps a given time series into a complex network: visibility graph (LACASA *et al.*, 2008), horizontal visibility graph (LUQUE *et al.*, 2010), multiscale limited penetrable horizontal visibility graph (GAO *et al.*, 2016) and phase space reconstruction (GAO; JIN, 2009). In this work, we chose the visibility graph approach due to its ease of implementation, low computational cost and because the constructed graph inherits several properties of the series in its structure (LACASA *et al.*, 2008). Additionally, Lacasa *et al.* (2008) shows that periodic series convert into regular graphs, random series do so into random graphs and fractal series convert into scale-free networks.

After constructing the network, we calculated the node similarity matrix (NSM) . In this matrix, each $s_{i,j}$ represents the similarity calculated between nodes N_i and N_j.

Let $\hat{y}_{n+1|k}$ be the pre-estimator for the time series next term, that is, y_{n+1}, considering only the influence of y_k. This pre-estimator is calculated using a linear equation, based on the connection between both observations: y_k and y_n. We introduced a multiplicative parameter, α, to the original angular coefficient, as shown in (4.1). Consider the points: (t_k, y_k) and (t_n, y_n).

The expression that gives $\hat{y}_{n+1|k}$ is:

$$\hat{y}_{n+1|k} = y_n + \alpha \left(\frac{y_n - y_k}{t_n - t_k} \right) \cdot (t_{n+1} - t_n), \tag{4.1}$$

where α is divided into two parts: a constant $K \geq 0$ and $\rho_{n,k}$, which are the autocorrelation between y_k and y_n.

$$\alpha = \rho_{n,k} - K. \tag{4.2}$$

Observing the equation (4.1), the closer α is to zero, the greater is the effect of y_n on $\hat{y}_{n+1}$, approximating the forecasting result to that obtained from the naive estimation. We define $\mathbf{Y}_{n+1}^m$ as the vector that contains all pre-estimators for y_{n+1}, calculated through the influence of the first m lags. $\mathbf{Y}_{n+1}^m$ is written as:

$$\mathbf{Y}_{n+1}^m = \left(\hat{y}_{n+1|n-m+1}, \cdots, \hat{y}_{n+1|n-1} \right). \tag{4.3}$$

Let $s_{i,n}$ be the similarity value between nodes N_i and N_n, associated to the time series terms y_i and y_n, respectively and calculated based on the Dice similarity, according to described in the equation (2.75). If the last m nodes, prior to y_n are considered in order to estimate $y_{(n+1)}$, then the NSM has dimension $m+1$ and the last row of the NSM contains the similarities between all last m observations and y_n. We define $\mathbf{S}_n^m$ as the vector formed by the first m elements of the last row of the NSM. Therefore, $\mathbf{S}_n^m$ is seen as:

$$\mathbf{S}_n^m = (s_{n-m+1,n}, s_{n-m+2,n}, \cdots, s_{n-1,n}). \tag{4.4}$$

It is possible to find $s_{k,n} = s_{i,n}$ with $k \neq i$, therefore more than one node can share the highest level of similarity with y_n. Define $\mathbf{I}_{\mathbf{S}_n^m}$ as the vector composed by the indexes of the elements contained in $\mathbf{S}_n^m$ which present the highest level of similarity to y_n. The one-step-ahead MVA estimator, $\hat{y}_{n+1}$ is calculated as:

$$\hat{y}_{n+1} = MAX \left\{ \hat{y}_{n+1|k} \in \mathbf{Y}_{n+1}^m \right\} + h(E_n), \forall k \in \mathbf{I}_{\mathbf{S}_n^m}, \tag{4.5}$$

where $h(E_n)$ is a non-linear function of the error obtained in the estimation of y_n, that is $E_n = \hat{y}_{(n)} - y_n$. The non-linear function $h(E_n)$ is given by:

$$h(E_n) = -E_n e^{-K \cdot J |E_n|}, \tag{4.6}$$

where J is a positive constant.This parameter J must be positive because otherwise, the exponent of the non-linear term can be highly positive and this will produce a divergent sequence of estimates. For some time series, $h(E_n)$ is considered zero. The decision of not include $h(E_n)$ in (4.5) is taken by observing the results achieved by MVA in the validation phase.

Based on the result expressed by the equation (4.5) one can observe that, in practice, the one-step-ahead estimate is calculated using just one observation. This may represent a loss of information because there are other previous observations highly similar to the last one, however, taking another function of some or all the pre-estimators in order to provide the one-step-ahead estimate represents another forecasting model which must be studied as a future work.

4.1.1 Statistical Measures of the MVA Estimator

Since $\hat{y}_{n+1}$ is a random variable, we must calculate its expected value and its variance. Consider the definition of $\hat{y}_{n+1}$ without the nonlinear term defined in (4.6). The inclusion of this term creates an analytic problem since the statistical measures would be undefined.

We define $T_v = t_n - t_v$ and since $t_{n-1} - t_n = 1$, the equation (4.1) can be rewritten as

$$\hat{y}_{n+1|k} = y_n + \alpha\left(\frac{y_n - y_k}{T_k}\right) = \left(1 + \frac{\alpha}{T_k}\right) y_n - \left(\frac{\alpha}{T_k}\right) y_k. \tag{4.7}$$

It is known that, for any random variable X and Y, and any real numbers a and b, the $Var\left(aX - bY\right) = a^2 Var\left(X\right) + b^2 Var\left(X\right) - 2abCOV\left(X, Y\right)$. Given the non-negative characteristic of the variance, it is impossible to achieve negative values of variance, no matter the values of a and b. Based on this result, considering (4.7), and using the properties of expectancy, which is a linear operator, we obtain the variance of $\hat{y}_{n+1|k}$, that is a non-negative number, no matter the value of α.

$$Var\left(\hat{y}_{n+1|k}\right) = Var\left(y_n\right)\left(1 + \frac{\alpha}{T_k}\right)^2 + Var\left(y_k\right)\left(\frac{\alpha}{T_k}\right)^2 - \frac{2\alpha}{T_k}\left(1 + \frac{\alpha}{T_k}\right) COV\left(y_n, y_k\right) \tag{4.8}$$

The covariance between $\hat{y}_{n+1|k}$ and $\hat{y}_{n+1|j}$, for $k \neq j$, is given by:

$$\begin{aligned} &COV\left(\hat{y}_{n+1|k}, \hat{y}_{n+1|j}\right) = \\ &= Var\left(y_n\right)\left(1 + \frac{\alpha}{T_k}\right)\left(1 + \frac{\alpha}{T_j}\right) - \frac{\alpha}{T_j}\left(1 + \frac{\alpha}{T_k}\right) COV\left(y_n, y_j\right) - \\ &- \frac{\alpha}{T_k}\left(1 + \frac{\alpha}{T_j}\right) COV\left(y_n, y_k\right) + \frac{\alpha^2}{T_k T_j} COV\left(y_k, y_j\right). \end{aligned} \tag{4.9}$$

Consider $MAX\left\{\hat{y}_{n+1|k} \in \mathbf{Y}_{n+1}^m\right\} = \hat{y}_{n+1|v}$, thus $\hat{y}_{n+1} = \hat{y}_{n+1|v}$ and the expected value of $\hat{y}_{n+1}$ is given by

$$E\left[\hat{y}_{n+1}\right] = E\left[y_n\right] + \frac{\alpha}{T_v}\left(E\left[y_n\right] - E\left[y_v\right]\right). \tag{4.10}$$

The variance of $\hat{y}_{n+1}$ is given by

$$Var\left(\hat{y}_{n+1}\right) = Var\left(y_n\right)\left(1+\frac{\alpha}{T_v}\right)^2 + Var\left(y_v\right)\left(\frac{\alpha}{T_v}\right)^2 - \frac{2\alpha}{T_v}\left(1+\frac{\alpha}{T_v}\right)COV\left(y_n, y_v\right). \tag{4.11}$$

Take the case of a white noise process, that is, $\{y_t\}_{t=1}^{t=n}$ independent and identically distributed, therefore $Var\left(y_k\right) = \sigma^2, \forall k \in \{1, 2, \ldots, n\}$ and $COV\left(y_k, y_j\right) = 0, \forall k \neq j$. With these conditions, equations (4.8), (4.9) and (4.11) are reduced to:

$$Var\left(\hat{y}_{n+1|k}\right) = \left(\left(1+\frac{\alpha}{T_k}\right)^2 + \left(\frac{\alpha}{T_k}\right)^2\right)\sigma^2, \tag{4.12}$$

$$COV\left(\hat{y}_{n+1|k}, \hat{y}_{n+1|j}\right) = \left(1+\frac{\alpha}{T_k}\right)\left(1+\frac{\alpha}{T_j}\right)\sigma^2, \tag{4.13}$$

and

$$Var\left(\hat{y}_{n+1}\right) = \left(\left(1+\frac{\alpha}{T_v}\right)^2 + \left(\frac{\alpha}{T_v}\right)^2\right)\sigma^2. \tag{4.14}$$

Given the definition, $Var\left(X\right) = E\left[\left(X - E\left[X\right]\right)^2\right]$, $Var\left(\hat{y}_{n+1}\right) = f\left(\alpha\right)$ is a non-negative quadratic function that can be minimized. According to (4.2), K is the parameter of the MVA estimator which has to be adjusted in order to produce the minimum variance. Let $\hat{K}$ be the value of K that minimizes $Var\left(\hat{y}_{n+1}\right)$, thus, $\hat{K}$ also minimizes the sum of squared errors produced by $\hat{y}_{n+1}$. By calculating $\frac{\partial SSE\left(\hat{y}_{n+1}\right)}{\partial K} = 0$ the value of $\hat{K}$ can be derived.

Take a n-element time series $\{y_1, \cdots, y_n\}$, and a window of size w. All of these n elements will be used in the forecasting task. The observations from y_1 to y_w will be used to produce $\hat{y}_{w+1}$. In order to calculate $\hat{K}$, we consider the $n - w$ forecasts obtained from $\hat{y}_{w+1}$ to $\hat{y}_n$, then the calculated $\hat{K}$ is applied to the last window $\{y_{n-w+1}, y_{n-w+2}, \cdots, y_n\}$ to produce $\hat{y}_{n+1}$. Without any loss of generality, suppose that v_i is the order of the element in the $i\text{-}th$ window correspondent to the maximum pre-estimator. Just for a better organization of the text, define V as the order of the last observation in the training set, $a = w + 1$ and $T_{v_i} = t_{i-1} - t_{v_i}$. The value of $\hat{K}$ is given by:

$$\hat{K} = \frac{\sum_{i=a}^{V}\left(\left(\frac{y_{i-1} - y_{v_i}}{T_{v_i}}\right)^2 \rho_{i-1,v_i} + \left(\frac{y_{i-1} - y_{v_i}}{T_{v_i}}\right)\left(y_i - y_{i-1}\right)\right)}{\sum_{i=a}^{V}\left(\frac{y_{i-1} - y_{v_i}}{T_{v_i}}\right)^2} \tag{4.15}$$

The proof for the result presented by the equation 4.15 is found in the appendix C.3.

4.1.2 The model parameters

According to presented in the equations (4.2) and (4.6) there are two parameters, K and J, respectively that must be defined before the forecasting process. In order to find the best values for K and J capable of providing the best fit in the data we proposed a partitioning of the original time series into three sets: training, validation and testing sets. The simulation has to be performed on the training and validation sets many times as necessary and after that, the best values of K and J are applied to the testing set to ascertain the method's ability to make predictions.

The separation of the dataset into training, validation and test set has be done, considering that the validation set must be representative of the test set, therefore, we can expect, but not guarantee, that the model fitted in the validation set will present a similar perform in the test set.

In order to generate the results presented in the section 5 we kept J constant and variate K, until find the best sum of squared error (SSE) considering the validation set. For example, initially take K from 1 to 6, using step of 1. Suppose $K = 3$ gives the best SSE in the validation set. Secondly refine the search considering K from 2 to 4, in steps of 0.1. Suppose $K = 2.6$ gives the smallest SSE in the validation set. Now refine the search from 2.5 and 2.7, using steps of 0.01. Suppose $K = 2.66$ gives the smallest SSE. Refine the pursuance one more time, searching from 2.65 to 2.67, with steps of 0.001. Register the value of K associated to the smallest SSE. This procedure is repeated considering other values of J.

We started with $J \in \{1, 10, 100, 500\}$. For most cases, higher values of J produces the effect of not considering the non-linear term. After testing some values of J we create a list of the obtained SSE's. There is no guarantee that the pair (J, K) related to the smallest SSE achieved in the training and validation phase will produce the smallest SSE in the testing set, however, we kept this procedure to find the pair (J, K) to be used in the testing phase.

4.1.3 Algorithm description

We present two algorithms: (1) Steps of the MVA forecasting task, where the only interest is to calculate the MVA one-step-ahead estimator and the parameters K, J are known; and, (2) Steps to provide accuracy comparison against other methods.

Consider a n-element time series $\{y_1, y_2, \cdots, y_n\}$. Firstly we define the window size w. This is done by taking the first $(w - 1)$ lags, which are in the statistical relevance region provided by the ACF graph, see section 3.3.

Algorithm 1: Steps of the MVA forecasting task

Input: (K, J, w)

Output: $\hat{y}_{MV,n+1}$

Data: Y=$\{y_{n-w+1}, \cdots, y_{n-1}, y_n\}$

1) Read the data.

2) Transform the window into a network using the visibility graph approach. See section 2.6.

3) Calculate the Node Similarity Matrix (NSM) using the Dice Similarity approach. See equation (2.75).

4) Create $\mathbf{S}_n^{(w-1)}$, the last row of the NSM. See equation (4.4).

5) Find and register the nodes associated to the the greater value in $\mathbf{S}_n^{(w-1)}$.

6) Calculate the pre-estimators related to the nodes registered in the previous step. See equation (4.1).

7) Calculate $\hat{y}_{MV,(n+1)}$, taking the maximum of the pre-estimators calculated in the previous step. See equation (4.5).

With the purpose of compare MVA accuracy against other forecasting methods, we recommend using the steps presented in the algorithm 2. The sets $\mathbf{K}$ and $\mathbf{J}$ contains the values of k and J, respectively, which will be tested in order to find the best MVA parameters to be used in the test set. According to was explained in the section 3.3 the MVA is applied in the first window of size w, that is, $\{y_1, y_2, \cdots, y_w\}$ with the objective to estimate y_{w+1}. The whole dataset is divided into: training set, $\{y_{w+1}, \cdots, y_{w+t}\}$; validation set $\{y_{w+t+1}, \cdots, y_{w+t+v}\}$; and test set, $\{y_{w+t+v+1}, \cdots, y_n\}$. With the purpose of providing accuracy comparison we merge the training and validation sets into a single set, which is used to find the parameters K^* and J^* which gives the best accuracy. Once these parameters are obtained, we proceed to the one-step-ahead forecasting task using the test set. Firstly the we calculate $\hat{y}_{w+t+v+1}$, then we update the window and calculate $\hat{y}_{w+t+v+2}$. We keep following this procedure until calculate the estimate for the last term, $\hat{y}_n$. It is important to ratify that all of these estimates were calculated using the same parameters K^* and J^* obtained in the merge of the training and validation sets. With the set of estimates $\{\hat{y}_{w+t+v+1}, \cdots, \hat{y}_n\}$ and the real values of the test set $\{y_{w+t+v+1}, \cdots, y_n\}$, we calculate the MAE, MAPE and RMSE in order to compare the MVA accuracy with the other comparative methods.

Algorithm 2: Steps for accuracy comparison against other methods

Input: $(\mathbf{K}, \mathbf{J}, w)$

Output: $\hat{y}_{MV,n+1}$

Data: $TrSet = \{y_{w+1}, y_{w+2}, \cdots, y_{w+t}\}$

// Training Set

Data: $ValSet = \{y_{w+t+1}, y_{w+t+2}, \cdots, y_{w+t+v}\}$

// Validation Set

Data: $TeSet = \{y_{w+t+v+1}, y_{w+t+v+2}, \cdots, y_n\}$

// Test Set

for $j \in \{1, 2, \cdots, n(\mathbf{K}) \cdot n(\mathbf{J})\}$ **do**

  for $i \in \{1, 2, \cdots, t+v\}$ **do**

    Read the i-th window: $\{z_1 = y_i, z_2 = y_{i+1}, \cdots, z_w = y_{w+i-1}\}$.

    Calculate $\hat{y}_{w+i}$ using the algorithm 1.

  Calculate and register $SSE(j) = \sum_{k=t+1}^{t+v-1} (\hat{y}_{w+k} - y_{w+k})^2$.

Take the pair (J^*, K^*) associated to the $\min\{SSE\}$.

for $i \in \{1, 2, \cdots, n-w-t-v\}$ **do**

  Read the i-th window: $\{z_1 = y_{t+v+i}, \cdots, z_w = y_{t+v+w+i-1}\}$.

  Calculate $\hat{y}_{w+t+v+i}$ using the algorithm 1.

  Register $\hat{y}_{w+t+v+i}$.

Calculate MAE, MAPE and RMSE according to (3.2),(3.3) and (3.4), respectively.

Also, if one is only interested in estimate the next term of the time series using MVA but wants to search for the best parameters, then the algorithm 2 applies, but the validation and training sets must merge to contain $(n-w-1)$ elements and the test set has to be composed by the last element y_n. The search for the best parameters can be done based on the methodology described in the section 4.1.2.

4.1.4 MVA Prediction Interval

In the appendix A.12 we present, respectively, the formula and the proper derivation of the prediction interval based on a time series with normal distribution. However, the normal distribution assumption is rarely confirmed and in these cases, another approach is needed in order to obtain the prediction interval. In this particular section we present one solution to this

problem, using the bootstrap simulation, discussed in Efron and Tibshirani (1994).

Consider that $\bar{y}$ is an estimator for y_{n+1}, so the estimation error is $e = y_{n+1} - \bar{y}$ with mean $E[e] = E[y_{n+1}] - E[\bar{y}]$ and variance $Var(e) = Var(y_{n+1}) + Var(\bar{y}) - 2COV(y_{n+1}, \bar{y})$. As consequence, the statistic $\left(\frac{e - E[e]}{s_e}\right) \sim W$, where W has an unknown distribution. Using the same derivation, as presented at the appendix A.13 we find the upper and lower limits of the prediction interval. The non-parametric bootstrap simulation is performed to estimate the distribution of W. Follow the simulation steps for the $i\text{-}th$ interaction, where V is the size of the training set, $i \in \{1, 2, \cdots, T\}$, and T is the size of the test set. In the following steps, we are using bold letters to represent sets.

1. Define the original sample: $\mathbf{Z}_i = \left\{\hat{\mathbf{Y}}_{MV,tr}(1), \cdots, \hat{\mathbf{Y}}_{MV,tr}(V), \hat{\mathbf{Y}}_{MV,te}(1), \cdots, \hat{\mathbf{Y}}_{MV,te}(i)\right\}$.

2. Define $y^i_{n+1} = \hat{\mathbf{Y}}_{MV,te}(i)$, $\mathbf{E}^0_i = y^i_{n+1} \cdot \mathbf{U} - \mathbf{Z}_i$ and $s_{\mathbf{E}^0_i} = \sqrt{\dfrac{\sum_{u=1}^{i+V} \left(E^{0,u}_i - \overline{\mathbf{E}}^0_i\right)^2}{i+V-1}}$, where $\mathbf{U}$ is a set of one's with the same size as $\mathbf{Z}_i$, the subscribe $\overline{\mathbf{X}}$ relates to the arithmetic mean of the elements of set $\mathbf{X}$, and $E^{0,u}_i$ is the $u\text{-}th$ element of $\mathbf{E}^0_i$.

3. For each value of $k \in \{1, 2, \cdots, B\}$, calculate $\mathbf{Z}^k_i$ that is the $k\text{-}th$ re-sample, with replacement, of $\mathbf{Z}_i$ and:

   (a) calculate $E^k_i = y^i_{n+1} - \overline{\mathbf{Z}}^k_i$;

   (b) calculate $s_{\mathbf{E}_{i,k}} = \sqrt{\dfrac{\sum_{t=1}^{k} \left(E^t_i - \overline{\mathbf{E}}_{i,k}\right)^2}{k-1}}$, where $\mathbf{E}_{i,k} = \left\{E^1_i, \cdots, E^k_i\right\}$.

   (c) calculate $W^k_i = \dfrac{E^k_i - \overline{\mathbf{E}}_{i,k}}{s_{\mathbf{E}_{i,k}}}$ and $\mathbf{W}_{i,k} = \left\{W^1_i, \cdots, W^k_i\right\}$.

4. Take $\mathbf{W}_{i,B;0.025}$ and $\mathbf{W}_{i,B;0.975}$, that are, respectively, the 2.5-th and 97.5-th percentiles of $\mathbf{W}_{i,B}$.

5. Calculate the upper, L^u_i and lower, L^l_i limits of the $i\text{-}th$ prediction interval based on the equations 4.16 and 4.17, respectively.

$$L^u_i = \overline{\mathbf{E}}^0_i + \overline{\mathbf{Z}}_i + \mathbf{W}_{i,B;0.975} \cdot s_{\mathbf{E}^0_i} \tag{4.16}$$

$$L^l_i = \overline{\mathbf{E}}^0_i + \overline{\mathbf{Z}}_i + \mathbf{W}_{i,B;0.025} \cdot s_{\mathbf{E}^0_i} \tag{4.17}$$

4.1.5 The Impact of a Superior Outlier on MVA's Prediction

In the case of the existence of a superior outlier in the time series, the Visibility Graph will create two separated networks connected by one single node. This happens because the outlier works as a visibility barrier between the two sets of observations that are before and after the outlier. Naturally, all the information that could be included in the set before the

outlier will not be used in the forecasting process since all observations from this set will share at maximum one common neighbor with the last observation, so the value of the dice similarity will be low or zero. Also, the degree of the node related to the outlier will be the higher in the complex network, so the dice similarity between the outlier and the last observation will be low, hence the outlier will never be chosen in order to calculate the pre-estimator. Based on these considerations, the prejudice related to the presence of an outlier in the time series will be the loss of the information that could be contained in the set of the observations that located before the outlier.

4.2 Method 2: Hybrid Means of Multiple Approaches

The Hybrid Means of Multiple Approaches (HMMA) is a hybrid method formed by the combination of two of more forecasting models. The motivation to implement HMMA was the search for accuracy better than those obtained through MVA and any other forecasting method. For some time series, MVA obtained outstanding results, but for others the comparative methods presented similar results to MVA, considering the values of the error measurements defined by (3.2), (3.3) and (3.4), or the statistical comparison from the Diebold-Mariano test. Inspired by this situation, we defined a hybrid approach considering both results. We ran both techniques, and using weighted quadratic, arithmetic, geometric and harmonic means, we combined $\hat{y}_{MV}$, given by MVA, with $\hat{y}_A$, provided by another approach, to derive four new results. Using simulation, in the section 5, we show that this strategy can be more accurate than MVA and the other method, and in the worst-case scenario, HMMA statistically performed as well as the both models.

4.2.1 HMMA-Arithmetic Mean (HAM)

The first HMMA version is the **HMMA-Arithmetic Mean** which is calculated as:

$$\hat{y}_{HAM,(n+1)} = \beta \cdot \hat{y}_{MV,(n+1)} + (1-\beta) \cdot \hat{y}_{A,(n+1)}, \tag{4.18}$$

where β is calculated according to described in the section 4.2.5.

4.2.2 HMMA-Geometric Mean (H-GM)

The second variant of the HMMA is the **HMMA-geometric mean** which is defined by the product of two factors, f_{MV} and f_A, presented by Equations (4.19) and (4.20).

$$f_{MV} = \left(\left|\hat{y}_{MV,(n+1)}\right|\right)^{\beta}, \tag{4.19}$$

and,

$$f_A = \left(\left|\hat{y}_{A,(n+1)}\right|\right)^{1-\beta}. \tag{4.20}$$

The value of $\hat{y}_{HGM,(n+1)}$ can be derived from Equations (4.21) and (4.22). In the case of $\hat{y}_{MV,(n+1)} > 0$ and $\hat{y}_{A,(n+1)} > 0$,

$$\hat{y}_{HGM,(n+1)} = f_{MV} \cdot f_A. \tag{4.21}$$

But if $\hat{y}_{MV,(n+1)} < 0$ or $\hat{y}_{A,(n+1)} < 0$, hence:

$$\hat{y}_{HGM,(n+1)} = -f_{MV} \cdot f_A. \tag{4.22}$$

4.2.3 HMMA-Harmonic Mean (H-HM)

The third HMMA version is the **HMMA-harmonic mean**. It is defined as the harmonic mean between $\hat{y}_{(MV,(n+1))}$ and $\hat{y}_{(A,(n+1))}$, which is calculated as:

$$\hat{y}_{HHM,(n+1)} = \frac{\hat{y}_{MV,(n+1)}\hat{y}_{A,(n+1)}}{(1-\beta)\,\hat{y}_{MV,(n+1)} + \beta\hat{y}_{A,(n+1)}}. \tag{4.23}$$

4.2.4 HMMA-Quadratic Mean (H-QM)

The fourth and last HMMA variant is the **HMMA-quadratic mean**. It is defined as the quadratic mean between $\hat{y}_{MV,(n+1)}$ and $\hat{y}_{A,(n+1)}$, which is calculated as

$$\hat{y}_{HQM,(n+1)} = \sqrt{\beta(\hat{y}_{MV,(n+1)})^2 + (1-\beta)\,(\hat{y}_{A,(n+1)})^2}. \tag{4.24}$$

4.2.5 Calculating β

We are going to present two ways to calculate the β values. In the first they are calculated based on errors made in estimating observations immediately preceding the current one. Defining $e_1 = y_{n-1} - \hat{y}_{MV,(n-1)}$ and $e_2 = y_{n-1} - \hat{y}_{A,(n-1)}$, if $e_1^2 < e_2^2$, hence:

$$\beta = \frac{e_2^2}{e_1^2 + e_2^2}. \tag{4.25}$$

In the case of $e_1^2 > e_2^2$, so

$$\beta = \frac{e_1^2}{e_1^2 + e_2^2}. \tag{4.26}$$

The second way is produce the values of the parameter β is based on the minimization of the sum of squared errors:

$$SSE_{HAM} = \sum_{i=1}^{n-w-1} \left(y_i - \hat{y}_{HAM,i}\right)^2. \tag{4.27}$$

The expression which gives the β associated with the smallest sum os squared errors

is given by

$$\hat{\beta} = \frac{\sum_{i=1}^{n-w-1} (y_{A,i} - y_i)(y_{A,i} - y_{MV,i})}{\sum_{i=1}^{n-w-1} (y_{A,i} - y_{MV,i})^2}. \tag{4.28}$$

The proof for the result shown in equation 4.28 is presented in the appendix C.4. It is important to comment that the expression of $\hat{\beta}$ presented in the equation (4.28) was calculated considering the arithmetic mean. We did not calculated β estimators based on the other types of means used in this work. In practice, we used the equation (4.28) to calculate the estimators based on the other types of means. The parameter β, in fact, used to provide the HMMA estimator is found by testing both strategies: fixed β, using the equation (4.28), and variable using the equations (4.25) and (4.26), considering the accuracy results obtained in the training and validation sets. In the case of the objective is just to find the one-step-ahead estimator, then the training and validation sets are merged to one unique set containing the observations from y_{w+1} to y_{n-1}.

4.2.6 Algorithm description

Before starting the simulation, for a given time series with n observations, we define the size w of the training sets (window). This is done by taking the first k lags, which are in the statistical relevance region provided by the ACF graph. The simulation followed these steps.

1. Define the window of size w relative to the ith interaction: $\{z_1 = y_i, \cdots, z_w = y_{w+i-1}\}$, considering $i \in \{1, 2, \cdots, n-w-1\}$.
2. For each interaction $i \in \{1, 2, \cdots, n-w-1\}$ calculate $\hat{y}_{A,i}$.
3. For each interaction $i \in \{1, 2, \cdots, n-w-1\}$ calculate $\hat{y}_{MV,i}$.
4. From the vectors $\hat{\mathbf{y}}_A$ and $\hat{\mathbf{y}}_{MV}$, calculate the estimators $\hat{\mathbf{y}}_{HQM}$, $\hat{\mathbf{y}}_{HAM}$, $\hat{\mathbf{y}}_{HGM}$ and $\hat{\mathbf{y}}_{HHM}$ according to equations (4.24), (4.18), (4.22) and (4.23), respectively. Use $\beta = 0.5$ in the first interaction and in all others, consider β according to was described in the section 4.2.5.
5. Calculate the SSE associated to each vector $\hat{\mathbf{y}}_{HQM}$, $\hat{\mathbf{y}}_{HAM}$, $\hat{\mathbf{y}}_{HGM}$ and $\hat{\mathbf{y}}_{HHM}$.
6. Calculate $\hat{y}_{A,n-w}$, $\hat{y}_{MV,n-w}$ and the β associated to the last estimation in the training set.
7. Use the type of mean associated to the smallest SSE, obtained from the training set, to calculate the one-step-ahead HMMA estimator, $\hat{y}_{H,n+1}$, considering the last window, $\{z_1 = y_{n-w+1}, \cdots, z_w = y_n\}$.

4.2.7 Why HMMA Works

In Section 4.2, HMMA is defined as a hybrid method relating MVA and any other method. According to Section 4.1, MVA is a forecasting approach that seeks relations between nodes of a complex network to provide forecasts. In a complex network, two nodes tend to be linked if they have a higher similarity index (MAO; XIAO, 2019). Therefore, taking into account that MVA provides forecasts for y_{n+1}, considering the influence of the previous time series observations that have a greater similarity index with y_n, we can expect that MVA will succeed in reproducing the patterns contained along the time series.

We are sure that MVA is not the best method suitable to provide forecasts considering all time series, therefore, for many other time series, some methods can perform better than MVA in terms of accuracy. However, a combination of MVA and other method always can be proposed in order to seek better accuracy. The question is: how the forecasts produced by MVA and another method can be properly combined?

Section 2.4 presents the inequality means theorem for two positive numbers. When $\hat{y}_{MV,(n+1)}$ and $\hat{y}_{A,(n+1)}$ are applied to equations (4.24), (4.18), (4.22) and (4.23), new one-step-ahead estimators are derived: $\hat{y}_{HQM,(n+1)}$, $\hat{y}_{HAM,(n+1)}$, $\hat{y}_{HGM,(n+1)}$ and $\hat{y}_{HHM,(n+1)}$. The result expressed by equation (2.81) guarantees the following relation:

$$\hat{\mathbf{y}}_{HQM} \geq \hat{\mathbf{y}}_{HAM} \geq \hat{\mathbf{y}}_{HGM} \geq \hat{\mathbf{y}}_{HHM} \tag{4.29}$$

Figure 4.2 illustrates the position of each of the four means created from $\hat{y}_{MV,(n+1)}$ and $\hat{y}_{A,(n+1)}$, in both cases: (a) $\hat{y}_{MV,(n+1)} \leq \hat{y}_{A,(n+1)}$; and, (b) $\hat{y}_{MV,(n+1)} \geq \hat{y}_{A,(n+1)}$. According to Figure4.2,

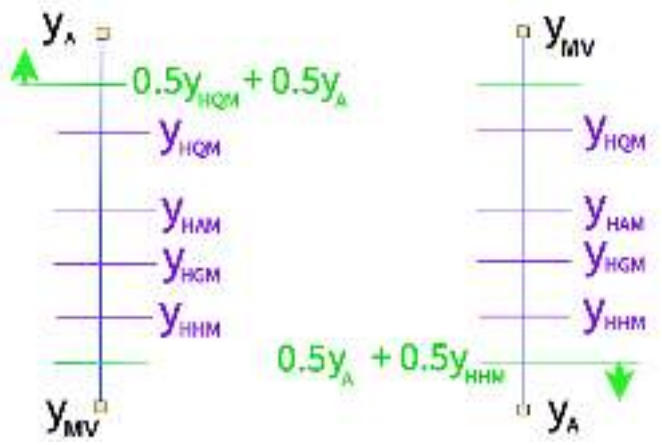


Figure 4.2: Illustration of Means Inequality applied for $\hat{y}_{MV,(n+1)}$ and $\hat{y}_{A,(n+1)}$. Figure Source: Drawn by the authors.

the absolute error will be smaller for the other method if

$$y_{n+1} > \frac{\hat{y}_{HQM,(n+1)} + \hat{y}_{A,(n+1)}}{2}, \tag{4.30}$$

in the case of $\hat{y}_{MV,(n+1)} < \hat{y}_{A,(n+1)}$; or,

$$y_{n+1} < \frac{\hat{y}_{HHM,(n+1)} + \hat{y}_{A,(n+1)}}{2}; \tag{4.31}$$

in the case of $\hat{y}_{MV,(n+1)} > \hat{y}_{A,(n+1)}$. In any other situation, the absolute error will be smaller for HMMA because there are five other estimators that will be near to y_{n+1}. Thus, the probability of y_{n+1} being nearer to any of HMMA's possibilities is greater than y_{n+1} being nearer than only $\hat{y}_{A,(n+1)}$.

Additionally to was explained above, the additive combination can be more accurate, at least to the less accurate between the two estimators considered in the combination. Observe the following theorem stated and proofed by this author.

Theorem 4.1. Consider $\mathbf{X}$ and $\mathbf{W}$ estimates for the known m-dimensional vector $\mathbf{Y}$, such as the sum of squared errors (SSE) for $\mathbf{X}$ is higher than for $\mathbf{W}$. Let a and b positive numbers with $a + b = 1$. Under these conditions, whether $Z = a\mathbf{X} + b\mathbf{W}$, therefore, $SSE(a\mathbf{X} + b\mathbf{W}) < SSE(\mathbf{X})$.

Proof of theorem 4.1. $SSE(\mathbf{X}) = (\mathbf{Y} - \mathbf{X})^T(\mathbf{Y} - \mathbf{X})$, where $\mathbf{Y}^T$ is the transpose of the column matrix $\mathbf{Y}$. Suppose that $SSE(a\mathbf{X} + b\mathbf{W}) \geq SSE(\mathbf{X})$, so:

$$(\mathbf{Y} - a\mathbf{X} - b\mathbf{W})^T(\mathbf{Y} - a\mathbf{X} - b\mathbf{W}) \geq (\mathbf{Y} - \mathbf{X})^T(\mathbf{Y} - \mathbf{X}) \Leftrightarrow$$

$$\Leftrightarrow \mathbf{Y}^T\mathbf{Y}\text{–}a\mathbf{Y}^T\mathbf{X}\text{–}b\mathbf{Y}^T\mathbf{W}\text{–}a\mathbf{X}^T\mathbf{Y} + a^2\mathbf{X}^T\mathbf{X} + ab\mathbf{X}^T\mathbf{W}\text{–}b\mathbf{W}^T\mathbf{Y} + ab\mathbf{W}^T\mathbf{X} + b^2\mathbf{W}^T\mathbf{W} \geq$$

$$\geq \mathbf{Y}^T\mathbf{Y}\text{–}\mathbf{Y}^T\mathbf{X}\text{–}\mathbf{X}^T\mathbf{Y} + \mathbf{X}^T X. \tag{4.32}$$

But $\mathbf{Y}^T\mathbf{X} = \mathbf{X}^T\mathbf{Y}$, $\mathbf{X}^T\mathbf{W} = \mathbf{W}^T\mathbf{X}$ and $\mathbf{W}^T\mathbf{Y} = \mathbf{Y}^T\mathbf{W}$. Hence, from (4.32),

$$2ab\mathbf{X}^T\mathbf{W} + b^2\mathbf{W}^T\mathbf{W} + 2(1 - a)\mathbf{X}^T\mathbf{Y} + (a^2 - 1)\mathbf{X}^T\mathbf{X} \geq 2b\mathbf{Y}^T\mathbf{W}. \tag{4.33}$$

It was given that $SSE(\mathbf{X}) > SSE(\mathbf{W})$, that is:

$$\mathbf{X}^T\mathbf{X} - 2\mathbf{X}^T\mathbf{Y} > \mathbf{W}^T\mathbf{W} - 2\mathbf{Y}^T\mathbf{W} \Leftrightarrow 2\mathbf{Y}^T\mathbf{W} > \mathbf{W}^T\mathbf{W}\text{–}\mathbf{X}^T\mathbf{X} + 2\mathbf{X}^T\mathbf{Y}. \tag{4.34}$$

From (4.33) and (4.34) comes:

$$2ab\mathbf{X}^T\mathbf{W} + b^2\mathbf{W}^T\mathbf{W} + 2(1 - a)\mathbf{X}^T\mathbf{Y} + (a^2 - 1)\mathbf{X}^T\mathbf{X} > b(\mathbf{W}^T\mathbf{W}\text{–}\mathbf{X}^T\mathbf{X} + 2\mathbf{X}^T\mathbf{Y}) \Leftrightarrow$$

$$\Leftrightarrow 2ab\mathbf{X}^T\mathbf{W} + (b^2 - b)\mathbf{W}^T\mathbf{W} + 2(1 - a - b)\mathbf{X}^T\mathbf{Y} + (a^2 - 1 + b)\mathbf{X}^T\mathbf{X} > 0 \tag{4.35}$$

Given that $a + b = 1$, so, the term $2(1 - a - b)\mathbf{X}^T\mathbf{Y} = 0$. Therefore:

$$2ab\mathbf{X}^T\mathbf{W} + (b^2\text{–}b)\mathbf{W}^T\mathbf{W} + (a^2\text{–}1 + b)\mathbf{X}^T\mathbf{X} > 0 \Leftrightarrow$$

$$\Leftrightarrow b(1–b)\mathbf{W}^T\mathbf{W}–2b(1–b)\mathbf{X}^T\mathbf{W} + a(1–a)\mathbf{X}^T\mathbf{X} < 0 \tag{4.36}$$

But: $a(1–a) = b(1–b)$. Hence:

$$b(1–b)\mathbf{W}^T\mathbf{W}–2b(1–b)\mathbf{X}^T\mathbf{W} + b(1–b)\mathbf{X}^T\mathbf{X} < 0 \Leftrightarrow$$

$$\Leftrightarrow b(1–b)(\mathbf{W}^T\mathbf{W}–2\mathbf{X}^T\mathbf{W} + \mathbf{X}^T\mathbf{X}) < 0. \tag{4.37}$$

It is known that $b(1–b)$, by definition, is positive, so $(\mathbf{W}^T\mathbf{W}–2\mathbf{X}^T\mathbf{W} + \mathbf{X}^T\mathbf{X}) < 0$. But,

$$(\mathbf{W}^T\mathbf{W}–2\mathbf{X}^T\mathbf{W} + \mathbf{X}^T\mathbf{X}) = (\mathbf{W}–\mathbf{X})^T(\mathbf{W}–\mathbf{X}) = \sum_{k=1}^{m}(w_k - x_k)^2 > 0. \tag{4.38}$$

Given this absurd result, we can conclude that our initial assumption was wrong and therefore, $(\mathbf{Y} - a\mathbf{X} - b\mathbf{W})^T(\mathbf{Y} - a\mathbf{X} - b\mathbf{W}) < (\mathbf{Y} - \mathbf{X})^T(\mathbf{Y} - \mathbf{X})$ what proves that $SSE(a\mathbf{X} + b\mathbf{W}) < SSE(\mathbf{X})$. □

The result proved above only regards the arithmetic mean and do not give any information of how more accurate the combined estimator will be than the worst one.

It is important to comment that the theorem 4.1 regards the known datasets, that is, in practice, it is applied to encourage the realization of the ensemble model in the training and validation tests, since they are composed by known data. The superiority in terms of SSE of the weighted arithmetic mean is only proofed considering the less accurate of the two models to be combined, and nothing can be concluded about the weighted arithmetic mean be more accurate than the best estimate. Ratifying what was commented in the section 4.1.2, the separation of the dataset into training, validation and test set has to be done, considering that the validation set must be representative of the test set, therefore, we can expect, but not guarantee, that the model fitted in the validation set will present a similar perform in the test set. Given the theorem 4.1, considering the validation set, HMMA, surely, will not be worse than the two original estimates, but there is a possibility of it be greater than both, as happened in some cases which will be presented in the section 5.4.3.

5 Results and Analysis

In this section, we present the performance comparison among the following forecasting approaches: Maximum Visibility, Hybrid Mean of Multiples Approaches (HMMA), Mao-Xiao Approach (MXA) (MAO; XIAO, 2019), Auto Regressive Integrated Moving Average (ARIMA), Support Vector Regression (SVR), Long Short Term Memory (LSTM), Multilayer Perceptron (MLP), Hybrid additive ARIMA-ANN (HAAA), Hybrid additive ETS-ANN (HAEA) and naive estimation. To compare the accuracies quantitatively, we calculated the error measures defined in the equations (3.2), (3.3), and (3.4). It is important to point out that all forecasting processes are **one-step-ahead** type.

MXA was chosen as a comparative method because, according to Mao and Xiao (2019), it is considered a successful complex-network-based forecasting method with a good prediction performance. The ARIMA model was chosen because it is widely used as a benchmark in the literature, so if MVA or HMMA are more accurate than ARIMA, then one or both can be taken as benchmarks against other new prediction methods. The methods SVR, LSTM and MLP are also being considered benchmarks given to the wide amount of works who use these methods to provide time series forecasts. Inspired on the results of the M4-competition, presented by Makridakis, Spiliotis and Assimakopoulos (2018), combination methods tend to perform better than the pure one, therefore, we considered important compare the MVA and HMMA forecasting results to other succeeded hybrid methods. We choose the Hybrid additive ARIMA-ANN (HAAA) and Hybrid additive ETS-ANN (HAEA). Lastly, we also considered the naive estimation because, despite its simplicity, for some kinds of time series, it has beaten ARIMA and even some hybrid models in terms of performance.

Before starting each forecasting process, we characterized the respective time series. First, by running the ADF and/or KPSS tests, we determine if each time series is level, trend or non-stationary. Next, we tested seasonal behavior using the Webel-Ollech test. Finally, through the Ljung-Box test, we check if there is any dependence between the pairs of the time series observations.

Here, the only pre-processing job was: (a) to delete or properly replace all "NA", non-available, data; and (b) normalizing the data according to de following rule:

$$z_i = \frac{y_i - \min\{\mathbf{Y}\}}{\max\{\mathbf{Y}\} - \min\{\mathbf{Y}\}}, \tag{5.1}$$

where $\mathbf{Y}$ is the original dataset. Actually, the normalization process, or any other is not necessary. We did that with the purpose of having comparable results, free of any scale. In the case of ARIMA, we used the R function, *auto.arima*, which automatically performs all necessary preprocessing to obtain the best ARIMA model. The size of the window used in each forecasting process was chosen considering the results of the autocorrelation function, so only the most significant terms, statistically speaking, were taken into account. It is important

to emphasize that the mathematical derivations to obtain the best K and J to be considered on MVA is beyond the scope of this work. Hence for each forecasting simulation, we found the pair (J^*, K^*) using simulation according to explained in section 4.1.2. In all cases, to provide the HMMA estimators, the values of β were calculated according to was described in the section 4.2.5.

In order to compare the forecasting performance achieved from MVA, HMMA, Dynamic ARIMA, MXA, MLP, LSTM, SVR, HAAA, HAEA and naive estimation we chose the following time series: three classic time series from different fields, provided by the R *datasets* package; a Brazilian econometric time series composed of daily closing prices of the Bovespa index; and one recursive time series that was artificially generated. It was generated two sets of results: the first done based on the original data, and the second based on the normalized data according described in the equation 5.1.

It is important to point out that the accuracies obtained by each method, considering each time series, were taken from the same test set. We are emphasizing that this measure was taken to ensure that comparisons between methods were made fairly.

5.1 Characterization of the five time series

This section is dedicated to showing the particularities of each time series used in this research. The appropriate hypothesis tests, presented in the section 2.1.10, were chosen to characterize the series as stationary, seasonal or independent. In all tables presented from this point of the work, the acronym WO regards the Webel-Ollech test, for characterizing seasonal time series.

5.1.1 International Airline Passengers Time Series

Consider the monthly number of passengers for international airlines, in thousands of people, from 1949 to 1960, illustrated in the appendix D.1. This a 144-element time series, which in the R *astsa* package is called *AirPassengers*, but which is also presented by Box and Jenkins (1976).

As one can observe in the autocorrelation function presented in the Figure 4.1, a lag of up to approximately 130 can cause any significant impact on the last observation. However, if we take 130 as a window, many other non-significant previous observations may introduce noise to the forecasting process; hence, we choose window 40. We applied this dataset to the statistical tests presented in section 2.1.10. The Table 5.1 shows the characteristics of this dataset.

Table 5.1: Characteristics of the airline passengers time series based on statistical tests

Statistical Test	p-value	Result
ADF	< 0.01	TS
KPSS	> 0.1	TS
Ljung-Box	< 0.01	NI
WO/Period		Seasonal - 12

Legend: NS - Not Stationary, TS - Trend Stationary, LS - Level Stationary, I - Independent, NI - Not Independent.

Considering the forecasting process with the normalized data, MVA had $K = 0.89$, $J = 0.01$ and the nonlinear term, expressed by equation (4.6), was included in the model. The comparison based on the original data was done with $K = 0.886$, $J = 6$. The first 40 observations were used in the first window, so the rest of the 104 observations, that is, y_{41} to y_{144} were divided into training and test set. The training and validation sets were merged into a single set of 70% of these remaining data, that is, size 77. Naturally, the test set is composed by 31 observations, that is, the same values of K and J were used to calculate estimates for the 31 last observations following the sliding window process described in the section 3.3.

5.1.2 Lynx Trappings Time Series

In agreement with Campbell and Walker (1977), 'lynx' is a time series of annual numbers of lynx trapped in Canada from 1821 to 1934. There are 114 observations in this time series, see appendix D.2. The Table 5.2 presents the characteristics of this time series based on the results of the proper statistical tests.

Table 5.2: Characteristics of the lynx trappings time series based on statistical tests

Statistical Test	p-value	Result
ADF	< 0.01	TS
KPSS	> 0.1	TS
Ljung-Box	< 0.01	NI
WO/Period		Seasonal - 19

Legend: NS - Not Stationary, TS - Trend Stationary, LS - Level Stationary, I - Independent, NI - Not Independent.

The ACF indicates that the first 77 past terms may contain significant information with respect to the current element, see appendix D.3, therefore we chose $w = 77$. Additionally, the

most significant values of autocorrelation between terms occur from time to time, suggesting that the series has a seasonal behavior, which was confirmed through the application of the WO test.

Considering the forecasting process with the normalized data, MVA had $K = 0.6$, $J = 2$ and the nonlinear term, expressed by equation (4.6), was included in the model. The comparison based on the original data was done with $K = 0.89$, $J = 0.01$. The first 77 observations composed the first window so the last 37 elements were divided into training and test sets. We toke 70% of these remaining observations to compose the training set, which gives a set with 26 elements. Hence, the test set is composed by the last 11 observations.

5.1.3 Financial Time Series

Ibovespa is the most important performance indicator of stocks traded on the Brazilian financial market (B3, 2020). It was created in 1968 and has since served as a reference for investors worldwide (B3, 2020). The third time series is made up of daily closing prices of Ibovespa from January 1, 2019 to July 28, 2020, see appendix D.4. The 389 data used in this simulation were taken from the Yahoo Finance website and, according to can be observed in the appendix D.5, the ACF indicates window of length 200.

Table 5.3: Characteristics of the IBOVESPA time series based on the statistical tests.

Statistical Test	**p-value**	**Result**
ADF	0.45	NS
KPSS	< 0.01	NS
Ljung-Box	< 0.01	NI
WO/Period		Seasonal - 7

Legend: NS - Not Stationary, TS - Trend Stationary, LS - Level Stationary, I - Independent, NI - Not Independent.

According to the proper statistical tests, this is a non-stationary dataset, with no trend, seasonal and with dependency among its observations, see Table 5.3. In this case, we calibrated MVA with $K = 0.75$, $J = 0.003$ for the original data and $K = 0.844$, $J = 150$ for the normalized data. In both cases, the nonlinear term has been considered in the model.

The first 200 observations were used in the first window, so the rest of the 189 observations, were split into training and test set. The training and validation sets were merged into a single set of 50% of these remaining data, that is, size 95. Therefore, the test set is composed by the 94 remaining observations.

5.1.4 New Haven Annual Temperature Time Series

The fourth dataset is the 60-element time series named "*nhtem*" on the R *datasets* package which regards the mean annual temperature, in degrees Fahrenheit, in New Haven, Connecticut, from 1912 to 1971 (TEAM; WORLDWIDE, 2021). In this forecasting process, we calibrated MVA with $K = 1$, $J = 100$ for the original data and $K = 1.087$, $J = 500$ for the normalized data. In both cases, the nonlinear term has been considered in the model.

The first 10 observations were used in the first window, so the rest of the 50 observations, that is, y_{11} to y_{60} were split into training and test set. The training and validation sets were merged into a single set of 80% of these remaining data, that is, size 40. Naturally, the test set is composed by 10 observations, that is, the same values of K and J were used to calculate estimates for the 10 last observations following the sliding window process.

5.1.5 Recursive Time Series

The last dataset comprises a recurrence-based time series with 100 observations. Each observation is given by the following law:

$$y_t = 0.54y_{t-1} + 0.17y_{t-2} + 0.3y_{t-3} + 0.4w_t, \tag{5.2}$$

where w_t is a random variable normally distributed with zero mean and variance 1. Additionally, $y_0 = 10, y_1 = 9.5, y_2 = 9$ and $t \in \{0, 1, 2, \cdots, 99\}$.

In this forecasting process, we calibrated MVA with $K = 2.5$, $J = 0.9$ for the original data and $K = 1.9$, $J = 100$ for the normalized data. In both cases, the nonlinear term has been considered in the model. The first 30 observations were used in the first window, so the rest of the 70 observations, that is, y_{31} to y_{100} were split into training and test set. The training and validation sets were merged into a single set of 75% of these remaining data, that is, size 53. Therefore, the test set is composed by the 17 remaining observations. The same values of K and J were used to calculate estimates for the 17 last observations following the sliding window process described in the section 3.3.

5.1.6 Summary of the characteristics of the five time series

The Table 5.4 summarizes the characteristics of the five time series presented in this work, showing all their features, confirmed by proper statistical tests.

Table 5.4: The characteristics of the five time series.

Dataset	Level	Trend	Seasonal	Indep
Air Pass	No	Yes	Yes	No
Lynx	No	Yes	Yes	No
Ibovespa	No	No	Yes	No
nhtemp	No	Yes	Yes	No
RBTS	No	No	Yes	No

5.2 Adjusting the Machine Learning Models

One of the most determining steps in creating a suitable forecasting model based on Machine Learning approaches such as ANN and SVR models is the choice of the input variables.

5.2.1 The SVR Models

In this work we started the search for the best SVR hyper-parameters according to was considered in Samsudin, Shabri and Saad (2010): $C = 10$ and $\epsilon = 0.1$. The kernel choice is for the radial basis function (RBF) starts with $\gamma = 0.1$ since, according to Ma and Guo (2014) and Zhou *et al.* (2009), this kernel has good performance in time series forecasting problem, it is able to capture nonlinear characteristics of the time series and presents less numerical problems.

We applied the SVR models in the training set, for multiples combinations of (C, ϵ, γ). The set or parameters which provided the smallest MAE was applied to the testing set. Some forecasting strategies were tested to perform the SVR. Firstly we compared the 10-fold cross validation technique according to described in Samsudin, Shabri and Saad (2010) with the "leave one out cross validation" (LOOCV) based on described in Vehtari, Gelman and Gabry (2017). The second one gives better results in the training sets. However, the walk-forward validation technique, presented by Kaastra and Boyd (1996), and used in all other methods, gave the smallest MAE considering the training sets. Each window of size w, let say, $y_1, \cdots, y_w$, works as the input set $\mathbf{X} \in \mathrm{R}^w$ and the next term, that is y_{w+1} is the output $Y \in \mathrm{R}$. After a model is obtained, we use this fitted model to estimate y_{w+2} using the window $y_2, \cdots, y_{w+1}$. This procedure is continued until finding the estimate for the last element of the time series. At the end of the process, considering a $m-$ element test set, we have a vector of estimates, $\hat{y}_{n-m+1}, \cdots, \hat{y}_n$ and the equivalent set of real elements and based on these two sets, all error measures MAE, MAPE and RMSE are evaluated.

The Tables 5.5 and 5.6 shows the parameters considered in the SVR models considering the original data and the normalized one, respectively.

Table 5.5: Hyperparameters considered in each SVR model applied in the original data.

Time Series	**C**	γ	ϵ
Air Pass	30	38	0.1
Lynx	3	140	0.1
Ibovespa	15	40	0.1
nhtemp	1	20	0.1
RBTS	2	0.05	0.1

Table 5.6: Hyperparameters considered in each SVR model applied in the normalized data.

Time Series	**C**	γ	ϵ
Air Pass	30	38	0.1
Lynx	3	107	0.1
Ibovespa	15	40	0.1
nhtemp	1	20	0.1
RBTS	3	1	0.1

5.2.2 The LSTM Models

An LSTM model is an recurrent network model with memory, that is, this model can remember information from earlier points in the network (KAUSHIK *et al.*, 2020). The architecture of an LSTM consists of units called memory cells, self-connections and gates which are special multiplicative units (HOCHREITER; SCHMIDHUBER, 1997). According to Kaushik *et al.* (2020) the connections keeps in memory the temporal state of the memory cells and the gates control the flow of information.

We considered a LSTM model according to proposed by Kaushik *et al.* (2020). The hyperparameters used to reach the best accuracy in the validation set, for each time series, are described in the Table 5.7. The column called 'hl' gives the number of neurons in the hidden layer.

Table 5.7: Hyperparameters considered in each LSTM model.

Time Series	epochs	hl	lrate
Air Pass	1000	512	0.1
Lynx	200	240	0.1
Ibovespa	500	128	0.1
nhtemp	200	64	0.1
RBTS	40	600	0.1

5.2.3 The MLP Models

We considered a MLP structure composed by three layers of nodes: an input layer, a hidden layer and an output layer. All neurons were processed by the sigmoidal activation function, except for the output layer for which a linear activation function was used. The forecasting tasks were produced by the *mlp* function included in the *nnfor* R package. For each time series, the testing set is the same considered to all other forecasting methods and the forecasts were combined using the median operator. The Table 5.8 shows the hyperparameters which provide the best accuracy in the training and validation sets.

Table 5.8: Hyperparameters considered in each MLP model.

Time Series	epochs	h=nodes	lags
Air Pass	15	5	15
Lynx	10	10	29
Ibovespa	500	128	0.1
nhtemp	18	5	5
RBTS	25	8	50

To enable reproducibility, we set the seed at 2, so all randomly generated initial weights are always be the same.

5.3 Adjusting the Hybrid Models

The last two time series forecasting methods are hybrid approaches. These are additive combinations of ANN with ARIMA and ANN with ETS. We consider the number of nodes in the ANN model's input layer to be equal to the window size considered for all other forecasting methods. We tried other values of w, but in all cases, the obtained accuracy, relative to the validation sets, only got worse.

According to Samsudin, Shabri and Saad (2010) and Khashei and Bijari (2011), a three-layer feed-forward ANN model is commonly used for time series forecasting. We normalized the training and testing data to fit in the interval $[0, 1]$ according to equation (5.1). The hyperbolic tangent function was used as the transfer function between the input and the hidden layers because it resulted in the best accuracy. We chose the linear function as the transfer function between the hidden and the output layers due to its robustness for continuous output variables (SAMSUDIN; SHABRI; SAAD, 2010). The optimal number of neurons in the hidden layer, h, was obtained by trial and error, but initially we used previous practical results (KANG, 1992): $h = 0.5i$, where i is the number of input data. The initial model weights were chosen considering a uniform distribution in the interval $[-0.1, 0.1]$, with seed 2. Each training assignment was performed minimizing the mean absolute error.

5.3.1 The ARIMA-ANN Models

The hybrid additive approach combining ARIMA and ANN was implemented according to described in Chindanur and B (2015) using the Khashei-Bijari approach (KHASHEI; BIJARI, 2011). The Tables 5.9 and 5.10 show the best number of nodes in the hidden layer for each time series considering the normalized and original data, respectively. These parameters were defined from the accuracy obtained on the training and validation sets.

Table 5.9: Parameters considered in each ANN-ARIMA model applied in the normalized data.

Time Series	Number of hidden nodes
Air Pass	5
Lynx	3
Ibovespa	5
nhtemp	10
RBTS	20

Table 5.10: Parameters considered in each ANN-ARIMA model applied in the original data.

Time Series	Number of hidden nodes
Air Pass	1
Lynx	1
Ibovespa	1
nhtemp	3
RBTS	3

5.3.2 The ETS-ANN Models

The ETS models are composed by a family of time series models with a basal state space model containing an error term (E) and a level, trend (T) or seasonal (S) components. The hybrid additive approach combining ETS and ANN was implemented according to described in Purohit *et al.* (2021). The ETS-models were defined based on the R function "ets" from the package forecast. This function is able to calculate the best ETS model which fits with the given time series. The Tables 5.11 and 5.12 show the best number of nodes in the hidden layer for each time series considering the normalized and original data, respectively. These parameters were defined from the accuracy obtained on the training and validation sets.

Table 5.11: Parameters considered in each ANN-ETS model.

Time Series	Number of hidden nodes
Air Pass	5
Lynx	20
Ibovespa	20
nhtemp	14
RBTS	42

Table 5.12: Parameters considered in each ANN-ETS model applied in the original data.

Time Series	Number of hidden nodes
Air Pass	4
Lynx	3
Ibovespa	6
nhtemp	6
RBTS	9

5.4 Comparing the error results

In the last three sections we described the five datasets used in this work and we presented the eight forecasting techniques and their respective set of parameters considered in each forecasting process. Firstly, we will present the error comparisons among MVA and each comparative method considering the normalized and the original data. Next, HMMA results are presented.

5.4.1 Errors obtained from MVA and the comparative methods considering the normalized data

In this section we show the comparisons among the results obtained by MVA and the eight comparative methods in terms of MAE, MAPE and RMSE, according to defined in the equations (3.2), (3.3) and (3.4), respectively. All accuracy results presented in this section were obtained using the normalized data. In order to verify whether the residuals produced by each method are stationary, we applied the KPSS and ADF tests, according to described in the sections 2.1.10.2 and 2.1.10.1, respectively. Actually, according to Kwiatkowski *et al.* (1992), the KPSS test tends to reject the null hypothesis too often, so, to overcome this issue, if the null hypothesis is rejected, then we run the ADF test to confirm the time series non-stationarity behavior. This step is important because the Diebold-Mariano test is used to compare the statistical superiority between MVA and the comparative methods and the condition for using this hypothesis test is whether the residuals are stationary. The results of accuracy, presented in the Table 5.13, were obtained considering the test set, using the parameters adjusted in the training and validation sets.

Table 5.13: Comparative performance indicators obtained from the forecasting process of the five normalized time series considering MVA and the eight comparative methods.

Time Series: AirPassengers									
	MVA	MXA	DA	Naive	HAAA	HAEA	LSTM	MLP	SVR
MAE	0.0711	0.0894	0.0897	0.0887	0.0923	0.0972	0.0800	0.0995	0.0791
MAPE	10.89	14.26	13.74	13.93	14.24	15.35	26.52	14.98	12.39
RMSE	0.0833	0.1090	0.1095	0.1030	0.1071	0.1144	0.0941	0.1151	0.0927
Time Series: lynx									
	MVA	MXA	DA	Naive	HAAA	HAEA	LSTM	MLP	SVR
MAE	0.0700	0.1393	0.0877	0.1104	0.1115	0.0725	0.1053	0.1267	0.0828
MAPE	38.81	64.92	32.88	52.91	54.18	43.07	155.94	60.78	39.48
RMSE	0.1045	0.1514	0.1125	0.1268	0.1241	0.0970	0.1226	0.1525	0.0964
Time Series: IBOV									
	MVA	MXA	DA	Naive	HAAA	HAEA	LSTM	MLP	SVR
MAE	0.0272	0.0308	0.0470	0.0276	0.0287	0.0429	0.0815	0.0361	0.0386
MAPE	23.53	24.71	45.59	23.85	22.97	42.63	146.3	25.13	27.52
RMSE	0.0372	0.0448	0.0881	0.0379	0.0382	0.0643	0.1362	0.0525	0.0485
Time Series: nhtemp									
	MVA	MXA	DA	Naive	HAAA	HAEA	LSTM	MLP	SVR
MAE	0.0906	0.1139	0.1094	0.0970	0.1131	0.1125	0.1105	0.1222	0.0982
MAPE	17.16	22.74	21.53	18.98	20.52	21.53	16.89	24.49	19.29
RMSE	0.1060	0.1392	0.1324	0.1174	0.1406	0.1309	0.1290	0.1543	0.1233
Time Series: RBTS									
	MVA	MXA	DA	Naive	HAAA	HAEA	LSTM	MLP	SVR
MAE	0.0227	0.0322	0.0308	0.0279	0.0346	0.0462	0.0298	0.0285	0.0289
MAPE	2.38	3.38	3.26	2.93	3.59	4.83	2.82	2.98	3.01
RMSE	0.0286	0.0378	0.0367	0.0323	0.0439	0.0555	0.0326	0.0364	0.0345

According to the contents for the Air Passengers dataset presented in the Table 5.13, MVA outperformed all other methods based on the MAE, MAPE and RMSE values. The KPSS test showed that all methods, except the LSTM model, provided stationary residuals. Applying the ADF test we confirmed the non-stationary behavior of the LSTM residuals. For the Air Passengers time series, MVA had its superiority confirmed by the Diebold-Mariano(DM) test, see Table 5.14, when compared to all methods except for LSTM and SVR for which the quality of the forecasts are statistically equivalent. Given that the LSTM residuals are non-stationary, the statistical comparison using the DM test is not valid. In the Figure 5.1 we can see the

forecasts for the Air Passengers time series considering MVA, MXA, LSTM and SVR. Since the DM test did not find any method to be superior to MVA and considering that its errors are the smallest, we can conclude that MVA is the most accurate method, among all those who participated in this study, to make predictions for this time series.

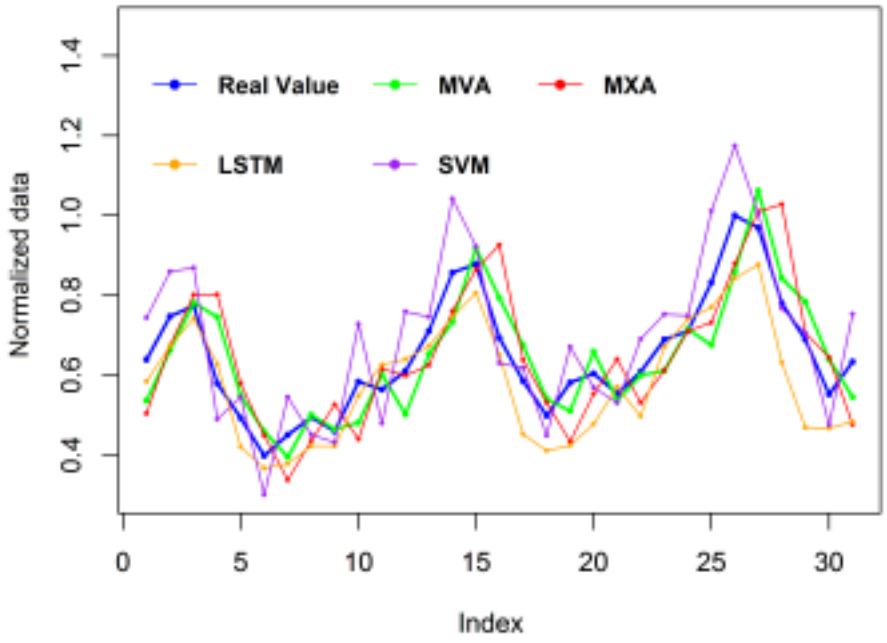


Figure 5.1: Forecasts for Air Passengers time series considering MVA, MXA, LSTM and SVR. Source: Drawn by the author.

With respect to the results obtained for the Lynx time series, the KPSS test was applied to the residuals produced by all forecasting models and the stationarity was confirmed in all cases. Considering the results of Tables 5.13 and 5.14 for the Lynx time series, MXA, naive, Additive-ARIMA-ANN, LSTM and MLP are less accurate than MVA, quantitative and qualitatively given that their values of MAE, MAPE and RMSE are higher than MVA's, and the DM test indicates MVA's superiority. However, MVA, ARIMA, Additive-ANN-ETS (HAEA) and SVR showed similar results considering error measures and statistical comparison. In order to illustrate the closeness of these methods to the real values in the test set, we have plotted the comparison graph for lynx time series and presented it in the Figure 5.2.

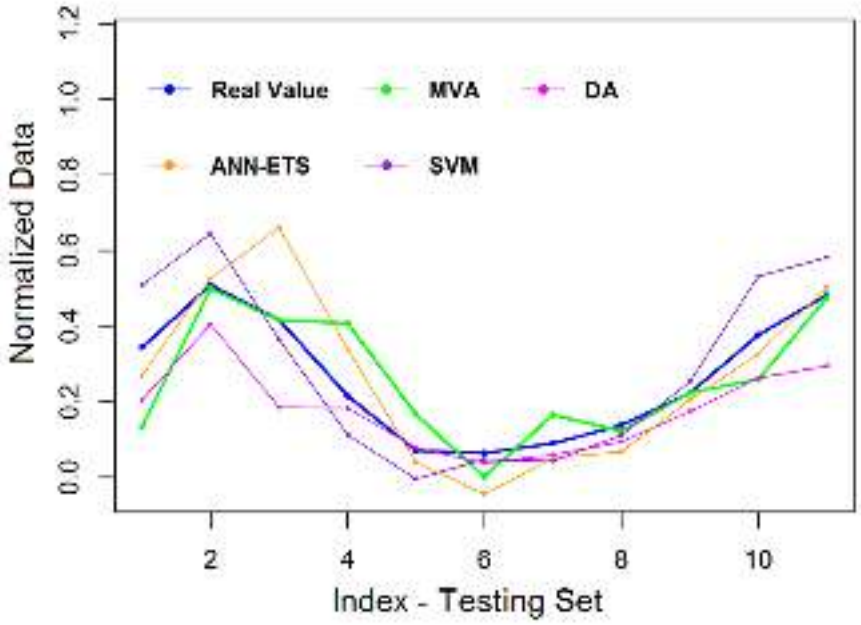


Figure 5.2: Forecasts for lynx time series considering MVA, DA, Additive-ANN-ETS and SVR. Source: Drawn by the author.

Consider the results obtained for the IBOVESPA time series. The KPSS test was applied to the residuals produced by all forecasting models and the stationarity was confirmed in all cases except for the LSTM model which had non-stationary residuals confirmed by the ADF test. Comparing the accuracy results obtained by each method for the IBOVESPA time series, MVA outperformed all other methods. Naive is the only method which provided similar results, statistically speaking, confirmed by the DM test. In the Figure 5.3 we can see the forecasts for the IBOVESPA time series considering MVA, MXA, Additive-ANN-ARIMA and Naive. Definitely, in order to produce forecasts for the IBOVESPA time series, MVA is the most indicated method among all methods considered in this study.

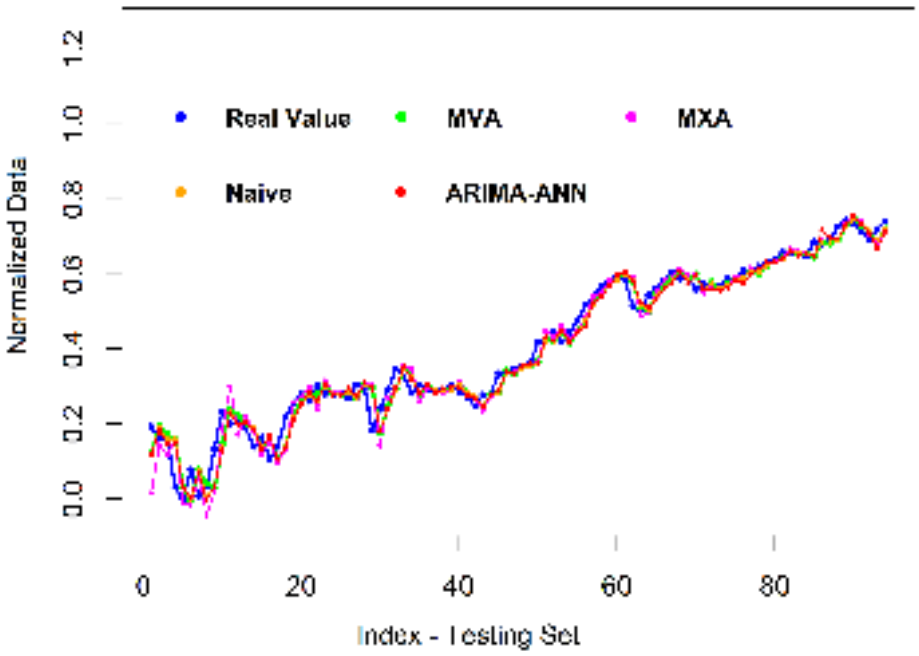


Figure 5.3: Forecasts for IBOVESPA time series considering MVA, MXA, Additive-ANN-ARIMA and Naive. Source: Drawn by the author.

The fourth time series, was chosen precisely because it is a dataset with characteristics that are difficult to define. How would all methods respond in the forecasting process for this time series? It is classified as trend stationary, seasonal and not-independent dataset, however, the p-values obtained from each hypothesis test is very close to the 10% type I error. It was expected that most methods would present statistically similar results (thought confirmed by the results in the Table 5.14), therefore the most accurate method would have to be defined based on the error measures. According to the results presented in the Table 5.13, MVA outperforms all other methods in terms of MAE, MAPE and RMSE, which indicates that the variance of the magnitudes of the errors obtained by MVA is smaller than that produced by the benchmarks, thus, MVA is considered the most accurate method in the forecasting assignment for this time series.

Lastly, we present the results for a recurrence based time series artificially created. This dataset was created with the intention of favoring the Dynamic ARIMA method, since the sequence is based on an AR(3) model. Dynamic ARIMA was expected to achieve the best results, however, MVA had the best accuracy. The conclusion is similar to that obtained in the case of the air passengers time series, since MVA obtained the smallest errors, but tied with other methods in the statistical comparison of the quality of the estimation. None of the considered methods were classified by the DM test as superior to MVA.

We consider important to emphasize that all MVA forecasts, presented in this work, were obtained based on using the Dice similarity. We considered using other methods, such as

Table 5.14: P-values obtained from the application of the Diebold-Mariano test in the forecasting residuals of the five datasets

Time Series: AirPassengers (Tipe I error = 10%)								
	MXA	ARIMA	Naive	HAAA	HAEA	LSTM	MLP	SVR
MVA	0.0470	0.0170	0.0076	0.0055	0.0130	0.2033	0.0012	0.1483
Time Series: Lynx (Tipe I error = 10%)								
	MXA	ARIMA	Naive	HAAA	HAEA	LSTM	MLP	SVR
MVA	0.0114	0.3263	0.0396	0.0491	0.4847	0.0904	0.0875	0.2937
Time Series: IBOV (Tipe I error = 10%)								
	MXA	ARIMA	Naive	HAAA	HAEA	LSTM	MLP	SVR
MVA	0.0321	< 0.001	0.2587	0.0563	< 0.001	< 0.001	0.0012	< 0.001
Time Series: nhtemp (Tipe I error = 10%)								
	MXA	ARIMA	Naive	HAAA	HAEA	LSTM	MLP	SVR
MVA	0.1558	0.1637	0.3462	0.1830	0.1237	< 0.001	0.0560	0.3486
Recurrence-based Time Series (Tipe I error = 10%)								
	MXA	ARIMA	Naive	HAAA	HAEA	LSTM	MLP	SVR
MVA	0.0235	0.1070	0.0409	0.1263	0.0055	0.1594	0.1324	0.1854

Jaccard (ARNABOLDI *et al.*, 2015) and the Inverse Log-weighted (ADAMIC; ADAR, 2003), both cited in the section 2.3.3. In order to compare the results, the obtained RMSE, for each time series, was taken into account.

Table 5.15: Comparative MVA's RMSE when calculated from Jaccard, Dice and Inverse Log-Weighted similarities considering the five time series.

	RMSE - MVA		
	Jaccard	**Dice**	**InvLogWeighted**
Air Passengers	0.0804	0.0804	0.0954
Linx	0.1016	0.1016	0.0980
IBOVESPA	0.0372	0.0372	0.0384
NHtemp	0.1060	0.1060	0.1158
RBTS	0.0281	0.0281	0.0297

The Jaccard method led to the same results obtained from the Dice similarity index, considering the five time series presented here. The Inverse Log-weighted similarity produced similar results to those presented here. Better if compared to that obtained from Dice, for the Lynx time series and worse in the other cases. The comparative analysis of the MVA accuracy based on the different measures of similarities is beyond the scope of this work, but we ratify that this is an important topic to be considered since improved versions of MVA can be developed.

5.4.2 Errors obtained from MVA and the comparative methods considering the original data

We created this section in order to evaluate whether there is any difference on the results in the case of applying the original data, without data normalization. In the last section it was shown that MLP and LSTM did not presented accuracy comparable with MVA in the five simulations. So, in the next set of results we considered comparing MVA with MXA, DA, Naive, HAAA, HAEA and SVR only.

Table 5.16: Comparative performance indicators obtained from the forecasting process of the five original time series considering MVA and six comparative methods.

Time Series: AirPassengers							
	MVA	MXA	DA	Naive	HAAA	HAEA	SVR
MAE	37.097	46.272	45.064	45.967	45.9677	44.708	41.027
MAPE	8.32	10.71	10.08	10.50	10.50	10.17	9.33
RMSE	43.310	56.445	54.890	53.347	53.347	52.551	48.044
Time Series: lynx							
	MVA	MXA	DA	Naive	HAAA	HAEA	SVR
MAE	566.001	969.02	563.600	767.45	767.45	552.91	609.86
MAPE	49.27	62.33	32.29	50.66	50.66	34.71	43.46
RMSE	639.68	1052.61	674.15	881.79	881.81	724.97	717.78
Time Series: IBOV							
	MVA	MXA	DA	Naive	HAAA	HAEA	SVR
MAE	1531.75	1742.64	1840.01	1561.59	1548.83	1821.91	2190.18
MAPE	1.95	2.24	2.38	1.99	1.97	2.32	2.74
RMSE	2130.36	2529.58	2687.946	2136.09	2126.87	2678.081	2752.38
Time Series: nhtemp							
	MVA	MXA	DA	Naive	HAAA	HAEA	SVR
MAE	0.6075	0.7635	0.6812	0.6500	0.6500	0.6232	0.6541
MAPE	1.17	1.48	1.32	1.26	1.26	1.20	1.27
RMSE	0.7107	0.9332	0.8588	0.7867	0.7867	0.7481	0.8187
Time Series: RBTS							
	MVA	MXA	DA	Naive	HAAA	HAEA	SVR
MAE	0.2872	0.3853	0.3847	0.3352	0.3737	0.4101	0.3243
MAPE	1.40	1.88	1.89	1.64	1.82	2.01	1.59
RMSE	0.3477	0.4537	0.4651	0.3878	0.4313	0.5251	0.3971

One can observe in the Table 5.16 that the results are similar to those obtained considering the normalized data. As well as what is observed for the normalized data, in the Air Passengers time series, MVA outperformed all other methods based on the MAE, MAPE and RMSE values. Considering the lynx time series MVA, HAEA, DA and SVR presented similar results and better than the other comparative methods. This same results can be observed based on the analysis regarding the normalized data. Regarding the IBOVESPA time series, MVA, Naive and HAAA presented the best accuracy considering both types of data: the normalized and the original data. Finally, considering the two last time series, MVA outperformed all other comparative methods

based on the results of MAE, MAPE and RMSE, and this results can be observed considering the normalized and the original data.

5.4.3 Errors obtained from HMMA and the comparative methods considering the normalized data

In this section, we present the errors comparisons between the Hybrid Means of Multiple Approaches (HMMA) and each respective pair of approaches in its standalone results. We repeated the simulation for several values of β in order to define the type of mean will compose the HMMA estimators. Once defined the type of mean, we proceeded as defined in the section 4.2.5 to define whether way should be used to calculate the best β.

As discussed in the immediately preceding section, considering the Air Passengers time series, the SVR model has an accuracy statistically similar to MVA but with larger errors. Therefore, we chose the SVR model to be combined to MVA in order to produce the HMMA estimator for this time series. For the Air Passengers time series, the training set revealed that the fixed β, found based on a search using the training set, gives the smallest SSE. The search gave $\beta = 0.778225$ and, for this time series, HMMA is composed by the harmonic mean. The Table 5.17 presents the accuracies obtained from HMMA, MVA and SVR for the Air Passengers time series, considering the testing set.

Table 5.17: Comparative performance indicators obtained from HMMA, MVA and SVR for the Air Passengers time series.

Time Series: Air Passengers			
	MVA	SVR	HMMA
MAE	0.0700	0.0791	0.0690
RMSE	0.0833	0.0927	0.0815

The estimation of the parameter β is designed to give the smallest SSE. According to the Table 5.17 the hybrid formulation outperformed both methods in terms of MAE and RMSE. Statistically, HMMA is more accurate than SVR but has similar accuracy is compared to MVA. These findings are an illustration of the result expressed in the theorem 4.1. Figure 5.4 shows the comparison among the forecasts for Air Passengers time series considering MVA, SVR and HMMA applied to the test set.

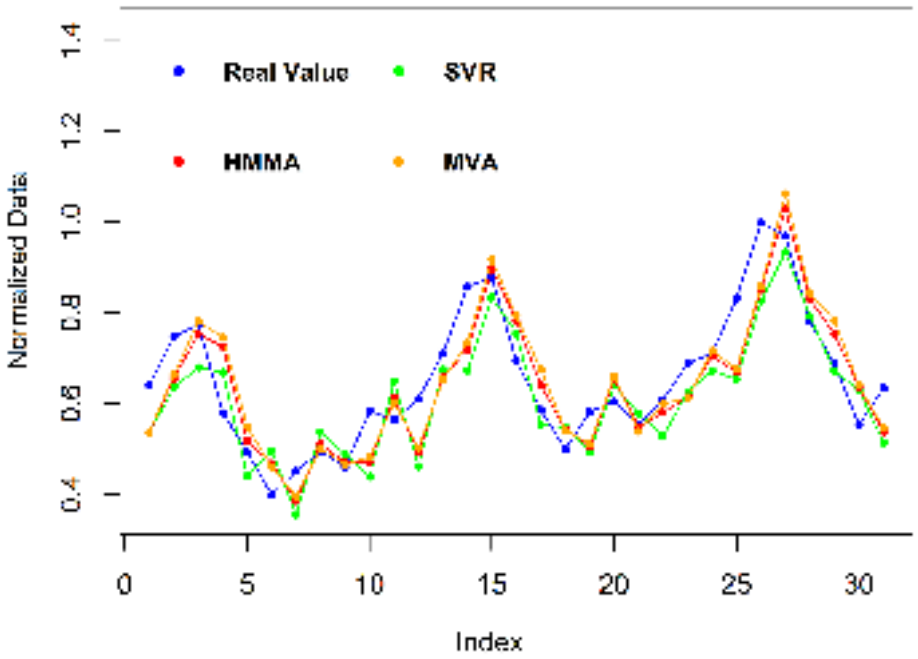


Figure 5.4: Forecasts for the Air Passengers time series considering MVA, SVR and HMMA. Source: Drawn by the author.

Considering the Lynx time series, the Hybrid-Additive-ANN-ETS (HAEA) obtained the best accuracy results whether compared to the other comparative methods, see Table 5.13, so we chose the HAEA estimators to combine with MVA and produce HMMA. After running the simulation for several values of β the arithmetic mean was chosen to provide HMMA estimators. Given the accuracies achieved from the training set, the value of $\beta = 0.7857$, used in the forecasting process, was the same: such from the search among several values of β, as from that obtained using the equation 4.28. This were already expected since the arithmetic mean was used in this case and the equation 4.28 was derived using the arithmetic mean. The Table 5.18 presents the accuracies obtained from HMMA, MVA and HAEA for the Lynx time series, considering the testing set.

Table 5.18: Comparative performance indicators obtained from HMMA, MVA and HAEA for the Lynx time series.

Time Series: Lynx			
	MVA	HAEA	HMMA
MAE	0.0714	0.0725	0.0672
RMSE	0.1016	0.1144	0.0930

It can be observed from the Table 5.18 that HMMA is more accurate than MVA and HAEA. This is one of those cases where the ensemble achieves accuracy superior to its

standalone methods because the harmonic mean could produce values closer to the real values than the original estimators. The Figure 5.5 shows the comparison among the forecasts for the Lynx time series considering MVA, HAEA and HMMA applied to the test set.

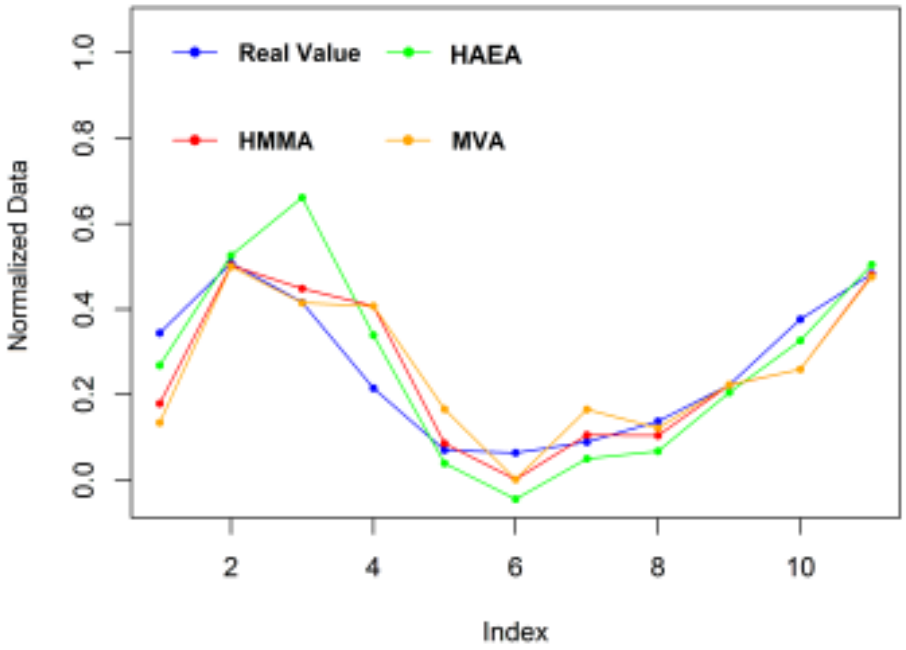


Figure 5.5: Forecasts for the Lynx time series considering MVA, HAEA and HMMA. Source: Drawn by the author.

The third analysis is regarding the IBOVESPA time series. Based on the results shown in the Table 5.13, the naive estimation obtained the best accuracy results whether compared to the other comparative methods, so its estimators were used to combine with MVA and produce HMMA. After running the simulation for several values of β the arithmetic mean was chosen to provide HMMA estimators. Given the accuracies achieved from the training set, the fixed value of β had to be used in the forecasting process, see equation 4.28. Surprisingly, the equation 4.28 gave $\beta = 1.946712$, which is greater than 1. Indeed, in the proof of the equation 4.28, presented in the appendix C.4, there is no impediment for the β value to be greater than 1, however, the theorem 4.1 is not valid in this case. Given this situation, we considered combining MVA to the second best accurate comparative method, which is the HAAA. In this case, the simulation, looking to the training set, also indicated that a fixed $\beta = 0.596153$ provided the smallest SSE, thus given the theorem 4.1, a RMSE smaller than HAAA's is expected. The Table 5.19 presents the accuracies obtained from HMMA, MVA, HAAA and naive for the IBOVESPA time series, considering the testing set.

Table 5.19: Comparative performance indicators obtained from HMMA, MVA, HAAA and Naive for the IBOVESPA time series.

	Time Series: IBOVESPA				
	MVA	Naive	HMMA (Naive)	HAAA	HMMA (HAAA)
MAE	0.0272	0.0276	0.0277	0.0287	0.0278
RMSE	0.0372	0.0379	0.0375	0.0382	0.0377

Statistically, all estimators are similar, given the results of the DM test. As can be observed, when MVA is combined to the naive estimation, the HMMA accuracy is worse than both original estimators and it is likely once the theorem 4.1 is not valid. However, when HMMA is produced by the combination of MVA with HAAA, the best β is smaller than one, so the the theorem 4.1 is valid and, as expected, HMMA quantitative accuracy is at least greater than HAAA's. For this time series, it is worth always simulate MVA and HMMA in order to produce forecasts because by including one or two observations, the best predictor, considering both techniques, can change. The Figure 5.6 shows the closeness among the forecasts for the IBOVESPA time series produced by MVA, Naive and HMMA (HAAA) applied to the test set.

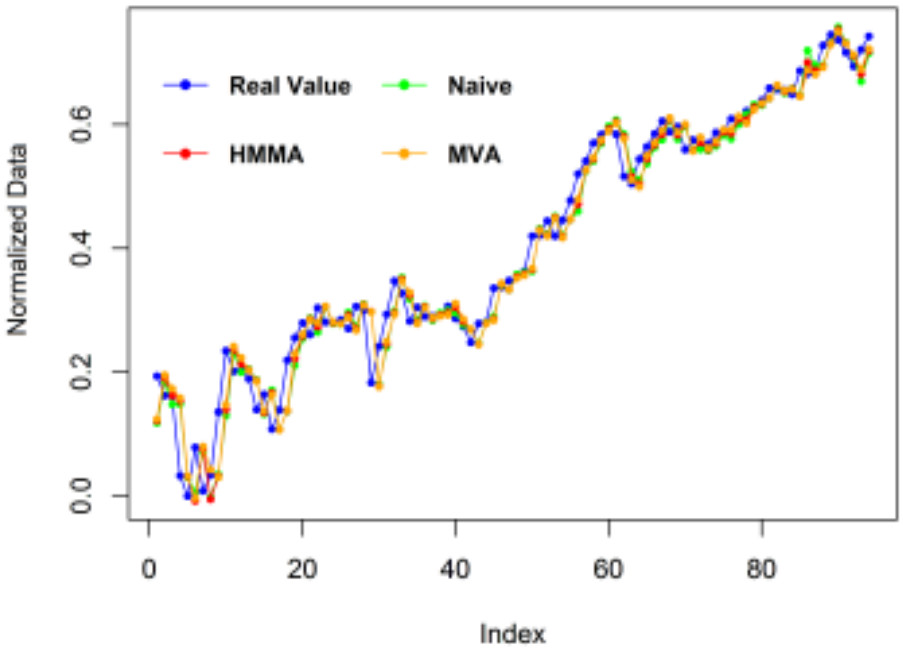


Figure 5.6: Forecasts for the IBOVESPA time series considering MVA, Naive and HMMA. Source: Drawn by the author.

Consider the Nhtemp time series, defined in the section 2.1.3. Based on the results shown in the Table 5.13, the naive estimation obtained the best accuracy results whether compared to the other comparative methods, so its estimators were used to combine with MVA and

produce HMMA. After running the simulation for several values of β the quadratic mean was chosen to provide HMMA estimators. Given the accuracies achieved based on the training set, the fixed value of $\beta = 0.819561$ had to be used in the forecasting process, see equation 4.28. Since $\beta \in [0, 1]$, according to the theorem 4.1, we expect that HMMA's RMSE will be at least better than Naive's. The Table 5.20 presents the accuracies obtained from HMMA, MVA and naive for the Nhtemp time series, considering the test set.

Table 5.20: Comparative performance indicators obtained from HMMA, MVA and Naive for the Nhtemp time series.

Time Series: Nhtemp			
	MVA	Naive	HMMA
MAE	0.0906	0.0970	0.0897
RMSE	0.1060	0.1174	0.1063

Statistically, all estimators are similar, given the results of the DM test. Indeed, HMMA's RMSE is smaller than Naive's, since all conditions to the theorem 4.1 are satisfied. Despite HMMA presents the smallest MAE, we can not state that this method is, quantitatively, the most accurate for this time series, once the HMMA's RMSE is practically the same as that found for MVA, hence, the variance of the magnitudes of the errors obtained by the two methods are statistically and quantitatively equal.

Considering the recurrence-based time series, the SVR obtained the best accuracy results whether compared to the other comparative methods, see Table 5.13, so we chose the SVR estimators to combine with MVA and produce HMMA. After running the simulation for several values of β, considering the training set, the quadratic mean was chosen to provide HMMA estimators. Given the accuracies achieved based on the training set, the fixed value of $\beta = 0.88342$ had to be used in the forecasting process, see equation 4.28. Since $\beta \in [0, 1]$, according to the theorem 4.1, we expect that HMMA's RMSE will be at least better than SVR's. The Table 5.21 presents the accuracies obtained from HMMA, MVA and SVR for the Recurrence-Based time series, considering the testing set.

Table 5.21: Comparative performance indicators obtained from HMMA, MVA and SVR for the Recurrence-Based time series.

Time Series: Recurrence-Based			
	MVA	SVR	HMMA
MAE	0.0227	0.0289	0.0219
RMSE	0.0286	0.0345	0.0281

Statistically, all estimators are similar, given the results of the DM test. Indeed, HMMA's MAE and RMSE are smaller than SVR's, since all conditions to the theorem 4.1 are satisfied, but surprisingly, HMMA's MAE and RMSE are smaller than the MVA either. Observing this result, we can affirm that, quantitatively, the variance of the magnitudes of the errors obtained by HMMA is smaller than that obtained by both methods which were used to produce HMMA, hence, the combination is the most accurate in order to produce forecasts for this time series.

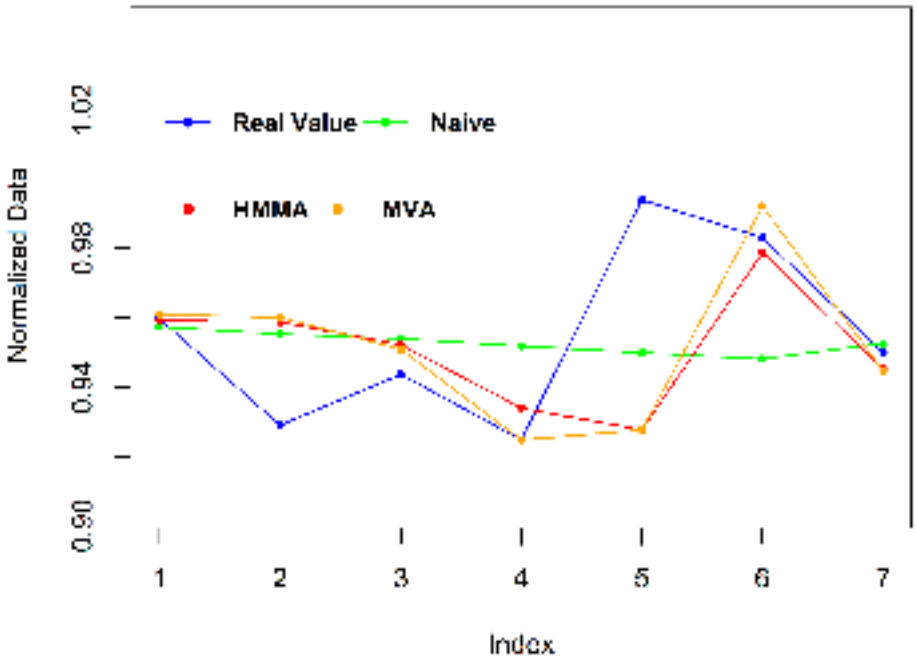


Figure 5.7: Forecasts for the Recurrence-Based time series considering MVA, SVR and HMMA. Source: Drawn by the author.

It is important to point out that results, as obtained from the Air Passengers, Lynx and Recurrence-Based time series, are sufficient to do encouraging, at least, include HMMA among the candidates for providing estimation because given the randomness of the real time series, HMMA can outperform the other methods originally considered in the forecasting assignment.

5.5 Time Complexity of MVA

As we commented that one of the advantages of MVA is its low computational cost, it is important to present a brief discussion about the time complexity of the method. Defining n as the size of the window to be mapped in a graph, consider the following worst-case time complexity analysis for each step of the MVA algorithm.

1. Read the time series data - Complexity: $O(n)$.

2. Normalize the data - Complexity: $O(n)$.

3. Map the time series into a network (visibility graph) - Complexity: $O(n^2)$. See Lan *et al.* (2015).

4. Calculate the Dice similarity - Complexity: $O(dn^2)$, where d is the maximum degree of the vertices in the graph. See Csardi and Nepusz (2006).

5. Use of the function which.max to find the nodes which share the greatest similarity with y_n - Complexity: $O(n)$.

6. Calculate the pre-estimators - Complexity: $O(n)$.

7. Take the maximum among the pre-estimators - Complexity: $O(n)$.

All these steps are performed sequentially, so the complexity of the whole algorithm is given by the maximum among the individual complexities, that is, $O(dn^2)$. Obviously, the maximum value for d is $(n-1)$, so in the worst case scenario, our code is $O(n^3)$. However, this is very unlikely, given the inputs are real time series, so, rarely, is one observation connected to all other observations. In summary, the complexity for MVA depends on the maximum degree of the vertices in the graph, but based on the randomness of the real time series, this maximum degree is relatively low, for example, considering the time series used in this work, the maximum degree were always less than 5% of the time series length. Therefore, MVA can be considered approximately $O(n^2)$ in most applications.

Just for an example, we ran the simulation of the MVA with fixed K and J, considering 95% (also very unlikely) of the data to compose the window. So the algorithm just read the data, normalized it, constructed the complex network, calculated the dice similarities of all pairs of vertices, found the greater value among all values, calculated the pre-estimators considering only the points which presented this greater level of similarity with the current observation, and finally took as the MVA one-step-ahead estimator the maximum value among all pre-estimators. We ran this simulation for an artificial time series AR(1) composed by: 100, 500, 1000, 5000 and 10000 points, which means, respectively, networks with 95, 475, 950, 4750 and 9500 vertices. The execution times were, respectively, 0.007, 0.183, 0.340, 6.827 and 27.927 seconds, which is in consonance with quadratic time complexity. The simulations were performed in a Intel(R) Core(TM) i7-2630QM CPU @ 2.00GHz with 8.00GB (RAM) processor based on x64 under Windows 8.1.

6 Conclusion

In summary, this study addresses one new time series forecasting method, the Maximum Visibility approach (MVA) and ensemble method, the Hybrid Means of Multiple Approaches (HMMA). To calculate the forecast with MVA, we used the visibility graph technique to map a given time series into a complex network, and then we calculated the node similarity matrix. All nodes with the higher similarity index with the node associated to the last time series observation can be chosen to provide the calculation of the $\hat{y}_{MV,n+1}$. We calculated HMMA forecasts by combining the results obtained from MVA and the best among the comparative method, in four different ways: arithmetic mean, geometric mean, harmonic mean and quadratic mean. This strategy was proposed because, through the means inequalities, HMMA produces, at the same time, five one-step-ahead time series estimators, which can increase the probability of reaching the next term of the real time series.

Just in order to compare the performances, we chose the comparative methods ARIMA, the Mao-Xiao approach (MAO; XIAO, 2019), naive estimation, support vector regression (SVR), long-short term memory (LSTM), multilayer perceptron (MLP) and the hybrid additive models ARIMA-ANN and ETS-ANN, where ANN refers to the artificial neural networks and ETS is the error trend seasonal components. The nine methods discussed in this work were applied to five different time series. Each dataset has some or none of the characteristics: seasonal behavior, level or trend stationarity, and dependence between pairs of observations. It is important to note that the time series used in this work also have different time horizons: daily, monthly, annual, and weekly data. The performance results were measured using the mean absolute error (MAE), mean absolute percentage error (MAPE), and root mean squared error (RMSE). Additionally, the Diebold-Mariano test (DIEBOLD; MARIANO, 1995) was considered to evaluate the statistical comparison between accuracy achieved by MVA and each comparative method. All comparisons were calculated considering the normalized data and the original ones.

Considering the normalized data and based on the MAE and RMSE values, MVA outperformed all comparative methods for all time series, except for the Lynx time series, in which the methods ARIMA, additive ARIMA-ETS and SVR reached the same level of accuracy as that obtained by MVA. Looking to the accuracy comparison regarding the original data, one can observe that the results are similar to those one obtained from the normalized data. This only ratify that MVA can be applied to the original data and no data normalization is necessarily needed. This set of results reveals that our proposed method is capable of presenting at least similar accuracy, when compared to other well-established methods in the literature, considering varied applications and datasets with different time horizons. Therefore, our method can contribute not only in the academic field, but also in practical aspects.

One can observe that HMMA also outperformed all comparative methods, considering all five time series and, in the case of the Lynx time series, HMMA presented accuracy higher than both methods used in the ensemble. Not only is HMMA a new forecasting technique,

but also a way to provide other time series forecasting approaches. Since HMMA combines the results of two forecasting methods using the means inequalities, this same strategy can be applied to other well-known accurate methods, in the seek of better accuracy results. The same principles of creating four new one-step-ahead estimators, which can be closer to the real value, stands.

Among the advantages observed with the use of MVA in the forecasting task are the ease of implementation, low computational cost and adaptability to different types of time series, considering characteristics of stationarity, seasonality and dependence between observations. This study showed that the only pre-processing necessary is to delete or properly replace the 'NA' data (not available) if any, no further pre-processing is necessary, thus, the prediction process can be carried out considering the original data. Given the inclusion of the non-linear term, MVA is able to quickly correct the error made in the previous prediction. We can point out as a disadvantage the lower capacity to predict the reversal of the trend of the time series. The greatest errors produced by MVA were mainly at trend reversal points.

We will end this section by making it clear that the results presented here do not mean that MVA and HMMA will be able to outperform all forecasting methods, considering all time series. However, such results certainly indicate that our both techniques are excellent options of approaches to be considered in time series forecasting processes, both for research purposes and for practical use.

6.1 Future Works

Despite the good forecasting performance of HMMA shown in this work, this method can be improved. In future works, one could: investigate of the impact on accuracy obtained by using different time series mapping methods in complex networks; create of a hybrid methodology to combine MVA with MLP, LSTM, ANN or other prediction methods based on machine learning, such as the Elman Recurrent Neural Networks (ERNN), similarly to what was proposed by Aladag, Egrioglu and Kadilar (2009); develop a mathematical derivation for best values of β_A and β_S, presented by equation (4.26), and α by equation (4.1); create MVA and HMMA forecasting accuracy considering *h-step-ahead*, with $h > 1$; study adequate data pre-processing to increase HMMA and MVA forecasting accuracy; provide the comparative analysis of the MVA accuracy based on the different measures of similarities; and develop multidimensional versions of MVA and HMMA, where more than one time series can provide information to achieve better forecasting performance of one or more time series at the same time.

References

ABDULLAH, A.; PILLAI, T. R.; CAI, L. Intrusion detection forecasting using time series for improving cyber defence. *International Journal of Intelligent Systems and Applications in Engineering*, v. 3, 01 2015.

ADAMIC, L.; ADAR, E. Friends and neighbors on the web. *Social Networks*, v. 25, p. 211–230, 07 2003. Disponível em: ⟨http://social.cs.uiuc.edu/class/cs591kgk/friendsadamic.pdf⟩. Acesso em: 27 Ago. 2020.

ADHIKARI, R.; AGRAWAL, R. K. *An Introductory Study on Time Series Modeling and Forecasting*. 2013.

AKAIKE, H. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, v. 19, n. 6, p. 716–723, 1974.

ALADAG, C. H.; EGRIOGLU, E.; KADILAR, C. Forecasting nonlinear time series with a hybrid methodology. *Applied Mathematics Letters*, v. 22, n. 9, p. 1467–1470, 2009. ISSN 0893-9659. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0893965909001475⟩.

ANDREWS, S.; HAMARNEH, G. *Multi-Region Probabilistic Dice Similarity Coefficient using the Aitchison Distance and Bipartite Graph Matching*. 2015. Disponível em: ⟨https://arxiv.org/pdf/1509.07244.pdf⟩. Acesso em: 27 Ago. 2020.

ANGHINONI, L.; ZHAO, L.; ZHENG, Q.; ZHANG, J. Time series trend detection and forecasting using complex network topology analysis. *2018 International Joint Conference on Neural Networks (IJCNN)*, p. 1–7, 2018.

ARNABOLDI, V.; PASSARELLA, A.; CONTI, M.; DUNBAR, R. I. Chapter 5 - evolutionary dynamics in twitter ego networks. In: ARNABOLDI, V.; PASSARELLA, A.; CONTI, M.; DUNBAR, R. I. (Ed.). *Online Social Networks*. Boston: Elsevier, 2015, (Computer Science Reviews and Trends). p. 75 – 92. ISBN 978-0-12-803023-3. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/B9780128030233000059⟩. Acesso em: 27 Ago. 2020.

B3. 2020.

BAGGIO, R.; SAINAGHI, R. Mapping time series into networks as a tool to assess the complex dynamics of tourism systems. 2015.

BAI, Y.-t.; WANG, X.-y.; JIN, X.-b.; ZHAO, Z.-y.; ZHANG, B.-h. A neuron-based kalman filter with nonlinear autoregressive model. *Sensors*, v. 20, n. 1, 2020. Disponível em: ⟨https://www.mdpi.com/1424-8220/20/1/299⟩. Acesso em: 02 abr. 2020.

BENJAMIN, M. A.; RIGBY, R. A.; STASINOPOULOS, D. M. Generalized autoregressive moving average models. *Journal of the American Statistical Association*, Taylor and Francis, v. 98, n. 461, p. 214–223, 2003. Disponível em: ⟨https://doi.org/10.1198/016214503388619238⟩. Acesso em: 01 abr. 2020.

BERTELS, J.; EELBODE, T.; BERMAN, M.; VANDERMEULEN, D.; MAES, F.; BISSCHOPS, R.; BLASCHKO, M. B. Optimizing the dice score and jaccard index for medical image segmentation: Theory and practice. *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*, Springer International Publishing, p. 92–100, 2019. ISSN 1611-3349. Disponível em: ⟨http://dx.doi.org/10.1007/978-3-030-32245-8_11⟩.

BIAZZI, J. Luiz de. Exponential smoothing for intermittent demand with demand basis updated more frequently than seasonality factors. *Gestao e Producao*, v. 26, n. 1, 2019. Disponível em: ⟨https://doi.org/10.1590/0104-530X1297-19⟩.

BIGGS, N. *Algebric Graph Theory*. 2nd. ed. Cambridge University Press, 1993. 7 p. Disponível em: ⟨https://superoles.files.wordpress.com/2015/09/n-biggs-algebraic-graph-theory-1993.pdf⟩. Acesso em: 15 abr. 2020.

BOLLOBAS, B. *Modern Graph Theory*. [S.l.]: Springer, 2002. ISBN 0387984887.

BOX, G.; JENKINS, G.; COAUT, J.; JENKINS, G.; REINSEL, G.; COAUT, R. *Time Series Analysis: Forecasting and Control*. [S.l.]: Prentice Hall, 1994. (Forecasting and Control Series). ISBN 9780130607744.

BOX, G.; JENKINS, G.; REINSEL, C. *Time series analysis: forecasting and control, 4th Edition*. [S.l.: s.n.], 2013. 784 p. ISBN 9781118619193.

BOX, G.; JENKINS, G.; REINSEL, C.; LJUNG, G. M. *Time Series Analysis: Forecasting and Control, 5th Edition*. Hoboken, New Jersey: [s.n.], 2016. ISBN 9781118675021.

BOX, G. E.; JENKINS, G. M.; REINSEL, G. C.; LJUNG, G. M. *Time series analysis: forecasting and control*. [S.l.]: John Wiley & Sons, 2015.

BOX, G. E. P. *Time Series Analysis: Forecasting and Control (Wiley Series in Probability and Statistics)*. Wiley, 2015. ISBN 1118675029. Disponível em: ⟨https://www.xarg.org/ref/a/1118675029/⟩. Acesso em: 31 mar. 2020.

BOX, G. E. P.; JENKINS, G. M. Book; Book/Illustrated. *Time series analysis : forecasting and control*. San Francisco : Holden-Day, 1970. ISBN 0816210942. Disponível em: ⟨http://www.gbv.de/dms/hbz/toc/ht000495926.pdf⟩. Acesso em: 01 abr. 2020.

BOX, G. E. P.; JENKINS, G. M. *Time series analysis : forecasting and control*. third. [S.l.]: San Francisco : Holden-Day, 1976. (G).

BROCKWELL, P. J.; DAVIS, R. A. *Time Series: Theory and Method*. New York: Springer, 1991.

BRONSHTEIN, I.; SEMENDYAYEV, K.; MUSIOL, G.; MUEHLIG, H. *Handbook of Mathematics*. 4th. ed. [S.l.]: Springer, 2004. 346 p.

BROWN, R. *Statistical Forecasting for Inventory Control*. McGraw-Hill, 1959. Disponível em: ⟨https://books.google.com.br/books?id=oKI8AAAAIAAJ⟩.

CAMPBELL, M. J.; WALKER, A. M. A survey of statistical work on the mackenzie river series of annual canadian lynx trappings for the years 1821-1934 and a new analysis. *Journal of the Royal Statistical Society. Series A (General)*, [Royal Statistical Society, Wiley], v. 140, n. 4, p. 411–431, 1977. ISSN 00359238. Disponível em: ⟨http://www.jstor.org/stable/2345277⟩. Acesso em: 15 Ago. 2020.

CARMONA-BENíTEZ, R. B.; NIETO, M. R. Sarima damp trend grey forecasting model for airline industry. *Journal of Air Transport Management*, v. 82, p. 101736, 2020. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S0969699719301711⟩. Acesso em: 03 abr. 2020.

CHANG, C.-C.; LIN, C.-J. Libsvm: A library for support vector machines. Association for Computing Machinery, v. 2, n. 3, 2011. ISSN 2157-6904. Disponível em: ⟨https://doi.org/10.1145/1961189.1961199⟩.

CHEN, S.-M. Forecasting enrollments based on fuzzy time series. *Fuzzy Sets and Systems*, v. 81, n. 3, p. 311 – 319, 1996. ISSN 0165-0114. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/0165011495002200⟩. Acesso em: 13 ago. 2020.

CHINDANUR, n. b.; B, E. Performance comparison of four new arima-ann prediction models on internet traffic data. *Journal of Telecommunications and Information Technology*, v. 2015, p. 67–75, 01 2015.

CHUNG, H.; LEE, S. J.; PARK, J. G. *Deep neural network using trainable activation functions*. 2016. 348-352 p.

CLAUSET, A.; MOORE, C.; NEWMAN, M. E. J. Hierarchical structure and the prediction of missing links in networks. *Nature*, Springer Science and Business Media LLC, v. 453, n. 7191, p. 98–101, May 2008. ISSN 1476-4687. Disponível em: ⟨http://dx.doi.org/10.1038/nature06830⟩.

COCHRANE, J. H. *Time Series for Macroeconomics and Finance*. [s.n.], 1997. Disponível em: ⟨http://econ.lse.ac.uk/staff/wdenhaan/teach/cochrane.pdf⟩. Acesso em: 01 abr. 2020.

COSTA, L. da F.; RODRIGUES, F.; TRAVIESO, G.; BOAS, P. V. Characterization of complex networks: A survey of measurements. *Advances in Physics*, v. 56, p. 167–242, 01 2007. Disponível em: ⟨https://www.researchgate.net/publication/232874573_Characterization_of_Complex_Networks_A_Survey_of_measurements⟩. Acesso em: 08 abr. 2020.

CSARDI, G.; NEPUSZ, T. The igraph software package for complex network research. *InterJournal Complex Systems*, n. 1695, 2006. Disponível em: ⟨http://cneurocvs.rmki.kfki.hu/igraph/doc-0.5.1/igraph-docs.pdf⟩.

DAI, X.; LIU, J.; ZHANG, H. Application of ar model in the analysis of preearthquake ionospheric anomalies. *Mathematical Problems in Engineering*, Hindawi Publishing Corporation, v. 2015, p. 157184, Oct 2015. ISSN 1024-123X. Disponível em: ⟨https://doi.org/10.1155/2015/157184⟩.

DICE, L. R. Measures of the amount of ecologic association between species. *Ecology*, John Wiley and Sons, v. 26, n. 3, p. 297–302, Jul 1945.

DIEBOLD, F.; MARIANO, R. Comparing predictive accuracy. *Journal of Business and Economic Statistics*, v. 13, p. 253—-263, 07 1995. Disponível em: ⟨http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.454.4490&rep=rep1&type=pdf⟩. Acesso em: 30 Out. 2020.

DIESTEL, R. *Graph Theory*. New York: Springer-Verlag, 2005. ISBN 9783540261834. Disponível em: ⟨http://www.dsc.ufcg.edu.br/~ulrich/disciplinas/GraphTheory.pdf⟩. Acesso em: 17 abr. 2020.

DONG, L.; LI, Y.; YIN, H.; LE, H.; RUI, M. The algorithm of link prediction on social network. *Mathematical Problems in Engineering*, Hindawi Publishing Corporation, v. 2013, p. 125123, Sep 2013. ISSN 1024-123X. Disponível em: ⟨https://doi.org/10.1155/2013/125123⟩.

EFRON, B.; TIBSHIRANI, R. Book. *An Introduction to the Bootstrap*. 1st. ed. [S.l.]: Chapman and Hall/CRC, 1994. ISBN 9780429246593.

EGRIOGLU, E.; FILDES, R. A new bootstrapped hybrid artificial neural network approach for time series forecasting. *Computational Economics*, Nov 2020. ISSN 1572-9974. Disponível em: ⟨https://doi.org/10.1007/s10614-020-10073-7⟩.

EMMANOUILIDIS, K.; KARPETIS, C. The defense–growth nexus: A review of time series methods and empirical results. *Defence and Peace Economics*, Routledge, v. 31, n. 1, p. 86–104, 2020. Disponível em: ⟨https://doi.org/10.1080/10242694.2018.1428261⟩. Acesso em: 30 mar. 2020.

EULER, L. Solutio problematis ad geometriam situs pertinentis. *The Euler Archives - All works*, Scholarly Commons, n. 53, 1741. Disponível em: ⟨https://scholarlycommons.pacific.edu/euler-works/53/⟩. Acesso em: 06 abr. 2020.

FEOFILOFF, P.; KOHAYAKAWA, Y.; WAKABAYASHI, Y. Uma introdução sucinta à teoria dos grafos. 01 2009. Disponível em: ⟨https://www.researchgate.net/publication/327057443_Uma_Introducao_Sucinta_a_Teoria_dos_Grafos/link/5b7583bba6fdcc87df810f1e/download⟩. Acesso em: 15 abr. 2020.

FERREIRA, L. N. *Time series data mining using complex networks*. Doutorado em Ciências de Computação e Matemática Computacional, 2017. Disponível em: ⟨https://teses.usp.br/teses/disponiveis/55/55134/tde-01022018-144118/en.php⟩. Acesso em: 07 abr. 2020.

FILATOV, D. M.; LYUBUSHIN, A. A. Precursory analysis of gps time series for seismic hazard assessment. *Pure and Applied Geophysics*, v. 177, n. 1, p. 509–530, 2020. ISSN 1420-9136. Disponível em: ⟨https://doi.org/10.1007/s00024-018-2079-3⟩. Acesso em: 30 mar. 2020.

FREITAS, C. G. S.; AQUINO, A. L. L.; RAMOS, H. S.; FRERY, A. C.; ROSSO, O. A. A detailed characterization of complex networks using information theory. *Scientific Reports*, v. 9, n. 1, p. 16689, 2019. ISSN 2045-2322. Disponível em: ⟨https://doi.org/10.1038/s41598-019-53167-5⟩. Acesso em: 17 abr. 2020.

FULLER, W. A. *Introduction to Statistical Time Series*. 2nd ed. ed. [S.l.]: John Wiley and Sons, 1996. (Wiley series in Probability and Statistics). ISBN 9780470317754.

GAO, X.; SITHARAM, M.; ROITBERG, A. Bounds on the jensen gap, and implications for mean-concentrated distributions. *The Australian Journal of Mathematical Analysis and Applications*, v. 141, p. 1–16, 11 2019. Disponível em: ⟨https://ajmaa.org/searchroot/files/pdf/v16n2/v16i2p14.pdf⟩. Acesso em: 10 aug. 2021.

GAO, Z.; JIN, N. Complex network from time series based on phase space reconstruction. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, v. 19, n. 3, p. 033137, 2009. Disponível em: ⟨https://doi.org/10.1063/1.3227736⟩. Acesso em: 08 abr. 2020.

GAO, Z.-K.; CAI, Q.; YANG, Y.-X.; DANG, W.-D.; ZHANG, S.-S. Multiscale limited penetrable horizontal visibility graph for analyzing nonlinear timeseries. *Scientific Reports*, v. 6, n. 1, p. 35622, 2016. ISSN 2045-2322. Disponível em: ⟨https://doi.org/10.1038/srep35622⟩. Acesso em: 08 abr. 2020.

GIBBONS, A. *Algorithmic Graph Theory*. 1. ed. [S.l.]: Cambridge University Press, 1985. 2 p.

GREENBERG, D. F. Time series analysis of crime rates. *Journal of Quantitative Criminology*, v. 17, n. 4, p. 291–327, 2001. ISSN 1573-7799. Disponível em: ⟨https://doi.org/10.1023/A:1012507119569⟩. Acesso em: 30 mar. 2020.

HAMILTON, J. D. *Time Series Analysis*. 1. ed. [S.l.]: Princeton University Press, 1994. ISBN 0691042896.

HIPEL, K.; MCLEOD, A. *Time Series Modelling of Water Resources and Environmental Systems*. 1. ed. [S.l.]: Amsterdam: Elsevier, 1994. v. 45.

HOCHREITER, S.; SCHMIDHUBER, J. Long Short-Term Memory. *Neural Computation*, v. 9, n. 8, p. 1735–1780, 11 1997. ISSN 0899-7667. Disponível em: ⟨https://doi.org/10.1162/neco.1997.9.8.1735⟩.

HOLT, C. C. Forecasting seasonals and trends by exponentially weighted moving averages. *International Journal of Forecasting*, v. 20, n. 1, p. 5–10, 2004. ISSN 0169-2070. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0169207003001134⟩.

HU, Y. Iterative and recursive least squares estimation algorithms for moving average systems. *Simulation Modelling Practice and Theory*, v. 34, p. 12 – 19, 2013. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S1569190X1300004X⟩. Acesso em: 01 abr. 2020.

HYLLEBERG, S. (Ed.). *Modelling Seasonality*. Oxford University Press, 1992. Disponível em: ⟨https://EconPapers.repec.org/RePEc:oxp:obooks:9780198773184⟩.

HYNDMAN, R.; KHANDAKAR, Y. Automatic time series forecasting: The forecast package for r. *Journal of Statistical Software, Articles*, v. 27, n. 3, p. 1–22, 2008. ISSN 1548-7660. Disponível em: ⟨https://www.jstatsoft.org/v027/i03⟩.

HYNDMAN, R.; KOEHLER, A. B.; ORD, J. K.; SNYDER, R. D. *Forecasting with exponential smoothing: the state space approach*. [S.l.]: Springer Science & Business Media, 2008.

HYNDMAN, R. J.; ATHANASOPOULOS, G. Book. *Forecasting : principles and practice*. 2. ed. Melbourn, Australia: OTexts.com, 2018. Disponível em: ⟨https://otexts.com/fpp2/arima-r.html⟩. Acesso em: 20 Set. 2020.

HYNDMAN, R. J.; KOEHLER, A. B. Another look at measures of forecast accuracy. *International Journal of Forecasting*, v. 22, n. 4, p. 679–688, 2006. ISSN 0169-2070. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0169207006000239⟩.

IELTS. *Underground Station passenger numbers in London*. 2020. Disponível em: ⟨https://www.ielts-mentor.com/writing-sample/academic-writing-task-1/95-underground-station-passenger-numbers-in-london⟩.

IMHOFF, M.; BAUER, M.; GATHER, U.; LöHLEIN, D. *Time series analysis in intensive care medicine*. Dortmund, 1998. Disponível em: ⟨http://hdl.handle.net/10419/77286⟩. Acesso em: 30 mar. 2020.

INVESTING. *Ibovespa (BVSP)*. 2021. Disponível em: ⟨https://br.investing.com/indices/bovespa-chart⟩.

JIANG, W.; WEI, B.; TANG, Y.; ZHOU, D. Ordered visibility graph average aggregation operator: An application in produced water management. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, AIP Publishing LLC, v. 27, n. 2, p. 023117, 2017.

JOFIPASI, C. A.; MIFTAHUDDIN; HIZIR. Selection for the best ETS (error, trend, seasonal) model to forecast weather in the aceh besar district. *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, v. 352, p. 012055, may 2018. Disponível em: ⟨https://doi.org/10.1088/1757-899x/352/1/012055⟩.

JUNGNICKEL, D. *Algorithms and Computation in Mathematics*. 2. ed. Springer, 2005. v. 5. Disponível em: ⟨https://link.springer.com/content/pdf/bfm\%3A978-3-540-26908-3\%2F1.pdf⟩. Acesso em: 06 abr. 2020.

JUNGNICKEL, D. *Graphs, Networks and Algorithms*. [S.l.]: Springer-Verlag Berlin Heidelberg, 2013. v. 5. 676 p.

KAASTRA, I.; BOYD, M. Designing a neural network for forecasting financial and economic time series. *Neurocomputing*, v. 10, n. 3, p. 215–236, 1996. ISSN 0925-2312. Financial Applications, Part II. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/0925231295000399⟩.

KANG, S. Y. *An Investigation of the Use of Feedforward Neural Networks for Forecasting*. Tese (Doutorado), USA, 1992. UMI Order No. GAX92-01899.

KAUSHIK, S.; CHOUDHURY, A.; SHERON, P. K.; DASGUPTA, N.; NATARAJAN, S.; PICKETT, L. A.; DUTT, V. Ai in healthcare: Time-series forecasting using statistical, neural, and ensemble architectures. *Frontiers in Big Data*, v. 3, p. 4, 2020. ISSN 2624-909X. Disponível em: ⟨https://www.frontiersin.org/article/10.3389/fdata.2020.00004⟩.

KHASHEI, M.; BIJARI, M. A novel hybridization of artificial neural networks and arima models for time series forecasting. *Applied Soft Computing*, v. 11, n. 2, p. 2664–2675, 2011. ISSN 1568-4946. The Impact of Soft Computing for the Progress of Artificial Intelligence. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S1568494610002759⟩.

KIM, J.; HASTAK, M. Social network analysis: Characteristics of online social networks after a disaster. *International Journal of Information Management*, v. 38, n. 1, p. 86 – 96, 2018. ISSN 0268-4012. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S026840121730525X⟩. Acesso em: 17 abr. 2020.

KOKOSZKA, P.; YOUNG, G. Kpss test for functional time series. *Statistics*, Taylor and Francis, v. 50, n. 5, p. 957–973, 2016. Disponível em: ⟨https://doi.org/10.1080/02331888.2015.1128937⟩.

KOTU, V.; DESHPANDE, B. *Predictive Analytics and Data Mining*. 1. ed. [s.n.], 2015. 446 p. Disponível em: ⟨https://www.sciencedirect.com/topics/computer-science/data-mining⟩. Acesso em: 03 abr. 2020.

KRAMER, M. A.; EDEN, U. T.; CASH, S. S.; KOLACZYK, E. D. Network inference with confidence from multivariate time series. *Phys. Rev. E*, American Physical Society, v. 79, p. 061916, Jun 2009. Disponível em: ⟨https://link.aps.org/doi/10.1103/PhysRevE.79.061916⟩. Acesso em: 08 abr. 2020.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. *ImageNet Classification with Deep Convolutional Neural Networks*. Curran Associates, Inc., 2012. Disponível em: ⟨https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf⟩.

KUNST, R. M.; WAGNER, M. Economic forecasting: editors' introduction. *Empirical Economics*, v. 58, n. 1, p. 1–5, 2020. ISSN 1435-8921. Disponível em: ⟨https://doi.org/10.1007/s00181-019-01820-3⟩. Acesso em: 30 mar. 2020.

KWIATKOWSKI, D.; PHILLIPS, P. C.; SCHMIDT, P.; SHIN, Y. Testing the null hypothesis of stationarity against the alternative of a unit root. *Journal of Econometrics*, Elsevier Science Publishers B.V, North-Holland, v. 140, n. 54, p. 159–178, 1992. Disponível em: ⟨http://debis.deu.edu.tr/userweb/onder.hanedar/dosyalar/kpss.pdf⟩. Acesso em: 15 Ago. 2020.

LACASA, L.; LUQUE, B.; BALLESTEROS, F.; LUQUE, J.; NUÑO, J. C. From time series to complex networks: The visibility graph. *Proceedings of the National Academy of Sciences*, National Academy of Sciences, v. 105, n. 13, p. 4972–4975, 2008. ISSN 0027-8424. Disponível em: ⟨https://www.pnas.org/content/105/13/4972⟩. Acesso em: 30 mar. 2020.

LACASA, L.; NICOSIA, V.; LATORA, V. Network structure of multivariate time series. *Scientific Reports*, v. 5, n. 1, p. 15508, 2015. ISSN 2045-2322. Disponível em: ⟨https://doi.org/10.1038/srep15508⟩. Acesso em: 30 mar. 2020.

LAN, X.; MO, H.; CHEN, S.; LIU, Q.; DENG, Y. Fast transformation from time series to visibility graphs. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, v. 25, n. 8, p. 083105, 2015. Disponível em: ⟨https://doi.org/10.1063/1.4927835⟩.

LATORA, V.; NICOSIA, V.; RUSSO, G. *Complex Networks: Principles, Methods and Applications*. [S.l.]: Cambridge University Press, 2017. 594 p. ISBN 9781107103184. Acesso em: 17 abr. 2020.

LöFFLER, A.; KRUSCHWITZ, L. *Expectation and Lebesgue Integral*. Cham: Springer International Publishing, 2019. 69-86 p. ISBN 978-3-030-20103-6. Disponível em: ⟨https://doi.org/10.1007/978-3-030-20103-6_5⟩. Acesso em: 19 Ago. 2020.

LIU, W.; Lü, L. Link prediction based on local random walk. *EPL (Europhysics Letters)*, IOP Publishing, v. 89, n. 5, p. 58007, Mar 2010. ISSN 1286-4854. Disponível em: ⟨http://dx.doi.org/10.1209/0295-5075/89/58007⟩. Acesso em: 01 ago. 2020.

LJUNG, G.; BOX, G. On a measure of lack of fit in time series models. *Biometrika*, v. 65, 08 1978. Disponível em: ⟨https://www.researchgate.net/publication/246995234_On_a_Measure_of_Lack_of_Fit_in_Time_Series_Models⟩. Acesso em: 17 Ago. 2020.

LOPES, F. J. A.; TáBOAS, P. Z. Euler e as pontes de königsberg. *Revista Brasileira de História da Matemática*, Sociedade Brasileira de História da Matemática, v. 15, n. 30, p. 23 – 32, 2015. Disponível em: ⟨http://www.rbhm.org.br/issues/RBHM\%20-\%20vol.15,no30/3\%20-\%20Frederico\%20Lopes.pdf⟩. Acesso em: 06 abr. 2020.

LU, L.; ZHOU, T. Link prediction in complex networks: A survey. *ArXiv*, abs/1010.0725, 2010.

LUQUE, B.; LACASA, L.; BALLESTEROS, F.; LUQUE, J. Horizontal visibility graphs: Exact results for random time series. *Phys. Rev. E*, American Physical Society, v. 80, p. 046103, Oct 2010. Disponível em: ⟨https://link.aps.org/doi/10.1103/PhysRevE.80.046103⟩. Acesso em: 08 abr. 2020.

LYTRAS, D. On seasonality: Comparing x-13arima-seats diagnostics for quarterly series. *U.S. Census Working Papers*, 2015. Disponível em: ⟨https://www.census.gov/library/working-papers/2015/adrm/lytras-01.html⟩. Acesso em: 02 Set. 2020.

MA, Y.; GUO, G. *Support Vector Machines Applications*. [S.l.]: Springer, Cham, 2014. ISBN 978-3-319-02300-7.

MACROTRENDS. *South Korea Infant Mortality Rate 1950-2021*. 2021. Disponível em: ⟨https://www.macrotrends.net/countries/KOR/south-korea/infant-mortality-rate⟩.

MAKRIDAKIS, S.; SPILIOTIS, E.; ASSIMAKOPOULOS, V. Statistical and machine learning forecasting methods: Concerns and ways forward. *PloS one*, Public Library of Science, v. 13, n. 3, p. e0194889–e0194889, Mar 2018. ISSN 1932-6203. 29584784[pmid]. Disponível em: ⟨https://pubmed.ncbi.nlm.nih.gov/29584784⟩.

MAO, S.; XIAO, F. Time series forecasting based on complex network analysis. *IEEE Access*, v. 7, p. 40220 – 40229, 2019. Disponível em: ⟨https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8669744⟩. Acesso em: 20 jul. 2020.

MARTINEZ, N. D.; HAWKINS, B. A.; DAWAH, H. A.; FEIFAREK, B. P. Effects of sampling effort on characterization of food-web structure. *Ecology*, John Wiley and Sons, v. 80, n. 3, p. 1044–1055, Apr 1999.

MARUSICH, L. R.; BUCHLER, N. Time series modeling of army mission command communication networks: an event-driven analysis. *Computational and Mathematical Organization Theory*, v. 22, n. 4, p. 467–486, 2016. Disponível em: ⟨https://doi.org/10.1007/s10588-016-9211-7⟩. Acesso em: 30 mar. 2020.

MARWAN, N.; DONGES, J. F.; ZOU, Y.; DONNER, R. V.; KURTHS, J. Complex network approach for recurrence analysis of time series. *Physics Letters A*, Elsevier BV, v. 373, n. 46, p. 4246–4254, Nov 2009. ISSN 0375-9601. Disponível em: ⟨http://dx.doi.org/10.1016/j.physleta.2009.09.042⟩.

MCQUARRIE, A. D. R.; TSAI, C.-L. *Regression and Time Series Model Selection*. WORLD SCIENTIFIC, 1998. Disponível em: ⟨https://www.worldscientific.com/doi/abs/10.1142/3573⟩.

MüLLER, U. A theory of robust long-run variance estimation. *Journal of Econometrics*, v. 141, p. 1331–1352, 02 2007. Disponível em: ⟨https://www.princeton.edu/~umueller/robLRVP.pdf⟩.

MOREIRA, F. Equações recorrentes. Rumoaoita, 2006. Disponível em: ⟨https://rumoaoita.com/wp-content/uploads/2017/03/topicos_adicionais_equacoes_recorrentes_ita.pdf⟩.

MORETTIN, P. A.; TOLOI, C. M. d. C. *Análise de séries temporais*. 2006.

MUCHNIK, L.; PEI, S.; PARRA, L. C.; REIS, S. D. S.; JR, J. S. A.; HAVLIN, S.; MAKSE, H. A. Origins of power-law degree distribution in the heterogeneity of human activity in social networks. *Scientific Reports*, v. 3, n. 1, p. 1783, 2013. ISSN 2045-2322. Disponível em: ⟨https://doi.org/10.1038/srep01783⟩. Acesso em: 17 abr. 2020.

NGUYEN, D. Q.; PHAN, M. N.; ZELINKA, I. Periodic time series forecasting with bidirectional long short-term memory: Periodic time series forecasting with bidirectional lstm. In: *2021 The 5th International Conference on Machine Learning and Soft Computing*. New York, NY, USA: Association for Computing Machinery, 2021. (ICMLSC'21), p. 60–64. ISBN 9781450387613. Disponível em: ⟨https://doi.org/10.1145/3453800.3453812⟩.

NISBET, R.; MINER, G.; ELDER, J. *Handbook of Statistical Analysis and Data Mining Applications*. 1. ed. [s.n.], 2009. Disponível em: ⟨https://www.sciencedirect.com/topics/computer-science/data-mining⟩. Acesso em: 03 abr. 2020.

OGATA, K. (Ed.). *Discrete-Time Control Systems*. Second edition. [S.l.]: Prentice-Hall, 1995.

OZAKI, T. 11 - spatial time series modeling for fmri data analysis in neurosciences. In: Subba Rao, T.; Subba Rao, S.; RAO, C. (Ed.). *Time Series Analysis: Methods and Applications*. Elsevier, 2012, (Handbook of Statistics, v. 30). p. 297 – 313. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/B9780444538581000119⟩. Acesso em: 01 ago. 2020.

PACIFIC, U. *The Euler Archive - All Works*. 2020. Disponível em: ⟨https://scholarlycommons.pacific.edu/euler-works/53/⟩. Acesso em: 06 abr. 2020.

PANIGRAHI, S.; BEHERA, D. H. A hybrid ets-ann model for time series forecasting. *Eng. Appl. of AI*, v. 66, p. 49–59, 11 2017.

PATHAK, H. K.; SINGH, P. A New Bandwidth Interval Based Forecasting Method for Enrollments Using Fuzzy Time Series. *Applied Mathematics*, Vol.02, No.04, p. 4, 2011. Disponível em: ⟨https://www.scirp.org/html/4515.html⟩. Acesso em: 13 ago. 2020.

PETRIS, G. Book. *Dynamic Linear Models with R*. [S.l.]: Springer, 2009. ISBN 9780387772370.

PIATETSKY-SHAPIRO, G.; FRAWLEY, W. *Knowledge discovery in databases*. [S.l.: s.n.], 1991.

POHLERT, T. The pairwise multiple comparison of mean ranks package (pmcmr). *R package*, 2014. Disponível em: ⟨http://CRAN.R-project.org/package=PMCMR⟩. Acesso em: 02 Set. 2020.

PUROHIT, S.; PANIGRAHI, S.; SETHY, P.; BEHERA, S. Time series forecasting of price of agricultural products using hybrid methods. *Applied Artificial Intelligence*, 09 2021. Disponível em: ⟨https://www.researchgate.net/publication/354686351_Time_Series_Forecasting_of_Price_of_Agricultural_Products_Using_Hybrid_Methods⟩.

RADHAKRISHNAN, G.; GUPTA, D.; SINDHUULA, S.; KHOKHAWAT, S.; TSB, S. Experimentation and analysis of time series data from multi-path robotic environment. In: *2015 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*. [S.l.: s.n.], 2015. p. 1–6.

RAO, S. *A course in Time Series Analysis*. [s.n.], 2021. v. 56. 512 p. Disponível em: ⟨https://web.stat.tamu.edu/~suhasini/teaching673/time_series.pdf⟩.

REHMAN, A.; JINGDONG, L.; CHANDIO, A. A.; HUSSAIN, I.; WAGAN, S. A.; MEMON, Q. U. A. Economic perspectives of cotton crop in pakistan: A time series analysis (1970–2015) (part 1). *Journal of the Saudi Society of Agricultural Sciences*, v. 18, n. 1, p. 49 – 54, 2019. ISSN 1658-077X. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S1658077X1630162X⟩. Acesso em: 30 mar. 2020.

RENCHER, A. C. *Methods of multivariate analysis*. 2nd ed. ed. J. Wiley, 2002. (Wiley series in probability and mathematical statistics). ISBN 9780471418894,9780471461722,0471418897. Disponível em: ⟨http://gen.lib.rus.ec/book/index.php?md5=6a17f17011a284ab9394481ad6811c46⟩. Acesso em: 20 Ago. 2020.

ROHMAH, M. F.; PUTRA, I. K. G. D.; HARTATI, R. S.; ARDIANTORO, L. Comparison four kernels of svr to predict consumer price index. *Journal of Physics: Conference Series*, IOP Publishing, v. 1737, 2021. Disponível em: ⟨https://iopscience.iop.org/article/10.1088/1742-6596/1737/1/012018/pdf⟩.

ROSENBLATT, F. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. 03 1961. Disponível em: ⟨https://apps.dtic.mil/sti/citations/AD0256582⟩.

RUMELHART, D.; HINTON, G.; WILLIAMS, R. Long short-term memory. *Nature*, p. 533–536, 1986.

SAMSUDIN, R.; SHABRI, A.; SAAD, P. A comparison of time series forecasting using support vector machine and artificial neural network model. *Journal of Applied Sciences*, 2010. Disponível em: ⟨https://scialert.net/fulltext/?doi=jas.2010.950.958⟩.

SCARSOGLIO, S.; RIDOLFI, L.; LACOBELLO, G. Complex network unveling spatial patterns in turbulence. 2016.

SCHAUMBERGER, N. Another proof of the inequality between power means. *The College Mathematics Journal*, Mathematical Association of America, v. 19, n. 1, p. 56–58, 1988. ISSN 07468342, 19311346. Disponível em: ⟨http://www.jstor.org/stable/2686705⟩. Acesso em: 10 aug. 2021.

SHIBLEE, M.; KALRA, P.; CHANDRA, B. Time series prediction with multilayer perceptron (mlp): A new generalized error based approach. In: . [S.l.: s.n.], 2008. p. 37–44. ISBN 978-3-642-03039-0.

SHUMWAY, R.; STOFFER, D. *Time Series Analysis and Its Applications With R Examples*. [S.l.: s.n.], 2011. v. 9. Acesso em: 30 mar. 2020.

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

SKIENA, S. *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*. [S.l.]: Addison-Wesley, 1990. 135-136 p. ISBN 0201509431.

SáNDOR, B.; SCHNEIDER, B.; LáZáR, Z. I.; ERCSEY-RAVASZ, M. A novel measure inspired by lyapunov exponents for the characterization of dynamics in state-transition networks. *Entropy*, v. 23, n. 1, 2021. ISSN 1099-4300. Disponível em: ⟨https://www.mdpi.com/1099-4300/23/1/103⟩.

STEEN, M. van. Graph theory and complex networks: An introduction. In: . [S.l.: s.n.], 2010.

STOICA, P.; SELÉN, Y. Model-order selection: a review of information criterion rules. *IEEE Signal Processing Magazine*, v. 21, p. 36–47, 2004. Disponível em: ⟨http://www.sal.ufl.edu/eel6935/2008/01311138_ModelOrderSelection_Stoica.pdf⟩.

STROGATZ, S. H. Exploring complex networks. *Nature*, v. 410, n. 6825, p. 268–276, Mar 2001. ISSN 1476-4687. Disponível em: ⟨https://doi.org/10.1038/35065725⟩.

TADDY, M. *Business Data Science: Combining Machine Learning and Economics to Optimize, Automate, and Accelerate Business Decisions*. McGraw-Hill Education, 2019. ISBN 9781260452785. Disponível em: ⟨https://books.google.com.br/books?id=yPOUDwAAQBAJ⟩.

TAKEMOTO, K.; AKUTSU, T. Analysis of the effect of degree correlation on the size of minimum dominating sets in complex networks. *PLOS ONE*, Public Library of Science, v. 11, n. 6, p. 1–11, 06 2016. Disponível em: ⟨https://doi.org/10.1371/journal.pone.0157868⟩.

TEAM, R. C.; WORLDWIDE contributors. *datasets-package: The R Datasets Package*. [S.l.], 2021. R package version 4.2.0. Disponível em: ⟨https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/nhtemp.html⟩. Acesso em: 18 Aug. 2021.

TELESCA, L.; LOVALLO, M. Analysis of seismic sequences by using the method of visibility graph. *EPL (Europhysics Letters)*, IOP Publishing, v. 97, n. 5, p. 50002, feb 2012. Disponível em: ⟨https://doi.org/10.1209\%2F0295-5075\%2F97\%2F50002⟩.

TOIVONEN, R.; ONNELA, J.-P.; SARAMäKI, J.; HYVöNEN, J.; KASKI, K. A model for social networks. *Physica A: Statistical Mechanics and its Applications*, v. 371, n. 2, p. 851 – 860, 2006. ISSN 0378-4371. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S0378437106003931⟩. Acesso em: 17 abr. 2020.

TORRES, J.; HADJOUT, D.; SEBAA, A.; MARTINEZ-ALVAREZ, F.; TRONCOSO, A. Deep learning for time series forecasting: A survey. *Big Data*, v. 9, 12 2020. Disponível em: ⟨https://www.researchgate.net/publication/347364694_Deep_Learning_for_Time_Series_Forecasting_A_Survey⟩.

TRADINGECONOMICS. *Brazil Car Production*. 2021. Disponível em: ⟨https://tradingeconomics.com/brazil/car-production⟩.

VEHTARI, A.; GELMAN, A.; GABRY, J. Practical bayesian model evaluation using leave-one-out cross-validation and waic. *Statistics and Computing*, v. 27, n. 5, p. 1413–1432, Sep 2017. ISSN 1573-1375. Disponível em: ⟨https://doi.org/10.1007/s11222-016-9696-4⟩.

WEBEL, K.; OLLECH, D. An overall seasonality test based on recursive feature elimination in conditional random forests. In: *International Conference on Time Series and Forecasting.Proceedings of Papers.* [S.l.: s.n.], 2018. v. 1, p. 20 – 31. ISBN 978-84-17293-57-4. Acesso em: 02 Set. 2020.

WEI, Z.-W.; WANG, B.-H. Emergence of fractal scaling in complex networks. *Phys. Rev. E*, American Physical Society, v. 94, p. 032309, Sep 2016. Disponível em: ⟨https://link.aps.org/doi/10.1103/PhysRevE.94.032309⟩.

WEN, Q. Asset Growth and Stock Market Returns: A Time-Series Analysis*. *Review of Finance*, v. 23, n. 3, p. 599–628, 06 2018. ISSN 1572-3097. Disponível em: ⟨https://doi.org/10.1093/rof/rfy018⟩. Acesso em: 30 mar. 2020.

WEST, D. B. *Introduction to Graph Theory*. 2nd. ed. [S.l.]: Pearson Education, 2000. 2 p.

WINTERS, P. R. Forecasting sales by exponentially weighted moving averages. *Management Science*, INFORMS, v. 6, n. 3, p. 324–342, 1960. ISSN 00251909, 15265501. Disponível em: ⟨http://www.jstor.org/stable/2627346⟩.

WIT, E.; HEUVEL, E. v. d.; ROMEIJN, J.-W. 'all models are wrong...': an introduction to model uncertainty. *Statistica Neerlandica*, v. 66, n. 3, p. 217–236, 2012. Disponível em: ⟨https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9574.2012.00530.x⟩.

WORLDPOPULATIONREVIEW. *Brazil Population 2021 (Live)*. 2021. Disponível em: ⟨https://worldpopulationreview.com/countries/brazil-population⟩.

XENARIOS, I.; RICE, D. W.; SALWINSKI, L.; BARON, M. K.; MARCOTTE, E. M.; EISENBERG, D. Dip: the database of interacting proteins. *Nucleic acids research*, Oxford University Press, v. 28, n. 1, p. 289–291, Jan 2000. ISSN 0305-1048. Disponível em: ⟨https://pubmed.ncbi.nlm.nih.gov/10592249⟩. Acesso em: 22 abr. 2020.

YANG, Y.; YANG, H. Complex network-based time series analysis. *Physica A: Statistical Mechanics and its Applications*, v. 387, n. 5, p. 1381–1386, 2008. ISSN 0378-4371. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0378437107011235⟩.

YU, H.-K. Weighted fuzzy time series models for taiex forecasting. *Physica A: Statistical Mechanics and its Applications*, v. 349, n. 3, p. 609 – 624, 2005. ISSN 0378-4371. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S0378437104014128⟩. Acesso em: 13 ago. 2020.

ZAKI, M. J. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Press, 2014. Disponível em: ⟨https://www.xarg.org/ref/a/0521766338/⟩. Acesso em: 04 abr. 2020.

ZENIL, H.; KIANI, N. A.; TEGNÉR, J. A review of graph and network complexity from an algorithmic information perspective. *Entropy (Basel, Switzerland)*, MDPI, v. 20, n. 8, p. 551, Jul 2018. ISSN 1099-4300. Disponível em: ⟨https://pubmed.ncbi.nlm.nih.gov/33265640⟩.

ZHANG, J.; SMALL, M. Complex network from pseudoperiodic time series: Topology versus dynamics. *Physical review letters*, v. 96, p. 238701, 07 2006.

ZHANG, R.; ASHURI, B.; DENG, Y. A novel method for forecasting time series based on fuzzy logic and visibility graph. *Advances in Data Analysis and Classification*, v. 11, n. 4, p. 759–783, Dec 2017. ISSN 1862-5355. Disponível em: ⟨https://doi.org/10.1007/s11634-017-0300-3⟩. Acesso em: 01 ago. 2020.

ZHOU, H.; LI, W.; ZHANG, C.; LIU, J. Ice breakup forecast in the reach of the yellow river: the support vector machines approach. *Hydrology and Earth System Sciences Discussions*, v. 6, 04 2009.

ZINOVIEV, D. *Complex Network Analysis in Python. Recognize → Construct → Visualize → Analyze → Interpret*. [S.l.: s.n.], 2017. ISBN 978-1-68050-269-5.

ZOU, Y.; DONNER, R. V.; MARWAN, N.; DONGES, J. F.; KURTHS, J. Complex network approaches to nonlinear time series analysis. *Physics Reports*, v. 787, p. 1 – 97, 2019. ISSN 0370-1573. Complex network approaches to nonlinear time series analysis. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S037015731830276X⟩. Acesso em: 08 abr. 2020.

Appendix A

A.1 Proof of equation 2.4

To proof equation 2.4, consider any two real numbers α and β, and any two random variables, Y_t and X_t, from the same space of probability $(\Omega, \mathcal{F}, P)$.

Proof of equation 2.4.

$$\begin{aligned} E\left[\alpha X_t + \beta W_t\right] &= \int_\Omega \left(\alpha X_t(\omega) + \beta W_t(\omega)\right) dP(\omega) = \\ &= \alpha \int_\Omega X_t(\omega) dP(\omega) + \beta \int_\Omega W_t(\omega) dP(\omega) = \\ &= \alpha E\left[X_t\right] + \beta E\left[W_t\right] \end{aligned} \tag{A.1}$$

□

A.2 Proof of equation 2.6

To proof equation 2.6, consider a random variable Y_t from the space of probability $(\Omega, \mathcal{F}, P)$.

Proof of equation 2.6.

$$\begin{aligned} Var\left(Y_t\right) &= E\left[\left(Y_t - E\left[Y_t\right]\right)^2\right] = E\left[Y_t^2 - 2Y_t E\left[Y_t\right] + \left(E\left[Y_t\right]\right)^2\right] = \\ &= E\left[Y_t^2\right] - 2\left(E\left[Y_t\right]\right)^2 + \left(E\left[Y_t\right]\right)^2 \\ &= E\left[Y_t^2\right] - \mu_{Y_t}^2. \end{aligned} \tag{A.2}$$

□

A.3 Proof of equation 2.14

Proof of equation 2.14. Let $\mathbf{u}$ and $\mathbf{v}$ be vectors such as θ is the angle between them. The inner product between $\mathbf{u}$ and $\mathbf{v}$ is calculated as $<\mathbf{u} \cdot \mathbf{v}> = |\mathbf{u}| \cdot |\mathbf{v}| \cos(\theta)$. But, as $|cos(\theta)| \leq 1$, hence:

$$\left| \frac{<\mathbf{u} \cdot \mathbf{v}>}{|\mathbf{u}| \cdot |\mathbf{v}|} \right| \leq 1. \tag{A.3}$$

The result expressed in equation A.3 is known as *Cauchy-Schwartz inequality.* Let $\mathbf{u} = (x_1 - \mu_x, x_2 - \mu_x, \cdots, x_n - \mu_x)$ and $\mathbf{v} = (y_1 - \mu_y, y_2 - \mu_y, \cdots, y_n - \mu_y)$, applying the

vectors $\mathbf{u}$ and $\mathbf{v}$ in *Cauchy-Schwartz inequality*, as describing just above, yields:

$$\left| \frac{\sum_{i=1}^{n} (x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_{i=1}^{n} (x_i - \mu_x)^2} \sqrt{\sum_{i=1}^{n} (y_i - \mu_y)^2}} \right| \leq 1. \tag{A.4}$$

This result, given by equation A.4 is known as the sample correlation, ρ_{XY}, between two random variables X and Y and it means that $-1 \leq \rho_{XY} \leq 1$. Define both vectors $\mathbf{u} = (y_{k+1} - \bar{y}, y_{k+2} - \bar{y}, \cdots, y_n - \bar{y})$ and $\mathbf{v} = (y_1 - \bar{y}, y_2 - \bar{y}, \cdots, y_{n-k} - \bar{y})$. The inner product between $\mathbf{u}$ and $\mathbf{v}$, $<\mathbf{u} \cdot \mathbf{v}>$ is calculated as:

$$<\mathbf{u} \cdot \mathbf{v}> = \sum_{i=k+1}^{n} (y_i - \bar{y})(y_{i-k} - \bar{y}). \tag{A.5}$$

Follows the modulus of both vectors $\mathbf{u}$ and $\mathbf{v}$ and each correspondent inequalities shown in equations A.6 and A.7.

$$|\mathbf{u}| = \sqrt{\sum_{i=k+1}^{n} (y_i - \bar{y})^2} \leq \sqrt{\sum_{i=1}^{n} (y_i - \bar{y})^2} \tag{A.6}$$

$$|\mathbf{v}| = \sqrt{\sum_{i=k+1}^{n} (y_{i-k} - \bar{y})^2} \leq \sqrt{\sum_{i=1}^{n} (y_i - \bar{y})^2} \tag{A.7}$$

Taking into account equations A.6 and A.7, comes:

$$|\mathbf{u}| \cdot |\mathbf{v}| \leq \sum_{i=1}^{n} (y_i - \bar{y})^2 \tag{A.8}$$

The estimator $\hat{\rho}_{t,k}$, defined in equation 2.13 is combined with equation A.8 and Cauchy-Schwartz inequality, therefore, comes the following result:

$$|\hat{\rho}_{t,k}| = \frac{|<\mathbf{u} \cdot \mathbf{v}>|}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \leq \frac{|\mathbf{u}| \cdot |\mathbf{v}|}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \leq \frac{\sum_{i=1}^{n} (y_i - \bar{y})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} = 1. \tag{A.9}$$

□

A.4 Proof of equation 2.22

In order to verify the claim given by equation 2.22, consider the following calculations.

Proof of equation 2.22.

$$E\left[\left(Y_{n+1} - g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)\right)^2\right] = E\left[\left(Y_{n+1} - E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] + E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] - g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)\right)^2\right] =$$

$$= E\left[\left(Y_{n+1} - E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right]\right)^2\right] -$$

$$- 2E\left[\left(Y_{n+1} - E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right]\right)\left(E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] - g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)\right)\right] +$$

$$+ E\left[\left(E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] - g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)\right)^2\right]. \tag{A.10}$$

Given the result shown in equation A.10, the three terms on the right hand of the last equality must be discussed separately. Starting with the middle term, that is:

$$E\left[\left(Y_{n+1} - E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right]\right)\left(E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] - g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)\right)\right]. \tag{A.11}$$

Define $l_{n+1} = \left(Y_{n+1} - E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right]\right)\left(E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] - g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)\right)$. Considering that $\mathbf{Y}_{\boldsymbol{n,k}}$ is already known, it becomes reasonable calculating $E\left[l_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right]$ and taking into account the result of the *Law of Interacted Expectation*, $E\left[l_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] = E\left[l_{n+1}\right]$ (HAMILTON, 1994) . Given that $\mathbf{Y}_{\boldsymbol{n,k}}$ is fully known, $E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right]$ and $g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)$ are constant, with respect to $\mathbf{Y}_{\boldsymbol{n,k}}$, hence the factor $\left(E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] - g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)\right)$ can be excluded of the expectancy, therefore:

$$E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] = \overbrace{E\left[\left(Y_{n+1} - E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right]\right)|\mathbf{Y}_{\boldsymbol{n,k}}\right]}^{=0} \times E\left[\left(E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] - g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)\right)\right] = 0 \tag{A.12}$$

The next term in equation A.10 to be analyzed is $E\left[\left(Y_{n+1} - E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right]\right)^2\right]$. Note that this term is always non-negative and not dependent of $g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)$, hence, any choice taken to $g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)$ will impact this term. Therefore, the smaller is the term $E\left[\left(E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] - g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)\right)^2\right]$, the smaller is $E\left[\left(Y_{n+1} - g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)\right)^2\right]$. It is easy to observe that the term $E\left[\left(E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right] - g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right)\right)^2\right]$ is always non-negative and its minimum value is achieved when $g\left(\mathbf{Y}_{\boldsymbol{n,k}}\right) = E\left[Y_{n+1}|\mathbf{Y}_{\boldsymbol{n,k}}\right]$. Hence, the claim given by equation 2.22 is correct.

□

A.5 Proof of equation 2.25

The vector $\boldsymbol{\beta}$ must be calculated in the task of minimizing the mean squared error (MSE), given by

$$MSE = E\left[\left(\hat{Y}_{n+1|k} - \boldsymbol{\beta}^{\mathbf{T}}\mathbf{Y}_{\boldsymbol{n,k}}\right)^2\right] = E\left[\left(\hat{Y}_{n+1|k} - \boldsymbol{\beta}^{\mathbf{T}}\mathbf{Y}_{\boldsymbol{n,k}}\right)\cdot\left(\hat{Y}_{n+1|k} - \boldsymbol{\beta}^{\mathbf{T}}\mathbf{Y}_{\boldsymbol{n,k}}\right)^{\mathbf{T}}\right] =$$

$$= E\left[\left(\hat{Y}_{n+1|k} - \boldsymbol{\beta}^{\mathbf{T}}\mathbf{Y}_{\boldsymbol{n,k}}\right)\cdot\left(\hat{Y}_{n+1|k} - \mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\boldsymbol{\beta}\right)\right] = E\left[\hat{Y}^2_{n+1|k}\right] - \boldsymbol{\beta}^{\mathbf{T}}E\left[\hat{Y}_{n+1|k}\cdot\mathbf{Y}_{\boldsymbol{n,k}}\right] -$$

$$- E\left[\hat{Y}_{n+1|k}\cdot\mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\right]\boldsymbol{\beta} + E\left[\boldsymbol{\beta}^{\mathbf{T}}\mathbf{Y}_{\boldsymbol{n,k}}\mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\boldsymbol{\beta}\right].$$

Taking into account that $\boldsymbol{\beta}^{\mathbf{T}}\mathbf{Y}_{\boldsymbol{n,k}} = \mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\boldsymbol{\beta}$,

$$MSE = E\left[\hat{Y}^2_{n+1|k}\right] - 2\boldsymbol{\beta}^{\mathbf{T}}E\left[\hat{Y}_{n+1|k}\cdot\mathbf{Y}_{\boldsymbol{n,k}}\right] + E\left[\boldsymbol{\beta}^{\mathbf{T}}\mathbf{Y}_{\boldsymbol{n,k}}\mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\boldsymbol{\beta}\right]. \tag{A.13}$$

Differentiating the result achieved by A.13 with respect to β_i and making the result equals to null-vector, yields:

$$-2E\left[\hat{Y}_{n+1|k}\cdot\mathbf{Y}_{\boldsymbol{n,k}}\right] + 2E\left[\mathbf{Y}_{\boldsymbol{n,k}}\mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\right]\hat{\boldsymbol{\beta}} = \mathbf{0} \Leftrightarrow$$

$$\Leftrightarrow \hat{\boldsymbol{\beta}} = E\left[\mathbf{Y}_{\boldsymbol{n,k}}\mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\right]^{-1}E\left[\hat{Y}_{n+1|k}\cdot\mathbf{Y}_{\boldsymbol{n,k}}\right]. \tag{A.14}$$

A.6 Proof of equation 2.31

Proof of equation 2.31.

$$SSE\ \left(\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}}\right) = \left(\mathbf{Y}_{\boldsymbol{n,k}} - \mathbf{Y}\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}}\right)^{\mathbf{T}}\cdot\left(\mathbf{Y}_{\boldsymbol{n,k}} - \mathbf{Y}\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}}\right) =$$

$$= \left(\mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}} - \hat{\boldsymbol{\beta}}^{\mathbf{T}}_{\boldsymbol{OLS}}\mathbf{Y}^{\mathbf{T}}\right)\cdot\left(\mathbf{Y}_{\boldsymbol{n,k}} - \mathbf{Y}\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}}\right) =$$

$$= \mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\mathbf{Y}_{\boldsymbol{n,k}} - \mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\mathbf{Y}\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}} - \hat{\boldsymbol{\beta}}^{\mathbf{T}}_{\boldsymbol{OLS}}\mathbf{Y}^{\mathbf{T}}\mathbf{Y}_{\boldsymbol{n,k}} + \hat{\boldsymbol{\beta}}^{\mathbf{T}}_{\boldsymbol{OLS}}\mathbf{Y}^{\mathbf{T}}\mathbf{Y}\hat{\boldsymbol{\beta}}_{\boldsymbol{OLS}} =$$

Given the result obtained from equation 2.29,

$$SSE\ = \mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\mathbf{Y}_{\boldsymbol{n,k}} - \mathbf{Y}^{\mathbf{T}}_{\boldsymbol{n,k}}\mathbf{Y}\left(\mathbf{Y}^{\mathbf{T}}\mathbf{Y}\right)^{-1}\mathbf{Y}^{\mathbf{T}}\mathbf{Y}_{\boldsymbol{n,k}}. \tag{A.15}$$

□

A.7 Proof of equation 2.37

Consider that $\left[Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_n, \mathbf{Y}_{ex}\right)\right]^{\mathrm{T}}$ and $\left[Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_{n-k}, \mathbf{Y}_{ex}\right)\right]^{\mathrm{T}}$ are replaced by A and B, respectively, in the equations 2.34 and 2.35.

Proof of equation 2.37.

$$
\begin{aligned}
\gamma_{Y_n Y_{n-k} \backslash \mathbf{Y}_{ex}} &= COV\left(\left(Y_n - P_{\mathbf{Y}_{ex}}(Y_n)\right), \left(Y_{n-k} - P_{\mathbf{Y}_{ex}}(Y_{n-k})\right)\right) = \\
&= COV\left(\left(Y_n - A\mathbf{Y}_{ex}\right), \left(Y_{n-k} - B\mathbf{Y}_{ex}\right)\right) = \\
&= COV\left(Y_n, Y_{n-k}\right) - A \cdot COV\left(\mathbf{Y}_{ex}, Y_{n-k}\right) - B \cdot COV\left(Y_n, \mathbf{Y}_{ex}\right) \\
&+ A \cdot Var\left(\mathbf{Y}_{ex}\right) \cdot B^{\mathrm{T}} = COV\left(Y_n, Y_{n-k}\right) - \\
&- COV\left(Y_n, \mathbf{Y}_{ex}\right)^{\mathrm{T}} Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_{n-k}, \mathbf{Y}_{ex}\right) - \\
&- COV\left(Y_{n-k}, \mathbf{Y}_{ex}\right)^{\mathrm{T}} Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(\mathbf{Y}_{ex}, Y_n\right) + \\
&+ COV\left(Y_n, \mathbf{Y}_{ex}\right)^{\mathrm{T}} Var\left(\mathbf{Y}_{ex}\right)^{-1} Var\left(\mathbf{Y}_{ex}\right) Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_{n-k}, \mathbf{Y}_{ex}\right) = \\
&= COV\left(Y_n, Y_{n-k}\right) - COV\left(Y_n, \mathbf{Y}_{ex}\right)^{\mathrm{T}} Var\left(\mathbf{Y}_{ex}\right)^{-1} COV\left(Y_{n-k}, \mathbf{Y}_{ex}\right)
\end{aligned}
\tag{A.16}
$$

□

A.8 Proof of equation 2.38

Similarly to what was done in equation A.16, the variances of $(Y_n - P_{\mathbf{Y}_{ex}}(Y_n))$ and $(Y_{n-k} - P_{\mathbf{Y}_{ex}}(Y_{n-k}))$ can be evaluated and their results are:

Proof of equation 2.38.

$$Var\left(Y_n - P_{\mathbf{Y}_{ex}}(Y_n)\right) = Var\left(Y_n - A\mathbf{Y}_{ex}\right) = E\left[\left(Y_n - A\mathbf{Y}_{ex}\right)^2\right] - \left(E\left[Y_n\right] - AE\left[\mathbf{Y}_{ex}\right]\right)^2 =$$

$$= E\left[y_n^2\right] - 2AE\left[Y_n\mathbf{Y}_{ex}\right] + E\left[A\mathbf{Y}_{ex}\mathbf{Y}_{ex}^{\mathrm{T}}A^{\mathrm{T}}\right] - E\left[y_n\right]^2 - 2AE\left[y_n\right]E\left[\mathbf{Y}_{ex}\right] -$$

$$- AE\left[\mathbf{Y}_{ex}\right]E\left[\mathbf{Y}_{ex}\right]^{\mathrm{T}}A^{\mathrm{T}} = Var\left(Y_n\right) - 2A\cdot COV\left(Y_n, \mathbf{Y}_{ex}\right) + 2A\cdot Var\left(\mathbf{Y}_{ex}\right)A^{\mathrm{T}} =$$

$$= Var\left(Y_n\right) - 2COV\left(Y_n, \mathbf{Y}_{ex}\right)^{\mathrm{T}}Var\left(\mathbf{Y}_{ex}\right)^{-1}COV\left(Y_n, \mathbf{Y}_{ex}\right) +$$

$$+ COV\left(Y_n, \mathbf{Y}_{ex}\right)^{\mathrm{T}}Var\left(\mathbf{Y}_{ex}\right)^{-1}Var\left(\mathbf{Y}_{ex}\right)Var\left(\mathbf{Y}_{ex}\right)^{-1}COV\left(Y_n, \mathbf{Y}_{ex}\right) =$$

$$= Var\left(Y_n\right) - COV\left(Y_n, \mathbf{Y}_{ex}\right)^{\mathrm{T}}Var\left(\mathbf{Y}_{ex}\right)^{-1}COV\left(Y_n, \mathbf{Y}_{ex}\right), \tag{A.17}$$

and analogously,

$$Var\ \left(Y_{n-k} - P_{\mathbf{Y}_{ex}}(Y_{n-k})\right) =$$

$$= Var\left(Y_{n-k}\right) - \left[COV\left(Y_{n-k}, \mathbf{Y}_{ex}\right)\right]^T Var\left(\mathbf{Y}_{ex}\right)^{-1}COV\left(Y_{n-k}, \mathbf{Y}_{ex}\right). \tag{A.18}$$

□

A.9 Proof of equations 2.43, 2.44 and 2.45

Let $\{y_n\}$ be the random walk process starting with $y_0 = 0$ and defined by equation 2.41. Therefore, $E\left[\epsilon_t\right] = 0$, $Var\left(\epsilon_t\right) = \sigma_\epsilon^2$ and $COV\left(y_{t-1}, \epsilon_t\right) = 0$ since y_{t-1} and ϵ_t are independent. Also, $Var\left(y_1\right) = Var\left(\epsilon_1\right) = \sigma_\epsilon^2$ and $E\left[y_1\right] = \delta$ since $y_0 = 0$.

Proof of equation 2.43.

$$Var\left(y_t\right) = E\left[\left(\delta + y_{t-1} + \epsilon_t\right)^2\right] - \left(E\left[\delta + y_{t-1} + \epsilon_t\right]\right)^2 =$$

$$= E\left[\left(\delta\right)^2\right] + E\left[\left(y_{t-1}\right)^2\right] + E\left[\left(\epsilon_t\right)^2\right] + 2\delta E\left[y_{t-1}\right] + 2\delta E\left[\epsilon_t\right] + 2E\left[y_{t-1}\epsilon_t\right] -$$

$$- \left(E\left[\delta\right]\right)^2 - \left(E\left[y_{t-1}\right]\right)^2 - \left(E\left[\epsilon_t\right]\right)^2 - 2\delta E\left[y_{t-1}\right] - 2\delta E\left[\epsilon_t\right] - 2E\left[y_{t-1}\right]E\left[\epsilon_t\right] =$$

$$= Var(y_{t-1}) + \sigma_\epsilon^2 + 2COV\left(y_{t-1}, \epsilon_t\right) = Var(y_{t-1}) + \sigma_\epsilon^2.$$

Hence,

$$Var(y_2) + \cdots + Var(y_n) = Var(y_1) + \cdots + Var(y_{n-1}) + (n-1)\sigma_\epsilon^2 \Leftrightarrow$$

$$\Leftrightarrow Var(y_n) = \overbrace{Var(y_1)}^{=\sigma_\epsilon^2} + (n-1)\sigma_\epsilon^2 = n\sigma_\epsilon^2 \tag{A.19}$$

Considering the particular case of the random walk process defined in Figure 2.5, $n = t$ and $\sigma_\epsilon = 1$, so $Var(y_t) = t$.

□

Proof of equation 2.44.

$$E\left[y_t\right] = E\left[\overbrace{\delta}^{constant} + y_{t-1} + \epsilon_t\right] = \delta + E\left[y_{t-1}\right] + \overbrace{E\left[\epsilon_t\right]}^{=0} \Leftrightarrow$$

$$\Leftrightarrow E\left[y_2\right] + \cdots + E\left[y_n\right] = E\left[y_1\right] + \cdots + E\left[y_{n-1}\right] + (n-1)\delta \Leftrightarrow$$

$$\Leftrightarrow E\left[y_n\right] = n\delta. \tag{A.20}$$

□

In order to demonstrate equation 2.45 consider the random walk process given by equation 2.42. Given that $\{\epsilon_1, \epsilon_2, \cdots, \epsilon_n\}$ is a sequence of independent and identically distributed random variables , $E\left[\epsilon_i \epsilon_j\right] = 0, \forall i \neq j$.

Proof of equations 2.45.

$$\gamma_k = COV\left(y_n, y_{n-k}\right) = COV\left(\sum_{i=1}^{n}\epsilon_i, \sum_{i=1}^{n-k}\epsilon_i\right) =$$

$$= E\left[\sum_{i=1}^{n}\epsilon_i \cdot \sum_{i=1}^{n-k}\epsilon_i\right] - E\left[\sum_{i=1}^{n}\epsilon_i\right] E\left[\sum_{i=1}^{n-k}\epsilon_i\right] =$$

$$= E\left[\sum_{i=1}^{n}\epsilon_i \cdot \sum_{i=1}^{n-k}\epsilon_i\right] - \sum_{i=1}^{n}\overbrace{E\left[\epsilon_i\right]}^{=0}\sum_{i=1}^{n-k}\overbrace{E\left[\epsilon_i\right]}^{=0} = E\left[\sum_{i=1}^{n}\epsilon_i \cdot \sum_{i=1}^{n-k}\epsilon_i\right] =$$

$$= E\left[\left(\epsilon_1 + \cdot + \epsilon_{n-k}\right)\left(\epsilon_1 + \cdot + \epsilon_n\right)\right] = \sum_{i=1}^{n-k} E\left[\epsilon_i^2\right] + 2\sum_{i=1}^{n-k}\sum_{j=i+1}^{n}\overbrace{E\left[\epsilon_i\epsilon_j\right]}^{=0} = (n-k)\sigma_\epsilon^2. \tag{A.21}$$

□

A.10 Proof Means Inequality theorem

Let $\{x_1, x_2, \cdots, x_n\}$ be a set of positive real numbers and taking $w = w_1 + \cdots + w_n$, with $w_k \geq 0$ and $w > 0$.

Proof of Means Inequality theorem. According to Gao, Sitharam and Roitberg (2019), for concave functions, the Jensen Inequality applies:

$$f\left(\frac{\sum_{k=1}^{n} w_k x_k}{\sum_{k=1}^{n} w_k}\right) \geq \frac{\sum_{k=1}^{n} w_k f\left(x_k\right)}{\sum_{k=1}^{n} w_k}. \tag{A.22}$$

Given that $f :]0, +\infty] \to \Re$, such as $f(x) = \ln(x)$ is a concave function $\forall x \in D_f$:

$$\ln\left(\frac{\sum_{k=1}^{n} w_k x_k}{w}\right) \geq \frac{\sum_{k=1}^{n} w_k \ln\left(x_k\right)}{w} = \frac{\sum_{k=1}^{n} \ln\left(x_k^{w_k}\right)}{w} = \frac{\ln\left(\prod_{k=1}^{n} x_k^{w_k}\right)}{w} = \ln\left(\left(\prod_{k=1}^{n} x_k^{w_k}\right)^{\frac{1}{w}}\right). \tag{A.23}$$

Considering (A.23) and taking into account that $f(x)$ is a injective function,

$$\frac{\sum_{k=1}^{n} w_k x_k}{w} \geq \sqrt[w]{\prod_{k=1}^{n} x_k^{w_k}}. \tag{A.24}$$

□

A.11 Derivation of the mean confidence interval

Given that the n-element random sample $y_1, y_2, \cdots, y_n$ is taken from a normal distribution $y \sim N(\mu, \sigma^2)$, hence, $\overline{y} \sim N\left(\mu, \frac{\sigma^2}{n}\right)$.

Derivation of the mean confidence interval. The statistic $\left(\frac{\overline{y}-\mu}{\sigma/\sqrt{n}}\right) \sim Z$, however, observe that μ and σ are unknown, so it is necessary replace σ by s, which is the sample standard deviation, but as consequence, the distribution of the statistic changes to a t-student with n-1 degrees of freedom: $\left(\frac{\overline{y}-\mu}{s/\sqrt{n}}\right) \sim t_{n-1}$.

$$P\ \ (L_i \leq \mu \leq L_s) = P\left(-L_s \leq -\mu \leq -L_i\right) = P\left((\overline{y} - L_s) \leq (\overline{y} - \mu) \leq (\overline{y} - L_i)\right) =$$

$$= \ P\left(\frac{(\overline{y} - L_s)}{s/\sqrt{n}} \leq \overbrace{\frac{(\overline{y} - \mu)}{s/\sqrt{n}}}^{t_{n-1}} \leq \frac{(\overline{y} - L_i)}{s/\sqrt{n}}\right) \tag{A.25}$$

Given that t_{n-1} is symmetrical, let's take:

$$\frac{(\overline{y} - L_i)}{s/\sqrt{n}} = -\left|t_{\alpha/2;n-1}\right| \qquad \Leftrightarrow \qquad L_i = \overline{y} - \left|t_{\alpha/2;n-1}\right| \frac{s}{\sqrt{n}}$$

and

$$\frac{(\overline{y} - L_s)}{s/\sqrt{n}} = \left|t_{\alpha/2;n-1}\right| \qquad \Leftrightarrow \qquad L_s = \overline{y} + \left|t_{\alpha/2;n-1}\right| \frac{s}{\sqrt{n}}.$$

□

A.12 Derivation of the prediction confidence interval

Given that the n-element random sample $y_1, y_2, \cdots, y_n$ is taken from a normal distribution $y \sim N(\mu, \sigma^2)$, hence, $\overline{y} \sim N\left(\mu, \frac{\sigma^2}{n}\right)$.

Derivation of the prediction confidence interval. The statistic $\left(\frac{\overline{y}-\mu}{\sigma/\sqrt{n}}\right) \sim Z$, however, observe that μ and σ are unknown, so it is necessary replace σ by s, which is the sample standard deviation, but as consequence, the distribution of the statistic changes to a t-student with n-1 degrees of freedom: $\left(\frac{\overline{y}-\mu}{s/\sqrt{n}}\right) \sim t_{n-1}$. Consider that $\bar{y}$ is an estimator for y_{n+1}, so the estimation error is $e = y_{n+1} - \bar{y}$ with zero mean and variance $Var(e) = \left(1+\frac{1}{n}\right)\sigma^2$. As consequence, the statistic $\left(\frac{y_{n+1}-\bar{y}}{\sigma\sqrt{1+\frac{1}{n}}}\right) \sim Z$, and since σ is unknown, by replacing it to s, $\left(\frac{y_{n+1}-\bar{y}}{s\sqrt{1+\frac{1}{n}}}\right) \sim t_{n-1}$.

$$P(L_i \leq y_{n+1} \leq L_s) = P(L_i - \bar{y} \leq y_{n+1} - \bar{y} \leq L_s - \bar{y}) =$$

$$= P\left(\frac{L_i-\bar{y}}{s\sqrt{1+\frac{1}{n}}} \leq \frac{y_{n+1}-\bar{y}}{s\sqrt{1+\frac{1}{n}}} \leq \frac{L_s-\bar{y}}{s\sqrt{1+\frac{1}{n}}}\right) = P\left(\frac{L_i-\bar{y}}{s\sqrt{1+\frac{1}{n}}} \leq t_{n-1} \leq \frac{L_s-\bar{y}}{s\sqrt{1+\frac{1}{n}}}\right) \tag{A.26}$$

Given that t_{n-1} is symmetrical, let's take:

$$\frac{L_i-\bar{y}}{s\sqrt{1+\frac{1}{n}}} = -\left|t_{\alpha/2;n-1}\right| \qquad \Leftrightarrow \qquad L_i = \overline{y} - s\left|t_{\alpha/2;n-1}\right|\sqrt{1+\frac{1}{n}}$$

and

$$\frac{L_s-\bar{y}}{s\sqrt{1+\frac{1}{n}}} = \left|t_{\alpha/2;n-1}\right| \qquad \Leftrightarrow \qquad L_s = \overline{y} + s\left|t_{\alpha/2;n-1}\right|\sqrt{1+\frac{1}{n}}.$$

Hence,

$$IC\left(\mu;1-\alpha\right):\left]\bar{y}-\left|t_{\alpha/2;n-1}\right|s\sqrt{1+\frac{1}{n}};\bar{y}+\left|t_{\alpha/2;n-1}\right|s\sqrt{1+\frac{1}{n}}\right[. \tag{A.27}$$

□

A.13 Converting an AR(1) into a MA(∞)

Statement: Consider an stationary AR(1) model $y_t = c+\phi_1 y_{t-1}+\epsilon_t$. This model can be written as a MA(∞) model given by:

$$y_t = \frac{c}{1-\phi_1}+\sum_{i=0}^{\infty}\phi_1^i\epsilon_{t-i}. \tag{A.28}$$

Proof of the statement. Demonstration by Mathematical Induction.

Firstly we will demonstrate that it is possible to write an AR(1) model as the following *quasi*-MA(n) model, given by:

$$y_t = c\sum_{i=0}^{n}\phi_1^k+\sum_{i=0}^{n}\phi_1^i\epsilon_{t-i}+\phi_1^{n+1}y_{t-(n+1)} \tag{A.29}$$

Base of induction: the property is valid to $k = 2$.

$$\begin{aligned} y_t &= c+\phi_1 y_{t-1}+\epsilon_t = \\ &= c+\phi_1\left(c+\phi_1 y_{t-2}+\epsilon_{t-1}\right)+\epsilon_t = c\left(1+\phi_1\right)+\phi_1\epsilon_{t-1}+\epsilon_t+\phi_1^2 y_{t-2}. \end{aligned} \tag{A.30}$$

Inductive step: Admitting that the property is valid to $k = u$.

$$y_t = c\sum_{i=0}^{u}\phi_1^k+\sum_{i=0}^{u}\phi_1^i\epsilon_{t-i}+\phi_1^{u+1}y_{t-(u+1)} \tag{A.31}$$

Proofing that whether the property is valid to $k = u$, therefore it is valid to $k = u+1$.

$$
\begin{aligned}
y_t &= c\sum_{i=0}^{u}\phi_1^i + \sum_{i=0}^{u}\phi_1^i\epsilon_{t-i} + \phi_1^{u+1}y_{t-(u+1)} = \\
&= c\sum_{i=0}^{u}\phi_1^i + \sum_{i=0}^{u}\phi_1^i\epsilon_{t-i} + \phi_1^{u+1}\left(c + \phi_1 y_{t-(n+2)} + \epsilon_{t-(n+1)}\right) = \\
&= c\sum_{i=0}^{u}\phi_1^i + \sum_{i=0}^{u}\phi_1^i\epsilon_{t-i} + c\phi_1^{u+1} + \phi_1^{u+2}y_{t-(n+2)} + \phi_1^{u+1}\epsilon_{t-(n+1)} = \\
&= c\sum_{i=0}^{u+1}\phi_1^i + \sum_{i=0}^{u+1}\phi_1^i\epsilon_{t-i} + \phi_1^{u+2}y_{t-(u+2)}. \qquad \text{(A.32)}
\end{aligned}
$$

Since the property is valid to $k = 2$, see base of induction, and given that if it is valid to $k = u$, therefore it is valid to $k = u + 1$, by mathematical induction, the property is valid to $k = n$, $\forall n \in \mathbb{N}$. Thus,

$$y_t = c\sum_{i=0}^{n}\phi_1^i + \sum_{i=0}^{n}\phi_1^i\epsilon_{t-i} + \phi_1^{n+1}y_{t-(n+1)} \qquad \text{(A.33)}$$

Given that y_t is stationary, $|\phi_1| < 1$. Applying the limit of y_t, when n tends to ∞ we find:

$$\lim_{n\longrightarrow\infty} y_t = y_t = c\sum_{i=0}^{\infty}\phi_1^i + \sum_{i=0}^{\infty}\phi_1^i\epsilon_{t-i} + \lim_{n\longrightarrow\infty}\left(\phi_1^{n+1}y_{t-(n+1)}\right). \qquad \text{(A.34)}$$

Given that $y_{t-(n+1)}$ is finite, and $\lim\limits_{n\longrightarrow\infty}\left(\phi_1^{n+1}\right) = 0$, so, $\lim_{n\longrightarrow\infty}\left(\phi_1^{n+1}y_{t-(n+1)}\right) = 0$. Thus,

$$\lim_{n\longrightarrow\infty} y_t = c\sum_{i=0}^{\infty}\phi_1^i + \sum_{i=0}^{\infty}\phi_1^i\epsilon_{t-i}. \qquad \text{(A.35)}$$

which is a MA(∞) model.
But, since $|\phi_1| < 1$, the first term of the equation A.35 is an infinite geometric series, which converges to a finite value according to: $\sum\limits_{i=0}^{\infty}\phi_1^i = \dfrac{1}{1-\phi_1}$. Therefore,

$$y_t = \frac{c}{1-\phi_1} + \sum_{i=0}^{\infty}\phi_1^i\epsilon_{t-i}, \qquad \text{(A.36)}$$

and the statement is proofed.

□

A.14 Statistical Measures from a MA(2) model

Consider the MA(2) model, according to described in the equation 3.28, that is, $w_t = \epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2}$. Given that ϵ_{t-k} is a white noise term, $E\left[\epsilon_{t-k}\right] = 0$ and $Var\left[\epsilon_{t-k}\right] =$

$E\left[\epsilon_{t-k}^2\right] = \sigma^2$, for all $k \geq 0$. The expected value of w_t is

$$E\left[w_t\right] = E\left[\epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2}\right] = E\left[\epsilon_t\right] + \theta_1 E\left[\epsilon_{t-1}\right] + \theta_2 E\left[\epsilon_{t-2}\right] = 0. \tag{A.37}$$

Since the white noise terms are uncorrelated, $E\left[\epsilon_t\epsilon_{t-k}\right] = E\left[\epsilon_t\right]E\left[\epsilon_{t-k}\right] = 0$, for all $k \neq 0$. The variance of w_t, γ_0, is given by

$$\begin{aligned} Var\left[w_t\right] &= E\left[w_t^2\right] - \left(\overbrace{E\left[w_t\right]}^{=0}\right)^2 = E\left[\left(\epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2}\right)^2\right] = \\ &= E\left[\epsilon_t^2\right] + \theta_1^2 E\left[\epsilon_{t-1}^2\right] + \theta_2^2 E\left[\epsilon_{t-2}^2\right] + 2\theta_1 E\left[\epsilon_t\epsilon_{t-1}\right] + 2\theta_2 E\left[\epsilon_t\epsilon_{t-2}\right] + \\ &+ 2\theta_1\theta_2 E\left[\epsilon_{t-1}\epsilon_{t-2}\right] = \left(1 + \theta_1^2 + \theta_2^2\right)\sigma^2. \end{aligned} \tag{A.38}$$

The autocovariance γ_1 is given by

$$\begin{aligned} \gamma_1 &= E\left[w_t \cdot w_{t-1}\right] = E\left[\left(\epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2}\right)\cdot\left(\epsilon_{t-1} + \theta_1\epsilon_{t-2} + \theta_2\epsilon_{t-3}\right)\right] = \\ &= \overbrace{E\left[\epsilon_t\epsilon_{t-1}\right]}^{=0} + \theta_1\overbrace{E\left[\epsilon_t\epsilon_{t-2}\right]}^{=0} + \theta_2\overbrace{E\left[\epsilon_t\epsilon_{t-3}\right]}^{=0} + \theta_1 E\left[\epsilon_{t-1}^2\right] + \theta_1^2\overbrace{E\left[\epsilon_{t-1}\epsilon_{t-2}\right]}^{=0} + \\ &+ \theta_1\theta_2\overbrace{E\left[\epsilon_{t-1}\epsilon_{t-3}\right]}^{=0} + \theta_2\overbrace{E\left[\epsilon_{t-1}\epsilon_{t-2}\right]}^{=0} + \theta_1\theta_2 E\left[\epsilon_{t-2}^2\right] + \theta_2^2\overbrace{E\left[\epsilon_{t-2}\epsilon_{t-3}\right]}^{=0} = \\ &= \left(\theta_1 + \theta_1\theta_2\right)\sigma^2. \end{aligned} \tag{A.39}$$

The autocovariance γ_2 is given by

$$\begin{aligned} \gamma_2 &= E\left[w_t \cdot w_{t-2}\right] = E\left[\left(\epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2}\right)\cdot\left(\epsilon_{t-2} + \theta_1\epsilon_{t-3} + \theta_2\epsilon_{t-4}\right)\right] = \\ &= \overbrace{E\left[\epsilon_t\epsilon_{t-2}\right]}^{=0} + \theta_1\overbrace{E\left[\epsilon_t\epsilon_{t-3}\right]}^{=0} + \theta_2\overbrace{E\left[\epsilon_t\epsilon_{t-4}\right]}^{=0} + \theta_1\overbrace{E\left[\epsilon_{t-1}\epsilon_{t-2}\right]}^{=0} + \theta_1^2\overbrace{E\left[\epsilon_{t-1}\epsilon_{t-3}\right]}^{=0} + \\ &+ \theta_1\theta_2\overbrace{E\left[\epsilon_{t-1}\epsilon_{t-4}\right]}^{=0} + \theta_2 E\left[\epsilon_{t-2}^2\right] + \theta_1\theta_2\overbrace{E\left[\epsilon_{t-2}\epsilon_{t-3}\right]}^{=0} + \theta_2^2\overbrace{E\left[\epsilon_{t-2}\epsilon_{t-4}\right]}^{=0} = \\ &= \theta_2\sigma^2. \end{aligned} \tag{A.40}$$

The autocovariance γ_k, with $k \geq 3$ is given by

$$\begin{aligned} \gamma_k &= E\left[w_t \cdot w_{t-k}\right] = E\left[\left(\epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2}\right)\cdot\left(\epsilon_{t-k} + \theta_1\epsilon_{t-k-1} + \theta_2\epsilon_{t-k-2}\right)\right] = \\ &= \overbrace{E\left[\epsilon_t\epsilon_{t-k}\right]}^{=0} + \theta_1\overbrace{E\left[\epsilon_t\epsilon_{t-k-1}\right]}^{=0} + \theta_2\overbrace{E\left[\epsilon_t\epsilon_{t-k-2}\right]}^{=0} + \theta_1\overbrace{E\left[\epsilon_{t-1}\epsilon_{t-k}\right]}^{=0} + \theta_1^2\overbrace{E\left[\epsilon_{t-1}\epsilon_{t-k-1}\right]}^{=0} + \\ &+ \theta_1\theta_2\overbrace{E\left[\epsilon_{t-1}\epsilon_{t-k-2}\right]}^{=0} + \theta_2\overbrace{E\left[\epsilon_{t-2}\epsilon_{t-k}\right]}^{=0} + \theta_1\theta_2\overbrace{E\left[\epsilon_{t-2}\epsilon_{t-k-1}\right]}^{=0} + \theta_2^2\overbrace{E\left[\epsilon_{t-2}\epsilon_{t-k-2}\right]}^{=0} = 0. \end{aligned} \tag{A.41}$$

Appendix B

From time series to complex networks: method not used in this research

B.1 Horizontal Visibility Graph

Consider a time series, represented in a bar plot as shown in Figure B.1. Suppose that this time series will be mapped into a graph where each node is associated to a unique bar. Any two nodes in the graph are linked if the correspondent bars can be connected by a horizontal line without crossing any other intermediate bar (LUQUE *et al.*, 2010).

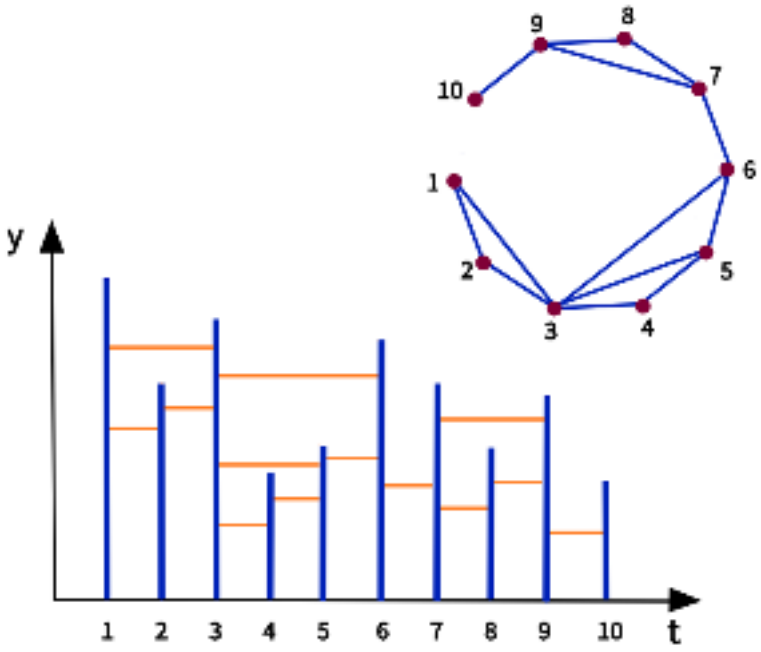


Figure B.1: Scheme for horizontal visibility graph methodology: A 10-elements time series is mapped into a graph where each node represents a bar, and each edge represents the connection between two bars. Source: Drawn by the author

In formal basis, two nodes v_i and v_k in the graph are connected only if, for each $t_i < t_j < t_k$, both $v_k < v_j$ and $v_k < v_i$ occur. The horizontal visibility graph leads to a natural graph-theoretical description of nonlinear systems with qualities in the spirit of symbolic dynamics (GAO *et al.*, 2016).

Appendix C

Proofs of the Statistical Results of MVA

C.1 Variance of the MVA Pre-estimator - Equation 4.8

Let $\hat{y}_{n+1|k}$ be the MVA pre-estimator for the next observation in the time series, y_{n+1}, considering only the influence of y_k. It is defined in equation 4.7.

Proof of the Variance of the MVA Pre-estimator.

$$Var\left(\hat{y}_{n+1|k}\right) =$$

$$= Var\left(y_n\right) + \left(\frac{\alpha}{T_k}\right)^2 Var\left(y_n - y_k\right) + \left(\frac{2\alpha}{T_k}\right) COV\left(y_n, (y_n - y_k)\right) =$$

$$= Var\left(y_n\right) + \left(\frac{\alpha}{T_k}\right)^2 \left[Var\left(y_n\right) + Var\left(y_k\right) - 2COV\left(y_n, y_k\right)\right] +$$

$$+ \left(\frac{2\alpha}{T_k}\right) \left[Var\left(y_n\right) - COV\left(y_n, y_k\right)\right] =$$

$$= Var\left(y_n\right)\left(1 + \frac{\alpha}{T_k}\right)^2 + Var\left(y_k\right)\left(\frac{\alpha}{T_k}\right)^2 - \frac{2\alpha}{T_k}\left(1 + \frac{\alpha}{T_k}\right) COV\left(y_n, y_k\right). \tag{C.1}$$

□

C.2 Covariance between two MVA Pre-estimators - Equation 4.9

Let $\hat{y}_{n+1|k}$ and $\hat{y}_{n+1|j}$ be two MVA pre-estimators for y_{n+1}, for $k \neq j$.

Proof of the Covariance between two MVA Pre-estimators.

$$
\begin{aligned}
&COV\left(\hat{y}_{n+1|k}, \hat{y}_{n+1|j}\right) = \\
&= E\left[\left(y_n + \frac{\alpha}{T_k}\left(y_n - y_k\right)\right)\left(y_n + \frac{\alpha}{T_j}\left(y_n - y_j\right)\right)\right] - \\
&- E\left[\left(y_n + \frac{\alpha}{T_k}\left(y_n - y_k\right)\right)\right] E\left[\left(y_n + \frac{\alpha}{T_j}\left(y_n - y_j\right)\right)\right] = \\
&= \left(E\left[y_n^2\right] - \left(E\left[y_n\right]\right)^2\right) + \frac{\alpha}{T_j}\left(E\left[y_n^2\right] - E\left[y_n y_j\right]\right) + \frac{\alpha}{T_k}\left(E\left[y_n^2\right] - E\left[y_n y_k\right]\right) + \\
&+ \frac{\alpha^2}{T_k T_j}\left(E\left[y_n^2\right] - E\left[y_n y_k\right] - E\left[y_n y_j\right] + E\left[y_k y_j\right]\right) - \\
&- \frac{\alpha}{T_j}\left(\left(E\left[y_n\right]\right)^2 - E\left[y_n\right] E\left[y_j\right]\right) - \frac{\alpha}{T_k}\left(\left(E\left[y_n\right]\right)^2 - E\left[y_n\right] E\left[y_k\right]\right) - \\
&- \frac{\alpha^2}{T_k T_j}\left(\left(E\left[y_n\right]\right)^2 - E\left[y_n\right] E\left[y_k\right] - E\left[y_n\right] E\left[y_j\right]\right) - \frac{\alpha^2}{T_k T_j}\left(E\left[y_k\right] E\left[y_j\right]\right) = \\
&= Var\left(y_n\right)\left(1 + \frac{\alpha}{T_k}\right)\left(1 + \frac{\alpha}{T_j}\right) - \frac{\alpha}{T_j}\left(1 + \frac{\alpha}{T_k}\right) COV\left(y_n, y_j\right) - \\
&- \frac{\alpha}{T_k}\left(1 + \frac{\alpha}{T_j}\right) COV\left(y_n, y_k\right) + \frac{\alpha^2}{T_k T_j} COV\left(y_k, y_j\right). \qquad \text{(C.2)}
\end{aligned}
$$

□

C.3 Derivation of the best K for the MVA model

Let $Var\left(\hat{y}_{n+1}\right)$ be the variance of the MVA estimator according to presented in the equation (4.11). The value of K that minimizes such variance can be determined by calculating $\frac{\partial SSE\left(\hat{y}_{(n+1)}\right)}{\partial K} = 0$. Suppose that v_i is the order of the element in the $i\text{-}th$ window correspondent to the maximum pre-estimator. Just for a better organization of the text, define V as the order of the last observation in the training set, $a = w + 1$, $T_{v_i} = t_{i-1} - t_{v_i}$, and ρ_{i-1,v_i} as the autocorrelation between y_{i-1} and y_{v_i}, where y_{v_i} is the maximum of the past observation with the highest level of similarity with y_{i-1}.

Proof of the equation 4.15.

$$SSE\left(\hat{y}_{n+1}\right) = \sum_{k=a}^{V}\left(y_i - \hat{y}_i\right)^2 = \sum_{k=a}^{V}\left(y_i - y_{i-1} - \left(\gamma_{i-1,v_i} - K\right)\left(\frac{y_{i-1} - y_{v_i}}{t_{i-1} - t_{v_i}}\right)\right)^2 =$$
$$= \sum_{k=a}^{V}\left(y_i - y_{i-1}\right)^2 - 2\sum_{k=a}^{V}\left(\gamma_{i-1,v_i} - K\right)\left(y_i - y_{i-1}\right)\left(\frac{y_{i-1} - y_{v_i}}{T_{v_i}}\right) +$$
$$+ \sum_{k=a}^{V}\left(\gamma_{i-1,v_i} - K\right)^2\left(\frac{y_{i-1} - y_{v_i}}{T_{v_i}}\right)^2 \tag{C.3}$$

$$\frac{\partial SSE\left(\hat{y}_{n+1}\right)}{\partial K} =$$
$$2\sum_{k=a}^{V}\left(y_i - y_{i-1}\right)\left(\frac{y_{i-1} - y_{v_i}}{T_{v_i}}\right) - 2\sum_{k=a}^{V}\left(\gamma_{i-1,v_i} - K\right)\left(\frac{y_{i-1} - y_{v_i}}{T_{v_i}}\right)^2 \tag{C.4}$$

Equaling (C.4) to zero and isolating K, comes:

$$\hat{K} = \frac{\sum_{i=a}^{V}\left(\left(\frac{y_{i-1} - y_{v_i}}{T_{v_i}}\right)^2 \rho_{i-1,v_i} - \left(\frac{y_{i-1} - y_{v_i}}{T_{v_i}}\right)\left(y_i - y_{i-1}\right)\right)}{\sum_{i=a}^{V}\left(\frac{y_{i-1} - y_{v_i}}{T_{v_i}}\right)^2} \tag{C.5}$$

□

C.4 Derivation of the estimator for β in the HAM model

Proof of the equation 4.28.

$$SSE_{HAM} = \sum_{i=1}^{n-w-1}\left(y_i - \hat{y}_{HAM,i}\right)^2 = \sum_{i=1}^{n-w-1}\left(y_i - \beta\hat{y}_{MV,i} - \left(1-\beta\right)\hat{y}_{AA,i}\right)^2 =$$
$$= \sum_{i=1}^{n-w-1} y_i^2 + \beta^2\sum_{i=1}^{n-w-1}\hat{y}_{MV,i}^2 + \left(1-\beta\right)^2\sum_{i=1}^{n-w-1}\hat{y}_{AA,i}^2 - 2\beta\sum_{i=1}^{n-w-1} y_i\hat{y}_{MV,i} \quad -$$
$$- 2\left(1-\beta\right)\sum_{i=1}^{n-w-1} y_i\hat{y}_{AA,i} + 2\beta\left(1-\beta\right)\sum_{i=1}^{n-w-1}\hat{y}_{MV,i}\hat{y}_{AA,i}. \tag{C.6}$$

$$\frac{\partial SSE_{HAM}}{\partial \beta} = 2\beta \sum_{i=1}^{n-w-1} \hat{y}_{MV,i}^2 + 2(\beta - 1) \sum_{i=1}^{n-w-1} \hat{y}_{AA,i}^2 - 2 \sum_{i=1}^{n-w-1} y_i \hat{y}_{MV,i} \quad +$$
$$+ 2 \sum_{i=1}^{n-w-1} y_i \hat{y}_{AA,i} + 2 \sum_{i=1}^{n-w-1} \hat{y}_{AA,i} \hat{y}_{MV,i} - 4\beta \sum_{i=1}^{n-w-1} \hat{y}_{AA,i} \hat{y}_{MV,i}. \tag{C.7}$$

Equaling (C.7) to zero, β turns into $\hat{\beta}$. Isolating $\hat{\beta}$ comes:

$$\hat{\beta} = \frac{\sum_{i=1}^{n-w-1} (\hat{y}_{AA,i} - y_i)(\hat{y}_{AA,i} - \hat{y}_{MV,i})}{\sum_{i=1}^{n-w-1} (\hat{y}_{AA,i} - \hat{y}_{MV,i})^2} \tag{C.8}$$

□

Appendix D

Important Figures

D.1 Plot of the Air Passengers Time Series

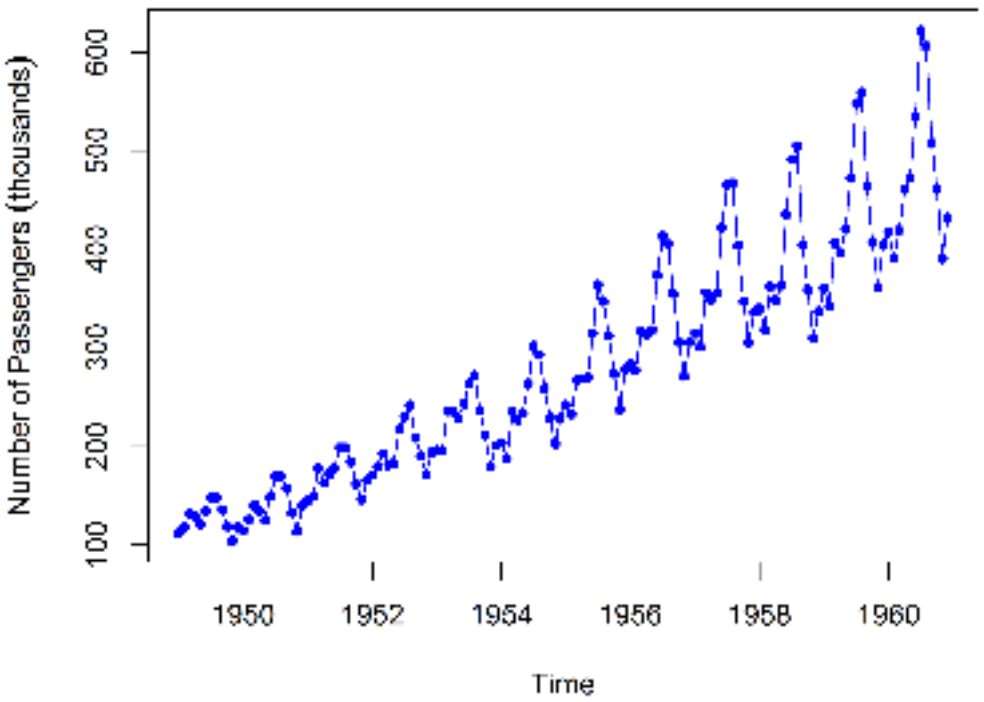


Figure D.1: Time series of the monthly number of passengers for international airlines, in thousands of people, from 1949 to 1960. Figure Source: Drawn by the author.

D.2 Plot of the Lynx Time Series

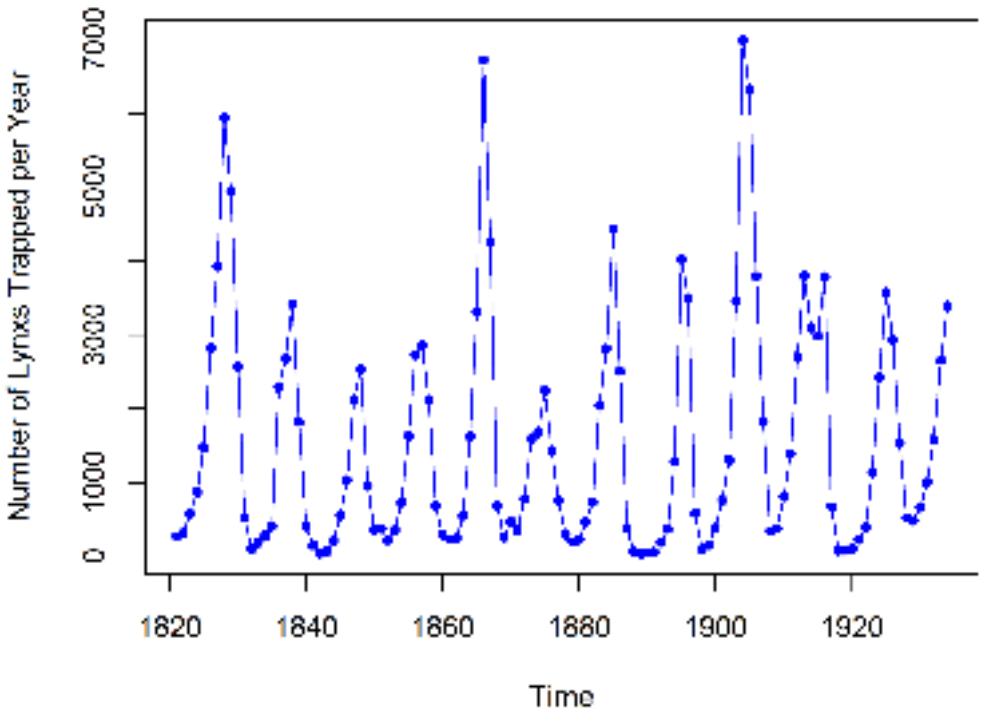


Figure D.2: Time series of the annual numbers of lynx trapped in Canada from 1821 to 1934. Figure Source: Drawn by the author.

D.3 Autocorrelation Function of the Lynx Time Series

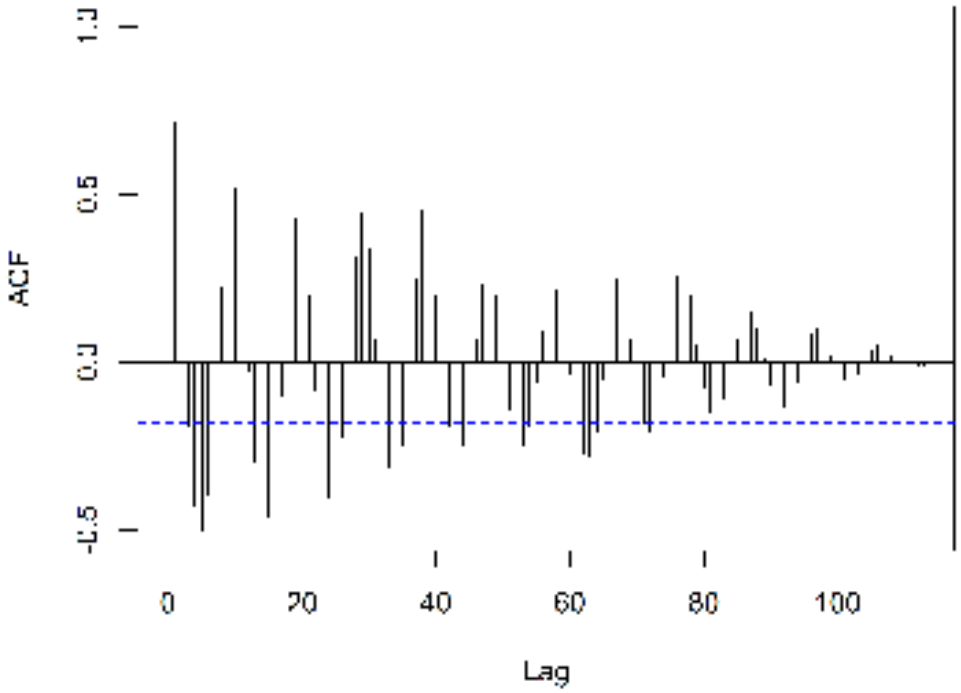


Figure D.3: Autocorrelation Function of the Lynx Time series. Figure Source: Drawn by the author.

D.4 Plot of the IBOVESPA Time Series

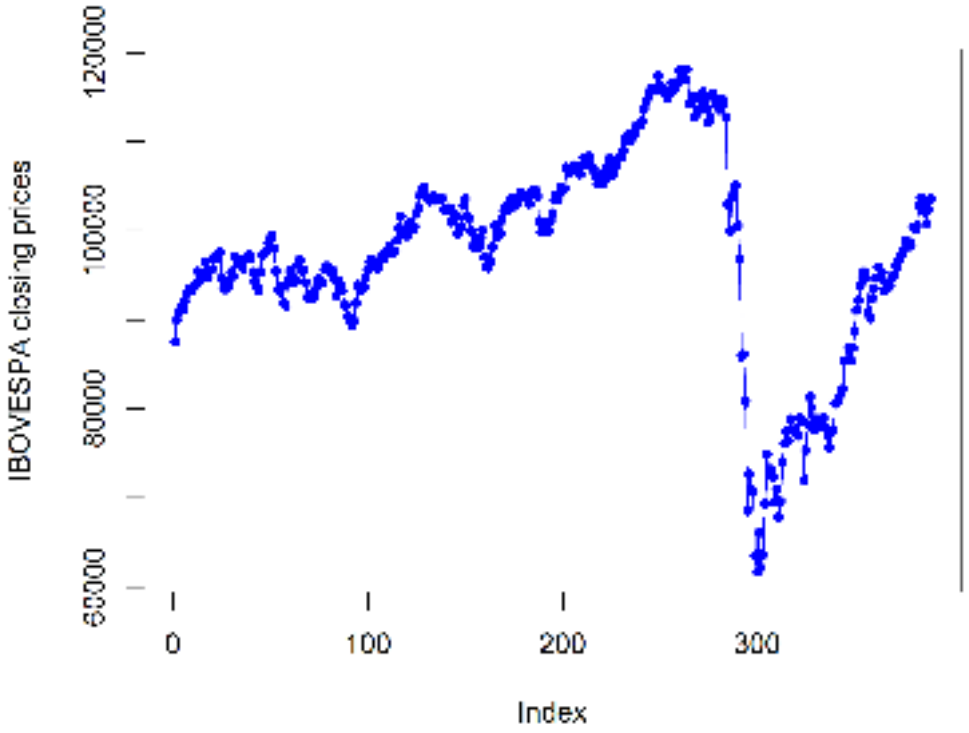


Figure D.4: Daily IBOVESPA stock closing prices from 2019/01/01 to 2020/07/28. Data Source: Yahoo Finance. Figure Source: Drawn by the author.

D.5 Autocorrelation Function of the IBOVESPA Time Series

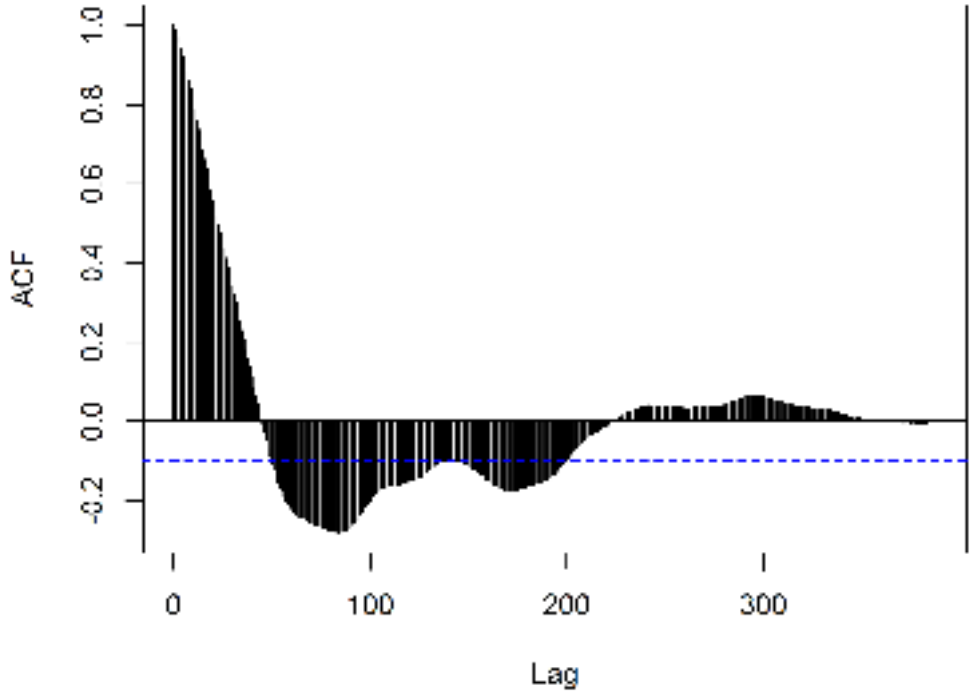


Figure D.5: Autocorrelation Function of the IBOVESPA Time series.
Figure Source: Drawn by the author.

D.6 Plot of the NHtemp Time Series

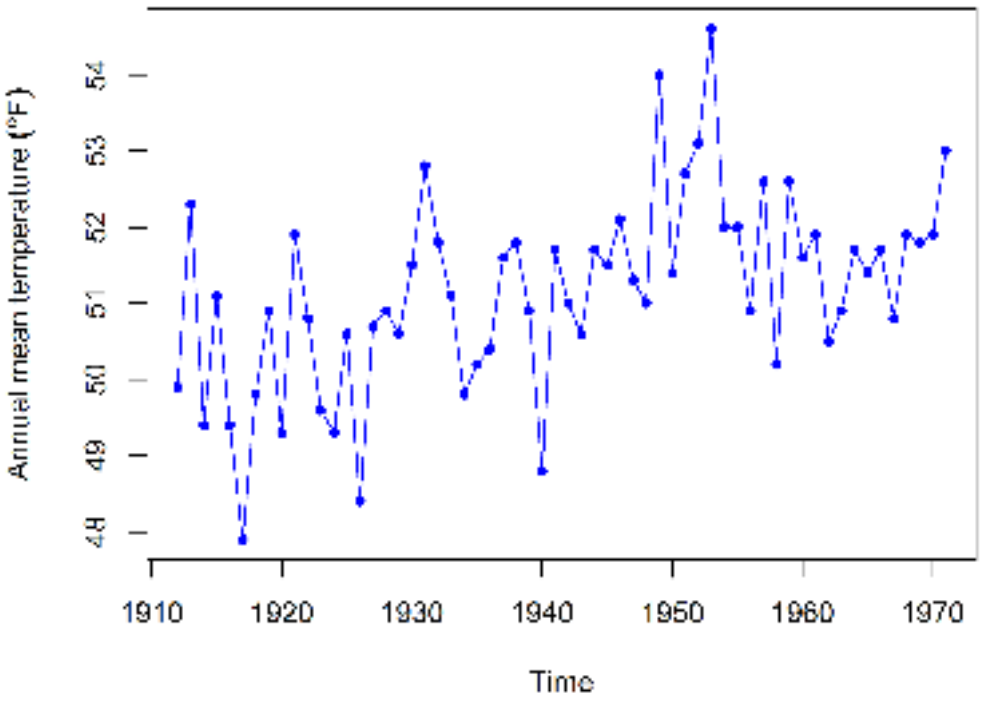


Figure D.6: The mean annual temperature, in degrees Fahrenheit, in New Haven, Connecticut, from 1912 to 1971. Figure Source: Drawn by the author.

D.7 Autocorrelation Function of the NHtemp Time Series

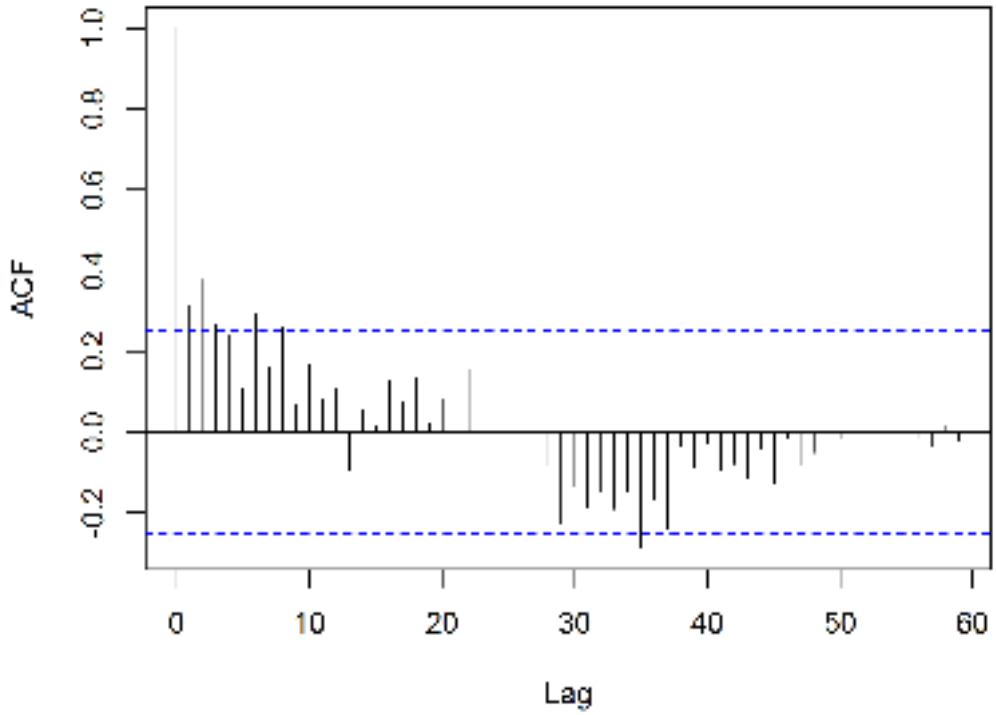


Figure D.7: Autocorrelation Function of the NHtemp Time series. Figure Source: Drawn by the author.

D.8 Plot of the Recurrence-Based Time Series

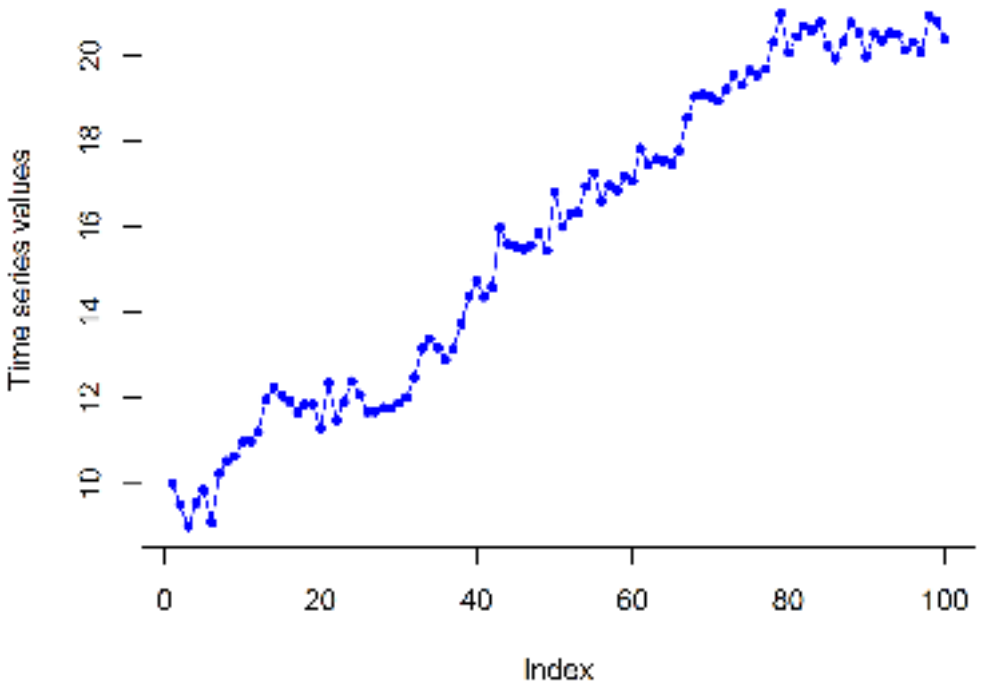


Figure D.8: The recursive time series according to described in the section 5.1.5. Figure Source: Drawn by the author.

D.9 Autocorrelation Function of the Recurrence-Based Time Series

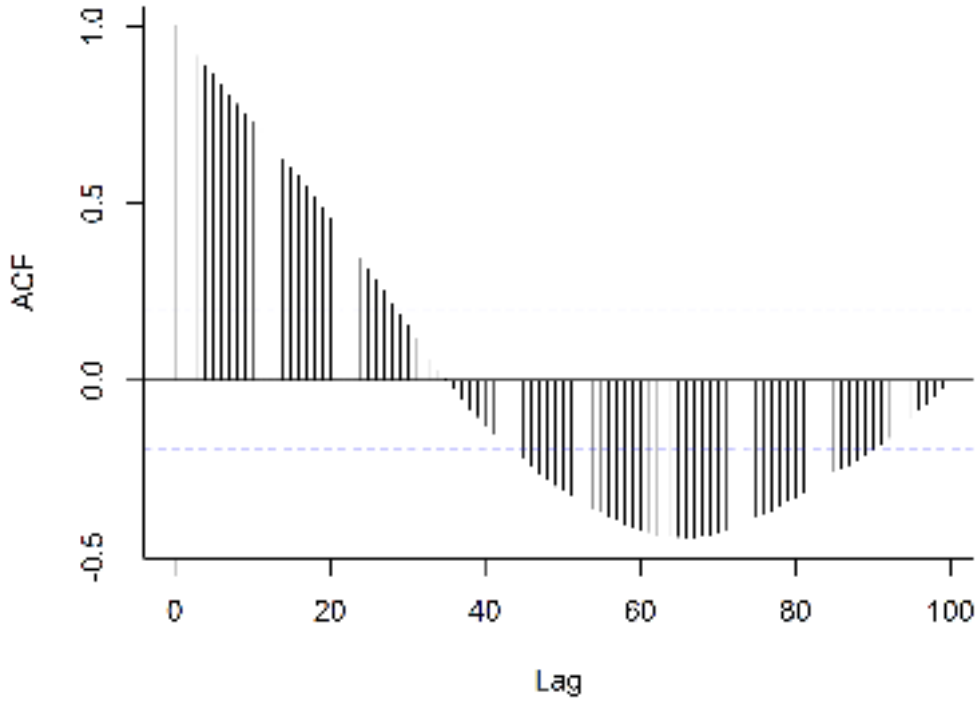


Figure D.9: Autocorrelation Function of the Recurrence-Based Time series. Figure Source: Drawn by the author.

Appendix E

R Codes Used in this Work

In this section we present the codes used to generate the accuracy comparisons between some of the time series forecasting methods used in this work. They are: Maximum Visibility Approach, Hybrid Multiple Means Approach, Dynamic ARIMA, Mao-Xiao Approach, Naive, Support Vector Regression, Hybrid ANN-ETS and the Hybrid ANN-ARIMA.

E.1 Maximum Visibility Approach R Code

In this section we present the R code used to calculate MAE, MAPE and RMSE from the one-step-ahead forecasting process using the Maximum Visibility Approach. The time series is split into training and test set. The first w observations compose the first window. The values of $\hat{y}_{w+1}, \cdots, \hat{y}_{w+t}$ are calculated and from this values MAE, MAPE and RMSE are calculated. We choose the values of K and J related to the minimum value of $\sqrt{MAE \cdot RMSE}$. Next, this pair of K^* and J^* are used to calculate the estimates of the test set: $\hat{y}_{w+t+1}, \cdots, \hat{y}_n$. These last set of estimates are used to calculate the MAE, MAPE and RMSE related to the test set.

```
library(e1071)
library(seastests)
library(urca)
library(zoom)
library(dgof)
library(quantmod)
library(ggplot2)
library(xts)
library(igraph)
library(corrplot)
library(forecast)
library(lmtest)
library(tseries)
library(PMCMR)
library(astsa)
library(tictoc)
library(xtable)
library(stargazer)
```

```
# This code was used to calculate MAE, MAPE and RMSE from
# the forecasting process of a given time series using
# the Maximum Visibility Approach.

for(ON in 1:1){

metodo = "dice"

# Which Time Series - Choose ts
# from 1, 2, 3, 4 or 5.
ts=2
if(ts==1){
        x1 = read.csv(file = "AirPassengers.csv")
        w=40
        x1=x1[,1]
        factor=0.7
        J=0.01
        K=0.89
}
if(ts==2){
        x1 = read.csv(file = "lynx.csv")
        w=77
        x1=x1[,1]
        factor=0.7
        J=0.001
        K=0.633
}
if(ts==3){
        x1 = read.csv(file = "IBOV.csv")
        w=200
        x1=x1[,1]
        factor=0.5
        J=0.005
        K=0.7
}
if(ts==4){
        x1 = read.csv(file = "nhtemp.csv")
```

```
        w=10
        x1=x1[,1]
        factor=0.8
        J=100
        K=1
}
if(ts==5){
        x1 = read.csv(file = "RBTS.csv")
        w=30
        x1=x1[,1]
        factor=0.75
        J=0.9
        K=2.5
}


n=length(x1)

D=matrix(rep(0,w*(n-w)),ncol=w)

for(u in 1:(n-w)){
        for(v in 1:w ){
                D[u,v]=x1[u+v-1]
        }
}

#Size of the training set
trai=ceiling(factor*(n-w))


for(Q in 1:1){
        for(qa in 1:1 ){
                yMV = c()
                yMVfor = c()
                ymaxY1 = c()
                ALFA= matrix(rep(0,(w)*trai),
                ncol=(w), byrow=TRUE)
                facNL=c()
        }
```

```
for(b in 1:trai){
        y = D[b,]
        x = b:(w+b-1)
        a = w
        y1n = y[w]
        tn = x[w]

        if(b==1){

                facNL[b]=0
        # Non-linear term added to the
        # pre-estimators and compose MVA forecast.
        }

        if(b>1){
        facNL[b]=-((yMV[(b-1)]-y1n))*
        (exp(-J*K*abs(yMV[(b-1)]-y1n)))
        # Non-linear term added to the
        # pre-estimators and compose MVA forecast.
        }

        # Constructing the Complex Network

        # Adjacency matrix

        MAdj1  = matrix(0,nrow = a, ncol = a)

        for(i in 1:(a-2)){
                MAdj1[i,(i+1)]=1

                m=(y[(i+1)]-y[i])/(x[(i+1)]-x[i])

                for(j in (i+2):a){

                        ytj=y[i]+m*(x[j]-x[i])

                        if(y[j]>=ytj){
                                m=(y[j]-y[i])/(x[j]-x[i])
```

```
                    MAdj1[i,j]=1
                }
        }
}

MAdj1[(i+1),(i+2)]=1

M1 = MAdj1 + t(MAdj1)

g1 = graph_from_adjacency_matrix(M1,
mode=c('undirected'))

Sim1Dice = similarity(g1,method = metodo)
#method = "dice"
#invlogweighted

Sim1 = Sim1Dice-1*diag(rep(1,a))

# Obtaining the vector of similarities
# between the last vertex and the others

SimComp1 = Sim1[a,1:(a-1)]

Y1 = as.vector(rep(0,(a-1)))

uu=which(SimComp1==max(SimComp1))
AC=acf(y,lag.max =length(y),plot=FALSE )
AU=rep(0,length(AC$acf))
for(u in 1:length(AC$acf)){
        AU[u]=AC$acf[(length(AC$acf)-u+1)]
}
ALFA[b,]=AU

for(u in uu){

        alfa=AU[u]

        if(b==1){
                Y1[u]= y1n + (-K+alfa)*((y1n
```

```
                        -y[u])/(tn - x[u]))
                }
                if(b>1){
                        Y1[u]= (y1n + (-K+alfa)*((y1n
                        -y[u])/(tn - x[u])))
                }

        }

        # Calculating MVA  #

        indexY1=which(SimComp1==max(SimComp1))

        yMV[b] = max(Y1[indexY1])+(facNL[b])

        real=x1[(w+1):(w+b)]

        erroMV=real-yMV

        SumErro=sum(erroMV^2)

        # Calculating the Training MAE, MAPE and RMSE
        # The values of K and J must be choosen
        # based on the smallest errors

        MAEMV=mean(abs(erroMV))
        MAPEMV=mean(abs(erroMV)/real)*100
        RMSEMV=sqrt(mean(erroMV^2))

        ########################################

        } # End of the for in b

}


print(c('MAE-MVA:', MAEMV,K,J))
print(c('RMSE-MVA:', RMSEMV,K,J))
print(c('Index:', sqrt(RMSEMV*MAEMV)))
```

```
# We used this index to choose K and J.
# The chosen parameters are related to
# the smallest value of this index.

print(c('Forecast starts',K))

yMVtrai=yMV

pred=n-w-trai ## Size of the
# test set

for(qa in 1:1 ){
        yMV = c()
        ALFA= matrix(rep(0,(w)*pred),ncol=(w),
        byrow=TRUE)
        facNL=c()
}
for(b in 1:pred){
        y = D[trai+b,]
        x = (trai+b):(trai+b+w-1)
        a = w
        y1n = y[w]
        tn = x[w]

        if(b==1){
                facNL[b]=-(yMVtrai[(length(yMVtrai))]-
                y[(w-1)])*(exp(-J*abs(yMVtrai[(length(yMVtrai))]
                -y[(w-1)])))
        }
        if(b>1){

                facNL[b]=-(yMVfor[(b-1)]-y1n)*
                (exp(-J*K*abs(yMVfor[(b-1)]-y1n)))

        }

        # Constructing Network #
        # Adjacency Matrix
```

```
MAdj1  = matrix(0,nrow = a, ncol = a)
for(i in 1:(a-2)){
        MAdj1[i,(i+1)]=1

        m=(y[(i+1)]-y[i])/(x[(i+1)]-x[i])

        for(j in (i+2):a){

                ytj=y[i]+m*(x[j]-x[i])

                if(y[j]>=ytj){
                        m=(y[j]-y[i])/(x[j]-x[i])
                        MAdj1[i,j]=1
                }
        }
}

MAdj1[(i+1),(i+2)]=1

M1 = MAdj1 + t(MAdj1)

grafo1 = graph_from_adjacency_matrix(M1,
mode=c('undirected'))

Sim1Dice = similarity(grafo1,method = metodo)

Sim1 = Sim1Dice-1*diag(rep(1,a))

# Obtaining the vector of similarities
# between the last vertex and the others

SimComp1 = Sim1[a,1:(a-1)]

Y1 = as.vector(rep(0,(a-1)))

uu=which(SimComp1==max(SimComp1))
AC=acf(y,lag.max =length(y),plot=FALSE )
AU=rep(0,length(AC$acf))
```

```
for(u in 1:length(AC$acf)){
        AU[u]=AC$acf[(length(AC$acf)-u+1)]
}
ALFA[b,]=AU

for(u in uu){

        alfa=AU[u]

        if(b==1){
                Y1[u]= y1n + (-K+alfa)*((y1n
                - y[u])/(tn - x[u]))
        }
        if(b>1){
                Y1[u]= (y1n + (-K+alfa)*((y1n
                - y[u])/(tn - x[u])))
        }

}

# Calculating MVA #

indexY1=which(SimComp1==max(SimComp1))

yMVfor[b] = max(Y1[indexY1])+(facNL[b])

real=x1[(trai+w+1):(trai+w+b)]

erroMV=real-yMVfor

MAEMVfor=mean(abs(erroMV))
MAPEMVfor=mean(abs(erroMV)/real)*100
RMSEMVfor=sqrt(mean(erroMV^2))

#####################################

} # End of the for in b

print(c(MAEMVfor,b))
```

```
        print(c('MAE: MV',MAEMVfor))
        print(c('MAPE: MV',MAPEMVfor))
        print(c('RMSE: MV',RMSEMVfor))


        ylim = c(min(yMVfor,real), max(yMVfor,real))
        ylim = c(min(yMVfor,real), max(yMVfor,real))
        xlim = c(1, length(yMVfor))
        plot(real, col="blue", ylim=ylim,
        xlim=xlim, type='l')
        points(real,type='o', pch=20,
        cex = 0.85, col='blue')
        par(new=TRUE)
        plot(yMVfor, col="magenta", ylim=ylim,
        xlim=xlim, type='l')
        points(yMVfor,type='o', pch=20,
        cex = 0.85, col='magenta')

} # End of the for in ON
```

E.2 Hybrid Means of Multiple Approaches R Code

In this section we present the R code used to provide the one-step-ahead forecasting using the Hybrid Means of Multiple Approaches. Firstly we use the training set to search the value of β which gives the smallest sum of squared error, fore each type of mean. Next, the algorithm produces the multiple values of β, according described in the equations (4.25) and (4.26), and apply this vector in the four types of means. In the end of this forecasting process, done in the training set, the sum of squared error is evaluated. Lastly, we use the result of the equation (4.28) to calculate the HMMA in the training set and next the SSE is evaluated. The minimum value of SSE reveals what type of β and what type of mean will be used in the test set.

```
library(e1071)
library(seastests)
library(urca)
library(zoom) # Invoke the Library
library(dgof)
library(quantmod)
library(ggplot2)
```

```
library(xts)
library(igraph)
library(corrplot)
library(forecast)
library(lmtest)
library(tseries)
library(PMCMR)
library(astsa)
library(tictoc)
library(xtable)
library(stargazer)
################################


for(QAAA in 1:1){
tic()
start=0.03
end=0.99
step=0.025
ts=2 ### Escolher ts entre 1, 2, 3, 4, 5, 6, 7 e 8.

Indexmin=c()

# Importing the datasets used to create HMMA

if(ts==1){
MV = read.csv(file = "yMVforAIR.csv")
MVt = read.csv(file = "yMVtraiAIR.csv")
Auxt = read.csv(file="SVMtraiAIR.csv")
Aux = read.csv(file="SVMforAIR.csv")
Aux=Aux[,1]
Auxt=Auxt[,1]
MV=MV[,1]
MVt=MVt[,1]
realt=read.csv(file = "realtraiAIR.csv")
real=read.csv(file = "realforAIR.csv")
real=real[,1]
realt=realt[,1]
}
```

```
if(ts==2){
MV = read.csv(file = "yMVforLYNX.csv")
MVt = read.csv(file = "yMVtraiLYNX.csv")
Auxt = read.csv(file="yANNetstraiLYNX.csv")
Aux = read.csv(file="yANNetsforLYNX.csv")
Aux=Aux[,1]
Auxt=Auxt[,1]
MV=MV[,1]
MVt=MVt[,1]
realt=read.csv(file = "realtraiLYNX.csv")
real=read.csv(file = "realforLYNX.csv")
real=real[,1]
realt=realt[,1]
realt=realt[-1]
realt=realt[-1]
MVt=MVt[-1]
MVt=MVt[-1]
}
if(ts==3){
MV = read.csv(file = "yMVforIBOV.csv")
MVt = read.csv(file = "yMVtraiIBOV.csv")
Auxt = read.csv(file="SVMtraiIBOV.csv")
Aux = read.csv(file="SVMforIBOV.csv")
Aux=Aux[,1]
Auxt=Auxt[,1]
MV=MV[,1]
MVt=MVt[,1]
realt=read.csv(file = "realtraiIBOV.csv")
real=read.csv(file = "realforIBOV.csv")
real=real[,1]
realt=realt[,1]
#MVt=MVt[-1]
#realt=realt[-1]

}
if(ts==4){
MV = read.csv(file = "yMVforNHT.csv")
MVt = read.csv(file = "yMVtraiNHT.csv")
Auxt = read.csv(file="SVMtraiNHT.csv")
```

```
Aux = read.csv(file="SVMforNHT.csv")
Aux=Aux[,1]
Auxt=Auxt[,1]
MV=MV[,1]
MVt=MVt[,1]
realt=read.csv(file = "realtraiNHT.csv")
real=read.csv(file = "realforNHT.csv")
real=real[,1]
realt=realt[,1]

}
if(ts==5){
MV = read.csv(file = "yMVforRBTS.csv")
MVt = read.csv(file = "yMVtraiRBTS.csv")
Auxt = read.csv(file="SVMtraiRBTS.csv")
Aux = read.csv(file="SVMforRBTS.csv")
Aux=Aux[,1]
Auxt=Auxt[,1]
MV=MV[,1]
MVt=MVt[,1]
realt=read.csv(file = "realtraiRBTS.csv")
real=read.csv(file = "realforRBTS.csv")
real=real[,1]
realt=realt[,1]
}

for(QAA in 1:5){
# In this for, the code search the best beta
# to be used in the HMMA estimation.

Beta=c()
KK=seq(start,end,step)
maeH=matrix(rep(0,4*length(KK)),ncol=4)
SSE=matrix(rep(0,4*length(KK)),ncol=4)

for(QA in 1:length(KK)){

beta=KK[QA]
```

```
##############################################

# Weighted Quadratic Mean

HQMt=sqrt(beta*MVt^2+(1-beta)*Auxt^2)

# Weighted Arithmetic Mean

HGMt=c()
for(u in 1:length(MVt)){
if(MVt[u]>0 && Auxt[u]>0){
        HGMt[u]=(MVt[u]^beta)*(Auxt[u]^(1-beta))

}
if(MVt[u]<0 || Auxt[u]<0){
        HGMt[u]=-(abs(MVt[u])^beta)*(abs(Auxt[u])^(1-beta))
}
}

# Weighted Arithmetic Mean

HAMt=beta*MVt+(1-beta)*Auxt

# Weighted Harmonic Mean

HHMt=Auxt*MVt/(beta*Auxt+(1-beta)*MVt)

# Evaluating the residuals

eQ=realt-HQMt
eA=realt-HAMt
eG=realt-HGMt
eH=realt-HHMt

# Evaluating sum of squared errors

SumEQ=sum(eQ^2)
SumEA=sum(eA^2)
SumEG=sum(eG^2)
```

```
SumEH=sum(eH^2)


# Evaluating MAE

MAEQ=mean(abs(eQ))
MAEA=mean(abs(eA))
MAEG=mean(abs(eG))
MAEH=mean(abs(eH))

# Evaluating RMSE

RMSEQ=sqrt(mean(eQ^2))
RMSEA=sqrt(mean(eA^2))
RMSEG=sqrt(mean(eG^2))
RMSEH=sqrt(mean(eH^2))

maeH[QA,1]=MAEQ
maeH[QA,2]=MAEA
maeH[QA,3]=MAEG
maeH[QA,4]=MAEH

Beta[QA]=beta

SSE[QA,1]=SumEQ
SSE[QA,2]=SumEA
SSE[QA,3]=SumEG
SSE[QA,4]=SumEH

}


if(QAA<=3){

IminSSE=which(SSE==min(SSE))

lin=IminSSE %% length(KK)
if(lin==0){
lin=length(KK)
```

```
}
if(IminSSE==length(KK)*4){
        col=4
}
if(IminSSE != length(KK)*4){col=IminSSE %/% length(KK)+1
}
beta=Beta[lin]

# Refining the search set for beta
start=beta-step
end=beta+step
step=step/10
}

if(QAA==4){

IminSSE=which(SSE==min(SSE))
print(c('size KK: ',length(KK)))
lin=IminSSE %% length(KK)
if(lin==0){
        lin=length(KK)
}
if(IminSSE == length(KK)*4){
        col=4
}
if(IminSSE != length(KK)*4){
        col=IminSSE %/% length(KK)+1
}

beta=Beta[lin]
type=col
start=beta
end=beta
step=step

}

}
print('Results for fixed beta in the training set')
```

```
print(c(beta, min(SSE)))

#*********************************************#
Ht=c()
SSEt=c()
BETAt=c()
BETAt[1]=0.5
for(u in 2:length(Auxt)){
eauxt=(realt[(u-1)]-Auxt[(u-1)])^2
emvt=(realt[(u-1)]-MVt[(u-1)])^2
if(eauxt>emvt){
BETAt[u]=eauxt/(eauxt+emvt)
}
if(eauxt<emvt){
BETAt[u]=emvt/(eauxt+emvt)
}
}

betat=BETAt

if(type==1){
Ht=sqrt(betat*MVt^2+(1-betat)*Auxt^2)
}

if(type==2){
Ht=(betat)*MVt+(1-betat)*Auxt
}

if(type==3){
Ht=(MVt^(betat))*(Auxt^(1-betat))
}
if(type==4){
Ht=Auxt*MVt/((1-betat)*MVt+(betat)*Auxt)
}

et=realt-Ht

SumEt=sum(et^2)
SSEt[1]=SumEt
```

```
print('Results for variable beta in the training set')
print(c(betat, SumEt))

# Calculating SSE in the training set with fixed beta

betat428=sum((Auxt-realt)*(Auxt-MVt))/sum((MVt-Auxt)^2)

if(type==1){
Ht=sqrt(betat428*MVt^2+(1-betat428)*Auxt^2)
}

if(type==2){
Ht=(betat428)*MVt+(1-betat428)*Auxt
}

if(type==3){
Ht=(MVt^(betat428))*(Auxt^(1-betat428))
}
if(type==4){
Ht=Auxt*MVt/((1-betat428)*MVt+(betat428)*Auxt)
}

et=realt-Ht

SumEt=sum(et^2)
SSEt[2]=SumEt

print('Results for fixed beta based on eq 4.28 in
 the training set')
print(c(betat428, SumEt))

#*****************************************#

BETA=c()
BETA[1]=0.5
for(u in 2:length(Aux)){
eaux=(real[(u-1)]-Aux[(u-1)])^2
emv=(real[(u-1)]-MV[(u-1)])^2
```

```
if(eaux>emv){
BETA[u]=eaux/(eaux+emv)
}
if(eaux<emv){
BETA[u]=emv/(eaux+emv)
}

}

betaF=BETA
betaF=sum((Auxt-realt)*(Auxt-MVt))/sum((MVt-Auxt)^2)
betaF=beta

if(type==1){
H=sqrt(betaF*MV^2+(1-betaF)*Aux^2)
}

if(type==2){
H=(betaF)*MV+(1-betaF)*Aux
}

if(type==3){
H=(MV^(betaF))*(Aux^(1-betaF))
}
if(type==4){
H=Aux*MV/((1-betaF)*MV+(betaF)*Aux)
}

erroH=real-H
MAE=mean(abs(erroH))
RMSE=sqrt(mean(erroH^2))
SSEf=sum(erroH^2)

print(c(MAE,RMSE,SSEf))
print(c(betaF))
print(c('type: ',type))


# Here we are using the Diebold-Mariano test
```

```
# to observe whether HMMA is more accurate
# then the methods used to generate the HMMA
# model.

eMV=real-MV
eAUX=real-Aux
teste = dm.test(eMV, erroH, alternative = "greater",
 h = 1, power = 1)
pMV=teste$p.value
teste = dm.test(eAUX, erroH, alternative = "greater",
 h = 1, power = 1)
pAUX=teste$p.value
print(c(pMV, pAUX))


png(file = "PLOT.png", width = 6, height = 5, units='in',
 res=600)
ylim = c(min(H,real,Aux)-0.02, (max(H,real,Aux)+0.05))
xlim = c(1, length(H))
plot(real, col="blue", ylim=ylim, xlim=xlim, type='l')
points(real,type='o', pch=20, cex = 0.85, col='blue')
par(new=TRUE)
plot(Aux, col="green", ylim=ylim, xlim=xlim, type='l')
points(Aux,type='o', pch=20, cex = 0.85, col='green')
par(new=TRUE)
plot(H, col="red", ylim=ylim, xlim=xlim, type='l')
points(H,type='o', pch=20, cex = 0.85, col='red')
par(new=TRUE)
plot(MV, col="orange", ylim=ylim, xlim=xlim, type='l')
points(MV,type='o', pch=20, cex = 0.85, col='orange')

metodos1=cbind(real, Aux)
metodos2=cbind(H, MV)
colnames(metodos1) = c("Real Value","Method 2")
colnames(metodos2) = c("HMMA","MVA")

legend(1, 1.025,
legend = colnames(metodos1),
bty = "n",
```

```
col = c("blue","green"),
pch = c(16, 16),
cex = 0.9,
text.font=2,horiz = TRUE,
lty = c(1,1)
)


legend(1, 1.0,
legend = colnames(metodos2),
bty = "n",
col = c("red","orange"),
pch = c(16, 16),
cex = 0.9,
text.font=2, horiz = TRUE,
lty = c(1, 1)
)
dev.off()

}
```

E.3 Mao-Xiao Approach and Naive R Code

In this section we present the R code used to provide the one-step-ahead forecasting using the Mao-Xiao Approach and Naive Estimation. No parameters must be estimated in these methods, hence, both methods are applied directly in the test set. Consider the m-element test set $y_{n-m+1}, y_{n-m+2}, \cdots, y_n$. The window considered to estimate the first element of the test set is the w-element set $y_{n-m-w+1}, \cdots, y_{n-m}$. The last window is composed by the observations $y_{n-w}, \cdots, y_{n-1}$, which is used to estimate y_n. The set $\hat{y}_{n-w+1}, \cdots, \hat{y}_n$ represents the MXA estimates for the test set and this set is used to calculate MAE, MAPE and RMSE.

```
library(e1071)

library(seastests)
library(urca)
library(zoom) # Invoke the Library
library(dgof)
library(quantmod)
library(ggplot2)
```

```
library(xts)
library(igraph)
library(corrplot)
library(forecast)
library(lmtest)
library(tseries)
library(PMCMR)
library(astsa)
library(tictoc)
library(xtable)
library(stargazer)


for(AA in 1:1){
ts=5

# orig is 0 if the input set is composed by the original data
# orig is 1 if the input set is composed by the normalized data
orig=0

if(ts==1){
x1 = read.csv(file = "AirPassengers.csv")
ini=40
x1=x1[,1]
factor=0.7

}
if(ts==2){
x1 = read.csv(file = "lynx.csv")
ini=77
x1=x1[,1]
factor=0.7
}

if(ts==3){
x1 = read.csv(file = "IBOV.csv")
ini=200
x1=x1[,1]
factor=0.5
```

```
}

if(ts==4){
x1 = read.csv(file = "nhtemp.csv")
ini=10
x1=x1[,1]
factor=0.8
}
if(ts==5){
x1 = read.csv(file = "RBTS.csv")
ini=30
x1=x1[,1]
factor=0.75
}

##############################################

n=length(x1)
w=ini
if(orig==0){
M=max(x1)
m=min(x1)
x1=(x1-m)/(M-m)
}

trai=ceiling(factor*(n-w))   # size of the training set
pred=length(x1)-(w+trai)     # size of the test set

D=matrix(rep(0,w*(n-w)),ncol=w)

for(u in 1:(n-w)){
for(v in 1:w ){
D[u,v]=x1[u+v-1]
}
}


for(qa in 1:1 ){
yMX = c()
```

```
yNai = c()
} # Fim do for em qa

for(b in 1:(n-w)){

# Building Network - visibility graph #
y = D[b,]
x = b:(w+b-1)
a = w
MAdj1  = matrix(0,nrow = a, ncol = a)

for(i in 1:(a-2)){
MAdj1[i,(i+1)]=1

m=(y[(i+1)]-y[i])/(x[(i+1)]-x[i])

for(j in (i+2):a){

ytj=y[i]+m*(x[j]-x[i])

if(y[j]>=ytj){
m=(y[j]-y[i])/(x[j]-x[i])
MAdj1[i,j]=1
}
}
}
MAdj1[(i+1),(i+2)]=1
M1 = MAdj1 + t(MAdj1)
grafo1 = graph_from_adjacency_matrix(M1, mode=c('directed'))
Sim1Dice = similarity(grafo1,method = "dice")
Sim1 = Sim1Dice-1*diag(rep(1,a))

## Calculating similarities between each previous observation
## and the last one

SimComp1 = Sim1[a,1:(a-1)]
y1n = y[w]
tn = x[w]
```

```
############ Method (Mao & Xiao, 2019) ##############

## Vertex with the higher similarity with the last point
## of the window

NC1=which.max(SimComp1)

## Calculating the angular coeficient
caC = (y[a]-y[NC1])/(x[a]-x[NC1])

#Pre-estimate for y(a+1)
yMXaux = caC*(1)+y[a]

## weight 1
wN1C = (a-NC1)/(a+1-NC1)

## complement of the weight 1
wNC = 1/(a+1-NC1)

### Calculating the one-step-ahead MXA estimation
yMX[b] = yMXaux*wN1C+y[a]*wNC

## Calculating Naive Approach ##
q=length(y)
yNai[b] = y[q]

}


yMXtrai=yMX[1:trai]  # Extracting MXA from the training set
yNaitrai=yNai[1:trai]  # Extracting Naive from the training set

realtrai=x1[(w+1):(w+trai)]

erronaitrai=realtrai-yNaitrai
erromxtrai=realtrai-yMXtrai

#### Calculando os MAEs SW e Arima
MAEmxtrai=mean(abs(erromxtrai))
```

```
MAPEmxtrai=mean(abs(erromxtrai)/realtrai)*100
RMSEmxtrai=sqrt(mean(erromxtrai^2))

MAEnaitrai=mean(abs(erronaitrai))
MAPEnaitrai=mean(abs(erronaitrai)/realtrai)*100
RMSEnaitrai=sqrt(mean(erronaitrai^2))

print(c('Naive-Trainning', MAEnaitrai,MAPEnaitrai,RMSEnaitrai))
print(c('MXA-Trainning', MAEmxtrai,MAPEmxtrai,RMSEmxtrai))
#####################################


yMXfor=yMX[(trai+1):(length(yMX))]
yNaifor=yNai[(trai+1):(length(yMX))]

realfor=x1[(w+trai+1):(n)]

indzero=which(realfor==0)

#erronaifor=realfor[-indzero]-yNaifor[-indzero]
#erromxfor=realfor[-indzero]-yMXfor[-indzero]

erronaifor=realfor-yNaifor
erromxfor=realfor-yMXfor

#### Calculando os MAEs SW e Arima
MAEmxfor=mean(abs(erromxfor))
#MAPEmxfor=mean(abs(erromxfor[-indzero])/realfor[-indzero])*100
MAPEmxfor=mean(abs(erromxfor)/realfor)*100
RMSEmxfor=sqrt(mean(erromxfor^2))

MAEnaifor=mean(abs(erronaifor))
#MAPEnaifor=mean(abs(erronaifor[-indzero])/realfor[-indzero])*100
MAPEnaifor=mean(abs(erronaifor)/realfor)*100
RMSEnaifor=sqrt(mean(erronaifor^2))
print(c('**************************************'))
print(c('Naive-Forecasting', MAEnaifor,MAPEnaifor,RMSEnaifor))
print(c('MXA-Forecasting', MAEmxfor,MAPEmxfor,RMSEmxfor))
```

```
ylim = c(min(yMXfor,yNaifor,realfor), max(yMXfor,yNaifor,realfor))
xlim = c(1, length(yMXfor))
plot(realfor, col="blue", ylim=ylim, xlim=xlim, type='l')
points(realfor,type='o', pch=20, cex = 0.85, col='blue')
par(new=TRUE)
plot(yNaifor, col="green", ylim=ylim, xlim=xlim, type='l')
points(yNaifor,type='o', pch=20, cex = 0.85, col='green')
par(new=TRUE)
plot(yMXfor, col="orange", ylim=ylim, xlim=xlim, type='l')
points(yMXfor,type='o', pch=20, cex = 0.85, col='orange')

}
```

E.4 Dynamic ARIMA R Code

```
library(e1071)

library(seastests)
library(urca)
library(zoom)
library(dgof)
library(quantmod)
library(ggplot2)
library(xts)
library(igraph)
library(corrplot)
library(forecast)
library(lmtest)
library(tseries)
library(PMCMR)
library(astsa)
library(tictoc)
library(xtable)
library(stargazer)
################################
```

```
for(Q in 1:1){
tic()

orig=0
ts=4

if(ts==1){
x1 = read.csv(file = "AirPassengers.csv")
ini=40
x1=x1[,1]
factor=0.7

}
if(ts==2){
x1 = read.csv(file = "lynx.csv")
ini=77
x1=x1[,1]
factor=0.7

}
if(ts==3){
x1 = read.csv(file = "IBOV.csv")
ini=200
x1=x1[,1]
factor=0.5

}
if(ts==4){
x1 = read.csv(file = "nhtemp.csv")
ini=10
x1=x1[,1]
factor=0.8

}
if(ts==5){
x1 = read.csv(file = "RBTS.csv")
ini=30
x1=x1[,1]
factor=0.75
```

```
}

n=length(x1)
w=ini

if(orig==0){
M=max(x1)
m=min(x1)
x1=(x1-m)/(M-m)
}


trai=ceiling(factor*(n-w)) # Size of the training set

pred=n-w-trai    # Size of the test set

#####################################################
for(qa in 1:1 ){

x1 = as.vector(x1)
x1n = x1

n = length(x1n)
t = as.vector(0:(n-1))
plot(t,x1n, type='l')

yreal = rep(0,(n-ini))
yArima = rep(0,(n-ini))
Modelos = rep(0,(n-ini))


} # Fim do for em qa

for(b in ini:(n-1)){

ind = (b-ini+1)

## Calculating Dynamic ARIMA ##
```

```
yAR = x1n[ind:b]

Lambda = BoxCox.lambda(yAR)

arimaY = auto.arima(yAR,stepwise = FALSE, seasonal=TRUE,
approximation = FALSE, allowdrift = T,allowmean = T,
lambda = Lambda)

YarimaFor = forecast(arimaY, level = c(95), h = 1)

yArima[ind]=YarimaFor$mean[1]

ya=yArima[ind]

Modelos[ind]=YarimaFor$method

}

yreal=x1[(ini+1):length(x1)]

yArimafor=yArima[(trai+1):(n-w)]
yrealfor=yreal[(trai+1):(n-w)]

Dif = (yArima-yreal)
Diffor = Dif[(trai+1):(n-w)]
ModDif = abs(Diffor)
bias = -mean(Diffor)

MAE=mean(ModDif)
MAPE=100*mean(abs(Dif[(trai+1):(n-w)]/yreal[(trai+1):(n-w)]))
RMSE=sqrt(mean(Dif[(trai+1):(n-w)]^2))

print(c(MAE,MAPE, RMSE))

## Performances


RESULTS1 = cbind((MAE),(MAPE),(RMSE))
colnames(RESULTS1) = c("MAE","MAPE","RMSE")
```

```
rownames(RESULTS1) = c("ARIMA")
print(RESULTS1)

ylim = c(min(yArimafor,yrealfor), max(yArimafor,yrealfor))
ylim = c(min(yArimafor,yrealfor), max(yArimafor,yrealfor))
xlim = c(1, length(yrealfor))
plot(yrealfor, col="blue", ylim=ylim, xlim=xlim, type='l')
points(yrealfor,type='o', pch=20, cex = 0.85, col='blue')
par(new=TRUE)
plot(yArimafor, col="magenta", ylim=ylim, xlim=xlim, type='l')
points(yArimafor,type='o', pch=20, cex = 0.85, col='magenta')


toc()

}
```

E.5 Suport Vector Regression R Code

```
library(e1071)
library(rpart)
library(fdm2id)
library(seastests)
library(urca)
library(zoom)
library(dgof)
library(quantmod)
library(ggplot2)
library(xts)
library(igraph)
library(corrplot)
library(forecast)
library(lmtest)
library(tseries)
library(PMCMR)
library(astsa)
library(tictoc)
library(xtable)
library(stargazer)
```

```
# prepare sample data in the form of data frame
# with cols of timesteps (x) and values (y)

for(BB in 1:1){

ts=2
for(AA in 1:1){

if(ts==1){
x1 = read.csv(file = "AirPassengers.csv")
ini=40
x1=x1[,1]
factor=0.7
val=36
if(orig==0){
C=30 ; gama=38 ; epsi=0.1
}
if(orig==1){
C=30 ; gama=38 ; epsi=0.1
}
}
if(ts==2){
x1 = read.csv(file = "lynx.csv")
ini=77
x1=x1[,1]
factor=0.7
val=13

if(orig==0){
C=3 ; gama=107 ; epsi=0.1
}
if(orig==1){
C=3 ; gama=140 ; epsi=0.1
}


}
```

```
if(ts==3){
x1 = read.csv(file = "IBOV.csv")
ini=200
x1=x1[,1]
factor=0.5
val=50

if(orig==0){
C=15 ; gama=40 ; epsi=0.1;
}
if(orig==1){
C=15 ; gama=40 ; epsi=0.1;
}
}
if(ts==4){
x1 = read.csv(file = "nhtemp.csv")
ini=10
x1=x1[,1]
factor=0.8
val=20
if(orig==0){
C=1 ; gama=20 ;epsi=0.1
}
if(orig==1){
C=1 ; gama=20 ;epsi=0.1
}


}
if(ts==5){
x1 = read.csv(file = "RBTS.csv")
ini=30
x1=x1[,1]
factor=0.75
val=4
if(orig==0){
C=3 ; gama=1 ;epsi=0.1
}
if(orig==1){
```

```
C=2 ; gama=0.05 ;epsi=0.1
}
}
}

n=length(x1); w=ini ; m=min(x1) ; M=max(x1)

X=x1

if(orig==0){
M=max(x1)
m=min(x1)
X=(x1-m)/(M-m)
}

D=matrix(rep(0,w*(n-w)),ncol=w)
for(u in 1:(n-w)){
for(v in 1:w ){
D[u,v]=X[u+v-1]
}
}
trai=ceiling(factor*(n-w)) # Size of the training set
pred=n-(trai+w) # Size of the test set

months=1:length(X)

for(q in 1:1){
SVMfor=c(); Trai=c();W=1:length(X)

for(u in 1:(trai)){

months=u:(u+w-1)
times=months
window=D[u,]
DF = data.frame(times,window)
colnames(DF)=c("x","y")

# train an svm model, consider further tuning
# parameters for lower MSE
```

```
svmodel = svm(y ~ x,data=DF, type="eps-regression",
kernel="radial",cost=C, epsilon=epsi, gamma=gama)

# specify timesteps for forecast, eg for all
# window + 1 month ahead
nd = u:(u+w)

###compute forecast for all the w+1 months
esti = predict(svmodel, newdata=data.frame(x=nd))
pre = unclass(esti)
Trai[u]=pre[(w+1)]
}

errosvmtrai=Trai[1:(trai-val)]-X[(w+1):(w+trai-val)]
realsvmtrai=X[(w+1):(w+trai-val)]
MAE=mean(abs(errosvmtrai))
MAPE=mean(abs(errosvmtrai)/realsvmtrai)*100
RMSE=sqrt(mean(errosvmtrai^2))

print(c('Training Errors:'))

RESULTSSVM = cbind((MAE),(MAPE),(RMSE))
colnames(RESULTSSVM) = c("MAE","MAPE","RMSE")
rownames(RESULTSSVM) = c("SVR")

print(RESULTSSVM)

errosvmval=Trai[(trai-val+1):trai]-X[(w+trai-val+1):(w+trai)]
realsvmval=X[(w+trai-val+1):(w+trai)]
MAE=mean(abs(errosvmval))
MAPE=mean(abs(errosvmval)/realsvmval)*100
RMSE=sqrt(mean(errosvmval^2))

print(c('Validation Errors:'))
RESULTSSVM = cbind((MAE),(MAPE),(RMSE))
colnames(RESULTSSVM) = c("MAE","MAPE","RMSE")
rownames(RESULTSSVM) = c("SVM")

print(RESULTSSVM)
```

```
print(c(sqrt(MAE*RMSE)))
print(c('-----------------------------------------------'))

# Starting prediction in the test set with the model
# defined in the training set

for(u in 1:pred){

months=(u+trai):(u+trai+w-1)
window=D[u+trai,]
DF = data.frame(months,window)
colnames(DF)=c("x","y")

# train an svm model, consider further tuning
# parameters for lower MSE
svmodel = svm(y ~ x,data=DF, type="eps-regression",
kernel="radial",cost=C, epsilon=epsi, gamma=gama)

# specify timesteps for forecast, eg for all
# window + 1 month ahead
nd = (u+trai):(u+trai+w)

###compute forecast for all the w+1 months
esti = predict(svmodel, newdata=data.frame(x=nd))
pre = unclass(esti)

SVMfor[u]=pre[(w+1)]
}

errosvmfor=SVMfor-X[(w+trai+1):n]
realsvmfor=X[(w+trai+1):n]

indzero=which(realsvmfor==0)
indzerotrai=which(realsvmtrai==0)
MAEfor=mean(abs(errosvmfor))


MAPEfor=mean(abs(errosvmfor)/realsvmfor)*100
```

```
MAPEfor=mean(abs(errosvmfor[-indzero])/realsvmfor[-indzero])*100
RMSEfor=sqrt(mean(errosvmfor^2))

print(c('Test set errors:'))
RESULTSSVM = cbind((MAEfor),(MAPEfor),(RMSEfor))
colnames(RESULTSSVM) = c("MAE","MAPE","RMSE")
rownames(RESULTSSVM) = c("SVR")

print(RESULTSSVM)

print(c("*********************************************"))

#plot the results

ylim = c(min(SVMfor,realsvmfor)-0.05,
 max(SVMfor,realsvmfor)+0.05)
xlim = c(1, length(realsvmfor))
plot(realsvmfor, col="blue", ylim=ylim, xlim=xlim, type='l')
points(realsvmfor,type='o', pch=20, cex = 0.9, col='blue')
par(new=TRUE)
plot(SVMfor, col="green", ylim=ylim, xlim=xlim, type='l')
points(SVMfor,type='o', pch=20, cex = 0.9, col='green')

}


}
```

E.6 Hybrid ANN-ETS R Code

```
require(validann)
require(dplyr)
require(ANN2)
library(e1071)
library(seastests)
library(urca)
library(zoom)
library(dgof)
library(quantmod)
```

```
library(ggplot2)
library(xts)
library(igraph)
library(corrplot)
library(forecast)
library(lmtest)
library(tseries)
library(PMCMR)
library(astsa)
library(tictoc)
library(xtable)
library(stargazer)


for(AB in 1:1){
h=3  # Number of hidden layers
orig=1 # orig is 1 if the original data is
#being used. Otherwise, that is, if the data
#are normalized, orig=0

##############################################
for(AA in 1:1){
ts=1
if(ts==1){
x1 = read.csv(file = "AirPassengers.csv")
ini=40
x1=x1[,1]
factor=0.7
val=36

}
if(ts==2){
x1 = read.csv(file = "lynx.csv")
ini=77
x1=x1[,1]
factor=0.7
val=13
}
if(ts==4){
```

```
x1 = read.csv(file = "IBOV.csv")
ini=200
x1=x1[,1]
factor=0.5
val=50
}
if(ts==6){
x1 = read.csv(file = "nhtemp.csv")
ini=10
x1=x1[,1]
factor=0.8
val=20
}
if(ts==7){
x1 = read.csv(file = "RBTS.csv")
ini=30
x1=x1[,1]
factor=0.75
val=4
}

}
################################################

set.seed(2)
X=x1
X = as.vector(X)

if(orig==0){
M=max(x1)
m=min(x1)
X=(x1-m)/(M-m)
}

n=length(X)
w=ini

D=matrix(rep(0,w*(n-w)),ncol=w)
```

```
for(u in 1:(n-w)){
for(v in 1:w ){
D[u,v]=X[u+v-1]
}
}

mae=function(y, y_hat){
return(sum((y-y_hat)^2))
}


#####################################################

trai=ceiling(factor*(n-w))  # Size of the training set
pred=n-(w+trai)  # Size of the test set
NETS=c()
yets=c()


# Producing the ETS one-step-ahead forecasts

for(u in 1:(n-w)){

xtrain=D[u,]
xtrain=as.vector(xtrain)
Yets = ets(xtrain, model='ZZZ')
yauxets = forecast(Yets)
yets[u]=yauxets$mean[1]

}

realets=X[(w+1):n]
erroets=realets-yets


for(u in 1:(n-w-2)){
print(u)
pesos=runif((h*(u+1)+h+1),-0.1,0.1)
xtrain=t(erroets[(1):(u)])  #
```

```
xteste=t(erroets[(2):(u+1)])  ### Starts getting N3
ytrain=erroets[(u+1)]
fit = ann(xtrain, ytrain, size =h, act_hid = "tanh",
act_out = "linear", Wts=pesos, objfn = mae)
NETS[u] = predict(fit,xteste)

}


yets1=yets[-1]
yets2=yets1[-1]
yANNets=yets2+NETS

realHEAA=X[(w+3):n]
erroHEAA=realHEAA-yANNets


realHEAAt=X[(w+3):(w+trai-val)]
erroHEAAt=realHEAAt-yANNets[1:(trai-val-2)]

MAEt=mean(abs(erroHEAAt))
MAPEt=mean(abs(erroHEAAt)/realHEAAt)*100
RMSEt=sqrt(mean(erroHEAAt^2))

print(c('Training set errors:'))
RESULTHEAA = cbind((MAEt),(MAPEt),(RMSEt))
colnames(RESULTHEAA) = c("MAE","MAPE","RMSE")
rownames(RESULTHEAA) = c("HEAA")

print(RESULTHEAA)
print(c('-------------------------------------'))

realHEAAv=X[(w+trai-val+1):(w+trai)]
erroHEAAv=realHEAAv-yANNets[(trai-val+1):(trai)]

MAEv=mean(abs(erroHEAAv))
MAPEv=mean(abs(erroHEAAv)/realHEAAv)*100
RMSEv=sqrt(mean(erroHEAAv^2))
SSE=10000*sum(erroHEAAv^2)
```

```
print(c('Validation set errors:'))
RESULTHEAA = cbind((MAEv),(MAPEv),(RMSEv))
colnames(RESULTHEAA) = c("MAE","MAPE","RMSE")
rownames(RESULTHEAA) = c("HEAA")

print(RESULTHEAA)
print(c('SSE:', SSE))
print(c('h:', h))
print(c('-------------------------------------'))

realHEAAfor=X[(w+trai+1):(n)]

EA=yANNets[(trai-1):(length(yANNets))]

erroHEAAfor=realHEAAfor-EA

MAEfor=mean(abs(erroHEAAfor))
MAPEfor=mean(abs(erroHEAAfor)/realHEAAfor)*100
RMSEfor=sqrt(mean(erroHEAAfor^2))

print(c('Test Set Errors:'))
RESULTHEAA = cbind((MAEfor),(MAPEfor),(RMSEfor))
colnames(RESULTHEAA) = c("MAE","MAPE","RMSE")
rownames(RESULTHEAA) = c("HEAA")

print(RESULTHEAA)
print(c('*****************************************'))



yANNetsfor=yANNets[(trai-1):(length(yANNets))]
yANNetsfor=EA

#plotting the results
ylim = c(min(yANNetsfor,realHEAAfor)-0.05, max(yANNetsfor,
realHEAAfor)+0.05)
xlim = c(1, length(yANNetsfor))
plot(realHEAAfor, col="blue", ylim=ylim, xlim=xlim, type='l')
```

```
points(realHEAAfor,type='o', pch=20, cex = 0.9, col='blue')
par(new=TRUE)
plot(yANNetsfor, col="green", ylim=ylim, xlim=xlim, type='l')
points(yANNetsfor,type='o', pch=20, cex = 0.9, col='green')


}
```

E.7 Hybrid ANN-ARIMA R Code

```
require(validann)
require(dplyr)
require(ANN2)


library(e1071)

library(seastests)
library(urca)
library(zoom) # Invoke the Library
library(dgof)
library(quantmod)
library(ggplot2)
library(xts)
library(igraph)
library(corrplot)
library(forecast)
library(lmtest)
library(tseries)
library(PMCMR)
library(astsa)
library(tictoc)
library(xtable)
library(stargazer)

for(AAA in 1:1){
h=3  # Number of hidden layers
orig=1 # orig is 1 if the original data is
#being used. Otherwise, that is, if the data
```

```
#are normalized, orig=0
#########################################################################
for(AA in 1:1){
ts=1
if(ts==1){
x1 = read.csv(file = "AirPassengers.csv")
ini=40
ArimaPD = read.csv(file="arimaAIRorig.csv")
if(orig==0){
ArimaPD = read.csv(file="AirPassARIMA.csv")
}
ArimaPD=ArimaPD[,1]
x1=x1[,1]
factor=0.7
val=36

}
if(ts==2){
x1 = read.csv(file = "lynx.csv")
ini=77
ArimaPD = read.csv(file="arimaLYNXorig.csv")
if(orig==0){
ArimaPD = read.csv(file="lynxARIMA.csv")
}
ArimaPD=ArimaPD[,1]
x1=x1[,1]
factor=0.7
val=13
}

if(ts==3){
x1 = read.csv(file = "IBOV.csv")
ini=200
ArimaPD = read.csv(file="arimaIBOVorig.csv")
if(orig==0){
ArimaPD = read.csv(file="IBOVARIMA.csv")
}
ArimaPD=ArimaPD[,1]
x1=x1[,1]
```

```
factor=0.5
val=50
}

if(ts==4){
x1 = read.csv(file = "nhtemp.csv")
ini=10
ArimaPD = read.csv(file="arimaNHTorig.csv")
if(orig==0){
ArimaPD = read.csv(file="nhtempARIMA.csv")
}
ArimaPD=ArimaPD[,1]
x1=x1[,1]
factor=0.8
val=20
}
if(ts==5){
x1 = read.csv(file = "RBTS.csv")
ini=30
ArimaPD = read.csv(file="arimaRBTSorig.csv")
if(orig==0){
ArimaPD = read.csv(file="RBTSARIMA.csv")
}
ArimaPD=ArimaPD[,1]
x1=x1[,1]
factor=0.75
val=4
}
}
##########################################################################

X=x1

if(orig==0){
M=max(x1)
m=min(x1)
X=(x1-m)/(M-m)
}
```

```
n=length(x1)
w=ini

months=1:length(x1)

D=matrix(rep(0,w*(n-w)),ncol=w)

for(u in 1:(n-w)){
for(v in 1:w ){
D[u,v]=X[u+v-1]
}
}

trai=ceiling(factor*(n-w))  # Size of the training set
pred=n-(w+trai)  # Size of the test set


mae=function(y, y_hat){
return(sum((y-y_hat)^2))
}

##########################################


Arima=ArimaPD  # The ARIMA forecasts were
# already calculated from
# the Dynamic ARIMA method

### Arima-ANN Kashei-Bijaris Method ###

for(q in 1:1){
real=X[(w+1):n]
erroArima=real-Arima

yArANNKB=c()
set.seed(2)
```

```
for(u in 1:(trai-1)){
### Starts with yAr(w+1) to get yArAnnKB(w+2)
xtrain=t(c(Arima[u],erroArima[1:u],X[u:(u+w-1)]))

# Estimate yArANNKB(w+2)
xteste=t(c(Arima[(u+1)],erroArima[2:(u+1)],X[(u+1):(u+w)]))
ytrain=X[(u+w)]

npesos=(u+1+w+1)*h+(h+1)  # Number of weights
pesos=runif(npesos,-0.1,0.1)
fit = ann(xtrain, ytrain, size =h, act_hid = "tanh",
act_out = "linear", Wts=pesos, objfn = mae)
yArANNKB[u] = predict(fit,xteste)

print(u)
}

realKBt=X[(w+2):(w+trai-val)]

erroKBt=yArANNKB[1:(trai-val-1)]-realKBt

MAEKBt=mean(abs(erroKBt))
MAPEKBt=mean(abs(erroKBt)/realKBt)*100
RMSEHKBt=sqrt(mean(erroKBt^2))

print(c('*************************************'))
print(c('Trainning Set Errors:'))
RESULTSHAAA_KB = cbind((MAEKBt),(MAPEKBt),(RMSEHKBt))
colnames(RESULTSHAAA_KB) = c("MAE","MAPE","RMSE")
rownames(RESULTSHAAA_KB) = c("Arima_ANN-KB")

print(RESULTSHAAA_KB)
print(c('-------------------------------------'))

realKBval=X[(w+trai-val+1):(w+trai)]

erroKBval=yArANNKB[(trai-val):(trai-1)]-realKBval

MAEKBval=mean(abs(erroKBval))
```

```
MAPEKBval=mean(abs(erroKBval)/realKBval)*100
RMSEHKBval=sqrt(mean(erroKBval^2))


print(c('Validation Set Errors:'))
RESULTSHAAA_KB = cbind((MAEKBval),(MAPEKBval),(RMSEHKBval))
colnames(RESULTSHAAA_KB) = c("MAE","MAPE","RMSE")
rownames(RESULTSHAAA_KB) = c("Arima_ANN-KB")

print(RESULTSHAAA_KB)
print(c('-----------------------------------------------'))
}


for(q in 1:1){
yArANNKB=c()
for(u in 1:(pred)){
npesos=(u+trai+w+1)*h+(h+1)  # Number of weights
pesos=runif(npesos,-0.1,0.1)

### Starts with yAr(w+1) to get yArAnnKB(w+2)
xtrain=t(c(Arima[u+trai-1],erroArima[1:(u+trai-1)],
X[(u+trai-1):(u+w+trai-2)]))

# Estimate yArANNKB(w+2)
xteste=t(c(Arima[(u+trai)],erroArima[2:(u+trai)],
X[(u+trai):(u+w+trai-1)]))
ytrain=X[(u+w+trai-1)]

fit = ann(xtrain, ytrain, size=h, act_hid = "tanh",
act_out = "linear", Wts=pesos, objfn = mae)
yArANNKB[u] = predict(fit,xteste)

print(u)
}
ArANNKB=yArANNKB
realKBfor=X[(w+trai+1):n]
erroKBfor=realKBfor-ArANNKB
ArANNKBfor=ArANNKB
```

```
erroKBfor=realKBfor-ArANNKBfor

MAEKBfor=mean(abs(erroKBfor))
MAPEKBfor=mean(abs(erroKBfor)/realKBfor)*100
RMSEKBfor=sqrt(mean(erroKBfor^2))

print(c('HAAAKB - Forecasting:',MAEKBfor,MAPEKBfor,RMSEKBfor))

print(c('Erros de Teste:'))
RESULTSHAAA_KB = cbind((MAEKBfor),(MAPEKBfor),(RMSEKBfor))
colnames(RESULTSHAAA_KB) = c("MAE","MAPE","RMSE")
rownames(RESULTSHAAA_KB) = c("Arima_ANN-KB")

print(RESULTSHAAA_KB)
print(c('**********************************************'))


#plotting the results
ylim = c(min(ArANNKB,realKBfor)-0.05, max(ArANNKB,realKBfor)+0.05)
xlim = c(1, length(realKBfor))
plot(realKBfor, col="blue", ylim=ylim, xlim=xlim, type='l')
points(realKBfor,type='o', pch=20, cex = 0.9, col='blue')
par(new=TRUE)
plot(ArANNKB, col="green", ylim=ylim, xlim=xlim, type='l')
points(ArANNKB,type='o', pch=20, cex = 0.9, col='green')


}

}
```

FOLHA DE REGISTRO DO DOCUMENTO

1. CLASSIFICAÇÃO/TIPO	2. DATA	3. REGISTRO N°	4. N° DE PÁGINAS
TD	14 de Março de 2022	DCTA/ITA/TD-004/2022	206

5. TÍTULO E SUBTÍTULO:

Maximum Visibility: A Novel Approach for Time Series Forecasting Based on Complex Network Theory.

6. AUTOR:

Filipe Rodrigues de Souza Moreira

7. INSTITUIÇÃO/ÓRGÃO INTERNO/DIVISÃO:

Instituto Tecnológico de Aeronáutica - ITA

8. PALAVRAS-CHAVE SUGERIDAS PELO AUTOR:

Time Series Forecasting; Natural Visibility Graph; Forecasting Model; Complex Network.

9. PALAVRAS-CHAVE RESULTANTES DA INDEXAÇÃO:

Análise de séries temporais; Redes complexas; Gráficos; Processamento de dados; Computação.

10. APRESENTAÇÃO: **X Nacional** **Internacional**

ITA, São José dos Campos. Curso de Doutorado. Programa de Pós-Graduação em Engenharia Eletrônica e Computação. Área de Sistemas e Controle. Orientador: Prof. Dr. Takashi Yoneyama; co-orientador: Prof. Dr. Filipe Alves Neto Verri. Defesa em 22/02/2022. Publicada em 2022.

11. RESUMO:

This work presents two time series forecasting methods. The first is the Maximum Visibility Approach (MVA), a new time series forecasting method based on the Complex Network theory. The second is the Hybrid Means of Multiple Approaches (HMMA) which is a hybrid methodology formed by the combination of two forecasting models. The MVA initially maps time series data into a complex network using the visibility graph method. Then, based on the similarity measures between the nodes in the network, MVA calculates the one-step-ahead forecasts. MVA does not use all past terms in the forecasting process, but only the most significant observations, which are indicated as a result of the autocorrelation function. The HMMA combines two forecasting methods using four types of means: quadratic, arithmetic, geometric and harmonic. Using the means inequalities, four new one-step-ahead estimators are created so, it is possible that one of these new estimators be closer to the true next term, and in this case, the combination will be more accurate than the two original estimators. These methods were applied to five different groups of data, most of them showing trend characteristics, seasonal variations and/or non-stationary behavior. We estimated error measures to evaluate the performances of MVA and HMMA. The results of the statistical tests and error measures revealed that both techniques has good performance compared to the accuracy obtained by the comparative methods considered in this work. In all cases, MVA and HMMA surpassed of achieved accuracy similar to the other forecasting methods in Literature, which confirms that this work will contribute to the field of time series forecasting not only in the theoretical aspect, but also in practice.

12. GRAU DE SIGILO:

(X) OSTENSIVO **( ) RESERVADO** **( ) SECRETO**