POLITECNICO DI TORINO

Master Program
in Communications and Computer Networks Engineering

Thesis

Robust Bootstrap and GARCH Process Application for Heating Demand in Germany

Relatori
Prof. Dr. Franco Pellerey
Prof. Dr. Abdelhak M. Zoubir
M.Sc. Pertami Kunz
Supervisor signature

. . . . . .
. . . . . .
. . . . . .

Candidato
Stefano Frigiola

Candidate signature

. . . . . .

Academic Year 2021-2022

To my family

Summary

Energy, in all of its forms, is a physical quantity without which one could not carry out all his/her daily activities. However, nowadays, we hear more and more about problems due to lack or waste of energy due to misuse by man and exaggerated consumption of it.

Others problem can be mentioned when speaking about heating. Nowaydays, in fact, due to an always- increasingly cost of it, retirees and people less wealthy cannot afford heating in their homes due to rising prices. Provisions on the heating demand would perhaps allow a choice from a financial point of view more inclined to meet the needs of the citizen, the main consumer. Forecastsing energy prediction is therefore something that would allow man to balance what is required and consumed by energy without encountering problems of overproduction and shortage and to energy supply companies to alleviate the economic burden on households.

To do this, increasingly sophisticated algorithms have been created to allow companies that supply energy to stipulate plans for forecasting consumption, demand, and production of energy itself, allowing them to better manage economic and environmental resources.

In the following, two main methods are reported: the classical residual bootstrap and the Influence Function bootstrap. In particular, the latter is a method capable of managing situations in which there are deviations from the assumed model and, with great satisfaction, the results obtained with this method have been better compared to the others, confirming that even in situations distant from the made assumptions, it is still possible to obtain more accurate future predictions.

Acknowledgements

Special thanks go to my parents, Pietro and Maria, and to my sisters, Paola and Emilia, who have never stopped and will never stop supporting me. Thanks to them who since day zero have always directed me towards the best by helping me in all the most important choices of my life.

Thanks to all my family, uncles and cousins without forgetting Osvaldo and Carmela, who have always encouraged me to make the best of myself.

Thanks to all my friends, a real second family. In the saddest and happiest days, they have always been there. I couldn't ask for anything better.

Thanks to the city of Darmstadt and all the people I met during this incredible year that just ended who made me feel at home from day one.

Thanks to Professor Abdelhak M. Zoubir and his PhD student Pertami Kunz, who introduced me to the bootstrap method for the first time and who gave me the opportunity to be followed by them for the drafting of my master's thesis.

Thanks to Professor Franco Pellerey who, albeit from a distance, agreed to follow me in this project.

Finally, thanks to my high school teacher, Professor Ponte, without whose math teachers I probably would never have undertaken the engineering degree.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

Energy is the physical quantity that measures the ability of a body or a physical system to do work, regardless of whether such work is or can actually be done. Energy is something that modern man cannot do without, as it is crucial for everyday life. We use energy for practically all the activities we carry out on a daily basis: whether it is a hot shower, going to a place using the car, turning on the air conditioner or just sitting and watching TV. Related to energy there is also heating. Heating systems, moreover, can be classified by fuel or energy source used like coal, diesel, gas, wood, geothermal, solar or electric energy. Nowadays, retirees and people who are less wealthy cannot afford heating in their homes due to rising prices of it. Faced with an ever-increasing demand for energy and heating, today we are witnessing depletions of non-renewable energy sources from which we draw for the production of electricity and for heating.
Fossil fuels (oil, coal and natural gas above all) must meet ever-growing energy and heating demands by introducing polluting substances into the atmosphere that are absolutely harmful to humans and the environment. In order to extend our energy reserves as long as possible and safeguard the health of the planet, it is therefore essential to save energy and implement a conscious reduction in consumption.
The solution to reduce environmental stress is energy saving based also on the use of alternative and renewable energies, which contribute improving our health and that of the environment.
To do this, however, plans are needed on forecasts of consumption and production of energy from renewable sources. In this way, plans could be implemented based on these forecasts and which allow exploiting renewable sources as much as possible when, according to forecasts, they can produce sufficient quantities, also allowing the storage of unused quantities to be used in times when renewable sources fail.
In the past, in fact, there have often been periods of great energy production in the face of not so high demand. In these cases there were two solutions, either to lose the excess energy produced or to try to store and conserve it, however having to face the large maintenance costs.
With accurate forecasting methods instead, a balance would be sought between requests and supply at all times, thus avoiding large wastes that would have significant long-term impacts from an environmental point of view.

Energy forecasting, in particular, is a technique to predict future energy needs to achieve demand and supply equilibrium. Forecasting on energy demands (load) and on the relative sale price is becoming nowadays a topic more and more analyzed for mainly environmental and, even more, economic reasons.
Making accurate forecasts on energy requests and consumption would allow energy companies to be able to make competitive decisions within the market reducing expenses and increasing revenues and, with the advent of renewable energy resources (RER), it has become crucial for companies and forecasters to know in advance how much energy RER will produce in the next hours and days.
In the past decades for energy forecasting problems, researchers have tried to use different methods, from linear regression to neural network and techniques with long short-term memory (LSTM). In the following, instead, the idea has been to use ARIMA-GARCH process to model the data in addition to robust Bootstrap algorithm to deal with outliers.
In the first section there is an introduction on what a time series is and on which steps are necessary for performing time series analysis. Particular attention is paid to ARIMA-GARCH model, the one adopted during the study. Then, the idea of bootstrap is presented, giving particular focus to what the residual bootstrap algorithm is. In the second and last section, the idea of robust bootstrap is deepened with an explanation on what the Influence Function (IF) Bootstrap is. Finally, the previously mentioned theory is adopted and the study under case is examined in detail.

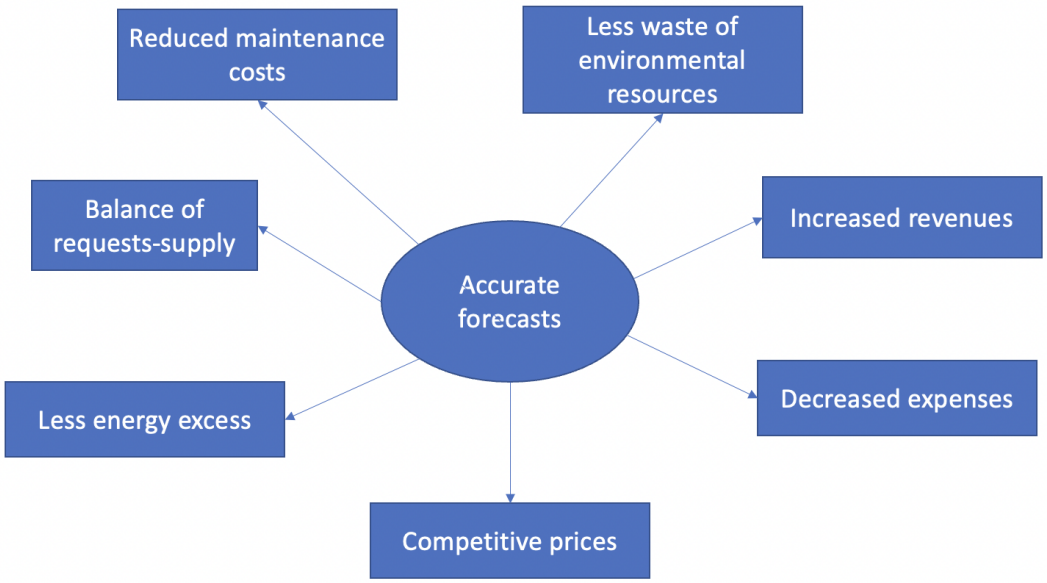

Figure 1.1: Example of possible results of good forecasts

Chapter 2

The problem and common approaches

2.1 Time series analysis

Time has always been a crucial factor since humans started recording data. Time series analysis, allows humans to study the world and understand how it is progressing by extracting meaningful statistics and other characteristics of the data.
Time series analysis is a specific way of analyzing a set of data collected over an interval of time by analysts who record data points at consistent intervals over a set of period of time rather than just recording the data points intermittently or randomly.
Time series data, differently from other kinds of data, can show how variables change over time. It usually requires numerous data points to reach consistency and reliability because in this way it is possible to have a representative sample size and that analysis can cut through noisy data.
Time series analysis helps organizations understand the underlying causes of trends or systemic patterns over time. Using data visualizations, business users can see seasonal trends and dig deeper into why these trends occur. With modern analytics platforms, these visualizations can go far beyond line graphs.
When organizations analyze data over consistent intervals, they can also use time series forecasting to predict the likelihood of future events. Time series forecasting is part of predictive analytics. It can show likely changes in the data, like seasonality or cyclic behavior, which provides a better understanding of data variables and helps forecast better. Examples of time series analysis in action include:

- heating demand
- weather data
- stock prices
- temperature readings

- energy consumptions predictions
- heart rate monitoring.

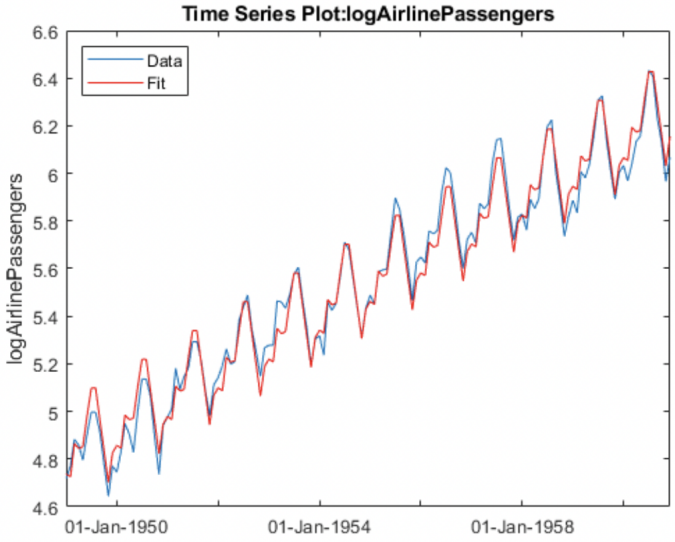

Figure 2.1: Example of monthly counts of airline passengers from 1949 to 1960
Image source: `https://de.mathworks.com/help/stats/time-series-regression-of-airline-passenger-data.html`

Since time series analysis includes many categories or variations of data, is very common to make complex models when dealing with time series data. Models that are too complex or that try to do too many things can lead to a lack of fit, being not able anymore to distinguish between random error and true relationship and leaving and to make correct forecast predictions.
make correct forecast predictions. As described in Tableau, models of time series analysis include:

- **Classification:** identifies and assigns categories to the data for example identify a word based on series of hand movement in sign language.
- **Curve fitting:** the process of constructing a curve, or mathematical function, with the best fit to a series of data points.
- **Descriptive analysis:** identifies patterns in time series data, like trends, cycles, or seasonal variation.
- **Explanative analysis:** attempts to understand the data and the relationship within it, as well as cause and effect.
- **Exploratory analysis:** highlights the main characteristics of the time series data, usually in a visual format.

- **Forecasting:** predicts future data. It uses historical data as a model for future data, predicting scenarios that could happen along future plot points.
- **Intervention analysis:** studies how an event can change the data.
- **Segmentation:** splits the data into segments to show the underlying properties of the source information. For example, the audio signal from a conference call can be partitioned into pieces corresponding to the times during which each person was speaking.

When dealing with time series data, it is important also to distinguish between the different possible components available, since each one of these requires different strategies to deal with:

- **Trend:** in which there is no fixed interval and any divergence within the given dataset is a continuous timeline. The trend would be negative, positive, or null and basically represents an increase or decrease in time-series value over time.
- **Seasonality:** in which regular or fixed interval shifts within the dataset in a continuous timeline. It refers to periodic fluctuations. For example, electricity consumption is high during the day and low during the night.
- **Cyclical:** in which there is no fixed interval, uncertainty in movement and its pattern.
- **Irregularity:** unexpected situations/events/scenarios and spikes in a short time span.

Usually, one tries to remove these components in order to have a stationary time series, data where there is no trend or seasonality information present in it. While discussing time series data types, there are so two major types: stationary and non-stationary.
For being stationary, a dataset should comply with the following requirements:

- the mean value should be completely constant in the data during the analysis.
- the variance should be constant with respect to the time frame.
- the covariance is independent of time.

Basically, a time series is said to be stationary if its statistical properties do not change over time.
During the time series analysis is important to verify if the dataset is stationary or not and this can be done considering different test:

- **Augmented Dickey-Fuller (ADF) test** also known as Unit Root Test. The ADF has the following assumptions:
  - Null hypothesis (H_0): series is non-stationary.

– Alternate hypothesis (H_1): series is stationary.

If p-value > 0.05, no evidence to reject (H_0).
If p-value ≤ 0.05, evidence to reject (H_0).

- **Kwiatkowki-Phillips-Schmidt-Shin (KPSS)** used for testing a null hypothesis (H_0), that will perceive the time series, as stationary around a deterministic trend against the alternative of a unit root.

For more information concerning the ADF and the KPSS tests, Mushtaq [2011] and Shin and Schmidt [1992] respectively, describe the two techniques in detail.
In time series analysis, it is mandatory to deal with stationary time series. To do that, Shanthababu Pandian shows there are two main methods for converting a non stationary time series into a stationary one:

- **Detrending:** it involves removing the trend effects from the given dataset and showing only the differences in values from the trend. It allows the cyclical pattern to be identified.
  The example below shows how to remove a linear trend from daily closing stock prices to emphasize the price fluctuations about the overall increase.

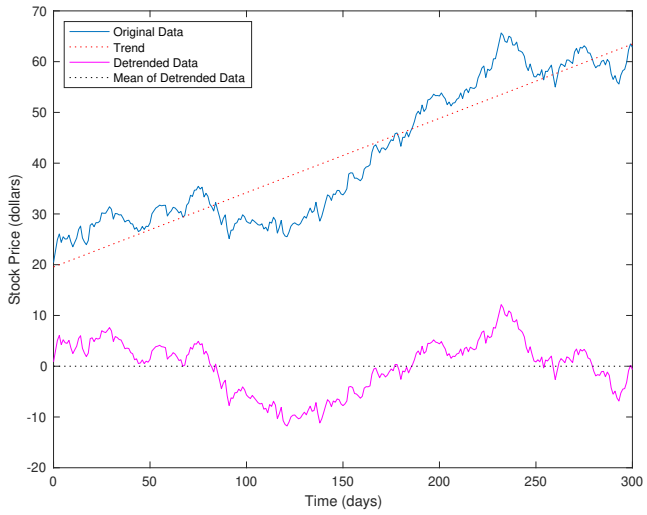

Figure 2.2: Example of detrending of daily closing stock prices
Image source:
`https://de.mathworks.com/help/matlab/data_analysis/detrending-data.html`

- **Differencing:** it consists in a simple transformation of the series into a new one used to remove the series dependence on time and stabilize the mean of the time series. In this way, trend and seasonality are reduced during the transformation.

The example below shows a nonseasonal difference of a time series, the quarterly U.S. GDP measured form 1947 to 2005. In particular a first difference of the series has been applied (more on this in Section 2.2):

$$\Delta y_t = y_t - y_{t-1}. \tag{2.1}$$

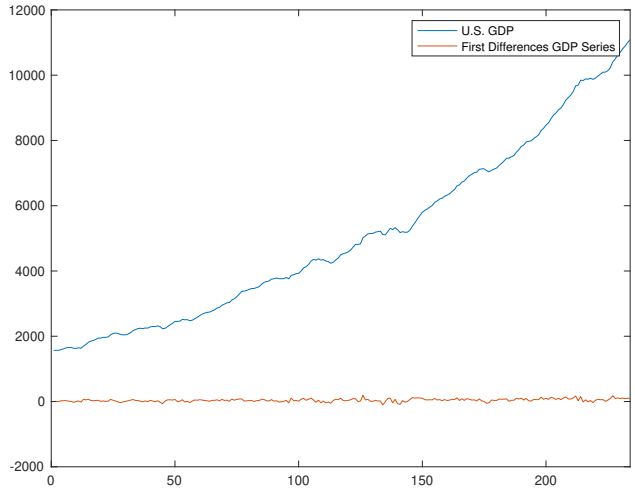

Figure 2.3: Example of Nonseasonal differencing of a time series
Source by: `https://de.mathworks.com/help/econ/nonseasonal-differencing.html`

After having adjusted the time-series, it is possible to proceed by performing what is called time-series forecasting, one of the most applied data science technique in business, finance and, weather heating and energy predictions.
Time series forecasting means to forecast or to predict the future value over a period of time. It entails developing models based on previous data and applying them to make observations and guide future strategic decisions.
The future is predicted or estimated based on what has already happened. Time series adds a time order dependence between observations.
This dependence is both a constraint and a structure that provides a source of additional information. Time series forecasting is a technique for the prediction of events through a sequence of time. It predicts future events by analyzing the trends of the past, on the assumption that future trends will hold similar to historical trends.
For performing predictions and forecasts of a time series, a model is needed but before it, is also useful to make a distinction between univariate and multivariate time series models.
The former use only variable without any external data and are based only on the relationship between past and present. The latter, instead, use multiple variable and can also use external data and is based on relationship between past and present and also between variables.

Having said this, there are different families of models:

- **Classical time series models:** are based on temporal variation inside a time series and work well with univariate time series.
- **Supervised models:** are a family of models used for machine learning task. A machine learning model is supervised when it uses clearly defined input variables and one or more output (target) variables.
- **Deep learning and recent models:** the increasing popularity of deep learning and artificial intelligence over the last years, has opened new possibilities for forecasting as well, as specific deep learning approaches have been invented that work well on time series data.

Classical time series models, in particular, are the most used and are the one adopted in the following study. They can be divided in different categories:

- **ARIMA family:** is a set of smaller models that can be combined. Each part of the ARIMA model can used as a stand-alone component, or the different building blocks can be combined.
  - **Autoregression (AR):** regression model that explains a variable's future value using its past (lagged) values. The order of an AR model (denoted as p), represents the number of lagged values to include in the model.
  - **Moving average (MA):** it uses the prediction error in previous time steps to predict the future. The number of steps back in time is the order q of the MA model.
  - **Autoregressive moving average (ARMA):** combines AR and MA and can therefore use both values and prediction errors from the past.
  - **Autoregressive integrated moving average (ARIMA):** it includes automatic differencing to the ARMA model by including a new parameter, d, representing the number of times that the time series needs to be differenced, in order to remove the series dependence on time and stabilize the mean of the time series.
  - **Seasonal autoregressive integrated moving average (SARIMA):** it adds seasonal effects into the ARIMA model. When dealing with SARIMA models, the parameters to consider are more. The most general representation is SARIMA$(p,d,q)(P,D,Q)_m$, where m is the number of observations per year (monthly data has $m = 12$, ad example), the letters (p,d,q) represent the non seasonal orders while capital letters (P,D,Q) represent the seasonal orders.
    For more information on SARIMA model, see `https://www.real-statistics.com/time-series-analysis/seasonal-arima-sarima/sarima-models/`.

  - **Seasonal autoregressive integrated moving average with exogenous regressors (SARIMAX):** whenever there is any variable that could help to improve the model, the model adopted should be SARIMAX where the X stands for external variables.
    For more information on SARIMAX model, see `https://www.statsmodels.org/stable/examples/notebooks/generated/statespace_sarimax_stata.html`.

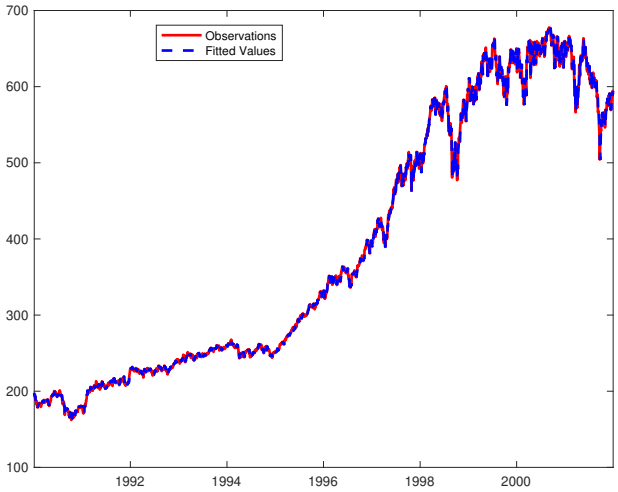

Figure 2.4: Example of ARIMA model fitted to NYSE composite closing prices
Image source: `https://de.mathworks.com/help/econ/arima.estimate.html`

- **Smoothing:** allows to make the curve smoother so that long term variability becomes more evident while short term patterns (noisy) are removed.

  - **Simple moving average:** consists of replacing the current value by the average of the current and a few past values.

  - **Simple exponential smoothing:** it takes a weighted average of past values such that value further back will count less in comparison to more recent values.

  - **Double exponential smoothing:** is used in presence of trends since the simple exponential smoothing is not able to distinguish between variations and trends correctly.

2.2 Stationary time series

Stationarity is an important concept in the field of time series analysis with considerable influence on how data is perceived and predicted.
For data to be stationary, the statistical properties of a system do not change over time, meaning that the overall behavior of the data should remain constant.
From a visual perspective, time plots not showing trends or seasonality can be considered stationary.
If it's not the case, data has to be transformed in order to be able to forecast the series using traditional time series models.
Different methods can be applied in order to reach stationarity:

- **Difference transform:** is a transform that helps stabilize the mean of the time series by removing changes in the level of a time series, which eliminates trend and seasonality.
  Let L be the lag operator such that:

  $$L^j x_t = x_{t-j}, t, j \in Z,$$

  and let Δ denote the difference operator.
  So the lag operator shifts a time value x_t back by j periods and it can be applied to all values in a series x_t and the result is a new series shifted back by j periods x_{t-j}.
  The difference operator is expressed as:

  $$\Delta^i x_t = (1 - L)^j \, x_t. \tag{2.2}$$

  Some combinations of the following difference operators were considered:

  - Δ_1 corresponds to the nonseasonal differencing to address the linear trend,
  - Δ_{24} corresponds to the number of hours in a day,
  - Δ_7 corresponds to the number of days in a week,
  - Δ_{365} corresponds to the number of days in a year,
  - Δ_{12} corresponds to the number of months in a year.

  The transformation that takes into account all the above operators is obtained by multiplying the corresponding lag operator polynomials:

  $$H(x_t) = \Delta_1\Delta_7\Delta_{12}\Delta_{24}\Delta_{365}x_t = (1-L)(1-L^7)(1-L^{12})(1-L^{24})(1-L^{365})x_t. \tag{2.3}$$

Let the lag operator polynomial be:

$$A(L) = 1 + \phi_1 L_1 + ... + \phi_d L^d, \phi_1, ..., \phi_d \in \{-1,1\}\,. \tag{2.4}$$

then the inverse seasonal difference operator is given by:

$$A^{-1}(L) = 1 - \phi_1 L^1 - ... - \phi_d L^d \tag{2.5}$$

- **Log transformation:** sometimes, differencing is not enough to remove trends in all non-stationary data and, for solve this problem before performing it, the time series can be transformed in a logarithmic scale.
  Log transformation can be used to stabilize the variance of a series with non-constant variance. The only limitation is represented by the fact that it can be applied only to positively valued time series. For negative data, one could add a suitable constant to make all the series positive before applying the transformation and remove it from the model before predicting future values.

- **Augmented Dickey-Fuller (ADF) test:** also known as the unit root test, a statistical test testing the null hypothesis that a unit root is present in a time series sample. Unit root is a characteristic of a time series that makes it non-stationary.
  An ADF test, tests the null hypothesis that $\alpha = 0$ (being α the coefficient of the first lag on x) in the following equation:

$$x_t = c + \beta t + \alpha x_{t-1} + \sum_{i=1}^{p} \phi_i \Delta x_{t-i} + \epsilon_t, \tag{2.6}$$

  where:

  – x_{t-1} is the lag 1 of time series,

  – Δx_{t-1} is the first difference of the series at time (t-1).

  Since the null hypothesis assumes the presence of unit root, that is $\alpha = 1$, the p-value obtained should be smaller than the significance level in order to reject the null hypothesis.

2.3 ARIMA-GARCH process

Time series analysis is a major branch in statistics that mainly focuses on analyzing data set to study the characteristics of the data and extract meaningful statistics in order to predict future values of the series.
The two most common approaches used in time series analysis to perform forecast of the series are frequency-domain (Fourier Transform) and time-domain (ARIMA and GARCH methods).
In the following study, the applied approach is the latter so particular attention has been spent on the properties of ARIMA and GARCH processes.

2.3.1 Autoregressive integrated moving average (ARIMA)

An autoregressive integrated moving average, or ARIMA, is a statistical analysis model that uses time series data to either better understand the data set or to predict future trends based on past values.
This model is a form of regression analysis that can be understood by outlining each of its components:

- **Autoregression (AR):** refers to a model that shows a changing variable that regresses on its own lagged values,
- **Integrated (I):** represents the differencing of raw observations to allow for the time series to become stationary,
- **Moving Average (MA):** incorporates the dependency between an observation and a residual error from a moving average model applied to lagged observations.

When dealing with ARIMA models, three parameters must be considered:

- *p:* the number of lag observations in the model,
- *d:* the number of times that the raw observations are differenced,
- *q:* the size of the moving average window.

ARIMA combines autoregressive features with those of moving averages. An AR(1) autoregressive process, for example, is one in which the current value is based on the preceding one, while an AR(2) process is one in which the current value is based on the previous two values.
A moving average, instead, is a calculation used to analyze data points by creating a series of averages of different subsets of the full data set in order to smooth out the influence of outliers.
As a result of this combination of techniques, ARIMA models can take into account trends, cycles, seasonality, and other non-static types of data when making forecasts.
General form of the ARMA(p,q) model:

$$X_t = \phi_0 + \epsilon_t + \sum_{j=1}^{p} \phi_j X_{t-j} + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}, t \in Z, \tag{2.7}$$

where:

- ϵ_t follows the standard normal distribution,
- $\phi_j = 1 - \phi_1 z -, ..., -\phi_p z^p$ meet the requirements of stationarity,
- $\alpha_i = \alpha_0 + \alpha_1 z + ... + \alpha_q z^q$, meet the requirements of reversibility (not having probabilistic properties depending on the direction of time).

General form of the ARIMA(p,d,q) model:

$$Y_t = (1 - L)^d X_t, \tag{2.8}$$

is a sequence of ARMA(p,q), and it indicates that X_t is a sequence of ARMA(p,q) and the model is shown as follows:

$$\phi(L)(1 - L)^d X_t = \alpha(L)\epsilon_t, t \in Z, \tag{2.9}$$

where:

- L represent the lag operator,
- (1-L) represents the finite difference operator,
- ϵ_t is defined as before,
- $\phi_Z = 1 - \phi_1 z -, ..., -\phi_p z^p$ is defined as before,
- $\alpha_z = \alpha_0 + \alpha_1 z + ... + \alpha_q z^q$ is defined as before.

2.3.2 Generalized autoregressive conditional heteroskedasticity (GARCH)

The autoregressive conditional heteroskedasticity (ARCH) model is a statistical model for time series data describing the variance of the current error term or innovation as a function of the actual sizes of the previous time periods error terms. This model is appropriate when the error variance in a time series follows an autoregressive (AR) model; if an autoregressive moving average (ARMA) model is assumed for the error variance, the model is called generalized autoregressive conditional heteroskedasticity (GARCH).
ARCH models are often used in modeling financial time series exhibiting time-varying volatility (the degree of variation of the series over time, usually measured as the standard deviation of the logarithmic returns) .
Let ϵ_t be the error terms; these terms are splitted into a sthocastic piece η_t and a time-dependent conditional variance h_t characterizing the typical size of the terms such that:

$$\epsilon_t = \sqrt{h_t}\eta_t, \tag{2.10}$$

where the innovations η_t are indipendent and identically distributed (i.i.d) random variables such that:

- $E[\eta_t] = 0$,
- $E[\eta_t^2] = 1$,
- follow a symmetric distribution $E[\eta_t^3] = 0$,
- $E[\eta_t^4] \leq \infty$.

Then, the series h_t is modeled by:

$$x_t = f\left(t, x_{t-1}, x_{t-2}, ...\right) + \epsilon_t, \tag{2.11}$$

$$h_t = \omega_0 + \sum_{i=1}^{s} \omega_i \epsilon_{t-i}^2, \tag{2.12}$$

where:

- ω_i is nonnegative,
- $f\left(t, x_{t-1}, x_{t-2}, ...\right)$ is the deterministic information fitting model of the series x_t.

If an autoregressive moving average model is assumed for the error variance, the model is generalized autoregressive conditional heteroskedasticity model and, in that case, the GARCH(r,s) model, is given by:

$$x_t = f\left(t, x_{t-1}, x_{t-2}, ...\right) + \epsilon_t, \tag{2.13}$$

$$\epsilon_t = \sqrt{h_t}\eta_t, \tag{2.14}$$

$$h_t = \omega + \sum_{i=1}^{s} \omega_i \epsilon_{t-i}^2 + \sum_{j=1}^{r} \gamma_j h_{t-j}. \tag{2.15}$$

where ω_i and γ_j are nonnegative.

It is an extension of the ARCH model and claims that h_t has AR $\sum_{j=1}^{r} \gamma_j h_{t-j}$ and ARCH term is $\sum_{i=1}^{s} \omega_i \epsilon_{t-i}^2$.
Basically this model is characterized by having a conditional variance not only influenced by past residuals but also by the lag of conditional variance themselves.

2.3.3 ARIMA-GARCH Models

In general time series modeling must fulfill the assumption of homoskedasticity (constant variance).
However, some kind of data (is especially the case of stock prices and inflation rates) usually show the phenomenon of cluster volatility, a period in which their prices show alternating changes for a long period followed by period indicating a stable state.
This situation can cause variance data not to be constant (heteroskedasticity). To overcome this situation the ARIMA(p,q)-GARCH(r,s) model can be used.

In particular, the model is the same as the classical ARIMA(p,q) , but now the error term ϵ_t does not follow anymore a standard normal distribution but follows the same distribution described in the case of GARCH models:

$$X_t = \phi_0 + \epsilon_t + \sum_{j=1}^{p} \phi_j X_{t-j} + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}, t \in Z, \tag{2.16}$$

$$\epsilon_t = \sqrt{h_t}\eta_t, \tag{2.17}$$

$$h_t = \omega_0 + \sum_{i=1}^{s} \omega_i \epsilon_{t-i}^2 + \sum_{j=1}^{r} \gamma_j h_{t-j}. \tag{2.18}$$

2.4 ARIMA order selection: AIC and BIC

When fitting a model to a data set it is important to assess the performance of the model with respect to how well it explains the data.
Supposing the goal is to select from among several candidate models, which criterion can be used to select the best model?
Given a set of data, the objective is to determine which of the candidate models best approximates the data and this involves trying to minimize the loss of information.
For doing so, as suggested by Hyndman and Athanasopoulos [2018] are the most common approaches in model selection: the Akaike information criterion (AIC) and the Bayesian information criterion (BIC).

- **AIC:** the Akaike information criterion estimates, given a collection of model for the data, the quality of each of them, relative to each of the other models. When fitting a statistical model to represent a data set, the representation will almost never be exact.; so some information is lost by using the model to represent the process and, the goal of AIC, is to estimate it.
  Less information is lost by a model, higher is the quality of that model.
  The AIC is computed as follows:

$$\text{AIC} = -2 \log L\left(\hat{\theta}\right) + 2k, \tag{2.19}$$

  where:

  - θ is the set of model parameters,
  - $L\left(\hat{\theta}\right)$ is the likelihood of the candidate model given the data when evaluated at the maximum likelihood estimate of θ,
  - k is the number of estimated parameters in the candidate model.

  The first component, $-2 \log L\left(\hat{\theta}\right)$, is the value of the log-likelihood function, $\log L\left(\hat{\theta}\right)$, which is the probability of obtaining the data given the candidate model. Since the log-likelihood function's value is multiplied by -2, ignoring the second component, the model with the minimum AIC is the one with the highest value for the log-likelihood function.
  However, to this first component an adjustment is added based on the number of estimated parameters. The more parameters, the greater the amount added to the first component, increasing the value for the AIC and penalizing the model.
  A trade-off is then present when AIC is adopted, between the goodness of fit of the model and the simplicity of it. Basically, AIC deals with both risk of overfitting and that of underfitting.
  When fitting models, it is possible to increase the log-likelihood by adding parameters, but doing so may result in overfitting. The AIC tries to resolve the problem by introducing a penalty term for the number of parameters in the model.
  When dealing with ARIMA models, it can be rewritten as:

$$\text{AIC} = -2 \log L\left(\hat{\theta}\right) + 2\left(p + q + k + 1\right), \tag{2.20}$$

where:

- $k = 1$ if the constant term in ARIMA model is different from zero,
- $k = 0$ if the constant term in ARIMA model is equal to zero.

- **BIC:** the Bayesian information criterion (BIC), is a criterion for model selection among a finite set of models; usually the model with lower BIC is preferred.
  The difference with AIC is the greater penalty term imposed for the number of parameters by the BIC.
  The BIC is computed as follows:

$$\mathrm{BIC} = -2 \log L\left(\hat{\theta}\right) + k \log n, \tag{2.21}$$

  where the terms are the same as described before for the AIC and n is the number of observations.
  When dealing with ARIMA models, it can be rewritten as:

$$\mathrm{BIC} = \mathrm{AIC} + [\log n - 2] - (p + q + k + 1)\,. \tag{2.22}$$

2.5 ACF and PACF

Even if the order of the ARIMA(p,q) model can be detected usig AIC and BIC criterions, another way is to look graphically at the ACF (autocorrelation function) and PACF (partial autocorrelation function) of the time series.
Autocorrelation is the correlation between two observations at different points in a time series. For example, values that are separated by an interval might have a strong positive or negative correlation. When these correlations are present, they indicate that past values influence the current value.
Mathematically speaking, the observations y_t and y_{t-k} are separated by k time units, where k is the lag, the number of intervals between two observations.
For each lag, there is a correlation. To identify which lags have significant correlations, understand the patterns and properties of a time series, the autocorrelation function (ACF) is adopted.
The ACF assesses the correlation between observations in a time series for a set of lags and, for two variables y_t and y_{t+k}, is given by:

$$\rho_k = \text{Corr}\left[y_t, y_{t+k}\right] = \frac{E\left[\left(y_t - \mu_t\right)\left(y_{t+k} - \mu_{t+k}\right)\right]}{\sigma_t \sigma_{t+k}} = \frac{\text{Cov}\left(y_t, ..., y_{t+k}\right)}{\sigma_t \sigma_{t+k}}, \tag{2.23}$$

where:

- E is the expected value operator,
- μ_t and μ_{t+k} are the means respectively for y_t and y_{t+k},
- σ_t and σ_{t+k} are their standard deviations.

The partial autocorrelation function is similar to the ACF but captures, instead, a direct correlation between time series and a lagged version of itself.
Is quite similar to ACF, since is defined as the correlation between two variables but, now, depending on all the observations that are in between:

$$\rho_k = Corr\left[y_t, y_{t+k} | y_{t+1}, ..., y_{t+k-1}\right]. \tag{2.24}$$

But what's the relation between ACF and PACF plots and ARIMA models?
It is possible to select the order p for AR(p) model based on significant spiked from the PACF plot. One more indication of the AR process is that the ACF plot decays more slowly.
In contrast to the AR model, it is possible to choose the order q for model MA(q) from ACF if this plot has a sharp cut-off after lag q. One more indication of the MA process is that the PACF plot decays more slowly.

Consider now a time series generated by an autoregression (AR) process with a lag of k. Since the ACF describes the autocorrelation between an observation and another one at a prior time step that includes direct and indirect dependence information, we woud expect

the ACF for AR(k) time series to be strong to a lag of k and the inertia of that relationship would carry on to subsequent lag values, trailing off at some point as the effect was weakened.
The PACF, instead, only describes the direct relationship between an observation and its lag and this would suggest that there would be no correlation for lag values beyond k.

2.6 Engle's test for heteroskedasticity

After having fitted an ARIMA(p,d,q) model and decided the best order of it, another test can be performed to confirm or reject the heteroskedasticity of the residuals; this test is also known as the Engle's test.
In the following, as suggested by Engle [1982], a simple time series is considered but the test can be performed in the same way considering more complex models, like ARIMA(p,d,q).

1. Consider a time series:
$$y_t = \mu_t + \epsilon_t, \tag{2.25}$$
where:
   - μ_t is the conditional mean of the process,
   - ϵ_t is an innovation process with mean zero.

2. Suppose the innovations are generated as:
$$\epsilon_t = \sigma_t z_t, \tag{2.26}$$
where z_t is an indipendent and identically distributed process with mean 0 and variance 1.
Thus $E[\epsilon_t \epsilon_{t+h}] = 0$ for all lags $h \neq 0$ and the innovations are uncorrelated.

3. Let H_t denote the history of the process available at time t. The conditional variance of y_t is:
$$\mathrm{Var}(y_t|H_{t-1}) = \mathrm{Var}(\epsilon_t|H_{t-1}) = E\left(\epsilon_t^2|H_{t-1}\right) = \sigma_t^2. \tag{2.27}$$
Thus, the conditional heteroscedasticity in the variance process is equivalent to the autocorrelation in the squared innovation process.

4. Define the residual series:
$$\epsilon_t = y_t - \hat{\mu}_t. \tag{2.28}$$
If all autocorrelation in the original series, y_t, is accounted for in the conditional mean model, then the residuals are uncorrelated with mean zero. However, the residuals, can still be serially dependent.

5. The alternative hypothesis for Engle's ARCH test is autocorrelation in the squared residuals, given by the regression:
$$H_a : \epsilon_t^2 = \alpha_0 + \alpha_1 \epsilon_{t-1}^2 + ... + \alpha_m \epsilon_{t-m}^2 + \mu_t, \tag{2.29}$$
where μ_t is a white noise error process.
The null hypothesis is:
$$H_0 : \alpha_0 = \alpha_1 = ... = \alpha_m = 0. \tag{2.30}$$

Basically the null hypothesis is that, in the absence of ARCH components, $\alpha_i = 0$ for all $i = 1, ..., m$. The alternative hypothesis is that, in the presence of ARCH components, at least one of the estimated α_i coefficients must be significant. After performing the Engle's ARCH test, is possible, using again AIC and BIC criterion, to choose the best ARIMA(p,q,d)-GARCH(r,s) model.

2.7 Time series forecasting

The concept of time series forecasting occurs when, through historical analysis and by building models, future predictions are made based on previous historical data.
Is basically the process of analyzing time series data using statistics and modeling to make predictions on the future outcomes.
In the following, time series forecasting has been performed in the differenced log domain and three different kinds of techniques have been analyzed:

- Forecasts through the Matlab function '*forecast*' which basically forecast univariate autoregressive integrated moving average (ARIMA) model responses or conditional variances.
- Classical Residual Bootstrap, where the idea is to make future predictions based on the residuals of the ARIMA-GARCH model. In particular the bootstrap technique is applied to the standardized residuals in order to generate paths of future outcomes without making assumptions about the probability distribution of them.
- Influence Function Bootstrap, where the idea is pretty close the one of Classical Residual Bootstrap with the difference that, here the IF is used to resample with probabilities in order to reduce outliers effects.

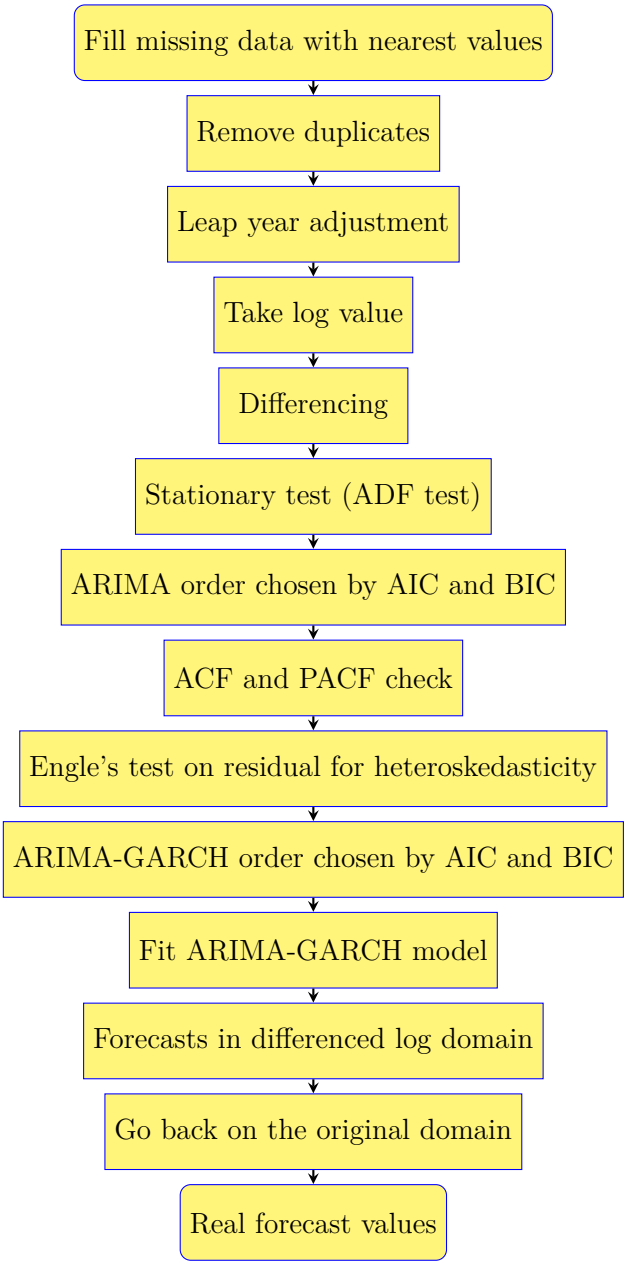

Figure 2.5: Steps for time series analysis

2.8 Case under study

A dataset from `https://data.open-power-system-data.org` (an open power system data platform providing European power system data) has been used, describing the total heating demand in Megawatt in Germany from 2015 to 2020.
The dataset has been modified such that missing values were imputed by taking the nearest non-empty value; moreover, the data points from February 29th of leap years were removed and, to better capture the relative change between subsequent data points and lower the computational range, the sample points are transformed by taking their natural log values.

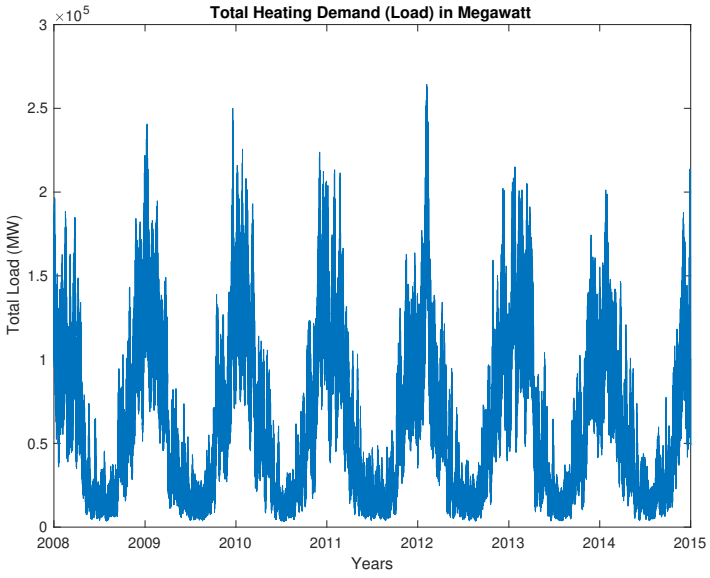

Figure 2.6: Dataset of heating demand in Germany

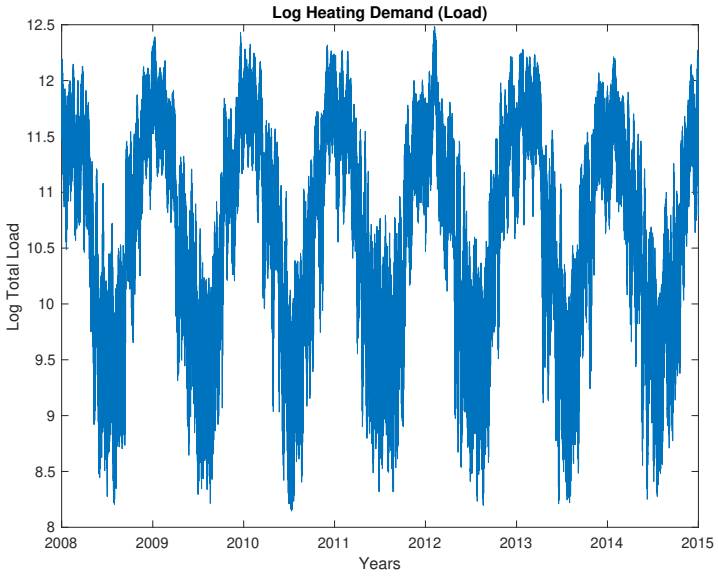

Figure 2.7: Logarithmic representation of heating demand in Germany

After that, the series has been detrended and seasonal effects have been removed by performing the before mentioned differencing method.
In particular, for this dataset, the combination $H\left(x_t\right) = \Delta_1 \cdot \Delta_{12} \cdot \Delta_{24} x_t, t = 1, ..., T$, was found to be the best.
Once Augmented Dickey-Fuller (ADF) test has been performed on H(x_t), the series could be considered stationary since the rejection of the hypothesis that there is a unit root was confirmed.

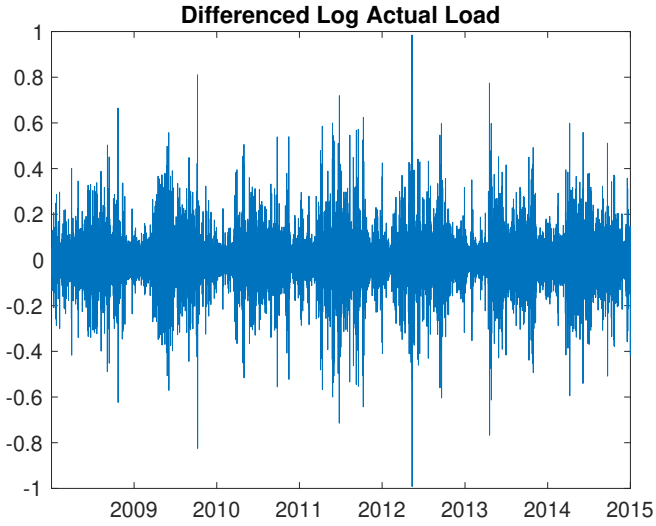

Figure 2.8: Nonseasonal and seasonal differenced log heating demand

Then, the AIC and BIC criterion have been considered in order to choose the right ARIMA order.
The minimum values of AIC and BIC, in particular, have been obtained with p=1 and q=1 and used in order to fit the data to the ARIMA model; moreover, the ACF and PACF of the differenced logarithmic dataset have been plotted to graphically understand the possible moving average and autoregressive orders and to confirm, once more, the correct ARIMA orders.

p \ q	1	2	3	4
1	**-6.8925**	-5.9117	-6.6158	-6.2038
2	-5.9666	-4.4964	-4.7185	-5.1511
3	-6.0872	-3.7490	-5.0644	-5.0156
4	-5.7726	-3.4614	-3.3556	-6.2295

Table 2.1: AIC calculation results ($\cdot 10^4$) for ARMA order

p \ q	1	2	3	4
1	**-6.8910**	-5.9106	-6.6145	-6.2031
2	-5.9656	-4.4951	-4.7170	-5.1493
3	-6.0859	-3.7475	-5.0625	-5.0136
4	-5.7710	-3.4596	-3.3536	-6.2271

Table 2.2: BIC calculation results ($\cdot 10^4$) for ARMA order

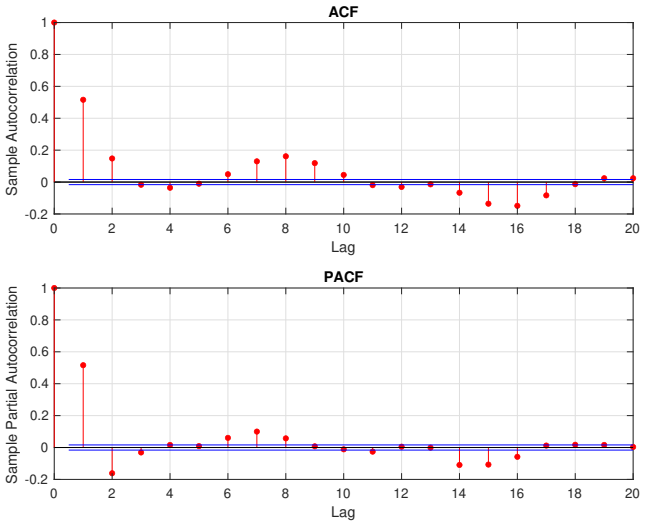

Figure 2.9: ACF and PACF of differenced logarithmic data

As showed by Figure 2.9, both the ACF and PACF have the biggest correlation value at lag 0, and then they start slowly decreasing. This result, based on the theory discussed previously, means that is reasonable to adopt an ARIMA model with $p = 1$ and $q = 1$, as suggested before by the AIC and BIC criterion.
Having found the order, an ARMA(1,1) model has been constructed and fitted to the data. But, before doing that, the residuals have been analyzed in order to perform an Engle test to check for the presence of ARCH effects.

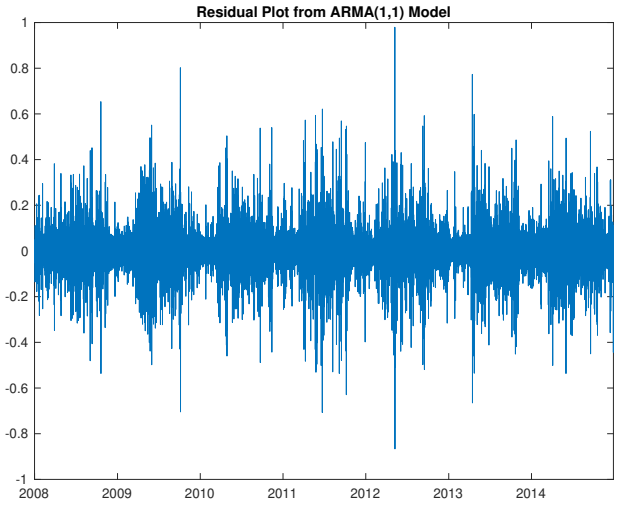

Figure 2.10: ARIMA model residual

Beyond the residuals, also the volatility, obtained by taking the square root of the conditional variance, was analyzed.
Furthermore, the residuals have been also standardized by taking the ratio of their values and the volatility. The standardized residuals were then used for performing bootstrap based forecasting.

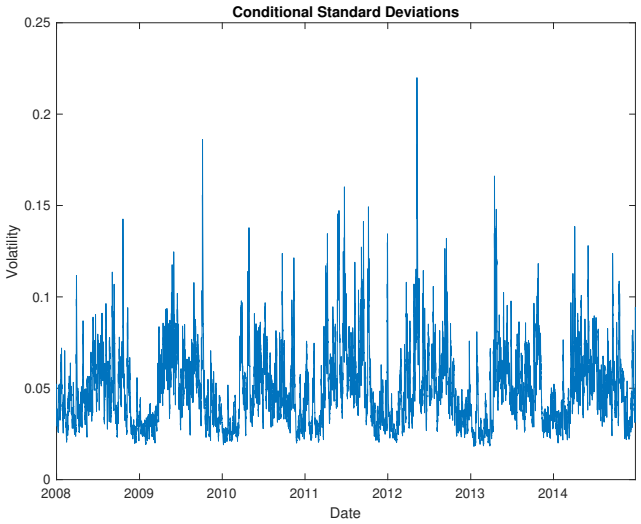

Figure 2.11: Conditional standard deviation of the ARMA model

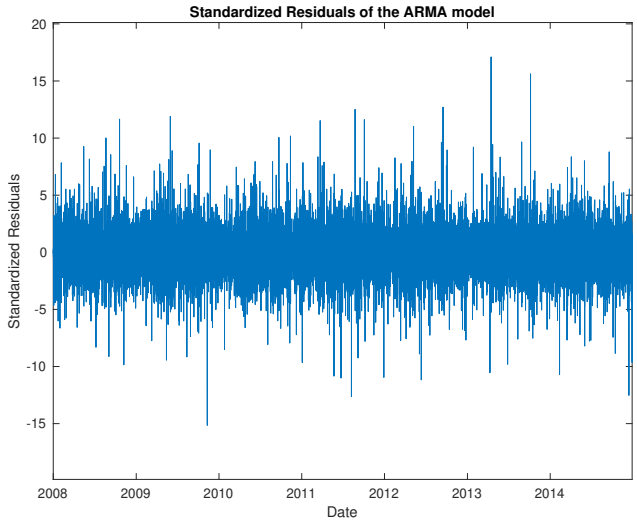

Figure 2.12: Standardized residuals of the ARMA model

Then, the Engle test has been performed on the residual indicating that the null hypothesis of homoskedasticity must be rejected. In other words, residuals are heteroskedastic and so, the composite model ARIMA(p,d,q)-GARCH(r,s) was considered.
Again, AIC and BIC were used in order to get, this time, not only the ARIMA orders but also those of the GARCH process. The most parsimonious model was chosen to be ARIMA(1,1,1)-GARCH(1,1).

p \ q	1	2	3	4
1	**-7.7239**	-7.1342	-7.3981	-6.8984
2	-7.1668	-5.4369	-5.6055	-4.3500
3	-6.8174	-4.6190	-5.9132	-4.2164
4	-6.9153	-4.3477	-4.2170	-6.6766

Table 2.3: AIC calculation results ($\cdot 10^4$) for GARCH order

p \ q	1	2	3	4
1	**-7.7226**	-7.1331	-7.3973	-6.8969
2	-7.1657	-5.4356	-5.6039	-4.3482
3	-6.8161	-4.6174	-5.9113	-4.2143
4	-6.9137	-4.3459	-4.2149	-6.6743

Table 2.4: BIC calculation results ($\cdot 10^4$) for GARCH order

Having found the correct ARIMA(p,d,q)-GARCH(r,s) orders, it was finally possible to construct the model and fit it to the data set. As can be seen in Figure 2.13, and in particular in the zoomed version of it, Figure 2.14, the model fits pretty well the data set. This demonstrates graphically that the previous described analysis and techniques were necessary in order to achieve good performances.

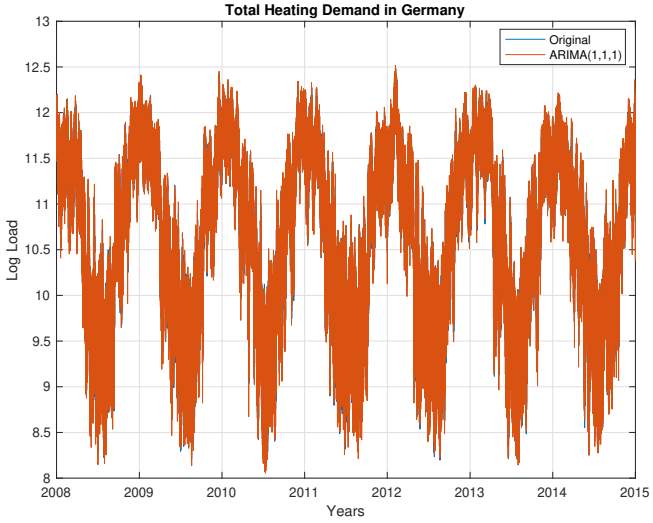

Figure 2.13: Real dataset and ARIMA-GARCH fitted model

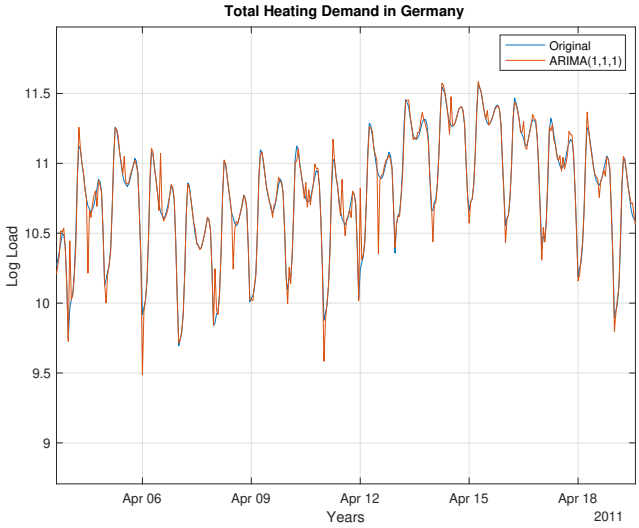

Figure 2.14: Zoom real dataset and ARIMA-GARCH fitted model

In particular, when fitting an ARIMA-GARCH model in Matlab, the software provides different properties of the model itself like the parameters adopted in the model equation. The following parameter values regarding the ARIMA-GARCH process under study were obtained:

ARIMA(1,1,1)	VALUE	STD.ERROR	t-statistic	PValue
Constant	-1.6001e-05	0.00066138	-0.024194	0.9807
AR(1)	0.39605	0.0050722	78.081	0
MA(1)	0.37814	0.0061805	61.183	0

Table 2.5: ARIMA parameters

GARCH(1,1)	VALUE	STD.ERROR	t-statistic	PValue
Constant	3.4596e-05	5.3e-06	6.5275	6.686e-11
GARCH(1)	0.97133	0.0013558	716.43	0
ARCH(1)	0.026932	0.0012318	21.863	0

Table 2.6: GARCH parameters

Considering the above parameters and the model mentioned in equation (2.16) and equation (2.18), it was possible to reconstruct analytically the adopted model:

$$X_t = -1.6001e-05 + 0.39605X_{t-1} + 0.37814\epsilon_{t-1} + \epsilon_t, \tag{2.31}$$

$$h_t = 3.4596e-05 + 0.97133h_{t-1} + 0.026932\epsilon_{t-1}^2. \tag{2.32}$$

Having fitted the model to the dataset, it was possible to perform forecast in differenced log domain and finally get the prediction values back in the original domain applying the inverse seasonal difference operator to the known historical values for the past d data point:

$$\hat{x}_{t+1} = H^{-1}\left(\hat{x}_{t+1}\right) = \left[-\phi_d, ..., -\phi_1, 1\right]^T \left[x_{t-d+1}, ..., x_t, \hat{x}_{t+1}\right], \tag{2.33}$$

where $\phi_1, .., \phi_d$ are as defined in Eq. (2.5).

After having fitted the model, predictions have been performed with and without bootstrap algorithms.
For the former case, the Matlab command *forecast* has been used, which forecast univariate autoregressive integrated moving average (ARIMA) model responses or conditional variances for performing prediction into the future, in a time range beyond the last instant of measured data.
For each one-hour-ahead prediction, the preceding $N_{train} = 30$ data points were used and a total of 100 total predictions were performed.
Without bootstrapping, the predictions gave the following results for what concerns the

performance metrics and, in particular, the root mean squared error (RMSE), mean absolute error (MAE), and the mean absolute percentage error (MAPE):

RMSE (MW)	MAE (MW)	MAPE (%)
9.0940e+03	5.6054e+03	0.0613

Table 2.7: Performance without bootstrap

$$\text{RMSE} = \sqrt{\frac{1}{N}\left(A_t - F_t\right)^2}, \tag{2.34}$$

$$\text{MAE} = \frac{1}{N}\sum_{t=1}^{N}\left|A_t - F_t\right|, \tag{2.35}$$

$$\text{MAPE} = \frac{100}{N}\sum_{t=1}^{N}\left|\frac{A_t - F_t}{A_t}\right|(\%), \tag{2.36}$$

where:

- A_t denotes the t-th actual value,
- F_t denotes the t-th forecasted values.

Basically:

- RMSE is a measure of the differences between values predicted by a model or an estimator and the values observed. Thus, RMSE is the average distance measured vertically from the actual value to the corresponding predicted value on the fit line. The range of RMSE is $(0, \infty)$; the smaller the RMSE value is, the higher the accuracy of the prediction model. The units of RMSE are the same as original units,
- MAE is an arithmetic average of the absolute errors between predictions and real vlaue. The MAE is often called the mean absolute deviation (MAD). The range of MAE is $(0, \infty)$; the smaller the MAE value is, the higher the accuracy of the prediction model. The unit of MAE iis the same as original data. In comparison with MAE, the RMSE has a relatively high weight for large errors, because the errors are squared before averaging,
- MAPE is a measure of prediction accuracy of a forecasting method. Is a percentage-dependent metric since is a unit-free measure which calculates the average of the percentage error. The disadvantage of the MAPE is that the MAPE is scale-sensitive; it will get extreme values if the actual value is quite small.

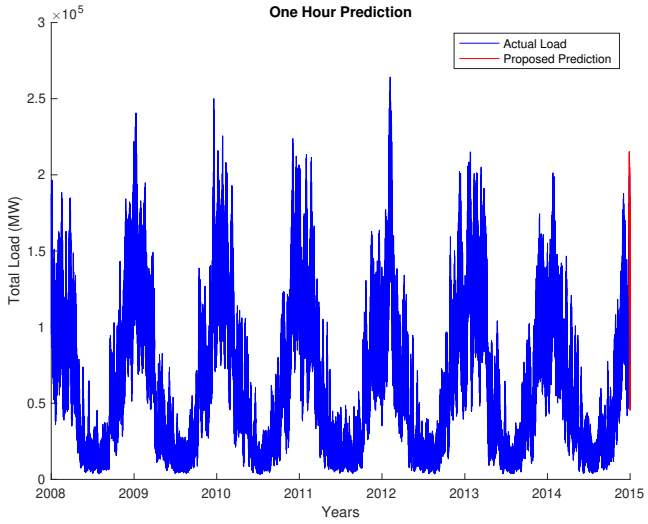

Figure 2.15: Prediction without bootstrapping

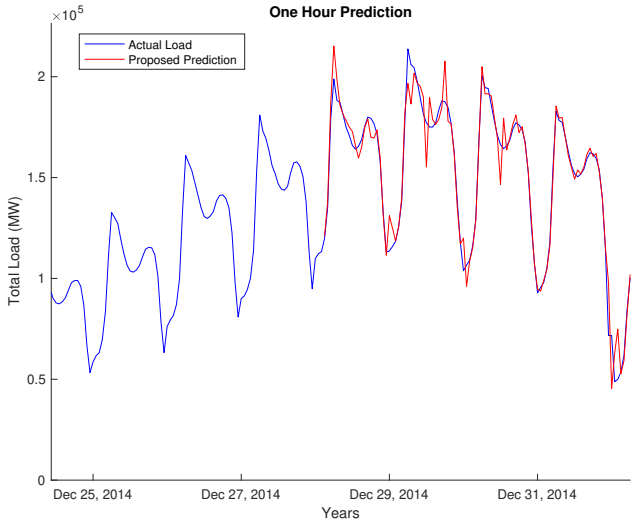

Figure 2.16: Zoom in prediction without bootstrapping

2.9 Introduction to bootstrap methods

Statistics is the science of learning from data. Statistical knowledge aids in the proper methods for collecting data, using correct methods for analyzing them, and effectively presenting the results derived from data. The primary task of a statistician is to summarize a sample based study and generalize the finding to the parent population in a scientific manner. A technical term for a sample summary number is (sample) statistic. Some basic sample statistics are sample mean, sample median, sample standard deviation.
Of course, a summary statistic like the sample mean will fluctuate from sample to sample and a statistician would like to know the magnitude of these fluctuations around the corresponding population parameter in an overall sense. This is then used in assessing margin of errors.
The entire picture of all possible values of a sample statistics presented in the form of a probability distribution is called a sampling distribution. A general intuitive method applicable to just about any kind of sample statistic that keeps the user away from the technical tedium has got its own special appeal. Bootstrap is such a method.
To understand bootstrap, suppose it were possible to draw repeated samples (of the same size) from the population of interest, many times. Then, one would get a fairly good idea about the sampling distribution of a particular statistic from the collection of its values arising from these repeated samples. But, that does not make sense as it would be too expensive and defeat the purpose of a sample study, gathering information cheaply in a timely fashion.

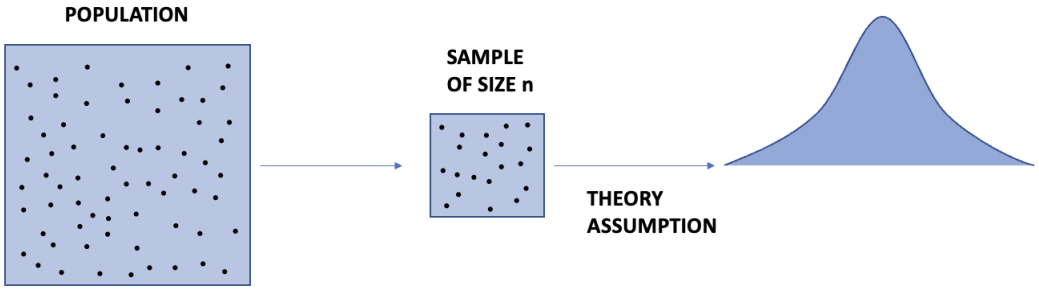

Figure 2.17: Traditional approach

The idea behind bootstrap is to use the data of a sample study at hand as a surrogate population, for the purpose of approximating the sampling distribution of a statistic; i.e. to resample (with replacement) from the sample data at hand and create numerous "phantom samples" known as bootstrap samples. The sample summary is then computed on each of the bootstrap samples. A histogram of the set of these computed values is referred to as the bootstrap distribution of the statistic.
So bootstrapping is a statistical procedure that resamples a single dataset to create many simulated samples, allowing for the calculation of standard errors, confidence intervals, and hypothesis testing.
Bootstrapping is any test or metric that uses random sampling with replacement and it estimates the properties of an estimator by measuring these properties when sampling from an approximating distribution.
The traditional approach draws one sample of size n from the population, and that sample is used to calculate population estimates to then make inferences on. Now, in reality, only one sample has been observed. However, there is the idea of a sampling distribution, which is a theoretical set of all possible estimates if the population were to be resampled.
The theory states that, under certain conditions such as large sample sizes, the sampling distribution will be approximately normal, and the standard deviation of the distribution will be equal to the standard error.
But what happens if the sample is not sufficiently large? Then, it cannot necessarily be assumed that the theoretical sampling distribution is normal. This then makes it difficult to determine the standard error of the estimate, and harder to draw reasonable conclusions from the data.
As with the traditional approach, a sample of size n is drawn from the population within the bootstrapping approach. Let call this sample S.
Then, rather than using theory to determine all possible estimates, the sampling distribution is created by resampling observations with replacement from S, m-times, with each resamples set having n observations. Now, if sample appropriately, S should be representative of the population. Therefore, by resampling S m-times with replacement, it would be as if m samples were drawn from the original population, and the estimates derived would be representative of the theoretical distribution under the traditional approach.
It must be noted that increasing the number of resamples, m, will not increase the amount of information in the data. That is, resampling the original set 100000 times is not more useful than only resampling it 1000 times. The amount of information within the set is dependent on the sample size, n, which will remain constant throughout each resample. The benefit of more resamples, then, is to derive a better estimate of the sampling distribution.

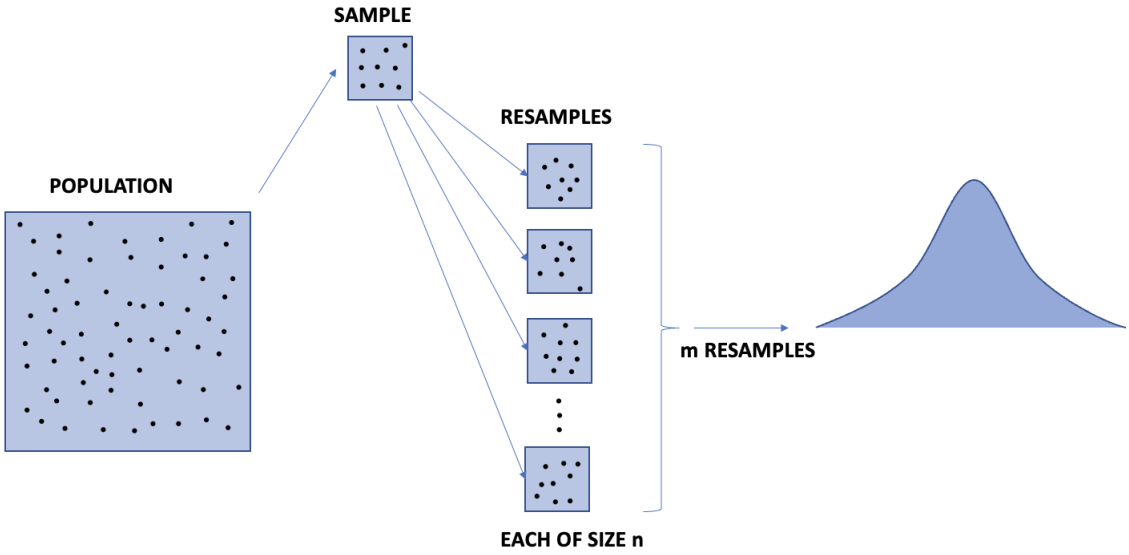

Figure 2.18: Bootstrap approach

Both traditional and bootstrapping approaches require the use of appropriately drawn samples to make inferences about populations. However, the most major difference between these two methods is the mechanics behind estimating the sampling distribution.
The traditional procedure requires one to have a test statistic that satisfies particular assumptions in order to achieve valid results, and this is largely dependent on the experimental design. The traditional approach also uses theory to tell what the sampling distribution should look like, but the results fall apart if the assumptions of the theory are not met.
The boostrapping method, on the other hand, takes the original sample data and then resamples it to create many (simulated) samples. This approach does not rely on the theory since the sampling distribution can simply be observed, and one does not have to worry about any assumptions. This technique allows for accurate estimates of statistics, which is crucial when using data to make decisions.

Bootstrap, as suggested by Zoubir and Iskandler [2007] is usually used for estimation of statistical characteristics such as bias, variance distribution functions, confidence intervals, and more in general hypothesis tests and model selection.
It is particularly useful when I know little about the statistics of the data or I have only a small amount of data so that I cannot use asymptotic results.

The bootstrap, overall, does with a computer what the experimenter would do in practice if it were possible:

1. The observations are randomly re assigned and the estimates re computed.

2. These assignments and re computations are done many times and treated as repeated experiments.

The bootstrap idea is to simulate the probability mechanism of the real world by substituting the unknowns with estimates derived from the data.

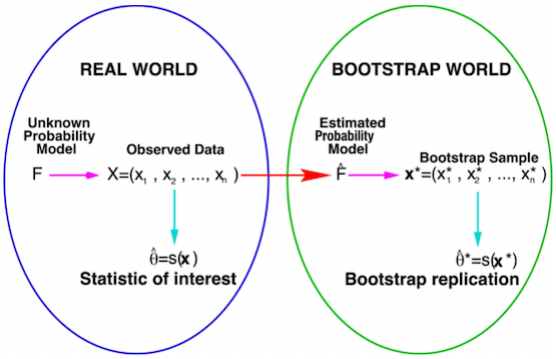

Figure 2.19: Bootstrap main idea

There are more ways of performing case resampling but the most common is the Monte Carlo algorithm. First, the data have to be resampled and the size of the resample must be equal to the size of the original data set.
Then the statistics of interest is computed from the resample from the first step. This routine is repeated many times to get a more precise estimate of the bootstrap distribution of the statistic.
Suppose that there is a series of measurements collected in $x = \{x_1, x_2, ..., x_n\}$ which are realizations of the random sample $X = \{X_1, X_2, ..., X_n\}$ drawn from some unspecified distribution F_X.
Let $\hat{\theta} = \hat{\theta}(X)$ be an estimator of some parameter θ of F_X, which could be, for example, the mean $\theta = \mu_X$ of F_X estimated by the sample mean $\hat{\theta} = \hat{\mu}_X = \frac{1}{n}\sum_{i=1}^{n} X_i$.
The aim is to find characteristics of $\hat{\theta}$ such as the distribution of $\hat{\theta}$.
If F is unknown and asymptotic theory does not apply, the bootstrap provides the answer.
Its paradigm suggests substitution of the unknown distribution F_X by the empirical distribution of the data, $\hat{F}_X$. Practically, it means that the original data is reused through resampling to create what is called a bootstrap sample which has the same size as the original one: $x_b^* = \{x_1^*, x_2^*, ..., x_n^*\}$ for $b = 1,2, ..., B$, where $x_i^*, i = 1,2, ...n$ are obtained, for example, by drawing at random with replacement from x.

1. Conduct the experiment to obtain $X = \{X_1, X_2, ..., X_n\}$ and find the estimator $\hat{\theta}$ from X.

2. Construct the empirical distribution $\hat{F}_\theta$ which puts equal mass 1/n to each observation $X_1 = x_1, X_2 = x_2, ..., X_n = x_n$.

3. From $\hat{F}_\theta$ draw a sample $X^* = X_1^*, X_2^*, ..., X_n^*$, called the bootstrap (re)sample.

4. Approximate the distribution of $\hat{\theta}$ by that of $\hat{\theta}^*$ derived from X^*.

With a large number B of bootstrap parameter estimates, it is then possible to approximate the distribution of $\hat{\theta}$ by that of $\hat{\theta}^*$, derived from the bootstrap sample x^*. So, the distribution of $F_{\hat{\theta}}$ of $\hat{\theta}$ is approximated by $\hat{F}_{\hat{\theta}^*}$, the distribution of $\hat{\theta}^*$.

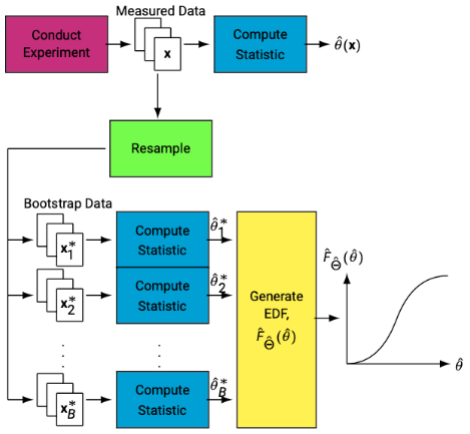

Figure 2.20: Bootstrap algorithm for distribution approximation

Furthermore, there are different types of bootstrap schemes and, in the following, will be discussed the one based on resampling residuals, a common approach in regression analysis.
The method proceeds as follows:

1. Fit the model and retain the fitted values $\hat{y}_i$ and the residuals $\hat{\epsilon}_i = y_i - \hat{y}_i, (i = 1, ..., n)$.

2. For each pair (x_i, y_i), in which x_i is the explanatory variable, add a randomly resampled residual, $\hat{\epsilon}_i$, to the fitted value $\hat{y}_i$. In other words, create synthetic response variables $y_i^* = \hat{y}_i + \hat{\epsilon}_j$, where j is selected randomly from the list (1,...,n) for every i.

3. Refit the model using the fictitious response variables y_i^*, and retain the quantities of interest (often the parameters, $\hat{\mu}_i^*$, estimated from the synthetic y_i^*).

4. Repeat steps 2 and 3 many times.

This scheme has the advantage that it retains the information in the explanatory variables. In the following page, a full explanation of the classical residual bootstrap ARMA(p,q)-GARCH(r,s) is presented step by step.

2.10 Classical residual bootstrap ARMA(*p*,*q*)-GARCH(*r*,*s*) model

1. Obtain the QML estimator from the past N_{train} samples,

$$\hat{\theta} = (\hat{\phi}_0, ..., \hat{\phi}_p, \hat{\alpha}_0, ..., \hat{\alpha}_q, \hat{\omega}_0, ..., \hat{\omega}_s, \hat{\gamma}_0, ..., \hat{\gamma}_r), \tag{2.37}$$

and calculate the residuals:

$$\hat{\epsilon}_\tau = X_\tau - \hat{\phi}_0 - \sum_{i=1}^{p} \hat{\phi}_i X_{\tau-i} - \sum_{j=1}^{q} \hat{\alpha}_j \hat{\epsilon}_{\tau-j}, \quad \tau = t - N_{train} + 1, ..., t. \tag{2.38}$$

2. Compute the variance:

$$\hat{h}_\tau = \hat{\omega}_0 + \sum_{j=1}^{s} \hat{\omega}_j \hat{\epsilon}^2_{\tau-1} + \sum_{i=1}^{r} \hat{\gamma}_i \hat{h}_{\tau-1}, \quad \tau = t - N_{train} + 1, ..., t. \tag{2.39}$$

3. Compute the standardized residuals:

$$\tilde{\epsilon}_\tau = \frac{\hat{\epsilon}_\tau}{\sqrt{\hat{h}_\tau}}, \quad \tau = t - N_{train} + 1, ..., t. \tag{2.40}$$

4. Resample $\tilde{\epsilon}_\tau$ with replacement to obtain $\tilde{\epsilon}^*_\tau, \quad \tau = t - N_{train} + 1, ..., t.$
With $\hat{\theta}$, x_t, $\hat{h_\tau}$, and $\tilde{\epsilon}^*_\tau$ as the disturbance path, estimate the predicted value at time t+1, $\tilde{x}$, according to:

$$X_t = \phi_0 + \epsilon_t + \sum_{j=1}^{p} \phi_j X_{t-j} + \sum_{j=1}^{q} \alpha_j \epsilon_{t-j}. \tag{2.41}$$

5. Repeat Step 4 B times to obtain the bootstrapped predictions, $\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_B$ and sort these into $\tilde{x}_{(1)} \leq \tilde{x}_{(2)}, ... \leq \tilde{x}_{(B)}$.

6. Obtain the 100(1-α) % confidence interval from $\tilde{x}_{(1)} \leq \tilde{x}_{(2)}, ... \leq \tilde{x}_{(B)}$ and the bootstrapped estimated value from Median ($\tilde{x}_{(1)} \leq \tilde{x}_{(2)}, ... \leq \tilde{x}_{(B)}$).

In the same way as before, for each one-hour-ahead prediction, the preceding $N_{train} = 30$ data points are used, 100 predictions are obtained and the number of bootstrap resamples is $B = 100$.

RMSE (MW)	MAE (MW)	MAPE (%)
8.9156e+03	5.3232e+03	0.0422

Table 2.8: Performance with residual bootstrap

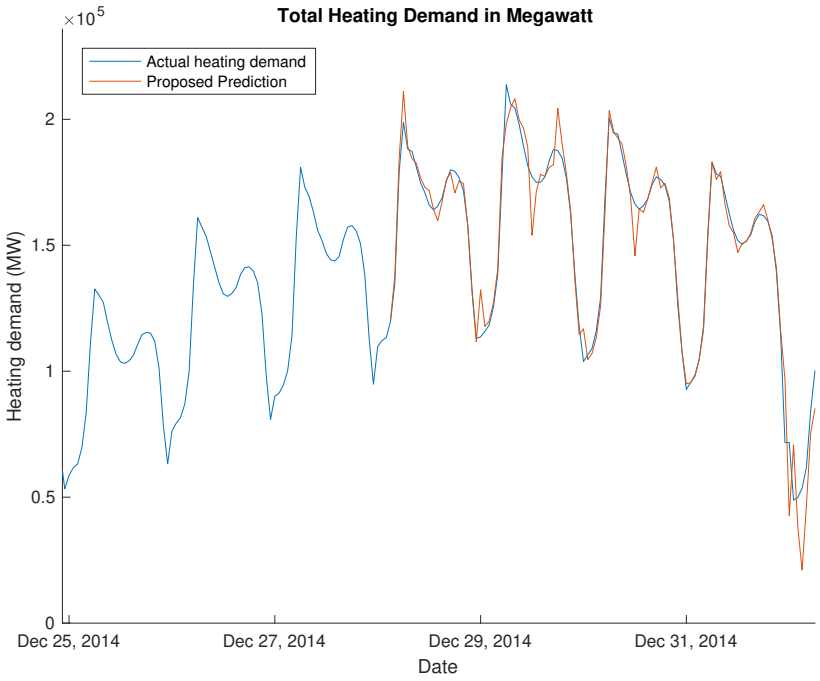

Figure 2.21: Residual bootstrap predictions

Chapter 3

Proposed approach

3.1 Outliers introduction

Outliers in time series data are values that significantly differ from the patterns and trends of the other values in the time series. Identifying outliers in time series analysis is important because outliers influence the forecast model that is used to forecast future values. Even a small number of outliers, if not handled correctly, can in fact drastically reduce the accuracy and reliability of the forecasts.
In ARIMA models, it has been found that outliers can result in significantly negatively biased estimates of the AR(1) coefficient and positively biased estimates of the MA(1) coefficient, and may also result in model misspecifications. Similarly, for the GARCH model, the presence of outliers may lead to problems in estimation, specification, forecasting and interpretation.
In ARIMA models, the accuracy of the point forecasts immediately following an outlier has been shown to be severely impaired. However, point forecasts are significantly less affected by an outlier when such observations are more than two periods away from the forecast origin. This because the effects of past observations on subsequent forecasts diminish exponentially with the distance from the forecast origin. Moreover, the impacts of outliers on the parameter estimates affects the forecasts substantially less.
A common and simple approach for detecting a single outlier in a linear regression model is selecting the observation with the largest absolute studentised residual, or the observation with the largest impact on the residual sum of squares upon deletion. However, this method is not appropriate for detecting outliers in time series, since the observations are usually neither normally nor independently distributed.
Classical time series analysis entertains two types of outliers: additive and innovation outliers.
An additive outlier only affects a single observation such as a typo or a recording error while, an innovation outlier affects many observations, and it often signifies an external disturbance to the series that has a gradually decaying impact.

Considering the case of GARCH processes, there are two different scenarios. First, there exist situations in which an outlier does not have any impact on volatility and this type of disturbance is called level outlier (LO).
The second scenario is when external disturbances, known as volatility outlier (VO), affect the volatility of the series of interest.
In the following some theory concerning outlier detection is shown.

3.1.1 Test to detect level outliers

When the outlier is in the level, it only affects the series at the observation where it occurs. The effect is not carried into the subsequent observations or into the volatility.
Let's consider the following GARCH(r,s) model free of outliers:

$$y_t = \sqrt{h_t}\epsilon_t, \tag{3.1}$$

$$h_t = \delta + \sum_{i=1}^{s} \alpha_i y_{t-i}^2 + \sum_{j=1}^{r} \beta_j h_{t-j}, \tag{3.2}$$

where:

- the sequence of errors $\{\epsilon_t\}$ is indipendent and identically distributed with zero mean and unit variance,
- $\delta > 0$, $\alpha_i \geq 0$, and $0 \leq \beta_i < 1$, for $i = 1, ..., s$, and $j = 1, ..., r$,
- the quantity h_t is the conditional variance of y_t on the past information, $h_t = Var\,(y_t|F_{t-1})$, where $F_{t-1} = \{y_{t-1}, y_{t-2}, ...\}$.
- the process is the same as the one described in equation

In financial applications, y_t are the returns and $\sqrt{h_t}$ is the volatility. In the following, equation (3.1) and equation (3.2) are referred to as the mean and volatility equation, respectively.
Considering now a GARCH(1,1) model and, calling the observed series z_t, the level outlier model for a single outlier of size γ at the τ-th observation is given by:

$$z_t = y_t + \gamma I_{[t=\tau]}, \tag{3.3}$$

$$h_t = \delta + \alpha_1 y_{t-1}^2 + \beta_1 h_{t-1}, \tag{3.4}$$

where:

- $y_t = \gamma\sqrt{h_t}\epsilon_t$, as defined in equation (3.1),
- $I_{[t=\tau]}$ is the indicator function, which is equal to one when $t = \tau$, and zero otherwise.

When ϵ_t has standard Gaussian distribution, the log-likelihood, conditional on z_1 and h_1, is given by:

$$\begin{aligned} L_{lo}\,(\gamma;\Theta) = -\frac{1}{2}\sum_{i=2}^{n} log\left[\delta + \alpha_1\left(z_{t-1} - \gamma I_{[t-1=\tau]}\right)^2 + \beta_1 h_{t-1}\right] - \\ -\frac{1}{2}\sum_{i=2}^{n}\frac{\left(z_t - \gamma I_{[t-1=\tau]}\right)^2}{\delta + \alpha_1\left(z_{t-1} - \gamma I_{[t-1=\tau]}\right)^2 + \beta_1 h_{t-1}}, \end{aligned} \tag{3.5}$$

where $\Theta = (\delta, \alpha_1, \beta_1)^T$.

Because the likelihood depends only on the absolute value of $\left(z_\tau - \gamma I_{[t=\tau]}\right)$, outliers equal to γ and $(2z_\tau - \gamma)$ at the τ-th observation will produce the same likelihood. In particular, considering no outlier and outliers equal to $2z_t$, at every observation will produce the same likelihood.
A solution is to consider outliers of smaller absolute values, to take the outlier with the same sign as y_τ, or equivalently to assume the magnitude of γ is smaller than the magnitude of the observed value y_τ.
A possible way for detecting additive outliers in a time series is the following.

$$LM_\tau^{lo} = LM_\tau^{vo} \left\{ 1 + \hat{\alpha}_1 \hat{h}_\tau \sum_{j=1}^{n-\tau} \hat{\beta}_1^{j-1} \left(\frac{1}{\hat{h}_{\tau+j}} - \frac{z_{\tau+j}^2}{\hat{h}_{\tau+j}^2} \right) \right\}^2 \left(1 + 2\hat{\alpha}_1^2 \hat{h}_\tau^2 \sum_{j=1}^{n-\tau} \frac{\hat{\beta}_1^{2j-2}}{\hat{h}_{\tau+j}^2} \right)^{-1} \tag{3.6}$$

where:

- hat symbol means estimates under the null hypothesis that there is no outlier,
- LM_τ^{vo}, is the test statistic for volatility outlier given by equation (3.10).

Because $\mid \beta_1 \mid < 1$, β_1^i converges to zero and the summations in equation (3.6) can be truncated.
In many applications the position of the outlier is not known and, one case use the test statistics:

$$LM_{max}^{lo} = max_{2 \leq \tau \leq n} LM_\tau^{lo}. \tag{3.7}$$

The distribution of the test statistics can be found using simulation, using the GARCH(1,1) model without outlier fitted to the observed data series for data generating model.

3.1.2 Test to detect volatility outliers

The effect of a volatility outlier is carried into the future observations via the volatility equation. The observation equation is the same as before, where γ now is the volatility outlier and the volatility equation is:

$$h_t = \delta + \alpha_1 z_{t-1}^2 + \beta_1 h_{t-1}. \tag{3.8}$$

For Gaussian distribution, the log-likelihood, conditional to z_1 and h_1, is given by:

$$L_{lo}(\gamma, \Theta) = -\frac{1}{2}\sum_{i=2}^{n} \log\left(\delta + \alpha_1 z_{t-1}^2 + \beta_1 h_{t-1}\right) - \frac{1}{2}\sum_{i=2}^{n} \frac{(z_t - \gamma I_{t=\tau})^2}{\delta + \alpha_1 z_{t-1}^2 + \beta_1 h_{t-1}}, \tag{3.9}$$

and there is no identification problem.
Hotta and Tsay [2018] proposed a Lagrange multiplier test for detecting volatility outliers. A Lagrange multiplier test is a general principle for testing hypotheses about parameters in a likelihood framework as suggested by Arellano [2002].
The test for the presence of a volatility outlier at the τ-th observation, in particular, is given by the square of the standardized return on observation τ:

$$LM_{\tau}^{vo} = \frac{y_{\tau}^2}{\hat{\alpha} + \hat{\alpha}_1 z_{\tau-1}^2 + \hat{\beta}_1 \hat{h}_{\tau-1}}, \tag{3.10}$$

where the hat means the maximum likelihood estimates under the null hypothesis but, as in the level outlier test presented before, the estimates can be replaced by any robust estimates.
When the position of the outlier is unknown, the test statistic $LM_{max}^{vo} = max_{2\leq\tau\leq n} LM_{\tau}^{vo}$ is employed to detect a volatility outlier.
When the parameters and h_1 are known LM_{τ}^{vo} , $\tau = 2, ..., n$, are independent with χ_1^2 distribution.
Based on this fact, Hotta and Tsay [2018] suggested approximating the distribution of LM_{max}^{vo} by the distribution of the maximum of an independent identically distributed (i.i.d) sample from the χ_1^2 distribution.

3.1.3 Test to detect level and volatility outliers simultaneously

In some procedures, tests can be used to detect the level and volatility outliers simultaneously. In general, these first involve testing whether there is any type of outlier and then deciding the type of outlier.
Doornik and Ooms [2005] presented a likelihood ratio test for detecting multiple level and volatility outliers.
They presented the test for an ARMA-GARCH model but, for simplicity, in the following is assumed that the return series was already filtered by a suitable model, which could be an ARMA model.
The GARCH(1,1) model with outliers is fitted to the residual series z_t. The model for possible outliers at the τ-th observation, for the GARCH(1,1) model, is given by:

$$z_t = y_t + \gamma_l I_{[t=\tau]}, \tag{3.11}$$

$$h_t = \delta + \alpha_1 y_{t-1}^2 + \beta_1 h_{t-1} + \gamma_v I_{[t-1=\tau]}, \tag{3.12}$$

where $y_t = \sqrt{h_t}\epsilon_t$, as defined in equation (3.1).

Under the null hypothesis of no outliers, $\gamma_l = \gamma_v = 0$ and under the hypothesis of existence of only a level outlier, $\gamma_l \neq 0$ and $\gamma_v = 0$.
Comparing with the volatility outlier model given by equation (3.3) and equation (3.8) with the model given by equation (3.11) and equation (3.12), there is a volatility outlier, as defined in Section 3.1.2, when:

$$\gamma_l = \gamma, \tag{3.13}$$

$$h_{\tau+1} = \delta + (\alpha_1 y_\tau + \gamma)^2 + \beta_1 h_\tau = \delta + \alpha_1 y_\tau^2 + \beta_1 h_\tau + \gamma_v, \tag{3.14}$$

where:

- $\gamma_l = \gamma$,
- $\gamma_v = 2\alpha_1 y_\tau \gamma + \alpha_1 \gamma^2$.

The parameter γ_v is different from zero and since the interest is more in outliers that increase the volatility, $\alpha_1 > 0$. Also, considering that γ has the same sign as γ_τ, in the presence of volatility outliers should be $\gamma_v > 0$.
The procedure suggested by Doornik and Ooms [2005] for detecting multiple outliers when the position of the outlier is not known is given in four steps:

- **Step 1:** estimate the GARCH model without outlier to obtain the log-likelihood $\hat{l}_0$ and the conditional variance estimates $\hat{h}_t$.

– **Step 2:** denote by "s" the position of the outlier which has the largest absolute standardized value, the one which maximizes $\mid z_t/\hat{h}_t \mid$. Estimate the model given by equation (3.11) and equation (3.12) with $\tau = s$.
Call the estimates of the outlier parameters $\hat{\gamma}_{l,s}$ and $\hat{\gamma}_{v,s}$ and the corresponding log-likelihood $\hat{L}_s$.

– **Step 3:** if $2\left(\hat{L}_0 - \hat{L}_s\right) < C_n^\alpha$, no outlier is detected at the level of significance α and the procedure stops. Doornik and Ooms [2005] suggested critical values based on extreme value theory.

– **Step 4:** if an outlier is detected, this step selects the type of outlier:

(a): if $\hat{\gamma}_v > 0$, then there is a level outlier; otherwise continue with steps 4(b).

(b): consider $\tau = s$ and $\gamma = \hat{\gamma}_{l,s}$, in the volatility model given by equation (3.3) and equation (3.4) and the associated log-likelihood given by equation (3.5). Find the maximum likelihood estimate and denote the associated log-likelihood as $\hat{L}_{ao,s}$.

(c): consider $\tau = s$ and $\gamma = \hat{\gamma}_{v,s}$, in the volatility model given by equation (3.3) and equation (3.8) and the associated log-likelihood given by equation (3.9). Find the maximum likelihood estimate and denote the associated log-likelihood as $\hat{L}_{vo,s}$.

(d): if $\hat{L}_{ao,s} > \hat{L}_{vo,s}$, the outlier is a level outlier otherwise it's a volatility outlier.

Each time an outlier is detected, its presence is incorporated in the model and the four-step procedure is applied until no outlier is detected.

3.2 Robust bootstrap idea

Robust statistical signal processing, as suggested by Zoubir et al. [2018], is part of statistical signal processing that, broadly, involves making inference based on observations of signals that have been distorted or corrupted in some unknown manner.
Classical statistical signal processing relies strongly on the normal (Gaussian) distribution, which provides, in many situations, a reasonable model for the data. It also allows for closed form derivations of optimal procedures. However, there have been deviations from Gaussianity reported in numerous campaigns.
Robust statistical methods, instead, account for the fact that the postulated models for the data are fullfilled only approximately and not exactly. In contrast to classic parametric procedures, robust methods are not significantly affected by small changes in the data, such as outliers or small model departures. They also provide near-optimal performance when the assumptions hold exactly.
While optimality is clearly desirable, robustness is the engineer's choice.
Optimality, only under the assumed (nominal) distribution is in fact useless if the estimator is applied to data that does not follow this distribution.
Furthermore, even slight deviations from the assumed distribution may cause the estimator's performance to drastically degrade or to completely break down. On the other hand, robust methods do not depend critically on the exact fulfillment of the model assumptions and are designed in such a way that they behave nearly optimally, if the assumed model is correct, while small deviations from the model assumptions degrade performance only slightly and larger deviations do not cause a catastrophe.
In recent years, a few of robust bootstrap methods have been developed like:

- Influence Function bootstrap: which assigns resampling probabilities based on the influence function in such a way that potential outliers are assigned small probabilities of appearing in any bootstrap resample. This method will be further analyzed, since it has been adopted for the case under study.

- Stratified bootstrap: which divides data points into groups and bootstrap each stratum individually in such a way that the fraction of contamination in every bootstrap sample is kept more representative, in terms of contamination level of the original sample.

- Fast and robust bootstrap: which avoids recomputation of fixed-point equations for each bootstrap sample via a smart approximation.

So, when dealing with outliers or, in general, contaminated time series, two are the possible approaches:

– follow the idea analyzed in Section 3.1 of modeling and correcting the outliers before applying forecasting techniques. The forecasting technique will be applied to the clean data set.

– develop a robust bootstrap algorithm able to, even in presence of outliers, reproduce the time series correctly without need of adjusting for outliers before. Predictions will be based on the robust bootstrap algorithm which, internally, will provide for the manipulation of the outliers. This has been the choice for this study case since the IF bootstrap algorithm has been implemented and used for making future predictions on the heating demand in Germany.

3.3 Influence Function bootstrap

The existence of outliers in a sample is an obvious problem which can become even worse when the usual bootstrap is applied, because some resamples may have higher contamination level than the initial sample.
It is therefore important to develop bootstrap procedures that prevent some or all of these problems.
Amado and Pires [2004] suggested a modification of the bootstrap procedure in order to solve these problems which consists of forming each bootstrap sample by resampling with different probabilities so that the potentially more harmful observations have smaller probabilities of selection.
The aim is to protect the bootstrap procedure against a given number of arbitrary outliers and at the same time to be computationally less time-consuming than resampling using a robust estimator.
The control mechanism consists of an alteration of the resampling probabilities assigning more importance to some samples than others and using the Influence Function to compute those selection probabilities.
In general, this procedure leads to resampling less frequently highly influent observations (those with smaller robust empirical standardized influence function) while, at the same time, resampling with equal probabilities the observations belonging to the main structure.
Before going deeper inside the implementation of the IF bootstrap, let's define what an Influence Function is. As suggested by Zoubir et al. [2018], to understand the IF, is better to introduce before the so called Tukey's sensitivity curve or empirical IF:

$$\mathbf{SC}\left(\mathbf{y}, \hat{\beta}_{\mathbf{N}}\right) = \mathbf{N} \cdot \left(\hat{\beta}_{\mathbf{N}}\left(\mathbf{y_1}, \mathbf{y_2}, ..., \mathbf{y_{N-1}}, \mathbf{y}\right) - \hat{\beta}_{\mathbf{N-1}}\left(\mathbf{y_1}, \mathbf{y_2}, ..., \mathbf{y_{N-1}}\right)\right). \tag{3.15}$$

The sensitivity curve displays the bias of an estimator $\hat{\beta}_N$ when an additional observation, that takes on the value y, is added to a sample $y_{N-1} = (y_1, y_2, ..., y_{N-1})^T$.
Here $\hat{\beta}_N(\cdot)$ represents an estimator of a deterministic parameter β based on a sample of size N.
Having defined the sensitivity curve, it'is possible to consider the Influence Function of an estimator $\hat{\beta}$ as its counterpart, as N tends to infinity.
Let $\hat{\beta}_\infty$ denote the estimator as N tends to infinity and let T(F) be its functional representation. Then, the IF shows the approximate behavior of the asymptotic estimator when the sample contains a small fraction ϵ of identical outliers.
The IF is defined as follows:

$$\mathbf{IF}\left(\mathbf{y}; \mathbf{T}\left(\mathbf{F}\right), \mathbf{F}\right) = \lim_{\epsilon \downarrow \mathbf{0}} \frac{\mathbf{T}\left(\mathbf{F}_\epsilon\right) - \mathbf{T}\left(\mathbf{F}\right)}{\epsilon} = \left[\frac{\partial \mathbf{T}\left(\mathbf{F}_\epsilon\right)}{\partial \epsilon}\right]_{\epsilon=\mathbf{0}}, \tag{3.16}$$

where $T(F)$ and $T(F_\epsilon)$ are the functional representations of the estimator when the data is distributed following, respectively, F and the contaminated distribution:

$$F_\epsilon = (1 - \epsilon) F + \epsilon \delta_y, \tag{3.17}$$

with δ_y being the point-mass probability on y and ϵ the fraction of contamination. The main argument of the IF is y, the position of the infinitesimal contamination. The IF is the first derivative of an estimator T(F) evaluated based on the underlying distribution function F while the parameter y is a coordinate in the space of probability distributions. Consistent with equation (3.19), the assumption is made that the true distribution lies in an ϵ-neighborhood of the nominal distribution F. By evaluating the influence at F, the goal is to gain insight into the behavior of T(F) within the entire neighborhood.
Having defined the sensitivity curve and the Influence Function, let's start now describing the influence function bootstrap.
Let $F(\Omega)$ be some specified nominal parametric model, and let the actual distribution of the data belong to a contamination neighborhood. Here, $\Omega = (\beta, \tau)$, where $\beta \in R^p$ is the parameter vector of interest, while $\tau \in R^p$ are the nuisance parameters. Then, the Standardized Influence Function (SIF) of some non-robust estimator $\hat{\beta}^{nr}$, e.g. the LSE, is defined as:

$$\mathbf{SIF}\left(\mathbf{x}, \hat{\beta}^{\mathbf{nr}}, \mathbf{F}\left(\mathbf{\Omega}\right)\right) = \left(\mathbf{IF}\left(\mathbf{x}, \hat{\beta}^{\mathbf{nr}}, \mathbf{F}\left(\mathbf{\Omega}\right)\right)^{\mathbf{T}} \mathbf{var}^{-\mathbf{1}}\left(\hat{\beta}_{\infty}^{\mathbf{nr}}\right) \mathbf{IF}\left(\mathbf{x}, \hat{\beta}^{\mathbf{nr}}, \mathbf{F}\left(\mathbf{\Omega}\right)\right)\right)^{\frac{\mathbf{1}}{\mathbf{2}}}, \tag{3.18}$$

with:

$$\mathbf{var}^{-\mathbf{1}}\left(\hat{\beta}_{\text{inf}}\right) = \mathbf{E}\left[\mathbf{IF}\left(\mathbf{IF}(\mathbf{x}, \hat{\beta}, \mathbf{F}\left(\mathbf{\Omega}\right)\right) \mathbf{IF}\left(\mathbf{x}, \hat{\beta}, \mathbf{F}\left(\mathbf{\Omega}\right)\right)^{\mathbf{T}}\right], \tag{3.19}$$

denoting the asymptotic variance of an estimator, and where the expectation is taken with respect to the nominal distribution $F\left(\Omega\right)$. Then, the robust standardized empirical Influence Function $RESIF\left(x, \hat{\beta}^{nr}, \hat{\Omega}^{r}\right)$ is obtained by substituting robust estimates $\hat{\Omega}^{r}$ of Ω into equation (3.20).
For the simple linear regression model, $\mathbf{\Omega} = (\beta, \sigma)^{T}$, where σ^2 denotes the variance of the residuals, it is the case that:

$$r_i = y_i - x_i\beta, \quad i = 1, ..., N. \tag{3.20}$$

Moreover, the LSE is an M-estimator with $\rho\left(\frac{r_i}{\sigma}\right) = \left(\frac{r_i}{\sigma}\right)^2, \quad i = 1, ..., N$. Then, it follows that:

$$\mathbf{SIF}\left(r_i, \sigma\right) = \left(\frac{r_i^2}{\sigma^2}\right)^{\frac{1}{2}} = \frac{\mid r_i \mid}{\sigma}, \quad i = 1, ..., N. \tag{3.21}$$

The robust empirical SIF is then obtained by inserting robust estimates of β and σ, that is:

$$\mathbf{RESIF}\left(\hat{r_i}, \hat{\sigma}\right) = \frac{\mid y_i - x_i\hat{\beta} \mid}{\hat{\sigma}}, \quad i = 1, ..., N. \tag{3.22}$$

Now, the computational cost of the IFB is reasonable because robust estimation is performed only once prior to bootstrapping. Every bootstrapped estimate $\hat{\beta}^{*}$ is

obtained through the classical LSE.
The LSE is a solution to the zero gradient equation:

$$\nabla_\beta * L_{RSS}\left(\hat{\beta}\right) = 0 \Leftrightarrow \sum_{i=1}^{N}\left(y_i - x_{[i]}^T\hat{\beta}\right)x_{[i]}^* = 0, \tag{3.23}$$

which can be represented more compactly in matrix form as:

$$X^H\left(y - X\hat{\beta}\right) = 0 \Leftrightarrow X^H X\hat{\beta} = X^H y. \tag{3.24}$$

Assuming X is full rank, the LSE is the unique solution of the normal equation and the LSE can be expressed as a linear function of y as:

$$\hat{\beta} = \left(X^H X\right)^{-1} X^H y. \tag{3.25}$$

The LSE fit is thus:

$$\hat{y} = X\hat{\beta} = X\left(X^H X\right)^{-1} X^H y = Hy, \tag{3.26}$$

where:

$$H = X\left(X^H X\right)^{-1} X^H, \tag{3.27}$$

is called the hat matrix which is a projection matrix $\left(H^H = H \;\; and \;\; H^2 = H\right)$.

Going back to the IF bootstrap, let $X = \{(x_i, y_i), i = 1, ..., N\}$ be an available sample and let $\hat{\beta}$ and $\hat{\sigma}$ be the associated estimates of regression and scale, respectively.
The IF Bootstrap can be explained as follows:

1. Conduct experiment and collect measurements. From the original sample, compute an estimate, $\hat{\beta}(X)$.

2. Compute $RESIF(\hat{r}_i, \hat{\sigma})$ at each data point $i = 1,2, ..., N$ by evaluating equation (3.24). Then, compute the weights w_i according to:

$$w_i = \mathbb{1}_{[0,c]}\left(|\mathbf{RESIF}(\hat{r}_i, \hat{\sigma})|\right) + \phi\left(c, |\mathbf{RESIF}(\hat{r}_i, \hat{\sigma})|\right) \times \mathbb{1}_{(c,+\infty]}\left(|\mathbf{RESIF}(\hat{r}_i, \hat{\sigma})|\right), i = 1,2, ..., N. \tag{3.28}$$

   Here, $\mathbb{1}_A$ is the indicator function of the set A, c>0 is a tuning constant and ϕ is a non-negative function satisfying the two conditions:
   Condition 1:

$$\lim_{t\to\infty} t^2\phi(c,t) = 0 \quad (for\ fixed\ c) \tag{3.29}$$

   Condition 2:

$$\frac{\partial}{\partial t}\phi(c,t)\,|_{t=c} = 0 \tag{3.30}$$

3. Obtain bootstrap resamples by sampling with probabilities $\mathbf{p} = (p_1, p_2, ..., p_N)^T$ where:

$$p_i = \frac{w_i}{\sum_{n=1}^{N} w_n}. \tag{3.31}$$

The first condition on ϕ, protects the bootstrap distribution from the harmful effects of outliers, while the second condition preserves the efficiency of the procedure with clean data.
Moreover, a flexible family of functions from which ϕ can be chosen is the kernel of the pdf of the t-distribution and its limiting form, the normal distribution:

$$\phi(x; c, d, \gamma) = \begin{cases} \left(1 + \frac{(x-c)^2}{\gamma d^2}\right)^{-\frac{\gamma+1}{2}} & 0 < \gamma < \infty \\ exp\left(-\frac{(x-c)^2}{2d^2}\right) & \gamma = \infty \end{cases}$$

Here, c is the location parameter which is equal to the tuning constant in equation (3.28), d is the scale and γ is the shape parameter.
Condition 1 is satisfied if $\gamma > 1$ and condition 2 is satisfied if c is the tuning constant. In order to reduce the number of parameters to be adjusted to two (c and γ), the parameter d can further be set equal to c.

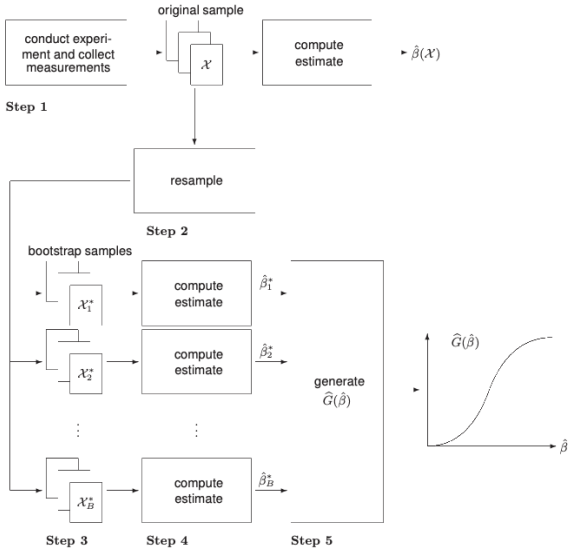

Figure 3.1: Implementation of the nonparametric bootstrap to estimate the distribution function $G\left(\hat{\beta}\right)$ of $\hat{\beta}$ which is an estimator for the unknown parameter β

1. Obtain the QML estimator from the past N_{train} samples,

$$\hat{\theta} = (\hat{\phi}_0, ..., \hat{\phi}_p, \hat{\alpha}_0, ..., \hat{\alpha}_q, \hat{\omega}_0, ..., \hat{\omega}_s, \hat{\gamma}_0, ..., \hat{\gamma}_r), \tag{3.32}$$

and calculate the residuals:

$$\hat{\epsilon}_\tau = X_\tau - \hat{\phi}_0 - \sum_{i=1}^{p} \hat{\phi}_i X_{\tau-i} - \sum_{j=1}^{q} \hat{\alpha}_j \hat{\epsilon}_{\tau-j}, \quad \tau = t - N_{train} + 1, ..., t. \tag{3.33}$$

2. Compute the variance:

$$\hat{h}_\tau = \hat{\omega}_0 + \sum_{j=1}^{s} \hat{\omega}_j \hat{\epsilon}^2_{\tau-1} + \sum_{i=1}^{r} \hat{\gamma}_i \hat{h}_{\tau-1}, \quad \tau = t - N_{train} + 1, ..., t. \tag{3.34}$$

3. Calculate the robust empirical Standardized Influence Function :

$$RESIF\left(r_i, \hat{\sigma}\right) = \frac{|y_i - x_i\hat{\beta}|}{\hat{\sigma}}, i = 1,2, ..., N, \tag{3.35}$$

being $r_i = |y_i - x_i\hat{\beta}|$ and $\hat{\sigma}$ a robust estimates of the standard deviation of the residuals.
Using the ARIMA-GARCH notation, the RESIF can even be rewritten considering $\hat{\epsilon}_\tau = r_i$ and $\hat{\sigma}$ being the standard deviation of the residuals obtained by the square root of the variance $\hat{h_\tau}$:

$$RESIF\left(r_i, \hat{\sigma}\right) = \frac{\hat{\epsilon}_\tau}{\hat{\sigma}}. \tag{3.36}$$

4. Compute the weights w_i according to:

$$w_i = \mathbb{1}_{[0,c]}\left(|\mathbf{RESIF}\left(\hat{r}_i, \hat{\sigma}\right)|\right) + \phi\left(c, |\mathbf{RESIF}\left(\hat{r}_i, \hat{\sigma}\right)|\right) \times \mathbb{1}_{(c,+\infty]}\left(|\mathbf{RESIF}\left(\hat{r}_i, \hat{\sigma}\right)|\right), i = 1,2, ..., N. \tag{3.37}$$

5. Obtain bootstrap resamples by resampling with replacement with probabilities $p = (p_1, p_2, ..., p_N)^T$ where:

$$p_i = \frac{w_i}{\sum_{n=1}^{N} w_n}. \tag{3.38}$$

6. Repeat step 5 B times to obtain the bootstrapped prediction $\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_B$ and sort these into $\tilde{x}_{(1)} \leq \tilde{x}_{(2)}, ... \leq \tilde{x}_{(B)}$.
7. Obtain the 100(1-α) % confidence interval from $\tilde{x}_{(1)} \leq \tilde{x}_{(2)}, ... \leq \tilde{x}_{(B)}$ and the bootstrapped estimated value from Median $(\tilde{x}_{(1)} \leq \tilde{x}_{(2)}, ... \leq \tilde{x}_{(B)})$.

3.4 Influence Function bootstrap applied to the clean time series

In the following, some of the results obtained are represented and discussed, still considering the same ARMA(1,1)-GARCH(1,1) model as before.
In particular, a comparison between the performances of applying Influence Function bootstrap and classical residual bootstrap (both implemented through the use of Matlab) have been studied in order to understand which method is more suitable for making predictions on future data values.
For the first case, moreover, different simulations have been performed using different values of tuning constant c and shape parameter γ to check which range of values suits better for making future predictions.
For computational time costs, the performances have been compared considering a case with 10 Monte Carlo simulations, 30 trained data taken from the dataset, 50 bootstrap resamples, and 100 total predictions.

C value	γ value	RMSE (MW)	MAE (MW)	MAPE (%)
0.1	1.1	8.5034e+03	4.8123e+03	0.0532
0.5	1.1	8.5123e+03	4.9234e+03	0.0535
1	1.1	8.5198e+03	4.9027e+03	0.0472
1.5	**1.1**	8.5013e+03	4.8928+03	**0.0461**
2	1.1	8.465e+03	4.8521e+03	0.0499
1.5	1.01	8.489e+03	4.8768e+03	0.0511
1.5	1.3	8.517e+03	4.8148e+03	0.0477
1.5	2.5	8.612e+03	4.7791e+03	0.0589
1.5	5	8.445e+03	4.7649e+03	0.3202
1.5	**10**	**8.432e+03**	**4.7566e+03**	0.0511
1.5	15	8.901e+03	4.8128e+03	0.0543

Table 3.1: Performance with different c and γ values

As can be observed in Table 3.1, the best performances have been obtained for the case of $c = 1.5$ and $\gamma = 10$. However, looking at the case of $c = 1.5$ and $\gamma = 1.1$, the results are not that different so, another comparison has been performed considering now 10 Monte Carlo simulations, 30 trained data, 100 bootstrap resamples and 100 data points to be predicted, to achieve bigger accuracy. The following results have been obtained:

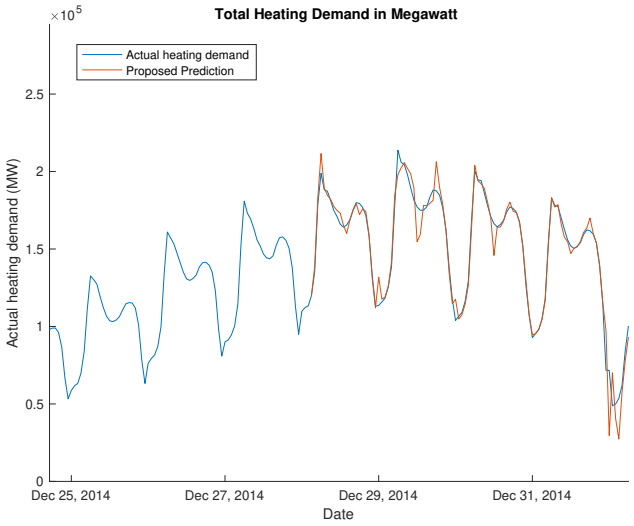

Figure 3.2: Influence Function bootstrap predictions with $\gamma = 1.1$

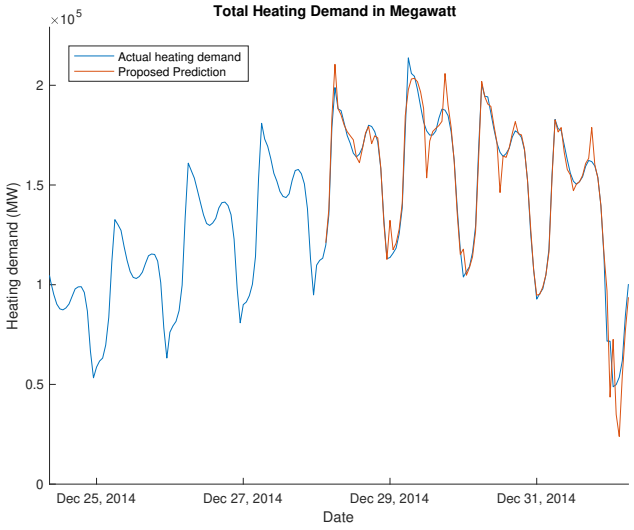

Figure 3.3: Influence Function bootstrap predictions with $\gamma = 10$

C value	γ value	RMSE (MW)	MAE (MW)	MAPE (%)
1.5	1.1	8.1273e+03	4.5631e+03	0.039
1.5	10	8.2671e+03	4.6712e+03	0.045

Table 3.2: Performance comparison when $\gamma = 1.1$ and $\gamma = 10$

Looking at the Table 3.2 above, it is possible to observe how the algorithm is able to achieve better performances when $\gamma = 1.1$ with a fixed value of c= 1.5. For this reason, in the following, the case of $\gamma = 1.1$ has been considered.
After having identified the most suitable values for what concern the tuning constant c and the shape parameter γ, a new simulation has been carried out taking into account the real dataset without modification in order to see the differences concerning the performances of classical residual bootstrap and Influence Function bootstrap (with c set to 1.5 and γ to 10).
For both cases, a number of 10 Monte-Carlo simulations has been performed considering a number of 100 predictions, using 30 trained data and 100 bootstrap resamples, from the original dataset.
As done previously, again, MAPE, MAE, and RMSE have been estimated:

Resampling method	RMSE (MW)	MAE (MW)	MAPE (%)
Residual Bootstrap	8.9156e+03	5.3232e+03	0.0422
IF Bootstrap	8.1273e+03	4.5631e+03	0.0390

Table 3.3: Performance comparison between residual and IF bootstrap

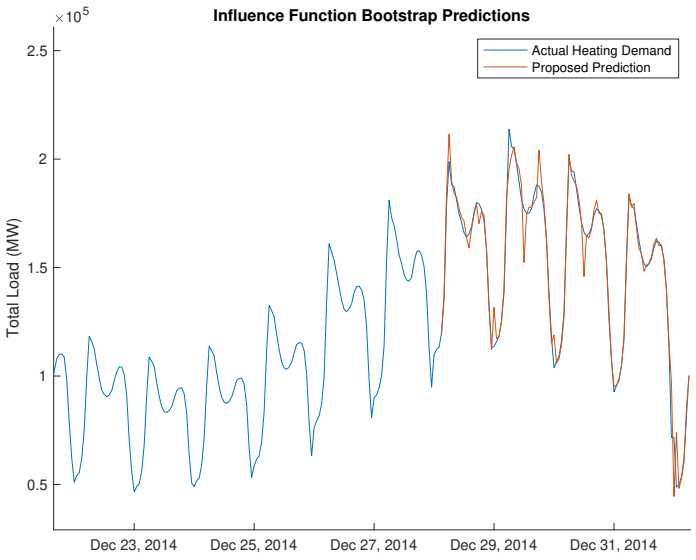

Figure 3.4: Predictions based on Influence Function bootstrap

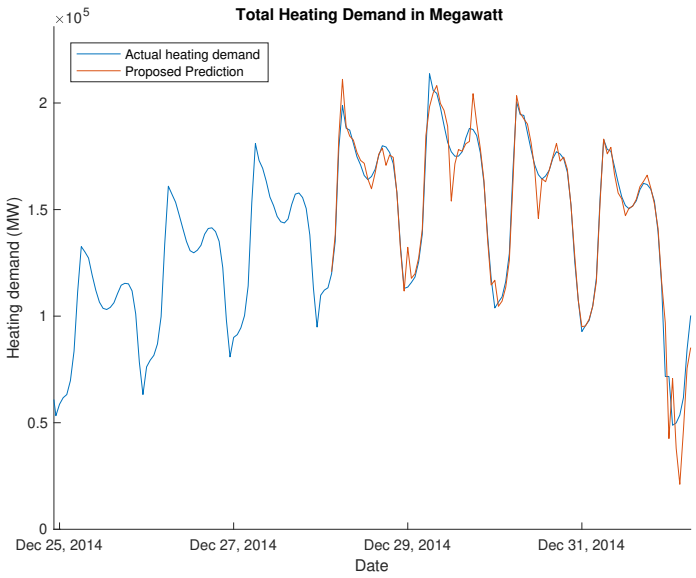

Figure 3.5: Residual bootstrap predictions

3.5 Influence Function bootstrap applied to contaminated time series

After having analyzed the real dataset and the prediction performances of both classical residual bootstrap and IF bootstrap, a new comparison has been carried out in the cases where some outliers are added to the original time series.
In this sense, three different cases have been considered:

- only level outliers present,
- only volatility outliers present,
- both level and volatility outliers present simultaneously.

3.5.1 Level outliers case

Following the model analyzed in equation (3.3), additive outliers have been included in time series at different time steps in order to check whether the IF bootstrap can be really considered a robust technique in comparison to the classical residual bootstrap. In particular the original dataset, collecting a total number of 61320 values, has been transformed by increasing the magnitude of some data points such that:

Indices of modified data	Original mean value	New mean value
5000:5010	1.1381e+04	4.6381e+04
20000:20100	6.9259e+04	1.0226e+05
45000:45700	1.2890e+05	1.6040e+05

Table 3.4: Level outliers introduction

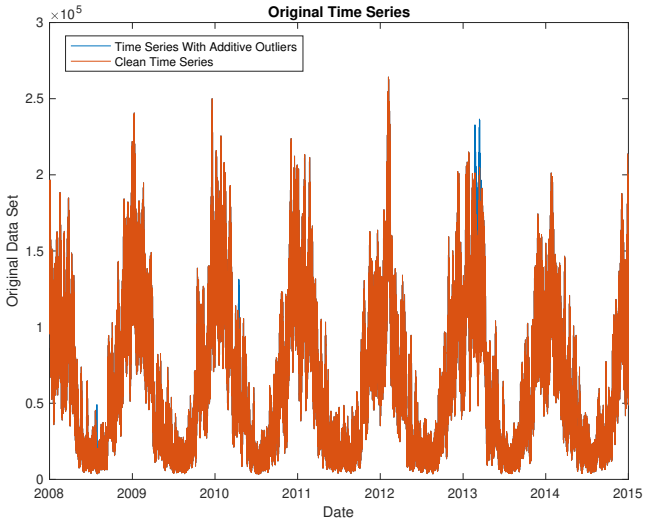

Figure 3.6: Example of additive outliers

For doing a comparison between the two methods, a number of 10 Monte-Carlo simulations has been performed considering a number of 100 predictions, using 30 trained data form the original dataset, and considering 100 bootstrap resamples. The following comparison of MAE, MAPE, and RMSE has been obtained:

Resampling method	RMSE (MW)	MAE (MW)	MAPE (%)
Residual Bootstrap	9.2355e+03	5.4599e+03	0.0512
IF Bootstrap	8.5532e+03	4.7891e+03	0.0434

Table 3.5: Performance with level outliers

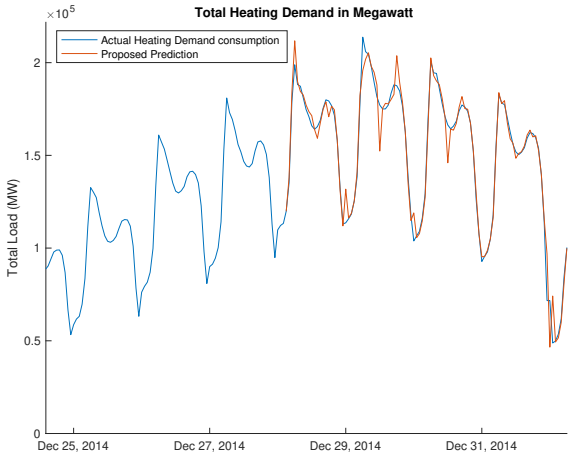

Figure 3.7: Influence Function bootstrap with additive outliers

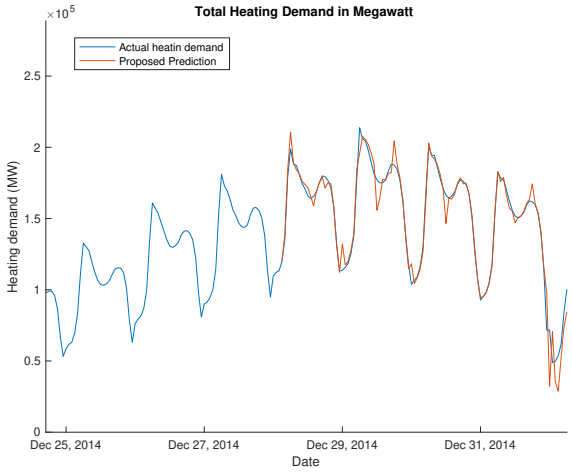

Figure 3.8: Classical residual bootstrap with additive outliers

As can be observed in Table 3.5, the IF bootstrap performances are better than those of residual classical bootstrap. Furthermore, looking at Figure 3.7 and Figure 3.8, it is possible to observe also graphically how the forecasted prediction of the Influence Function bootstrap are better following the ARIMA-GARCH model fitted to the data set.
By manually introducing manually some outliers, it has been possible to verify the correctness of the implemented algorithm of IF bootstrap.

3.5.2 Volatility outliers case

Considering the model discussed in equation (3.10), in the same way as in Section 3.5.1, a comparison between classical residual bootstrap and Influence Function bootstrap has been studied after having added innovation outliers in the clean time series.
In the same way as before, some volatility outliers have been introduced:

Indices of modified data	Original mean value	New mean value
5000:5010	0.0525	0.3526
20000:20100	0.0347	0.2447
45000:45700	0.0276	0.3576

Table 3.6: Level outliers introduction

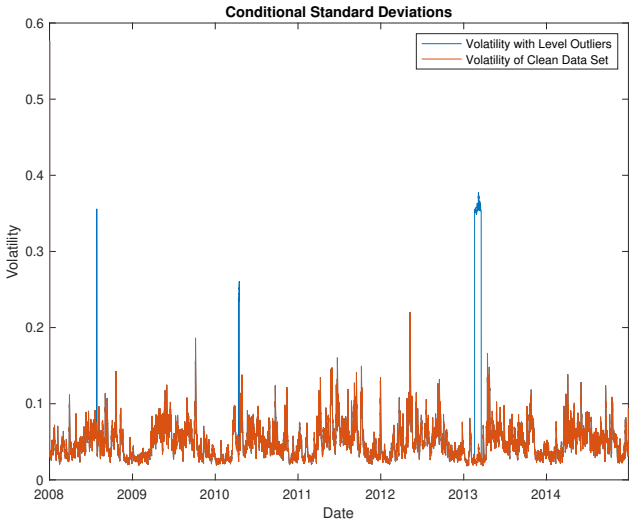

Figure 3.9: Example of innovative outliers

Once more, a number of 10 Monte-Carlo simulations has been performed considering a number of 100 predictions, using 30 trained data form the original dataset, and considering 100 bootstrap resamples.
The following results have been obtained:

Resampling method	RMSE (MW)	MAE (MW)	MAPE (%)
Residual Bootstrap	9.3100e+03	5.6788e+03	0.0525
IF Bootstrap	8.7113e+03	4.8197e+03	0.0477

Table 3.7: Performance with volatility outliers

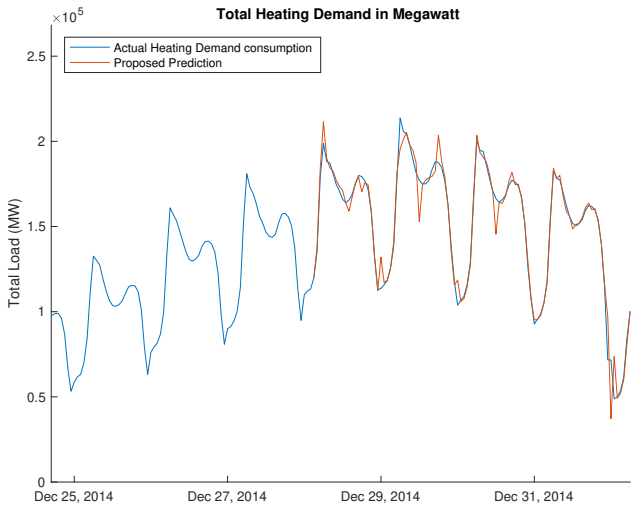

Figure 3.10: Influence Function bootstrap with volatility outliers

Concerning the case of volatility outliers, the situation is a bit different since the performances are, in general a bit worse. Graphically, at Figure 3.10 ad Figure 3.11 , it is possible to see that the predictions is working fine expect at the end for the case of classical residual bootstrap, demonstrating once more the improvements of IF bootstrap.
Moreover, as in the case of level outliers, also for volatility ones, it can be seen in the Table 3.7 how the IF bootstrap manages to achieve better performance than the classical one based on residuals.

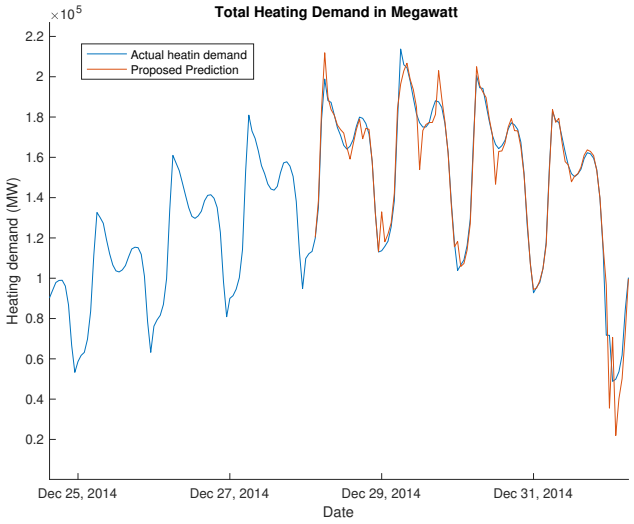

Figure 3.11: Classical residual bootstrap with volatility outliers

3.5.3 Level and volatility outliers simultaneously

The third case combines the previous two in order to check the performances when both additive and volatility outliers are present in the data set. The same additive and volatility outliers have been introduced.
In the same way as before, a number of 10 Monte-Carlo simulations has been performed considering a number of 100 predictions, using 30 trained data form the original dataset, and considering 100 bootstrap resamples.
A comparison of MAE, MAPE, and RMSE has been conducted to check whether Influence Function bootstrap is able to reach better results than classical residual bootstrap also in this case.

Resampling method	RMSE (MW)	MAE (MW)	MAPE (%)
Residual Bootstrap	9.5623e+03	5.8391e+03	0.0612
IF Bootstrap	8.9045e+03	5.1161e+03	0.0568

Table 3.8: Performance with level and volatility outliers

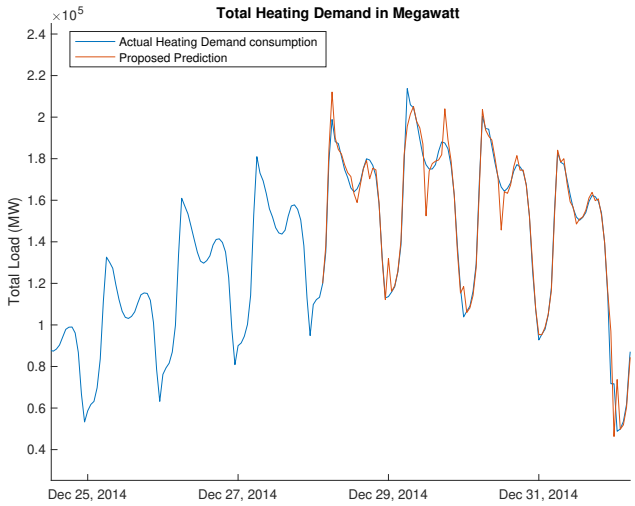

Figure 3.12: Influence Function bootstrap with additive and volatility outliers

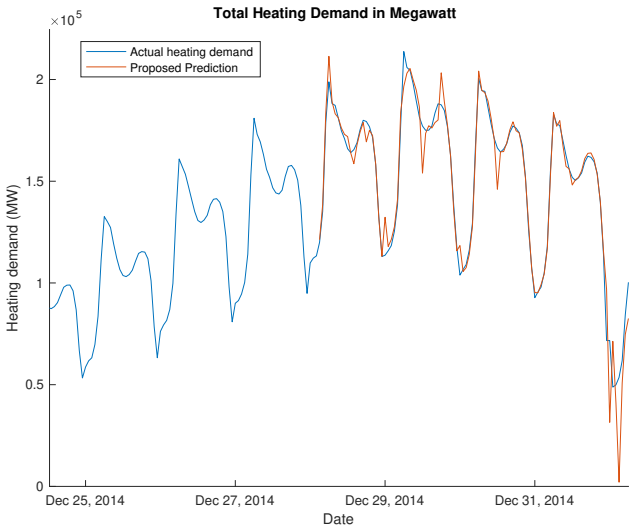

Figure 3.13: Classical residual bootstrap with additive and volatility outliers

The same situation as before arised also when considering at the same time both level and volatility outliers. Of course, in this scenario, the performances are a bit worse but still showing an improvement when considering the Influence Function bootstrap with respect to the classical residual bootstrap.

3.6 Summary of results

In order to have a total view of the results obtained previously, in the various cases, a summary table is shown below. All the simulations below were performed considering $\gamma = 10$, 30 trained data from the original dataset, 10 Monte-Carlo simulations, 100 bootstrap resamples and a total of 100 predictions.

Data Set	Resampling method	RMSE (MW)	MAE (MW)	MAPE (%)
Original	Residual Bootstrap	8.9156e+03	5.3232e+03	0.0422
Original	IF Bootstrap	8.1273e+03	4.5631e+03	0.0390
Additive Outliers	Residual Bootstrap	9.2355e+03	5.4599e+03	0.0512
Additive Outliers	IF Bootstrap	8.5532e+03	4.7891e+03	0.0434
Volatility Outliers	Residual Bootstrap	9.3100e+03	5.6788e+03	0.0525
Volatility Outliers	IF Bootstrap	8.7113e+03	4.8197e+03	0.0477
Both Outliers	Residual Bootstrap	9.5623e+03	5.8391e+03	0.0612
Both Outliers	IF Bootstrap	8.9045e+03	5.1161e+03	0.0568

Table 3.9: Summary of performances

As learned from the table above and from the previous sections, robust bootstrap techniques and, in particular that of the IF bootstrap, are an efficient way to handle situations where there are outliers.
In particular, such methods, as in the case just described, can be applied in various situations to make predictions based on past data. In the case of energy generation from renewable sources such as wind, the implementation of these techniques would allow to predict the future more accurately than standard methods, favoring a more efficient energy production and management than ever, reducing waste and meeting the demand on the market.
It would also lead to a possible increase in revenues for energy supply companies as they could develop precise plans according to the forecasts analyzed. Nowadays, with the open conflict between Ukraine and Russia, the energy problem is affecting several countries, especially in Europe which from one day to the next have found themselves having to fill shortages of resources and having to face this energy crisis. Although the outbreak of a war is an unpredictable phenomenon, accurate forecasts on energy production would allow the various countries to balance consumption and production by reducing waste and, favoring the development of sustainable plans from an economic point of view, without having to go to affect in suddenly the pockets and wallets of end consumers.
The result achieved with the Influence Function bootstrap method is of considerable importance even though, at first glance, the performances achieved are not so better than those of the classical residual bootstrap if we think about RMSE, MAE, and MAPE.
Suffice it to say that the presence of outliers in a dataset is something unpredictable

(such as the outbreak of a war) and being able to manage them, in the long run, brings advantages especially if we are talking about ARIMA-GARCH models that are based on past data. If these were treated normally like other data, they would undermine performance and disadvantage accurate forecasting. Furthermore, the results achieved could be further improved by increasing the number of simulations or the number of bootstrap resamplings but, in any case, they represent a let-through for the development of other robust bootstrap techniques.
These techniques, in fact, although already analyzed for several years, are still little recognized worldwide when instead, in my opinion, they represent an important future opportunity for two main reasons: being able to be applied in the case of small sample sizes and being able to foresee the future accurately even in the presence of deviations and outliers.

3.7 Application to wind power generation

In order to validate the adopted algorithm, it has been applied also to another dataset, still from Data Platform (an open power system data platform providing European power system data), collecting wind power generation and capacities from 2015 to 2020.
The same steps as for the heating demand case have been performed in order to check whether the Influence Function bootstrap is able to achieve better performances, in comparison with the classical residual bootstrap, also in different scenarios.
Like has been done previously, after careful examinations (differencing transform, ADF and Engle test, AIC and BIC criterion), the new dataset has been modeled using an ARIMA(1,1)-GARCH(1,1) process.

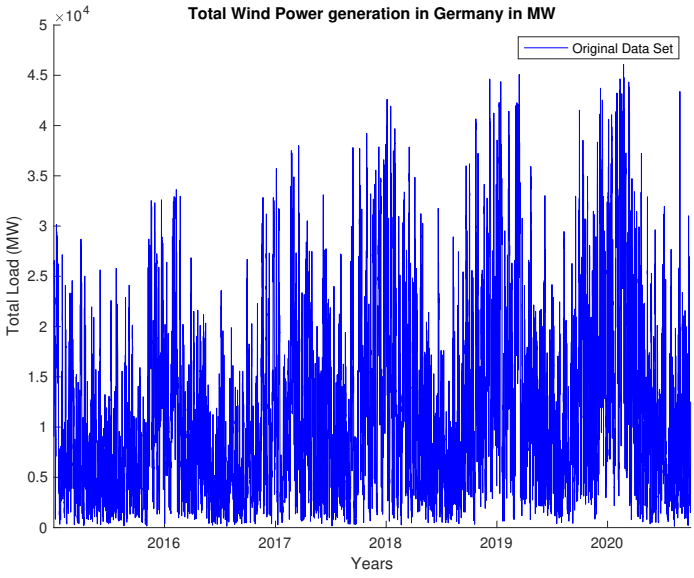

Figure 3.14: Wind power generation in Germany in MW

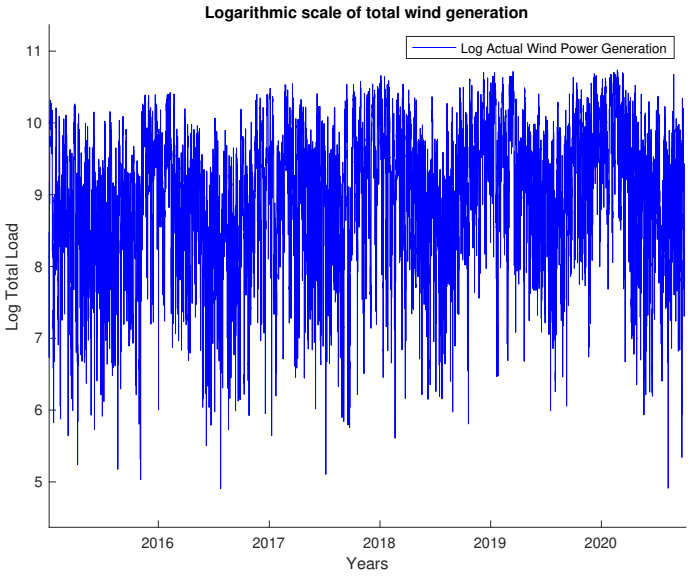

Figure 3.15: Logarithmic data representation

Concerning the differencing method, the combination $H\left(x_t\right) = \Delta_1 \cdot \Delta_{12} \cdot \Delta_{24} x_t, t = 1, ..., T$, was found to be the best.
Once Augmented Dickey-Fuller (ADF) test has been performed on H(x_t), the series could be considered stationary since the rejection of the hypothesis that there is a unit root was confirmed.

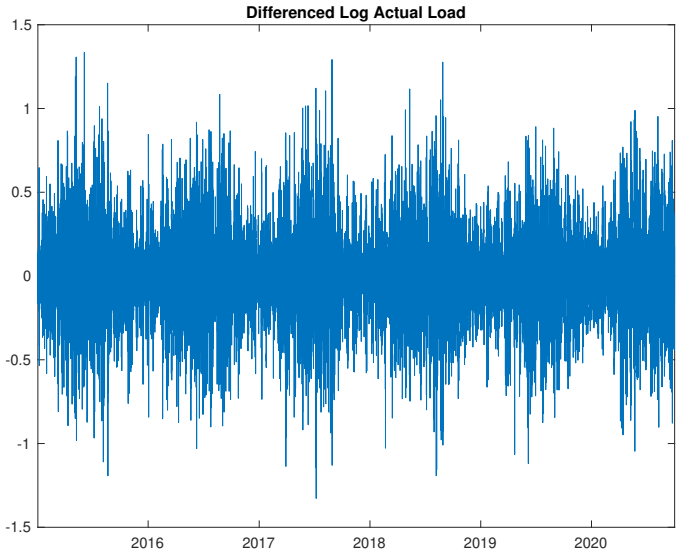

Figure 3.16: Nonseasonal and seasonal differenced log actual wind generation

p \ q	1	2	3	4
1	**-6.1034**	-5.9691	-5.8830	-5.8595
2	-6.0846	-2.9452	-2.9680	-2.9427
3	-4.9347	-2.7844	-2.1442	-2.1450
4	-5.0588	-2.9459	-2.1134	-1.8517

Table 3.10: AIC calculation results ($\cdot 10^4$) for ARMA order

p \ q	1	2	3	4
1	**-6.1026**	-5.9691	-5.8830	-5.8595
2	-6.0836	-2.9439	-2.9664	-2.99409
3	-4.9334	-2.7828	-2.1424	-2.1429
4	-5.0572	-2.9440	-2.1114	-1.8493

Table 3.11: BIC calculation results ($\cdot 10^4$) for ARMA order

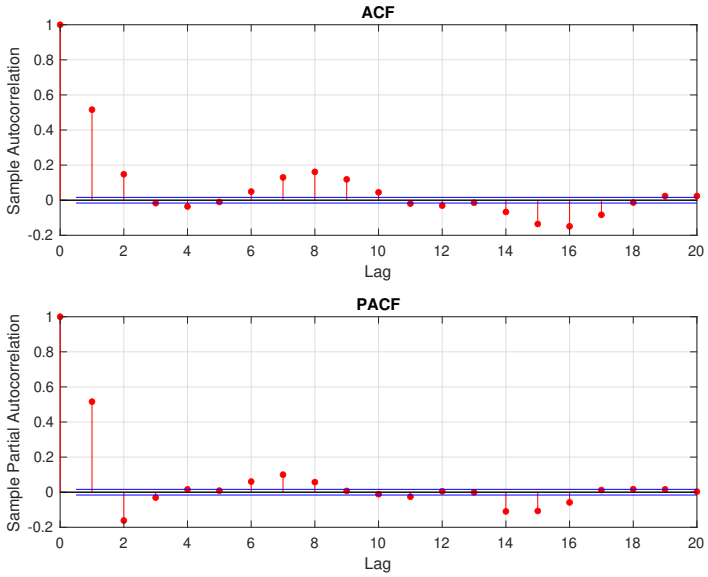

Figure 3.17: ACF and PACF of differenced logarithmic data

As showed by the above plot, both the ACF and PACF have the biggest correlation values at lag 0 and lag 1, and then they start slowly decreasing. This result, based on the theory discussed previously, means that is reasonable to adopt an ARIMA model with $p = 1$ and $q = 1$, as suggested before by the AIC and BIC criterion.
Having found the order, an ARMA(1,1) model has been constructed and fitted to the data. But, before doing that, the residuals have been analyzed in order to perform an Engle test to check for the presence of ARCH effects.

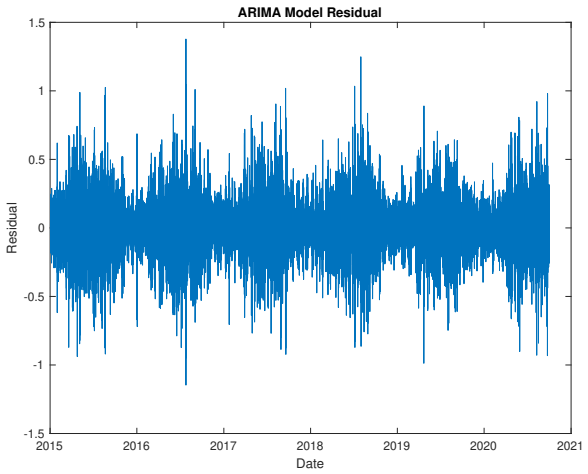

Figure 3.18: ARIMA model residual

Beyond the residuals, also the volatility, obtained by taking the square root of the conditional variance, was analyzed.
Furthermore, the residuals have been also standardized by taking the ratio of their values and the volatility. The standardized residuals were then used for performing bootstrap based forecasting.

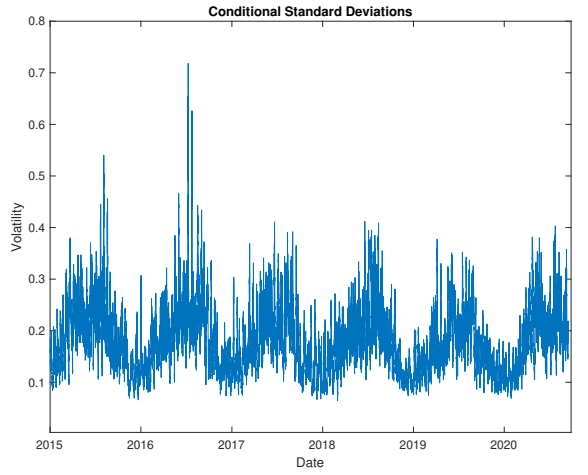

Figure 3.19: Conditional standard deviation of the ARMA model

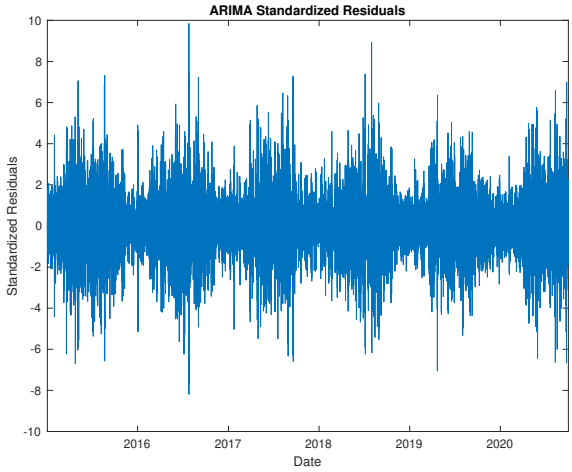

Figure 3.20: Standardized residuals of the ARMA model

Then, the Engle test has been performed on the residual indicating that the null hypothesis of homoskedasticity could be rejected. In other words, residuals are heteroskedastic and so, the composite model ARIMA(p,d,q)-GARCH(r,s) was considered.
Again, AIC and BIC were used in order to get, this time, not only the ARIMA orders but also those of the GARCH process. The most parsimonious model was chosen to be ARIMA(1,1,1)-GARCH(1,1).

p \ q	1	2	3	4
1	**-7.6309**	-7.5200	-7.4654	-7.4536
2	-7.6134	-4.7223	-4.7273	-4.7221
3	-6.4601	-4.5652	-4.7560	-4.1338
4	-6.5737	-4.7227	-4.1083	-3.9724

Table 3.12: AIC calculation results ($\cdot 10^4$) for GARCH order

p \ q	1	2	3	4
1	**-7.6301**	-7.5190	-7.4641	-7.4520
2	-7.6124	-4.7210	-4.7258	-4.7202
3	-6.4588	-4.5636	-4.7542	-4.1317
4	-6.5721	-4.7208	-4.1062	-3.9701

Table 3.13: BIC calculation results ($\cdot 10^4$) for GARCH order

Having found the correct ARIMA(p,d,q)-GARCH(r,s) orders, it was finally possible to construct the model and fit it to the data set. As can be seen in Figure 3.21, and in particular in the zoomed version of it, Figure 3.22, the model fits pretty well the data set.
This demonstrates graphically that the previous described analysis and techniques were necessary in order to achieve good performances.

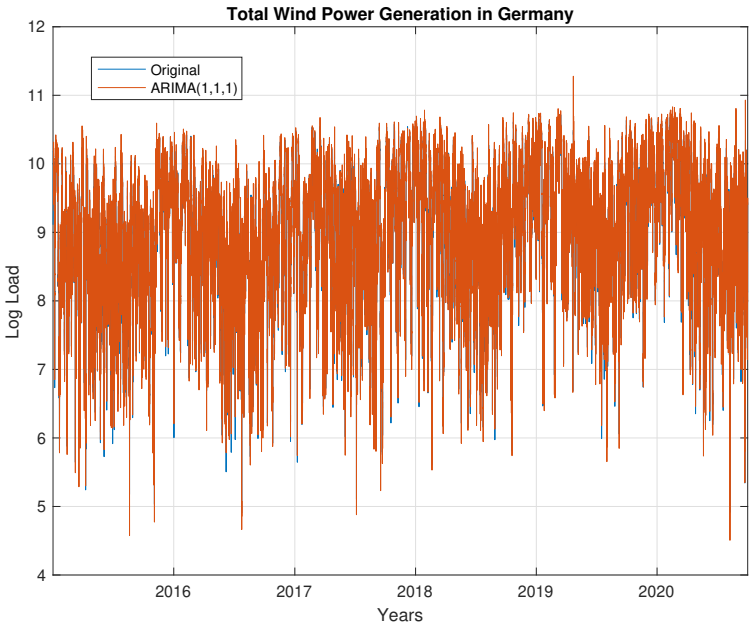

Figure 3.21: Real dataset and ARIMA-GARCH fitted model

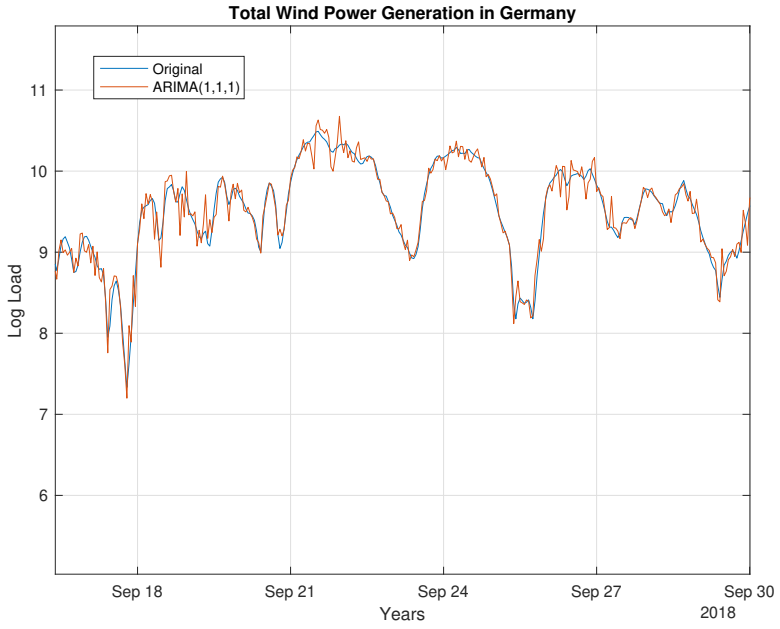

Figure 3.22: Zoom in real dataset and ARIMA-GARCH fitted model

The following parameter values regarding the ARIMA-GARCH process under study were obtained:

ARIMA(1,1,1)	VALUE	STD.ERROR	t-statistic	PValue
Constant	-1.2e-05	0.00048472	-0.024619	0.98036
AR(1)	0.67949	0.0039897	170.31	0
MA(1)	0.26516	0.0054102	49.011	0

Table 3.14: ARIMA parameters

GARCH(1,1)	VALUE	STD.ERROR	t-statistic	PValue
Constant	8.179e-05	5.092e-06	16.062	4.6949e-58
GARCH(1)	0.90804	0.001525	595.42	0
ARCH(1)	0.091964	0.0017181	53.525	0

Table 3.15: GARCH parameters

Considering the above parameters and the model mentioned in equation (2.16) and equation (2.18), it was possible to reconstruct analytically the adopted model:

$$X_t = -1.2e-05 + 0.679X_{t-1} + 0.265\epsilon_{t-1} + \epsilon_t, \tag{3.39}$$

$$h_t = 8.179e-05 + 0.908h_{t-1} + 0.092\epsilon_{t-1}^2. \tag{3.40}$$

Having fitted the model to the dataset, it was possible to perform forecast in differenced log domain and finally get the prediction values back in the original domain applying the inverse seasonal difference operator to the known historical values for the past d data point:

$$\hat{x}_{t+1} = H^{-1}\left(\hat{x}_{t+1}\right) = \left[-\phi_d, ..., -\phi_1, 1\right]^T \left[x_{t-d+1}, ..., x_t, \hat{x}_{t+1}\right], \tag{3.41}$$

where $\phi_1, .., \phi_d$ are as defined in Eq. (2.5).

After having fitted the model, predictions have been performed with and without bootstrap algorithms.
For each one-hour-ahead prediction, the preceding $N_{train} = 30$ data points were used and a total of 100 total predictions were performed.
Without bootstrapping, the predictions gave the following results for what concerns the performance metrics and, in particular, the root mean squared error (RMSE), mean absolute error (MAE), and the mean absolute percentage error (MAPE):

RMSE (MW)	MAE (MW)	MAPE (%)
2.5043e+03	1.3057e+03	1.5642

Table 3.16: Performance without bootstrap

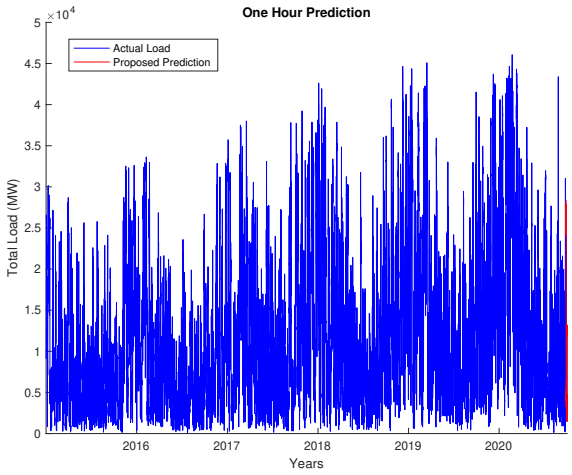

Figure 3.23: Prediction without bootstrapping

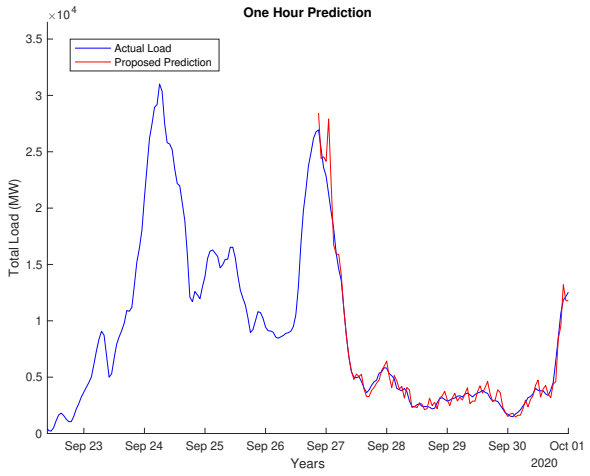

Figure 3.24: Zoom in prediction without bootstrapping

In the same way as before, for each one-hour-ahead prediction, the preceding $N_{train} = 30$ data points are used, 100 predictions are obtained and the number of bootstrap resamples is $B = 100$.

RMSE (MW)	MAE (MW)	MAPE (%)
1.7881e+03	1.2987e+03	0.2366

Table 3.17: Performance with residual bootstrap

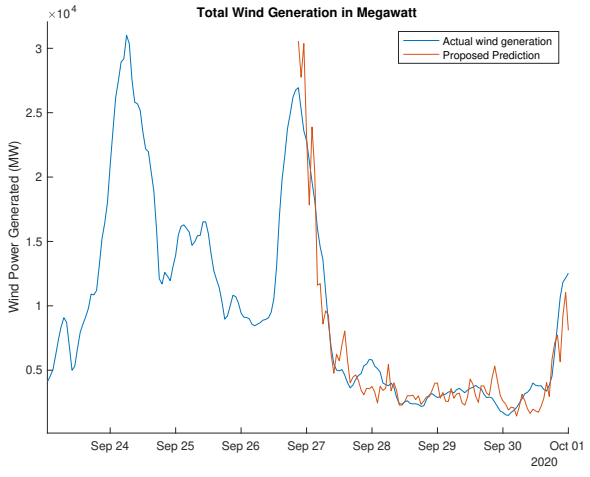

Figure 3.25: Residual bootstrap predictions

In the following, some of the results obtained are represented and discussed when using the IF bootstrap technique for making future predictions.
In particular, a comparison between the performances of applying Influence Function bootstrap and classical residual bootstrap have been studied in order to understand which method is more suitable for making predictions on future data values.
For the first case, moreover, different simulations have been performed using different values of tuning constant c and shape parameter γ to check which range of values suits better for making future predictions.
For computational time costs, the performances have been compared considering a case with 10 Monte Carlo simulations, 30 trained data, 50 bootstrap resamples, and 100 total predictions.

C value	γ value	RMSE (MW)	MAE (MW)	MAPE (%)
0.1	1.1	1.9979e+03	2.1599e+03	0.3245
0.5	1.1	2.1034e+03	2.2099e+03	0.3211
1	1.1	2.1167e+03	2.1363e+03	0.3222
1.5	**1.1**	1.9971e+03	2.1233e+03	**0.2987**
2	1.1	2.2982e+03	2.1546e+03	0.3461
1.5	1.01	2.2660e+03	2.0962e+03	0.4187
1.5	1.3	2.4533e+03	2.1583e+03	0.4143
1.5	2.5	2.2403e+03	2.0413e+03	0.3356
1.5	5	2.1987e+03	1.9877e+03	0.3200
1.5	**10**	**1.9823e+03**	**1.3448e+03**	0.3119
1.5	15	2.1021e+03	1.5788e+03	0.3357

Table 3.18: Performance with different c and γ values

As can be observed in Table 3.18, the best performances have been obtained for the case of $c = 1.5$ and $\gamma = 10$. However, looking at the case of $c = 1.5$ and $\gamma = 1.1$, the results are not that different so, another comparison has been performed considering a bigger number of data to be predicted, 100, to achieve bigger accuracy. The following results have been obtained:

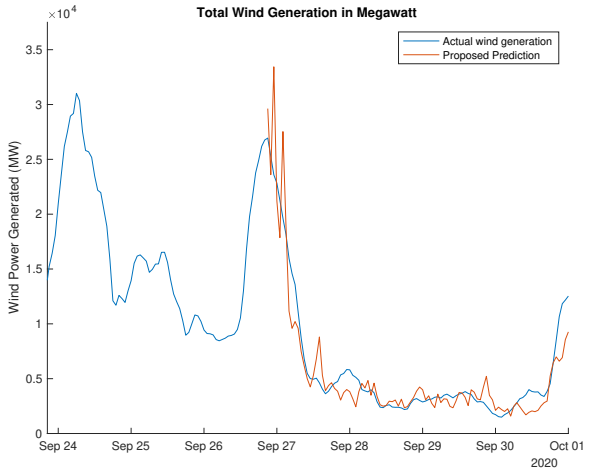

Figure 3.26: Influence Function bootstrap predictions with $\gamma = 1.1$

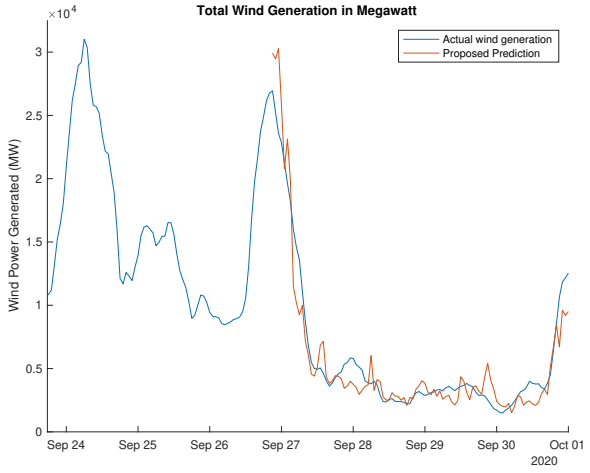

Figure 3.27: Influence Function bootstrap predictions with $\gamma = 10$

C value	γ value	RMSE (MW)	MAE (MW)	MAPE (%)
1.5	1.1	1.7408e+03	1.2608e+03	0.2219
1.5	10	1.6868e+03	1.1599e+03	0.2235

Table 3.19: Performance comparison when $\gamma = 1.1$ and $\gamma = 10$

Basically, with a fixed value of tuning constant, is not that easy to find a good estimate for what concern the shape parameter since, with $\gamma = 1.1$ smaller MAPE is obtained but bigger MAE and RMSE.

In the following, however, the case of $\gamma = 10$ has been considered.
After having identified the most suitable values for what concern the tuning constant c and the shape parameter γ, a new simulation has been carried out taking into account the real dataset without modification in order to see the differences concerning the performances of classical residual bootstrap and Influence Function bootstrap (with c set to 1.5 and γ to 10).
For both cases, a number of 10 Monte-Carlo simulations has been performed considering a number of 100 predictions, using 30 trained data and 100 bootstrap resamples, form the original dataset.
As done previously, again, MAPE, MAE, and RMSE have been estimated:

Resampling method	RMSE (MW)	MAE (MW)	MAPE (%)
Residual Bootstrap	1.7881e+03	1.2987e+03	0.2366
IF Bootstrap	1.6868e+03	1.1599+03	0.2235

Table 3.20: Performance comparison between residual and IF bootstrap

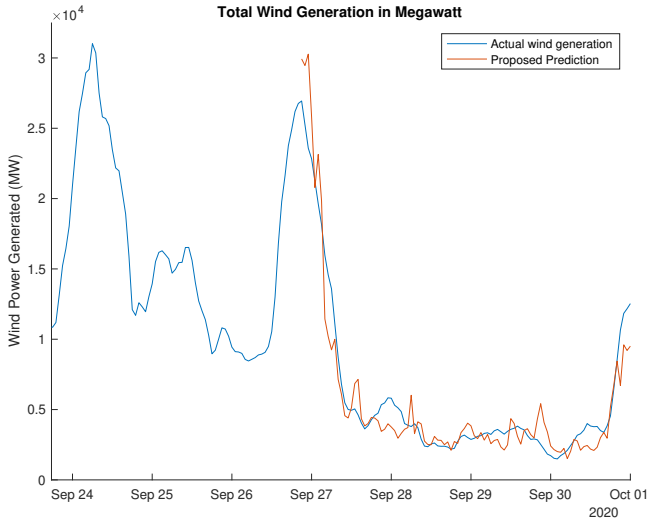

Figure 3.28: Predictions based on Influence Function bootstrap

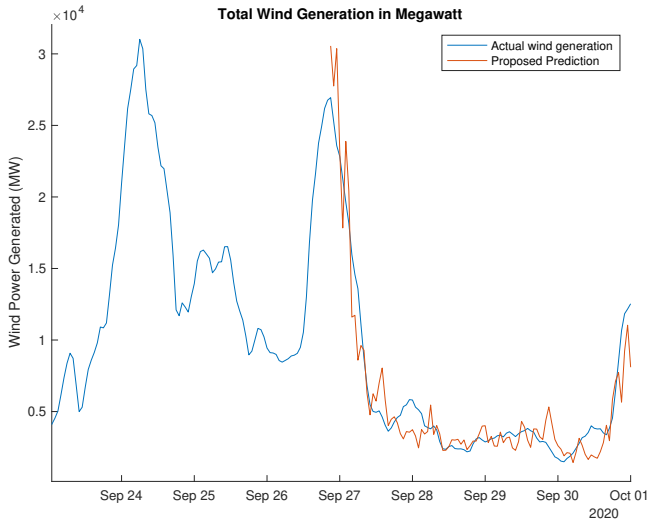

Figure 3.29: Predictions based on classical residual Bootstrap

After this, the data set has been modified in order to add volatility and level outliers and check wheter, also in this cases, the IF bootstrap is more robust than the classical residual bootstrap by comparing the perfomance metrics MAE, MAPE, and RMSE.

3.7.1 Level outliers case

The original dataset, collecting a total number of 50350 values, has been transformed such that:

Indices of modified data	Original mean value	New mean value
5000:5010	2.0268e+04	3.5268e+04
20000:20100	1.3435e+04	3.6435e+04
45000:45700	2.2443e+04	2.4943e+04

Table 3.21: Level outliers introduction

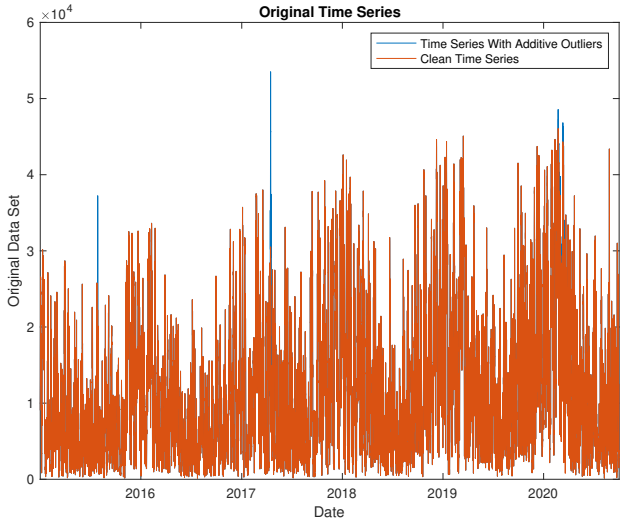

Figure 3.30: Example of additive outliers

A number of 25 Monte-Carlo simulations has been performed considering a number of 100 predictions, using 30 trained data form the original dataset, and considering 100 bootstrap resamples.
A comparison of MAE, MAPE, and RMSE has been conducted.

Resampling method	RMSE (MW)	MAE (MW)	MAPE (%)
Residual Bootstrap	2.3454e+03	2.1011e+03	0.2495
IF Bootstrap	2.1500e+03	1.9195e+03	0.2447

Table 3.22: Performance with level outliers

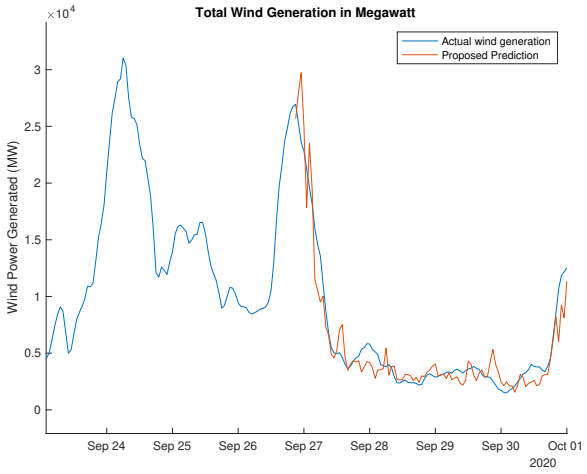

Figure 3.31: Influence Function bootstrap with additive outliers

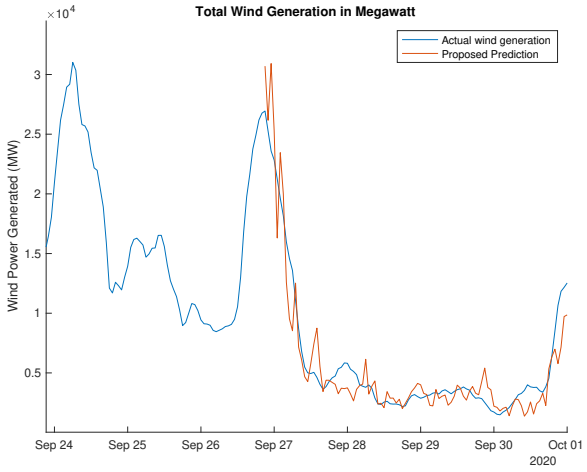

Figure 3.32: Classical residual bootstrap with additive outliers

As can be observed in Table 3.22, the IF bootstrap performances are better than those of residual classical bootstrap. Furthermore, looking at Figure 3.31 and Figure 3.32 it is possible to observe also graphically, how the forecasted prediction of the Influence Function bootstrap are better following the ARIMA-GARCH model fitted to the data set.

3.7.2 Volatility outliers case

Considering the model discussed in equation (3.10), in the same way as in previous section, a comparison between classical residual bootstrap and Influence Function bootstrap has been studied after having added innovation outliers in the clean time series giving the following modifications:

Indices of modified data	Original mean value	New mean value
5000:5010	0.1388	0.4388
20000:20100	0.1595	0.3695
45000:45700	0.1441	0.4741

Table 3.23: Level outliers introduction

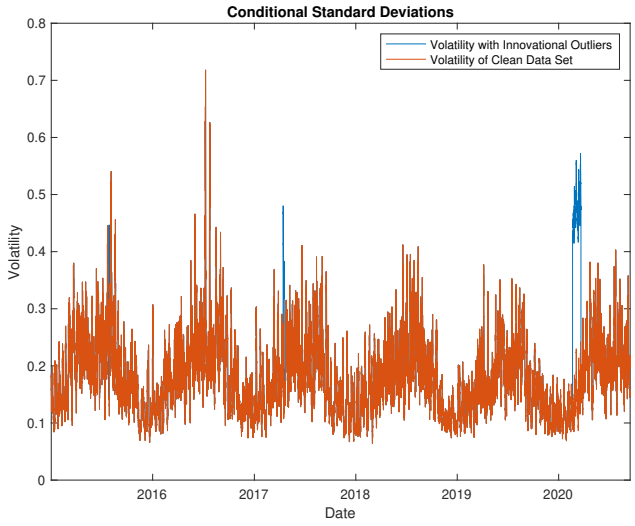

Figure 3.33: Example of innovational outliers

Resampling method	RMSE (MW)	MAE (MW)	MAPE (%)
Residual Bootstrap	2.4215e+03	2.1590e+03	0.2503
IF Bootstrap	2.1929e+03	1.9205e+03	0.2472

Table 3.24: Performance with volatility outliers

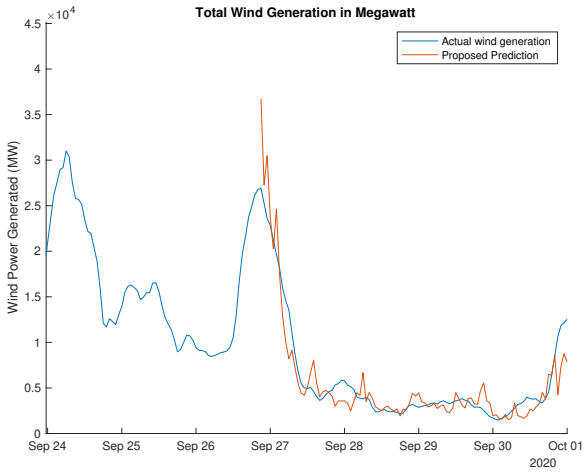

Figure 3.34: Influence Function bootstrap with volatility outliers

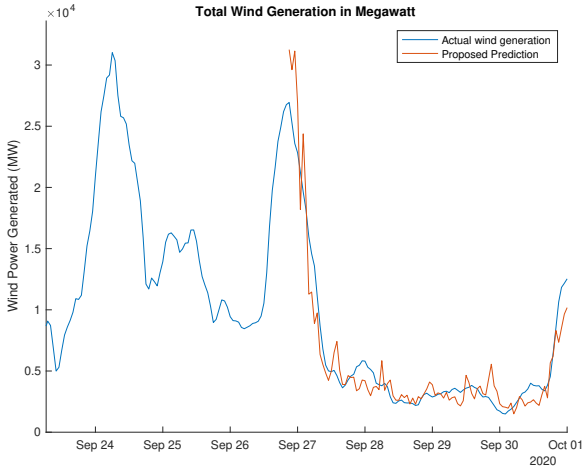

Figure 3.35: Classical residual bootstrap with volatility outliers

Concerning the case of volatility outliers, the situation is a bit different since the performances are, in general a bit worse. Graphically, at Figure 3.34 and Figure 3.35, it is possible to see that the predictions is not working really fine at the beginning (September 27) but, after some hours, the prediction line start to follow the real trend of the data set, thus showing also in this case that the bootstrap methods are particularly useful for making predictions.
Moreover, as in the case of level outliers, also for volatility ones, it can be seen in the Table 3.24 how the IF bootstrap manages to achieve better performance than the classical one based on residuals.

3.7.3 Level and volatility outliers simultaneously

The third case combines the previous two in order to check the performances when both additive and volatility outliers are present in the data set.
In the same way as before, a number of 25 Monte-Carlo simulations has been performed considering a number of 100 predictions, using 30 trained data form the original dataset, and considering 100 bootstrap resamples.
A comparison of MAE, MAPE, and RMSE has been conducted to check whether Influence Function bootstrap is able to reach better results than classical residual bootstrap also in this case.

Resampling method	RMSE (MW)	MAE (MW)	MAPE (%)
Residual Bootstrap	2.5991e+03	2.2544e+03	0.2697
IF Bootstrap	2.2509e+03	2.1005e+03	0.2525

Table 3.25: Performance with level and volatility outliers

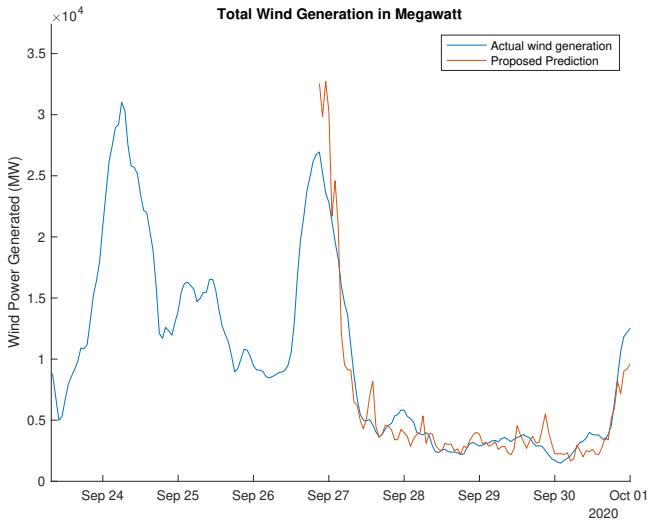

Figure 3.36: Influence Function bootstrap with additive and volatility outliers

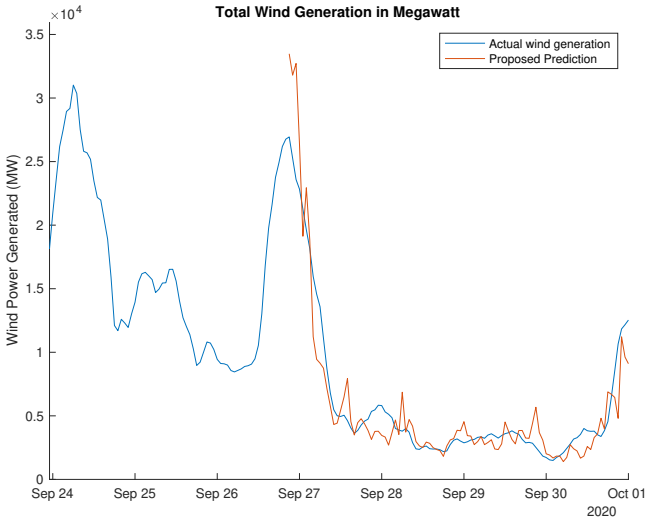

Figure 3.37: Classical residual bootstrap with additive and volatility outliers

The same situation as before arised also when considering at the same time both level and volatility outliers. Of course, in this scenario, the performances are a bit worse but still showing an improvement when considering the Influence Function bootstrap with respect to the classical residual bootstrap.

In order to have a total view of the results obtained previously, in the various cases, a summary table is shown below. All the simulations below were performed considering $\gamma = 10$, 30 trained data from the original dataset, 25 Monte-Carlo simulations, 100 bootstrap resamples and a total of 100 predictions.

Data Set	Resampling method	RMSE (MW)	MAE (MW)	MAPE (%)
Original	Residual Bootstrap	1.7881e+03	1.2987e+03	0.2366
Original	IF Bootstrap	1.6868e+03	1.1599e+03	0.2235
Additive Outliers	Residual Bootstrap	2.3454e+03	2.1011e+03	0.2495
Additive Outliers	IF Bootstrap	2.1500e+03	1.9195e+03	0.2447
Volatility Outliers	Residual Bootstrap	2.4215e+03	2.1590e+03	0.2503
Volatility Outliers	IF Bootstrap	2.1929e+03	1.9205e+03	0.2472
Both Outliers	Residual Bootstrap	2.5991e+03	2.2544e+03	0.2697
Both Outliers	IF Bootstrap	2.2509e+03	2.1005e+03	0.2525

Table 3.26: Summary of performances

Chapter 4

Conclusion

Taking up what was said at the beginning of this document, time series forecasting is an important topic not yet fully explored and which, with increasingly refined techniques, would allow a balance between demand and energy production without encountering waste and allowing supply companies to increase their revenues and refine their financial decisions.
In this document was showed that the wind power generation can be predicted with a reasonable accuracy by fitting the historical data to the ARMA(1,1)-GARCH(1,1) model. Before fitting the model, a series of seasonal and non-seasonal de-trending was proposed and applied to the original time series to make it stationary.
A simple inverse differencing technique (as descirbed at page 32) was then used to bring back the results to the original domain for better interpretability.
A residual bootstrapping method to obtain future prediction was proposed by taking the median value of the bootstrapped resamples and compared with a new robust bootstrap method, the so called Influence Function bootstrap.
Several comparisons of the models has been carried on, under different circumstances: clean time series data, contaminated time series with level outliers, volatility outliers and time series data contaminated with both kinds of outliers simultaneously.
In all the cases, the robust bootstrap algorithm showed improvements with respect to the residual bootstrap, due to the outlier manipulation present in the former.
However, an ongoing attempt can be the implementation of the other robust bootstrap techniques explained in Section 3.2 to see whether the other methods can achieve even better performances.

Bibliography

[1] C. Amado and A. M. Pires. Robust bootstrap with non random weights based on the influence function. *Communications in Statistics-Simulation and Computation*, 33(2):377–396, 2004.

[2] M. Arellano. Lagrange multiplier test, 2002.

[3] J. A. Doornik and M. Ooms. Outlier detection in garch models. Technical report, Tinbergen Institute Discussion Paper, 2005.

[4] R. F. Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the econometric society*, pages 987–1007, 1982.

[5] L. K. Hotta and R. S. Tsay. Outliers in garch processes. In *Economic Time Series*, pages 355–376. Chapman and Hall/CRC, 2018.

[6] R. J. Hyndman and G. Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[7] R. Mushtaq. Augmented dickey fuller test. 2011.

[8] Y. Shin and P. Schmidt. The kpss stationarity test as a unit root test. *Economics Letters*, 38(4):387–392, 1992.

[9] A. M. Zoubir and D. R. Iskandler. Bootstrap methods and applications. *IEEE Signal Processing Magazine*, 24(4):10–19, 2007.

[10] A. M. Zoubir, V. Koivunen, E. Ollila, and M. Muma. *Robust statistics for signal processing*. Cambridge University Press, 2018.