Improving Forecast Accuracy for ARIMA Model Using Bagging with Bootstrap

Hongyi Zheng

April 25, 2024

Preface

Plagiarism Declaration

This piece of work is a result of my own work except where it forms an assessment based on group project work. In the case of a group project, the work has been prepared in collaboration with other members of the group. Material from the work of others not involved in the project has been acknowledged and quotations and paraphrases suitably indicated.

Contents

Preface **i**

1 Introduction **1**
1.1 Time Series Models . . . 1
1.2 Bootstrap Methods . . . 2
1.3 Improving Forecast Accuracy for Time Series Model . . . 3
1.4 Report Structure . . . 3

2 Time Series Forecasting **5**
2.1 Time Series Characteristics . . . 5
2.1.1 Time Series Pattern . . . 5
2.1.2 Stationarity . . . 6
2.1.3 Autocorrelation . . . 7
2.2 Data Preparation . . . 8
2.2.1 Data Transformation . . . 8
2.2.2 Data Decomposition . . . 9
2.3 Time Series Model . . . 11
2.3.1 General Linear Process . . . 11
2.3.2 Linear Stationary Model . . . 12
2.3.3 Linear Non-stationary Model . . . 14
2.4 Model Specification . . . 15
2.4.1 Identification . . . 15
2.4.2 Estimation . . . 16
2.4.3 Forecasting . . . 17
2.5 Conclusion . . . 18

3 Bootstrap **19**
3.1 Efron's Bootstrap . . . 19
3.2 Moving Block Bootstrap . . . 22
3.2.1 Moving Block Bootstrap Demonstration . . . 22
3.2.2 Remainder Moving Block Bootstrap . . . 23
3.3 Sieve Bootstrap . . . 24
3.3.1 Sieve Bootstrap Demonstration . . . 25
3.3.2 Remainder Sieve Bootstrap . . . 26
3.4 Bootstrap Aggregating (Bagging) . . . 27
3.5 Overall Process . . . 28
3.6 Conclusion . . . 30

4 Application **31**
4.1 Datasets . . . 31
4.2 Evaluation Methodology . . . 31
4.2.1 Training and Test Sets . . . 32
4.2.2 Accuracy Metrics . . . 32
4.2.3 Nonparametric Tests . . . 34
4.3 Method Application . . . 35
4.3.1 Data Preparation . . . 35
4.3.2 Moving Block Bootstrap . . . 36
4.3.3 Sieve Bootstrap . . . 38
4.3.4 Forecasting . . . 38
4.3.5 Bagging . . . 38
4.4 Result Analysis . . . 39
4.4.1 Agreement Analysis . . . 39
4.4.2 Difference Analysis . . . 41
4.5 Conclusion . . . 43

5 Conclusion **44**
5.1 Discussions and Limitations . . . 44
5.2 Final Conclusion . . . 45

A Related Data and Code **46**

Chapter 1

Introduction

1.1 Time Series Models

A time series is a sequence of discrete data points indexed by time. Such data is ubiquitous in everyday life, for example, the daily closing prices of the FTSE 100 Index, the average monthly temperature in London, and the annual number of babies born in the UK, among others. Forecasting and analysing time series based on historical data and the potential impact of future events is crucial for people's decision-making process. For instance, Box et al. [6] suggested that the forecast for a time series data can lay a foundation for economic and business planning, production planning, inventory and production control, and control and optimising of industrial processes. Additionally, the quality of the prediction or the accuracy of the forecast is vital, which requires the time series to be accurately fitted and the hidden patterns to be identified. For this purpose, researchers have proposed many statistical models for time series forecasting, among which two families of models are widely used, the exponential smoothing models and the Autoregressive integrated moving-average (ARIMA) models [29].

Proposed by researchers in the 1950s, the exponential smoothing model features producing forecasts by calculating the weighted average of all historical data, with weights decreasing as the observations get older [29]. Based on this idea, the exponential smoothing models have evolved to include the analysis of various components of a series. This includes the trend, which is the long-term behaviour of a series; seasonality, which represents the recursive patterns of a series with a fixed frequency; and remainder, which captures the unpredictable aspects of a series. Moreover, by including or excluding trend or seasonal components and combining three components in an additive or multiplicative manner, 30 variations of the exponential smoothing model are formed according to Bergmeir et al. [4]. Thanks to the flexibility of the combination of the components, the exponential smoothing models are applicable to various time series and can produce reliable forecasts rapidly, which offers notable benefits that are crucial for industrial applications. For example, Winters [47] utilised the exponential smoothing model to forecast the monthly and bi-monthly sales data and argued that the exponential smoothing model is superior in terms of accuracy, efficiency, and adaptability compared to more conventional forecasting methods, like producing a forecast as the average of the past two periods and producing a forecast as the average of a specified previous period. He concluded that the exponential smoothing model is particularly suited to businesses that need to make frequent and reliable forecasts for a large number of products, demonstrating significant advantages in reducing the required storage for information and in its responsiveness to shifts in sales patterns.

Another widely used forecasting model family is the ARIMA model. Instead of describing the seasonality and trend within the data, the ARIMA model aims at capturing the serial dependence structure in the data [29]. The ARIMA model is the acronym for the autoregressive integrated moving-average model, which consists of three components. This includes the autoregressive (AR) component, which captures the relationships between an observation and a

number of lag observations; the integrated (I) component, which integrates the difference of observations to make the time series stationary (will be discussed in Section 2.1.2), addressing trends over time; and the moving-average (MA) component, which models the error term as a combination of previous error terms [6]. For example, given the time series data z_t, white noise a_t, $t = 1, \ldots, T$, the ARIMA model is,

$$(1 - \phi_1 B - \cdots - \phi_p B^p)(1 - B)^d z_t = c + (1 + \theta_1 B + \cdots + \theta_q B^q) a_t$$

where B is the backward operator ($B^n z_t = z_{t-n}$); ϕ, θ, are parameters for AR and MA components; p, q, are orders for AR and MA components, d is the degree for the differencing, and c is the constant. Moreover, similar to exponential smoothing, the ARIMA model can also model the seasonality of the data by multiplying the non-seasonal ARIMA model by a seasonal component, which will be discussed in detail in the following chapter. By including or excluding the autoregressive, integrated, moving-average, and seasonal components, the ARIMA model can produce a reasonable forecast for time series data. One interesting application of the ARIMA model is de Oliveira and Cyrino Oliveira's [15] work of combining the ARIMA model with a technique called bootstrap aggregating (will be introduced later) to forecast the electricity consumption data of several nations. They concluded that using ARIMA models in this way can avoid the complex and sensitive task of selecting exogenous variables, like economic factors that affect electricity consumption, and achieve relatively lower errors in long-run forecasting.

Despite both models having wide applications in forecasting, their specialised data types are different. The ARIMA models tend to perform better for longer, more stable time series data, yet the exponential smoothing models are more effective for noisier, more volatile data sets [35]. Nonetheless, both time series models are able to recognise the past patterns of time series data and generate robust forecasts.

1.2 Bootstrap Methods

As mentioned above, de Oliveira and Cyrino Oliveira [15] adopted a technique named bootstrap aggregating together with the ARIMA model to forecast. The technique, bootstrap aggregating, or bagging, was originally designed as a robust ensemble learning technique that enhances the stability and accuracy of machine learning algorithms [8]. In the context of regression in machine learning, by generating multiple versions of a predictor (a function mapping the input data to the output) through bootstrapping (sampling with replacement) and averaging the predictors to get an aggregated predictor, bagging can reduce the model variance and avoid overfitting [8]. Similarly, in the context of time series forecasting, bagging involves generating multiple subsets of the original time series data through bootstrapping and then applying the forecasting model to each of the subsets. The final forecast is typically the average of all the forecasts from each model [4].

Additionally, the resampling strategy used in bagging is the bootstrap, established by Efron [16]. Efron's bootstrap method assumes the data is independent and identically distributed (iid) and features generating resamples by sampling one observation at a time from the data with replacement until the resample has the same size as the raw data. It is a powerful tool for estimating the statistical properties of the data without further assumptions on its distribution. However, for time series data, Efron's method is not suitable as the time series data are often serial dependent and hence the fundamental iid assumption is violated. Therefore, other bootstrap methods that relieve the constraint of iid data and preserve the dependence structure of the data should be used when bootstrapping time series data. Künsch [33] proposed the moving block bootstrap, which resamples a number of consecutive observations several times from the data to maintain the dependence structure within the data. On the other hand, Bühlmann [9] established the sieve bootstrap, which features resampling the residuals one at a time with replacement until the desired size is reached after fitting a time series model to the data. In

addition to these two methods, there are other methods designed for serial-dependent data, such as the stationary bootstrap [40], the tapered block bootstrap [38], and the dependent wild bootstrap [43].

1.3 Improving Forecast Accuracy for Time Series Model

Although researchers have developed many time series forecasting models to deliver precise predictions, the enhancement of forecasting accuracy for the exponential smoothing models and the ARIMA model is still worth further investigation. Cordeiro and Neves [13] first proposed combining the bootstrap method with the exponential smoothing model in 2009 to improve its forecast performance. While he had some success with monthly and quarterly data, his method often produced poorer accuracy than the exponential smoothing-only method. Based on Cordeiro and Neves's idea, Bergmeir et al. [4] used a different data decomposition method and bootstrap resampling method to improve the exponential smoothing model. His method has the following steps: (1) convert the data using a power transformation; (2) decompose the data into seasonal, trend, and remainder components; (3) bootstrap the remainder component and combine the resamples with the decomposed components; (4) forecast each combined resamples; (5) bagging the point forecasts. The results showed that his proposed method outperformed the standard exponential smoothing model in monthly data. In addition to the research on using bootstrap to improve the exponential smoothing model, de Oliveira and Cyrino Oliveira [15] adopted a similar approach as Bergmeir et al. and investigated the possibility of using bagging for the ARIMA model to improve the forecast accuracy and yielded the result that bagging and bootstrap indeed improved the ARIMA model. However, their experiment dataset was limited to a specific frequency (monthly) and a specific type (electricity consumption data).

Inspired by previous research, we attempted to apply Bergmeir et al. [4]'s bagging and bootstrap methods applied to the exponential smoothing model, modified and adapted them to the ARIMA model, and analysed whether it could enhance the forecasting accuracy of the ARIMA model on more diverse datasets. In this report, we proposed using two bootstrap methods, the moving block bootstrap and the sieve bootstrap, to generate resamples for the time series data. In terms of bagging, we proposed aggregating by taking not only the mean but also the median of the point forecasts from each ARIMA model fitted to the resamples, since the median is less sensitive to outliers and might be appropriate to use for some kinds of data [15]. Consequently, there are four combinations or variations for our proposed method, which are moving block bootstrap with either mean or median bagging and sieve bootstrap with either mean or median bagging. The details of our proposed method will be discussed in the following chapters.

1.4 Report Structure

With the aim of investigating the effectiveness of using bagging with bootstrap to improve the ARIMA model, we have divided this report into five chapters, which are the introduction, time series forecasting, bootstrap, application, and conclusion. The following chapters are structured as follows,

In Chapter 2, time series forecasting, we will mainly introduce the topics related to time series forecasting, including the characteristics of the time series data and the methods used to prepare the data, followed by the development of the family of the ARIMA models and the specification of the ARIMA model.

In Chapter 3, bootstrap, we will present the idea of bootstrap from Efron's bootstrap method to the moving block bootstrap and the sieve bootstrap, followed by the concept of bagging and the complete process of applying the bagging with bootstrap to improve the forecast accuracy of the ARIMA model.

In Chapter 4, application, we will apply our method to a real-world dataset. Hence, we will first demonstrate our dataset and the methodology for result evaluation. Then, the results yielded from our process will be analysed.

In Chapter 5, conclusion, we will compare our results with results from other researchers, evaluate the limitations of our report and conclude the report with a final conclusion.

Chapter 2

Time Series Forecasting

In order to forecast time series data using the ARIMA model, several assumptions on the data are required. Box et al. [6] assumed that the time series data is discrete and the time intervals in between are equal. For instance, given the historical data upon a time t, $z_t, z_{t-1}, z_{t-2}, \ldots$ and the period l that aimed to forecast, the real values for the future period can be denoted as z_{t+l} and the forecast function that contains all the forecast for period l can be denoted as $\hat{z}_t(l)$. Then, the objective of forecasting is to find the appropriate forecast function that minimises the mean square deviation between z_{t+l} and $\hat{z}_t(l)$ for every time in future period l [6].

Additionally, Box et al. [6] also assumed that the time series data follows a known stochastic model to ensure the predictability of the data. In this report, we used the Autoregressive integrated moving-average (ARIMA) model proposed by Box et al. [6] to model the behaviour of the time series data. Moreover, before fitting the ARIMA model, the characteristics of the data need to be evaluated. This evaluation is necessary since it provides information on the nature of the data and for the data preparation step, which aims at fulfilling the assumptions of Bootstrap introduced in the next chapter.

For the following sections, the characteristics of time series data will be evaluated first. Then, the methods used in data preparation before fitting the model will be introduced. Finally, the fundamental model, the autoregressive integrated moving-average (ARIMA) model and how it is specified, will be explained in detail.

2.1 Time Series Characteristics

Given a series of data indexed by time, we could evaluate it from various perspectives. For instance, the mean and variance of the data in a period of time and the correlation of one data point to the others. Among all the perspectives, there are three key characteristics required to be assessed, which are the time series pattern, stationarity and autocorrelation. By evaluating these features of time series, we could have a general knowledge of the data collected and the data preparation stage could be performed to ensure the validity of further manipulation.

2.1.1 Time Series Pattern

The first characteristic is the pattern of the data. According to Hyndman and Athanasopoulos [29], there are three patterns for the time series data, which are trend, seasonal, and cyclic. A trend exists when the data is ascending or descending for a long period. Furthermore, this pattern does not require linearity of the data, but a rough direction. In terms of the seasonal pattern, it happens when the data exhibits the same pattern repeatedly according to a fixed time slot, such as a week, a month, or a year. As the cyclic, it occurs when the data exhibits irregular fluctuation in a large time slot (at least 2 years), due to economic factors. Noticeably,

a time series may exhibit no pattern or more than one pattern. An example of four different series is given below,

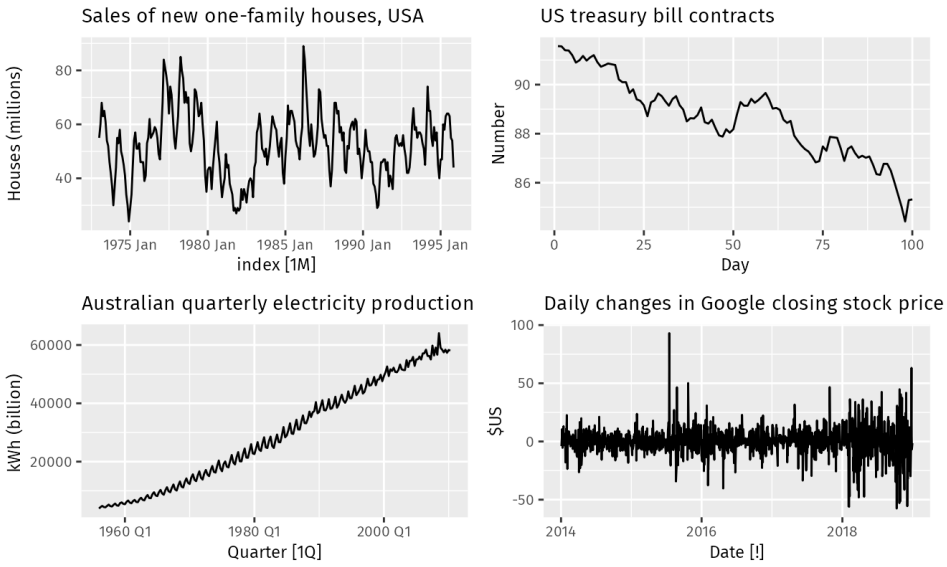

Figure 2.1: An example of time series patterns [29]

In Figure 2.1, the top-left series exhibits seasonality within one year while showing a cyclic pattern within a period of about 6-10 years. The top-right series demonstrates a downward trend with no other patterns, while the bottom-left series shows an upward trend with strong seasonality. As for the series at the bottom-right, it exhibits no patterns, which is hard to forecast with a simple model.

2.1.2 Stationarity

After assessing the patterns for time series data, another crucial characteristic that is required for evaluation is stationarity. The importance of this feature lies in the fact that it follows the assumption that the series is in statistical equilibrium, in which the time series model can be built upon [6] and bootstrapping can be performed [33]. Formally, a time series is strictly stationary if its properties are constant across time. For example, the joint probability distribution for time t_m to t_{m+h} is the same as that of time t_n to t_{n+h}, $\forall m, n, h$. That is, the probability distribution of the set of data within the whole series depends only on the time differences (length of the set). Less strictly, defined by Box et al. [6], the weak stationarity allows the covariances between neighbouring data, or autocovariance (will be discussed next), $Cov(z_t, z_{t+k}) = \gamma_k$, depends only on time differences or lag k while maintaining other assumptions for strict stationarity. In practice, the weak stationarity is adopted as it is enough for constructing the model.

Noticeably, time series with patterns may not be stationary. In Figure 2.1, the top-right and the bottom-left series have trend patterns, which indicates a changing mean, and therefore nonstationary. However, even for series with no patterns, the stationarity is not guaranteed. The bottom-right series oscillates around 0, which may indicate a constant mean of 0, yet the variance for the period after 2018 is larger than that of the period before 2015, which leads to nonstationarity. Thus, it is not sufficient to solely consider the patterns or stationarity when analysing the time series.

2.1.3 Autocorrelation

According to the assumption of stationarity, we can infer that the means and variances are constant across time. Then, the mean and variance for a stationary process are [6],

$$\mu = \mathbb{E}[z_t] = \int_{-\infty}^{\infty} z p(z)\, dz \tag{2.1}$$

$$\sigma_z^2 = \mathbb{E}[(z_t - \mu)^2] = \int_{-\infty}^{\infty} (z-\mu)^2 p(z)\, dz \tag{2.2}$$

where $p(\cdot)$ is the probability distribution for data, which is constant across time due to the stationarity assumption.

Furthermore, given the observed data $z_1, \ldots, z_N$, the mean and variance can be estimated by $\bar{z}$ and $\hat{\sigma}_z$ respectively [6],

$$\bar{z} = \frac{1}{N}\sum_{t=1}^{N} z_t \qquad \hat{\sigma}_z = \frac{1}{N}\sum_{t=1}^{N} (z_t - \bar{z}^2) \tag{2.3}$$

As discussed in the last subsection, the stationarity assumption provides that the autocovariance of the data depends on the lag k or time differences, which means for all time, the covariances between any pair of data is the same. This covariance is the autocovariance γ_k with lag k, which is given by [6],

$$\gamma_k = Cov(z_t, z_{t+k}) = \mathbb{E}[(z_t - \mu)(z_{t+k} - \mu)] \tag{2.4}$$

Then, the autocorrelation can be defined in a similar way as the autocovariance, just as the Pearson correlation coefficient with covariance, describing the data correlations. The formula for autocorrelation ρ_k with lag k is [6],

$$\rho_k = \frac{\mathbb{E}[(z_t - \mu)(z_{t+k} - \mu)]}{\sqrt{\mathbb{E}[(z_t-\mu)^2]\mathbb{E}[(z_{t+k}-\mu)^2]}} = \frac{\mathbb{E}[(z_t - \mu)(z_{t+k} - \mu)]}{\sigma_z^2} = \frac{\gamma_k}{\gamma_0} \tag{2.5}$$

since $\sigma_z^2 = \gamma_0 = Cov(z_t, z_t) = \sigma_{z_t}^2$ at time t and time $t+k$.

Consequently, the autocorrelation function (ACF) is the plot that maps the lag k with autocorrelation coefficients ρ_k. Moreover, given the mathematical representation of the autocorrelation function and the sample, the sample autocorrelation function can be estimated by the sample autocovariance $\hat{\gamma}_k$ in the form [32],

$$r_k = \frac{c_k}{c_0} \tag{2.6}$$

where

$$c_k = \hat{\gamma}_k = \frac{1}{N}\sum_{t=1}^{N-k} (z_t - \bar{z})(z_{t+k} - \bar{z}) \quad k = 0, 1, 2, \ldots, K \tag{2.7}$$

In practice, as Box et al. [6] stated, estimating a useful autocorrelation function requires at least 50 observations with K less than a quarter of the sample size.

For better analysing the time series, visualising the ACF is also helpful. For instance, an example is given in the book by Hyndman and Athanasopoulos [29] about the ACF for the monthly sale of an antidiabetic drug in Australia (Figure 2.2).

Without observing the plot of the original data, we can infer from the ACF plot that the data has a trend pattern due to the slow decrease of autocorrelation coefficients as the increase of lag. Also, seasonality also exists, since the plot shows that there are dings in the data in a one-year interval. The ACF plot produced by R is useful for understanding the nature of the observations and determining the parameters for the ARIMA model, which will be discussed in the following sections.

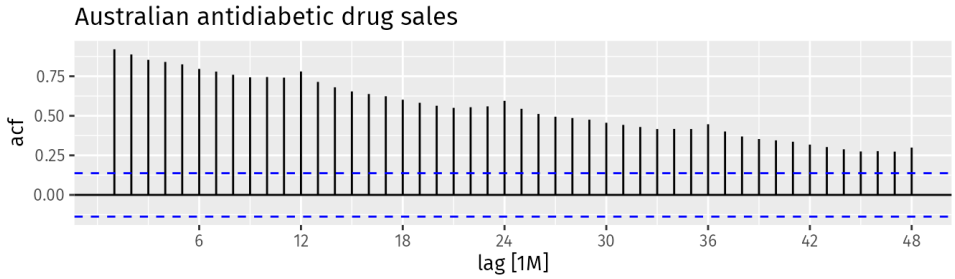

Figure 2.2: Example for ACF visualisation [29]

2.2 Data Preparation

In the previous section, the characteristics of the time series data are introduced. Based on the information provided by evaluating those characteristics, we could then preprocess the data before fitting the ARIMA model to achieve better performance. In order to improve the model fit and forecast accuracy, one fundamental standard that is required to meet is stationarity [6]. Moreover, as proposed by Bergmeir et al. [4], there are two manipulations that could be performed prior to applying the model for achieving stationarity, which are data transformation and data decomposition.

2.2.1 Data Transformation

The first method is named the Box-Cox transformation, which is a power transformation proposed by Box and Cox [7] in 1964. As many real-world time series are non-stationary with changing variances (heteroscedasticity), the method features stabilising the variance through a power function. Box and Cox [7] assumed the transformation is from z to $z^{(\lambda)}$, where the transformed data is,

$$z^{(\lambda)} = \begin{cases} \frac{z^{\lambda}-1}{\lambda} & \text{if } \lambda \neq 0 \\ \log z & \text{if } \lambda = 0 \end{cases} \tag{2.8}$$

where λ is a parameter that defines the transformation. When $\lambda = 1$, the transformed series is just the original series shifted downward for 1 unit. When $\lambda = 0$, the transformed series is the natural logarithm of the original series. Then, rather than using the method of maximising the likelihood function involving λ and unknown parameters by Box and Cox [7], we adopt Guerrero's [22] method to determine the value for λ. The main difference of this method compared to previous approaches is that it considers minimising the variation of the subseries' coefficients of variation.

Consider a random variable $X > 0$ with finite mean and variance, and assume the Box-Cox transformation of X is $T(X)$, which is continuously differentiable. Then, the transformation can be approximated by the Taylor's theorem,

$$T(X) \approx T[\mathbb{E}(X)] + T'[\mathbb{E}(X)](X - \mathbb{E}(X)) \tag{2.9}$$

where $T'(\cdot)$ is the derivative of the transformation. Take variance for both sides and obtain,

$$Var(T(X)) \approx (T'(\mathbb{E}(X)))^2 Var(X) \tag{2.10}$$

From Equation (2.10), we can observe that the variance of the original series is on the right with the stabilised variance on the left. Then, the coefficient of variation, $a > 0$, is a constant defined as [22],

$$\frac{\sqrt{Var(X)}}{(\mathbb{E}(X))^{1-\lambda}} = a \tag{2.11}$$

With the definition of the coefficient of variation (2.11) and the observations $z_t, \quad t = 1, \ldots, N$, Guerrero's approach is defined as follows: first, delete n observations from either the beginning or the end of the series so that the original series can be divided into H non-overlapping subseries with length $R = \frac{N-n}{H}$. The length of the subseries, R, is set to the length of the seasonality if it exists, and otherwise is set to 2. Next, the mean $\bar{z}_h$ and standard deviation s_h of the subseries are calculated. Finally, λ is determined by minimising the variation of the coefficient of variation, a, where,

$$\frac{s_h}{\bar{z}_h^{1-\lambda}} = a, \quad h = 1, \ldots, H \tag{2.12}$$

2.2.2 Data Decomposition

The second method implemented in this report that helps to improve stationarity is data decomposition. In this report, we adopt two methods, which are the Loess method and STL decomposition. For non-seasonal data, we apply Loess regression on the series to divide it into trend and remainder components. As for seasonal data, we implement the seasonal, trend decomposition using Loess (STL decomposition) to separate the original series into seasonal, trend, and remainder components [11]. As mentioned in Section 2.1.2, a time series with a trend pattern or seasonal pattern exhibits changing mean or variance, i.e. nonstationarity. Therefore, removing the seasonality and trend from the series could help to categorise the patterns and behaviours of the data and ensure stationarity.

Since the STL decomposition is composed of a series of smoothing operations that use Loess regression, we then introduce the Loess first. Define a neighbourhood for each data point, and then assign weights to the points in the neighbourhood according to their Euclidean distance to the origin of the neighbourhood. Next, a polynomial is fitted based on the weights to give the estimation. In detail, according to Cleveland et al. [11], suppose x_i and y_i with $i = 1, \ldots, n$ are the observations for the independent and dependent variables. Also, define the Loess regression curve, $\hat{g}(x)$, as the smoothing of y on x, which is defined everywhere, not just on x_i. For the neighbourhood, select q values from x_i that are closest to x, with $q \leq n$. Define the weight function $W(u)$ to be,

$$W(u) = \begin{cases} (1-u^3)^3 & \text{if } 0 \leq u < 1 \\ 0 & \text{if } u \geq 1 \end{cases} \tag{2.13}$$

Thus, the weight for any data point within the neighbourhood is,

$$v_i(x) = W\left(\frac{|x_i - x|}{max_i|x_i - x|}\right) \tag{2.14}$$

From Equation (2.14) we can observe that the weight for a data point decreases as its distance from x increases and eventually becomes zero at the furthest point. Finally, as suggested in Cleveland et al.'s paper, a polynomial with degree d either equals 1 or 2 is fitted to the data with weight $v_i(x)$ at each point (x_i, y_i), which gives $\hat{g}(x)$. Moreover, consider the other case for q, where $q > n$. Let $\lambda_n(x)$ to be the distance from x to the furthest x_i, then $\lambda_q(x)$ is defined as,

$$\lambda_q(x) = \lambda_n(x)\frac{q}{n} \tag{2.15}$$

Next, the weights for all the data points are calculated using this new λ_q. In this case, as q increases, $\hat{g}(x)$ will be smoother since the weight for each point tends to the same level. Eventually, as q tends to infinity, the weight $v_i(x)$ for all points will tend to 1, and $\hat{g}(x)$ will be an ordinary least-square (OLS) polynomial with degree d.

After defining the Loess regression, we can then introduce the STL decomposition in detail. Cleveland et al. [11] proposed that the decomposition consists of two nested loops, where an

inner loop is nested in an outer loop. The inner loop iterated for $n_{(i)}$ times with each iteration updating seasonal and trend components once. The outer loop iterated for $n_{(o)}$ times with each iteration consisting of a full run of the inner loop and a calculation of robustness weight, which will be used in the next run of the inner loop as the weights for Loess regression. Additionally, define the cycle-subseries as the series capturing cycle pattern with parameter $n_{(p)}$ controlling the number of elements within one cycle. For instance, for monthly data with annual periodicity, $n_{(p)}$ is 12. The first subseries is the January data and the second is the February data and so on.

As for the procedures of the inner loop, we assume linear additive for the seasonal, trend, and remainder components,

$$Y_v = S_v + T_v + R_v \tag{2.16}$$

where v controls the time of the series with $v = 1, \ldots, N$. For generality, we consider the loop just finished k^{th} iteration. Therefore, the seasonal and trend components at the start of $(k+1)^{th}$ iteration are $S_v^{(k)}$ and $T_v^{(k)}$ respectively. There are six steps for the inner loop, of which the second to the fourth steps are for updating seasonal components and the sixth step is for updating trend components. The steps are [11],

Step 1: Detrending: remove the trend components updated in the previous iteration, $Y_v - T_v^{(k)}$. For the first iteration in the first loop, the trend component due to be subtracted is set as, $T_v^{(0)} \equiv 0$.

Step 2: Cycle-subseries smoothing: apply Loess regression on each subseries with parameters $q = n_{(s)}$ and $d = 1$. The results of Loess fitting for all subseries are stored in a temporary seasonal series, $C_v^{(k+1)}$.

Step 3: Low-pass filtering of smoothed cycle-subseries: implement a low-pass filter on $C_v^{(k+1)}$, which consists of moving-average fitting for three times with the length for the first two to be $n_{(p)}$ and the last to be 3. Also, a Loess fitting with $q = n_{(l)}$ and $d = 1$ is applied after the moving-average. Then, the outputs are stored in another temporary series, $L_v^{(k+1)}$.

Step 4: Detrending of smoothed cycle-subseries: obtain the updated seasonal component by subtracting $L_v^{(k+1)}$ from $C_v^{(k+1)}$, as $S_v^{(k+1)} = C_v^{(k+1)} - L_v^{(k+1)}$.

Step 5: Deseasonalising: remove the updated seasonal components $S_v^{(k+1)}$ from Y_v.

Step 6: Trend smoothing: apply Loess on the series from the previous step with parameters $q = n_{(t)}$ and $d = 1$. Then, the trend component at this round of iteration, $T_v^{(k+1)}$, is smoothed values.

In terms of the outer loop, as the inner loop is nested inside the outer loop, we can calculate the remainder component, R_v, given the estimates of the seasonal and trend components after a full run of the inner loop, as

$$R_v = Y_v - S_v - T_v \tag{2.17}$$

Consequently, we can then compute the robustness weight with the remainder component to reflect how extreme the remainder is [11]. Let $h = 6 * \text{median}(R_v)$. Then, the robustness weight is defined as,

$$\rho_v = B\left(\frac{|R_v|}{h}\right) \tag{2.18}$$

where $B(\cdot)$ is in the form of,

$$B(u) = \begin{cases} (1-u^2)^2 & \text{if } 0 \le u < 1 \\ 0 & \text{if } u > 1 \end{cases} \tag{2.19}$$

Each iteration of the outer loop ends with an estimate of the remainder component R_v and a robustness weight ρ_v. For the next iteration, the robustness weight is applied to the data when applying Loess in Step 2 to Step 6 in the inner loop. Moreover, the trend component in the first step is set to be the trend component from the last step in the previous outer loop. After $n_{(o)}$ times of iterations, the process terminates and outputs the three separated components.

2.3 Time Series Model

After introducing the characteristics of the time series data and its related manipulations, we can now proceed to the area of the time series model. Time series models can be categorised in many ways, one of the most important of which is according to stationarity, that is, into stationary and nonstationary models. As defined in Section 2.1.2, stationary models assume that the process maintains statistical equilibrium (the probabilistic properties are time-invariant). In other words, the data fluctuates around a constant mean and has a constant variance. However, in the real world, many datasets are not stationary, such as daily stock prices, the number of people travelling by air each month, and the annual GDP of a country. Even though some of these datasets exhibit changing means, their variances are relatively constant. Thus, we can adopt nonstationary models to address the issue.

In this section, we will first discuss the general linear stochastic model. Then, based on the general model, we will introduce linear stationary models such as the Autoregressive model (AR), the Moving-average model (MA), and the Autoregressive-Moving-average model (ARMA). Lastly, we will introduce the linear nonstationary model developed based on these stationary models, the Autoregressive integrated Moving-average model (ARIMA).

2.3.1 General Linear Process

Based on the idea from Yule [49], Box et al. [6] summarised that observation from a time series, z_t, with its previous values highly dependent, can be described by a linear aggregation of infinite independent random shocks. The random shocks, $a_t, a_{t-1}, \ldots$, are uncorrelated and have zero mean,

$$\mathbb{E}[a_t] = 0, \quad \mathrm{Var}[a_t] = \sigma_a^2$$

Thus, the general linear process is [6],

$$\tilde{z}_t = a_t + \sum_{j=1}^{\infty} \psi_j a_{t-j} \tag{2.20}$$

where $\tilde{z}_t = z_t - \mu$. An important result, based on (2.20), is that any zero-mean purely non-deterministic stationary process $\tilde{z}_t$ can be described in the form of (2.20) with the coefficients converge, $\sum_{j=0}^{\infty} |\psi_j| < \infty$ [48]. Furthermore, (2.20) can also be written as a weighted aggregation of previous values equivalently,

$$\tilde{z}_t = a_t + \sum_{j=1}^{\infty} \pi_j \tilde{z}_{t-j} \tag{2.21}$$

As for the relationship between the coefficients ψ and π in (2.20) and (2.21), it can be analysed by introducing a backward shift operator, B, where,

$$Bz_t = z_{t-1} \quad \text{and} \quad B^j z_t = z_{t-j}$$

Then, Equation (2.20) can be rewritten as,

$$\tilde{z}_t = \left(1 + \sum_{j=1}^{\infty} \psi_j B^j\right) a_t = \psi(B) a_t \tag{2.22}$$

Also, Equation (2.21) can be rewritten as,

$$a_t = \left(1 - \sum_{j=1}^{\infty} \pi_j B^j\right) \tilde{z}_t = \pi(B)\tilde{z}_t \tag{2.23}$$

If substituting (2.23) into (2.22), the relationship of these two coefficients can be derived as,

$$\psi(B)\pi(B)\tilde{z}_t = \tilde{z}_t \quad \text{hence} \quad \psi(B)\pi(B) = 1$$

Moreover, there are two conditions that are crucial for the general linear process to follow, as they ensure the uniqueness and interpretability of the process, which are stationarity and invertibility. To ensure stationarity, as we mentioned a few lines ahead, the convergence of ψ needs to be guaranteed. The other condition is independent of stationarity, as it is related to the convergence of π. In summary, the process (2.20) is stationary if $\sum_{j=0}^{\infty} \psi_j < \infty$ and invertible if $\sum_{j=0}^{\infty} \pi_j < \infty$. Additionally, these conditions are also satisfied for $|B| < 1$ when the series of $\psi(B)$ and $\pi(B)$ converge [6].

2.3.2 Linear Stationary Model

In the previous section, we have introduced the two equivalent forms of the general linear process with an infinite number of parameters (ψ and π). This may reduce the general applicability of the model in the real world, since in reality, the length of data is often limited. Therefore, we now need to consider the case of finite parameters.

Autoregressive model Consider a special case of (2.21), which only has non-zero coefficients for the first p lags. Then, this finite order linear process AR(p) is,

$$\tilde{z}_t = \phi_1 \tilde{z}_{t-1} + \phi_2 \tilde{z}_{t-2} + \ldots + \phi_p \tilde{z}_{t-p} + a_t \tag{2.24}$$

The process above is defined as the autoregressive process with order p [6]. Equivalently, it can also be written as with the backward shift operator,

$$\begin{aligned} a_t &= (1 - \phi_1 B - \phi_2 B^2 - \ldots - \phi_p B^p)\tilde{z}_t \\ &= \phi(B)\tilde{z}_t \end{aligned} \tag{2.25}$$

As for stationarity and invertibility conditions, we can evaluate them by combining the AR model with the general linear process. Thus, we derive an equation by joining (2.22) and (2.25),

$$\tilde{z}_t = \phi^{-1}(B)a_t \equiv \psi(B)a_t = \sum_{j=0}^{\infty} \psi_j a_{t-j} \tag{2.26}$$

Since $\phi(B)$ is a polynomial, we can rewrite it as,

$$\phi(B) = \prod_{i=1}^{p} (1 - G_i B)$$

where G_i^{-1} are the roots for $\phi(B) = 0$. Hence, by implementing partial fraction decomposition, we can express $\psi(B)$ in terms of $\phi(B)$,

$$\psi(B) = \phi^{-1}(B) = \sum_{i=1}^{p} \left(\frac{K_i}{1 - G_i B}\right)$$

where K_i are the constants. As mentioned at the end of Section 2.3.1, the process is stationary if $\psi(B)$ converges with $|B| < 1$. Then, the weights $\psi_j = \sum_{i=1}^{p} K_i G_i^j$ must converge absolutely

to satisfy stationarity. This leads to $|G_i| < 1$ and $G_i^{-1} > 1$, i.e., the roots of $\phi(B) = 0$ must greater than 1 to fulfil the stationarity condition.

Next, since $\pi(B)$ can be written as a finite series,

$$\pi(B) = \phi(B) = (1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p)$$

it is not required to restrict the parameters for invertibility.

Moving-average model Consider a special case of (2.20), which only has q non-zero coefficients for the first q shocks. Then, the resulting process MA(q) can be written as,

$$\tilde{z}_t = a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \ldots - \theta_q a_{t-q} \tag{2.27}$$

Similarly, with the backward shift operator, (2.27) can be written as,

$$\begin{aligned} \tilde{z}_t &= (1 - \theta_1 B - \theta_2 B^2 - \ldots - \theta_q B^q) a_t \\ &= \theta(B) a_t \end{aligned} \tag{2.28}$$

To derive the invertibility condition, we connect (2.23) and (2.28),

$$a_t = \theta^{-1}(B)\tilde{z}_t \equiv \pi(B)\tilde{z}_t = \sum_{j=0}^{\infty} \pi_j \tilde{z}_{t-j} \tag{2.29}$$

Using the same method as previously,

$$\pi(B) = \theta^{-1}(B) = \sum_{i=1}^{q} \left(\frac{M_i}{1 - H_i B} \right)$$

where H_i^{-1} are the roots for $\theta(B) = 0$ and M_i are the constants. Therefore, the invertibility is satisfied when the weights $\pi_j = \sum_{i=1}^{q} M_i H_i^j$ converge. In other words, the roots for $\theta(B) = 0$ must be greater than 1 for an invertible process.

As for the stationarity, since $\psi(B)$ can be described by a finite series of $\theta(B)$, it is not necessary to control the parameters for stationarity.

Autoregressive Moving-average model Before formally deriving the next model, the ARMA model, we need to use the result that a finite MA process can be written as an infinite AR process [6]. Consider a MA(1) model,

$$\tilde{z}_t = a_t - \theta a_{t-1} = (1 - \theta B) a_t$$

Divide both sides by $(1 - \theta B)$, and obtain,

$$\frac{1}{(1 + \theta B)} \tilde{z}_t = a_t$$

On the left-hand side, the fraction can be expanded by Taylor expansion if $|\theta| < 1$,

$$(1 + \theta B + \theta^2 B^2 + \theta^3 B^3 + \ldots)\tilde{z}_t = a_t$$

Hence, we have an infinite order AR model with $\pi_j = \theta^j$,

$$\tilde{z}_t = -\theta \tilde{z}_{t-1} - \theta^2 \tilde{z}_{t-2} - \theta^3 \tilde{z}_{t-3} - \ldots + a_t$$

Based on the result, we can conclude that a finite MA process can be described by an infinite AR process. Similarly, a finite AR process can be described by an infinite MA process. As we mentioned at the start of this section, an infinite-order model is not suitable for analysis. Consequently, sometimes we need to combine terms from both processes to obtain a model that

has finite order. The model is the autoregressive moving-average model with order p and q, ARMA(p, q),

$$\tilde{z}_t = \phi_1 \tilde{z}_{t-1} + \ldots + \phi_p \tilde{z}_{t-p} + a_t - \theta_1 a_{t-1} - \ldots - \theta_q a_{t-q} \tag{2.30}$$

Alternatively, it can also be written as,

$$\phi(B)\tilde{z}_t = \theta(B)a_t \tag{2.31}$$

By dividing the terms $\phi(B)$ or $\theta(B)$ for both sides in (2.31), we can derive an infinite autoregressive expression with $\pi(B) = \phi(B)\theta^{-1}(B)$ and an infinite moving-average representation with $\psi(B) = \phi^{-1}(B)\theta(B)$. Therefore, based on the discussion of stationarity and invertibility for the AR and MA models, we can conclude that the ARMA process will be stationary if the roots of $\phi(B) = 0$ are outside the unit circle (greater than 1 absolutely), and invertible if the roots of $\theta(B) = 0$ are in the same condition [6].

2.3.3 Linear Non-stationary Model

Eventually, we will introduce the main model implemented in this report, the autoregressive integrated moving-average (ARIMA) model. Provided by Box et al. [6], the ARMA process is stationary when all the roots of $\phi(B) = 0$ are outside the unit cycle, and significant non-stationary if roots are inside the unit cycle. The remaining case, where the roots are equal to one, gives a valuable model representing a nonstationary series with homogeneous variance. Suppose $\varphi(B)$ is the nonstationary autoregressive operator with d roots equal one while others are greater than one, then we have,

$$\varphi(B)\tilde{z}_t = \phi(B)(1-B)^d \tilde{z}_t = \theta(B)a_t \tag{2.32}$$

where $\phi(B)$ is the autoregressive operator introduced in the section above. Denote the difference operator $\nabla = 1 - B$. Consequently, $\nabla^d \tilde{z}_t = \nabla^d z_t$ when $d \geq 1$. Simplify (2.32) to obtain,

$$\phi(B)\nabla^d z_t = \theta(B)a_t \tag{2.33}$$

Therefore, Equation (2.33) is the ARIMA process for nonstationary series defined by Box et al. [6], denoted as ARIMA(p, d, q).

The letter "I" inserted in ARMA indicates that Equation (2.33) can be obtained by summing (integrating) an "ARMA" process d times. The fiction ARMA process is defined as,

$$\phi(B)w_t = \theta(B)a_t \tag{2.34}$$

where w_t is the resulting series after d times differencing of z_t ($w_t = \nabla^d z_t$). In addition, the summation of w_t is,

$$z_t = S^d w_t \tag{2.35}$$

Here S^d is the infinite summation operator in order d. The first order case is,

$$S \cdot x_t = \sum_{h=-\infty}^{t} x_h = (1 + B + B^2 + \cdots)x_t = (1-B)^{-1}x_t = \nabla^{-1}x_t$$

Then, the second order operator applies S again on the result of the first order operator, and so on for the higher order. In summary, Equation (2.35) indicates that the ARIMA model (2.33) can be obtained by summing d times of Equation (2.34). However, since an infinite sum may lead to divergence, a finite summation operator,S_m, is used in the definition of the ARIMA model [6],

$$S_m = (1 + B + B^2 + \cdots + B^{m-1}) \equiv \frac{1 - B^m}{1 - B}$$

Moreover, Box et al. [6] observed in some periodic data that correlation not only appears in successive months in a particular year but also in the same month across successive years. While the ARIMA model (2.33) could capture the correlation between successive data, the correlation between seasons has been neglected. In order to better capture the repeated pattern and produce a better fit for the ARIMA model, they suggested multiplying the non-seasonal ARIMA model by a seasonal component.

To model the correlation between seasons, we could use a similar model as the (2.33), yet with different specifications,

$$\Phi_P(B^S)\nabla_S^D z_t = \Theta_Q(B^S)\alpha_t \tag{2.36}$$

Here, the model captures the correlation between data at time t and $t-s$, where S and s refer to the length of each season, Φ_P refers to the seasonal AR operator, ∇_S^D represents the seasonal differencing operator, and Θ_Q stands for the seasonal MA operator. Similarly, the model for $t-1$ is,

$$\Phi_P(B^S)\nabla_S^D z_{t-1} = \Theta_Q(B^S)\alpha_{t-1} \tag{2.37}$$

and so on. The error terms in these models, $\alpha_t, \alpha_{t-1}, \ldots$, are not uncorrelated in general. Hence, Box et al. [6] suggested using another ARIMA model to capture the relationship of these successive error terms,

$$\phi_p(B)\nabla^d \alpha_t = \theta_q(B)a_t \tag{2.38}$$

Therefore, if substitute (2.38) into (2.37), we can then obtain the full expression of the seasonal ARIMA model,

$$\phi_p(B)\Phi_P(B^S)\nabla^d\nabla_S^D z_t = \theta_q(B)\Theta_Q(B^S)a_t \tag{2.39}$$

In a simple way, the seasonal ARIMA model can be written as, ARIMA$(p,d,q)(P,D,Q)_S$.

2.4 Model Specification

Having introduced the ARIMA model family in the previous section, we will then briefly explain how the ARIMA model is specified. First, we will explain how to identify the model parameters (p,d,q,P,D,Q,S), followed by the introduction of the estimation methods of the coefficients. Finally, we will present how to forecast using the specified ARIMA.

2.4.1 Identification

As in Equation (2.39), the seasonal ARIMA model has 7 parameters to be determined. To start with, we can first determine the season length S by visual inspection. In Section 2.1.1 and 2.1.3, we mentioned that seasonality is a repeated behaviour of the time series data and can be identified in its plot and its ACF plot. Once the season length is fixed, the degree of differencing, d and D can be identified. According to Box et al. [6], when fixing a value for d, normally $d \in \{0,1,2\}$ in practice, if the values in the ACF plot of $\nabla^d z_t$ decrease quickly to near zero, then this degree of differencing is appropriate. As for the seasonal degree of differencing D, it is equal to the length of the season.

In terms of the parameters for AR and MA process (p,q,P,Q), they can also be determined by checking the ACF plot as well as the PACF plot. The PACF or partial ACF, measures the correlation between an observation z_t and z_{t-k}, after controlling for observations in between, $z_{t-k+1}, \ldots, z_{t-1}$. For instance, if the data follows the ARIMA$(p,d,0)$ model, its ACF is exponentially decreasing or sinusoidal while its PACF has a spike at lag p with no more spikes after p; if the data follows the ARIMA$(0,d,q)$ model, the behaviour of its ACF and PACF is reverse, with a significant spike in ACF at lag q [29]. Similarly, for seasonal case, if the data follows the ARIMA$(0,0,0)(0,0,1)_{12}$ model, its ACF only has a significant spike at lag 12 while its PACF decreases exponentially at seasonal lags $(12,24,36,\ldots)$; if the model is ARIMA$(0,0,0)(1,0,0)_{12}$,

the spike will appear in the PACF at lag 12 and ACF will decay exponentially at seasonal lags [29].

Although visual inspection using ACF and PACF is a useful method in identifying the parameters initially, in some cases with mixed models, this method may fail to provide clear identification. However, this is not a serious problem, as the model specification is not fixed, it requires further evaluation and modification if needed [6]. In spite of this, we can select a model using the model selection criteria. For example, the Akaike information criterion (AIC), Bayesian information criterion (BIC), and AIC with correction (AICc) [1, 42, 44]. These information criteria are defined as,

$$\begin{aligned} \text{AIC} &= -2ln(\hat{\mathcal{L}}) + 2k \\ \text{BIC} &= -2ln(\hat{\mathcal{L}}) + kln(n) \\ \text{AICc} &= \text{AIC} + \frac{2k^2 + 2k}{n - k - 1} \end{aligned}$$

where $\hat{\mathcal{L}}$ is the maximum value of the likelihood function for the model, k is the number of the parameters, and n is the size of the observations. In the application of this approach, we could calculate the information criterion of a range of models and select the model with the minimum information criterion. Despite minimising these information criteria strike a balance between bias and variance (too few lags lead to high bias and too many estimated coefficients lead to high variance), Box et al. [6] argued that when using this approach, the models have to be estimated by the maximum likelihood method (will be discussed in the next section), which requires intensive computation. Nevertheless, for simplicity, we use a built-in function in R, based on either AIC, BIC, or AICc, to automatically select the parameters in this report [30].

2.4.2 Estimation

When the parameters are selected, the coefficients can then be estimated through multiple methods, such as maximum likelihood estimation (MLE) or conditional sum of squares (CSS) [6, 29]. The MLE method produces the estimates of coefficients that maximise the probability of obtaining the observations. As for the ARIMA specifically, MLE is similar to the least squares method that minimises the sum of squared errors [29]. On the other hand, the CSS method estimates the coefficients by minimising the sum-of-squares function. Assume we have a series of $N = n + d$ observation $z_{-d+1}, \ldots, z_0, z_1, \ldots, z_n$, and define a new series $w_t = \nabla^d z_t$, $w_1, \ldots, w_n$. Then, estimating the coefficients ϕ and θ of z_t is the same as estimating the coefficients of a stationary ARMA model on w_t [6].

$$a_t = \tilde{w}_t - \phi_1 \tilde{w}_{t-1} - \ldots - \phi_p \tilde{w}_{t-p} + \theta_1 a_{t-1} + \ldots + \theta_q a_{t-q} \tag{2.40}$$

where $\tilde{w}_t = w_t - \mathbb{E}[w_t]$. We cannot plug w_t into (2.40) at the start, since the existence of differencing. Thus, we suppose the starting values w_* of w_t of size p and a_* of a_t of size q are given. Then, $a_t(\phi, \theta | w_*, a_*, w)$, $t = 1, \ldots, n$, can be calculated. If we assume errors are normally distributed, the log-likelihood function of the parameters conditional on the starting values (w_*, a_*) is,

$$\mathcal{L}_*(\phi, \theta, \sigma_a^2) = -\frac{n}{2} ln(\sigma_a^2) - \frac{S_*(\phi, \theta)}{2\sigma_a^2} \tag{2.41}$$

and the conditional sum-of-squares function is,

$$S_*(\phi, \theta) = \sum_{t=1}^{n} a_t^2(\phi, \theta | w_*, a_*, w) \tag{2.42}$$

By observing (2.41) and (2.42), we have that the data only appears in the S_* within $\mathcal{L}_*$. Hence, maximising the likelihood ($\mathcal{L}_*$) is the same as minimising the conditional sum-of-squares (S_*), and the estimates given by this are the conditional least squares estimates.

When the series is long, the unconditional likelihood can be approximated by the conditional likelihood with suitable starting values, w_* and a_*. One common choice for the values is setting the starting values equal to their unconditional expectation, otherwise, if $\mathbb{E}[w_t] = \mu = 0$ is not appropriate, take $w_* = \bar{w}$ [6]. However, this choice for the starting value might be inappropriate, when some of the roots of $\phi(B) = 0$ are close to the unit circle. To address this issue, one feasible way is to use (2.40) to calculate a_t, by setting $a_t = 0$ with $t = 1, \ldots, p$. Then, the sum of a_t^2 gives the CSS.

In our report specifically, we use the default function settings of auto.arima$(\cdot)$, which uses CSS to determine the starting values, w_* and a_*, and MLE to estimate the coefficients [30].

2.4.3 Forecasting

After identifying the parameters and estimating the coefficients, we obtain an ARIMA model for a given time series data. Then, we can produce the point forecasts based on the specified model, in three steps [29],

Step 1: Expand the equation for the ARIMA model (2.39), move the terms representing observations (z_t) to the left, and move the rest of the terms to the right.

Step 2: Replace time index t with $T + h$ for all terms, where h stands for the period ahead of the time of the latest observation.

Step 3: On the right-hand side of the equation, replace all future observations with predicted observations, all future errors with zero, and all past errors with observed errors at that time.

For example, consider a non-seasonal ARIMA$(3, 1, 1)$ model given by Hyndman and Athanasopoulos [29]. First, expand the model equation, apply the backward operator, and move all terms except z_t to the right.

$$(1 - \hat{\phi}_1 B - \hat{\phi}_2 B^2 - \hat{\phi}_3 B^3)(1 - B) z_t = (1 + \hat{\theta}_1 B) a_t$$

$$z_t = (1 + \hat{\phi}_1) z_{t-1} - (\hat{\phi}_1 - \hat{\phi}_2) z_{t-2} - (\hat{\phi}_2 - \hat{\phi}_3) z_{t-3} - \hat{\phi}_3 z_{t-4} + a_t + \hat{\theta}_1 a_{t-1}$$

Then, replace t with $T + h$, and start with $h = 1$,

$$z_{T+1} = (1 + \hat{\phi}_1) z_T - (\hat{\phi}_1 - \hat{\phi}_2) z_{T-1} - (\hat{\phi}_2 - \hat{\phi}_3) z_{T-2} - \hat{\phi}_3 z_{T-3} + a_{T+1} + \hat{\theta}_1 a_T$$

Assume we only have information about the data up to time $T + 1$, then a_{T+1} is set to zero. Thus, the forecast $\hat{z}_{T+1|T}$ is,

$$\hat{z}_{T+1|T} = (1 + \hat{\phi}_1) z_T - (\hat{\phi}_1 - \hat{\phi}_2) z_{T-1} - (\hat{\phi}_2 - \hat{\phi}_3) z_{T-2} - \hat{\phi}_3 z_{T-3} + \hat{\theta}_1 a_T$$

Similarly, at $T + 2$, z_{T+1} is replaced with the forecast at $T + 1$, and a_T is replaced with zero. This gives the forecast,

$$\hat{z}_{T+2|T} = (1 + \hat{\phi}_1) \hat{z}_{T+1|T} - (\hat{\phi}_1 - \hat{\phi}_2) z_T - (\hat{\phi}_2 - \hat{\phi}_3) z_{T-1} - \hat{\phi}_3 z_{T-2}$$

The process continues until the desired forecast period h is achieved.

In practice, the formula for the ARIMA model is complicated and may contain multiple terms, e.g. ARIMA$(2, 1, 1)(1, 1, 1)_{12}$. It is not feasible to calculate the forecast by hand or through recursive functions in R. Hence, in this report, we use the forecast$(\cdot)$ in the forecast package to perform forecasting automatically with a designated prediction period.

2.5 Conclusion

In this chapter, we first introduced the time series characteristics that require investigation and ensure before bootstrap resampling. There are three characteristics, which are pattern, stationarity, and autocorrelation. Identifying the patterns helps us understand the behaviour of the time series, stationarity is the prerequisite for bootstrap, and checking autocorrelation gives us information about the dependence structure. Then, we presented the Box-Cox transformation and data decomposition with Loess and STL as the data preparation techniques for guaranteeing stationarity. Next, we detailed the development of the ARIMA model from stationary models, AR, MA, to ARMA, and then to nonstationary model ARIMA. Finally, we concluded this chapter with an explanation of the procedure of identifying the parameters with visual inspection and information criteria, estimating the coefficients using MLE and CSS methods, and forecasting with the specified ARIMA model.

Chapter 3

Bootstrap

In the previous chapter, we introduced relevant information in time series forecasting. We began by discussing the three characteristics that are required to be analysed in time series forecasting, then introduced two commonly used methods in forecasting, and concluded with the introduction of stationary and non-stationary time series models. Indeed, applying the data processing methods and models mentioned in the previous chapter can provide good forecasts. The ARIMA model is widely used due to its flexibility and its ability to capture autocorrelation in non-stationary data. However, in this report, we want to go a step further. Inspired by Bergmeir et al. [4]'s pioneering research, we hope to improve the accuracy of ARIMA forecasts by applying bootstrap aggregating, an approach to enhance the accuracy of machine learning predictors, and bootstrap, a method for generating samples and estimating sample statistics. The detailed steps for combining these methods will be introduced at the end of this chapter.

In general, the bootstrap method is a resampling technique that estimates the distribution of a statistic over a sample. It involves repeatedly sampling, with replacement, from the observed dataset to create a large number of bootstrap samples of the same size as the original dataset. Originally established by Efron in 1979 based on the jackknife resampling technique, the bootstrap method was solely concerned with the estimation of the variance and bias of the sample [16, 17]. However, after more than 40 years of development, the bootstrap theory has expanded to not only estimate almost any statistics of a sample distribution (bias, variance, confidence intervals, prediction errors, etc.) but also enhance accuracy. Proposed by Breiman [8], bootstrap aggregating, or in short bagging, was originally designed to improve the stability and accuracy of machine learning algorithms. However, Bergmeir et al. [4] suggested that this method can also be used to enhance the accuracy of time series forecasting. This is achieved by combining multiple forecasts of bootstrapped series, which are generated by bootstrap designed for autocorrelated data. In the field of time series, based on Efron's Bootstrap for independent samples, Künsch [33] improved Efron's method to establish the moving block bootstrap, which maintains the dependence structure within data, enabling statistics of time series samples to be estimated. Furthermore, Bühlmann [9], inspired by the sieve method, proposed another resampling method for time series data called the sieve bootstrap, which aims at resampling the residuals after fitting a time series model.

In this chapter, we will first introduce Efron's bootstrap method followed by the moving block bootstrap and sieve bootstrap. Then we will introduce the method used to improve the forecast accuracy, bagging. Finally, we will conclude this chapter with the detailed steps of our proposed method for improving the forecast accuracy for the ARIMA using bagging with bootstrap.

3.1 Efron's Bootstrap

Efron [16] introduced the bootstrap method in 1979 as an improvement to former resampling

techniques. Due to its flexibility and simplicity, the bootstrap method has been widely employed in statistical inference to estimate the sampling distribution of a statistic. It addresses the difficulty of making inferences about population metrics in situations where the underlying distribution is unknown or complicated while maintaining generality to various statistics.

The bootstrap method is straightforward, in terms of its implementation. Consider a random sample of size n, $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$, sampled independently from an unknown distribution F. We are interested in some statistical properties of F, such as mean and standard deviation, denoted by a parameter $\theta(F)$. Assume the estimator of $\theta(F)$ is $t(\mathbf{X})$, which can be the sample mean, the sample standard deviation or others, a random variable can be defined as [16],

$$R(\mathbf{X}, F) = t(\mathbf{X}) - \theta(F)$$

Then, the distribution of R can be estimated when observing $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$. Aiming at estimating the distribution via bootstrap, the following steps are required [16]:

Step 1: Based on the observed data $\mathbf{x}$, construct an empirical cumulative distribution (ecdf) $\hat{F}$, with probability $\frac{1}{n}$ on each data point $x_1, x_2, \ldots, x_n$.

$$\hat{F}(x) := \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{x_i \leq x\} \quad \forall x \in \mathbb{R}$$

where $\mathbb{1}\{x_i \leq x\}$ is an indicator function.

$$\mathbb{1}\{x_i \leq x\} = \begin{cases} 1 & x_i \leq x \\ 0 & \text{otherwise} \end{cases}$$

Step 2: Draw a size n sample, $\mathbf{X}^* = \{X_1^*, X_2^*, \ldots, X_n^*\}$, from the fixed distribution $\hat{F}$ independently, which is equivalent to sampling uniformly without replacement from the data. The reason is that, sampling from $\{x_1, x_2, \ldots, x_n\}$ with probability $\frac{1}{n}$ has cumulative density function $F(x) = \sum_{t:t\leq x} p(t)$, where the probability here is,

$$p(t) = \begin{cases} \frac{1}{n} & \text{if } t \in \{x_1, \ldots, x_n\} \\ 0 & \text{otherwise} \end{cases}$$

Then, expand the sum to obtain $F(x) \equiv \hat{F}(x)$

Step 3: With the bootstrapped distribution, $R^* = R(\mathbf{X}^*, \hat{F})$, the distribution of $R(\mathbf{X}, F)$ can be approximated.

The rationale behind using ecdf to approximate cdf is that as more and more data is collected, the ecdf will converge to the true cdf, according to the Glivenko-Cantelli theorem [45]:

Theorem 3.1 *(Glivenko-Cantelli theorem) Let $X_1, X_2, \ldots, X_n$ be a random sample drawn from a distribution with cdf $F(x)$. And, the ecdf $\hat{F}(x)$ is built upon the random sample. Then,*

$$\sup_{x\in\mathbb{R}} |\hat{F}(x) - F(x)| \to 0 \quad \text{as } n \to \infty$$

The convergence is in probability.

Thus, given a random sample $\mathbf{X}$, the distribution of R can be estimated, yet a few points still need to be considered. According to Efron [16], the main obstacle to implementing bootstrap lies in the calculation of the bootstrap distribution R^*. He proposed three possible methods to calculate:

1. Direct theoretical calculation, as shown in Example 3.1.

2. Monte Carlo approximation: rather than using one bootstrapped sample $\mathbf{X}^*$, multiple realisations are generated, for instance, $\mathbf{X}^{*1}, \mathbf{X}^{*2}, \ldots, \mathbf{X}^{*B}$. Then, the actual distribution of R^* is approximated through the histogram given values of with the value of $R(\mathbf{X}^{*1}, \hat{F}), \ldots, R(\mathbf{X}^{*B}, \hat{F})$.

3. Taylor expansion: expands the defined vector $\mathbf{P}^* = (P_1^*, \ldots, P_n^*)$ around value $\mathbf{P}^* = (\frac{1}{n}, \ldots, \frac{1}{n})$, where $P_i^* = \frac{N_i}{n}$ and N_i is the number of ith bootstrapped sample X_i^* equals to ith observations x_i. Then, the bootstrap distribution can be approximated. Since this method is beyond the scope of our report, the details are omitted.

Example 3.1 *Consider an example given by Efron [16], an unknown distribution F of a random variable X puts all of its mass on one or zero. We are interested in the probability of X being one, i.e, $\theta(F) = \mathbb{P}(X = 1)$. Also, assume the estimator for $\theta(F)$ is $\bar{X} = \sum_{i=1}^{n} X_i/n$. Then, the random variable of interest is,*

$$R(\mathbf{X}, F) = \bar{X} - \theta(F)$$

When observed $\mathbf{X} = \mathbf{x} = (x_1, \ldots, x_n)$, the bootstrapped sample, $\mathbf{X}^* = (X_1^*, \ldots, X_n^*)$ is generated randomly from $\hat{F}$ (the empirical distribution constructed by giving equal probability to the observations $\mathbf{x}$). Hence, the bootstrapped sample has $\theta(\hat{F}) = \bar{x}$ probability of being one. Therefore, the bootstrap sampling process can be considered as sampling from a binomial distribution with probability $\bar{x}$. The bootstrap distribution,

$$R^* = R(\mathbf{X}^*, \hat{F}) = \bar{\mathbf{X}}^* - \bar{x}$$

has expectation and variance,

$$\mathbb{E}(R^*) = \mathbb{E}(\bar{\mathbf{X}}^* - \bar{x}) = 0 \quad Var(R^*) = \frac{\bar{x}(1-\bar{x})}{n}$$

The bootstrap distribution we desired is obtained.

Following the procedures above, the bootstrap distribution can be estimated without further knowledge about the data distribution. The non-parametric feature makes the bootstrap method a powerful tool when facing a situation where the parameters of interest are unknown or complex or the sample size is insufficient to make statistical inferences. For instance, apart from estimating the statistics of the sample (mean, variance, standard error, etc), the bootstrap method can also be used to estimate the confidence interval of the statistics of people's interest, such as the mean, median, and variance [14]. It can also be used to perform hypothesis tests by generating a distribution of the test statistic under the null hypothesis, which allows for the computation of p-values without relying on traditional distributional assumptions [14].

In addition, bootstrap is not only successful in the field of statistics, but it is also widely employed in data forecasting in the financial field, hypothesis testing in the biological field, and clustering in the field of data science. For example, in the econometrics domain, the bootstrap method can be applied to a linear regression model to estimate its coefficients [14]. This is done by resampling the pairs of the explanatory (independent) variable and the response (dependent) variable and fitting a linear regression model to give estimates. The bootstrap method in this case requires no further assumption on the variance homogeneity (the variances of two or more observations are equal) and linearity between independent and dependent variables, which offers a benefits in providing good estimates for the coefficients [14].

3.2 Moving Block Bootstrap

In the previous sections, the basic idea of using the bootstrap method to estimate has been presented. The fundamental assumption of Efron's bootstrap method is that samples are independent of each other. However, when the data is correlated, Efron's bootstrap method would give incorrect results, due to the ignorance of data dependence. Since the focus of this report is on time series data, which exhibits serial dependence generally, we cannot use Efron's bootstrap method to resample data but use the other bootstrap methods instead. Proposed by Künsch [33] in 1989, the moving block bootstrap method is a creative extension of the bootstrap idea to stationary dependent data. The term, block, represents a number of consecutive data points in the data. Given the block length fixed (the number of data points in a block fixed), the word, moving, indicates sampling blocks randomly as if they are iid. The method is practical for time series data as it preserves the data correlation within the blocks.

3.2.1 Moving Block Bootstrap Demonstration

Mathematically, based on process proposed by Künsch [33], suppose we have samples, $X_1, \ldots, X_n$, from a stationary process $(X_t)_{t\in\mathbb{Z}}$. Then, similar to the bootstrap method for iid data, Künsch constructs an empirical distribution according to the observations. However, to represent the distribution for blocks, the marginal empirical distribution is m-dimensional:

$$\rho_N^m = \frac{1}{N-m+1}\sum_{t=0}^{N-m}\delta_{(X_{t+1},\ldots,X_{t+m})} \tag{3.1}$$

where δ is the Dirac delta function representing the point mass for the value in the bracket. Alternatively, let Y_t denotes a block, where $Y_t = (X_t, X_{t+1}, \ldots, X_{t+m-1})$, $t = 1, \ldots, n-m+1$ and $n = N-m+1$, the marginal is,

$$\rho_N^m = \frac{1}{n}\sum_{t=1}^{n}\delta_{Y_t} \tag{3.2}$$

Thus, the statistics can be written as $T_N(X_1, X_2, \ldots, X_n) = T(\rho_N^m)$.

Before proceeding to the bootstrap procedures, several assumptions should be established with respect to the statistics T_N and the stationary process $(X_t)_{t\in\mathbb{Z}}$, in order to express the asymptotic properties of the variance of T_N.

(A1) Denote the marginal distribution of $(X_1, X_2, \ldots, X_m)$ to be F^m. Then, then $T_N \xrightarrow{a.s.} T(F^m)$.

(A2) $\forall y \in \mathbb{R}^m$, the influence function $IF(y, F^m) = \lim_{\epsilon\downarrow 0}\frac{1}{\epsilon}[T((1-\epsilon)F^m + \epsilon\delta_y) - T(F^m)]$ exists [23].

(A3) As $n \to \infty$, $\frac{1}{\sqrt{n}}\sum_{t=1}^{n} IF(Y_t, F^m)$ tends to normal with mean 0 and variance σ_{as}^2,

$$\sigma_{as}^2 = \sum_{k\in\mathbb{R}^m}\mathbb{E}[IF(Y_0, F^m)IF(Y_t, F^m)]$$

(A4) The statistics can be decomposed linearly, $T(\rho_N^m) = T(F^m) + \frac{1}{n}\sum_{t=1}^{n} IF(Y_t, F^m) + R_N$, where R_N is the remainder term with order $O_p(n^{-1/2})$.

From the assumptions, we can interpret that if (X_t) is ergodic, i.e, the mean of observations equals the mean of the whole process, then $\rho_N^m \to F^m$ weakly and (A1) is deduced from weak continuity of T. Moreover, from (A3) and (A4), $\sqrt{n}(T_N - T(F^m)) \sim N(0, \sigma_{as}^2)$ as n increases, can be deduced.

With assumptions established, continue to the moving block bootstrap. Let the block length be l, assume $n = kl$, and $k \in \mathbb{R}$. The marginal for bootstrap is,

$$\rho_N^{m,*} = \frac{1}{n} \sum_{j=1}^{k} \sum_{t=S_j+1}^{S_j+l} \delta_{Y_t}, \tag{3.3}$$

where $S_1, S_2, \ldots, S_k$ are iid random variables following uniform distribution on the set $\{0, 1, \ldots, n-l\}$, indicating the indices of each block. Hence, the bootstrapped statistic is given as $T_N^* = T(\rho_N^{m*})$.

As for the distribution of interest, the unknown distribution $T_N - T(F_m)$ can be approximated by $T_N^* - T_N$, based on the assumptions above, with blocks $Y_1, \ldots, Y_n$ fixed and sampling indices $S_1, \ldots, S_k$ varied. Then, the estimated variance for the bootstrapped statistic is,

$$\hat{\sigma}_{boot}^2 = Var(T_N^*) = \mathbb{E}[(T_N^* - \mathbb{E}[T_N^*])^2] \tag{3.4}$$

Noticeably, in practice, the variance estimation $\hat{\sigma}_{boot}^2$ and the distribution of $T_N^* - T_N$ are often obtained from simulation (e.g. Monte Carlo simulation).

Aimed at providing a visual demonstration of the sampling process for the moving block bootstrap method, an example of generating two bootstrapped samples, X^{*1} and X^{*2}, from a time series $X = (x_1, x_2, \ldots, x_{15})$, is given in Figure 3.1,

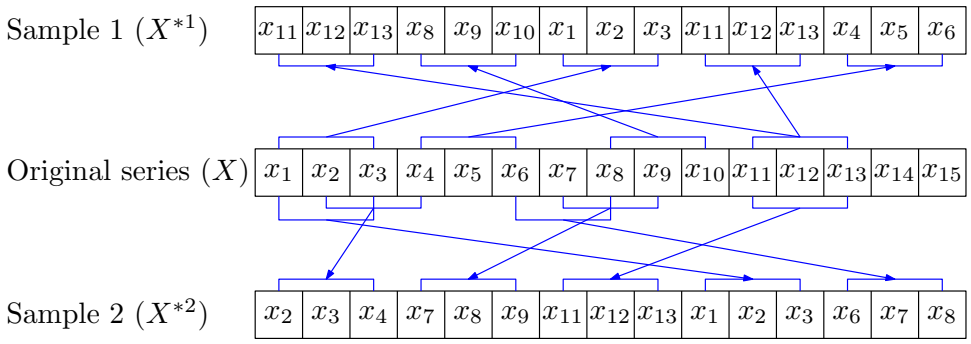

Figure 3.1: Example of the Moving Block Bootstrap [18]

In Figure 3.1, we assume the block length is 3. The elements in the original series are divided into groups of 3 consecutive elements $(x_t, \ldots, x_{t+2})$, where $t = 1, \ldots, 13$. Then, the required number of groups is drawn with replacements to form a new series. In this case, the required number is 5. Noticeably, the choice for the block length should be considered carefully. Our choice will be discussed in detail in Section 4.3.2.

3.2.2 Remainder Moving Block Bootstrap

By partitioning time series data and selecting randomly, the moving block bootstrap method is useful in preserving the autocorrelation relation within the blocks. Nevertheless, it neglects the correlation between adjacent sampled blocks and fails to ensure that any element in the data can be placed in any place in the bootstrap sample. In order to ensure the robustness and effectiveness of the bootstrap method, we need to secure the comprehensive coverage of the original data. One possible solution is to modify the resampling techniques. As proposed in the paper by Bergmeir et al. [4], the number of blocks to draw from the original data should be $\lfloor n/l \rfloor + 2$, rather than n/l. To maintain the same length as the sample, a random number of data, between 0 to $l - 1$, is discarded from the beginning. Last, delete a required number of data from the end to obtain the bootstrap sample of length n.

The rationale behind the modified resampling techniques is simple. By extracting $\lfloor n/l \rfloor + 2$ blocks with replacement, the new series not only has redundant space for the next step but also

ensures that every value in the original sample can be placed anywhere in the bootstrap sample. By randomly removing 0 to $l-1$ data from the beginning and deleting data from the end, the bootstrap sample can guarantee that the first and the last elements are not the elements at the block boundaries. This is crucial for the bootstrap sample to mimic the real behaviour of the stationary process $(X_t)_{t\in\mathbb{Z}}$.

Moreover, Bergmeir et al. [4] also proposed bootstrapping the remainders yielded from the Loess regression or STL decomposition rather than the original data. The reasons are three-fold. First, time series data often exhibits non-stationarity due to the existence of seasonality and trend. By separating these three components, the fundamental prerequisite for bootstrap, stationarity, is more likely to be achieved. Second, bootstrapping the original data without decomposition could disrupt the inherent structure of the time series, particularly the trend and seasonal patterns. By decomposing the series first, these structures are preserved in their respective components, and the bootstrapping of the remainder focuses on the random fluctuations around these structures. Third, the remainder component represents the irregular fluctuations or noise in the data after accounting for trend and seasonality. By bootstrapping this component, we could better capture and model the inherent randomness in the data more effectively [4].

In this report, specifically, we will follow Bergmeir et al.'s version of moving block bootstrap, which is sampling $\lfloor n/l \rfloor + 2$ blocks from the remainder component of the original sample. The detailed steps are as follows,

Step 1: Use Loess regression or STL decomposition to separate the original data into seasonal, trend, and remainder components. Denote the remainder component to be $X_1, \ldots, X_n$.

Step 2: Sample $\lfloor n/l \rfloor + 2$ blocks with length l from blocks $\{X_t, \ldots, X_{t+l}\}, \quad t = 1, \ldots, n-l$, and form a series.

Step 3: Randomly remove the first 0 to $l-1$ elements from the obtained series.

Step 4: Delete the required number of the elements from the tail of the series to shorten the series to n elements and get $X_1^{*b}, \ldots, X_n^{*b}, \quad b = 1, \ldots, B$.

Step 5: Assemble seasonal and trend components with the series from step 4 to obtain b^{th} bootstrapped sample. Go back to step 2, exit until $b = B$

3.3 Sieve Bootstrap

Although the moving block bootstrap is a robust non-parametric model for time series estimation, flaws exist as it fails to capture the complete correlation between different blocks and produces a sample that might not be stationary [9]. In order to improve the block bootstrap and produce a stationary sample, Politis and Romano [40] proposed an improved version of block bootstrap, named stationary bootstrap. Rather than fixing the block length as Künsch [33] assumed, the block length in stationary bootstrap follows a geometric distribution with a fixed tuning parameter. It seems to be a promising improvement of the original moving block bootstrap. However, Bühlmann [9] criticised that the tuning parameter in Politis and Romano's method is hard to control. Consequently, with the aim to address the cons while maintaining the pros of the block bootstrap, he established another resample scheme for time series data called sieve bootstrap. The fundamental concept of the method is approximating an infinite-dimensional non-parametric model by fitting it with a finite-dimensional parametric model, which follows the idea of the sieve method by Grenander [21]. Generally, the method features fitting an autoregressive model, for which the order is determined via minimising the Akaike information criterion (AIC), on the stationary data and resampling the centralised residuals. Then, the bootstrapped residuals are combined with the auto-regressive model to formulate a bootstrap sample.

3.3.1 Sieve Bootstrap Demonstration

Before introducing the detailed bootstrap procedures for the sieve bootstrap, we first need to select a suitable parametric model for the stationary process. Let $(X_t)_{t\in\mathbb{Z}}$ be a stationary stochastic process and $\mathbb{E}(X_t) = \mu_X$. Then, the difference between the process and its mean can be written as a moving-average (MA) process based on Wold's decomposition theorem [2]:

Theorem 3.2 *(Wold's decomposition theorem) If $(X_t)_{t\in\mathbb{Z}}$ is a stationary stochastic process with mean μ_X, then $X_t - \mu_X$ can be written as a one-sided infinite-order moving-average process,*

$$X_t - \mu_X = \sum_{j=0}^{\infty} \theta_j \epsilon_{t-j} \quad \textit{with} \quad \theta_0 = 1, t \in \mathbb{Z} \tag{3.5}$$

where θ_j are the coefficients and $(\epsilon_t)_{t\in\mathbb{Z}}$ are the uncorrelated random errors (white noise).

Given the process in (3.5), we also need to assume invertibility to the process, that is, $\sum_{j=0}^{\infty} |\theta_j|$ converges. This assumption is crucial for following interpretation and forecasting, as it ensures the uniqueness of the identification of the autocorrelation function. Thus, as mentioned in Section 2.3.1, X_t can be represented as a one-sided infinite-order autoregressive (AR) process:

$$\sum_{j=0}^{\infty} \phi_j \left(X_{t-j} - \mu_X\right) = \varepsilon_t, \quad \text{with} \quad \phi_0 = 1, t \in \mathbb{Z}, \sum_{j=0}^{\infty} \phi_j^2 < \infty \tag{3.6}$$

Although moving-average approximation can be adopted with (3.5), autoregressive approximation from (3.6) has better performance in various cases [5, 24, 28].

Now, with a suitable parametric model, we can proceed to the details of bootstrap procedures. Suppose a sample $X_1, X_2, \ldots, X_n$ from stationary process $(X_t)_{t\in\mathbb{Z}}$.

In the first step, fit an AR model (3.6), with order p (AR(p)) selected according to AIC, to the sample. The order of the model, p, increases as sample size increases and eventually tends to infinity if $n \to \infty$. Then, estimate the coefficient $\hat{\phi}_{1,n}, \ldots, \hat{\phi}_{p,n}$ of (3.6) by some estimates, for instance, the Yule-Walker estimates. This gives,

$$\hat{\varepsilon}_{t,n} = \sum_{j=0}^{p} \hat{\phi}_{j,n} \left(X_{t-j} - \bar{X}\right), \quad \hat{\phi}_{0,n} = 1 \quad (t = p+1, \ldots, n) \tag{3.7}$$

In the second step, establish a resampling scheme on the AR approximation yielded above. Given the residuals in (3.7), centre the residuals:

$$\tilde{\epsilon}_{t,n} = \hat{\epsilon}_{t,n} - \frac{1}{n-p} \sum_{t=p+1}^{n} \hat{\epsilon}_{t,n} \tag{3.8}$$

With the centred residuals $\tilde{\epsilon}_{t,n}$, following the idea introduced in the Section 3.1, we construct an empirical cdf $\hat{F}_{\epsilon,n}$,

$$\hat{F}_{\epsilon,n}(\cdot) = \frac{1}{n-p} \sum_{t=p+1}^{n} \mathbb{1}_{[\tilde{\epsilon}_{t,n} \le \cdot]} \tag{3.9}$$

In the final stage, resampling the residuals and substituting them into fit models to obtain bootstrap samples. For any $t \in \mathbb{Z}$, draw iid samples ϵ_t^* from ecdf $\hat{F}_{\epsilon,n}$ in (3.9). Finally, define the bootstrapped process $(X_t^*)_{t\in\mathbb{Z}}$ as,

$$\sum_{j=0}^{p} \hat{\phi}_{j,n} (X_{t-j}^* - \bar{X}) = \epsilon_t^* \quad \hat{\phi}_{0,n} = 1. \tag{3.10}$$

In practice, to sample $X_1^*, \ldots, X_n^*$, we often start with a value, like $\bar{X}$, and then generate an AR(p) process according to (3.10). Until stationarity is reached, throw away burn-in values to get a stationary process. Since the sieve bootstrap resample scheme is established, we can consider the statistics of interest $T_n(X_1, \ldots, X_n)$. By the plug-in principle [18], the bootstrap statistics can be written as,

$$T_n^* = T_n(X_1^*, \ldots, X_n^*)$$

Compared with the moving block bootstrap, the sieve bootstrap has three advantages [9]. First of all, the sieve bootstrap does not neglect the dependent structure between the sampled blocks as the moving block bootstrap and its samples are not the subsets of the original data. Second, as discussed in Section 3.2.1, when applying moving block bootstrap, the observations need to be vectorised (define $Y_t = (X_t, X_{t+1}, \ldots, X_{t+m-1})$, $t = 1, \ldots, n-m+1$) and the m-dimensional empirical cdf, F^m, is constructed based on Y_t. With different dimensions, different vectorised observations and empirical cdf need to be used, while the sieve bootstrap is free of these constructions and enjoys the properties of the plug-in principle [18]. Third, in contrast to the moving block bootstrap, the sieve bootstrap has the advantage of sampling data with missing values and unequally spaced data. The sieve bootstrap samples the data as if the data is complete or equally spaced, and then assigns empty values to the places in the sample where the data is missing. On the other hand, blocks with different lengths need to be used if the moving block bootstrap is applied.

3.3.2 Remainder Sieve Bootstrap

Due to the complexity of the real-world time series data, it is usually difficult to guarantee the property of stationarity. In Cordeiro and Neves [13], a method to alleviate difficulty is introduced. They suggested fitting exponential smoothing models (EXPO) on the time series data, then sieve bootstrapping the remainder term, if stationary, as Bühlmann does on the residuals obtained from AR fitting. In this report, to maintain consistency with the method mentioned in section 3.2.2 for better comparison, the remainder is obtained from the Loess regression or STL decomposition instead of EXPO fitting, according to the suggestion in the paper by de Oliveira and Cyrino Oliveira [15]. Moreover, following their proposal, the autoregressive moving-average model (ARMA(p, q), introduced in Section 2.3.2) is adopted to fit the remainder component, aimed at capturing any remaining information on the remainder and ensuring the white noise characteristic of the residual.

Thus, the detailed steps for sieve bootstrapping on the remainder are as follows,

Step 1: Use Loess regression to separate non-seasonal data to trend and remainder components. Use STL decomposition to separate the seasonal data into seasonal, trend, and remainder components.

Step 2: Adjust the ARMA(p, q) model using AIC with correction (AIC_c) or other methods to determine the order p and q, then fit the remainder.

Step 3: Center the residuals from the previous step.

Step 4: Bootstrap samples from the centred residuals.

Step 5: Combining the ARMA model from step 2 and bootstrapped residuals from step 4 to construct a new series.

Step 6: Assemble seasonal and trend components with the series from step 5 to obtain the bootstrapped series.

Repeat the loop from step 4 to step 6 for required times to obtain enough number of bootstrapped samples.

3.4 Bootstrap Aggregating (Bagging)

After introducing Efron's bootstrap method and the bootstrap methods for dependent data, we will finally introduce the method we used to improve the forecast accuracy of the ARIMA model. Proposed by Breiman [8], the bootstrap aggregating (bagging) was originally designed to augment the stability and precision of the machine learning algorithms. He observed that certain algorithms, such as decision trees, exhibit a notable susceptibility to variations in outcomes with minor modifications in the training dataset, and this phenomenon is referred to as instability. The bagging methodology seeks to mitigate this issue by generating multiple iterations of a model, or in the machine learning domain, a predictor. The generation process involves sampling the original dataset with replacement (bootstrap sampling), fitting a model for each resampled data, and estimating the parameter or model function of interest. Typically, these models are of the same type but are fit independently to different resampled data. Then, the results from each substitution are combined to form the final aggregated results.

For example, we could employ the bagging method to improve the stability of a regression model in predicting. First, bootstrapping the dataset to obtain a number of samples that are the same size as the original dataset. Then, tune the parameters of the regression model for each bootstrapped sample. Next, predict a data point that is known but not in the dataset using the fitted models. Finally, average the predictions to form the final prediction and evaluate the aggregated prediction with the known data point. Similarly, this methodology can also be applied in the context of time series forecasting. Bergmeir et al. [4] suggested applying the bagging method to aggregate the forecasts of the bootstrapped series and consequently reduce the forecast errors.

The rationale underneath bagging is apparent, assume $\mathcal{L}$ is a learning set consisting of data (y_n, x_n), $n = 1, \ldots, N$, for which x is the input value and y is the output value, and predictor ϕ. The predictor ϕ can be any functions, algorithms, or models of our interest (e.g. the ARIMA model). Suppose the data (y_n, x_n) in $\mathcal{L}$ is sampled independently from distribution $\mathbb{P}$. Then, the result of the aggregate predictor, ϕ_A, is the average of the results from each predictor over the entire learning set,

$$\phi_A(x, \mathcal{L}) = \mathbb{E}_{\mathcal{L}}[\phi(x, \mathcal{L})] \tag{3.11}$$

If we expand the formula for the mean-squared error (MSE) of the outputs from the predictors, according to the linearity of expectation,

$$\mathbb{E}_{\mathcal{L}}[y - \phi(x, \mathcal{L})]^2 = y^2 - 2y\mathbb{E}_{\mathcal{L}}[\phi(x, \mathcal{L})] + \mathbb{E}_{\mathcal{L}}[\phi^2(x, \mathcal{L})] \tag{3.12}$$

Substitute 3.11 into 3.12, and the use the inequality $\mathbb{E}[X^2] \geq \mathbb{E}[X]^2$, we have,

$$\mathbb{E}_{\mathcal{L}}[y - \phi(x, \mathcal{L})]^2 \geq (y - \phi_A(x, \mathcal{L}))^2 \tag{3.13}$$

Then, by integrating both sides, we can observe that the MSE of the result of the aggregate predictor is less or equal to the MSE of $\phi(x, \mathcal{L})$ over $\mathcal{L}$. Moreover, the improvement of the accuracy by aggregation depends on the disparity between $\mathbb{E}_{\mathcal{L}}[\phi(x, \mathcal{L})]^2$ and $\mathbb{E}_{\mathcal{L}}[\phi^2(x, \mathcal{L})]$. If $\phi(x, \mathcal{L})$ remains relatively stable with different instances of $\mathcal{L}$, then the two sides of the equation will be almost identical, rendering aggregation ineffective. However, the greater the variability in $\phi(x, \mathcal{L})$, the more beneficial aggregation becomes. It should be noted that ϕ_A consistently outperforms ϕ.

If ϕ_A depends not only on x but also the probability distribution $\mathbb{P}$ where $\mathcal{L}$ is drawn, the aggregate or bagged estimator is not $\phi_A(x, \mathbb{P})$, but

$$\phi_B = \phi_A(x, \mathbb{P}_{\mathcal{L}}) \tag{3.14}$$

where $\mathbb{P}_{\mathcal{L}}$ is the empirical distribution constructed by weighing each point $(y_n, x_n) \in \mathcal{L}$ with weights $\frac{1}{N}$. This distribution $\mathbb{P}_{\mathcal{L}}$ is also known as the bootstrap approximation to $\mathbb{P}$. In our

report specifically, the bootstrap methods used here are the moving block bootstrap and sieve bootstrap. Similar to the discussion for 3.13. When the procedure is unstable, bagging would provide considerable improvement in accuracy. When the procedure is stable, ϕ_B would not yield a better result, since the aggregation of predictors on data drawn from $\mathbb{P}$ is congruent to the actual predictor, $\phi_A(x, \mathbb{P}) \simeq \phi(x, \mathcal{L})$.

In the specific case of forecasting time series data, Petropoulos et al. [39] stated that the effectiveness of bagging can be attributed to its ability to address various forms of uncertainty inherent in forecasting models. These uncertainties include model uncertainty, data uncertainty, and parameter uncertainty. The aggregating technique mitigates these uncertainties by averaging out the noise and errors across different models and datasets. This process results in a more robust and accurate forecast, as it combines the strengths of multiple models and reduces the risk of overfitting to a particular dataset. In the context of time series, where data often exhibit non-stationarity and autocorrelation, bagging becomes particularly valuable. It allows for the creation of bootstrapped samples that maintain the essential characteristics of the original data, thus ensuring that the forecasts are reliable and representative of the underlying patterns in the data.

3.5 Overall Process

Having introduced all the methods and techniques used in this report, we can draw a general picture of the whole process of applying bagging with bootstrap in the ARIMA model. As discussed in previous chapters, our method integrated the proposed by Bergmeir et al. [4] and de Oliveira and Cyrino Oliveira [15]. Consequently, our proposed method has five steps, which are data transformation, data decomposition, bootstrapping, forecasting, and bagging. The flow chart for our method is presented in Figure 3.2.

1. **Data transformation**: the Box-Cox transformation is implemented. Given a time series, $X = (x_1, x_2, \ldots, x_n)$, the R function BoxCox$(\cdot)$ in the {forecast} package will be used to stabilise its variance. The parameter λ is chosen automatically by another R function BoxCox.lambda$(\cdot)$ using Guerrero's method (introduced in Section 2.2.1).

2. **Data decomposition**: the data will be classified into non-seasonal or seasonal through visual inspection of its plot and autocorrelation function. For seasonal data, the STL decomposition is implemented to separate the series into seasonal, trend, and remainder components by R function stl$(\cdot)$ in the {stats} package. In terms of the non-seasonal series, a Loess regression is fitted to the series through R function loess$(\cdot)$, with the model fit as the trend component and residuals as the remainder component.

3. **Bootstrapping**: after obtaining the remainder component from either the STL decomposition or Loess fitting, the moving block bootstrap and sieve bootstrap are performed on the remainders. As for the moving block bootstrap, we define a function based on the method discussed in Section 3.2.2, which produces B bootstrapped series, $R_1^*, \ldots, R_B^*$, by resampling blocks with a certain length with replacement. Next, the resampled series are combined with the seasonal and trend parts from the data decomposition step to create B bootstrapped series, $X_1^*, \ldots, X_B^*$. For the sieve bootstrap, we define a function based on the method mentioned in Section 3.3.2, in which an ARMA model is fitted to the remainder and the centred residuals are sampled. The parameters for the ARMA model are determined automatically by an R function auto.arima$(\cdot)$ in {forecast} package. Then, the sampled residuals are combined with the ARMA model fit and trend component from the previous stage to form B bootstrapped series. To finish this step, the bootstrapped samples are converted to the same scale as the raw series through the

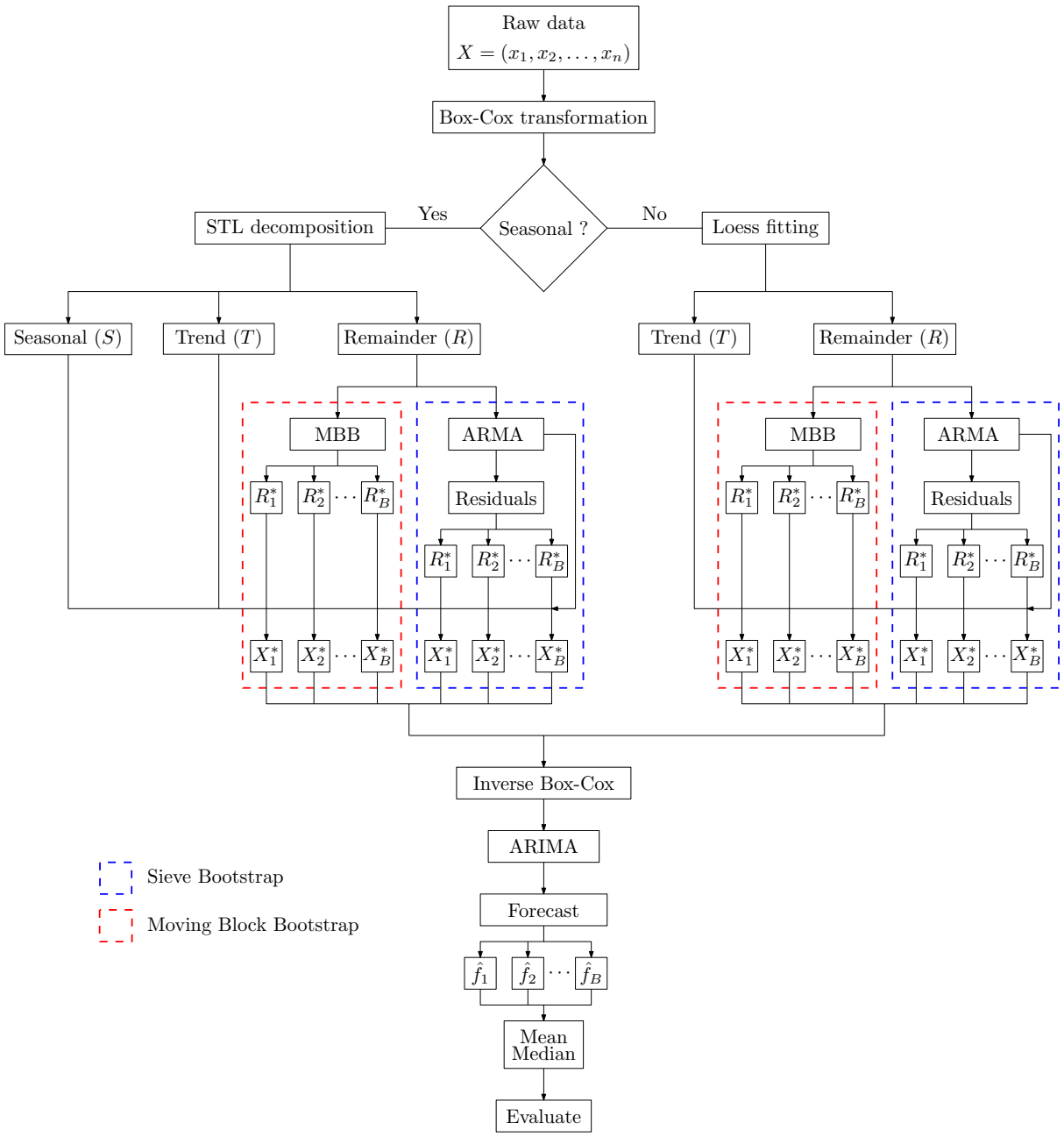

Figure 3.2: Overall Process

inverse Box-Cox transformation with the same λ as the Box-Cox transformation by R function InvBoxCox($\cdot$).

4. **Forecasting**: the bootstrapped series will be forecast using auto.arima($\cdot$), where the parameters are chosen automatically to give the best ARIMA model according to the either AIC, BIC, or AICc value. This will give B forecasts, $\hat{f}_1, \ldots, \hat{f}_B$.

5. **Bagging**: the forecasts $\hat{f}_1, \ldots, \hat{f}_B$ are aggregated by taking the mean and median. Finally, the aggregated results will be evaluated.

3.6 Conclusion

In this chapter, we mainly introduced the resampling technique, bootstrap, with its extensions to time series data, the moving block bootstrap and sieve bootstrap, and the fundamental technique for improving accuracy, bagging. Efron's bootstrap features a random sampling of independent data with replacement, while Künsch's moving block bootstrap samples equal size consecutive data points to relieve the constraint of data independence. As for the Bühlmann's sieve bootstrap, it features fitting parametric model first and then sampling the residuals. Moreover, bagging, a method designed to enhance machine learning algorithms, is used in this report to eliminate uncertainties and hence improve the forecast accuracy. After detailing the techniques used to improve the ARIMA, we concluded this chapter by establishing our method, which consisted of 5 steps, data transformation, data decomposition, bootstrapping, forecasting, and bagging. In the next chapter, we will apply our method to datasets to evaluate its effectiveness in improving the ARIMA forecast accuracy.

Chapter 4

Application

In the previous chapters, we first introduced the resampling techniques, moving block bootstrap and sieve bootstrap that generate resamples, followed by the bagging methodology that is used to improve the forecast accuracy of ARIMA. Then, in order to draw a general picture for our proposed method, we assemble all the methods introduced in Chapter 2 and Chapter 3 to give the full model design at the end of the last chapter.

With the method design introduced, we will then evaluate the effectiveness of our proposed method in improving forecast accuracy for ARIMA on some real-world datasets. In this report, we applied our method to three datasets, which are the Energy Trend datasets containing monthly and quarterly data, the Wolf Sunspot Number dataset and the Canadian Lynx dataset containing yearly data [19, 34, 37]. In this chapter, first, the datasets and evaluation methods used in the analysis will be introduced. Then the detailed steps for our methods on the datasets will be demonstrated, followed by the evaluation of final forecasts.

4.1 Datasets

In order to thoroughly investigate the effectiveness of our method in improving the forecast accuracy of the ARIMA model, we selected three real-world datasets with different frequencies. Provided by the Department of Energy Security and Net Zero in the UK, the Energy Trend dataset contains a series of monthly and quarterly data on the supply and demand of energy in the UK [19]. This data collection provides information on a variety of energy indicators, such as oil consumption, gas consumption, and renewable electricity production, over a long range of time. Due to the magnitude of the size and complexity of the Energy Trend dataset, we select 2 monthly and 2 quarterly consumption data to implement our method. As for the yearly data, we chose two datasets, namely the Wolf Sunspot number produced by the Royal Observatory of Belgium and the Canadian Lynx dataset enclosed in the Encyclopedia of Mathematics [37, 34]. The Wolf Sunspot dataset contains the mean of daily sunspot numbers in each year from 1700 to the present and the Canadian Lynx dataset recorded the annual number of lynx trapped in a river in Canada from 1821 to 1934.

The details of the data used in this report are listed in Table 4.1,

4.2 Evaluation Methodology

To evaluate our method in improving the ARIMA model, we need to analyse the genuine forecast errors rather than the residuals of the fitted model on the whole dataset [29]. Therefore, we need to divide the dataset into in-sample data or training set, and out-of-sample data or test set. Additionally, the accuracy metrics for forecast error evaluation also need to be determined. Consequently, in this section, we will introduce how we divide the dataset and what accuracy metrics will be used in this report.

Type	Data Name	Unit
Monthly	Total Inland Energy Consumption	Million Tonnes of Oil Equivalent
Monthly	Gas Output from Transmission	Million Cubic Metres
Quarterly	Final Energy Consumption	Million Tonnes of Oil Equivalent
Quarterly	Total Crude Oil Demand	Thousand Tonnes
Yearly	Wolf Sunspot number	number
Yearly	Canadian Lynx number	number

Table 4.1: Data used in this report

4.2.1 Training and Test Sets

Given the datasets, it is crucial to divide them into training and test sets for determining or evaluating the models. Typically, the training set and test set are in a ratio of 4:1, yet the length of the test set depends on the horizon of the forecast period [29]. For our report, we adopted the choice of the test set length from a forecasting competition, M Competition [36]. In this competition, the required forecasts for different types of data are 18 for monthly data, 8 for quarterly data, and 6 for yearly data [36]. The determination of this choice was based on the type of decisions that are best supported by each data frequency within businesses or governments. For example, the yearly data is commonly used to support long-term strategic decisions for 1 to 5 years ahead. As for quarterly and monthly data, they are usually adopted as sources for budget-making for a few months to 2 years in the future [36].

Therefore, in this report, we extract 30, 40, and 90 data points from the tail of the yearly, quarterly, and monthly data, respectively, and divide the training set and the test set in a ratio of four to one. Then, the size of the yearly, quarterly, and monthly training sets will be 24, 32, and 72, respectively, and the size of the test set will be 6, 8, and 18, respectively.

4.2.2 Accuracy Metrics

To evaluate the effectiveness of our model in improving the ARIMA model, we need to assess the point forecasts using some benchmarks, or in other words, accuracy metrics. These metrics measure the errors of forecasts, e_t, in different ways and the errors of forecasts are defined as,

$$e_{T+h} = z_{T+h} - \hat{z}_{T+h|T}$$

where h indicates the forecast periods ahead and T is the index of the last observation in the training set. The training set is $\{z_1, \ldots, z_T\}$, test set is $\{z_{T+1}, z_{T+2}, \ldots\}$, and forecasts are $\{\hat{z}_{T+1}, \hat{z}_{T+2}, \ldots\}$. Hyndman and Athanasopoulos [29] summarised that the accuracy metrics can be divided into three categories, which are scale-dependent errors, percentage errors, and scaled errors.

1. **Scale-dependent errors**: these accuracy metrics depend solely on the errors and are unable to compare across multiple time series data with different scales since the errors have the same unit or scale as the data. Even though scale-dependent errors are not suitable for use in compassion, they are widely used in measuring accuracy. Two commonly used scale-dependent errors mean absolute error (MAE) and root mean square error (RMSE)

are based on the absolute errors and squared errors respectively,

$$\begin{aligned} \text{MAE} &= \frac{1}{n}\sum_{i=1}^{n}|e_i| \\ \text{RMSE} &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}e_i^2} \end{aligned}$$

2. **Percentage errors**: rather than yielding a number with the same scale as the data, these metrics provide an aggregation of the percentage of forecast deviates from the actual value at each time. Two commonly used percentage errors are mean absolute percentage error (MAPE) and symmetric mean absolute percentage error (sMAPE) [3],

$$\begin{aligned} \text{MAPE} &= \frac{100\%}{n}\sum_{i=1}^{n}|\frac{e_i}{z_i}| \\ \text{sMAPE} &= \frac{100}{n}\sum_{i=1}^{n}\frac{|e_i|}{(z_i+\hat{z}_i)/2} \end{aligned}$$

   The sMAPE is defined to address the over-penalty problem of MAPE which penalises more on positive forecasts than negative forecasts. Moreover, when the test value z_t equals zero or close to zero, the MAPE may be undefined or has an extreme value. Due to the fact that the denominator of sMAPE can be negative, the value of sMAPE can be negative and it will fail to provide a measure of errors in percentage.

3. **Scaled errors**: As an alternative to percentage error, the scaled error metrics measure the scaled errors, q_j, based on the in-sample MAE of a naive one-step forecast method. The naive one-step forecast is a simple forecast method that sets the forecast as the value of the previous step. For non-seasonal data, the scaled error is,

$$q_t = \frac{e_t}{\frac{1}{n-1}\sum_{i=2}^{n}|z_i - z_{i-1}|}$$

   On the other hand, for the seasonal data, the scaled error is,

$$q_t = \frac{e_t}{\frac{1}{n-m}\sum_{i=2}^{n}|z_i - z_{i-m}|}$$

   where m is the season length. Consequently, the scaled error proposed by Hyndman and Koehler [31] is the mean absolute scaled error (MASE),

$$\text{MASE} = \text{mean}(|q_t|)$$

   Since the numerator and denominator both depend on the data scale, the scale of the error is independent of the data scale [29].

In this report, we selected one accuracy metric from each category in order to leverage the strengths of each metric and provide a comprehensive analysis. For scale-dependent error, we chose the RMSE, which measures the quadratic mean of the difference between test values and predicted values and provides information on the fit of the model. For percentage error, since all the data in our dataset is non-negative, we chose the MAPE as a measurement of the percentage error of our proposed method. For the scaled error, we used the MASE to compare the forecast accuracy of different series.

4.2.3 Nonparametric Tests

Once we obtain the accuracy metrics for each method on multiple datasets, we can then rank the metrics from low to high for the methods on each data and search for the method that ranks first. However, sometimes the rankings given by different metrics might not be consistent within one dataset. This may be problematic, since inconsistent rankings may influence our judgement on the optimal method. To address this issue, we introduce Kendall's coefficient of concordance or Kendall's W to examine whether the metrics agree with each other's ranking. Suppose that method i is given a rank, $r_{i,j}$ by a metric j, where in total k metrics and n methods. Then, the total rank for method i is, $R_i = \sum_{j=1}^{k} r_{i,j}$, and the average total rank for the methods is, $\frac{k(n+1)}{2}$. Define the sum of squares of the deviations of the total rank from the average as

$$S = \sum_{i=1}^{n} \left(R_i - \frac{k(n+1)}{2} \right) \tag{4.1}$$

Hence, Kendall's W is defined as [20],

$$W = \frac{12S}{k^2 n(n^2 - 1)} \tag{4.2}$$

Specifically, Kendall's W is defined as a ratio of S to S in complete agreement and ranges from 0 (no agreement) to 1 (complete agreement). To test the significance of the hypothesis that all metrics agree with each other (i.e. W close to 0), we can test the hypothesis by the approximate distribution of W, which is a chi-square distribution with $n - 1$ degrees of freedom,

$$k(n-1)W \sim \chi^2_{n-1} \tag{4.3}$$

If the null hypothesis for one dataset is rejected, then accuracy metrics will likely give a consistent ranking. Thus, we can analyse the performance of the methods based on the ranking given by the metrics.

With Kendall's W statistic, we could analyse which method has better forecast accuracy by evaluating the hypotheses and comparing the three metrics with the benchmark ARIMA model on a single data. However, we cannot provide any conclusion on which one is optimal among the full dataset, since the best model for each frequency of data may differ. Consequently, we need a test to detect the differences between methods and then determine the optimal method. As we mentioned in the previous paragraph, Kendall's W is a statistic providing information on rank correlation, yet it is a normalisation of the statistic of Friedman's test, which detects the differences between methods among different datasets [20]. Assume there are k methods and b datasets, and they are independent. The test statistic for the Friedman test without ties is [12],

$$T_1 = \frac{12}{bk(k+1)} \sum_{j=1}^{k} \left(R_j - \frac{b(k+1)}{2} \right)^2 \tag{4.4}$$

where $R_j = \frac{1}{b} \sum_{i=1}^{b} r_{ij}$ are the mean rank for each method across different datasets. In case of ties, the test statistics can be adjusted to [12],

$$T_1 = \frac{k-1}{A_1 - C_1} \sum_{j=1}^{k} \left(R_j - \frac{b(k+1)}{2} \right)^2 \tag{4.5}$$

where $A_1 = \sum_{i=1}^{b} \sum_{j=1}^{k} r_{ij}^2$ and $C_1 = \frac{bk(k+1)^2}{4}$. In order to more accurate approximation, researchers modified the test statistic T_1 to [12],

$$T_2 = \frac{(b-1)T_1}{b(k-1) - T_1} \tag{4.6}$$

Moreover, the null and alternative hypotheses for the test are,

H_0 : equal probability for each ranking for each method within one dataset,

H_1 : at least one method tends to provide fewer errors

Hence, H_0 is rejected if T_2 is over the $(1-\alpha)$ quantile of the F distribution, $F_{(k-1,(b-1)(k-1))}$.

If the null hypothesis for Friedman's test is rejected, we can then compare the methods to identify the differences. According to Conover [12], two methods are considered different if the inequality is satisfied,

$$|R_n - R_m| > t_{1-\frac{\alpha'}{2}} \left[\left(\frac{b^2 k(k+1)}{6(b-1)} \right) \left(1 - \frac{T_1}{b(k-1)} \right) \right]^{\frac{1}{2}} \tag{4.7}$$

where $n, m \in \{1, \ldots, k\}$, T_1 is from (4.4), and $t_{1-\frac{\alpha}{2}}$ is the t distribution with $(k-1)(b-1)$ degree of freedom and significance level α'. Noticeably, the significance level α' for the multiple comparisons is different from the significance level for Friedman's test and is calculated by controlling the family-wise error rate (FWER). The FWER measures the probability of making false positives (Type 1 error) among a family of tests, which is defined as [41],

$$\text{FWER} = 1 - (1 - \alpha')^m$$

where α' is the significance level for each test and m is the number of tests. Hence, the probability of producing one or more Type 1 errors within a family is controlled by ensuring FWER $\leq \alpha$. There are various procedures to control the FWER, such as Bonferroni's procedure [41], Holm's procedure [27], Hochberg's procedure [26], etc. In our report, for simplicity, we adopt Bonferroni's procedure for adjusting the significance level for the multiple comparison tests. The step for Bonferroni's procedure is simple, we reject the hypothesis for each comparison if its p-value is less than $\frac{\alpha}{m}$, where α is the significance level for Friedman's test and m is the number of comparisons.

4.3 Method Application

In this section, we will discuss the detailed application of our method according to the steps discussed in Section 3.5. For better demonstration, as an example, we will apply our method to one of the datasets, gas output from the transmission.

4.3.1 Data Preparation

Based on the steps introduced in Section 3.5, we first divide the original dataset into a training set and test set according to its frequency. Then, the Box-Cox transformation is applied to the training set to stabilise the variance and make the data stationary. This process is completed through an R function, BoxCox$(\cdot)$ in {forecast} package. Moreover, the parameter λ in the transformation is chosen according to Guerrero's method mentioned in Section 2.2.1 with restriction $\lambda \in [0, 1]$ [4]. For instance, the monthly gas output from transmission data before and after the Box-Cox transformation is presented in Figure 4.1. In the figure, we can observe that, compared with the original data on the top, the spikes in the transformed data on the bottom are more average and the range of the fluctuations is smaller, which helped to make the data stationary.

After the Box-Cox transformation, the seasonality of the data is checked through it ACF plot. If it is non-seasonal it would be decomposed by fitting a Loess model (loess$(\cdot)$ in {stats} package) with span 0.3. Due to the fact that an excessively high span results in an overly smoothed fitting curve, leading to the loss of critical information, and a too-small span causes

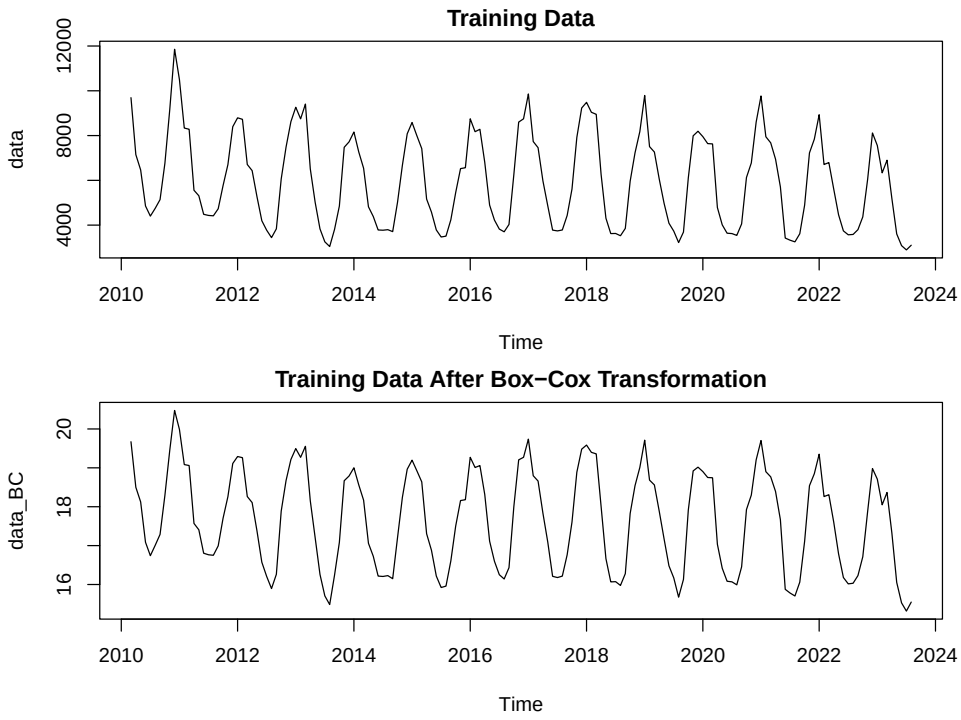

Figure 4.1: Box-Cox Transformation for Monthly Gas Output

an overfitting issue, deviating from the initial intention of separating data trends, we selected a span of 0.3 based on the visual inspection on different choices of span. If it is seasonal, it would be decomposed by the STL decomposition through stl(·) function in R. For demonstration, the ACF plot for the monthly gas output from transformation is given in Figure 4.2.

In Figure 4.2, we observe that there are significant pits within one-year intervals, which indicates its strong seasonality. Consequently, we apply the STL decomposition to the transformed data. The result is in Figure 4.3, where the second graph from the top represents the seasonal component, the third graph from the top shows the trend component and the bottom graph demonstrates the remainder component.

4.3.2 Moving Block Bootstrap

Before applying the moving block bootstrap to the remainder component, we need to decide the number of bootstrap iterations, B, and the block length, l, for the moving block bootstrap. In terms of the bootstrap iteration, Theorem 3.1 suggests that the more iterations generated, the closer the ecdf to the real cdf of the sample. However, due to the lack of computation resources and time, it is infeasible to generate an infinite number of bootstrapped samples. Furthermore, the purpose of using the bootstrap techniques is to eliminate the uncertainties of forecasting with a single series rather than obtaining the statistical properties of the data. Hence, considering previous similar research, we set the bootstrap iteration, $B = 100$ [15].

As for the block length, we considered the block length used in the paper by Bergmeir et al. [4], which is $l = 24$ for monthly data and $l = 8$ for yearly and quarterly data. The reason for setting two years as the block length for monthly data is to ensure any remaining seasonality is captured in the block. For demonstration, a graph containing 5 resamples of the monthly gas output data sampled by moving block bootstrap (in colours) and the original data (in black) is given in Figure 4.4 (MBB Resampled Series). From the top graph, we can observe that resamples generated by the moving block bootstrap resemble the behaviour of the input data.

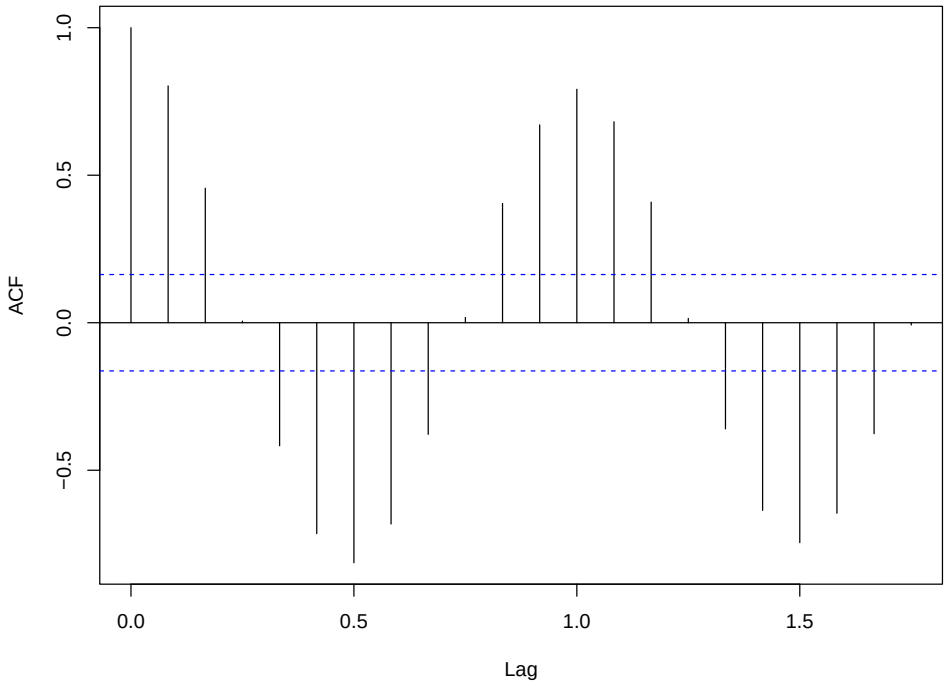

Figure 4.2: ACF for Monthly Gas Output

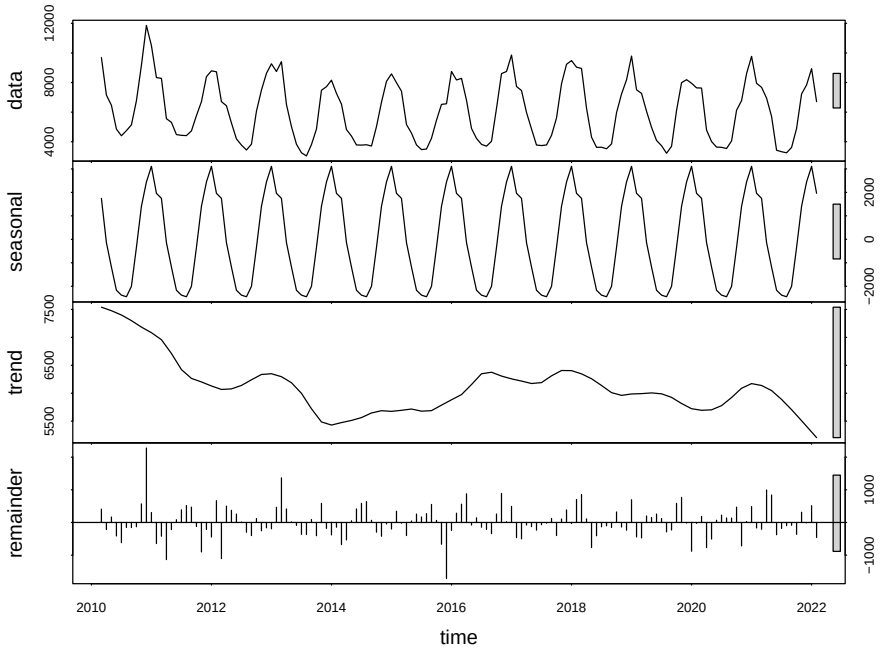

Figure 4.3: STL Decomposition of Monthly Gas Output

4.3.3 Sieve Bootstrap

Except for the moving block bootstrap, the other resampling method we used is the sieve bootstrap. Similarly, the number of resamples for the sieve bootstrap to generate is set as $B = 100$. The other parameters that need to be determined before the resampling are the parameters for the time series model fitted to the remainder component. In this report, we considered using the ARMA model to obtain residuals for resampling [15]. The parameters for the ARMA are chosen automatically by the auto.arima(·) function using the methods discussed in Section 2.4.1 and Section 2.4.2. By setting the maximum differencing degrees to be 0, the function is restricted to use ARMA model. For instance, a graph containing the input data (in black) and 5 resamples of the monthly gas output data sampled by sieve bootstrap (in colours) is given in Figure 4.4 (RSB Resampled Series). From the bottom graph, we can observe that the sieve bootstrap resamples successfully mimicked the original series.

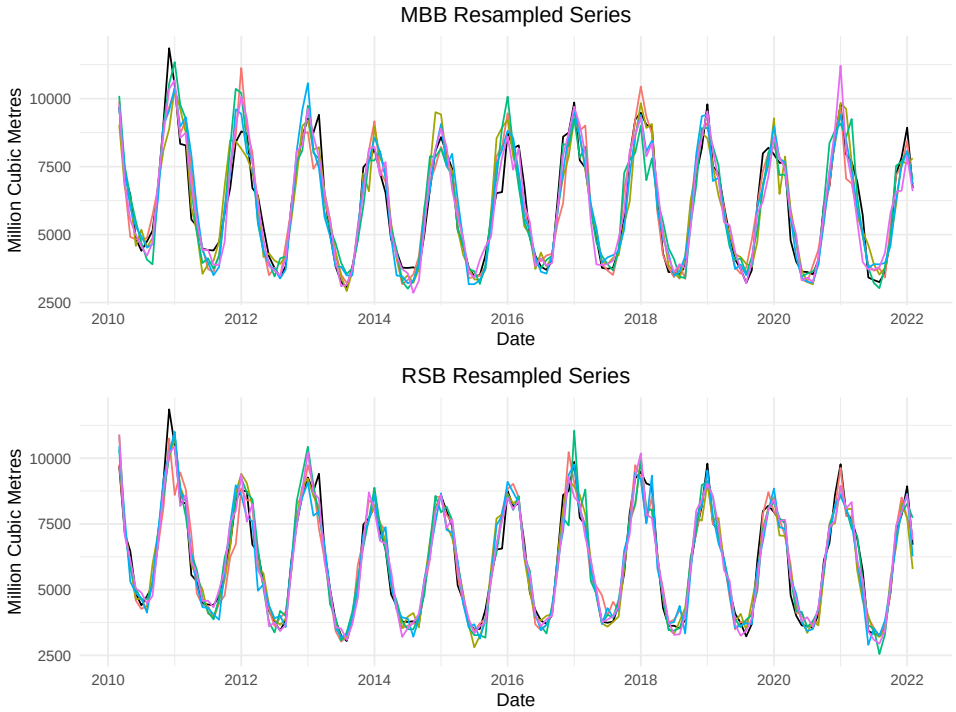

Figure 4.4: Bootstrapped resamples

4.3.4 Forecasting

The next step after bootstrapping is forecasting, which is performed with two powerful R functions, auto.arima(·) and forecast(·). Similar to the usage of the first function when performing the sieve bootstrap, it is used to fit the optimal ARIMA model for the bootstrapped series using the model identification and estimation methods introduced in Section 2.4.1 and 2.4.2. As for the latter function, it forecasts the data based on the fitted model given by the auto.arima function following the procedures discussed in Section 2.4.3. The forecast period argument in the function is in line with the length of the test set for evaluation purposes.

4.3.5 Bagging

In the final step, bagging, the mean and median of the point forecasts for each bootstrapped series from both bootstrap methods are calculated. Then, the aggregated results are computed based on the three accuracy metrics, RMSE, MAPE, and MASE. The accuracy metrics for our

proposed method implemented on the monthly gas output data are given in Table 4.2, where MBB represents the moving block bootstrap for the remainder, RSB means the sieve bootstrap for the remainder, and the words in the bracket stands for the method used in bagging. Based on the accuracy measurements yielded, we can then evaluate whether our using bagging with bootstrap improves the forecast accuracy for ARIMA or not.

	ARIMA	MBB(mean)	MBB(median)	RSB(mean)	RSB(median)
RMSE	674.037	599.952	580.843	551.560	550.826
MAPE	11.498	9.924	9.426	9.262	9.322
MASE	0.858	0.735	0.704	0.688	0.695

Table 4.2: Accuracy Metrics for Monthly Gas Output Data

4.4 Result Analysis

Following the same methodology implemented in Section 4.3, we applied our method to the rest of the datasets in Table 4.1. First, we will analyse the accuracy metrics for each method in each of the data and use Kendall's W to test the agreement between metrics in each of the data. Then, we will use Friedman's test to analyse the difference between methods.

4.4.1 Agreement Analysis

In the first two blocks in Table 4.3, the accuracy metrics for our proposed method and ARIMA model tested on monthly data are recorded. Since the p-values for Kendall's W are less than 0.05, the three metrics provide consistent ranking information for the methods in monthly data. However, even though the W statistic is significant and shows three metrics have agreement on the rankings under 0.05 significance level, the rankings are not necessarily identical. For a specific combination of the bootstrap and bagging, using sieve bootstrap and median bagging has the smallest forecast errors in terms of RMSE, yet using sieve bootstrap with mean bagging has the lowest percentage error (MAPE) and scaled error (MASE) in monthly total inland energy consumption dataset. As for the other monthly dataset, using moving block bootstrap with mean bagging has the lowest error in all three metrics. Nevertheless, for both monthly data, using moving block bootstrap and sieve bootstrap gives smaller errors when forecasting. Hence, we can conclude that our proposed method exhibits better forecast accuracy than the benchmark ARIMA model with both monthly data.

The results for quarterly data are presented in the third and fourth blocks in Table 4.3. Both Kendall's W statistics are significant and suggest agreement in the rankings. Noticeably, even though Kendall's W for quarterly total crude oil data is 1, its p-value is not approximately 0. This is due to the approximate distribution we used for W is chi-square distribution (discussed in Section 4.2.3). For the quarterly total crude oil data, our proposed method outperforms the standard ARIMA model and the moving block bootstrap with mean averaging dominates other methods. However, for the quarterly final energy consumption data, even though using the sieve bootstrap with median averaging could beat the ARIMA model in comparison between the three metrics, using the moving block bootstrap failed to obtain better performance than the ARIMA model. This may be due to the significant drop at the tail of the training set. The sudden decrease at the end influenced the Loess fitting to the training set and increased the remainders at the tail, which may create a non-stationary remainder part and weaken the power of the moving block bootstrap. For instance, as shown in Figure 4.5, the bootstrapped samples generated by the moving block bootstrap show poor imitation of the behaviour of the original series compared with the samples generated by the sieve bootstrap. Therefore, since using

moving block bootstrap fails to exceed the performance of the ARIMA, we cannot conclude that moving block bootstrap can improve the ARIMA, still we can conclude that using sieve bootstrap can improve the forecast accuracy with quarterly data.

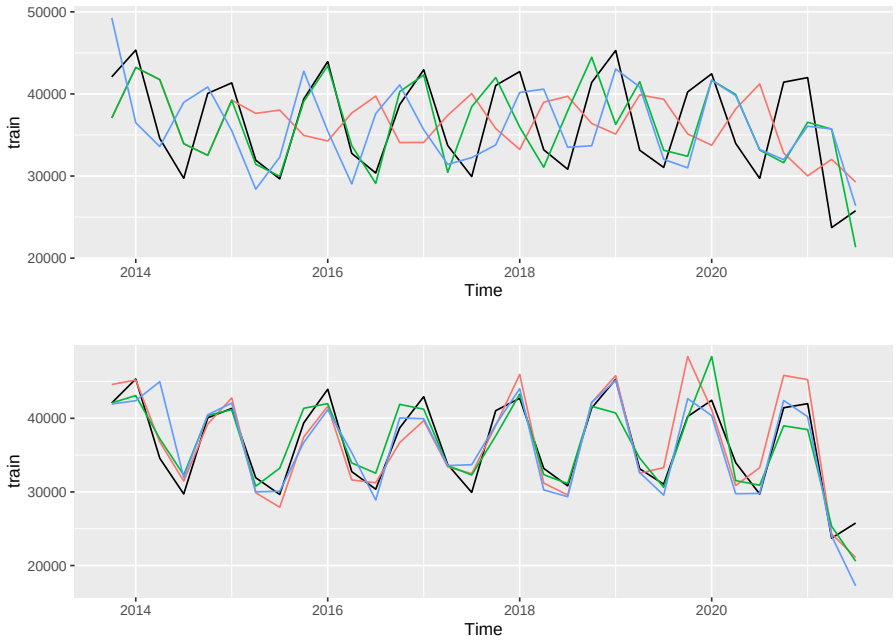

Figure 4.5: MBB (top) and RSB (bottom) Resamples for Quarterly Final Energy Consumption

In the last two blocks in Table 4.3, the yearly data results are presented. Noticeably, despite Kendall's W being significant in yearly sunspot number data, it fails to reject the null hypothesis in the yearly Canadian Lynx number data. This means that the three metrics give relatively distinct information in that dataset. Nonetheless, the results from our proposed methods with bootstrapping show noticeable improvement in terms of accuracy with yearly data (all three metrics for our method are less than that of the ARIMA model). Moreover, for both data, using the moving block bootstrap has better forecast accuracy compared with using the sieve bootstrap. However, which averaging method for moving block bootstrap has a more significant improvement in forecast accuracy for yearly data is hard to conclude, due to the metrics' disagreement in the rankings.

Forecast Methods	**Bagging**	**RMSE**	**MAPE**	**MASE**	**Kendall's W**	**p-value**
Monthly Total Inland Energy Consumption					0.956	0.022
ARIMA	-	674.037	11.498	0.858		
MBB.ARIMA	Mean	599.952	9.924	0.735		
	Median	580.843	9.426	0.704		
RSB.ARIMA	Mean	551.560	9.262	0.688		
	Median	550.826	9.322	0.695		
Monthly Gas Output from Transmission					0.899	0.031
ARIMA	-	0.810	5.316	0.815		
MBB.ARIMA	Mean	0.679	4.660	0.708		
	Median	0.681	4.685	0.710		
RSB.ARIMA	Mean	0.688	4.662	0.709		

Continued on next page

Forecast Methods	Bagging	RMSE	MAPE	MASE	Kendall's W	p-value
	Median	0.707	4.865	0.737		
Quarterly Final Energy Consumption					0.956	0.022
ARIMA	-	2096.734	5.676	1.266		
MBB.ARIMA	Mean	4608.193	13.183	2.672		
	Median	4854.479	13.704	2.766		
RSB.ARIMA	Mean	2169.33	5.461	1.246		
	Median	1779.784	5.076	1.135		
Quarterly Total Crude Oil Demand					1.000	0.017
ARIMA	-	1510.086	10.443	1.300		
MBB.ARIMA	Mean	736.042	4.542	0.571		
	Median	807.654	5.180	0.649		
RSB.ARIMA	Mean	1313.718	8.194	1.086		
	Median	1343.807	8.938	1.116		
Yearly Sunspot Number					0.956	0.022
ARIMA	-	33.760	305.805	1.205		
MBB.ARIMA	Mean	27.634	227.061	0.970		
	Median	27.702	225.309	0.972		
RSB.ARIMA	Mean	29.351	258.374	1.051		
	Median	29.303	255.097	1.046		
Yearly Canadian Lynx Number					0.622	0.113
ARIMA	-	612.923	36.221	0.543		
MBB.ARIMA	Mean	342.948	34.334	0.339		
	Median	353.022	36.051	0.333		
RSB.ARIMA	Mean	351.114	29.458	0.342		
	Median	362.889	30.046	0.352		

Table 4.3: Results for Our Datasets

In conclusion, with Kendall's W being tested, we gather that the accuracy metrics manage to provide consistent ranking information on the methods in most cases, but for the exception case, the metrics still provide useful information to support our argument. For the six datasets used for evaluation, our proposed methods managed to improve the forecast accuracy of the ARIMA model in five of the six datasets. For the one dataset that our method failed, due to the irregular behaviour of the training set, using the moving block bootstrap generates poor resamples and diminishes the power of bootstrap and bagging, yet using the sieve bootstrap could outperform the standard ARIMA model. Thus, based on the collected datasets, we can conclude that our method indeed improves the forecast accuracy for the ARIMA model for data with different frequencies in most cases and using sieve bootstrap can constantly provide forecast improvement.

4.4.2 Difference Analysis

In the previous section, we analysed our method by comparing the results of our method and the ARIMA model in datasets with different frequencies. Although our method showed potential to improve the ARIMA model in most cases, the global comparison with the full dataset has not yet been conducted. Therefore, in this section, we will analyse the difference between the methods using the Friedman rank sum test and the corresponding multiple comparison analysis on the entire dataset containing monthly, quarterly and yearly data.

To perform the Friedman test, we need to select a representation accuracy metric for each method. Since the units for different data vary, we thereby choose the scaled error (MASE) for cross-comparison. Then, for each data, the ranks for the MASE of each method are given and the sums of ranks for each method are calculated. The MASE ranks are given in Table 4.4, and in the data column, capital letters represent the data frequency and the numbers are the data number. For instance, the second monthly data, monthly gas output data is M_2 in the table.

Data	ARIMA	MBB(mean)	MBB(median)	RSB(mean)	RSB(median)	R_i
M_1	5	4	3	1	2	15
M_2	5	1	3	2	4	15
Q_1	3	4	5	2	1	15
Q_2	5	1	2	3	4	15
Y_1	5	1	2	4	3	15
Y_2	5	2	1	3	4	15
R_j	28	13	16	15	18	90

Table 4.4: MASE Ranks

Next, the critical values, T_1 and T_2, for Friedman's test are calculated based on (4.4) and (4.6).

$$T_1 = 9.2$$
$$T_2 = 3.108 > 2.16 \sim F_{(5,20)}$$

Thus, we reject the null hypothesis that all the methods are indifferent. Then, with the null hypothesis of Friedman's test rejected, we can proceed to the multiple comparison test. Based on (4.7), the right-hand side of the inequality is 9.828 with a 0.05 significance level. Assume the hypothesis for each comparison is two-tailed (whether the methods in comparison are equal). Then, the resulting p-values are given in Table 4.5,

Methods	ARIMA	MBB(mean)	MBB(median)	RSB(mean)
MBB(mean)	0.0047	-	-	-
MBB(median)	0.0192	0.5315	-	-
RSB(mean)	0.0121	0.6757	0.8341	-
RSB(median)	0.0465	0.3013	0.6757	0.5315

Table 4.5: Method Comparison p-values

Given the 0.05 significance level for Friedman's test, we can then obtain the significance level for the comparison by using Bonferroni's procedure (discussed in Section 4.2.3), which is 0.005 in this case. Then, in the first column of Table 4.5, we observe that only using moving block bootstrap with mean averaging gives significant results when compared to the ARIMA model under the full datasets. This indicates that one of our proposed combinations differs from the benchmark model with a certain significance level. However, the p-values for other combinations when compared with the ARIMA are insignificant, which means the hypotheses of indifference are not rejected and they failed to improve the ARIMA significantly. Moreover, the p-values between the methods with bootstrap techniques are also insignificant, which means the null hypotheses of indifference are not rejected. Since the p-value does not represent the probability of a null hypothesis being true nor the magnitude of an effect, we cannot compare the size of the p-values in columns 2 to 4 to give any valuable inference [46]. Therefore, the optimal combination of bootstrap and bagging techniques is infeasible to obtain based on current evidence. Nevertheless, combined with the conclusion yielded in the last section, we can still summarise that, under the entire dataset, using mean bagging with moving block bootstrap could improve forecast accuracy for the ARIMA model.

4.5 Conclusion

In this chapter, we applied our proposed method to a real-world dataset consisting of six times series with different frequencies, analysed the quality of the forecast through comparison between three accuracy metrics on data with the same frequency, and conducted the Friedman test on the whole dataset. In the first two sections of this chapter, we introduced the details of the experimental dataset and the methodology for analysis. Then, we implemented our method on one of the data as an example to specify the detailed settings of our method application. In Section 4.4, based on the results of accuracy metrics, we concluded that using bagging with bootstrap could improve the forecast accuracy for the ARIMA model in most cases, and applying the sieve bootstrap could provide constant improvement, yet adopting the moving block bootstrap could be negatively influenced by the non-stationarity of the input data. However, in further analysis of the full dataset, we discovered that, in general, using mean bagging with moving block bootstrap could improve the forecast accuracy for the ARIMA model on data with different frequencies.

Chapter 5

Conclusion

5.1 Discussions and Limitations

From previous research, the results obtained from using a combination of data transformation, data decomposition, bootstrap and bagging exhibited statistical significance in monthly data [4]. Our results also attest to previous research, since for monthly data, both sieve bootstrap and moving block bootstrap with either mean or median bagging could yield statistically significant results. Moreover, in Cordeiro and Neves [13] and Bergmeir et al. [4]'s research, they argued that the results for quarterly and yearly are not promising, due to the fact that the length of quarterly and monthly data is relatively shorter compared to monthly data, weakening the performance of the model. However, in our report, using bagging with bootstrap produced better forecasts than using ARIMA only in yearly data. This may be because the focused models are different in our report (ARIMA) and theirs (Exponential smoothing), and using bagging with bootstrap on ARIMA might be more effective for yearly data. For quarterly data, despite using the moving block bootstrap, which failed to outperform the ARIMA in one case, applying the sieve bootstrap showed consistent improvement in forecast accuracy. The evaluation of quarterly and yearly data complements the research gap in de Oliveira and Cyrino Oliveira [15]'s research since they only considered monthly data, and to some extent, proves the capability of bagging with bootstrap to improve forecast accuracy for the ARIMA model in quarterly and yearly data.

Moreover, it is also vital to identify the limitations of this report. The limitations may exist in the following three areas, data, Loess regression and bootstrap.

- **Data**: Consider prior research, Bergmeir et al. [4] and Cordeiro and Neves [13]'s study utilised the M3 Competition dataset, which consists of a total of 3,003 monthly, quarterly, and yearly data. In contrast, the dataset we employed is significantly smaller in scale. This is because of the lack of computation resources and space in this report. Furthermore, while our dataset includes yearly, quarterly, and monthly data, it neglects data of higher frequency, namely daily and hourly data.

- **Loess regression**: In our report, for simplicity, we set a fixed span for the Loess regression function in R to decompose non-seasonal data. The span we specified might not be optimal, as it is roughly estimated through the fitting graphs for different spans. As discussed in Hastie et al. [25]'s book, generalised cross-validation can be used to obtain the desired span by selecting the span with the lowest root mean square error of prediction (RMSEP).

- **Bootstrap**: In order to perform the moving block bootstrap, we need to specify the block length for each type of data, which in our case are 24 for monthly data and 8 for quarterly and yearly data. These choices might not be the optimal choice for block length

for our data, since the choices for length are taken from other research with different datasets. To address this issue, one may consider using the selection method proposed by Bühlmann and Künsch [10]. By treating the block length as inversely proportional to the bandwidth used in spectral density estimation, their method uses a local bandwidth selection procedure on the time series of the estimated influence function to determine the optimal block length. This approach equates the block length selection to finding an optimal bandwidth, where the estimated optimal block length is chosen as the integer closest to the inverse of this bandwidth, ensuring that the bootstrap method accurately reflects the dependent structure of the time series data.

5.2 Final Conclusion

In this report, we investigate the improvement of forecast accuracy in a widely used time series model, ARIMA, through the application of bagging and bootstrap techniques. The methodology encompasses the following steps: initially, raw data is transformed by a Box-Cox transformation to stabilize variance, followed by the separation of seasonality and trends using STL decomposition or Loess regression. Subsequently, the remainder components are resampled by the moving block bootstrap and the sieve bootstrap, respectively. Resampled data are then recombined with the seasonal and trend components, with ARIMA models applied to produce forecasts. The aggregation of point forecasts involves calculating both mean and median values. Analysing six yearly, quarterly, and monthly datasets with our proposed method reveals that the integration of bagging and bootstrap techniques significantly improves the forecast accuracy of the ARIMA model. Specifically, sieve bootstrap, in comparison to moving block bootstrap, exhibits consistent forecast accuracy improvement in the presence of unstable data. Furthermore, when considering a broader spectrum of data types, applying moving block bootstrap with mean bagging facilitates superior forecast accuracy in the ARIMA model compared to the standard ARIMA model.

For future research direction, we suggest searching for more alternatives for bootstrap methods, as the two methods used in this report were developed decades ago and more advanced variations of these bootstrap methods have emerged. Moreover, a more diverse and sufficient dataset is desirable for similar research in the future.

Appendix A

Related Data and Code

- The related data and R code used in this study have been uploaded to our GitHub code space: `https://github.com/Hongyi-Z/Project_3_Bootstrap.git`.
- As for the R package versions, they are listed in the table below,

Packages	Version
forecast	8.21.1
stats	4.3.2
readr	2.1.5
DescTools	0.99.54
ggplot2	3.4.4

Table A.1: Package Names and Versions Used in This Report

Bibliography

[1] H. Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

[2] T. W. Anderson. *The statistical analysis of time series*. John Wiley & Sons, 1971.

[3] J. Armstrong. *Long-range Forecasting: From Crystal Ball to Computer*. A Wiley Interscience Publication. Wiley, 1978. ISBN 9780471030027. URL `https://books.google.co.uk/books?id=7DAcAAAAIAAJ`.

[4] C. Bergmeir, R. J. Hyndman, and J. M. Benítez. Bagging exponential smoothing methods using stl decomposition and box-cox transformation. *International Journal of Forecasting*, 32(2):303–312, 2016. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2015.07.002. URL `https://www.sciencedirect.com/science/article/pii/S0169207015001120`.

[5] K. N. Berk. Consistent autoregressive spectral estimates. *The Annals of Statistics*, pages 489–502, 1974.

[6] G. Box, G. Jenkins, G. Reinsel, and G. Ljung. *Time Series Analysis: Forecasting and Control*. Wiley Series in Probability and Statistics. Wiley, 2015. ISBN 9781118674925.

[7] G. E. P. Box and D. R. Cox. An analysis of transformations. *Journal of the Royal Statistical Society: Series B (Methodological)*, 26(2):211–243, 1964. doi: https://doi.org/10.1111/j.2517-6161.1964.tb00553.x.

[8] L. Breiman. Bagging predictors. *Machine learning*, 24:123–140, 1996.

[9] P. Bühlmann. Sieve bootstrap for time series. *Bernoulli*, 3(2):123–148, 1997. ISSN 13507265. URL `http://www.jstor.org/stable/3318584`.

[10] P. Bühlmann and H. R. Künsch. Block length selection in the bootstrap for time series. *Computational Statistics & Data Analysis*, 31(3):295–310, 1999. ISSN 0167-9473. doi: https://doi.org/10.1016/S0167-9473(99)00014-6. URL `https://www.sciencedirect.com/science/article/pii/S0167947399000146`.

[11] R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning. Stl: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statisitics*, 6(1):3–73, 1990.

[12] W. Conover. *Practical Nonparametric Statistics, 3rd Edition*. Wiley, 1999. ISBN 978-0-471-16068-7.

[13] C. Cordeiro and M. M. Neves. Forecasting time series with BOOT.EXPOS procedure. *REVSTAT-Statistical Journal*, 7(2):135–149, 2009.

[14] A. C. Davison and D. V. Hinkley. *Bootstrap Methods and their Application*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1997.

[15] E. M. de Oliveira and F. L. Cyrino Oliveira. Forecasting mid-long term electric energy consumption through bagging arima and exponential smoothing methods. *Energy*, 144: 776–788, 2018. ISSN 0360-5442. doi: https://doi.org/10.1016/j.energy.2017.12.049. URL https://www.sciencedirect.com/science/article/pii/S0360544217320820.

[16] B. Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1 – 26, 1979. doi: 10.1214/aos/1176344552. URL https://doi.org/10.1214/aos/1176344552.

[17] B. Efron. Second thoughts on the bootstrap. *Statistical Science*, 18(2):135–140, 2003. ISSN 08834237. URL http://www.jstor.org/stable/3182843.

[18] B. Efron and R. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 1994. ISBN 9780412042317.

[19] D. for Energy Security Net Zero. *Energy Trends*, 2023. URL https://www.gov.uk/government/collections/energy-trends.

[20] J. D. Gibbons. *Nonparametric statistical inference*. McGraw-Hill series in probability and statistics. McGraw-Hill, New York, 1971. ISBN 0070231664.

[21] U. Grenander. *Abstract inference*. Wiley series in probability and mathematical statistics. Wiley, New York, 1981. ISBN 0471082678.

[22] V. M. Guerrero. Time-series analysis supported by power transformations. *Journal of Forecasting*, 12(1):37–48, 1993. doi: https://doi.org/10.1002/for.3980120104.

[23] F. R. Hampel. *Robust Statistics: the Approach Based on Influence Functions*. Wiley series in probability and statistics. Wiley, New York, N.Y.;, 1986. ISBN 9780471735779.

[24] E. J. Hannan. Rational transfer function approximation. *Statistical Science*, 2(2):135–151, 1987.

[25] T. Hastie, R. Tibshirani, and J. Friedman. *Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer, New York, second edition edition, 2009. ISBN 0387848576.

[26] Y. Hochberg. A sharper Bonferroni procedure for multiple tests of significance. *Biometrika*, 75(4):800–802, 12 1988. ISSN 0006-3444. doi: 10.1093/biomet/75.4.800. URL https://doi.org/10.1093/biomet/75.4.800.

[27] S. Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. ISSN 03036898, 14679469. URL http://www.jstor.org/stable/4615733.

[28] A. Hong-Zhi, C. Zhao-Guo, and E. J. Hannan. Autocorrelation, autoregression and autoregressive approximation. *The Annals of Statistics*, pages 926–936, 1982.

[29] R. J. Hyndman and G. Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2021.

[30] R. J. Hyndman and Y. Khandakar. Automatic time series forecasting: The forecast package for r. *Journal of Statistical Software*, 27(3):1–22, 2008. doi: 10.18637/jss.v027.i03. URL https://www.jstatsoft.org/index.php/jss/article/view/v027i03.

[31] R. J. Hyndman and A. B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688, 2006. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2006.03.001. URL https://www.sciencedirect.com/science/article/pii/S0169207006000239.

[32] G. Jenkins and D. Watts. *Spectral Analysis and Its Applications*. Holden-Day series in time series analysis and digital signal processing. Holden-Day, 1969. ISBN 9780816244645.

[33] H. R. Künsch. The jackknife and the bootstrap for general stationary observations. *The Annals of Statistics*, 17(3):1217–1241, 1989. ISSN 00905364. URL http://www.jstor.org/stable/2241719.

[34] K. Lim. *Canadian lynx data*. Kluwer Academic Publishers, 2002. ISBN 1402006098. URL http://encyclopediaofmath.org/index.php?title=Canadian_lynx_data&oldid=14880.

[35] S. Makridakis. *The Forecasting Accuracy of Major Time Series Methods*. Wiley, 1984. ISBN 9780471903277.

[36] S. Makridakis, E. Spiliotis, and V. Assimakopoulos. The m4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1):54–74, 2020. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2019.04.014. URL https://www.sciencedirect.com/science/article/pii/S0169207019301128. M4 Competition.

[37] R. O. of Belgium. Sunspot number, 2024. URL https://www.sidc.be/SILSO/datafiles.

[38] E. Paparoditis and D. N. Politis. Tapered block bootstrap. *Biometrika*, 88(4):1105–1119, 12 2001. ISSN 0006-3444. doi: 10.1093/biomet/88.4.1105. URL https://doi.org/10.1093/biomet/88.4.1105.

[39] F. Petropoulos, R. J. Hyndman, and C. Bergmeir. Exploring the sources of uncertainty: Why does bagging for time series forecasting work? *European Journal of Operational Research*, 268(2):545–554, 2018.

[40] D. N. Politis and J. P. Romano. The stationary bootstrap. *Journal of the American Statistical Association*, 89(428):1303–1313, 1994. ISSN 01621459. URL http://www.jstor.org/stable/2290993.

[41] G. Rupert Jr et al. *Simultaneous statistical inference*. Springer Science & Business Media, 2012.

[42] G. Schwarz. Estimating the dimension of a model. *The annals of statistics*, pages 461–464, 1978.

[43] X. Shao. The dependent wild bootstrap. *Journal of the American Statistical Association*, 105(489):218–235, 2010. doi: 10.1198/jasa.2009.tm08744. URL https://doi.org/10.1198/jasa.2009.tm08744.

[44] N. Sugiura. Further analysis of the data by akaike's information criterion and the finite corrections: further analysis of the data by akaike's. *Communications in Statistics-theory and Methods*, 7(1):13–26, 1978.

[45] H. G. Tucker. A Generalization of the Glivenko-Cantelli Theorem. *The Annals of Mathematical Statistics*, 30(3):828 – 830, 1959. doi: 10.1214/aoms/1177706212. URL https://doi.org/10.1214/aoms/1177706212.

[46] R. L. Wasserstein and N. A. Lazar. The asa statement on p-values: Context, process, and purpose. *The American Statistician*, 70(2):129–133, 2016. doi: 10.1080/00031305.2016.1154108. URL `https://doi.org/10.1080/00031305.2016.1154108`.

[47] P. R. Winters. Forecasting sales by exponentially weighted moving averages. *Management science*, 6(3):324–342, 1960. ISSN 0025-1909.

[48] H. Wold. *A study in the analysis of stationary time series*. PhD thesis, Almqvist & Wiksell, 1938.

[49] G. U. Yule. On a method of investigating periodicities in disturbed series with special reference to wolfer's sunspot numbers. *Philosphical Transactions*, pages 267–298, 1927.