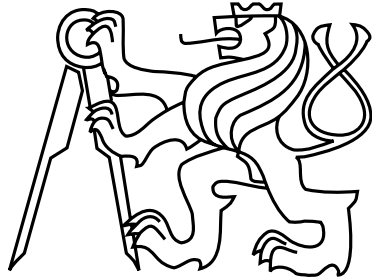


Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Computer Science



Master's Thesis

Time Series Data Prediction and Analysis

Oleg Ostashchuk

Supervisor: Ing. Miloš Kozák, Ph.D.

Study Programme: Open Informatics

Field of Study: Software Engineering

January 2017

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Computer Science

DIPLOMA THESIS ASSIGNMENT

Student: **Bc. Oleg Ostashchuk**

Study programme: Open Informatics
Specialisation: Software Engineering

Title of Diploma Thesis: **Prediction Time Series Data Analysis**

Guidelines:

1. Survey existing methods of time series analysis for prediction of time series behavior.
2. Study provided data, and select at least 3 methods for analysis. Discuss necessary data pre-processing for each suggested method.
3. Implement selected methods for data pre-processing and data analysis for each data set.
4. Extend selected dataset and method in order to improve accuracy of prediction for specified prediction period.
5. Evaluate the accuracy of implemented method and discuss an improvement.

Bibliography/Sources:

- [1] HILLIER, F.S.; LIEBERMAN, G.J.: Introduction to operations research. 7th Edition. Boston: McGraw-Hill, 2001, 1214 p. ISBN: 0-07232-169-5.
- [2] HAYKIN, S.O.: Neural networks and learning machines. 3rd Edition. New York: Prentice Hall, 2009, 936 p. ISBN: 0-13147-139-2.
- [3] MCKINNEY, W.: Python for data analysis. Beijing: O'Reilly Media, 2013, 466 p. ISBN: 1-44931-979-3.

Diploma Thesis Supervisor: Ing. Miloš Kozák, Ph.D.

Valid until the end of the summer semester of academic year 2016/2017



Prague, February 17, 2016

Acknowledgements

I would like to thank my supervisor Ing. Miloš Kozák, Ph.D. for his help, spent time, as well as his kindly approach for several years of collaboration during bachelor's and master's degree studies.

Declaration

I hereby declare that I have completed this thesis independently and that I have listed all the literature and publications used.

I have no objection to usage of this work in compliance with the act §60 Zkon . 121/2000Sb. (copyright law), and with the rights connected with the copyright act including the changes in the act.

In Prague on Jan 9, 2017

.....

Abstract

Given thesis deals with the problematic of time series analysis and forecasting. The aim of thesis is to survey an existing time series forecasting methods, including necessary data preprocessing steps. There are selected three promising forecasting methods, including ARIMA method, artificial neural networks method and double exponential smoothing method.

There are also selected three real life datasets from different areas. Individual forecasting models have been implemented for each dataset, in MATLAB programming environment. In practical part of thesis, there are demonstrated results of performed experiments, including dependency between forecasting accuracy and the size of training set.

At the end of the thesis, there are results summary and further improvements are discussed.

Abstrakt

Diplomová práce se věnuje problematice analýzy a prognózování časových řad. Cílem práce je prozkoumat existující metody prognózování časových řad, včetně potřebných kroků předzpracování dat. Jsou vybrané tři slibné metody prognózování, včetně ARIMA, metody prognózování pomocí Neuronových sítí a metody dvojitého exponenciálního vyrovnání.

Dále jsou vybrané tři datové sady z praxe, pro které byli v programovém prostředí MATLAB implementované jednotlivé modely prognózování. V praktické části práce jsou demonstrovány výsledky jednotlivých experimentů, včetně výkonu jednotlivých metod v závislosti na rozměru tzv. "trénovací sady" dat.

V závěru práce je provedené zhodnocení výsledku a jsou uvedené perspektivy pro další vylepšení kvality predikce.

Contents

List of Figures	x
List of Tables	xi
Abbreviations	xii
I Theoretical part	1
1 Introduction	2
1.1 Aims of the Thesis	2
2 Time Series Analysis	4
2.1 Introduction to Time Series	4
2.2 Time Series Types Classification	6
2.3 Aims of Time Series Analysis	7
2.4 Time series components	8
2.5 Autocorrelation and Partial Autocorrelation	11
2.5.1 Autocorrelation function	11
2.5.2 Partial Autocorrelation function	11
2.6 Time series forecasting	12
2.6.1 Forecasting without external factors	13
2.6.2 Forecasting with external factors	15
2.7 Data preprocessing	16
2.7.1 Outliers detection	16
2.7.2 Denoising and Smoothing	17
2.7.3 Differencing	18
2.7.4 Scaling	18
2.7.5 Normalization	19
3 Forecasting Methods	20
3.1 Regression models	20
3.2 Autoregressive and moving average models	21
3.3 Exponential smoothing models	22
3.3.1 Double exponential smoothing	23
3.4 Artificial neural networks models	24
3.4.1 Biological inspiration	24
3.4.2 Artificial neuron model	25

3.4.3	Types of Activation Function	27
3.4.4	Neural Network Architectures	28
3.4.5	Appropriate architecture	29
3.4.6	Networks training	30
3.4.7	Cross-validation	30
3.4.8	ANN forecasting model	32
3.5	Markov chain models	32
3.6	Forecasting models comparison	33
II	Practical part	35
4	Implementation	36
4.1	Data science tools	36
4.1.1	The R language	36
4.1.2	MATLAB	37
4.1.3	Python	37
4.2	Experiments definition	37
4.3	Forecasting Accuracy	39
5	Experiment 1	41
5.1	Data description	41
5.2	Data preprocessing	42
5.3	Forecasting results Part 1.	42
5.4	Forecasting results Part 2. - Extended Training set	44
6	Experiment 2	45
6.1	Data description	45
6.2	Data preprocessing	46
6.3	Forecasting results Part 1.	46
6.4	Forecasting results Part 2. - Extended Training set	48
7	Experiment 3	49
7.1	Data description	49
7.2	Data preprocessing	50
7.3	Forecasting results Part 1.	50
7.4	Forecasting results Part 2. - Extended Training set	52
8	Results Summary	53
8.1	Results Summary	53
9	Conclusion	56
9.0.1	Further improvements	56
	Bibliography	57

List of Figures

2.1	Oil prices	4
2.2	GDP progress of Czech Republic	5
2.3	Seismogram from HAWA station (Hanford, Washington, USA)	6
2.4	Monthly international airline passenger counts from 1949 to 1960	9
2.5	Time series components	10
2.6	ACF and PACF of monthly airline passenger counts	12
2.7	Time series forecasting without external factors	14
2.8	Time series forecasting with external factors	16
2.9	Outliers detection example	17
3.1	Biological neuron	25
3.2	Artificial neuron model	26
3.3	Activation functions	27
3.4	Artificial neural network example	28
3.5	Artificial neural network example	29
3.6	Overfitting example	31
3.7	Cross-validation	31
3.8	ANN and time series forecasting	32
3.9	Markov chain model	33
5.1	Internet traffic data (in bits)	41
5.2	ACF and PACF of Internet traffic data	42
6.1	Electricity Load (in kW for each 15 min interval)	45
6.2	ACF and PACF of Electricity Load data	46
7.1	Daily IBM stock prices (in USD)	49
7.2	ACF and PACF of IBM stock prices	50
8.1	Dependency between number of predicted values and forecasting accuracy	54

List of Tables

5.1	Results of Experiment 1 - part 1	43
5.2	Results of Experiment 1 - part 2	44
6.1	Results of Experiment 2 - part 1	47
6.2	Results of Experiment 2 - part 2	48
7.1	Results of Experiment 3 - part 1	51
8.1	Forecasting methods - Average error	53
8.2	Forecasting improvement by extension of training dataset	54

Abbreviations

NN	N eural N etwork
ANN	A rtificial N eural N etwork
FFNN	F eed F orward N eural N etwork
RNN	R ecurrent N eural N etwork
DES	D ouble E xponential S smoothing

Part I

Theoretical part

Chapter 1

Introduction

The word "prediction" originates from a Latin statement "praedicere", which was originally denoted by meanings "to say beforehand" or "to mention in advance". Today, "prediction" is usually referred to some kind of message or opinion about an event that is expected to happen in future. Inside the more formal science context, the process of making predictions about future by using scientific methods is usually denoted by term "forecasting". Processes that are usually required to be forecasted, are the most often stored in a so called time series format. [1]

Time series is a common mathematical expression that can be frequently observed in various texts about statistics, signal processing or econometrics. Every day, newspapers contain business sections, which report daily stock prices, foreign currency exchange rates or monthly rates of unemployment. Meteorology records usually consists of hourly wind speeds, daily maximum and minimum temperatures or annual rainfall. Geophysics are continuously observing processes like shaking or trembling of the earth, in order to predict possibly impending earthquakes. All these and certainly many other examples could be mentioned to describe the role of time series in our society. [2]

1.1 Aims of the Thesis

Today, there is plenty of various forecasting methods and each of them requires the corresponding conditions and proper data preprocessing. Performing a research in the given problematic, it can be observed, that the autoregressive methods and exponential smoothing belong to the most frequently used forecasting methods. Additionally, an Artificial intelligence, especially artificial neural networks demonstrate a great success with the assigned tasks, including the time series forecasting.

Qualitative forecasting is not an easy task. In connection with this, usability and the added value of the given thesis for the potentially interested reader is based on the implementation of the following aims:

1. Presentation of extensive and important information in the area of time series analysis and forecasting.
2. Introduction to time series forecasting methods and the corresponding data preprocessing.
3. Implementation of exemplary forecasting models by using Artificial neural networks and traditional statistical forecasting methods.
4. Demonstration of appropriate data preprocessing and the effectiveness of each forecasting model in the work over datasets from various potential sectors.
5. Comparison of results and recommendations about further improvements.

Chapter 2

Time Series Analysis

2.1 Introduction to Time Series

The term "time series" itself, denotes a data storing format, which consists of the two mandatory components - time units and the corresponding value assigned for the given time unit. Values of the series need to denote the same meaning and correlate among the nearby values. Restriction is, that at the same time there can be at most one value for each time unit. For example, sequences, which just enumerate some values, they do not fulfill the time series requirements. Figure 2.1 demonstrates oil prices progress over last 30 years, a typical example of time series.



FIGURE 2.1: Oil prices

In theory, there are two fundamental ways, how time series data are recorded. The first way, values are measured just for the specific timestamps, what may occur periodically, or occasionally according to concrete conditions, but anyway, result will be a discrete set of values, formally called discrete time series. This is very common case and frequently observed in practice. In economy sector, most of the indicators are measured periodically with the specific periods, therefore economic indicators represent an appropriate example of discrete time series. Figure 2.2 demonstrates example of discrete time series, GDP progress of Czech Republic, over the last 40 years.

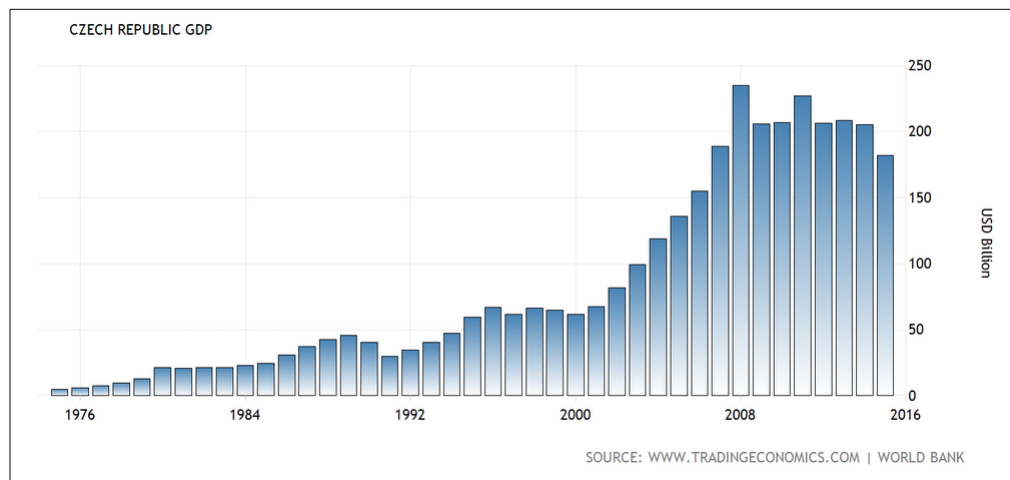


FIGURE 2.2: GDP progress of Czech Republic

The second option, data are measured and recorded continuously along the time intervals. Electrical signals from sensors, earth shakings, various indicators from medicine, like ECG, or many other scientific sensors, they all represent a continuous measurement of corresponding physical quantity. This kind of processes produces a continuous time series. Figure 2.3 demonstrates a seismogram from station HAWA (Hanford, Washington, USA), example of continuous time series.

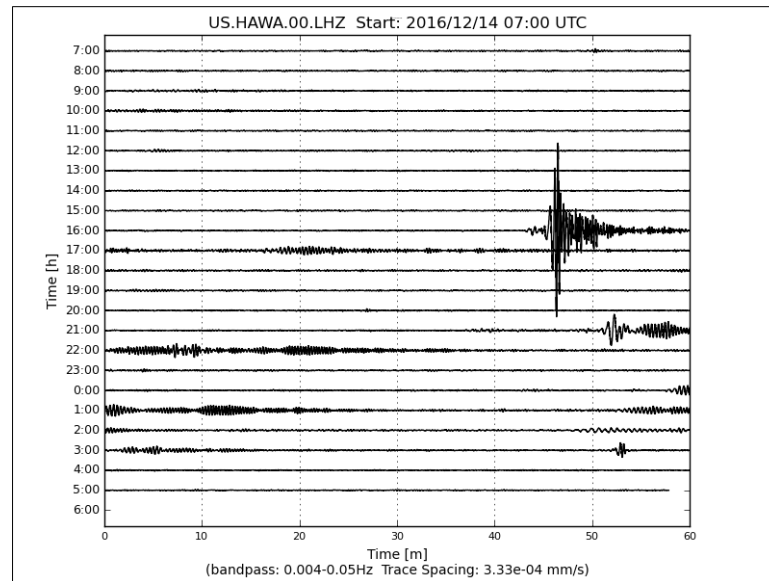


FIGURE 2.3: Seismogram from HAWA station (Hanford, Washington, USA)

Despite the fact, that many scientific processes produce continuous time series data, numerical approach of computer systems allows to store data only as the discrete values. Thus all further forecasting methods performed on computer, assume test data in the discrete values form.

2.2 Time Series Types Classification

There are many various time series classifications based on specific criteria. The most significant dependencies are: length of the time step, memory and stationarity.

Depending on the distance between recorded values, time series data are classified into:

- equidistant time series
- non-equidistant time series

Equidistant time series are formed, when its values are recorded periodically with a constant period length. A lot of physical or environmental processes are described by this kind of time series. Non-equidistant time series are those time series, which do not keep the constant distance between observations. Econometric indicators, like stock prices are not necessary performed within regular time intervals, they are regulated by a concrete supply and demand rates on the specific market. Therefore, this kind of series suitably demonstrates a non-equidistant time series example.

According to the rate of dependency between newly observed values and its predecessors, time series are divided into:

- long memory time series
- short memory time series

Time series with long memory are those, for which the autocorrelation function decreases slowly. [1] This kind of time series usually describes processes, which don't have fast turnovers. Traffic congestion, electric energy consumption, different physical or meteorological indicators, like air temperature measurements, all these processes are usually described by long memory time series. Short memory time series are those, for which autocorrelation function is decreasing more rapidly. Typical examples contain processes from the econometric sector.

Another classification of time series is based on their stationarity:

- stationary time series
- non-stationary time series

Stationary time series are time series, for which statistical properties like mean value or variance, are constant over time. These time series stay in relative equilibrium in relation to its corresponding mean values. Other time series belong to non-stationary time series. In industry, trading or economy, time series more frequently belongs to the non-stationary category. In order to deal with the forecasting task, non-stationary time series are usually transformed to the stationary ones, by the appropriate preprocessing methods.

2.3 Aims of Time Series Analysis

Time series analysis unites a group of methods for work with time series data, in order to extract the potentially useful information. There are two main goals of time series analysis:

1. Determination of the time series behavior - Identification of the important parameters and characteristics, which adequately describe the time series behavior.
2. Time series forecasting - Forecasting the future values of the time series, depending on its actual and past values.

Both of these goals require the time series model identification. As soon as the model is indentified, it can be exploited to interpret the time series behavior, for example, to understand the seasonal changes of the commodity prices. The model can also be used to extrapolate the time series, i.e. to forecast its future values.

2.4 Time series components

Usually, the most of analysis methods assume, that time series data contains the systematic component (typically comprising several components) and random noise (error), which complicates detection of the regular components. Therefore, the majority of methods, includes different noise filtration methods, in order to detect the regular components, or it has to performed during data preprocessing.

The most of the regular components belongs to two main classes. They belong to either a trend or seasonal component. The trend is a general systematic linear or non-linear component, which may change over time. Seasonal component is periodically repeating component. Both these types of regular components are usually presented in the time series simultaneously. For example, sales may increase from year to year, but there is a seasonal component, which reflects the significant growth of sales in December and a drop in August.

This model can be demonstrated on the series representing the monthly international airline passenger counts from 1949 to 1960. The graph of monthly passenger counts clearly demonstrates almost linear trend, i.e. stable increase from year to year (the number of transported passengers in 1960 is four times greater, than in 1949). In the same time, the progress of monthly rates within one year is repeating, and is similar from year to year (for example, the rate of passengers is higher in the periods of holidays).

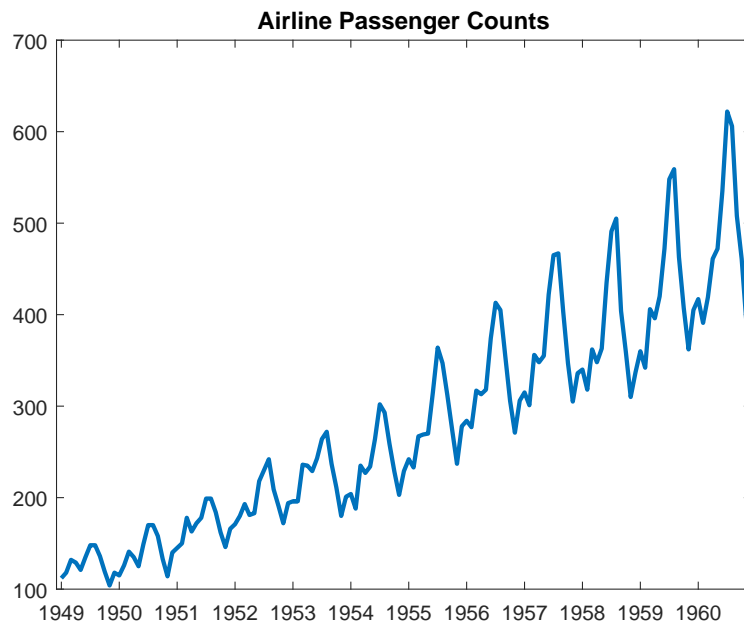


FIGURE 2.4: Monthly international airline passenger counts from 1949 to 1960

It has been already mentioned, that general model of time series usually contains several components: trend component $T(t)$, seasonal component $S(t)$, random noise component $R(t)$, and sometimes there is additionally mentioned a cyclical component $C(t)$. The difference between cyclical and seasonal components is, that seasonal components represents a regular seasonal periodicity, while cyclical component has a longer lasting effect and may vary from cycle to cycle. Very often, cyclical component is integrated into one trend component $T(t)$. Figure 2.5 demonstrate an example of time series decomposition.

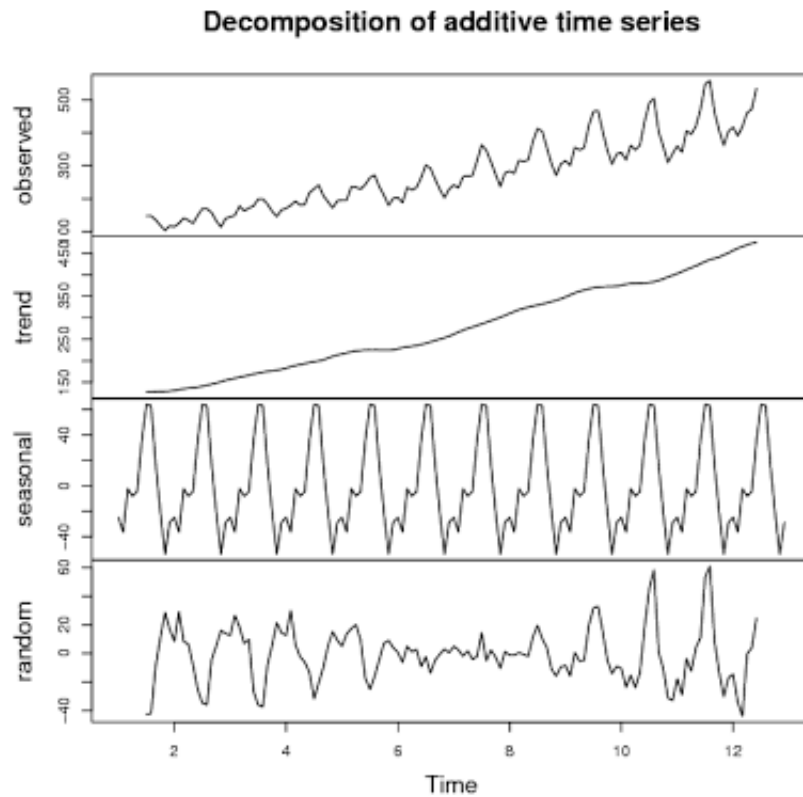


FIGURE 2.5: Time series components

Now, it is important to describe, how this components mathematically interact together, in order to compose a time series. The concrete functional relationships between the components may vary for different series. However, there are two main models, how they interact to each other:

- Additive model

$$Z(t) = T(t) + C(t) + S(t) + R(t) \quad (2.1)$$

- Multiplicative model

$$Z(t) = T(t) \times C(t) \times S(t) \times R(t) \quad (2.2)$$

Main difference between these two models may be observed in a growth rate. Previously mentioned example of monthly airline passenger counts, demonstrates a typical multiplicative model, where the amplitude of seasonal changes increases with the trend. The growths of the trend or seasonal components may be expressed in percentage (multiplicative model) or in absolute values (additive model). [2] [3]

2.5 Autocorrelation and Partial Autocorrelation

Dependencies between the actual and historical values represent a fundamental principle of time series forecasting. It can be easily observed, that each value of the series is very similar to its neighboring values. Additionally, time series contain a seasonal component, what means, that each value is also dependent on the values of identical time, but one season ago. Formally, any statistical dependency between two entities is denoted as a correlation, and is expressed by a corresponding coefficient.

2.5.1 Autocorrelation function

Autocorrelation function calculates the correlations between the time series and its shifted copies at different points in time. The autocorrelations are usually calculated for the specific range of lags (shifts) and are expressed in the form of graph, called correlogram (Figure 2.6). Investigation of autocorrelations, enables to detect important dependencies in time series data. [4]

2.5.2 Partial Autocorrelation function

Sometimes it can happen, that the first value is heavily dependent on the second value, the second value is heavily dependent on the third value and therefore the first value is also dependent on the third, and so on. This causes, that significant dependencies can be not found on the graph of autocorrelation function. Partial autocorrelation function is another important tool. It is a modification of autocorrelation function, which allows to eliminate the described problem.

Figure 2.6 demonstrates results of autocorrelation function and partial autocorrelation function for the time series data from the previous section (Monthly international airline passenger counts from 1949 to 1960).

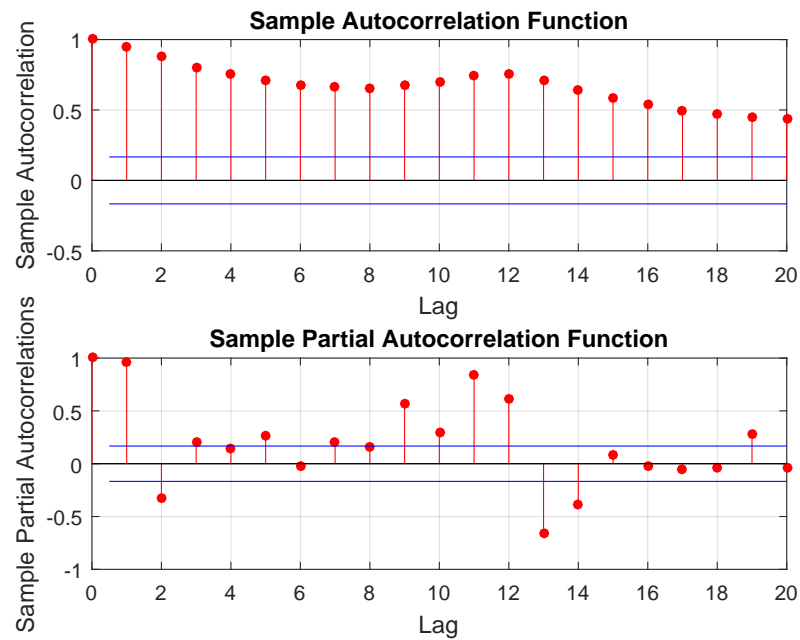


FIGURE 2.6: ACF and PACF of monthly airline passenger counts

2.6 Time series forecasting

Time series forecasting belongs to most important analysis methods, performed over the time series data. General idea is based on the fact, that information about the past events can be effectively exploited to create predictions about the future events. From the point of view of the time series data, this means, that forecasting models use already measured values to predict future values before they are observed.

When talking about the time series forecasting, it is necessary to emphasize the importance of distinction between two terms, "forecasting methods" and "forecasting models". Despite the fact, that both these terms have precisely specified meaning, in practice, they are often used mistakenly with the mixed meanings.

- Forecasting method – Denotes an algorithmic sequence of actions, that are necessary to perform, in order to obtain the time series forecasting model. Additionally, forecasting methods determines the way of quality assessment measurements.
- Forecasting model – Denotes a functional representation, that adequately describes a time series. On the basis of this forecasting model, future values of the time series are forecasted.

There are two main ways, how the time series forecasting tasks are defined. The first option is based on the computations, that use only the past values of the same time series, in order to predict the values in future. The second option allows to use not only the past values of the same time series, but also another external factors in addition, that can be useful for forecasting. In these cases, external factors are very often presented as another time series. Time series of the external factors are not obliged to have the same time step intervals, as the original time series data. Therefore, additional steps must be taken, in order to deal with this problem. It is also expected, that the external factors should have some influence on the original time series progress. For example, an intuitive external factors of energy consumption could be various meteorological indicators, like air temperature or air humidity.

2.6.1 Forecasting without external factors

Time series forecasting without external factors. If the observations of some stochastic process are available at discrete units of time $t = \{1, 2, \dots, T\}$, then the sequence of values $Z(t) = \{Z(i) \mid i \in T\} = \{Z(1), Z(2), \dots, Z(T)\}$ is denoted as a time series.

Let's assume that at the moment of time unit $-T$, it is necessary to make a forecast of $-l$ future values of the given process $Z(t)$. In other words, it is needed to determine the most probable future values for each of the time units $\{T+1, \dots, T+l\}$. Time unit $-T$ is a moment when the forecast is performed, it is usually named by term "**origin**". The parameter $-l$ is denoted as a "**leadtime**", it represents the number of future values that are going to be predicted.

In order to calculate the time series values at future time units, it is necessary to determine functional dependency that describes a relationship between past and future values of the given time series. The forecast is based on $-k$ past values, denoted as an input vector Z_T . As a result, the vector of $-l$ future predictions will be obtained, denoted as an output vector \hat{Z}_T . All predicted values $\hat{Z}(i)$ will be marked with sign $\hat{}$ in order to label them as predictions, not the real values.

$$Z_T = \begin{pmatrix} Z(T) \\ Z(T-1) \\ Z(T-2) \\ \vdots \\ Z(T-k) \end{pmatrix} \quad \hat{Z}_T = \begin{pmatrix} \hat{Z}(T+1) \\ \hat{Z}(T+2) \\ \vdots \\ \hat{Z}(T+l) \end{pmatrix} \quad (2.3)$$

$$f(Z_T) = \hat{Z}_T \quad (2.4)$$

The functional dependency (2.2) is usually denoted as forecast function and it represents the forecast model. The intuitive aim is to find the forecast function such that the deviations between predicted values and actual values, that will be observed later in future, are as small as possible.

$$\varepsilon_T = \begin{pmatrix} Z(T+1) \\ Z(T+2) \\ \vdots \\ Z(T+l) \end{pmatrix} - \begin{pmatrix} \hat{Z}(T+1) \\ \hat{Z}(T+2) \\ \vdots \\ \hat{Z}(T+l) \end{pmatrix} \quad (2.5)$$

Analysis of deviations vector (2.3) represents a basis of so called “loss function” or “error function”. This function measures the quality of forecast, based on the measured deviations. There are more options, how to calculate rate of quality from the deviations vector, usually root mean square error or mean absolute deviation are calculated. More details about error functions will be discussed in section 2.2. The formal objective of time series forecasting is then formulated as a minimization of loss function.

In addition to calculations of future values, sometimes it is required to determine accuracy limits. The accuracy of the forecasts may be expressed by calculating probability limits on either side of each forecast. These limits may be calculated for any convenient set of probabilities. They are such that the realized value of the time series, when it eventually occurs, will be included within these limits with the stated probability. [1]

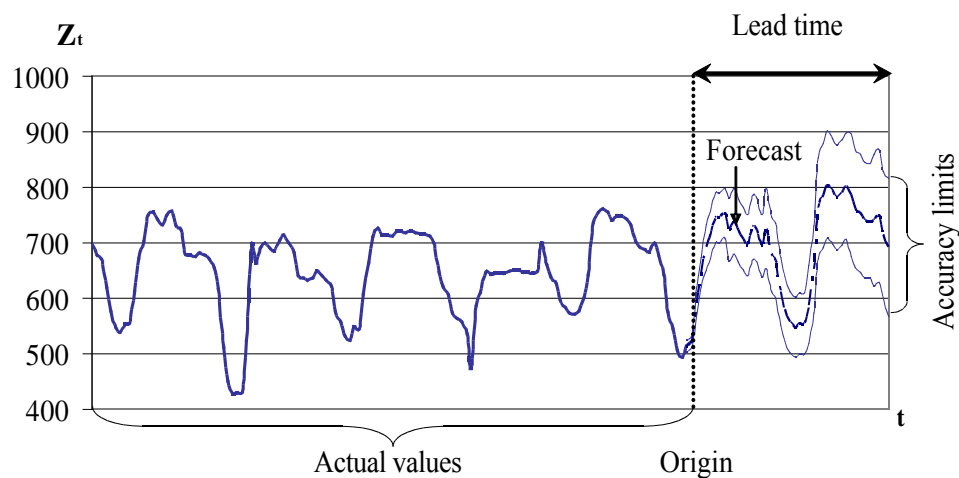


FIGURE 2.7: Time series forecasting without external factors

2.6.2 Forecasting with external factors

Time series process $Z(t)$ is specified at the discrete time units $t = \{1, 2, \dots, T\}$. It is assumed, that this time series is affected by a set of external factors $\{X_1(t_1), X_2(t_2), \dots, X_m(t_m)\}$. Each external factor is represented as an independent time series process. For example, an external factor $X_1(t_1)$ is specified at the corresponding discrete time units $t_1 = \{1, 2, \dots, T_1\}$.

The original time series $Z(t)$ and external factors $X_i(t_i)$ are not obliged to be specified at same time units. If the time units t, t_1, t_2, \dots, t_m are not equal, then it is necessary to recalculate the values of external factor to a single scale t .

Let's assume that at the moment of time unit T , it is necessary to make a forecast of $-l$ future values of the given process $Z(t)$. In order to calculate the predictions, it is necessary to determine functional dependency, that describes a relationship between past and future values, also considering the impact of external factors.

$$Z_T = \begin{pmatrix} Z(T) \\ Z(T-1) \\ Z(T-2) \\ \vdots \\ Z(T-k) \end{pmatrix} \quad X_{i,T} = \begin{pmatrix} X_i(T+l) \\ \vdots \\ X_i(T+1) \\ X_i(T) \\ X_i(T-1) \\ \vdots \\ X_i(T-k) \end{pmatrix} \quad \hat{Z}_T = \begin{pmatrix} \hat{Z}(T+l) \\ \vdots \\ \hat{Z}(T+2) \\ \hat{Z}(T+1) \end{pmatrix} \quad (2.6)$$

$$f(Z_T, X_{1,T}, X_{2,T}, \dots, X_{m,T}) = \hat{Z}_T \quad (2.7)$$

The functional dependency (2.5) is a forecast function and it represents the forecast model with external factors. The rest tasks are performed in the same way as they were in the case of forecasting without external factors. The main objective is to find the forecast function such that the deviations between predicted values and actual values, that will be observed later in future, are as small as possible. This objective formulates minimization task of so called "loss function" or "error function". More details about error functions will be discussed in section 2.2.

The accuracy limits may be calculated for any convenient set of probabilities. Accuracy limits are such that the realized value of the time series, when it eventually occurs, will be included within these limits with the stated probability. [1]

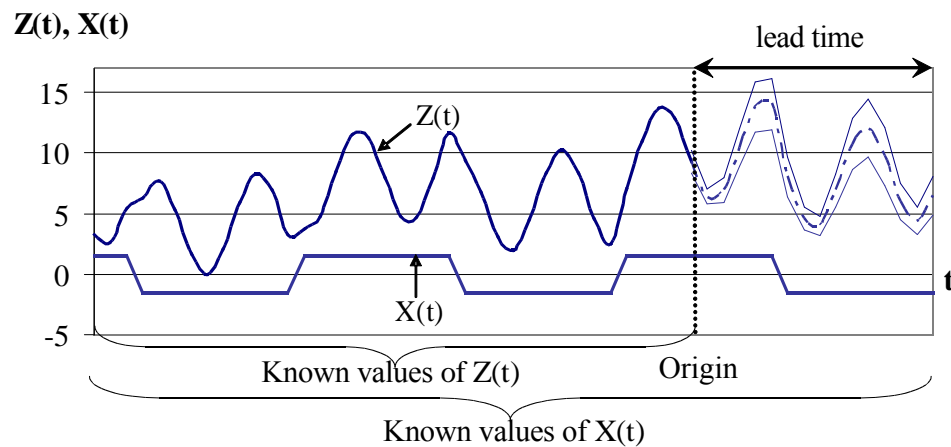


FIGURE 2.8: Time series forecasting with external factors

2.7 Data preprocessing

Before the raw time series data can be applied to the forecasting methods, usually they have to undergo several transformations. Proper data preprocessing significantly affects the forecast quality. Some forecasting methods, for example neural networks methods, have strict requirements for the format of input data. The absence of the proper data preprocessing, leads to the inefficiency of the given forecasting method.

2.7.1 Outliers detection

An outlier is an observation, that significantly differs from the other observations in the sample. In practice, very often can be observed situation, when data contain some outliers. Identification of potential outliers is very important preprocessing task, because of the following reasons:

1. Outlier may indicate mistakenly recorded data.
2. Sometimes the outlier may represent the correct data, but their presence decreases the effectiveness of the forecasting model. Therefore, their presence is undesired.

Outliers detection is usually performed by application of some appropriate filtering methods, for example "Hampel filter". [5] As soon as the outlier is detected, it can be excluded from the dataset, or replaced by the mean of its neighboring values.

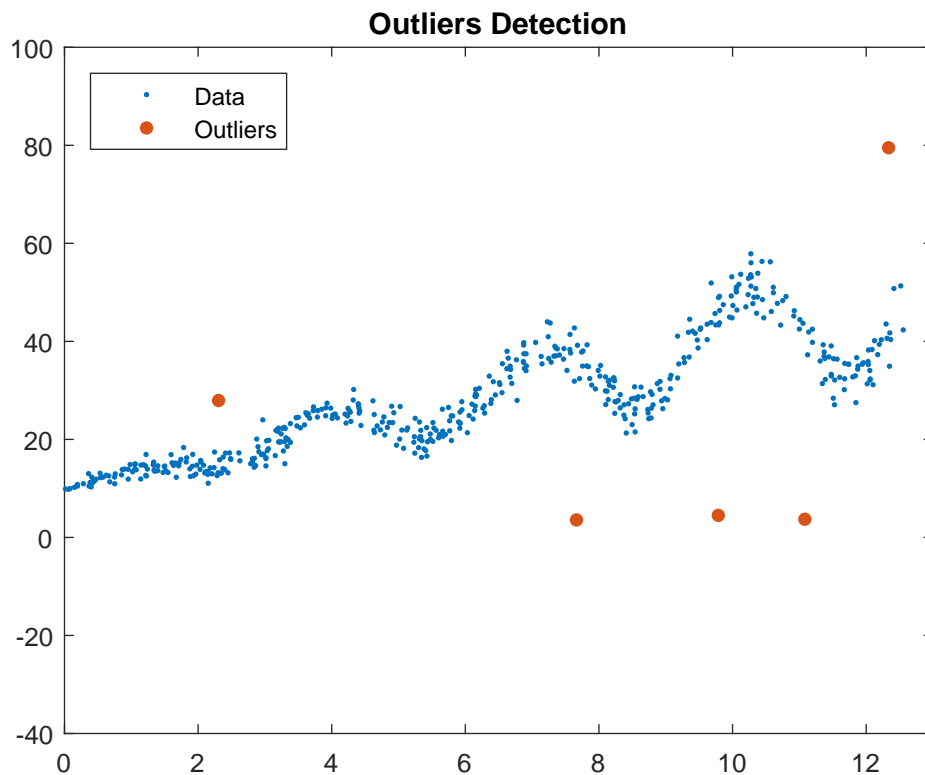


FIGURE 2.9: Outliers detection example

2.7.2 Denoising and Smoothing

Time series data almost always contain a random noise component. The purpose of denoising methods, is to filter and remove the unwanted noise. Smoothing of the processed data belongs to the most common denoising methods. Smoothing performs some kind of local averaging, which usually causes the elimination of unwanted noise signal. This can be explained by the fact, that random noise is known to be a stationary process, and stationary processes have a mean value equal to zero. Therefore, smoothing can be suitably used to remove the noise. The most popular smoothing algorithms are:

- Moving average filter - Each value in the series is replaced by the simple or weighted average of its neighboring values.
- Median filter - Similar to moving average, but values are replaced by median value.
- Local regression filter - Values are replaced by the smoothed curve with values fitted by least squares approach.

- Savitzky-Golay filter - "A generalized moving average filter with coefficients determined by an unweighted linear least-squares regression and a polynomial model of the specified degree." [6]

2.7.3 Differencing

In practice, very often happens, that it is necessary to forecast a non-stationary time series data. But the majority of forecasting methods can work only with the stationary series. There are several options, how this problem can be solved.

The most common option is differencing of the time series, which usually reduces the non-stationarity. Differencing can be performed multiple times, if there still remains some evidences of non-stationarity. Similarly the rate of relative differences can be used.

- Simple differencing:

$$D(t) = Z(t) - Z(t - 1) \quad (2.8)$$

- Relative differencing:

$$R(t) = \frac{Z(t) - Z(t - 1)}{Z(t - 1)} \quad (2.9)$$

Another option is application of logarithmic return rate. This is very similar method, it just use the logarithmic values instead of absolute values. Logarithmic return rate provides better scaling properties, which are useful if the original data contain an increasing oscillation character or exponential trend.

$$LR(t) = \log(Z(t)) - \log(Z(t - 1)) = \log\left(\frac{Z(t)}{Z(t - 1)}\right) \quad (2.10)$$

2.7.4 Scaling

Scaling is a transformation, that adjust scales of the values within some specific boundaries. The most common used scaling are transformations of values within $\langle -1, 1 \rangle$ range or $\langle 0, 1 \rangle$ range.

- Scaling range $\langle -1, 1 \rangle$

$$Z'(t) = \frac{2 \cdot Z(t) - (max + min)}{max - min} \quad (2.11)$$

- Scaling range $\langle 0, 1 \rangle$

$$Z'(t) = \frac{Z(t) - \min}{\max - \min} \quad (2.12)$$

Where $\min; \max$ corresponds to *minimum; maximum* values of the time series $Z(t)$.

2.7.5 Normalization

The general aim of normalization is an adjustment of the values by shifting and scaling, in order to obtain a so called normal distribution of the values. This produces a time series with mean property equal to 0 and standard deviation property equal to 1.

$$Z'(t) = \frac{Z(t) - \mu}{\sigma} \quad (2.13)$$

Where μ is the mean value and σ is the standard deviation of the given time series.

$$\mu = \frac{1}{n} \sum_{t=1}^n Z(t) \quad \sigma = \sqrt{\frac{1}{n} \sum_{t=1}^n (Z(t) - \mu)^2} \quad (2.14)$$

Chapter 3

Forecasting Methods

3.1 Regression models

There is a lot of tasks, which require the investigation of relationships between two and more variables. Regression analysis is a typical method, that is being used for this kind of problems. The aim of regression analysis is to estimate the dependencies between main variable and a set of external factors (regressors).

The linear regression model is the simplest and the most widely used regression model. It assumes, that there is a set of external factors $X_1(t), X_2(t), \dots, X_p(t)$, which have an impact on the given process $Z(t)$ and the relationship between them is linear. Forecasting model based on the linear regression is determined by an equation (2.12).

$$Z(t) = \alpha_0 + \alpha_1 X_1(t) + \alpha_2 X_2(t) + \dots + \alpha_p X_p(t) + \varepsilon_t \quad (3.1)$$

Where $\alpha_i, i = 0 \dots p$ are regression coefficients (parameters), ε is the approximation error. In order to obtain a forecasted values $Z(t)$ at time units t , it is necessary to have values $X_i(t)$ at time moment t , sometimes in practice this can be impossible in some kind of problems.

The nonlinear regression models are based on assumptions, that there is given a mathematical function, that describes relationship between given process $Z(t)$ and the external factor $X(t)$.

$$Z(t) = f(X(t), \alpha) + \varepsilon_t \quad (3.2)$$

While constructing the forecast model, it is necessary to determine the function parameters α . For example, $Z(t)$ dependency on $\sin(X(t))$

$$Z(t) = \alpha_1 \sin(X(t)) + \alpha_0 + \varepsilon_t \quad (3.3)$$

In order to construct this model it is sufficient only to determine the parameters $\alpha = (\alpha_0, \alpha_1)$. However in practice it is not very common, that type of functional dependency between process $Z(t)$ and external factor $X(t)$ is already known in advance. Therefore, nonlinear regression models are used less frequently, than the linear ones.

3.2 Autoregressive and moving average models

Autoregressive models are based on the idea, that values of process $Z(t)$ are linearly dependent on some number of past values of the same process $Z(t)$. In this model, the actual value of the process is expressed as a sum of finite linear combination of previous values and the impulses, called white noise.

$$Z(t) = c + \varphi_1.Z(t-1) + \varphi_2.Z(t-2) + \dots + \varphi_p.Z(t-p) + \varepsilon_t \quad (3.4)$$

where φ_i are parameters of the model; c is a constant; ε is white noise (error of the model).

The formula describes the autoregressive model of order p . This model is often denoted as $AR(p)$. The parameters c and φ_i are usually estimated by mean least squares or maximum likelihood methods.

The second model, moving average model. It plays very important role in time series description and is frequently used in relation with the autoregressive models. Moving average model of order q is described by formula:

$$Z(t) = \frac{1}{q}[Z(t-1) + Z(t-2) + \dots + Z(t-q)] + \varepsilon_t \quad (3.5)$$

where q is order of moving average and ε_t is prediction error.

In the books, moving average model of order q is usually denoted as $MA(q)$. Actually, moving average model is a finite impulse response filter applied to white noise.

In order to achieve better prediction quality, two previous models are often merged into one model, autoregressive and moving average model. Common model is denoted

as $ARMA(p, q)$ and it unites a moving average filter of order q and autoregression of filtered values of order p .

If the time series data show evidence of non-stationarity, then the initial differencing step can be applied to reduce the non-stationarity. This model is usually denoted as $ARIMA(p, d, q)$. The parameter d represents the degree of differencing, it corresponds to the *integrated* part of the model.

Another option is an $ARIMAX(p, d, q)$ model, that is an extension of $ARIMA(p, d, q)$ model. It is described by formula:

$$Z(t) = AR(p) + \alpha_1 X_1(t) + \dots + \alpha_S X_S(t) \quad (3.6)$$

This model is extended by the impact of external factors. In this model, the process $Z(t)$ is a result of model $MA(q)$, that are filtered values of the original process. Subsequently autoregressive forecasting, with additional regression parameters, corresponding to external factors, is performed.

3.3 Exponential smoothing models

Despite the fact, that Exponential smoothing methods were invented in the middle of 20th century, they are still frequently used, even today. Exponential smoothing models are widely used for modeling finance and economical processes. The basis of exponential smoothing, is an idea of repetitive revision of forecasting function, with each income of newly observed value. Exponential smoothing model assigns exponentially decreasing weights to past values, according to the age. Therefore, newly observed values have higher impact on forecasted value, than the elder ones. Functional representation of exponential smoothing model is expressed by the following equations:

$$Z(t) = S(t) + \varepsilon_t \quad (3.7)$$

$$S(t) = \alpha \cdot Z(t-1) + (1-\alpha) \cdot S(t-1) \quad (3.8)$$

$$S(1) = Z(0) \quad (3.9)$$

where $Z(t)$ is an actual value of the time series observed at time unit t ; $S(t)$ is a smoothed value at time t ; ε_t is an error between actual and smoothed value; α is a smoothing coefficient, $0 < \alpha < 1$. In this model, each subsequently smoothed value $S(t)$ is a weighted combination of previous time series value $Z(t - 1)$ and previously smoothed value $S(t - 1)$.

3.3.1 Double exponential smoothing

Double exponential smoothing, sometimes referred as "Holt-Winters double exponential smoothing" is an improved modification of simple exponential smoothing. This model is usually used for processes, which contain a trend component. In comparison to the simple exponential smoothing, in these cases, it is necessary to deal with additional smoothing coefficient related to trend component. The model is described by the following equations.

$$S(t) = \alpha \cdot Z(t) + (1 - \alpha) \cdot (S(t - 1) + B(t - 1)) \quad (3.10)$$

$$B(t) = \beta \cdot (S(t) - S(t - 1)) + (1 - \beta) \cdot B(t - 1) \quad (3.11)$$

$$S(1) = Z(1)B(1) = Z(1) - Z(0) \quad (3.12)$$

where $Z(t)$ is an actual value of the time series observed at time unit t ; $S(t)$ is a smoothed value at time t ; α is the data smoothing coefficient, $0 < \alpha < 1$; β is the trend smoothing coefficient, $0 < \beta < 1$.

Forecasting with double exponential smoothing

In order to obtain a forecasting model based on exponential smoothing, it is necessary to have some specific amount of historical values of the given time series. The model is being built, by solving an optimization task, which consists of finding the appropriate values of α and β parameters, such that MSE of the smoothed curve is minimal. As soon as the optimal values for parameters are estimated and the model is created, the forecasting of future values can be performed according to the following equations:

$$F(t + 1) = S(t) + B(t) \quad (3.13)$$

$$F(t + m) = S(t) + m \cdot B(t) \quad (3.14)$$

3.4 Artificial neural networks models

In the past few years, there can be observed a great interest in machine learning, especially in artificial neural networks sector. Artificial neural networks are tools, that are being used today for solving huge amount of tasks from different areas. The most frequent examples are time series forecasting, pattern recognition, data clustering and classification. Such a great success is determined by several reasons.

1. Artificial neural networks represent exclusively powerful tool, that enables to reproduce very complex nonlinear dependencies. For many years linear models played the leading role in the most areas, as there were a lot of well designed and optimized tools, which satisfactorily coped with assigned tasks, but problem was with tasks, for which the linear approximation is unsatisfactorily.
2. Artificial neural networks are learning from examples. Artificial neural networks receives a set of representative examples and then a learning process starts, which tries to find and extract the structure of data. Certainly, proper application of artificial neural network demands specific requirements for a correct formulation of representative data set and network's architecture. However, proper construction of a such artificial neural network allows to cope with tasks, which can be solved by the traditional algorithms only with the great difficulties. For example, pattern recognition task, practically used for face recognition, solving it in traditional way would result in a very complex problem. However, the same problem can be prospectively solved by the artificial neural networks. [7]

3.4.1 Biological inspiration

Artificial neural networks are results of researches in the field of Artificial intelligence. Human brain is known to be able to deal with the problems much more complex, than the computers solve. It consist of huge number of neurons connected with each other by numerous connections. Neurons are specific nerve cells, belonging to the nervous system, that are able to distribute electrical or chemical signals. Neuron cell has a branched structure consisting of the three main parts: information inputs - dendrites, information output - axon, and the nucleus. The axon branches of the cell are connected to the dendrites of other cells with the connections called synapses. Dendrites of the

neuron receive electrical signals from other neurons through the synapses. If the total rate of the input signals, received by the dendrites of the neuron, exceeds the determined threshold, then the given neuron is going to fire an action potential. It is short-lasting process, during which the neuron sends signals to its neighbors, which also may fire. The intensity of transferred signal strongly depends on the activity of synapse between two neurons. The process of learning basically stands for an appropriate changes of the activities of the synapses connections between neurons. [8]

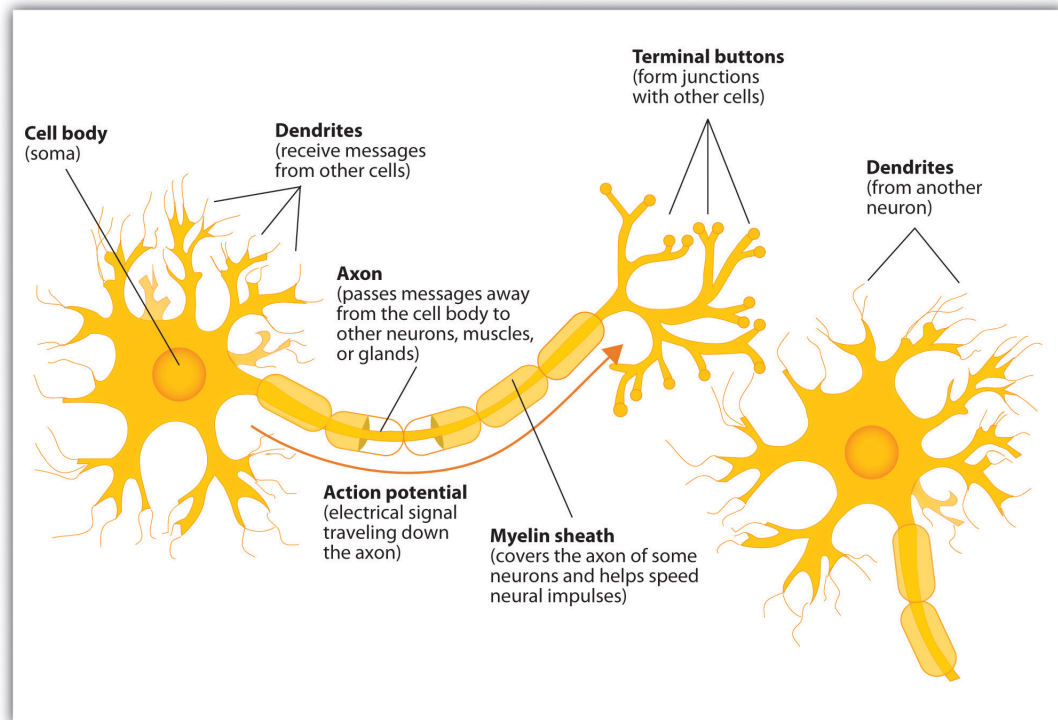


FIGURE 3.1: Biological neuron

3.4.2 Artificial neuron model

Artificial neuron represents a simplified model of the natural nervous cell. The evolution of artificial neurons contains several models, which have passed certain stages of development. Today, the most common artificial neuron is usually referred to the following model, determined by the three main components:

1. The set of synapses - Connecting links, each of which is characterized by its own weight. These weights correspond to the activities of synapses in biological neuron. The input signal x_j , that passes through the synapse j , which belongs to the neuron k , is multiplied by the weight w_{kj} .

2. The adder - Component, which calculates the weighted sum of signals, i.e. the linear combination. Additionally, for each neuron there is defined a threshold value b_k , denoted as "bias", which is added (or subtracted) to the weighted sum of signals. Obtained result is usually denoted as "induced local field" or "activation potential", depending on the value of b_k .
3. Activation function - Output obtained from the adder component, is passed further to the activation function. Activation function transforms the input and produces the output y_k , referred as output of neuron. [7]

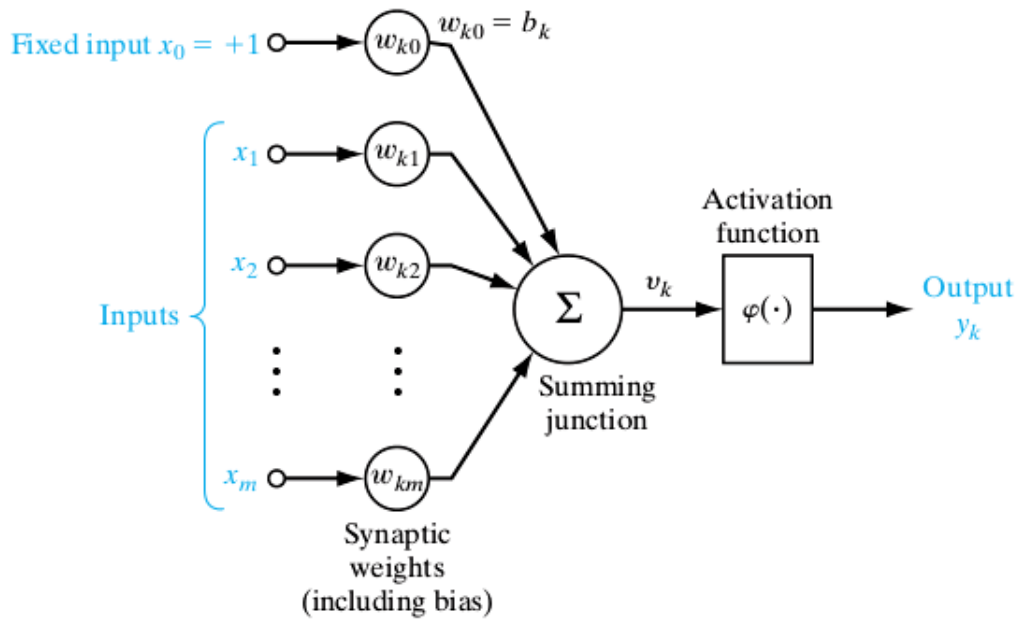


FIGURE 3.2: Artificial neuron model

In mathematical terms, artificial neuron depicted by figure 3.2 may be described by the following equations.

$$v_k = \sum_{j=0}^m w_{kj} \cdot x(j) \quad (3.15)$$

$$y_k = \varphi(v_k) \quad (3.16)$$

where $x_0 = 1$ and x_1, x_2, \dots, x_m are the input signals; $w_{k1}, w_{k2}, \dots, w_{km}$ are the respective synaptic weights of neuron k ; b_k is the bias; v_k is the "induced local field" or "activation potential"; $\varphi(\cdot)$ is the activation function; y_k is the output signal of the neuron. [7]

3.4.3 Types of Activation Function

The activation function $\varphi(v)$, defines the output of a neuron. There is a lot of suitable functions, that can be exploited as the activation function in artificial neurons. Appropriate selection of activation function strictly depends on the format of input and output values, and the task expected to be performed by a neural network. It is also important to mention, that the activation functions of individual neurons are not obliged to be identical, there can be easily used different activation functions inside one neural network.

The most popular activation functions [9]:

1. Threshold function - Sometimes called binary step function. Today, in practice, this activation function is used rarely. More often, it demonstrates original inspiration by the biological neuron.
2. Sigmoid function - Frequently used function, when output values are scaled in $[0;1]$ range.
3. Hyperbolic tangent function - Similar to sigmoid function. Output values are in $[-1;1]$ range.
4. Identity function
5. ReLU - In recent years, ReLU is becoming very popular.


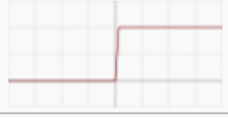
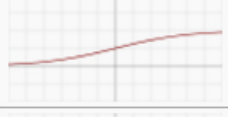
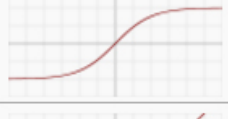
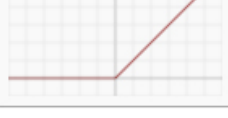
Identity		$f(x) = x$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Logistic (a.k.a. Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$
Tanh		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$

FIGURE 3.3: Activation functions

3.4.4 Neural Network Architectures

In the previous sections, there were described just the actions inside one artificial neuron. Now the main question is, how to connect the individual neurons to each other? In theory, neurons may be connected into neural networks with the very diverse structures. However, in practice, artificial neurons are usually grouped into layers, that later formulate a neural network.

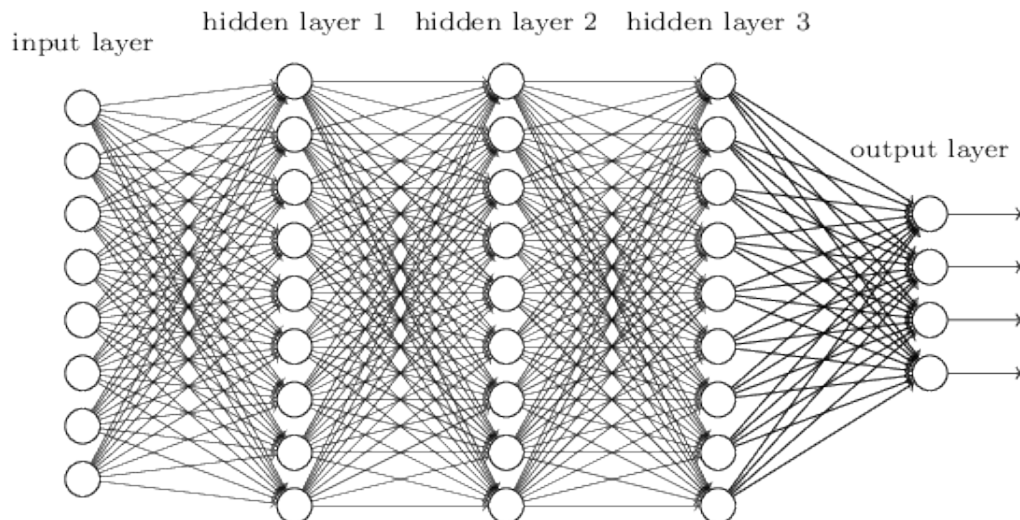


FIGURE 3.4: Artificial neural network example

Figure 3.4 demonstrates an example of neural network with one input layer, three hidden layers and one output layer. Actually, the input layer is not a real layer. It just represents the number of input values passed to the neural network. However, in the books, it is very often graphically demonstrated as the first layer of the network. All others are real layers, in sense of previously described rules. Each node in the hidden or output layer, represents a neuron. The arrows between neurons represent connections between them, and indicate the direction of signal processing. Any signal inside the network is eventually directed to the output layer, which represents an overall output of the network. All layers between the input and output layers, are called hidden layers. The name "hidden" is related to the fact, that neural network acts like a black box, and all communication with network is performed through the input and output layers, and everything, that happens inside, remains invisible to the user.

Generally, there are two main types of artificial neural networks structures:

1. Feedforward neural networks - Unites a group of networks, where the signal is passed strictly in one direction from the input layer to the output layers (Figure 3.4). Assumption is, that there is no cycles inside the network.

2. Recurrent neural networks - Represents a group of networks, which contain at least one cycle inside the network. The cycle inside the neural network means, that the output signal of some neuron, passing through the certain sequence of connections, may occur as the input to the neuron, that it has already reached. (Figure 3.5)

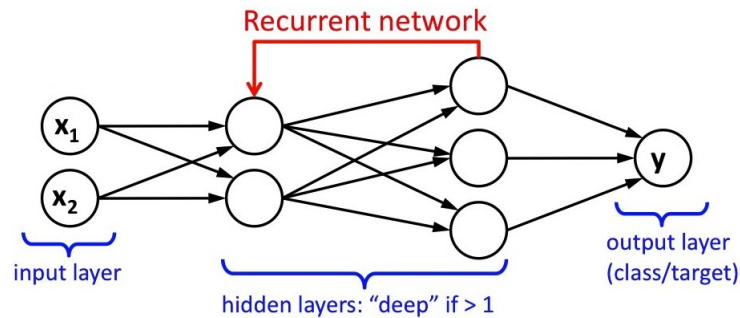


FIGURE 3.5: Artificial neural network example

Feedforward neural networks are used more frequently than recurrent networks, in part because the learning algorithms for recurrent networks are less powerful. Nevertheless, recurrent networks are still very popular. They are much closer to the biological neural networks and the idea how human brain works. Recurrent networks may be used to solve important problems, which can only be solved with great difficulty by feedforward networks. [10]

Additionally, artificial neural networks are classified as "deep" networks, if the number of hidden layers is greater than one. (Figure 3.4) [10]

3.4.5 Appropriate architecture

Selecting an appropriate architecture of neural network is an important step. When selecting an architecture, it is necessary to deal with following parameters:

- Number of neurons in input layer - Usually, number of neurons in input layer directly depends on the format of the input data. For example, if the neural network will be used for time series forecasting, the number of input neurons will correspond to the number historical values used for forecasting. If the neural network will be used for images classification, the number of input neurons will correspond the number of pixels of the images.
- Number of neurons in output layer - Similarly as with the number of input neurons, the number of output neurons directly depends on the performed task and the amount of output information. For example, if the neural network is used time

series forecasting, the number of output neurons will correspond to the number of forecasted values. If the neural network is used for classification of images with the handwritten number, the number of output neurons can be 10, each neuron for one number (class of images).

- Number of hidden layers - In mathematical theory, neural network with at least one hidden layer, is sufficient to approximate or learn dependencies of any non-linear function. Despite this, for many tasks it is much more suitable to use a neural network with more than one hidden layer. For more complex tasks, like images classification, are usually used deep neural networks with much more than one hidden layer. On the other hand, tasks, which do not contain so complex dependencies, they also do not require so complicated structures, as it will just lead to overfitting and decrease the performance. [10]
- Number of neurons in hidden layer - This parameter is also very sensitive to overfitting. Usually, there is no regular rule, how to choose the number neurons in hidden layer. There exist some recommendations, but the most reliable solution leads to the benchmarking. [7]

3.4.6 Networks training

Architecture selection is just the first step. After the neural network is constructed, it is still not ready to be exploited. During the initialization, the weights of connections between neurons are selected randomly. Before the neural network can be adequately used for required task, proper weights have to be found. This process is usually referred as a learning or training of the neural network.

There exist different learning algorithms, each of them is suitable for the specific network architecture. Backpropagation algorithm is one of the most popular training algorithms. It is very effective algorithm, but it can be used for training networks with at most one hidden layer. The majority of tasks can be easily solved by neural networks with one hidden layer, therefore backpropagation algorithm is suitable for these cases. In the case of deep networks, backpropagation leads to the vanishing gradient problem, and makes it impossible to use. [10]

3.4.7 Cross-validation

Before the learning process can be launched, it is necessary to perform data partitioning. Data are divided into three sets: training set, validation set and testing set. Usually, training set is the largest and it contain the data, which will be used for network training.

Validation set is used to deal with the overfitting problem. Overfitting is a common problem, which may occur when it is required to fit a model to the training data. After some moment the model perfectly fits the training set, but it will have low performance on the newly observed data.

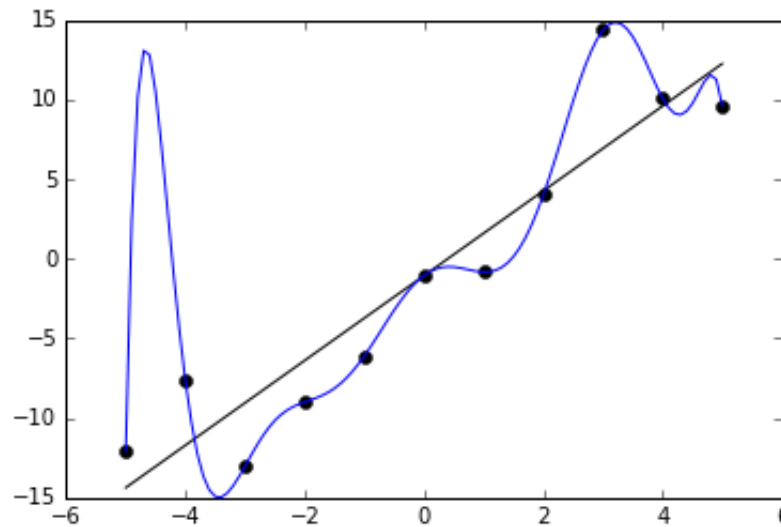


FIGURE 3.6: Overfitting example

In order to deal with the overfitting problem, the validation set is used. Neural network is trained on the training data, but the error is calculated for the validation set. Training is performed up to the moment when the error for validation set starts to increase.

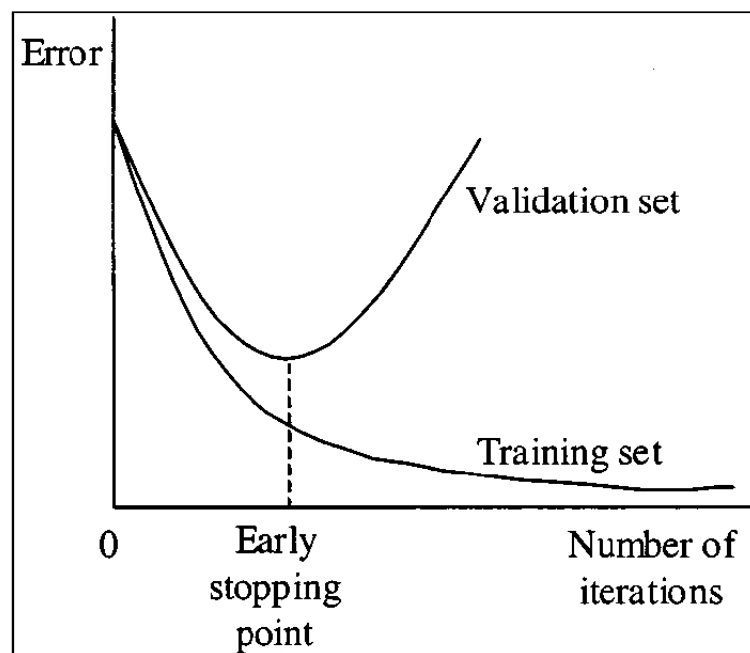


FIGURE 3.7: Cross-validation

Testing set doesn't participate in network training. It is just used to express the performance of network on the independent data.

3.4.8 ANN forecasting model

Artificial neural networks allow to create very powerful forecasting models. Its ability to deal with the non-linear dependencies gives a great advantage, in comparison to other forecasting methods. Before the time series data can be applied to the neural network, it is necessary to "cut" the data on the samples of the specific length, which corresponds to the number of neurons in the input layer. As well, it is required to prepare the target samples, what corresponds to the forecasted values.

As soon as the neural network is constructed and successfully trained, it represents a forecasting model and can be used for time series forecasting, as any other model.

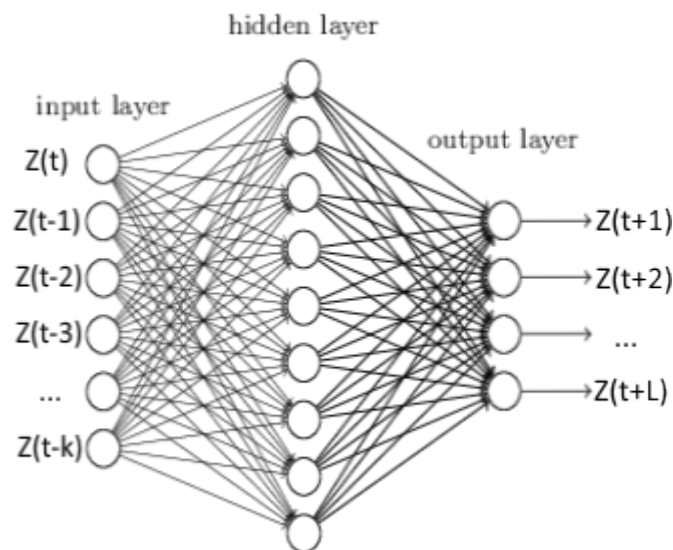


FIGURE 3.8: ANN and time series forecasting

3.5 Markov chain models

Forecasting models based on the Markov chains assume, that future state of the process is dependent only on its current state and is not dependent on its elder states. Markov chain models are applicable on the short-memory time series. Example of Markov chain for process with 3 states is illustrated on figure 1.3.

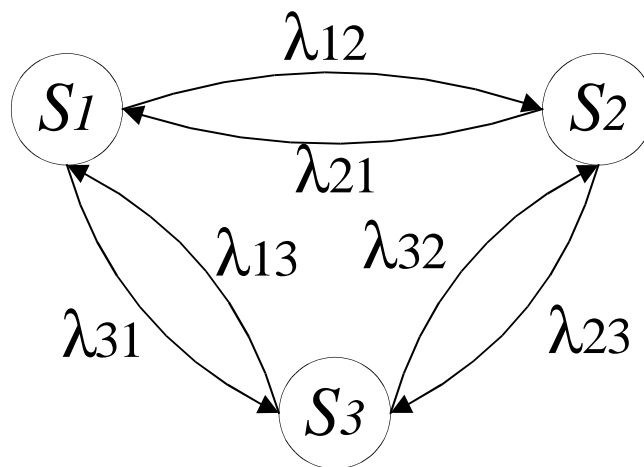


FIGURE 3.9: Markov chain model

In this model, $S1, S2, S3$ - are states of process $Z(t)$, α_{xy} - probability of transition from state x to state y . By building the Markov chain model, the set of states and corresponding transitions' probabilities are defined. If the current process state is defined, the future state is selected as the state with maximal transition probability. If the transition probabilities are properly stored in matrix, subsequent future values can be determined by probability matrix's multiplication and maximum probability selection.

3.6 Forecasting models comparison

Forecasting Model and Method	Advantages	Disadvantages
Regression models	The main advantages of the given models are: simplicity, flexibility and uniformity of calculations. Simplicity of model construction (only linear models). Transparency of all intermediate calculations.	Inefficiency and low adaptability of linear regression models for non-linear processes. Very complex non-linear model construction for the tasks with non-linear functional dependency.

Autoregressive and Moving average models	Transparency and uniformity of calculations and model's construction. Relatively not complicated model construction. The most popular frequently used forecasting method. A lot of publications and information about how to apply this method for the specific problems.	Large number of parameters required to be determined. Linearity, low adaptability and inefficiency with non-linear processes.
Artificial Neural Networks models	The main advantage of these models is a non-linearity. Neural networks can easily deal with the non-linear dependencies between future and past values of the processes. Great adaptability and scalability. Ability of parallel computations.	Large number of parameters and significant options necessary to be selected. High hardware performance requirements during the network training process. Complexity of architecture and absence of transparency.
Exponential smoothing models and methods	Transparency of intermediate calculations, simplicity and relative effectiveness. Easy model construction.	The disadvantage of this model is inflexibility.
Markov chains models	Transparency of intermediate calculations.	Impossibility of long term forecasting.

Part II

Practical part

Chapter 4

Implementation

4.1 Data science tools

Today, the amount of data science tools has significantly increased. When analyzing different researches, there can be observed three main tools, that are being used the most frequently. In this section there will be briefly introduced each of these tools and described their abilities for time series analysis.

4.1.1 The R language

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. The trend shows a significant increase in popularity of this tool. Initially, it wasn't developed specifically for time series analyzing and it didn't contained all required tools for time series forecasting. But its main advantage is based on the fact, that it is open source. Today (Jan 2017), there is around 10000 different packages contributed to its package repository. There are available all required tools for time series analysis, including filtering, decomposition, preprocessing and also time series forecasting tools including autoregressive methods. Another packages also enable to exploit neural networks computations. Generally, the R language demonstrates promising future development, and therefore it is one of the top rated data science tools.

4.1.2 MATLAB

MATLAB is a numerical computing environment and programming language, maintained by the MathWorks. MATLAB provides powerful matrix calculations with perfectly optimized algorithms and great abilities of parallel computations. The base MATLAB environment contains a lot of implemented functions, but for the more sophisticated projects, there should be involved additional toolboxes, which contain special functions for the given problematic. In the case of time series analysis, the base environment provides sufficient abilities for work with data in time series format. However, there can be found additional functionalities in the following toolboxes: Curve Fitting Toolbox, Econometrics Toolbox, Signal Processing Toolbox, Financial Instruments Toolbox, Financial Toolbox, Optimization Toolbox. There are also toolboxes for machine learning computations: Neural Network Toolbox, Statistics and Machine Learning Toolbox. Generally, MATLAB is a great data science tool, but its main disadvantage is higher price for licenses.

4.1.3 Python

Python is a widely used high-level programming language. Initially, Python was not developed for mathematical computations. It just contained typical mathematical functions, like any other programming language. Python is an open-source project and its main advantage is based on its third-party modules, which provide wide opportunities for different areas. In the case of neural networks, especially deep networks, Python provides a top rated libraries. Libraries like Theano and its high level modifications like Keras or Lasagne, enables well optimized work with deep neural networks, especially optimized computations on GPUs. There also exist many other libraries, which provide statistical and machine learning computations with the time series: scikit-learn, numpy, pandas, matplotlib.

4.2 Experiments definition

The main task of the practical part of the work is the implementation of various forecasting models, as well as the estimation of their performance measures on different datasets. To this purpose, there have been selected the following datasets.

- Forecasting of Internet traffic data - representing the information technology sector
- Household electricity load - representing energetic sector

- IBM daily stock prices - representing econometric sector

All experiments were performed in the computing environment MATLAB. All source code used in this work is available in the attachment of the thesis. For the purposes of thesis, there have been selected three the most promising forecasting methods.

1. ARIMA - According to the various researches, today, it is the most frequently used forecasting method.
2. Artificial Neural Networks method - Very promising machine learning approach.
3. Exponential smoothing - Despite the fact, that it is not so powerful as the previous methods, it still belongs to the frequently used methods.

ARIMA function is already implemented in MATALB, so it is not required to implement it from scratch. Separately, for the purposes of thesis, there were implemented tools for finding the parameters of the model with the best performance, based on MSE measure.

In the case of neural networks, MATLAB provides a toolbox for their realization. For the purposes of given task, it was decided to use feedforward neural networks, with varying number of neurons in the individual layers.

At the moment of writing the thesis, there was no function for realization of exponential smoothing in MATLAB. Therefore, for the purposes of thesis, there has been implemented double exponential smoothing forecasting method, described in section (3.3.1).

Each of the experiments will be performed by fulfilling the following steps:

1. Data preprocessing - Each dataset will undergo required preprocessing steps in order to prepare data for all three forecasting methods.
2. Data scaling - Additionally to the preprocessing, dataset values will be scaled into the same range, in order to enable better comparison of experiments.
3. Data partitioning - Each dataset will be divided into the training set (70%) and validation set (30%). Training set will be used for building the forecasting models, and validation set will be used for performance measuring.
4. Forecasting - Each forecasting method will be used for building models with varying number of input values as well as varying number of predicted values.

According to the assignment, each experiment will be performed twice, in order to demonstrate the dependency between forecasting accuracy and the size of training set. First time, only 50% of the training set will be used for models creation, and second time, an extended training set (100%) will be used.

4.3 Forecasting Accuracy

Forecasting accuracy is a measure, which expresses performance of forecasting model. It is a reverse value to the measure of forecasting error. There are more options, how to calculate the measure of forecasting error. Each of them expresses a little bit different information. At the beginning, it is necessary to define the forecast error. It is expressed as a deviation of predicted value and actual value:

$$\varepsilon(t) = Z(t) - \hat{Z}(t) \quad (4.1)$$

- Mean absolute percentage error (MAPE)

$$MAPE = \frac{1}{N} \sum_{t=1}^N \frac{|Z(t) - \hat{Z}(t)|}{Z(t)} \cdot 100\% \quad (4.2)$$

- Root Mean squared error (RMSE)

$$RMSE = \sqrt{\frac{1}{N} \sum_{t=1}^N (Z(t) - \hat{Z}(t))^2} \quad (4.3)$$

- Mean squared error (MSE)

$$MSE = \frac{1}{N} \sum_{t=1}^N (Z(t) - \hat{Z}(t))^2 \quad (4.4)$$

- Mean absolute error (MAE)

$$MAE = \frac{1}{N} \sum_{t=1}^N |Z(t) - \hat{Z}(t)| \quad (4.5)$$

- Sum of squared errors (SSE)

$$SSE = \sum_{t=1}^N (Z(t) - \hat{Z}(t))^2 \quad (4.6)$$

The suitability of MSE, RMSE, MAE and SSE measures is quite similar. They differ only a little bit, for example strong errors are penalized by RMSE less than by other measures. MAE and RMSE represent a scale dependent measure, while others are not scale dependent. All these measures are suitable for comparison of different forecasting methods on the same test data.

MAPE is one of the most frequently used forecasting error measures. It expresses the percentage error, what makes it easily understandable. It is a suitable measure for comparing the performance of one forecasting method on different testing data. But it has one significant shortcoming, it can be used only for time series with values much greater than 1. Otherwise, if the actual value of the series is close to 0, then a denominator will contain a very small number, what will make MAPE measure close to infinity. This will not express a correct performance.

To deal with this problem there can be used MSE with a little modification. MSE will be divided by the variance of actual values and expressed in percents.

$$MSE [\%] = \frac{MSE}{Var(Z)} \cdot 100\% \quad (4.7)$$

Chapter 5

Experiment 1

5.1 Data description

In Experiment 1 will be used Internet traffic data (in bits) from an ISP. Data represent an aggregated traffic in the United Kingdom academic network backbone. Dataset was collected between 19 November 2004, at 09:30 hours and 27 January 2005, at 11:11 hours. Values are collected at five minute intervals. Overall number of values in the dataset is 19888, what represents relatively sufficient number of observations for creating forecasting models.

Source: <https://datamarket.com/data/set/232g/>

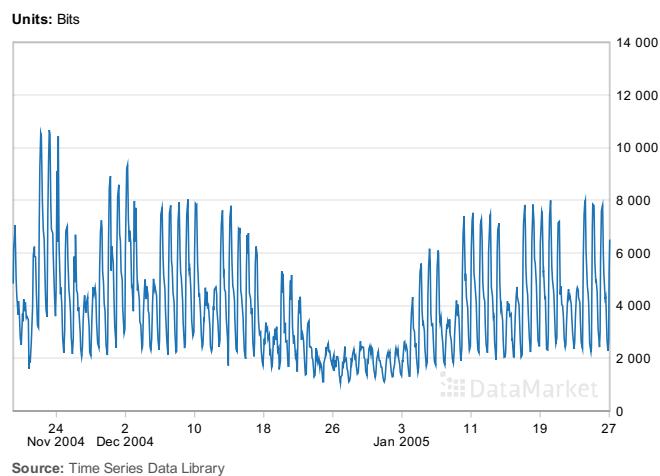


FIGURE 5.1: Internet traffic data (in bits)

From the visual overview of the plotted graph it can be easily observed, that given time series data have clear evidence of non-stationarity. The oscillations with increasing

amplitude, as well as slowly decreasing ACF function tells about confident evidence of the non-stationarity. Therefore, specific data preprocessing will be required.

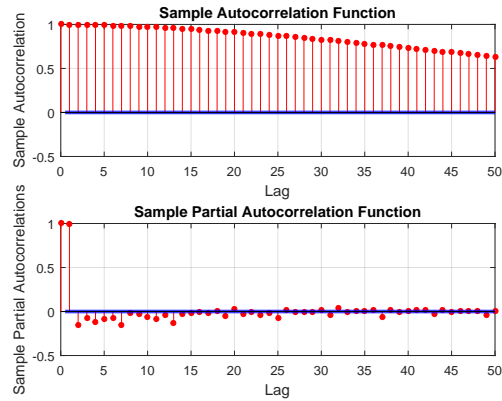


FIGURE 5.2: ACF and PACF of Internet traffic data

5.2 Data preprocessing

The following steps of data preprocessing have been performed:

1. Data transformation by $\log(x)$ function.
2. Differentiation of the first order.
3. Outliers removal using Hampel filter.
4. Denoising by using local regression.
5. Scaling values into range $\langle -1, 1 \rangle$.
6. Data partitioning.

5.3 Forecasting results Part 1.

In this section are demonstrated results of the first part of the experiment 1. In this case only 50% of the training data have been used for construction of individual forecasting models.

Double exponential smoothing models are defined as $DEV[x, y]$, where x denotes the number of input values (actual, already measured values), and y denotes number forecasted values.

ARIMA models are defined as $ARIMA(p, d, q)[x, y]$, where p denotes autoregressive parameter, d denotes differencing parameter, q denotes moving average parameter, x denotes the number of input values, and y denotes number of forecasted values

Neural Networks models are defined as $NN[x, h, y]$, where x denotes number of input neurons, h denotes number of neurons in hidden layer, and y denotes number of output neurons, as well as the number of forecasted values.

Model	MSE	RMSE	MSE [%]
ARIMA(20,0,3)[20;5]	0.00146	0.03821	2.0036%
NN[20;30;5]	0.0011685	0.034183	1.6036%
DES[20;5]	0.014928	0.12218	23.0513%
ARIMA(30,0,3)[30;5]	0.0013957	0.037359	1.9153%
NN[30;40;5]	0.0011166	0.033415	1.5323%
DES[30;5]	0.013797	0.11746	21.305%
ARIMA(40,0,3)[40;5]	0.0013202	0.036335	1.8118%
NN[40;50;5]	0.0010569	0.03251	1.4504%
DES[40;5]	0.013571	0.11649	20.9558%
ARIMA(20,0,3)[20;10]	0.010306	0.10152	14.088%
NN[20;30;10]	0.0082477	0.090817	11.2746%
DES[20;10]	0.045237	0.21269	69.8525%
ARIMA(30,0,3)[30;10]	0.010446	0.1022	14.2793%
NN[30;40;10]	0.0083591	0.091428	11.427%
DES[30;10]	0.044332	0.21055	68.4555%
ARIMA(40,0,3)[40;10]	0.0098256	0.099124	13.4317%
NN[40;50;10]	0.007862	0.088668	10.7475%
DES[40;10]	0.040713	0.20177	62.8673%
ARIMA(20,0,3)[20;15]	0.022708	0.15069	30.9259%
NN[20;30;15]	0.018182	0.13484	24.7629%
DES[20;15]	0.090473	0.30079	139.7051%
ARIMA(30,0,3)[30;15]	0.02342	0.15304	31.8968%
NN[30;40;15]	0.018751	0.13693	25.5373%
DES[30;15]	0.084068	0.28995	129.8149%
ARIMA(40,0,3)[40;15]	0.021987	0.14828	29.9445%
NN[40;50;15]	0.017606	0.13269	23.9784%
DES[40;15]	0.079181	0.28139	122.2687%

TABLE 5.1: Results of Experiment 1 - part 1

5.4 Forecasting results Part 2. - Extended Training set

In this section are demonstrated results of the second part of the experiment 1. Training set has been extended (100%), in order to improve the forecasting performance and demonstrate the dependency between forecasting accuracy and the size of the training set.

Model	MSE	RMSE	MSE [%]
ARIMA(20,0,3)[20;5]	0.0013272	0.036431	1.8214%
NN[20;30;5]	0.0010622	0.032591	1.4577%
DES[20;5]	0.013824	0.11757	21.346%
ARIMA(30,0,3)[30;5]	0.0012688	0.03562	1.7412%
NN[30;40;5]	0.0010156	0.031869	1.3938%
DES[30;5]	0.012767	0.11299	19.714%
ARIMA(40,0,3)[40;5]	0.0012002	0.034644	1.6471%
NN[40;50;5]	0.00096033	0.030989	1.3179%
DES[40;5]	0.01131	0.10635	17.464%
ARIMA(20,0,3)[20;10]	0.0093688	0.096793	12.8073%
NN[20;30;10]	0.0074952	0.086575	10.246%
DES[20;10]	0.044106	0.21001	68.107%
ARIMA(30,0,3)[30;10]	0.009496	0.097447	12.9811%
NN[30;40;10]	0.007601	0.087184	10.3906%
DES[30;10]	0.04312	0.20765	66.5849%
ARIMA(40,0,3)[40;10]	0.0089323	0.094511	12.2106%
NN[40;50;10]	0.0071472	0.084541	9.7703%
DES[40;10]	0.039668	0.19917	61.2543%
ARIMA(20,0,3)[20;15]	0.020643	0.14368	28.1145%
NN[20;30;15]	0.016527	0.12856	22.5079%
DES[20;15]	0.086105	0.29344	132.9594%
ARIMA(30,0,3)[30;15]	0.021291	0.14592	28.9971%
NN[30;40;15]	0.01704	0.13054	23.207%
DES[30;15]	0.081666	0.28577	126.1055%
ARIMA(40,0,3)[40;15]	0.019988	0.14138	27.2223%
NN[40;50;15]	0.016006	0.12652	21.7994%
DES[40;15]	0.075002	0.27387	115.8156%

TABLE 5.2: Results of Experiment 1 - part 2

Chapter 6

Experiment 2

6.1 Data description

In Experiment 2 will be used electricity load data (in kW for each 15 min interval). Data are obtained from "Center for Machine Learning and Intelligent Systems". Data represent electricity load of one anonymou household. Dataset was collected between 1 January 2012, and 31 December 2014. Values are collected at 15 minute intervals. Overall number of values in the dataset is 70080, what represents sufficient number of observations for creating forecasting models.

Source: <https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014>

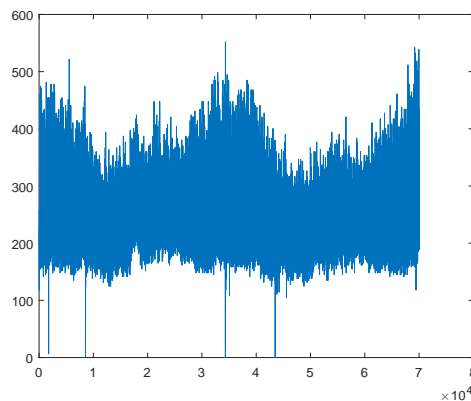


FIGURE 6.1: Electricity Load (in kW for each 15 min interval)

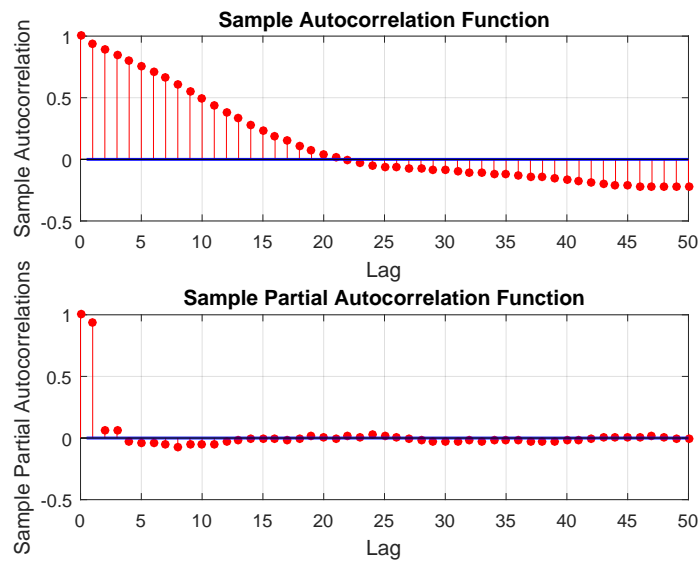


FIGURE 6.2: ACF and PACF of Electricity Load data

6.2 Data preprocessing

The following steps of data preprocessing have been performed:

1. Differentiation of the first order.
2. Outliers removal using Hampel filter.
3. Denoising by using local regression.
4. Scaling values into range $\langle -1, 1 \rangle$.
5. Data partitioning.

6.3 Forecasting results Part 1.

In this section are demonstrated results of the first part of the experiment 2. In this case only 50% of the training data have been used for construction of individual forecasting models.

Double exponential smoothing models are defined as $DEV[x, y]$, where x denotes the number of input values (actual, already measured values), and y denotes number forecasted values.

ARIMA models are defined as $ARIMA(p, d, q)[x, y]$, where p denotes autoregressive parameter, d denotes differencing parameter, q denotes moving average parameter, x denotes the number of input values, and y denotes number of forecasted values

Neural Networks models are defined as $NN[x, h, y]$, where x denotes number of input neurons, h denotes number of neurons in hidden layer, and y denotes number of output neurons, as well as the number of forecasted values.

Model	MSE	RMSE	MSE [%]
ARIMA(20,0,3)[20;5]	0.0011383	0.033738	1.927%
NN[20;30;5]	0.00091128	0.030187	1.5428%
DES[20;5]	0.014928	0.12218	23.0513%
ARIMA(30,0,3)[30;5]	0.0011314	0.033636	1.9154%
NN[30;40;5]	0.00090517	0.030086	1.5324%
DES[30;5]	0.014555	0.12064	22.4749%
ARIMA(40,0,3)[40;5]	0.0010319	0.032123	1.747%
NN[40;50;5]	0.00082654	0.02875	1.3993%
DES[40;5]	0.013356	0.11557	20.6232%
ARIMA(20,0,3)[20;10]	0.0070265	0.083824	12.1052%
NN[20;30;10]	0.0056273	0.075015	9.6946%
DES[20;10]	0.047831	0.2187	73.8594%
ARIMA(30,0,3)[30;10]	0.0075212	0.086725	12.9575%
NN[30;40;10]	0.00602	0.077589	10.3712%
DES[30;10]	0.044106	0.21001	68.107%
ARIMA(40,0,3)[40;10]	0.006681	0.081737	11.5099%
NN[40;50;10]	0.0053511	0.073151	9.2187%
DES[40;10]	0.038885	0.19719	60.045%
ARIMA(20,0,3)[20;15]	0.01444	0.12017	25.0393%
NN[20;30;15]	0.011556	0.1075	20.0387%
DES[20;15]	0.08516	0.29182	131.5014%
ARIMA(30,0,3)[30;15]	0.015982	0.12642	27.713%
NN[30;40;15]	0.012788	0.11308	22.1742%
DES[30;15]	0.082697	0.28757	127.6981%
ARIMA(40,0,3)[40;15]	0.01385	0.11769	24.0161%
NN[40;50;15]	0.011081	0.10526	19.2142%
DES[40;15]	0.077325	0.27807	119.4031%

TABLE 6.1: Results of Experiment 2 - part 1

6.4 Forecasting results Part 2. - Extended Training set

In this section are demonstrated results of the second part of the experiment 2. Training set has been extended (100%), in order to improve the forecasting performance and demonstrate the dependency between forecasting accuracy and the size of the training set.

Model	MSE	RMSE	MSE [%]
ARIMA(20,0,3)[20;5]	0.0010877	0.03298	1.8414%
NN[20;30;5]	0.0008711	0.029514	1.4748%
DES[20;5]	0.014555	0.12064	22.4749%
ARIMA(30,0,3)[30;5]	0.0010811	0.03288	1.8303%
NN[30;40;5]	0.00086497	0.02941	1.4644%
DES[30;5]	0.013822	0.11757	21.3436%
ARIMA(40,0,3)[40;5]	0.00098604	0.031401	1.6693%
NN[40;50;5]	0.00078969	0.028101	1.3369%
DES[40;5]	0.012538	0.11197	19.3613%
ARIMA(20,0,3)[20;10]	0.0067142	0.08194	11.5672%
NN[20;30;10]	0.0053772	0.07333	9.2638%
DES[20;10]	0.046593	0.21586	71.9479%
ARIMA(30,0,3)[30;10]	0.007187	0.084776	12.3816%
NN[30;40;10]	0.0057544	0.075858	9.9136%
DES[30;10]	0.042269	0.20559	65.2704%
ARIMA(40,0,3)[40;10]	0.0063841	0.0799	10.9984%
NN[40;50;10]	0.0051099	0.071483	8.8032%
DES[40;5]	0.037508	0.19367	57.9178%
ARIMA(20,0,3)[20;15]	0.013798	0.11747	23.9264%
NN[20;30;15]	0.011048	0.10511	19.1581%
DES[20;15]	0.083977	0.28979	129.6736%
ARIMA(30,0,3)[30;15]	0.015272	0.12358	26.4813%
NN[30;40;15]	0.012229	0.11058	21.2052%
DES[30;15]	0.080446	0.28363	124.2224%
ARIMA(40,0,3)[40;15]	0.013234	0.11504	22.9487%
NN[40;50;15]	0.010601	0.10296	18.3816%
DES[40;15]	0.076165	0.27598	117.6114%

TABLE 6.2: Results of Experiment 2 - part 2

Chapter 7

Experiment 3

7.1 Data description

In Experiment 3 will be used daily IBM stock prices (in USD). Data are obtained from Yahoo Finance API. Dataset was collected between 1 January 1990 and 31 December 2016. Values are collected at 1 day intervals. Overall number of values in the dataset is 6805, what represents relatively sufficient number of observations for creating forecasting models.

Source: <http://finance.yahoo.com/quote/IBM/history?p=IBM>

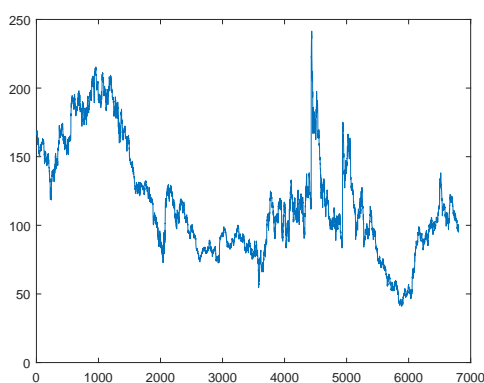


FIGURE 7.1: Daily IBM stock prices (in USD)

From the visual overview of the plotted graph as well as the results of ACF and PACF, it can be easily observed, that given time series data have clear evidence of non-stationarity.

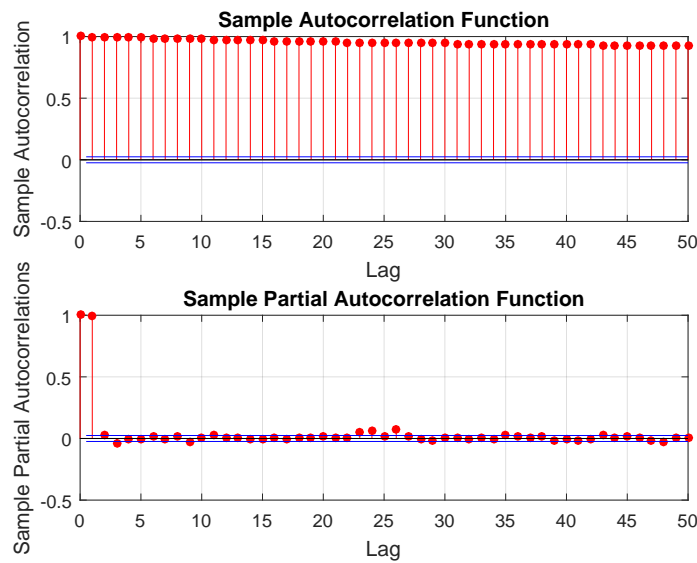


FIGURE 7.2: ACF and PACF of IBM stock prices

7.2 Data preprocessing

The following steps of data preprocessing have been performed:

1. Data transformation by $\log(x)$ function.
2. Differentiation of the first order.
3. Outliers removal using Hampel filter.
4. Denoising by using local regression.
5. Scaling values into range $< -1, 1 >$.
6. Data partitioning.

7.3 Forecasting results Part 1.

In this section are demonstrated results of the first part of the experiment 3. In this case only 50% of the training data have been used for construction of individual forecasting models.

Double exponential smoothing models are defined as $DEV[x, y]$, where x denotes the number of input values (actual, already measured values), and y denotes number forecasted values.

ARIMA models are defined as $ARIMA(p, d, q)[x, y]$, where p denotes autoregressive parameter, d denotes differencing parameter, q denotes moving average parameter, x denotes the number of input values, and y denotes number of forecasted values

Neural Networks models are defined as $NN[x, h, y]$, where x denotes number of input neurons, h denotes number of neurons in hidden layer, and y denotes number of output neurons, as well as the number of forecasted values.

Model	MSE	RMSE	MSE [%]
ARIMA(20,0,3)[20;5]	0.0022765	0.047713	3.8541%
NN[20;30;5]	0.0018226	0.042692	3.0857%
DES[20;5]	0.020961	0.14478	32.3678%
ARIMA(30,0,3)[30;5]	0.0022628	0.047569	3.8308%
NN[30;40;5]	0.0018124	0.042572	3.0684%
DES[30;5]	0.019988	0.14138	30.865%
ARIMA(40,0,3)[40;5]	0.0020638	0.045429	3.494%
NN[40;50;5]	0.001652	0.040645	2.7968%
DES[40;5]	0.018999	0.13784	29.338%
ARIMA(20,0,3)[20;10]	0.014053	0.11855	24.2103%
NN[20;30;10]	0.01125	0.10606	19.3809%
DES[20;10]	0.053763	0.23187	83.0187%
ARIMA(30,0,3)[30;10]	0.015042	0.12265	25.9149%
NN[30;40;10]	0.012047	0.10976	20.7545%
DES[30;10]	0.052852	0.2299	81.6125%
ARIMA(40,0,3)[40;10]	0.013362	0.11559	23.0199%
NN[40;50;10]	0.010695	0.10342	18.4257%
DES[40;10]	0.050896	0.2256	78.5912%
ARIMA(20,0,3)[20;15]	0.02888	0.16994	50.0785%
NN[20;30;15]	0.023119	0.15205	40.088%
DES[20;15]	0.09275	0.30455	143.2215%
ARIMA(30,0,3)[30;15]	0.031964	0.17878	55.426%
NN[30;40;15]	0.025578	0.15993	44.3533%
DES[30;15]	0.089538	0.29923	138.2619%
ARIMA(40,0,3)[40;15]	0.0277	0.16643	48.0322%
NN[40;50;15]	0.022173	0.14891	38.4487%
DES[40;15]	0.086332	0.29382	133.311%

TABLE 7.1: Results of Experiment 3 - part 1

7.4 Forecasting results Part 2. - Extended Training set

In this section are demonstrated results of the second part of the experiment 3. Training set has been extended (100%), in order to improve the forecasting performance and demonstrate the dependency between forecasting accuracy and the size of the training set.

Model	MSE	RMSE	MSE [%]
ARIMA(20,0,3)[20;5]	0.00215	0.046369	3.64%
NN[20;30;5]	0.001722	0.041497	2.9154%
DES[20;5]	0.020566	0.14341	31.7566%
ARIMA(30,0,3)[30;5]	0.0021371	0.046229	3.618%
NN[30;40;5]	0.0017109	0.041363	2.8965%
DES[30;5]	0.018999	0.13784	29.338%
ARIMA(40,0,3)[40;5]	0.0019491	0.044149	3.2999%
NN[40;50;5]	0.0015604	0.039502	2.6417%
DES[40;5]	0.018288	0.13523	28.2402%
ARIMA(20,0,3)[20;10]	0.013272	0.11521	22.8653%
NN[20;30;10]	0.010625	0.10308	18.3051%
DES[20;10]	0.053567	0.23145	82.716%
ARIMA(30,0,3)[30;10]	0.014207	0.11919	24.4752%
NN[30;40;10]	0.011375	0.10665	19.5963%
DES[30;10]	0.052216	0.22851	80.6299%
ARIMA(40,0,3)[40;10]	0.01262	0.11234	21.741%
NN[40;50;10]	0.010103	0.10052	17.4059%
DES[40;10]	0.049295	0.22202	76.1196%
ARIMA(20,0,3)[20;15]	0.027276	0.16515	47.2964%
NN[20;30;15]	0.021828	0.14774	37.8502%
DES[20;15]	0.092553	0.30422	142.9163%
ARIMA(30,0,3)[30;15]	0.030188	0.17375	52.3468%
NN[30;40;15]	0.02416	0.15543	41.8937%
DES[30;15]	0.088837	0.29806	137.1793%
ARIMA(40,0,3)[40;15]	0.026161	0.16174	45.3637%
NN[40;50;15]	0.020931	0.14468	36.2956%
DES[40;15]	0.08579	0.2929	132.473%

Chapter 8

Results Summary

8.1 Results Summary

Experiments performed in the previous sections gave a large amount of information about forecasting accuracy. In order to find out, which method had the best performance, there was calculated average error for each of the methods.

	ARIMA	NN	DES
Average error [%]	17.8319 %	14.2740 %	75.1143 %

TABLE 8.1: Forecasting methods - Average error

Table 8.1 demonstrates, that ANN forecasting methods have the lowest average error, calculated from all experiments. On the second place, there are ARIMA forecasting models, with a little bit worse result. The worst result was presented by the double exponential smoothing models. With the average error five times higher than NN models, exponential smoothing is out of competition.

The given results can be explained by the following facts. Exponential smoothing is relatively simple algorithm and it is not able to deal with prediction of fast turnovers, especially in the case of econometric time series. ARIMA models presented relatively good results, but they still belong to the class of linear models. ANN models with their ability to reproduce nonlinear dependencies, presented the best forecasting results.

Figure 8.1 demonstrates another interesting feature. All three forecasting methods present almost linear dependency between the lead time (number of predicted values) and forecasting accuracy.

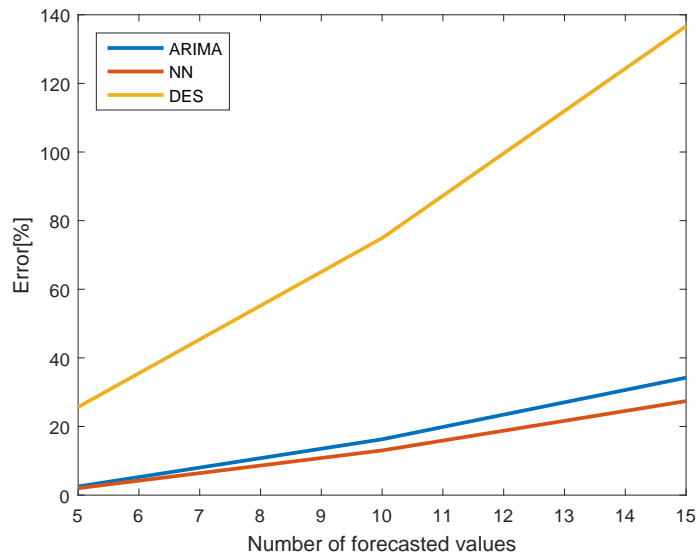


FIGURE 8.1: Dependency between number of predicted values and forecasting accuracy

Another very important investigation is related to the size of training set. According to the assignment, all experiments had to be performed in two phases. First time, all forecasting methods have been trained by using only the half of all training set. The second time, training sets have been extended and the same forecasting models have been built again. The purpose of this, is to investigate, how the size of training set influence forecasting performance. Each experiment in the previous sections, contains two tables with results corresponding to the specific training set. Table 8.2 demonstrates the average improvement of forecasting accuracy by extending the training set. Improvements are expressed as the average improvement of one forecasting method for specific experiment.

	ARIMA	NN	DES
Experiment 1 improvement	1.42 %	1.14 %	3.21 %
Experiment 2 improvement	0.59 %	0.46 %	1.88 %
Experiment 3 improvement	12.33 %	9.86 %	10.80 %

TABLE 8.2: Forecasting improvement by extension of training dataset

The presented improvements are quite reasonable. Experiment 2 demonstrates the lowest improvements for all forecasting methods. This can be explained by fact, that in experiment 2 has been used "electricity load" dataset, which represents relatively stationary time series, without any complex dependencies. Therefore, forecasting methods were able to extract enough useful information even from the half of the training set. Absolutely different situation can be observed with the experiment 3. In this case, the dataset represents relatively complicated time series, the daily IBM stock prices.

Generally, time series like this and also others from econometric area represent very complex and non-stationary processes. In this cases, each section of the series may contain relatively important and unique information. Therefore extension of the training set in experiment 3 had the most significant impact on forecasting performance, for all methods.

Chapter 9

Conclusion

In practical part of this thesis, the potential reader may familiarize with the most important aspects of time series analysis and forecasting. It has been also explained, how important it is to perform a proper data preprocessing. Very often, proper data preprocessing is considered as much harder and significant step, than construction of the forecasting model itself. Further, there were introduced main time series forecasting methods and compared their advantages and disadvantages.

The main aim of the thesis, was to select three promising methods and perform experiments on datasets from typical real life examples.

Experiments have demonstrated good results for two of the three selected forecasting methods. ANN models, as well as ARIMA models, both of them can be suitably used for forecasting time series of various complexity. Double exponential smoothing is not so powerful tool, therefore its exploitation for more complex series is not recommended. There have been also presented interesting investigations related to size of training set and its impact on forecasting performance.

9.0.1 Further improvements

Main directions for the further improvements of time series forecasting, is based mainly on the machine learning approach. Approaching the task seriously, very often requires deeper analysis of given problematic. Data clustering may be very useful in these cases. [7] It allows to find out unknown groupings inside datasets. This technique is often used for anomaly detection task. In combination with neural networks, there can be created very sophisticated forecasting models.

Bibliography

- [1] Gregory C. Reinsel Greta M. Ljung George E. P. Box, Gwilym M. Jenkins. *Time Series Analysis: Forecasting and Control*. Wiley, fifth edition, aug 2015.
- [2] Michael Falk. *A First Course on Time Series Analysis — Examples with SAS*. Chair of Statistics, University of Wurzburg, aug 2012.
- [3] Time series decomposition, jan 2017. URL <https://www.mathworks.com/help/econ/detrending.html>.
- [4] Autocorrelation and partial autocorrelation, jan 2017. URL <https://www.mathworks.com/help/econ/autocorrelation-and-partial-autocorrelation.html>.
- [5] Outlier removal using hampel filter, jan 2017. URL <https://www.mathworks.com/help/signal/ref/hampel.html>.
- [6] Savitzky-golay filtering, jan 2017. URL <https://www.mathworks.com/help/signal/ref/sgolayfilt.html>.
- [7] Simon S. HAYKIN. *Neural networks and learning machines*. New York: Prentice Hall,, third edition, 2009.
- [8] University of Maryland prof. Charles Stangor. The neuron is the building block of the nervous system, jan 2017. URL <http://2012books.lardbucket.org/books/beginning-psychology/s07-01-the-neuron-is-the-building-blo.html>.
- [9] Commonly used activation functions, jan 2017. URL <http://cs231n.github.io/neural-networks-1/>.
- [10] Michael A. Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015.