2017

# Forecasting Automobile Demand Via Artificial Neural Networks & Neuro-Fuzzy Systems

Armin Niaki

# Forecasting automobile demand via Artificial Neural Networks & Neuro-Fuzzy Systems

**Armin Niaki**

Thesis submitted to the
**Statler College of Engineering and Mineral Resources**
**at West Virginia University**
in partial fulfillment of the requirements
for the degree of

**Master of Science in**
**Industrial Engineering**

**Department of Industrial and Management Systems Engineering**

**West Virginia University**

**Morgantown, West Virginia**
**2017**

**Keywords:** Forecasting, Time Series, Artificial Intelligence, Artificial Neural Networks, Neuro-fuzzy systems, Adaptive Neuro-Fuzzy Inference System, Intelligent Algorithms, Box-Jenkins modeling.

## Abstract

Forecasting automobile demand via
Artificial Neural Networks and Neuro-Fuzzy Systems

By Armin Niaki
Master of Science in Industrial Engineering
West Virginia University
Majid Jaridi, Ph.D., Chair

The objective of this research is to obtain an accurate forecasting model for the demand for automobiles in Iran's domestic market. The model is constructed using production data for vehicles manufactured from 2006 to 2016, by Iranian car makers. The increasing demand for transportation and automobiles in Iran necessitated an accurate forecasting model for car manufacturing companies in Iran so that future demand is met. Demand is deduced as a function of the historical data. The monthly gold, rubber, and iron ore prices along with the monthly commodity metals price index and the Stock index of Iran are

Artificial neural network (ANN) and artificial neuro-fuzzy system (ANFIS) have been utilized in many fields such as energy consumption and load forecasting fields. The performances of the methodologies are investigated towards obtaining the most accurate forecasting model in terms of the forecast Mean Absolute Percentage Error (MAPE). It was concluded that the feedforward multi-layer perceptron network with back-propagation and the Levenberg-Marquardt learning algorithm provides forecasts with the lowest MAPE (5.85%) among the other models. Further development of the ANN network based on more data is recommended to enhance the model and obtain more accurate networks and subsequently improved forecasts.

**Keywords:** Forecasting, Time Series, Artificial Intelligence, Artificial Neural Networks, Neuro-fuzzy systems, Adaptive Neuro-Fuzzy Inference System, Intelligent Algorithms, Box-Jenkins modeling.

## Acknowledgements

I would first like to thank my parents for endlessly supporting me throughout the years, especially when I decided to make the transition from Architecture to Industrial Engineering. They continue to be my primary source of motivation and determination. It would be impossible to stand where I stand, without their support.

I would also like to thank and acknowledge my advisor and mentor, Professor Jaridi for all his support and kindness throughout my time at West Virginia University. I would not have made it this far without his guidance and generosity.

Also, I would like to thank Dr. Iskander and Dr. Creese for their support and guidance during the first year of my transition into Industrial Engineering. In addition, I would like to thank my thesis committee members, Dr. Currie and Dr. Yang for their helpful insight and being on my thesis committee.

## Table of Contents

## List of Figures

## List of Tables

## List of Abbreviations

| | |
|---|---|
| ACF | Autocorrelation Function |
| AI | Artificial Intelligence |
| AR | Autoregressive |
| ANFIS | Adaptive Neuro-Fuzzy Inference System |
| ANN | Artificial Neural Networks |
| ARIMA | Autoregressive integrated moving average model |
| ARMA | Autoregressive moving average |
| BS | Backward Shift |
| FIS | Fuzzy Inference System |
| GARCH | Generalized auto regressive conditional heteroscedasticity |
| GDP | Gross Domestic Product |
| LSE | Least Squared Error method |
| MA | Moving Average |
| MAPE | Mean Absolute Percentage Error |
| MF | Membership Function |
| MLP | Multi-layer Perceptron |
| MPE | Mean Percentage Error |
| MSE | Mean Squared Error |
| OECD | Organization for Economic Co-operation and Development |
| PACF | Partial-autocorrelation Function |
| RBF | Radial Basis Function network |
| SMP | System Marginal Price |
| SD | Similar Days |
| SAIPA | Société Anonyme Iranienne de Production Automobile |
| Tan-Sig | Sigmoid and Tangent function |

# CHAPTER 1 | Introduction

## 1.1.Introduction

To forecast is to predict or estimate a future event or trend. Manufacturing industries have been among ever-growing in most countries since the industrial revolution in the 18th century. Transportation necessities have played an important role in the development of the automotive manufacturing industry. Automotive developments have an imminent role among manufacturing industries and consist of highly complex projects (Nagel and Singleton, 2011). The complex projects in the automotive manufacturing industries present numerous challenges in management throughout the time horizon of projects. Furthermore, these projects are driven by shortening life cycle strategies and the continuous diversification of the manufacturing industry (Nurmi, Marttin, Rossi, 2007).

Automotive development projects are characterized as chronological, localized and objective work tasks using resources such as consumables and labor (Schwarze, 2010). An efficient process flow may be obtained through the management of project resources. Increasing customer requirements has led to intense competition in innovation between manufacturers, which has presented new challenges in the industry (Von Cube et al. 2014).

Forecasting methods are utilized by the automotive industries to foresee the market demand for their manufactured products. Production is the primary concern of manufacturing operations and is specified based on the forecast of the market demand, highlighting the necessity of forecasting (Haj-Shir Mohammadi, 2006). Forecasting offers valuable applications in inventory systems, production and product distribution (Fatemi-Ghomi, 2004). Moreover, forecasting presents applications in quality control, financial planning and investment analysis (Montgomery et al., 1990).

## 1.2.Problem Statement

The automotive industry is driven by numerous factors, each capable of significantly influencing the operations of manufacturing plants. The market demand is among such influential factors, which is of crucial importance for manufacturing operations. Automotive industries vastly benefit from the ability to forecast the demand for their products as it provides opportunities to allocate resources and manage operations in advance efficiently.

Given the increasing demand for transportation and automobiles in Iran throughout the past half century and no existing forecasting models for the demand, an accurate model would provide valuable information for manufacturing companies in the industry. Moreover, the lack of reliable forecasting models is mainly due to inaccessible data and secrecy among competing manufacturers.

## 1.3.Research Objective

The core objective of this research is to obtain an accurate forecasting model for the demand for automobiles in Iran's domestic market. The models are constructed on inferences made from the production data of vehicles manufactured from 2006 to 2016, obtained from Iran's Ministry of Industry, Mine & Trade's yearly reports, alongside other further mentioned sources. The forecasting model will provide monthly forecasts for the automobile demand in the domestic market. The prior will provide manufacturing industries with valuable information regarding the market demand, which will significantly influence the operations of manufacturing companies. Furthermore, other variables of interest mentioned before are considered to enhance the forecasting model, alongside providing opportunities to draw further statistical conclusions. The variables, within the prior specified time horizon, are as follows:

- The Monthly gold price (dollar per ounce).
- The Monthly Commodity Metals price index.

- The Monthly rubber price (cents per pound).
- The Monthly Stock index of Iran.
- The Monthly Iron ore price (dollar per dry metric ton).

Note that the five selected variables were chosen out of 9 variables according to the correlation value to the response. Variables with high significant correlation are selected, as demonstrated further in chapter 3.

## 1.4.Research Scope

This research focuses on the production data of automobiles produced from March 2006 to October 2016 in Iran. The demand rate for automobiles is inferred from the production data (e.g. production is directly related to demand due to the specific market characteristics assumed). Moreover, statistical inferences are made about the specified variables of interest that potentially have a relationship to the market demand. Note that based on preliminary surveys, little to none research has been performed within this area.

A significant amount of historical data is required to construct an accurate forecasting model. "Information is crucial for any decision making and development of planning decisions. Reliable and quality information facilitates and improves decision making processes. Any decision-making process requires analysis of the past and present states of a sector and a vision about its future (Bhattacharyya, 2011)."

## 1.5.Methodology

Forecasting processes pursue the objective of predicting future events or conditions based on historical data (Levenbach and Cleary, 2006). Forecasting methods are typically divided into the following methods:

- Quantitative Methods
- Qualitative Methods

All quantitative forecasting methods use the same principle: historical data is analyzed to identify a pattern or trend that explains the process. After the pattern is determined, it is extrapolated into the future to obtain predictions. On the contrary, qualitative methods are built to deal with long-term trends wherein the historical data and patterns are essential to apply statistical forecasting methods, only were not ready or did not apply (Wheelwright and Makridakis, 1985).

In this research, quantitative methods are utilized to obtain forecasting models for automobile demand in Iran. As demonstrated further in chapter 4, quantitative methods are divided into four general groups:

- Time Series
- Causal
- Neural networks
- Neuro-fuzzy systems

Artificial neural networks & neuro-fuzzy systems are the primary forecasting methods in this research, which will further be compared based on the specified performance measures. Also, Causal models are utilized in regards to the variables of interest. Furthermore, the Box-Jenkins methods also serve as a benchmark for comparison between the employed methods.

## 1.6.Software

The software used in this research is R, MATLAB, and Minitab. MATLAB can analyze data via Neural Networks. “MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy to use environment (Mathworks’ INC., 2001).” R is a statistical software used for Time Series analysis in this research. “R provides a wide variety of statistical (linear and nonlinear modeling, classical statistical tests, time-series analysis, classification, clustering,) and graphical techniques, and is highly extensible (R Foundation, 2016).”

## 1.7.Assumptions & Conditions

The following assumptions are necessary for this research:

- The forecasts performed in this research are based on limited historical data of the automobile production rates.
- The market demand for automobiles is inferred from the observed automobile production rate e.g. increase of production constitutes a prior increase in demand.
- The database is available for all of the collected data.
- The Neural Networks tool available in MATLAB is capable of forecasting based on the data.
- The Box-Jenkins procedure in Minitab is capable of forecasting based on the data.
- The process is assumed to be a stationary process. A stationary process is defined as a stochastic process that maintains the same joint probability distribution through time shifts. Note that differencing is applied to achieve stationarity for processes that are not stationary.
- The process is assumed to be an ergodic process. An ergodic process is defined as a stochastic process that can be deduced from a single sufficient random sample of the process (Leipzig: J.A, Barth. 1898).
- Expected value of the error is equal to zero; moreover, the variance is equal to $\sigma_\varepsilon^2$.

# CHAPTER 2 | Literature Review

This chapter covers intelligent algorithms forecasting techniques, time series forecasting algorithms, and a comparative discussion of reviewed papers alongside their methodologies and findings.

“Forecasting can be formed in many different ways. The method chosen for forecast depends on the purpose and importance of the intended forecasts (Pankratz, 1983).” As mentioned earlier in Chapter 1, uncertainty associated with the future of processes make forecasts entirely necessary. “Energy is indispensable to the function of every industry and the everyday life of individuals, and, therefore it is considered as a national security issue. The main areas of energy use are the residential, commercial, industrial and transportation sectors (Levenback and Cleary, 2006).” Based on the importance of energy, specifically in the industrial and manufacturing sectors, forecasting proves to be an imminent tool. “Analyses of energy problems have attracted interdisciplinary interests and researchers from various fields that have left their impressions on these studies. The influence of engineering, operations research and other decision support systems in the field of energy and economics has been profound (Bhattacharyya, 2011).” The reviewed journal papers for this research suggest a rapid and increasing growth in the usage of artificial intelligence in the forecasting field.

A forecasting method which was performed by the exponential weighted moving average method was presented 57 years ago by Muth (1960). This study included simple and seasonal effects that provided a linear trend and utilized actual data to examine this modeling methodology. Furthermore, Trigg’s study of automatic monitoring of forecasting processes uses a first-order exponential model based on data containing jumps.

Harrison (1962) performed a study for short term sales forecasting. Several short-term sales prediction methods are examined in this study. These methods include the Box-Jenkins method and Brown’s method. This study demonstrates that the multi-parameter procedures for short-term forecasting of sales do not provide significantly better results compared to the other methods. Harrison recommends the one parameter exponential procedure to forecast non-seasonal sales data.

Crane & Eeatly (1967) introduce a two stage exponential model with exponential smoothing and multiple regression. The exponential process forecasts were applied along with other independent variables in constructing a multiple regression model. Economic data series related to bank deposits were modeled with the use of this combined forecasting method. Furthermore, Tsokos (1971) investigated forecasting models for short-term forecasting of economic data. Some of the processes such as the autoregressive, moving average and the mixture of the two were studied in short-term forecasting models for commodity contracts. These forecasts were evaluated based on the minimum residual variance performance measure.

Fox (1972) investigates outliers in time series. Two general types of outliers are considered in this study. The first type is the gross error in the sample of observations. The second type of outlier affects not only present observations but also subsequent observations as well. A ratio performance measure was introduced to examine the problems caused by the outliers.

Box & Jenkins (1994) developed the multiplicative ARIMA model based on airline data. The issue of seasonal variation, not addressed by the classical ARIMA model, is adopted in this model. Furthermore, once the period of a time series is identified, the multiplicative ARIMA model can be utilized to forecast this series with more precision compared to the classical ARIMA model.

Forecasts are necessary given the fact that not only is the future uncertain for any realistic process, but making decisions based on accurate forecasts are far more efficient compared to a decision that is not based on forecasts. Lu et al. (1993) attempted to evaluate artificial neural networks as a technique for short-term load forecasting. This study presents the obtained results from investigating whether the artificial neural network model is system dependent, and case dependent based on the collected data from two utilities. Moreover, the effectiveness of a next 24 hour artificial neural networks (ANN) model for predicting a 24-hour load profile at one time is compared with the traditional next one-hour artificial neural networks model. The artificial neural networks model in this study utilizes the feedforward and backpropagation network. The utilized artificial neural networks model in this study proved to be an extremely useful tool for short-term load forecasting. Artificial neural networks models have presented reliable and promising results in load forecasting. Also, the last study indicates that in general, the artificial neural network model is case independent except for cases that season changing periods or abrupt variations in the temperature and load conditions are present. Furthermore, based on this study, the initial testing of another 24 hour ANN model is recommended to be further investigated.

Szkuta et al. (1999) presented a System Marginal Price (SMP) short-term forecasting implementation via the ANN computing technique. This approach utilizes a three-layered ANN paradigm with backpropagation. The retrospective SMP real world historical data, acquired from the deregulated Victorian power system was implemented for training and testing purposes. The results presented, confirm the importance of the ANN based approach in forecasting the SMP.

Hippert et al. (2001) evaluated and reviewed many papers that demonstrated the application of artificial neural networks for short-time load forecasting. Some scholars have described and published successful experiments and tests in forecasting via artificial neural networks. Moshiri and Foroutan (2005) compared artificial neural networks with traditional statistical nonlinear (GARCH) and linear (ARIMA) models. The intent of their research was to obtain more accurate results in forecasting the daily oil price. The results indicate that the artificial neural network model provides more accurate results compared to the other models.

Tarafdar and Kashtiban (2005) demonstrated artificial neural networks' modeling capabilities in power systems. In their paper, the main advantages of applying neural networks modeling are described as follows:

- Artificial neural networks models are more capable of dealing with stochastic variations of the scheduled operation point with increasing data,

- Artificial neural network models are very fast and provide online processing and classification, and
- Artificial neural network models have the ability of implicit nonlinear modeling alongside filtering system data.

Based on the results of this paper, several other types of research indicate that artificial neural networks' modeling relies on conventional simulations to produce training vectors (especially if the historical data is noisy). However, significant challenges such as the training time, selection of training vector, upgrading trained neural nets and the integration of technologies remain to be tackled in the usage of artificial neural network models for power systems.

Sinaie et al. (2005) investigated the Tehran stock index forecasting through the usage of artificial neural network modeling and compared its performance with the ARIMA model. The results indicate that artificial neural networks with the back propagation training algorithm had the ability to forecast more accurately than the linear ARIMA model.

Ahmari Nejad et al. (2005) studied electricity price forecasting in the energy market. The underlying idea in this paper is to analyze the given problem regarding energy demand and the customers' behavior throughout the energy market. Artificial neural networks modeling with a multi-layer perceptron architecture is utilized in this research. Out of the total 1,036 data points, 836 were designated for training and the remaining 200 data points were used to test the accuracy of the constructed model. Based on the results, the artificial neural network model provided satisfactory results in forecasting the electricity price.

Sarafraz and Afsar (2005) attempted to predict the price of gold via regression and an ANFIS model. In this study, a total of 520 data points were taken into consideration. 260 data points were used as training data, 130 data points as test data and the remaining 130 points were designated as evaluation data. The results indicate that the regression model is capable of predicting gold price with 93% accuracy. However, the neuro-fuzzy system provides a 99.23% accuracy in forecasting the price of gold.

A hybrid neural network and fuzzy logic model based on theoretical arguments in Takagi-Sugeno model (ANFIS) were utilized by Azar and Afsar (2006) to forecast stock prices. In this paper, the proposed fuzzy neural network model was compared to the ARIMA model. Results indicate that the fuzzy neural network model provides more accurate forecasts than the ARIMA model. Also, the fuzzy neural network system presented a unique rapid convergence alongside high precision to predict stock prices, based on the daily stock price over a period of 5 years. The experimental results indicate that the use of an artificial neural network model and a fuzzy logic model is a successful combination that provides significant reductions in the forecast error.

Catalao et al. (2007) demonstrated the necessity of short-term price forecasts for consumers and production units to derive appropriate bidding strategies within energy markets. Accurate forecasting tools are necessary for maximizing the customers' satisfaction and the production unit's profit. A three-layered feedforward artificial neural network, trained by the Levenberg-Marquardt algorithm is utilized in this research to forecast the next period's (168 hour periods) electricity price. The neural network toolbox of MATLAB was chosen for this task due to its flexibility and simplicity. The primary input/training data for the artificial neural network was obtained from historical data in the year 2002 from Spain. The artificial neural network was examined against the ARIMA model.

Farjam Nia et al. (2007) utilized ARIMA and artificial neural networks models in 2007 to forecast the daily price of crude oil between 1983 and 2005. Note that sensitivity analysis was used to select input parameters in forecasting oil price trends. Like previously reviewed studies, the artificial neural network model provided superior results compared to ARIMA's results.

Mandal et al. (2007) investigated an artificial neural network model based on the similar days (SD) method, to forecast the next day's price in the competitive electricity market, where price forecasting provides estimates of the electricity price in the upcoming days. A multi-layer feedforward artificial neural network was introduced to forecast the electricity price in the next 24 hours. The neural network model consisted of an input layer, a hidden layer, and a single output layer. The model was trained using historical data of the past 45 days (starting from a day before the forecast day ~ today), also, the historical data pertaining to 45 days before and after the forecast day, from the previous year. The utilized training algorithm for this model is the error back-propagation training algorithm. Results obtained by simulation indicate that the utilized algorithm in this research is efficient, accurate and robust and provide more accurate forecasts for the electricity prices for any given day of the week.

A short-term forecasting model for crude oil prices based on a three layer, feedforward artificial neural networks model with back propagation algorithm was introduced by Haidar et al. (2008). The network structure was selected after systematic, rigorous tests involving a large number of experiments on the historical data. Moreover, two groups of inputs have been chosen and tested for the model. The first group selected is the crude oil futures data and the second group is the market data, which includes S&P 500, the price of gold, the dollar index, and heating oil prices. The results indicate that a feedforward neural network model, adequately designed with the appropriate selection of training inputs provide the capability to forecast noisy time series with precision.

Yaghubi et al. (2008) utilized the Chen model for variable length and the Yu model to deal with the uncertainty associated with forecasting a currency market. The indicated two models run in c# 2.0, based on the available data (http://fx.sauder.ubc.ca/). The Yu model recommends the weighted fuzzy time series to reduce uncertainty, and the Chen model attempts to optimize the fuzzy time series using genetic algorithms. Results indicate that the combination of the two models provides more precise results for forecasting.

Artificial neural networks model performance was compared to the ARIMA model's performance by Esmaili and Oloomi in 2008. The utilized neural networks model with backpropagation training provided energy price forecasting in the first step and energy load and price forecasting once again, in the second step. The utilized time series consisted of hourly data points throughout a year. The observed results indicate that the artificial neural networks model maintained less error compared to the ARIMA model. However, the neural network model spent more time than the ARIMA model to predict the energy prices. Furthermore, authors suggest the neural networks model for precision and accuracy and the ARIMA model for time efficiency.

A fuzzy-based model for time series forecasting was developed by Zalloi (2009), which was comprised of the following 3 phases:

1. Data Sampling: Historical data may be categorized into various time scales such as daily, weekly, monthly or yearly data. In daily historical data, inconsistencies and missing points for days which the market is closed, are present in the data. Thus, weekly and monthly data without such inconsistencies are utilized as an alternative. Note that the network's performance depends upon the quality of the data used for training. This phase consists of the following 3 step process: Data Division, Data normalization and the determination of network architecture.

2. Network Training: In this phase, the following three main tasks are carried out: Initialization, Sample training, and Sample validation.

3. Future Price Forecasting: Using the moving window technique in the MATLAB software, PourKazemi and Asadi (2009) selected the best neural network to predict the oil price and oil resources in the member countries of the Organization for Economic Co-operation and Development (OECD). Furthermore, the ARIMA technique was utilized to forecast alongside the developed neural network. 30 observations demonstrate that the designed neural network (with the backpropagation training algorithm) provided more appropriate and accurate forecasts compared to the ARIMA forecasting technique.

ZaraNejad and Hamid (2009) used neural networks in MATLAB to forecast Iran's inflation rate. The research investigates the performance of three different training algorithms for the neural network. Conjugate Gradient, Quasi-Newton, and Levenberg-Marquard are the three training algorithms. Time series data was gathered for the period between 1959 and 2007. The Levenberg-Marquard algorithm provided less error among the other algorithms and was subsequently chosen for the network.

The Iranian GDP growth was investigated through estimation by Mirbagheri (2010). Mirbagheri compares the predictive results of fuzzy-logic and neural-fuzzy methods. The

Takagi-Sugeno fuzzy inference system (ANFIS) is utilized to design the fuzzy neural network for forecasting the GDP growth. Based on the provided comparing criteria specified in the research, the paper demonstrates that the neural-fuzzy method provides better results compared to the fuzzy-logic. Ultimately, neural-fuzzy methods are recommended in forecasting annual growth.

Sadeghi et al. (2011) investigated the performance of the ARIMA model alongside neural networks in short-term forecasting of the Organization of the Petroleum Exporting Countries (OPEC) crude oil price. The results indicated that the neural networks model with backpropagation training was significantly better and more time efficient than the ARIMA technique.

Artificial neural networks, which were previously applied to load forecasting problems with success, are now used for electricity price forecasting. This is mainly due to the well-known advantage of neural networks, which is being able to approximate nonlinear functions alongside solving problems with complex (not well defined/not easily computable) input-output relationships. Neural networks provide this advantage because they are data-driven processes. The numerical results presented in Sadeghi et al. confirmed the high value of artificial neural network models in forecasting short-term electricity prices.

Akhavan Niaki and Hoseinzade (2013) utilized artificial neural networks to forecast the S&P500 index. The research implements Design of experiments and analysis of variances to identify and select its input variables and compares the forecasts to a regression model. Their results indicate that the ANN could significantly improve the trading profit compared to the other strategy.

A forecasting model for energy price and consumption for Iranian Industrial sectors using ANN and ANFIS was developed in by Mirsoltani, Akhavan Niaki (2013). This study is based on the dataset containing the monthly price and consumption of gas, oil, petrol, and liquid petroleum gas throughout the study period. The ANFIS model is identified as the superior model in this research.

# CHAPTER 3 | Data Description & Remedial Procedures

This section demonstrates the utilized data in this research, also the performed remedial procedures. The monthly production quantity is selected as the response. Furthermore, The Monthly crude oil price, Monthly gold price, Monthly commodity metals Index, Monthly rubber price, Monthly aluminum price, Monthly steel price, Monthly copper price, Monthly lead price, Monthly stock index, and Monthly iron ore price are selected for the predictor variables. The variables with the highest significant correlation to the response are further identified and utilized for the ANN and ANFIS models.

## 3.1. Data Description

Historical data pertaining to the factors of interest in this research have been collected from Iran's Ministry of Industry, Mine & Trade's yearly reports in addition to other specified sources. The production of all sedan vehicles throughout 128 months of production (2006-2016) is obtained from the annual reports. Furthermore, as mentioned earlier, the demand is inferred based on the recorded production data due to the unique characteristic of the domestic market. Figure 3.1 shows the total number of automobiles produced per during this study's time horizon.

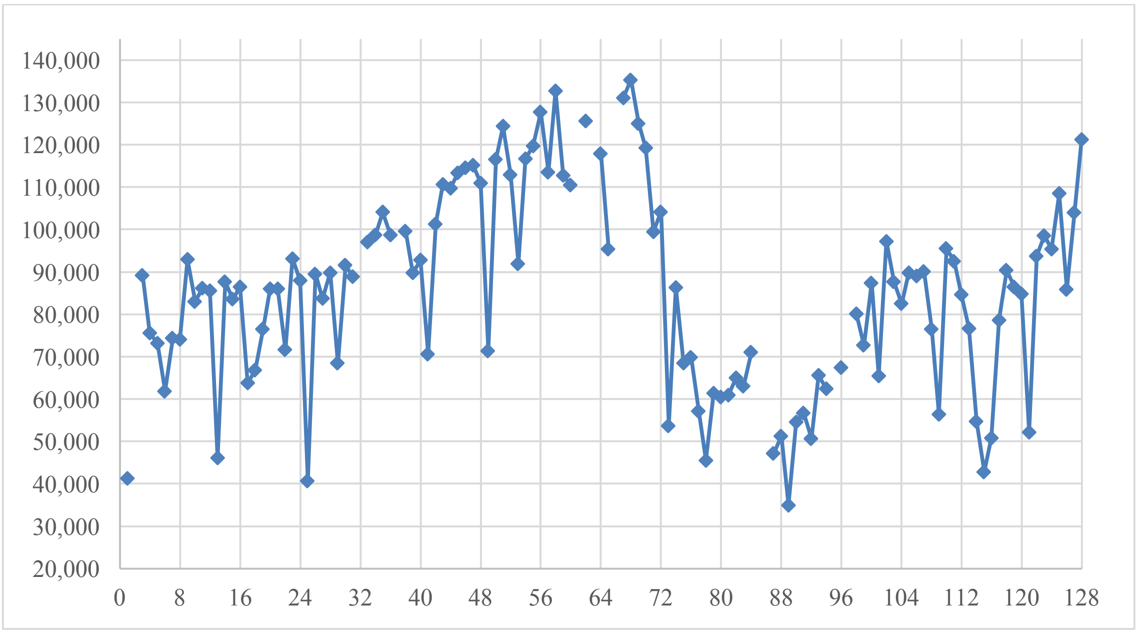


Figure 3.1 – Monthly production quantity of sedan vehicles in Iran (2006-2016)

Table 3.1 shows the data for automobile production in 2006-2016. Note that data for months 2, 32, 37, 61, 63, 66, 85, 86, 95, and 97 are missing. Also, there are no missing values for months 108-128.

Table 3.1 – First 108 Monthly production quantity of sedan vehicles in Iran (Raw data, 2006-2016)

| Month # | Production | Month # | Production | Month # | Production |
|---|---|---|---|---|---|
| 1 | 41,267 | 37 | | 73 | 53,740 |
| 2 | | 38 | 99,592 | 74 | 86,356 |
| 3 | 89,124 | 39 | 89,754 | 75 | 68,509 |
| 4 | 75,562 | 40 | 92,790 | 76 | 69,857 |
| 5 | 73,205 | 41 | 70,653 | 77 | 57,110 |
| 6 | 61,880 | 42 | 101,321 | 78 | 45,550 |
| 7 | 74,327 | 43 | 110,711 | 79 | 61,336 |
| 8 | 74,004 | 44 | 109,774 | 80 | 60,470 |
| 9 | 92,920 | 45 | 113,404 | 81 | 60,869 |
| 10 | 83,024 | 46 | 114,524 | 82 | 64,984 |
| 11 | 86,182 | 47 | 115,240 | 83 | 63,039 |
| 12 | 85,552 | 48 | 110,978 | 84 | 71,028 |
| 13 | 46,058 | 49 | 71,383 | 85 | |
| 14 | 87,629 | 50 | 116,550 | 86 | |
| 15 | 83,566 | 51 | 124,390 | 87 | 47,146 |
| 16 | 86,467 | 52 | 112,847 | 88 | 51,280 |
| 17 | 63,733 | 53 | 91,954 | 89 | 34,969 |
| 18 | 66,817 | 54 | 116,615 | 90 | 54,519 |
| 19 | 76,444 | 55 | 119,720 | 91 | 56,648 |
| 20 | 85,974 | 56 | 127,690 | 92 | 50,689 |
| 21 | 85,994 | 57 | 113,517 | 93 | 65,603 |
| 22 | 71,584 | 58 | 132,723 | 94 | 62,508 |
| 23 | 93,130 | 59 | 112,766 | 95 | |

| | | | | | |
|---|---|---|---|---|---|
| 24 | 87,910 | 60 | 110,482 | 96 | 67,378 |
| 25 | 40,604 | 61 | | 97 | |
| 26 | 89,489 | 62 | 125,648 | 98 | 80,144 |
| 27 | 83,729 | 63 | | 99 | 72,651 |
| 28 | 89,856 | 64 | 117,908 | 100 | 87,442 |
| 29 | 68,434 | 65 | 95,406 | 101 | 65,497 |
| 30 | 91,557 | 66 | | 102 | 97,255 |
| 31 | 88,883 | 67 | 131,090 | 103 | 87,666 |
| 32 | | 68 | 135,286 | 104 | 82,505 |
| 33 | 97,069 | 69 | 125,020 | 105 | 89,806 |
| 34 | 98,761 | 70 | 119,221 | 106 | 89,024 |
| 35 | 104,089 | 71 | 99,498 | 107 | 90,035 |
| 36 | 98,731 | 72 | 104,204 | 108 | 76,449 |

As seen in the previous figure and table, the production volume trend is increasing in the first 68 months.

The monthly gold price is another selected variable. The price of gold can be regarded as an important economic indicator. Figure 3.2 below shows the progression of gold price throughout the time horizon; also, notice that the price of an ounce of gold has increased by almost $1,400 between 2006 and 2016.

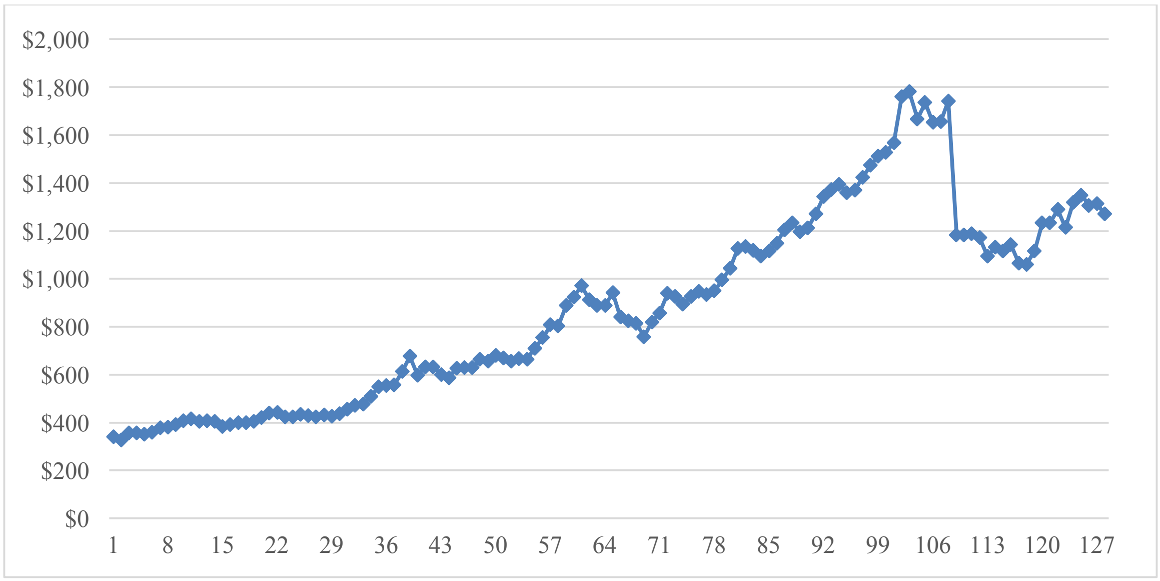


Figure 3.2 – Monthly gold price per ounce (Source: http://data.okfn.org/data/core/gold-prices)

Another utilized variable is the commodity metals price index. This index consists of copper, aluminum, iron ore, tin, nickel, zinc, lead, and Uranium Monthly price index. Figure 3.3 shows this index throughout the time horizon.

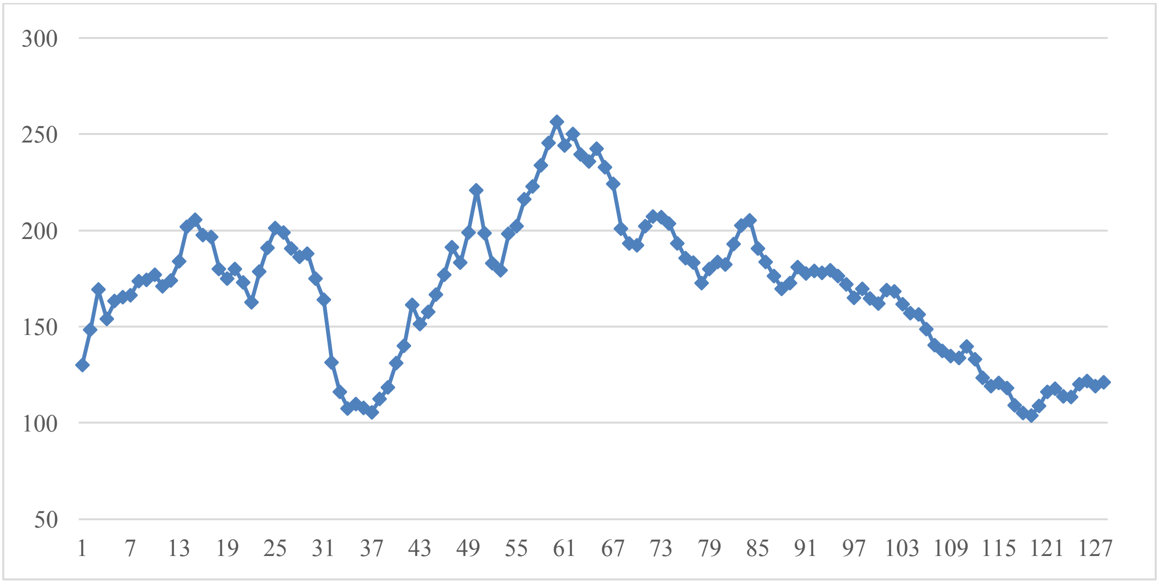


Figure 3.3 – commodity metals price index (Source: http://www.indexmundi.com/commodities/?commodity=metals-price-index)

The monthly rubber price is another variable of interest, which is demonstrated in figure 3.4 below. Rubber is a material required in various parts and components of vehicles and thus was selected as a variable of interest.

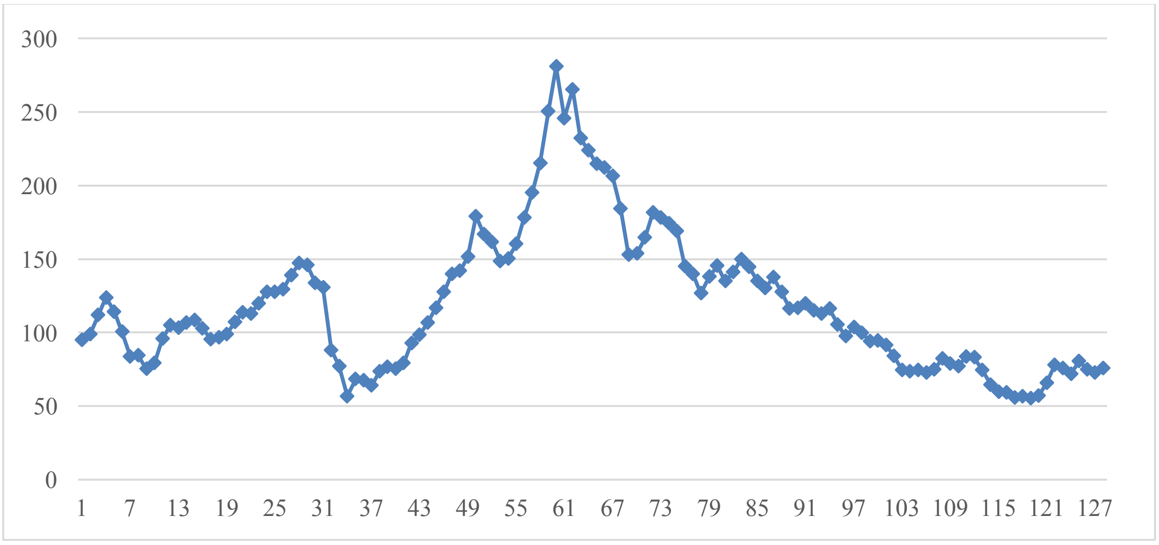


Figure 3.4 – Monthly rubber price – cents per Ibs. (Source: http://www.indexmundi.com/commodities/?commodity=rubber&months=180/)

Iran’s stock index is also a selected variable for the study as demonstrated in figure 3.5 below. The domestic stock index can be considered as an economic indicator for manufacturing companies.

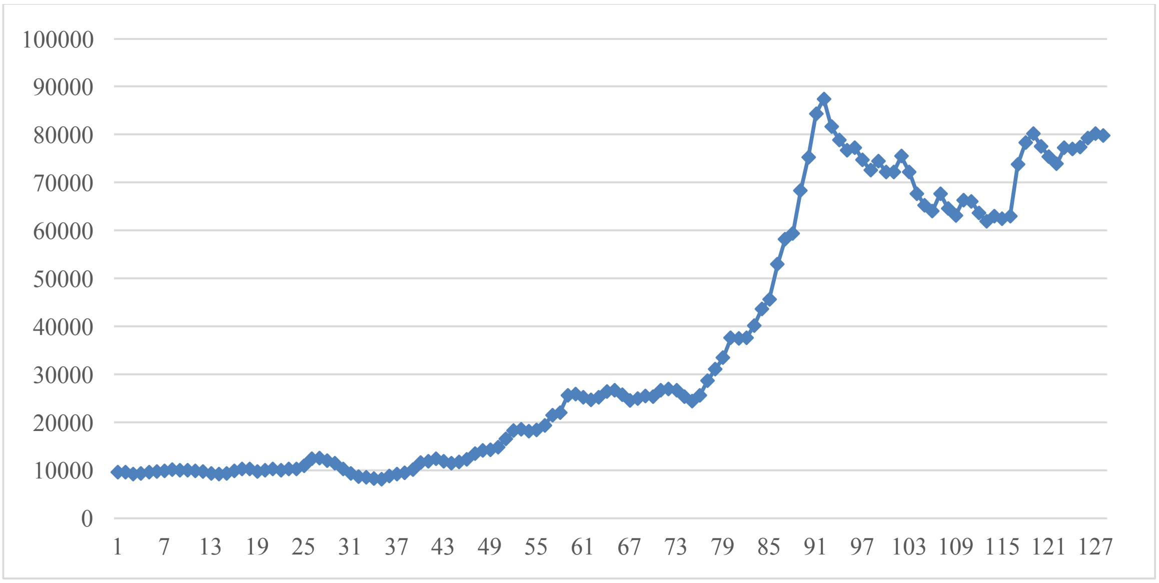


Figure 3.5 – Stock Index (Source: http://www.tse.ir/en/)

Moreover, the Iron ore price (dollar per dry metric ton) is the last selected variable for the study as demonstrated in figure 3.6 below.

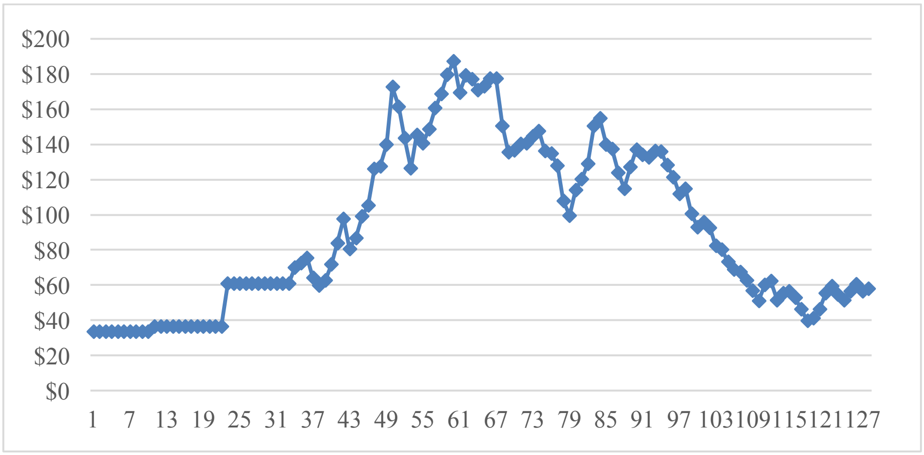


Figure 3.6 – Monthly Iron ore price – dollars per dry metric ton (Source: http://www.tse.ir/en/)

## 3.2. Correlation Analysis

Correlation analysis is performed to obtain a better understanding of the relationship between the identified variables of interest and the response. Note that these variables are selected as inputs due to potential predictive power over the response. Correlation among variables leads to multicollinearity problems in regression models, however, ANN and ANFIS models are not affected by collinear predictors. The following tables show the calculated correlation values and their corresponding P-Values.

Table 3.2 – Correlation values

| | production | crude oil | gold | alum | steel | copper | rubber | stock | lead | iron ore |
|---|---|---|---|---|---|---|---|---|---|---|
| production | 1.00 | -0.15 | -0.19 | 0.09 | 0.18 | 0.02 | 0.33 | -0.29 | 0.06 | 0.26 |
| crude oil | -0.15 | 1.00 | 0.085 | 0.36 | 0.75 | 0.79 | 0.69 | -0.08 | 0.50 | 0.65 |
| **gold** | **-0.12** | 0.08 | 1.00 | -0.60 | -0.18 | -0.12 | -0.15 | 0.91 | 0.03 | 0.27 |
| alum | 0.09 | 0.36 | -0.61 | 1.00 | 0.65 | 0.67 | 0.46 | -0.62 | 0.37 | -0.06 |
| **steel** | **0.18** | 0.75 | -0.18 | 0.65 | 1.00 | 0.94 | 0.89 | -0.36 | 0.60 | 0.68 |
| copper | 0.02 | 0.79 | -0.12 | 0.67 | 0.94 | 1.00 | 0.81 | -0.27 | 0.61 | 0.55 |
| **rubber** | **0.33** | 0.69 | -0.15 | 0.46 | 0.89 | 0.81 | 1.00 | -0.33 | 0.45 | 0.79 |
| **stock** | **-0.29** | -0.08 | 0.91 | -0.62 | -0.36 | -0.27 | -0.33 | 1.00 | -0.05 | 0.05 |
| lead | 0.06 | 0.51 | 0.03 | 0.37 | 0.60 | 0.61 | 0.45 | -0.05 | 1.00 | 0.33 |
| **iron ore** | **0.26** | 0.65 | 0.27 | -0.06 | 0.68 | 0.55 | 0.79 | 0.05 | 0.33 | 1.00 |

Based on the correlation values and their significance, Monthly gold, steel, iron ore and rubber prices alongside the stock index variables are demonstrating significant association in regards to the response at a $\alpha = 0.05$ significance level. Note that the monthly rubber price and the monthly stock index are the most highly correlated variables in regards to the response.

Table 3.3 – Correlation P-Values

| | production | crude oil | gold | alum | steel | copper | rubber | stock | lead | iron ore |
|---|---|---|---|---|---|---|---|---|---|---|
| production | 1.00 | 0.09 | 0.03 | 0.29 | 0.04 | 0.84 | <0.001 | <0.001 | 0.49 | 0.003 |
| crude oil | 0.09 | 1.00 | 0.35 | <0.001 | <0.001 | <0.001 | <0.001 | 0.38 | <0.001 | <0.001 |
| **gold** | **0.03** | 0.35 | 1.00 | <0.001 | 0.04 | 0.16 | 0.09 | <0.001 | 0.77 | 0.002 |
| alum | 0.29 | <0.001 | <0.001 | 1.00 | <0.001 | <0.001 | <0.001 | <0.001 | <0.001 | 0.52 |
| **steel** | **0.04** | <0.001 | 0.04 | <0.001 | 1.00 | <0.001 | <0.001 | <0.001 | <0.001 | <0.001 |
| copper | 0.84 | <0.001 | 0.16 | <0.001 | <0.001 | 1.00 | <0.001 | 0.002 | <0.001 | <0.001 |
| **rubber** | **0.0001** | <0.001 | 0.09 | <0.001 | <0.001 | <0.001 | 1.00 | <0.001 | <0.001 | <0.001 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **stock** | **0.0009** | 0.38 | <0.001 | <0.001 | <0.001 | <0.001 | <0.001 | 1.0000 | 0.57 | 0.54 |
| lead | 0.49 | <0.001 | 0.77 | <0.001 | <0.001 | <0.001 | <0.001 | 0.57 | 1.0000 | <0.001 |
| **iron ore** | **0.0034** | <0.001 | 0.002 | 0.52 | <0.001 | <0.001 | <0.001 | 0.54 | <0.001 | 1.0000 |

## 3.2. Cleanup & Diagnostic Procedures

This section demonstrates the data cleanup procedures performed on the data. The procedures are carried out to identify and resolve abnormalities that exist within the data set. Furthermore, data enhancement techniques are carried out to fill the missing values within the data set. The following criteria are necessary to observe to obtain a high-quality data set:

- Validity/Accuracy – Degree of conformity of the data to the conditions set.
- De-cleansing – Detecting and resolving abnormalities and errors.
- Completeness – Obtaining a complete and accurate data set.
- Uniformity – Utilizing sensible units of measures across different variables.

Data cleansing procedures present some challenges such as the loss of information and error correction problems. Given the vast range of data set types and challenges, it is impossible to administer a general data-cleansing procedure for data sets beforehand. As seen earlier, the data about the monthly production include missing data points, and thus requires remedial procedures for the missing points. To resolve this problem via statistical methods, the following procedure is carried out:

- Identifying missing values, and possible outliers.
- Calculating an average for the months with missing/abnormal values.
- Obtaining a weighted average for the missing/abnormal values.

Based on the collected data, values are missing for months: 2, 32, 37, 61, 63, 66, 85, 86, 95 and 97. Also, months 1, 13, 25 and 49 are considered as outliers after comparing with other data points and thus require remedial procedures alongside the months with missing values. A weighted average is calculated for each of the missing monthly production values to remedy the problematic data points. Furthermore, the following algorithm is carried out for each of the missing monthly values:

- Identify month with missing production value.
- Calculate an average for the missing month based on the actual values of the same month of different years, denoted by $\overline{y_\alpha}$.
- Identify two prior and two ahead monthly production values of the missing month and average the four production quantities, denoted by $\overline{x_\alpha}$.
- Calculate the weighted average: $\bar{X} = (0.5)(\overline{y_\alpha}) + (0.5)(\overline{x_\alpha})$

Figure 3.7 shows the final obtained graph of the total number of automobiles produced per period throughout the time horizon below.

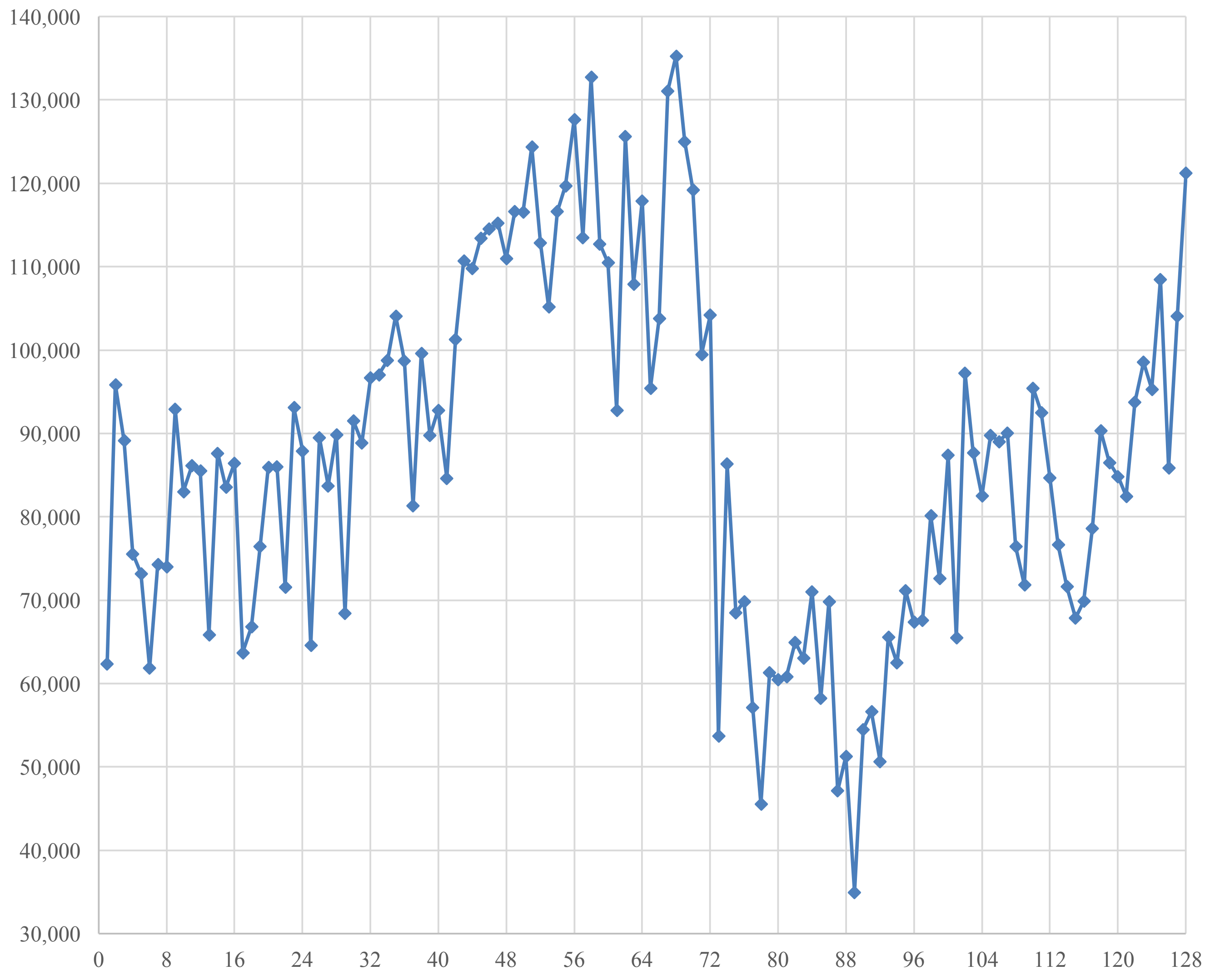


Figure 3.7 – Monthly production quantity of sedan vehicles in Iran (2006-2016)

# CHAPTER 4 | Methodology

Forecasting methods can be divided into quantitative and qualitative methods. In this chapter, several different quantitative methodologies are discussed and performance measures for the accuracy of forecasts are shown.

## 4.1. Definition of Forecasting

The most basic definition of forecasting is to predict or estimate a future event or trend. In other words, "Forecasting is a process that has an objective of predicting future events or conditions (Levenbach and Cleary, 2006)." As mentioned earlier, any decision-making problem may hugely benefit from an accurate and reliable forecast.

## 4.2. Necessity of Forecasting

Forecasts are quite necessary given that not only is the future uncertain for any given process but also that decisions made based on accurate forecasts are far more efficient compared to a decision that is not based on scientific forecasts. Economists and managers incorporate forecasts/predictions widely within their practice. It only makes sense that governments, industries, and financial institutions vastly benefit from the information provided by an accurate forecast. Predictions of the current state of the economy are thus a necessary input into the decision-making processes (Holden et al., 1990).

The accuracy of the forecasting model plays an imminent role in the decision making process. Forecasting models should provide a measure of accuracy for the utilized technique. Managers have high confidence in the precision of the report developed by forecasting techniques (Donovan, 1983). There amount of historical data required for an accurate model dictates that forecasts deal with average probabilities, as no other easy arrangement is possible or practicable (Wolfe, 1966).

“Depending on the decision being made, forecasts may be needed for points in time that are the number of days, weeks, months, quarters or years in the future.” The forecast period time/time horizon or period is the time duration in which the forecast is considered valid (Donovan, 1983).

### 4.3. Forecasting Methods

Forecasting methods can be typically divided into the following categories:

- Qualitative Methods
- Quantitative Methods

#### 4.2.1. Qualitative Methods

Qualitative forecasting methods gained much popularity during the 1960-70s. Some organizations in the early 1980s utilized methods such as the Delphi method and the cross-impact matrix method. These qualitative technological methods are capable of dealing with long-term trends wherein the historical data and patterns are essential to apply statistical forecasting methods, only were not ready or did not apply (Wheelwright and Makridakis, 1985).

The main purpose of qualitative methods is to collect consistent, unbiased data in a systematic order, relating to the factors of interest. Qualitative methods use schemes alongside human judgment to transform qualitative data into numeric estimations. These methods are usually used in prediction processes that are based on limited or restricted historical data. Market research, surveys, visionary forecasts, the Delphi method and historical analogies are among these methods (Levenbach and Cleary, 2006).

### 4.2.2. Quantitative Methods

All quantitative forecasting methods use the same principle: Historical data is analyzed to identify a pattern or trend that explains the process. After the pattern is determined, it is extrapolated into the future to obtain predictions. This principle heavily depends on the assumption that the identified pattern will not change its behavior and continues to behave in the same manner as it has in the past. Any quantitative forecasting method that does not encompass the latter assumption cannot provide accurate and reliable forecasts. Due to this assumption, qualitative forecasting methods are more precise within a short period, compared to long time frames. Quantitative forecasting methods can be divided into the following groups:

- Time Series
- Causal
- Neural networks
- Neuro-fuzzy systems

In the causal models, all different variables related to the forecasting variable are identified to construct the model, which is later used to forecast the primary variable. The main types of causal models are:

- Multiple Regression: The relationship between the dependent and independent variables are demonstrated through the regression equation.
- Econometrics: This method utilizes a system of interrelated regression equations. Furthermore, regression analysis is performed to obtain estimates of variable coefficients.
- Multivariate Box - Jenkins: Existing independent variables are the dependent variables to be forecasted in the Box-Jenkins model. This model attempts to relate the independent variables to the dependent variables via transfer functions (Donovan 1983).

Box-Jenkins methodology, alongside Artificial Neural Networks and Neuro-Fuzzy system, are described in the following sections.

### 4.2.2.1. Time Series Forecasting

#### 4.2.2.1.1. Introduction

A succession of data points in succeeding order, usually occurring in uniform intervals is referred to as a time series. In some time-series, the value of the studied variable can be estimated at any given moment of time. These time series are referred to as continuous time series, for example, temperature or profit rates. On the contrary, discrete time series consist of observations made at predetermined, equal interval time points to obtain hourly, monthly, quarterly or yearly observations (O'Donovan, 1983). Note that most time series models pertain to discrete time series models.

Time series data are most useful in forecasting processes that change over time. The primary purpose of forecasting based on time series data is to predict the sequence of observations in the future. Time series forecasting utilizes information about the variable of interest only and does not discover influencing factors within the time series. The prior constitutes that the forecast model will extrapolate the identified trend while ignoring other factors that may affect the process.

#### 4.2.2.1.2. Time Series Characterization

Any given time series is a sequence of observations obtained on the variable of interest. Furthermore, the variable of interest is observed at discrete time points, usually equally spaced. $x(t)$ denotes the observation for period $t$.

#### 4.2.2.1.3. Box-Jenkins Models

In 1970, two statisticians named George Box and Gwilym Jenkins applied autoregressive moving average (ARMA –ARIMA) models to obtain the best-fitted model of a time series model based on the past value of a time series. The suggested model by Box & Jenkins used an iterative three-stage modeling approach as follows (Box Jenkins, 1970):

1. <u>Model Identification/Selection:</u> The seasonality in the dependent series is identified assuming stationary variables. Further, plots of the autocorrelation and partial autocorrelation functions of the dependent time series are used to choose the autoregressive/moving average component.

2. Parameter Estimation: The parameters of the time series are estimated using computation algorithms to obtain the best-fitted coefficients in the selected ARIMA model. Methods include maximum likelihood estimation or non-linear least-squares estimation.

3. Model Validation: Is performed by testing the conformity of the estimated model to the stationary univariate process. Moreover, residuals are assumed independent and constant in mean and variance throughout time. Misspecifications can be identified by plotting the mean/variance of the residuals over time and performing a Ljung-Box test or by plotting the autocorrelation and partial autocorrelation of the residuals. If the estimation is deemed to be inadequate, the algorithm returns to the first step to readjust the coefficients.

In the development of the Box-Jenkins model, the first step is to determine whether the time series is stationary, also if there is any seasonality within the data which needs to be modeled. A run sequence plot is utilized to determine stationarity, which should demonstrate constant location and scale. Stationarity can also be detected from an autocorrelation plot. Non-stationarity is indicated by a plot with slow decay. Moreover, the seasonality can be detected from an autocorrelation plot.

After identifying whether the time series is stationarity or whether it suggests indications of seasonality, the order of the autoregressive and moving average terms (i.e. $p$ and $q$) are identified. Next, the model parameters are estimated. The main approaches to Box-Jenkins models fitting are maximum likelihood estimation and nonlinear least-squares. The likelihood equations for the full Box-Jenkins model were demonstrated by Brockwell and Davis (1991).

Consider a time series in which successive observations are represented by a linear combination of independent random variables. These variables have a distribution with a mean equal to zero and a variance equal to $\sigma_\varepsilon^2$. Note that if $\varepsilon_t$ has a normal distribution, it is defined as white noise. Furthermore, the linear combination of the error variable $\varepsilon_t$ is defined as demonstrated below in Equation 4.1.

$$x(t) = \mu + \sum_{j=0}^{\infty} \psi_j \varepsilon_{t-j} \quad with \;\; \psi_0 = 1 \tag{4.1}$$

Now the “backward shift” operator is introduced as B, such that:

$$B^j \varepsilon_t = \varepsilon_{t-j} \qquad 4.2$$

Equation 4.1 may be written as the following using the previous notation, which is called a “Linear Filter.”

$$x(t) = \mu + (\psi_0 B^0 + \psi_1 B^2 + \cdots)\varepsilon_t \qquad 4.3$$

Given the above, a time series is a function that transforms a white-noise process into a series. Models that are derived from Equation 4.3 are capable of representing both stationary and nonstationary time series.

Assume that the time series generated from Equation 4.3 is stationary. This indicates that the statistical properties of the series are not affected by time shifts, i.e. the properties of $n$ observations at origin $t$: $x_t, x_{t+1}, x_{t+n-1}$ are identical to those of $n$ observations at origin $t + k$: $x_{t+k}, x_{t+k+1}, x_{t+k+n-1}$. For a stationary time-series, we have:

$$E(x(t)) = E\left(\mu + \sum_{j=0}^{\infty} \psi_j \varepsilon_{t-j}\right) = \mu \qquad 4.4$$

The variance for a stationary process is as follows, note that the variance exists only if $\sum_{j=0}^{\infty} \psi_j^2$ convergers.

$$\gamma_0 = \sigma_\varepsilon^2 \sum_{j=0}^{\infty} \psi_j^2 \qquad 4.5$$

The covariance between $x_t$ and another observation separated by $k$ units of time $x_{t+k}$ is denoted as autocovariance at lag $k$:

$$\begin{aligned} \gamma_k &= Cov(x_t, x_{t+k}) \\ &= E[x_t - E(x_t)][x_{t+k} - E(x_{t+k})] \end{aligned}$$

$$\gamma_k = \sigma_\varepsilon^2 \sum_{j=0}^{\infty} \psi_j \psi_{j+k} \qquad 4.6$$

In the context of Box-Jenkins modeling, the autocorrelation is a useful statistic to utilize. In general, the correlation between two random variables $w$ and $z$ is defined as:

$$\rho_{wz} = \frac{Cov(w,z)}{\sqrt{V(w)V(z)}} \qquad 4.7$$

The autocorrelation at lag $k$ refers to the correlation between any two observations in a time series that are $k$ periods apart:

$$\rho_k = \frac{Cov(x_t, x_{t+k})}{\sqrt{V(x_t)V(x_{t+k})}} = \frac{\gamma_k}{\gamma_0} \qquad 4.8$$

A graphical representation of $\rho_k$ vs. $k$ is called the autocorrelation function. Note that $\rho_k$ is dimensionless, also $-1 \leq \rho_k \leq 1$, & $\rho_k = \rho_{-k}$. Now consider three random variables $w, y, z$ with a joint distribution $f\ (w, y, z)$. Then the conditional distribution of $w, y$ and $z$ are defined as:

$$h(w, y|z) = \frac{f(w,y,z)}{\iint_{-\infty}^{\infty} f(w,y,z)dwdy} \qquad 4.9$$

The correlation coefficient between $w$ and $y$ in the conditional distribution $h(w, y|z)$ is called the partial/conditional correlation coefficient. The partial correlation between $w$ and $y$ is the simple correlation between $w$ and $y$ with the effect of their correlation with $z$ removed. In the notation of this research, the $k^{th}$ partial auto correlated coefficient is referred to as $\phi_{kk}$. Note that $\phi_{00} = \rho_0 = 1$.

An important special case of Equation 4.1, is the following model, which is called the autoregressive process of order $p$, i.e. $AR(P)$:

$$x(t) = \xi + \left(\phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p}\right) + \varepsilon_t \qquad 4.10$$

The $AR(P)$ process can be written in terms of the backward-shift operator, $B$:

$$x(t) = \xi + \left(\phi_1 B^1 + \phi_2 B^2 + \cdots + \phi_p B^p\right)x_t + \varepsilon_t \qquad 4.11$$

Now if we let $\Phi(\mathrm{B}) = 1 - \phi_1 B^1 - \phi_2 B^2 - \cdots - \phi_p B^p$ then:

$$\Phi_p(\mathrm{B})x_t = \xi + \varepsilon_t \qquad 4.12$$

Furthermore, it is helpful to correct for the mean in this process. Let $\tilde{x}_t = x_t - \mu$ for all $t$ so the $AR(P)$ process can be written as:

$$\Phi_p(\mathrm{B})\tilde{x}_t = \xi + \varepsilon_t \qquad 4.13$$

Box-Jenkins have proven that the $AR(P)$ model is stationary if the roots of the polynomial $\Phi_p(\mathrm{B}) = 0$ lie outside the unit circle. In any case, $\sum_{j=0}^{\infty} \psi_j$ must converge for the process, $x_t$ to be stationary.

Another special case of Equation 4.1 occurs when the first $q$ weights are non-zero, which represents a moving average process of order q, that is:

$$x(t) = \mu + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - -\theta_q \varepsilon_{t-q} \qquad 4.14$$

In terms of the backward-shift operator, the MA(q) process is defined as:

$$x(t) = \mu + \theta_q(B)\varepsilon_t \qquad 4.15$$

$$E[x(t)] = \mu \qquad 4.16$$

$$Var[x(t)] = \gamma_0 = \sigma_\varepsilon^2 \sum_{i=0}^{q} \theta_i^2 \qquad with\ \theta_0 = 1 \qquad 4.17$$

The autocovariance of lag $k$ is as defined as follows:

$$\gamma_k = \begin{cases} \sigma_\varepsilon^2(-\theta_k + \theta_1\theta_{k+1} + \theta_2\theta_{k+2} + \cdots + \theta_{q-k}\theta_q & for\ k = 1,2,\dots,q \\ 0 & for\ k > q \end{cases} \qquad 4.19$$

The autocorrelation function is defined as:

$$\rho_k = \begin{cases} \frac{-\theta_k + \theta_1\theta_{\square+1} + \theta_2\theta_{k+2} + \cdots + \theta_{q-k}\theta_q}{1 + \theta_1^2 + \theta_2^2 + \cdots + \theta_q^2} & for\ k = 1,2,\dots,q \\ 0 & for\ k > q \end{cases} \qquad 4.20$$

### 4.2.2.2. Artificial Intelligence and Neural Networks (ANN)

Artificial Intelligence (AI) is referred to as the ability to acquire and apply knowledge/skills by machines or software. It is also referred to as the study and design of intelligent agents (Poole, Mackworth & Goebel, 1998). An intelligent agent is a system that perceives its environment and takes actions that maximize its chances of success (Luger & Stubblefield, 2004). The term was first introduced during the 1950's by John McCarthy, defined as "The science and engineering of making intelligent machines (McCarthy, 2007)."

The expert system, which was derived after the physical symbol system hypothesis supplied new motivations by the 1970s, arguably had the most commercial impact among the numerous effective techniques developed for limited regions. “Traditional AI researchers have been slow to welcome the connectionists, being skeptical of their claims and the premises underlying neural networks” (Finlay & Dix, 1996).

Artificial intelligence is a very broad idea that has applications in fields such as cognitive psychology, philosophy, mathematics, cybernetics and, computer science. Artificial Neural Networks (ANN) is an important branch of AI. The core objective of ANN is to construct a model similar to how the human brain functions. As a brief, neural network tasks in the human brain is defined as follows:

Neural networks consist of a vast number of simple processing elements called neurons, units, cells or nodes. Each of these neurons is related to others by directing communications links with a specifc weight associated to the link. A neuron sends its activation as a signal to several other neurons during this process (Sandberg et al., 2001). “In fact, a neuron conducts its signals via its axon that projects from its cell body, and it receives signals from other neurons over the connections between their axons and its dendrites.” These links are called synapses, and the connection cells are separated by a small gap about 200 Nano millimeters wide (Nauck et al.,1997). Figure 4.1 provides a simplified illustration of connected neurons.

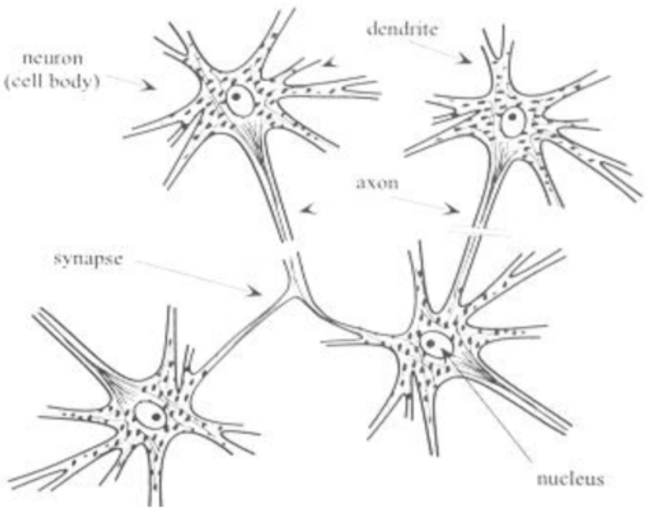


Figure 4.1 - Sketch of four connecting neurons (Re: Nauck et al., 1997)

Artificial neural networks are constructed according to mathematical models of neurons. It is possible to model some of the brain's task, such as making relations between patterns or the ability to save data in the memory, by setting the neurons into various configurations. Artificial neural networks have the capacity to learn complex tasks, which have proven difficult for rule-based systems (Picton, 2000). These networks are developed as generalizations of the mathematical models of human cognition or neural biology. The basic assumptions for this process are as follows:

- Information is processed through many simple elements/neurons.
- Signals pass between neurons over connections links.
- Each connection link has its designated weight. The weight of the link multiplies the transmitted signal.
- Each neuron applies an activity task to its net input (the sum of weighted inputs signals to the neuron) to determine its output signal.

Furthermore, any neural network can be characterized by the following:

- The architecture of the network: The pattern of connections between neurons.
- The training/learning algorithm of the network: The method of choosing the weights on connections between neurons.
- The activation function.

Note that there are several other classifications of neural networks such as layer classification and learning process classification.

#### 4.2.2.2.1. Learning Process Classification

In general, the learning process of neural networks may be classified into the following two classes:

- Supervised Learning: For the supervised learning neural network, the teacher compares the output of the network with the target/expected value. The primary goal is to reduce the difference between the forecasted output and the real observed value. This classification consists of three main types:

    - Error Correction Learning in which the network attempts to minimize the error (Kartalopoulos, 1996).
    - Reinforcement Learning in which the network only gives acceptable or non-acceptable outputs and is rewarded if the output is acceptable and punished if the output is not accepted (Nauck et al., 1997).
    - Stochastic Learning in which the network randomly selects weights for its connections makes changes in the weight values.

- Unsupervised Learning: In this classification, there are no target outputs. Neural networks with unsupervised learning do not receive information from the environment. This type of learning is mostly used for pattern recognition and clustering. The general types of unsupervised learning are as follows:
    - Hebbian Learning, in which the synaptic contact strength between two nodes, is modified according to the correlated activity degree between input and output information.
    - Competitive Learning, in which several neurons are located within the output layer, and each neuron competes with the others to generate the most accurate forecast output. As soon as the first neuron achieves the goal, all others subsequently fail. This type of unsupervised learning is suited for finding data clusters (Kartalopoulos, 1996).

#### 4.2.2.2.2. Layer Classification

Artificial neural networks can be classified into two main types based on the neural network's layers:

- Feed-forward networks: These types of networks consist of inputs, outputs, and hidden layers. The signal from the input feeds the next layer (output) in a single direction. This process continues until all signals have traveled to the output.

Feed-forward networks include the following types, further discussed in section 5.2.4:

  - Perceptron
  - Radial Basis Function (RBF)

- Feedback networks: In these types of networks, a portion of the output signals return to the input to modify its characteristics. These types of networks can be divided into the following types:

  - Adaptive Resonance Networks, which is a two layered, feedback network type. The best feature of this type is its ability to switch from a plastic mode to a stable mode, where the internal parameters of the network are modified to the designated parts for them without losing any information learned in the past. There is a circuit between the input and output layers to compare the inputs to a threshold that indicates whether a new class pattern should be created for the input pattern or not.

  - Hopfield Networks, which are weighted networks. In this type of network, each link connecting two neurons has the same weights in either direction. Hopfield networks have experienced limited commercial applications due to the relatively long time the process takes to be carried out. These types of networks have applications in the field of simulated annealing or improving the characteristics of crystals and metals. Feedback networks are often used in optimization and control systems (Picton, 2000).

Artificial neural networks have proved useful in some applications; however, many have great origin necessities (Finlay and Dix, 1996). Furthermore, the primary function of the neural network plays the most important role in deciding the correct type of utilized network.

#### 4.2.2.2.3. History & Development of Neural Networks

Neural network resources go as far back as the 1940s when the first mathematical pattern of a biological neuron was published by McCulloch and Pitts (Picton, 2000). In 1949, Donald O. Hebb demonstrated the structure of neural networks' learning process. It was during the 1950-60s that scientists were able to develop the first artificial neural network with the ability to learn based on Hebb's rule. Research papers on neural networks during the 1970s were primarily concerned with associative memory and neurophysiological models (Nauck et al., 1997).

Nowadays, researchers need to become familiar with a wider assortment of networks, all differing in network architecture, learning strategies and, weighting methods (Picton, 2000). Modern research in the field of neural networks, also referred to as connectionism, includes the development of new network architectures and learning algorithms. The research also tests the applicability of these newer models to information processing tasks (Nauck et al., 1997). Artificial neural networks have been utilized in a vast range of applications such as pattern classification, identification, optimization, prediction and automatic control. Despite different structures and training paradigms, all neural network applications are special cases of vector mapping (Tarafdar and Kashtiban, 2005). They have also been widely touted as solving many forecasting problems (Marque et al., 1992). The well-known task approximator in predicting and system modeling has lately shown great applicability in time series analysis and forecasting (Kamruzzaman & Saker, 2003).

#### 4.2.2.2.4. Artificial Neural Network Types

As mentioned earlier, there are several different types of networks with diverse applications. However, based on the research goal, a feed-forward network type is selected for this research. Feed-forward networks can be divided into Perceptron and Radial Basis Function (RBF) networks.

Perceptrons are divided into single-layer and multi-layer perceptrons. The single-layer perceptron is used for simple linear problems whereas the multi-layer perceptron is utilized for complex and nonlinear problems. Note that the multi-layer perceptron has the advantage of solving problems, compared to the single-layer perceptron. In addition to technical documents, in practical figures, the most important model that is used in over 90% of all neural networks is the multi-layer perceptron trained by the backpropagation learning algorithms (Nauck et al., 1997). Researchers have indicated that the multi-layer perceptron may be utilized as a universal approximator.

The multi-layer perceptron was first introduced in 1958 by Frank Rosenblatt. A multi-layer perceptron has an input layer of source nodes and an output layer of neurons (i.e. computations nodes). The two layers link the network to the environment. In addition to these layers, the multi-layer perceptron usually has at least one layer of hidden neurons that are not directly accessible. The hidden neurons take important characteristics contained in the input information (Sandberg et al., 2001). Also note that multi-layer perceptron (MLP) networks are feed-forward networks with many layers that are typically trained with backpropagation (Aris & Mohammad, 2008). Figure 4.2 illustrates the schematic layers of a perceptron.

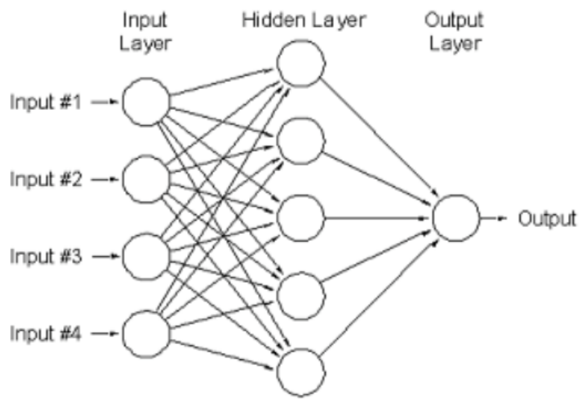


Figure 4.2 - Schematic Representation of a Perceptron (Re: http://mines.humanoriented.com/)

Multi-layer perceptron and backpropagation are often used for supervised learning to minimize the prediction error based on the training dataset. The backpropagation indicates a backward propagation of error through the network (Nauck et al., 1997). This process propagates errors through the network from the output layers towards the input layers during the training phase. This is necessary because hidden nodes have no training target value and these hidden nodes are trained based on error values from outer layers. The output layer is the only layer, which has a target value within this process.

By back propagating the errors through the network, original weights are adjusted. The training process is continued until the observed values for the errors in the weights are small enough to be accepted. The type of activation task used in neural network nodes can be a factor of the type of data being learned. The training algorithm may differ depending on the network architecture; however, the most common training algorithm used to design networks is the backpropagation algorithm (Lawrence, 1997). The backpropagation algorithm is schematically shown in figure 4.3.

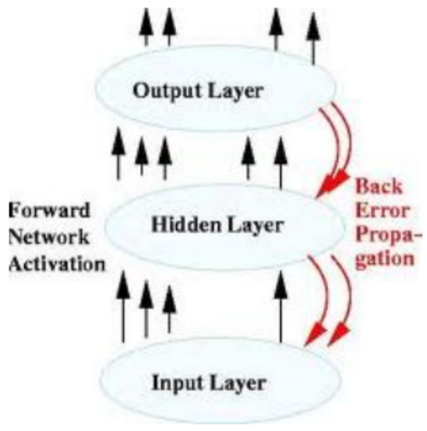


Figure 4.3 - The Scheme of Backpropagation algorithm (Re: http://www.cs.bham.ac.uk/)

Backpropagation learning is performed in one of the following major modes:

- Sequential Mode (Online or Pattern Mode): Adjustments are made to the free parameters of the network on an example-by-example basis in this mode. This mode is best suited to model classification.
- Batch Mode: Similar to the Sequential Mode, adjustments are made to the free parameters of the network on an epoch by epoch basis, where each epoch consists of the entire set of training examples. This mode is best suited for nonlinear regression (Sandberg et al., 2001).

Another important type of neural networks is the Radial Basis Function network. The RBF network is a multi-layer feed-forward neural network, which uses different transfer functions compared to the multi-layer perceptron type. The inspiring idea for RBF network creation is derived from traditional statistics. It is a linear combination of radial basis functions consisting of an input layer, a hidden layer, and one output layer. Note that the connection between the input and hidden layers is not weighted. Its applications are generally in function approximation, time series forecasting and controlling (Nauck et al., 1997). Table 4.1 provides a comparison between the two discussed network types.

Table 4.1 - Perceptron vs. RBF Network

| **Network Type** | **Network Features** |
|---|---|
| Perceptron | • At least one or more hidden layers<br>• Based on the Sigmoid function<br>• +90% Application in ANN functions<br>• Popular in Network researches<br>• Multi-layered feed forward progress<br>• Weighted connections between layers<br>• Learning algorithm: Backpropagation |
| Radial Basis Function | • Only one hidden layer<br>• Based on the Gaussian function<br>• Requires hidden neurons<br>• Applications in forecasting, controlling & function approximation<br>• Multi-layered feed forward<br>• No weighted connections between input and hidden layers<br>• Learning algorithm: Backpropagation |

Based on the prior presented information, also in line with the goals of this research, the Perceptron network is selected. The main concern in neural networks is to improve the performance of the network. To improve the perceptron learning algorithm with backpropagation learning, faster learning algorithms are suggested. Learning algorithms such as Quasi-Newton, Conjugate Gradient and Levenberg-Marquardt are examples of fast learning algorithms.

The back-propagation algorithm has many variations; however, the simple backpropagation and momentum are among the most important variations. Table 4.2 shows the features of backpropagation learning algorithms alongside faster algorithms, derived from neural networks.

Table 4.2 – Learning algorithm features

| **Learning Algorithm** | **Algorithm Features** |
|---|---|
| Simple backpropagation | • Weights are adjusted in the steepest descent direction (negative of the gradient) |
| Backpropagation with Momentum | • Refined gradient step<br>• Faster results compared to simple gradient descent<br>• Slow progression for practical application |
| Quasi-Newton | • Often converges faster than conjugate gradient method<br>• Not efficient for feed-forward neural networks |
| Conjugate Gradient | • Faster convergence than the steepest descent direction<br>• Better learning rates than simple backpropagation |
| Levenberg-Marquardt | • Fastest convergence rate among various algorithms<br>• Most accurate method among algorithms<br>• Presents lower squared errors among algorithms<br>• Training algorithm for small and medium size networks |

Table 4.2 shows each of the different learning algorithm's features. Based on the latter information, the Levenberg-Marquardt learning algorithm is chosen as the training algorithm for the network in this thesis.

Transfer functions play a significant role in network structures. This function determines the activation value, which is output to the rest of the networks; There are several types of transfer functions for neural networks. However, the Sigmoid and Tangent function (Tan-Sig) is widely utilized for hidden layers. Moreover, the linear transfer function is used for the output layer, employed in backpropagation networks. When linear output neurons are used, the network outputs can take on any value, because if the sigmoid transfer function is used for output neurons, the outputs are limited to a small range (Nauck et al., 1997).

#### 4.2.2.3. Neuro-Fuzzy Systems

Fuzzy logic was introduced by Lotfi A. Zadeh in 1965. Fuzzy logic is primarily based on the idea of fuzzy sets, which could be utilized to model linguistic terms (i.e. human expressions such as small, large, etc.). "In fuzzy logic, it is possible to formulate fuzzy rules that incorporate linguistic expressions, and apply the rules to decision-making processes." Similar to neural networks, fuzzy systems are utilized for a wide range of purposes from control functions to data analysis and decision making problems.

Fuzzy systems can be used for the same purposes as neural networks. The difference between the two is that fuzzy systems are constructed on explicit information, which is ultimately expressed in the form of linguistic rules rather than being built by a learning algorithm. Due to this difference, applying learning algorithms to a fuzzy system presents its difficulties. However, because both neural networks and fuzzy systems became popular during the late 1980s, the two were combined to form neuro-fuzzy systems.

Fuzzy systems and neural networks are both prevalent tools in the soft computing area. Lotfi A. Zadeh coined the soft computing term, which consists of approaches to human reasons that make use of human bearing for incompleteness, uncertainty, imprecision, and, fuzziness in decision-making processes. Soft computing is mainly concerned with combinations of fuzzy systems, neural networks, evolutionary computation, and, probabilistic reasoning.

Combining fuzzy systems with neural networks would constitute an interpretable model, which is capable of learning and, can use problem-specific prior information (Nauck et al., 1997). The different types of combinatory systems are as follows:

- Fuzzy Neural Networks: In this type of combinatory system, fuzzy methods are employed to improve the learning capabilities or the performance of a neural network using fuzzy rules to change the learning rate or by the construction of networks that work with fuzzy inputs.
- Concurrent Neural/Fuzzy Systems: For the concurrent neural/fuzzy systems, the neural network and fuzzy system both have the same task. Often the neural network preprocesses the inputs to the fuzzy system or post processes the outputs from the fuzzy system.
- Cooperative Neuro-Fuzzy Models: In this type of system, a neural network is utilized to define the fuzzy system's parameters. The fuzzy system will progress without the neural network after the learning phase. These systems are a simple sub-category of neuro-fuzzy systems which have applications in commercial fuzzy development tools
- Hybrid Neuro-Fuzzy Models: This type of system utilizes a structure comprised of a neural network and a fuzzy system. Modern neuro-fuzzy tools are in fact hybrid neuro-fuzzy models. ANFIS is classified in this model (Nauck et al., 1997).

#### 4.2.2.3.1. Adaptive Neuro-Fuzzy Inference System Model (ANFIS)

The Adaptive Neuro-Fuzzy Inference System (ANFIS) model was introduced in the early 1990s (Jang et. al, 1993). This model was initially developed by Sugeno and Kang and was developed further by Jang. The selection of a special neuro-fuzzy architecture is dependent on the reachable historical data. The ANFIS system used in this thesis consists of the following five function blocks:

1. A rule base is fuzzy if-then rule is utilized.
2. A dataset, which defines the membership function of the fuzzy sets employed in the defined fuzzy rules.

3. A decision-making unit, which is responsible for performing inference operations on the fuzzy rules.
4. A fuzzification inference, which transforms the hard inputs into degrees of the match with linguistic values.
5. A defuzzification inference, which, transforms the results into a hard output.

Figure 4.4 shows the ANFIS functional blocks described above.

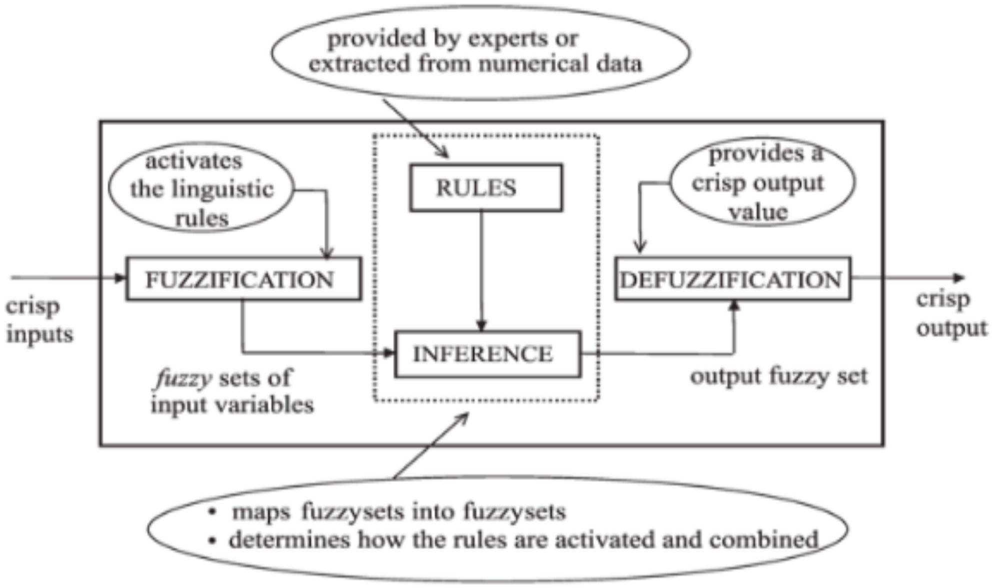

provided by experts or extracted from numerical data
activates the linguistic rules
RULES
provides a crisp output value
crisp inputs
FUZZIFICATION
DEFUZZIFICATION
crisp output
INFERENCE
fuzzy sets of input variables
output fuzzy set
• maps fuzzysets into fuzzysets
• determines how the rules are activated and combined

Figure 4.4 - ANFIS Functional Blocks (Re: www.scielo.br.gif)

A fuzzy inference system, employing fuzzy if-then rules can model the qualitative features of human information and reasoning processes without employing precise quantitative analyses (Shing & Jang, 1993). Jang's ANFIS model is among the first hybrid neuro-fuzzy systems that were utilized for function approximations. The model shows a Sugeno-type fuzzy system in a special five-layered feed-forward network architecture. A primary section of ANFIS models are the fuzzy if-then rule is as follows:

$$R_r\colon if\ \chi_i\ is\ A_{j1}^{(r)} \dots \chi_n is\ A_{jn}^{(r)}\ \ then\ y = a_0^{(r)} +\ a_1^{(r)}\chi_1 + \cdots + a_n^{(r)}\chi_n \quad 4.21$$

The ANFIS model adjusts the membership functions of previous and consequent factors. Note that the mentioned rule only utilizes a single output. Each output requires a specific extra linear combination, by using an extra set of consequent factors for each rule. The following can be extended to ANFIS models with multiple output variables:

- ANFIS network structures contain ***n*** units in the initial layer $U_0$
- Other layers $(U_1, U_2, \dots, U_5)$ present the following functions:
    - Layer 1: Each unit in layer $U_1$ stores three parameters to define a bell-shaped membership function that represent the following linguistic term:

$$\mu_j^i(\chi_i) = \frac{1}{1+[(\frac{\chi_i - c}{a})^2]^b} \qquad 4.22$$

Where $\chi_i$ is an input variable. Each unit is precisely related to only one input unit and computes the membership degree of the input value gained.

    - Layer 2: Each rule is shown by one unit within $U_2$. Each unit is linked to the previous layer units that come from the antecedent of the rule. The inputs into a unit $R_r \in U_2$, are degrees of membership that are multiplied to determine the fulfillment degree $T_r$ for the represented rule $R_r$.
    - Layer 3: The relative degree of completing is computed for each rule $R_r$ within this layer by using the following equation:

$$\overline{T_r} = \frac{T_r}{\sum_{R_i \in U_2} T_i} \qquad 4.23$$

Note that each unit is connected to all the rule units in $U_2$.

    - Layer 4: The units of layer four $U_4$ are connected to all input units and precisely one unit in layer three. Each unit computes the output of a rule $R_r$using the following:

$$O_r = \overline{T}_r(a_0^{(r)} + a_1^{(r)}\chi_1 + \cdots + a_n^{(r)}\chi_n) \qquad 4.24$$

    - Layer 5: An output unit is responsible for computing the final output $y$ through summing all outputs coming out of layer 4.

Because ANFIS only employs differentiable functions, learning algorithms from the neural network theory are easily applied. The combination of backpropagation and least squares estimation (LSE) is used in ANFIS by default. To obtain the membership function (i.e. the antecedent parameters), backpropagation and LSE are utilized to determine the coefficient of the linear combinations in the rules consequents. The learning process is divided into two primary parts:

- Part I: Input patterns are propagated, and the optimal consequent parameters are estimated by an iterative least mean squares method, while the antecedent parameters are theoretically assumed to be fixed for the current cycle according to the training set.
- Part II: The patterns are propagated again, and the epoch backpropagation is utilized to modify antecedent parameters, while consequent parameters remain fixed. Moreover, this method is iterated for the learning process.

Jang's learning process is composed of the following four steps:

1. All models from the training set are propagated, and consequent factors are determined by iterative LSE. Note that the antecedent parameters remain fixed.
2. All models are propagated again, and antecedent factors are updated through backpropagation.
3. If the calculated error decreases within four consecutive steps, this constitutes an increase in the learning rate (10% increase). If the calculated error is subject to consecutive combinations of increase and decrease in the error, then the learning rate will decrease by 10%.
4. The learning process is terminated when the calculated error is negligible. Otherwise, the process will continue to progress.

When the number of factors in a given situation is large, the choice of consequent factors may be costly. Moreover, to prevent high costs, backpropagation may be utilized for these factors. Although, for the demonstrated algorithm above, Jang's "Hybrid learning rule" provides more accurate results compared to other algorithms.

The ANFIS framework ensures that each different linguistic term is illustrated by only one fuzzy set. However, this learning procedure will not provide the opportunity to apply constraints to the membership functions (constraints that limit modifications applied to the membership function). ANFIS is only able to realize Sugeno type fuzzy systems, thus making it difficult to interpret the learning procedure outcome. This constitutes that ANFIS is preferable for applications where performance is more important than description. Moreover, ANFIS is utilized best for problems wherein the space structure can be interpreted (i.e. Situations where specifying fuzzy states are acceptable). “These fuzzy states are used in the antecedents for the rules represented within ANFIS.” Note that the consequent factors may be initialized with random values and the learning procedure ultimately determines the final values for these factors. Also, ANFIS has the capability of constructing a neural network structure for a Sugeno fuzzy system, which may help in implementing the system within a neural network environment (note: this capability is not essential for the application). Figure 4.5 shows the ANFIS structure.

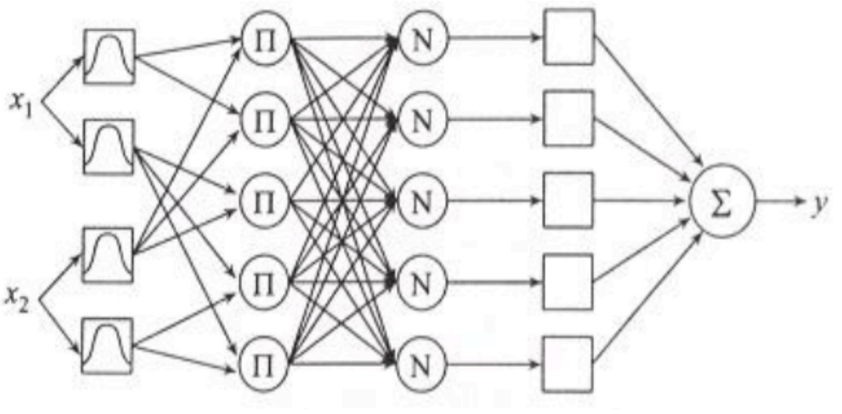


Figure 4.5 - ANFIS Structure (Re: Nauck et al. 1997)

An ANFIS system consists of $\boldsymbol{n}$ inputs, $\boldsymbol{k}$ rule units, and a single output unit. It is trained with a learning problem of $\boldsymbol{p}$ patterns. The computation of the output of such a system can be specified as:

$$y = \frac{\sum_{i=1}^{k} T_i y_i}{\sum_{i=1}^{k} T_i} = \sum_{i=1}^{k} \overline{T}_i y_i = \sum_{i=1}^{k} \overline{T}_i \left(a_0^{(i)} + a_1^{(i)} \chi_1 + \cdots + a_n^{(i)} \chi_n\right) \quad 4.25$$

Where $T_r = \prod_{i=1}^{n} \mu_{jr}^{i}(\chi_i)$ is the fulfillment degree of rule $R_r$ and $\overline{T}_r = \frac{T_r}{\sum_{i=1}^{n} T_i}$ is the normalized fulfillment degree. Then linear expression for $y$ indicates that the consequent parameters $a_i^{(r)}$are linear as well. Thus the parameters can be estimated by Least Squared error method (LSE).

Let $N$ be a matrix containing one row per each pattern of the training set. Each row consists of $k$ repetitions of $(1, \overline{T}_1^{(i)}, \dots, \overline{T}_k^{(i)})$, where $\overline{T}_j^{(r)}$ is the normalized fulfillment degree of rule $j$ after the $i^{th}$ pattern has been propagated. Moreover, let $T$ be the column vector of the target output values from the training set. Now let:

$$A = (a_0^{(1)}, \dots, a_n^{(1)}, \dots, a_0^{(k)}, \dots, a_n^{(k)})^{\,T} \qquad 4.26$$

Constitute the column vector of all consequent parameters of all rules. The consequent parameters are determined by the following matrix equation:

$$NA = T \qquad 4.27$$

With $k$ rule units, we have the following number of consequent parameters:

$$M = k(n+1) \qquad 4.28$$

Note: $M$ constitutes the size of the column vector of all consequent parameters.

Furthermore, $N$ is a $P{\times}M$ matrix. The size of $T$ is equal to $B$. Due to the general fact that we usually have more training patterns than parameters, i.e. $P$ is greater than $M$, the problem is over-determined and usually obtains no certain solution. To solve this problem, the least squares estimate, $A^*$, of $A$, is determined to minimize the squared error $[NA - T]^2$. The latter is done by:

$$A^* = (N^T N)^{-1}.NT \qquad 4.29$$

Where $N^T$is the transpose matrix of $N$ and is equal to $(N^T N)^{-1}$.

Moreover, $N$ is the pseudo-inverse of $N$ if $(N^T N)$ is a non-singular matrix. However, due to the computation expenses and inefficiencies resulted from singular matrices, an iterative LSE method is used by Jang to train the ANFIS which is demonstrated further below:

Let $n_i^T$be the $i^{th}$ row vector of matrix $N$ and let $t_i^T$ be the $i^{th}$element of vector $T$. Then a solution for $A$ can be computed iteratively by evaluating the following equations:

$$A_{i+1} = A + \Sigma_{i+1} n_{i+1} \left( t_{i+1}^T - n_{i+1}^T A_i \right) \qquad 4.30$$

$$\Sigma_{i+1} = \Sigma_i \frac{S_i n_{i+1} n_{i-1}{}^T \Sigma_i}{1 + n_{i+1} n_{i+1}{}^T \Sigma_i} \qquad for \quad i = (0,1,\dots,p-1) \qquad 4.31$$

Where $\Sigma$ is the covariance matrix and, $A^*$, the least squares estimate is equal to $A_p$.

The initial necessary conditions for this procedure are as follows:

- $A_0 = 0$ 4.32
- $\Sigma_0 = \gamma I_M$ 4.33

Where γ a large positive is number, and $I_M$is the identity matrix $(M \times M)$.

Note that if the ANFIS network has more than a single output, for example $l$, then $T$ is going to be a $(P \times l)$ matrix and, $t_i^T$is the row vector of$T$. Subsequently, matrix $A$ transforms into a $(M \times l)$ matrix.

Antecedent parameter modifications are determined by the backpropagation method. Let $P$ be a parameter of the fuzzy set $\mu_{j_r}^{(i)}$ based on the antecedent of an arbitrary rule $R_r$. The change in $P$ for a single rule $R_r$ after a pattern has been propagated is realized, where rule $y^*$ is defined as the target output value. The error measure, $E$ is the sum of squared differences between the target and actual output values. By the iterative application of the chain rule, the following equation is obtained as:

$$\Delta_p = -\delta \frac{\partial E}{\partial P} = -\delta \frac{\partial E}{\partial y} \frac{\partial y}{\partial \bar{T}_r} \frac{\partial \bar{T}_r}{\partial T_r} \frac{\partial T_r}{\partial \mu_j^{(i)}} \frac{\partial \mu_j^{(i)}}{\partial P}$$

$$= \delta (y^* - y) y_r \frac{\bar{T}_r (1 - \bar{T}_r)}{T_r} \frac{T_r}{\mu_{j_r}} \frac{\partial \mu_j}{\partial P}$$

$$\Delta_p = \frac{\delta}{\mu_{j_r}} y_r (y^* - y)\, \bar{T}_r (1 - \bar{T}_r) \frac{\partial \mu_j^{(i)}}{\partial P} \qquad 4.34$$

Where δ is the learning rate.

Note that the following is obtained for the three parameters of a fuzzy set $\mu_{j_r}^{(i)}$, for the last factor of the equation:

$$\frac{\partial \mu_{j_r}^{(i)}}{\partial a} = -\frac{\mu_{j_r}^{(i)} (\chi_i)^2}{a} \left(\frac{(\chi_i - c)^2}{a}\right)^b \qquad 4.35$$

$$\frac{\partial \mu_{j_r}^{(i)}}{\partial a} = -b\, \mu_{j_r}^{(i)} (\chi_i)^2 \left(\frac{(\chi_i - c)^2}{a}\right)^b \qquad 4.36$$

$$\frac{\partial \mu_{j_r}^{(i)}}{\partial a} = \frac{2b\, \mu_{j_r}^{(i)} (\chi_i)^2}{\chi_i - c} \left(\frac{(\chi_i - c)^2}{a}\right)^b \qquad 4.37$$

(Nauck et al., 1997)

An essential component of the ANFIS structure is the shape of the membership function. A membership function (MF) is defined as a curve that indicates how the input space is mapped to a membership value between 0 and 1. The simplest MF's are obtained through the usage of straight lines. The triangular membership function (function name: 'trimf') is the simplest of the membership functions. Another example of the simple membership functions is the trapezoidal membership function (function name: 'trapmf'), which has a flat top. Figure 4.6 shows the mentioned membership functions below.

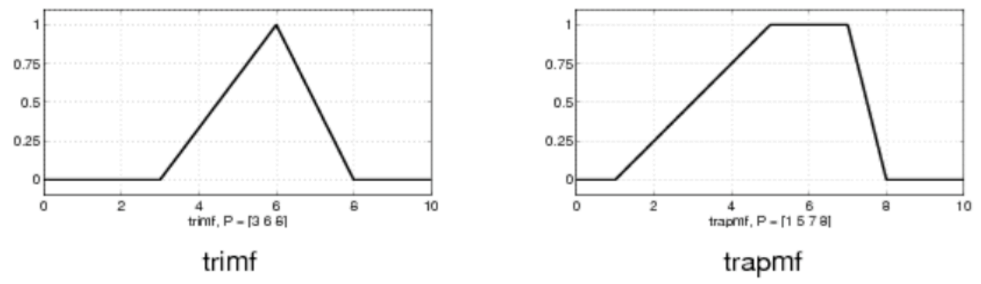


Figure 4.6 - Straight Line Membership Functions (Re: MATLAB help section)

Moreover, two membership functions contain the Gaussian distribution curve. The simple Gaussian curve and a two-sided composite of two different Gaussian curves are among the types of Gaussian distributions (function names: ‘gaussmf’ & ‘gauss2mf’).

Another function that has one more parameter than the Gaussian membership function is the generalized Bell membership function (function name: ‘gbellmf’) and thus providing the ability for this membership function to approach a non-fuzzy set, given the free parameter is tuned.

The Gaussian and bell membership functions are among the most common membership functions utilized to determine fuzzy sets. Figure 4.7 shows the general scheme of the Gaussian and Bell membership functions.

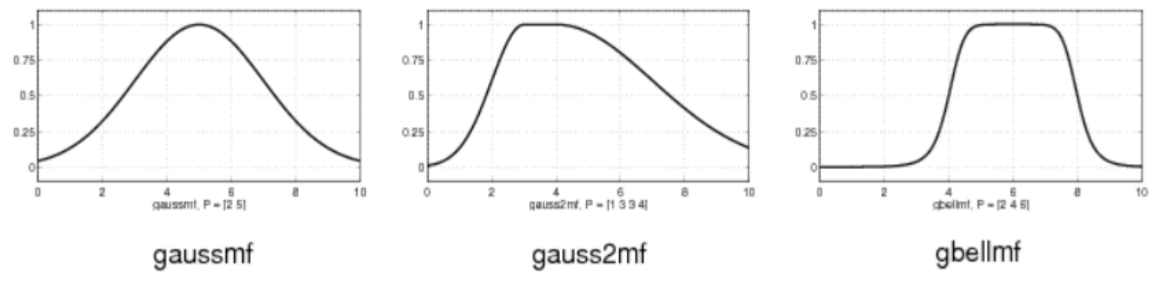


Figure 4.7 - Gaussian and Bell membership functions (Re: MATLAB help section)

It is important to note that the Gaussian and bell membership functions are not able to specify asymmetric membership functions. The following membership functions address this problem. The Sigmoidal membership function is a function, which is either open to the left or right. Figure 4.8 shows the sigmoidal membership function.

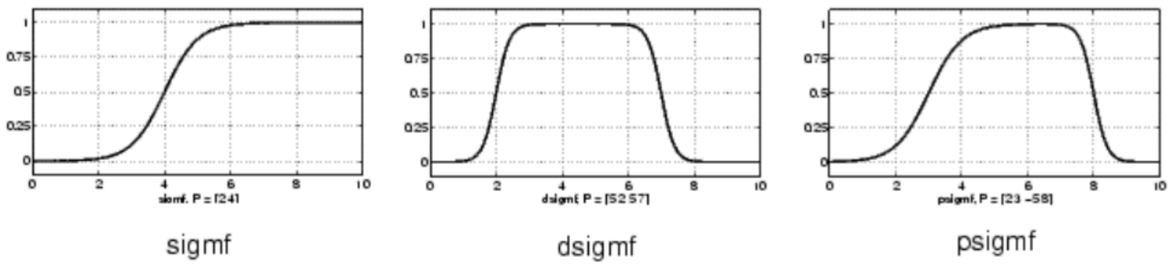


Figure 4.8 - Sigmoidal Membership functions (Re: MATLAB help section)

Another type of membership functions is the Polynomial based curves. The three different membership functions within this type are the ***Z***, ***S***, and ***P*** curves. The *'zmf'* function is an asymmetrical polynomial curve open to left. The *'smf''* function is the mirror image function, open to the right and the *'pimf'* function has a value of zero on both left and right extremes with a rise in between. The three different types of polynomial membership functions are presented in figure 4.9 below.

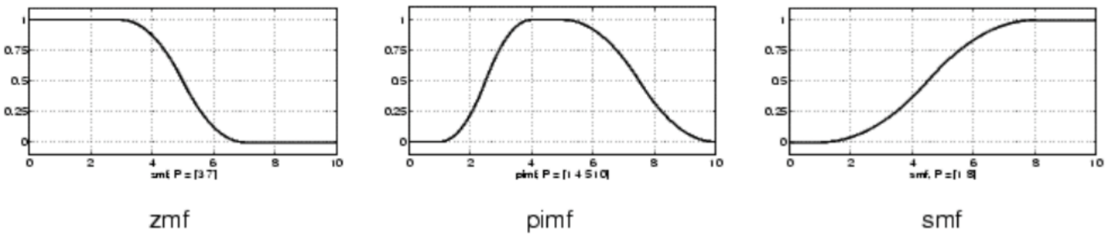


Figure 4.9 -Polynomial Membership functions (Re: MATLAB help section)

Furthermore, there is a broad range of membership functions available to utilize for ANFIS, as demonstrated before. Based on the discussion provided earlier also the experimental results provided in chapter 5, the trapezoidal membership function (function name: 'trapmf') is selected and implemented in the ANFIS structure for this research.

### 4.3. Forecasting Accuracy

In forecasting, accuracy refers to the model's ability to reproduce results similar to the observed historical data. There are several performance measures of the accuracy of forecasting models such as:

- Mean Absolute Percentage Error (MAPE): This performance measure is calculated based on the absolute error for each period within the time series and is defined in formula 4.38 as:

$$MAPE = \frac{(100)\sum_{i=1}^{n}|PE_i|}{n} \tag{4.38}$$

Where $PE_i = (actual - forecast)/actual.$

- Mean Squared Error (MSE): The mean squared error is obtained by squaring the value of each time interval error and computing the mean of those values. Note that SSE - Sum of Squared Errors is the sum of the squared difference between each real data point and its predicted point, defined in Equation 4.39 as:

$$MSE = \frac{SSE}{n} = \frac{\sum_{i=1}^{n} e_i^2}{n} \tag{4.39}$$

- Mean Percentage Error (MPE): Another measure of accuracy for forecasting models is the MPE, defined in Equation 4.40 as:

$$MPE = \frac{\sum_{i=1}^{n} PE_i}{n} \tag{4.40}$$

In this research, MAPE is used as the primary accuracy measure for a comparative description of the real and predicted data for both utilized models.

# CHAPTER 5 | Results

This chapter includes the forecasting models constructed and demonstrates the performance of each of them. The Box-Jenkins models, along with the linear, exponential growth and quadratic models is discussed first. Then the ANN model and the neuro-fuzzy ANFIS models are developed. The methodology for the development of each model is stated in the following sections.

As noted before, the core objective of this research is to obtain an accurate forecasting model for the demand for automobiles in Iran's domestic market. The monthly production data for vehicles manufactured in Iran from 2006 to 2016 was obtained for the main response and were used in the modeling exercises. Figure 5.1 shows the time series (first 116 months) of the production. Also, nine additional predictor variables, the monthly crude oil, gold, aluminum, steel, copper, rubber, lead, and iron ore prices along with the domestic stock index were also selected as possible inputs for the ANN and ANFIS models.

In order to be consistent with the developed forecasting methodologies, the 128 available data points are divided into the first 116 and the last 12 months. All investigated methodologies use the first set of data for model development. This decision was made based on the non-stationary nature of the time series data of the problem at hand, which could not be remedied without modifying the data set. Dividing the time horizon into two periods was also investigated, however, models provided poor results due to the small number of data points (72 and 56 months/points). The ANN and ANFIS models use 87 months for model development, 14 months for model testing, and 15 months for model validation purposes. The best identified ANN and ANFIS are then used to generate forecasts for the last 12 months.

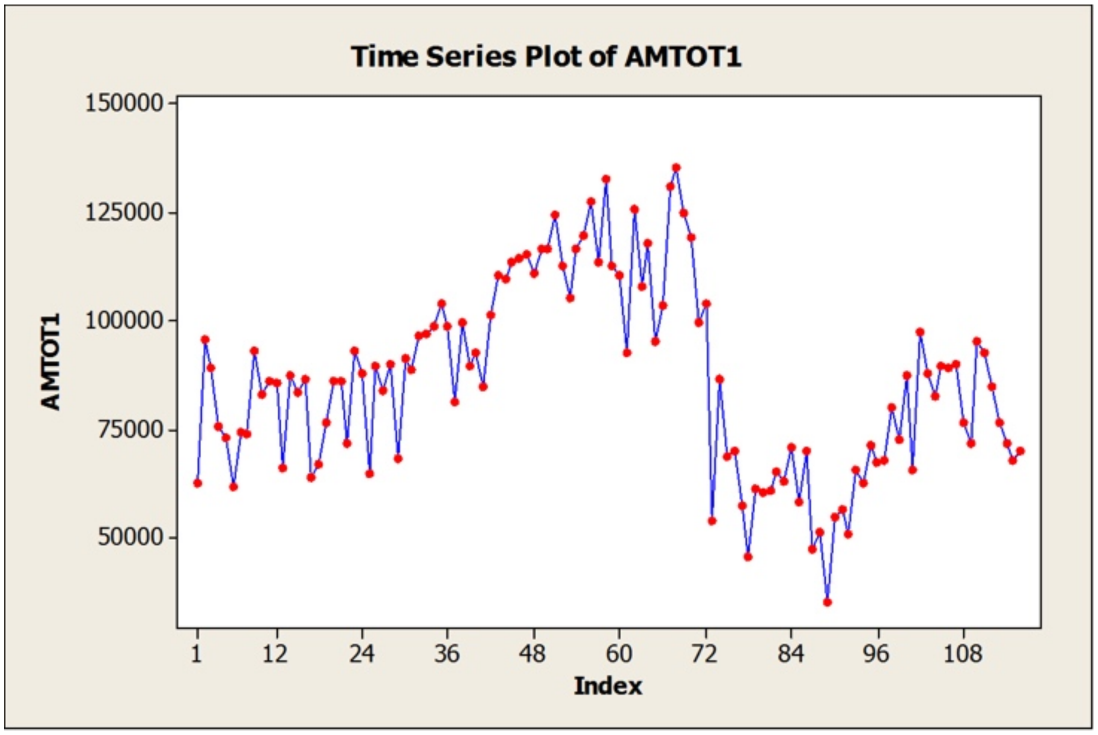


Figure 5.1 – Production time series, first 116 months (2006-2016)

Correlation analysis was carried out to detect any significant correlation among input variables to the ANN and ANFIS models. Based on the significance of the calculated correlations, five of the original predictor variables were identified as the inputs for the ANFIS models. They are monthly prices of gold, steel, rubber, iron ore, and the stock index while all nine variables were used in the ANN model.

Minitab was utilized to develop the Box-Jenkins, linear, exponential growth and quadratic models. MATLAB was used for the development of the ANN and ANFIS models. The performance metric selected for this research is the MAPE. The MAPE of the developed forecasting models is calculated and compared to identify the most accurate model for forecasting the demand for automobiles. The investigated models and results are provided in sections 5.1 through 5.7.

## 5.1. Box-Jenkins model

The Box-Jenkins methodology applies autoregressive moving average (ARMA, ARIMA) models to obtain the best-fitted model of the response based on historical data. The iterative three-stage modeling approach is carried out to identify the best performing model. The first 116 data points were set aside to construct the model, and the last 12 data points were designated to validate the model and calculate the MAPE. As previously mentioned, a large portion of the data points was allocated for model development purposes due to the non-stationary nature of the dataset. Furthermore, two general methodologies were carried out as follows:

- First, a Box-Jenkins model was fitted using the first 116 data points and was used to generate 12 monthly forecasts.
- Then the following process was carried out in order to establish a better comparison to the ANN and ANFIS models. A Box-Jenkins model was fitted to the first 116 data points and was used to generate a single time period forecast. Then, the generated forecast was used alongside the original 116 data points to fit a new Box-Jenkins model. This process was carried out to construct 12 models, each providing a single time period forecast for a total of 12 monthly forecasts.

The first step in the development of Box-Jenkins model is to determine whether the time series is stationary. To investigate the stationarity of the time series, the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots are utilized. Figures 5.2 and 5.3, which demonstrate the Autocorrelation function and Partial Autocorrelation function plots, indicate that the time series is non-stationary. The ACF plot of residuals shows deviance from the confidence bands at lag 2. Also, the PACF plot is deviating from the confidence band at lag 2 indicating that differencing by d=2 or 3 would remedy this problem. Note that a data point is lost each time the series is differenced.

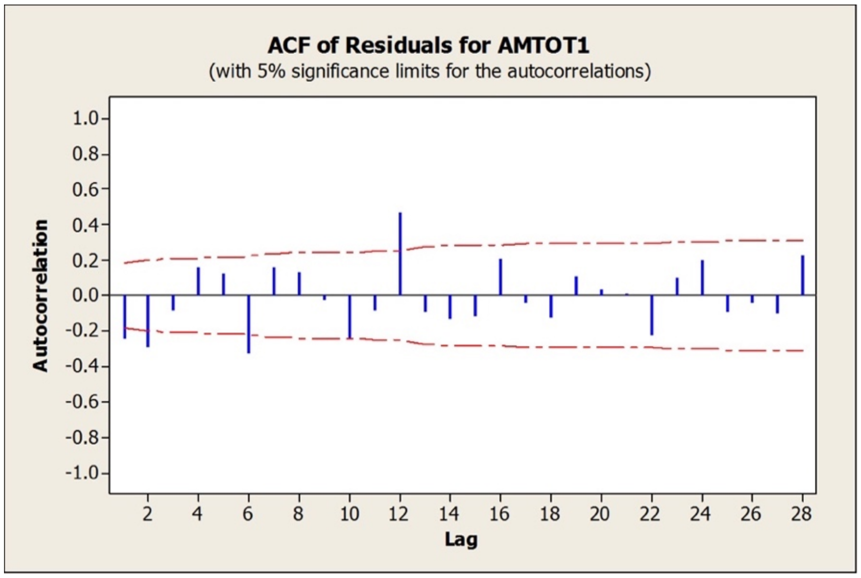


Figure 5.2 – Autocorrelation Function of Residuals

The second step in the Box-Jenkins methodology is parameter estimation. The parameters are estimated using computation algorithms to obtain the best-fitted coefficients in the selected ARIMA-ARMA model. This is done via maximum likelihood estimation or non-linear least-squares estimation. Several ARIMA-ARMA(*p,d,q*) models were developed and compared using Minitab. The ARMA (1,1,0) seasonal model, alongside the ARMA (1,3,1) nonseasonal model provided the lowest MAPE values among all the ARIMA models investigated.

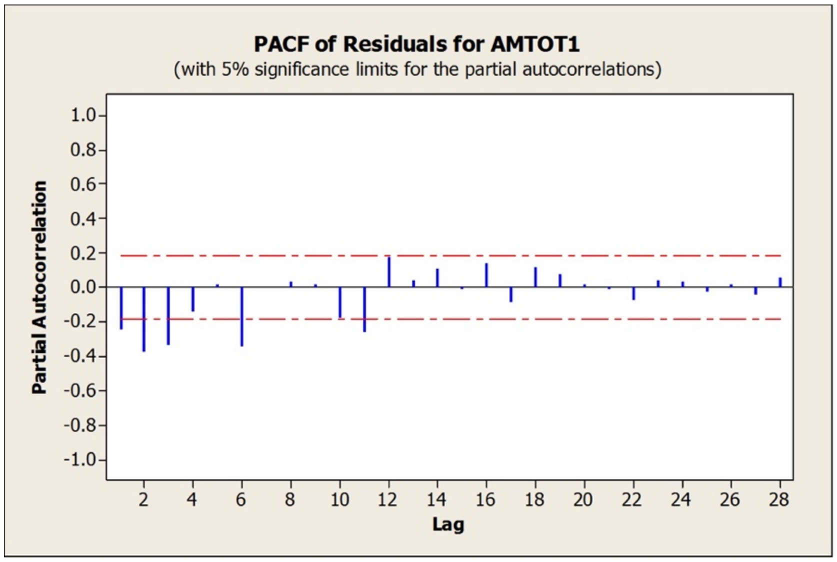


Figure 5.3– Partial Autocorrelation Function of Residuals

Moreover, the ARIMA (1,3,1) non-seasonal model is identified as the better performing model with MAPE=18.9963. Table 5.1 provides parameter estimates for the ARIMA (1,3,1) model and Table 5.2 show the modified Ljung-Box statistic. Note the significance of both model parameters.

Table 5.1 – ARIMA (1,3,1) parameter estimates

| **Type** | **Coef.** | **SE Coef.** | **T** | **P-value** |
|---|---|---|---|---|
| AR 1 | -0.74 | 0.0646 | -11.452 | <0.00 |
| MA 1 | 0.99 | 0.0077 | 128.54 | <0.00 |
| Constant | 34.60 | 33.57 | 1.03 | 0.305 |

As previously discussed, the AR(p) model can be written as:

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t \qquad 5.1$$

Where $\varphi_i's$ are model parameters, $c$ is a constant, and $\varepsilon_t$ are white noise. The MA(q) model can be written as:

$$X_t = \mu + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \varepsilon_t \qquad 5.2$$

Where $\theta_i's$ are model parameters, $\mu$ is the expectation of $x(t)$, and $\varepsilon_t$ are white noise. As previously discussed in chapter 4, the ARIMA (*p, d, q*) model has three parameters of interest. *p* represents the Autoregressive component order, *d* represents the order of differencing of the response to achieve a stationary process and *q* is the Moving Average component order. Note that the models are all fitted using Minitab. The P-value for the AR and MA term both validate the significance of the model components. In order to check the normality assumption, the normal probability plot of the residuals vs. fitted value plots and the residual histogram were constructed. As seen in Figure 5.4, the normality assumption is fairly met, and the residuals seem to be scattered randomly. The general model equation is obtained using the software output as:

$$\widehat{X_t} = 34.60 + (-0.74)X_{t-1} + (0.99)e_{t-1} + \varepsilon_{t-1} \qquad 5.3$$

The Ljung-Box test aims to identify whether any group of Autocorrelations of a time series are different from zero and tests the overall randomness based on a number of lags (h). The null hypothesis of the Ljung-Box test is that the data is independently distributed (correlation in the sample population is zero) and the test statistic follows a Chi-square distribution with h degrees of freedom (Ljung, 1978). The Ljung-Box P-Values indicate that the correlation in the sample population is not zero at any lag. The transformation of the response is investigated next.

Table 5.2 – Modified Ljung-Box statistic

| **Lag** | **12** | **24** | **36** | **48** |
|---|---|---|---|---|
| Chi-Square | 76.8 | 107.4 | 145.9 | 200.4 |
| DF | 9 | 21 | 33 | 45 |
| P-Value | <0.00 | <0.00 | <0.00 | <0.00 |

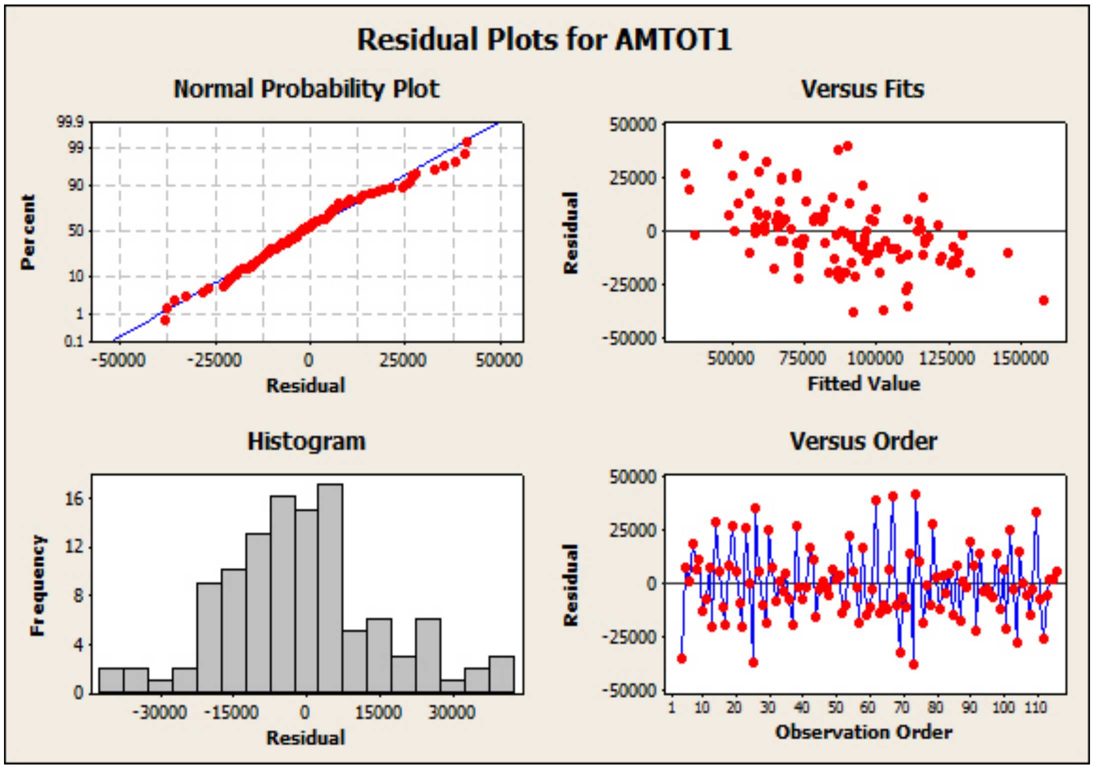


Figure 5.4 – ARIMA (1,3,1) nonseasonal model residual plots

Based on the plots shown in Figure 5.4, specifically, the residual vs. fitted value plot, the transformation of the response was investigated in order to improve the model. Appropriate transformations of the data often result in improvements in model performance. “The Box-Cox transformation is a useful transformation, defined as:

$$T(Y) = (Y^{\lambda} - 1)/\lambda \qquad 5.4$$

Where the response is denoted as Y and $\lambda$ is the transformation parameter. For $\lambda = 0$, the natural log of the data is taken instead of using the above formula. Given a particular transformation such as the Box-Cox transformation, it is helpful to define a measure of the normality of the resulting transformation. One measure is to compute the correlation coefficient of a normal probability plot. The correlation is computed between the vertical and horizontal axis variables of the probability plot and is a convenient measure of the linearity of the probability plot. The Box-Cox normality plot is a plot of these correlation coefficients for various values of the $\lambda$ parameter. The value of $\lambda$ corresponding to the maximum correlation on the plot is then the optimal choice of $\lambda$ (NIST/SEMATECH e-handbook of Statistical Methods, 2012)."

Figure 5.5 shows the Log-Likelihood vs. Lambda, which denotes the optimal transformation parameter.

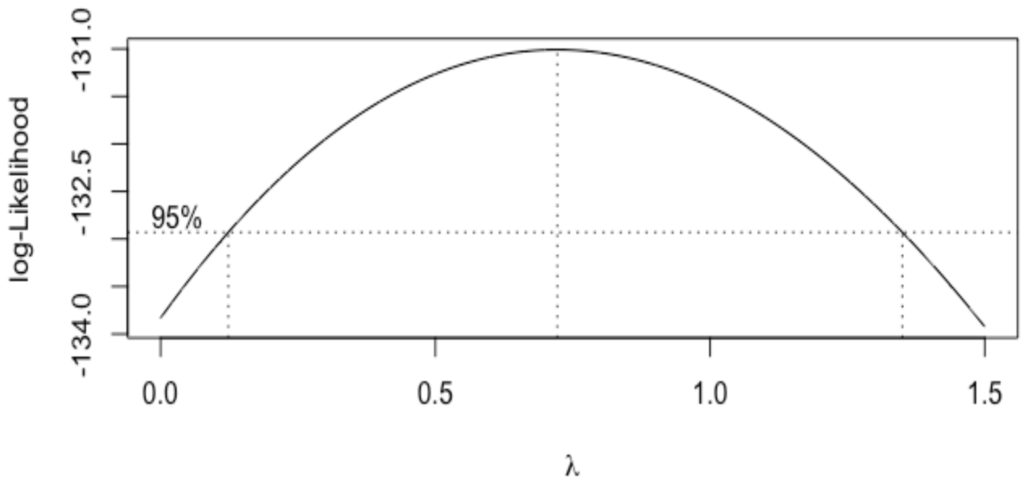


Figure 5.5 – Box-Cox transformation plot

As seen above, the 95% confidence interval for lambda includes the value of 1, indicating that transformations on the response will not enhance nor improve model performance. Finally, Table 5.3 provides the ARIMA (1,3,1) forecasts for the last 12 months, the actual values, and the MAPE=18.99 as follows:

Table 5.3 – ARIMA (1,3,1) nonseasonal model Forecasts and MAPE

| Actual | Forecast | APE |
|---|---|---|
| 78,642 | 69,614 | 11.48 |
| 90,378 | 73,072 | 19.15 |
| 86,508 | 75,823 | 12.35 |
| 84,847 | 81,189 | 04.31 |
| 82,481 | 86,749 | 05.17 |
| 93,739 | 94,326 | 0.63 |
| 98,568 | 102,608 | 4.10 |
| 95,314 | 112,599 | 18.13 |
| 108,489 | 123,591 | 13.92 |
| 85,880 | 136,143 | 58.53 |
| 104,052 | 149,876 | 44.04 |
| 121,272 | 165,104 | 36.14 |
|  | **MAPE =** | **18.99** |

Furthermore, as stated before, 12 different Box-Jenkins models were fitted as to establish a fair comparison to the ANN and ANFIS models. Note that each model was used to generate a single forecast. Table 5.4 provides the 12 forecasted values obtained using this methodology. Note the decrease in the calculated MAPE=9.68 compared to the general Box-Jenkins model MAPE (9.31% difference).

Table 5.4 – Box-Jenkins model(s) Forecasts and MAPE

| Actual | Forecast | APE |
|---|---|---|
| 78,642 | 69,614 | 11.48 |
| 90,378 | 73,355 | 18.84 |
| 86,508 | 74,926 | 13.39 |
| 84,847 | 80,265 | 5.4 |
| 82,481 | 82,805 | 0.39 |
| 93,739 | 89,038 | 5.01 |
| 98,568 | 92,481 | 6.18 |
| 95,314 | 99,562 | 4.46 |
| 108,489 | 103,919 | 4.21 |
| 85,880 | 111,867 | 30.26 |
| 104,052 | 117,157 | 12.59 |
| 121,272 | 125,996 | 3.90 |
|  | **MAPE =** | **9.68** |

The differential between the two methodologies is due to the robustness of the latter methodology as it only forecasted a time step ahead compared to 12 steps ahead of the former methodology. Furthermore, equations 5.4 through 5.15 provide each model's equation as follows:

$$\widehat{X_t} = 34.60 + (-0.74)X_{t-1} + (0.99)e_{t-1} + \varepsilon_{t-1} \tag{5.4}$$

$$\widehat{X_t} = 35.52 + (-0.74)X_{t-1} + (0.99)e_{t-1} + \varepsilon_{t-1} \tag{5.5}$$

$$\widehat{X_t} = 16.76 + (-0.73)X_{t-1} + (1.02)e_{t-1} + \varepsilon_{t-1} \tag{5.6}$$

$$\widehat{X_t} = 30.58 + (-0.73)X_{t-1} + (0.99)e_{t-1} + \varepsilon_{t-1} \tag{5.7}$$

$$\widehat{X_t} = 14.97 + (-0.74)X_{t-1} + (1.02)e_{t-1} + \varepsilon_{t-1} \tag{5.8}$$

$$\widehat{X_t} = 24.95 + (-0.74)X_{t-1} + (0.99)e_{t-1} + \varepsilon_{t-1} \tag{5.9}$$

$$\widehat{X_t} = 05.33 + (-0.74)X_{t-1} + (0.99)e_{t-1} + \varepsilon_{t-1} \tag{5.10}$$

$$\widehat{X_t} = 24.06 + (-0.73)X_{t-1} + (0.99)e_{t-1} + \varepsilon_{t-1} \tag{5.11}$$

$$\widehat{X_t} = 06.69 + (-0.73)X_{t-1} + (0.99)e_{t-1} + \varepsilon_{t-1} \tag{5.12}$$

$$\widehat{X_t} = 23.60 + (-0.73)X_{t-1} + (0.99)e_{t-1} + \varepsilon_{t-1} \tag{5.13}$$

$$\widehat{X_t} = 06.91 + (-0.73)X_{t-1} + (0.99)e_{t-1} + \varepsilon_{t-1} \tag{5.14}$$

$$\widehat{X_t} = 23.21 + (-0.73)X_{t-1} + (0.99)e_{t-1} + \varepsilon_{t-1} \tag{5.15}$$

## 5.2. Simple Linear Regression

A simple linear model for the production data was fitted using Minitab. The first 116 data points were used to fit the line, and the last 12 data points were used to calculate the MAPE of this model. Equation 5.16 shows the equation of the fitted line.

$$Y(t) = 93{,}799 - 123.66(t) + \varepsilon_t \qquad 5.16$$

Figure 5.6 shows the normal probability plot, residual vs. fitted value plots and the residual histogram for the model. Note the visible trend in the residual vs. fitted values plot, an indication of non-constant variance of the residuals. Transformations of the response, as previously suggested, do not improve the performance of the model.

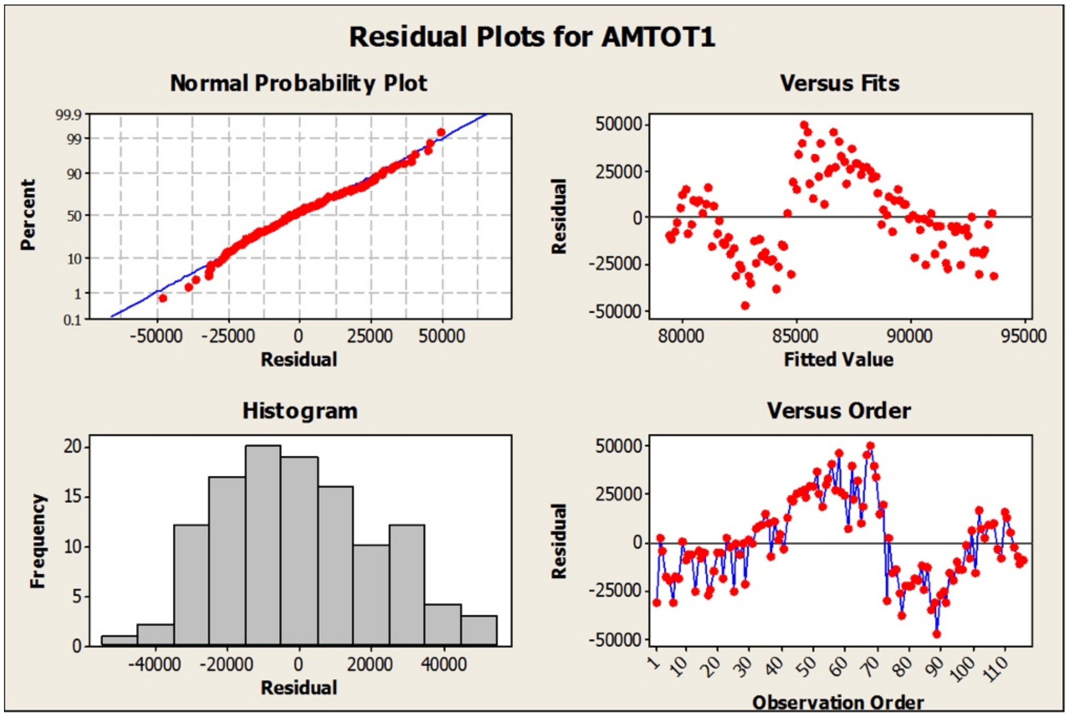


Figure 5.6 – Simple Linear model residual plots

As seen above, the normality assumption is met according to the normal probability plot, however, as mentioned before, the residuals vs. fitted value plot shows a noticeable trend. This is due to the irregular monthly production values of the industry in Iran. Furthermore, previously mentioned remedial measures were carried out to remedy this problem as much as possible, without significantly altering the original data. Figure 5.7 shows the trend analysis plot for the linear model.

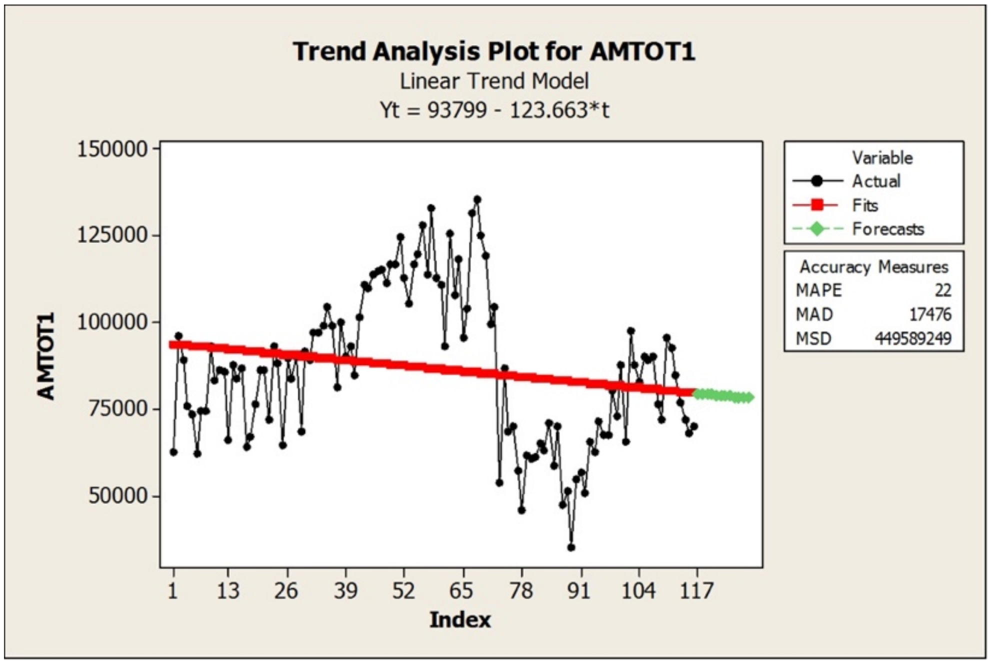


Figure 5.7 – Simple Linear model trend analysis plot

As seen in Figure 5.7, the linear model includes a negative trend that has resulted from the significant decrease in production after time index 68. Furthermore, the linear model provides the following forecasts for the last 12 months, shown along with the actual values and the calculated MAPE for this model in Table 5.5. The calculated $R^2$ for this model is equal to 3.7%.

Table 5.5 – Simple Linear Model Forecasts and MAPE

| **Actual** | **Forecast** | **APE** |
|---|---|---|
| 78,642 | 79,330 | 0.87 |
| 90,378 | 79,206 | 12.36 |
| 86,508 | 79,083 | 8.58 |
| 84,847 | 78,959 | 6.94 |
| 82,481 | 78,835 | 4.42 |
| 93,739 | 78,712 | 16.03 |
| 98,568 | 78,588 | 20.27 |
| 95,314 | 78,464 | 17.68 |
| 108,489 | 78,341 | 27.79 |
| 85,880 | 78,217 | 8.93 |
| 104,052 | 78,093 | 24.95 |
| 121,272 | 77970 | 35.71 |
| | **MAPE =** | **15.38** |

## 5.3. Exponential growth model

The exponential growth model for the production data is fitted using Minitab. Similar to the Box-Jenkins and linear models, the first 116 data points are utilized to fit the model, and the last 12 data points are set aside to calculate the MAPE of this model. Equation 5.17 shows the equation of the fitted line based on the general model $Y(t) = a * b^t + \varepsilon_t$. Note how the value $b = 0.998$ is close to 1 – an indication of insignificance in this case.

$$Y(t) = 92626 * 0.998^t + \varepsilon_t \qquad 5.17$$

Figure 5.8 shows the normal probability plot, residuals vs. fitted value plots and the residual histogram. Note the obvious trend in the residual vs. fitted values plot, an indication of non-constant variance of residuals. Although transforming the response, as seen before, improves the performance of the model, the improvement was not sufficient as to compete with the ANN or ANFIS models.

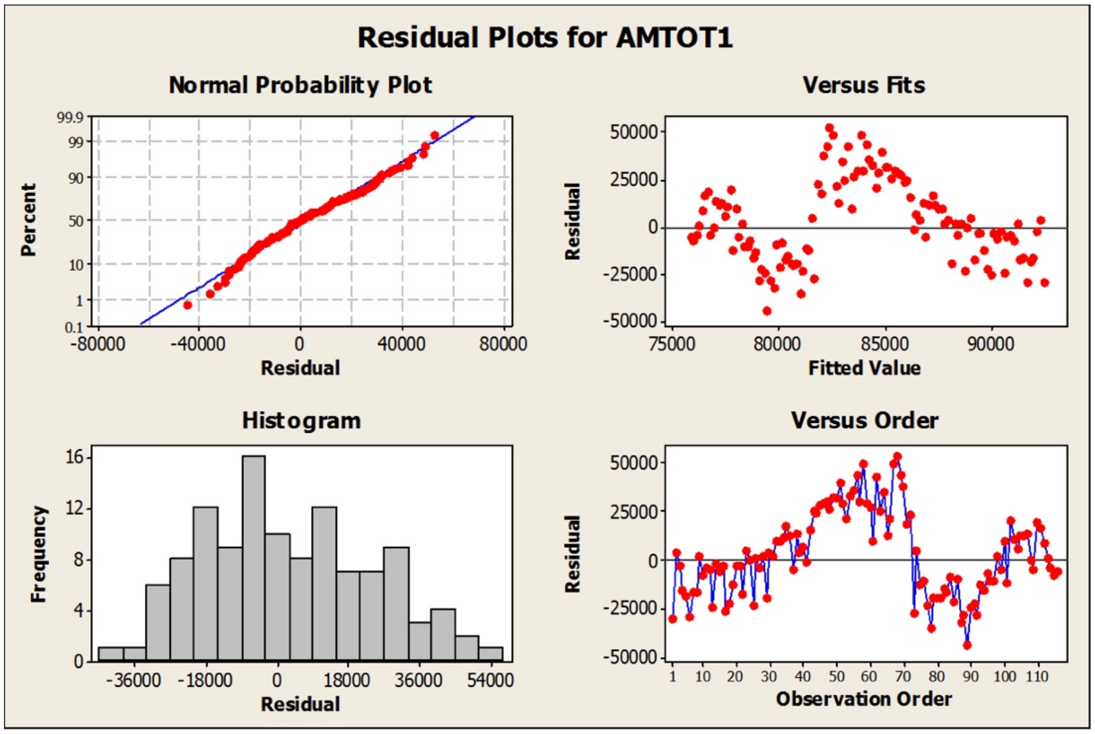


Figure 5.8 – Exponential growth model residual plots

As shown before, the transformation of the response does not improve the performance of the model. Figure 5.9 shows the trend analysis plot for the exponential growth model. As seen in the figure, the exponential growth model also includes a negative trend has resulted from the significant decrease in production.

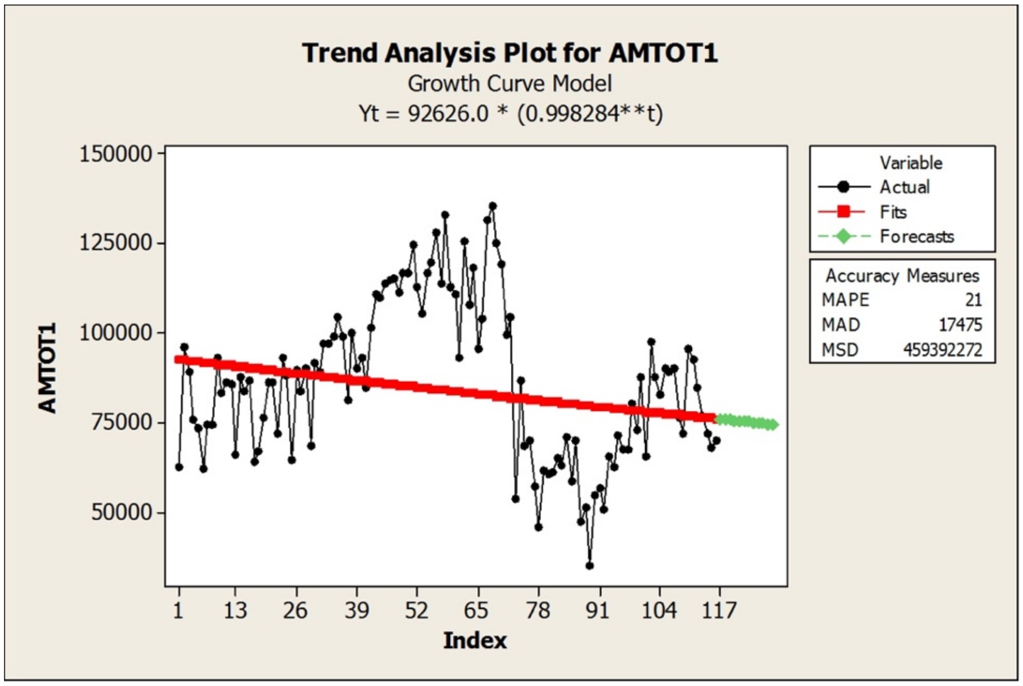


Figure 5.9 – Exponential growth trend analysis plots

Furthermore, the exponential growth model provides the following forecasts for the last 12 months, provided alongside the actual values and the calculated MAPE for this model in Table 5.6.

Table 5.6 – Exponential Growth Model Forecasts and MAPE

| **Actual** | **Forecast** | **APE** |
|---|---|---|
| 78,642 | 75,764 | 3.66 |
| 90,378 | 75,634 | 16.31 |
| 86,508 | 75,504 | 12.72 |
| 84,847 | 75,374 | 11.16 |
| 82,481 | 75,245 | 08.77 |
| 93,739 | 75,516 | 19.44 |
| 98,568 | 74,987 | 23.92 |
| 95,314 | 74,858 | 21.46 |
| 108,489 | 74,730 | 31.12 |
| 85,880 | 74,602 | 13.13 |
| 104,052 | 74,474 | 28.43 |
| 121,272 | 74,345 | 38.70 |
| | **MAPE =** | **19.07** |

## 5.4. Quadratic model

The quadratic model for the production data is fitted using Minitab. Similar to the Box-Jenkins, linear, and exponential growth model, the first 116 data points are utilized to fit the model, and the last 12 data points are set aside to calculate the MAPE of this model. The general equation of the quadratic model can be expressed as $Y(t) = ax^2 + bx + c + \varepsilon_t$. Equation 5.18 shows the equation of the fitted line.

$$Y(t) = -9.05(t^2) + 935(t) + 72973 + \varepsilon_t. \qquad 5.18$$

Figure 5.10 shows the normal probability plot, residuals vs. fitted value plots and the residual histogram. Note the visible trend in the residual vs. fitted values plot, an indication of non-constant variance of residuals.

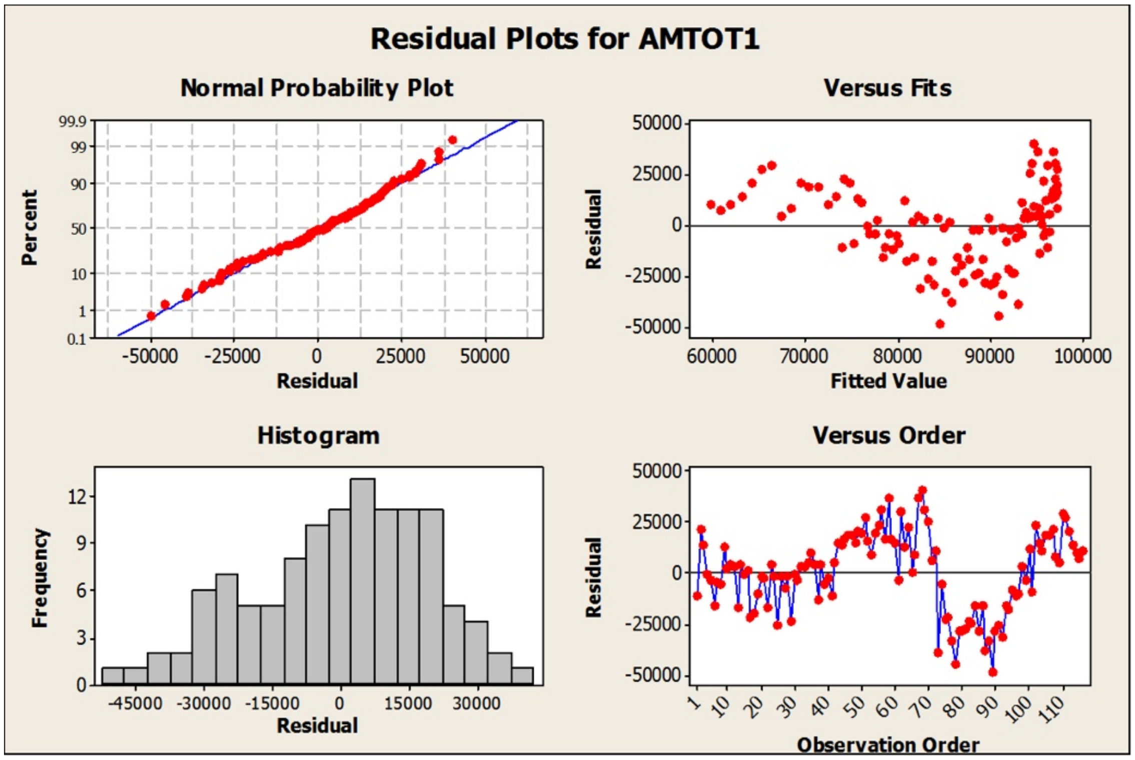


Figure 5.10 – Quadratic Model residual plots

Figure 5.11 shows the trend analysis plot for the quadratic model. As seen in the figure, this model is also forecasting a negative trend that has resulted from the significant decrease in production similar to the linear and exponential growth models.

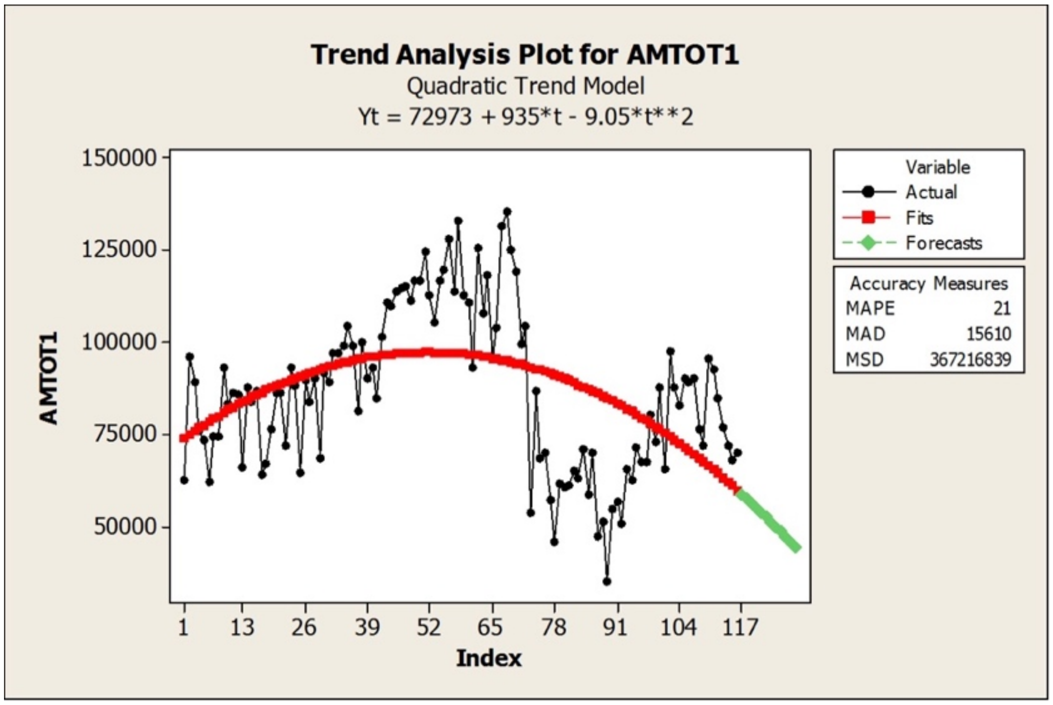


Figure 5.11 – Quadratic Model trend analysis plot

As seen above, the forecasted trend for the quadratic model is very inaccurate. Furthermore, the following forecasts for the last 12 months, are shown along with the actual values and the calculated MAPE for this model in Table 5.7, below. The calculated $R^2$ for this model is equal to 21.3%.

Table 5.7 – Quadratic model Forecasts and MAPE

| **Actual** | **Forecast** | **APE** |
|---|---|---|
| 78,642 | 58,504 | 25.61 |
| 90,378 | 57,312 | 36.59 |
| 86,508 | 56,103 | 35.15 |
| 84,847 | 54,875 | 35.33 |
| 82,481 | 53,629 | 34.98 |
| 93,739 | 52,365 | 44.14 |
| 98,568 | 51,082 | 48.18 |
| 95,314 | 49,782 | 47.77 |
| 108,489 | 48,464 | 55.33 |
| 85,880 | 47,127 | 45.13 |
| 104,052 | 45,773 | 56.01 |
| 121,272 | 44,340 | 63.44 |
| | **MAPE =** | **43.97** |

### 5.5. Artificial Neural Network (ANN) model

As discussed earlier, the following steps are necessary to construct an effective neural network:

- Determining the purpose of the network.
- Gathering data based on the specified network purpose.
- Selecting the appropriate neural network based on the prior specified purpose and data.
- Selecting the most suitable learning algorithm based on the network type and data.
- Selecting the appropriate transfer function according to the network type and learning algorithm.
- Adjusting network parameters such as the number of layers and nodes, or test numbers and training epochs.

Based on the research objective, the feed-forward multi-layer perceptron network with back-propagation and the Levenberg-Marquardt learning algorithm is selected. Moreover, the Tan-Sig transfer function is utilized for the hidden layers, while the linear transfer function is used in the output layer. Figure 5.12 provides a graphical illustration of the constructed network.

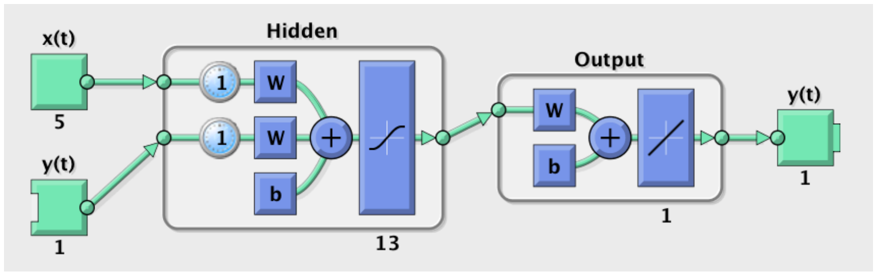


Figure 5.12 – Neural network illustration

Most researchers employ only one hidden layer for forecasting purposes (Nack et al., 1997). Thus, one hidden layer is utilized in this research as well. Furthermore, as one node is used for output layers empirically (Zhang et al., 1998), only one node is utilized in the constructed neural network model in this research. To obtain the best performing neural network (i.e. to optimize the number of hidden layer nodes and differencing period), a MATLAB script was written and utilized, which identifies the best performing neural network according to the set performance metric (MAPE & MSE) based on:

- Differencing: i=1,2, or 3 to achieve stationarity,
- Number of nodes in the hidden layer: j=10:15, and
- Testing time steps: k=12.

The MATLAB code is included in the Appendix. The best performing neural network is identified as a multi-layer perceptron network consisting of two input nodes, one on the nine prior selected predictors and the other to the target response. Moreover, this network has 13 nodes with one hidden layer and one node in its output layer. As seen in Figure 5.13, the best validation performance of 118,185,035.38 is obtained at epoch seven. This indicates that the network reached a minimum value of MSE after seven iterations. Note that the training continued for more than 5 iterations before stopping. Moreover, there are no indications of over-fitting in the plot as the validation and test trends are similar.

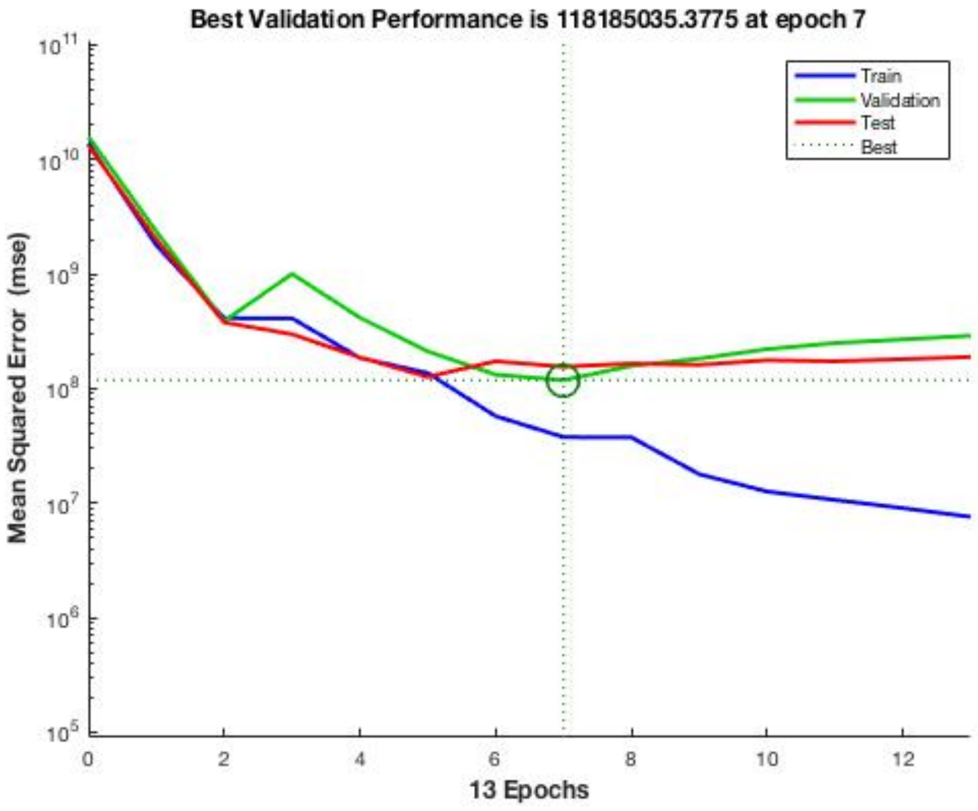


Figure 5.13 – Validation Performance

Next, to validate the network, regression plots are utilized, which show the relationship between network outputs and the target values. The network is trained based on 75% of the data, tested against 15%, and validated against the remaining 15% prior to the model MAPE calculation. Figure 5.14 shows the training, validation, and test regression plots. Note the satisfactory values of 0.96, 0.91, and 0.84 for the training, validation and test R-values respectively, alongside a total value of 0.921 for the model. The perfect result (i.e. perfect output value) is shown with the dashed line in each plot, and the solid line shows the best fit linear regression between target and output values. All the calculated R values suggest satisfactory fits, respectively.

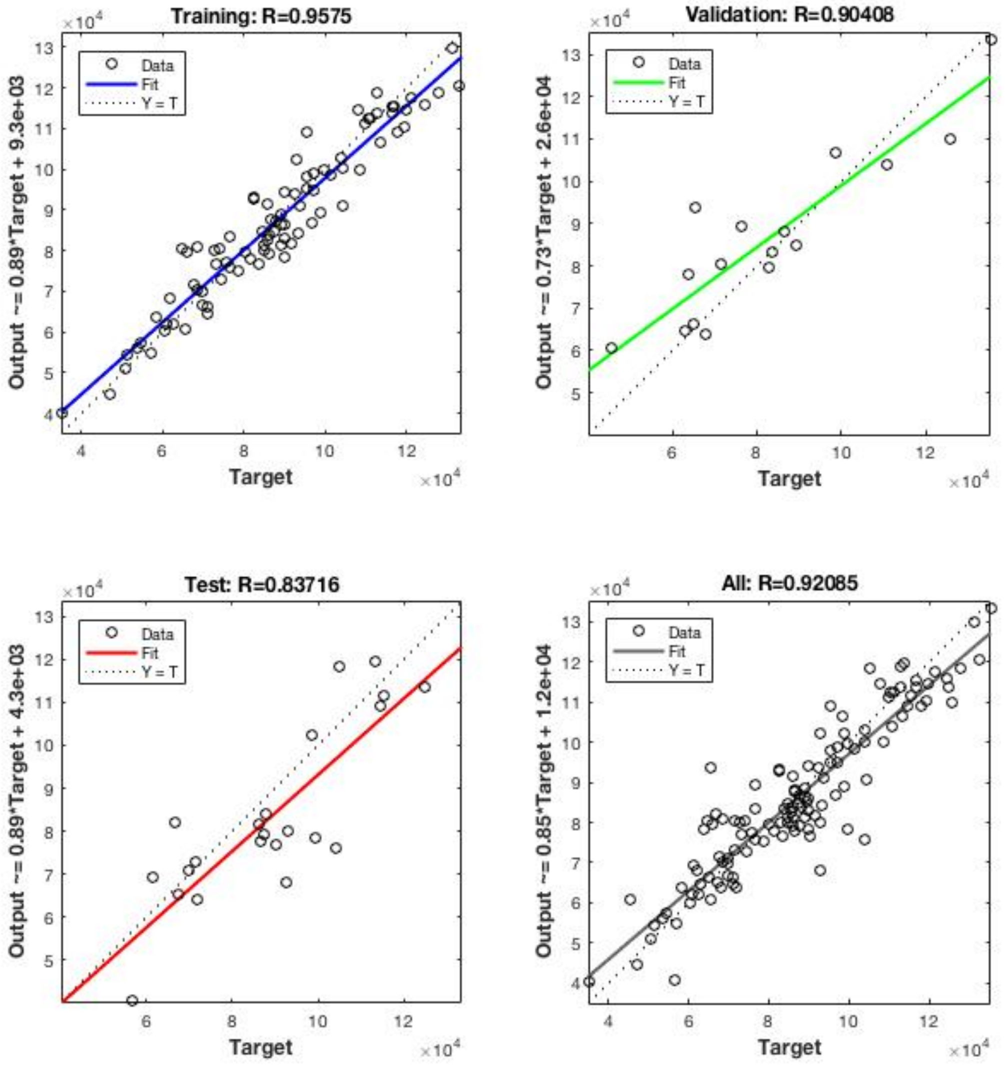


Figure 5.14 – Regression plots

Figures 5.15-16 show the error Autocorrelation plot and the error histogram for the neural network model. As seen below, the error Autocorrelations are within the confidence limit for almost all lags.

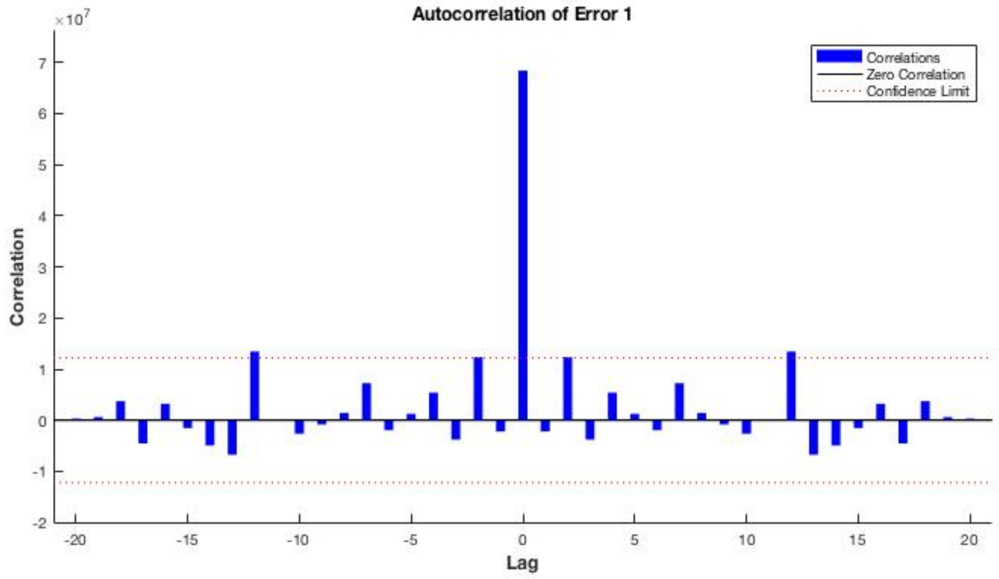


Figure 5.15 – Error Autocorrelation

Figure 5.16 shows that the neural network model errors are normally distributed.

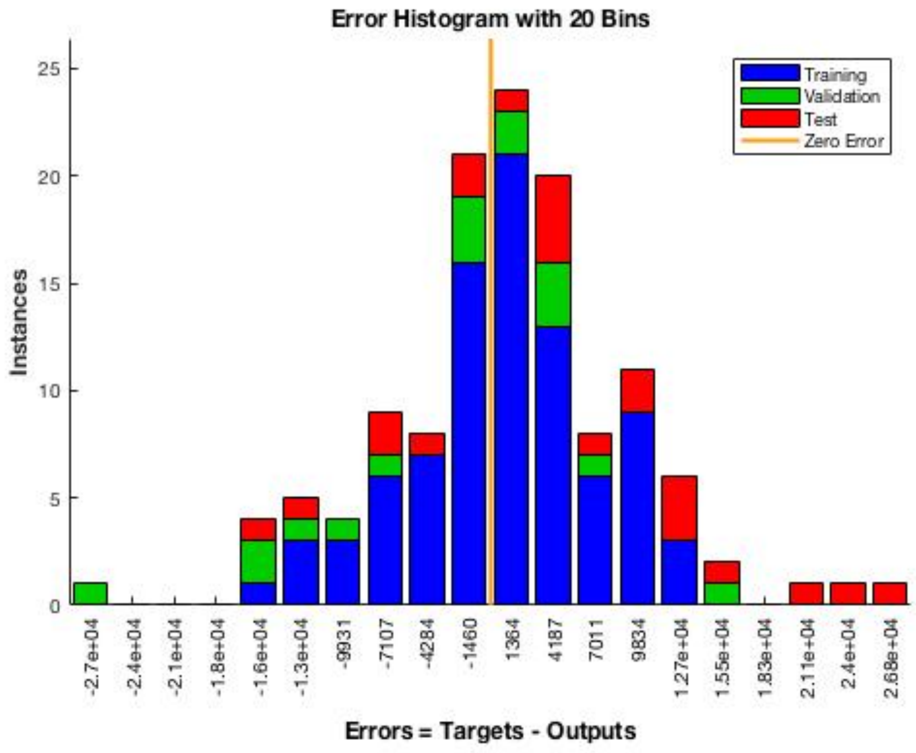


Figure 5.16 – Error Histogram

During model development, the last 12 monthly values were withheld from the network to test its performance. Based on the neural network's forecasted values for the last 12 months, the MAPE of this neural network is calculated as 5.85, suggesting significant superiority over the previously developed models. Figure 5.17 shows the network's performance on the training data and shows actual data alongside the predicted values.

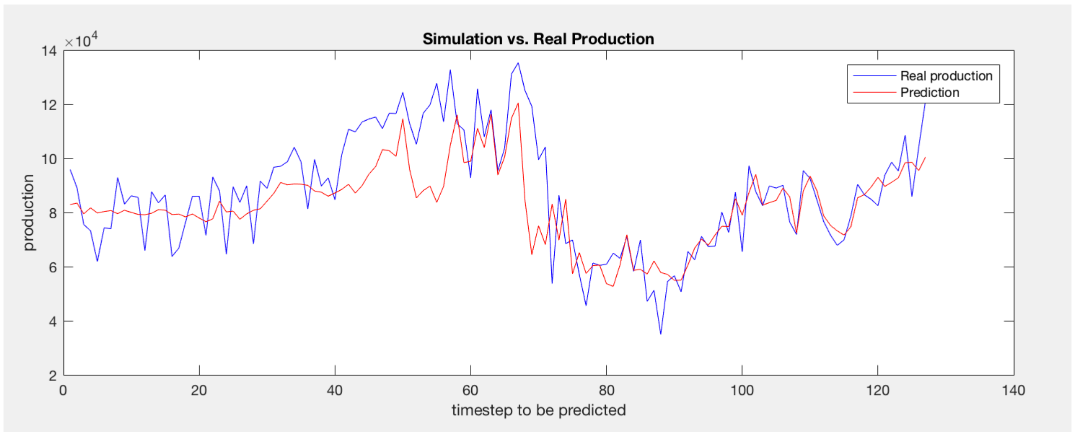
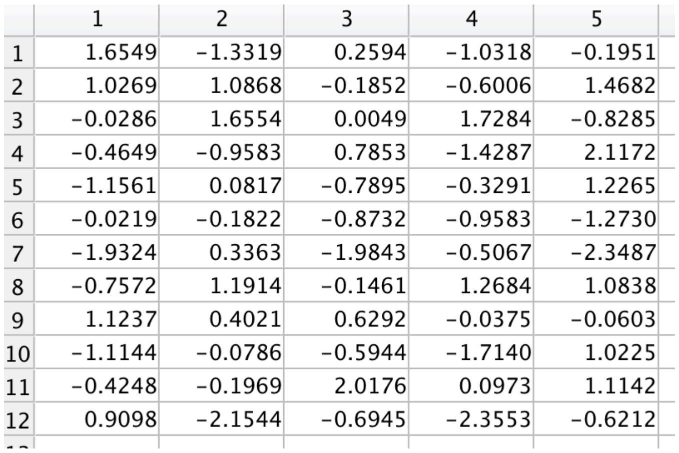


Figure 5.17 – Neural network performance

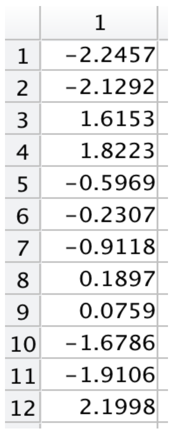
Moreover, the exact network weights, biases and analytic solution for the selected network is provided from the software solution as:

- Input layer weight matrix (IW):



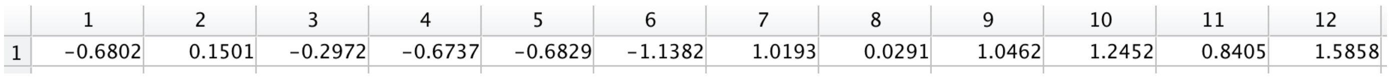
| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 1.6549 | -1.3319 | 0.2594 | -1.0318 | -0.1951 |
| 2 | 1.0269 | 1.0868 | -0.1852 | -0.6006 | 1.4682 |
| 3 | -0.0286 | 1.6554 | 0.0049 | 1.7284 | -0.8285 |
| 4 | -0.4649 | -0.9583 | 0.7853 | -1.4287 | 2.1172 |
| 5 | -1.1561 | 0.0817 | -0.7895 | -0.3291 | 1.2265 |
| 6 | -0.0219 | -0.1822 | -0.8732 | -0.9583 | -1.2730 |
| 7 | -1.9324 | 0.3363 | -1.9843 | -0.5067 | -2.3487 |
| 8 | -0.7572 | 1.1914 | -0.1461 | 1.2684 | 1.0838 |
| 9 | 1.1237 | 0.4021 | 0.6292 | -0.0375 | -0.0603 |
| 10 | -1.1144 | -0.0786 | -0.5944 | -1.7140 | 1.0225 |
| 11 | -0.4248 | -0.1969 | 2.0176 | 0.0973 | 1.1142 |
| 12 | 0.9098 | -2.1544 | -0.6945 | -2.3553 | -0.6212 |

- Input bias weight ($B_1$):



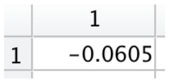
| | 1 |
|---|---|
| 1 | −2.2457 |
| 2 | −2.1292 |
| 3 | 1.6153 |
| 4 | 1.8223 |
| 5 | −0.5969 |
| 6 | −0.2307 |
| 7 | −0.9118 |
| 8 | 0.1897 |
| 9 | 0.0759 |
| 10 | −1.6786 |
| 11 | −1.9106 |
| 12 | 2.1998 |

- Output layer weight matrix (LW):



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | −0.6802 | 0.1501 | −0.2972 | −0.6737 | −0.6829 | −1.1382 | 1.0193 | 0.0291 | 1.0462 | 1.2452 | 0.8405 | 1.5858 |

- Output bias weight ($B_2$):



| | 1 |
|---|---|
| 1 | −0.0605 |

- Analytical solution:

$$Y(t) = B_2 + LW * \tanh\,(B_1 + IW * X(t))$$

where Y(t) is the response and X(t) is the input matrix

Finally, note that the MATLAB script used, generally takes significant run time to identify and construct the best performing model (code available in Appendix). Table 5.8 provides the ANN forecasts and calculated MAPE.

Table 5.8 – ANN model Forecasts and MAPE

| Actual | Forecast | APE |
|---|---|---|
| 78,642 | 76,238 | 3.06 |
| 90,378 | 87,921 | 2.72 |
| 86,508 | 82,314 | 4.85 |
| 84,847 | 71,698 | 15.50 |
| 82,481 | 77,342 | 6.23 |
| 93,739 | 89,634 | 4.38 |
| 98,568 | 95,341 | 3.27 |
| 95,314 | 106,487 | 11.72 |
| 108,489 | 114,540 | 5.58 |
| 85,880 | 85,796 | 0.10 |
| 104,052 | 93,872 | 9.78 |
| 121,272 | 117,649 | 2.98 |
| | **MAPE =** | **5.84** |

## 5.6. Adaptive Neuro-Fuzzy Inference System (ANFIS) model

Based on the previously discussed material in chapter 4, a series of different ANFIS models are developed in this research. The main features of ANFIS models are summarized in Figure 5.18 below:

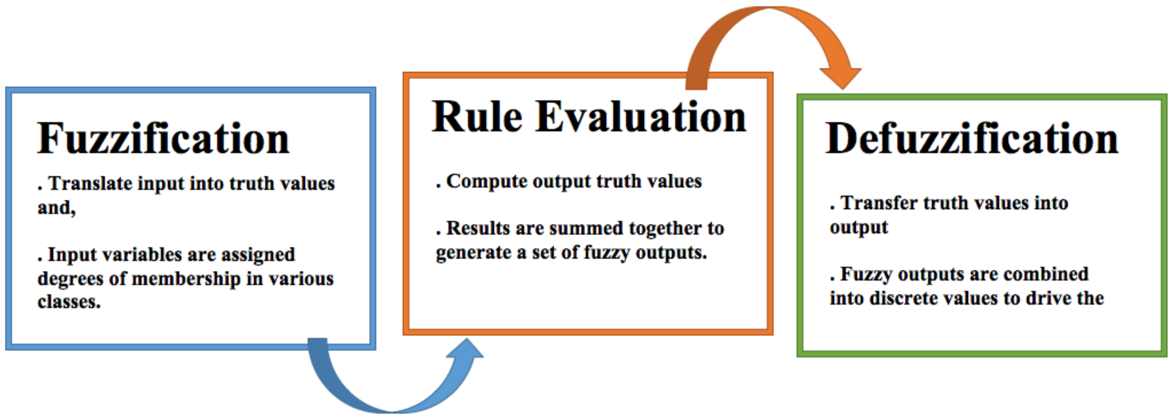

Fuzzification
. Translate input into truth values and,
. Input variables are assigned degrees of membership in various classes.
Rule Evaluation
. Compute output truth values
. Results are summed together to generate a set of fuzzy outputs.
Defuzzification
. Transfer truth values into output
. Fuzzy outputs are combined into discrete values to drive the

Figure 5.18 – ANFIS highlights

To obtain the best performing model based on the research data, two types of Fuzzy Interface Systems (FIS) are utilized based on grid partitioning and sub-clustering. Note that each of the previously mentioned FIS's is constructed based on their requirements as follows:

- Sub-clustering: The parameters associated with sub-clustering include the range of influence, the squash factor, the accept and reject ratios.
- Grid partitioning: The number of input MF's, the MF type along with the output MF type.

Furthermore, according to the above-mentioned general types of FIS, numerous models of each methodology were developed and tested to find the best performing model. Note that the input variables of the constructed models are the five previously identified variables with the highest and most significant correlation values with the response (Gold, Steel, Rubber, Stock Index, and Iron ore monthly prices/index). The training and testing data proportions are similar to those of the ANN model. The relevant formulas have been presented in chapter 4. The two identified best performing ANFIS models are as follows:

- **Sub-clustering model:**

  As mentioned earlier, the parameters to be estimated in this model are:

  1. The range of influence,
  2. The squash factor, and
  3. The accept and reject ratios.

  Furthermore, the utilized MATLAB code constructs and tests different models based on the above-mentioned parameters starting with an initial analysis of each of these parameters. The initial analysis concluded that the optimum range for the radius is between the values of 0.5 and 0.6. Also, the optimum value for the epoch and squash factor were determined as 1000-2000 and 1.25 respectively. Figure 5.19 shows a schematic of this ANFIS model.

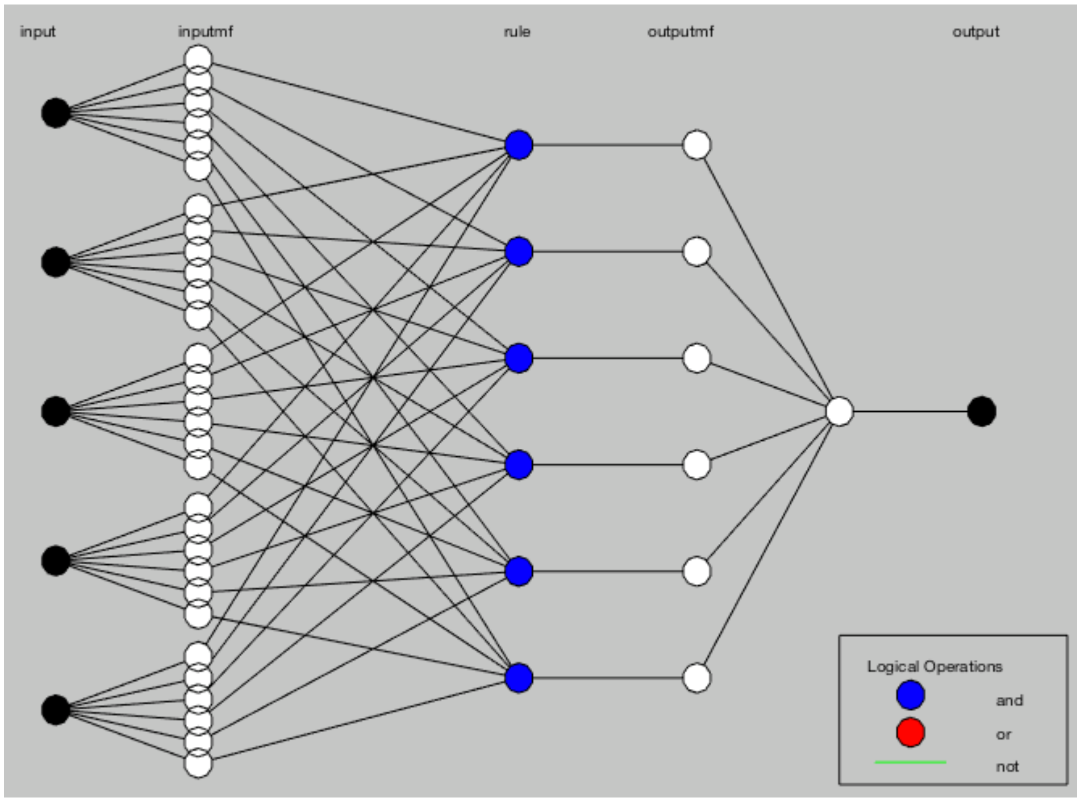


Figure 5.19 – ANFIS structure illustration

Table 5.9, provides the ANFIS model forecasts and the calculated MAPE=21.91 for the model. The MATLAB code written to obtain the result is provided in the Appendix.

Table 5.9 – ANFIS, sub-clustering Model Forecasts and MAPE

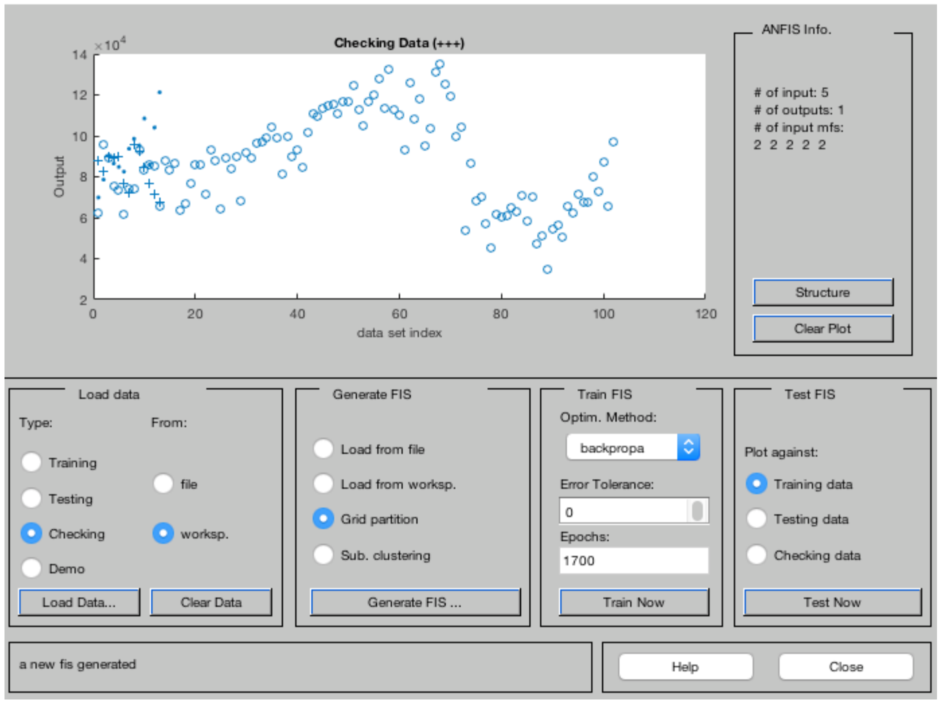
| **Actual** | **Forecast** | **APE** |
|---|---|---|
| 78,642 | 68,234 | 13.24 |
| 90,378 | 74,215 | 17.88 |
| 86,508 | 76,984 | 11.01 |
| 84,847 | 71,689 | 15.51 |
| 82,481 | 68,412 | 17.06 |
| 93,739 | 76,891 | 17.98 |
| 98,568 | 84,213 | 14.56 |
| 95,314 | 61,324 | 35.66 |
| 108,489 | 80,214 | 26.06 |
| 85,880 | 72,154 | 15.98 |
| 104,052 | 64,231 | 38.27 |
| 121,272 | 73,152 | 39.68 |
|  | **MAPE =** | **21.91** |

- **Grid partitioning model:**

  For the model constructed using grid partitioning, the parameters of interest are as follows:

  1. Input MF type and number, and
  2. Output MF type.

  MATLAB provides many different MFs, as described in chapter 4. The 'trapmf' function is identified as the optimum MF based on the research data. This model has five inputs and one output using the weighted average detonated as 'wtaver' in MATLAB as the defuzzification method. The general interface of the 'anfisedit' toolbox in MATLAB is demonstrated in Figure 5.20, below.



ANFIS Info.

# of input: 5
# of outputs: 1
# of input mfs:
2 2 2 2 2

Load data

Type: Training, Testing, [✓ Checking], Demo

From: file, [✓ worksp.]

Generate FIS

Load from file, Load from worksp., [✓ Grid partition], Sub. clustering

Train FIS

Optim. Method: backpropa

Error Tolerance: 0

Epochs: 1700

Test FIS

Plot against: [✓ Training data], Testing data, Checking data

a new fis generated

Figure 5.20 – ANFIS toolbox (MATLAB)

A schematic of the model is further illustrated in Figure 5.21.

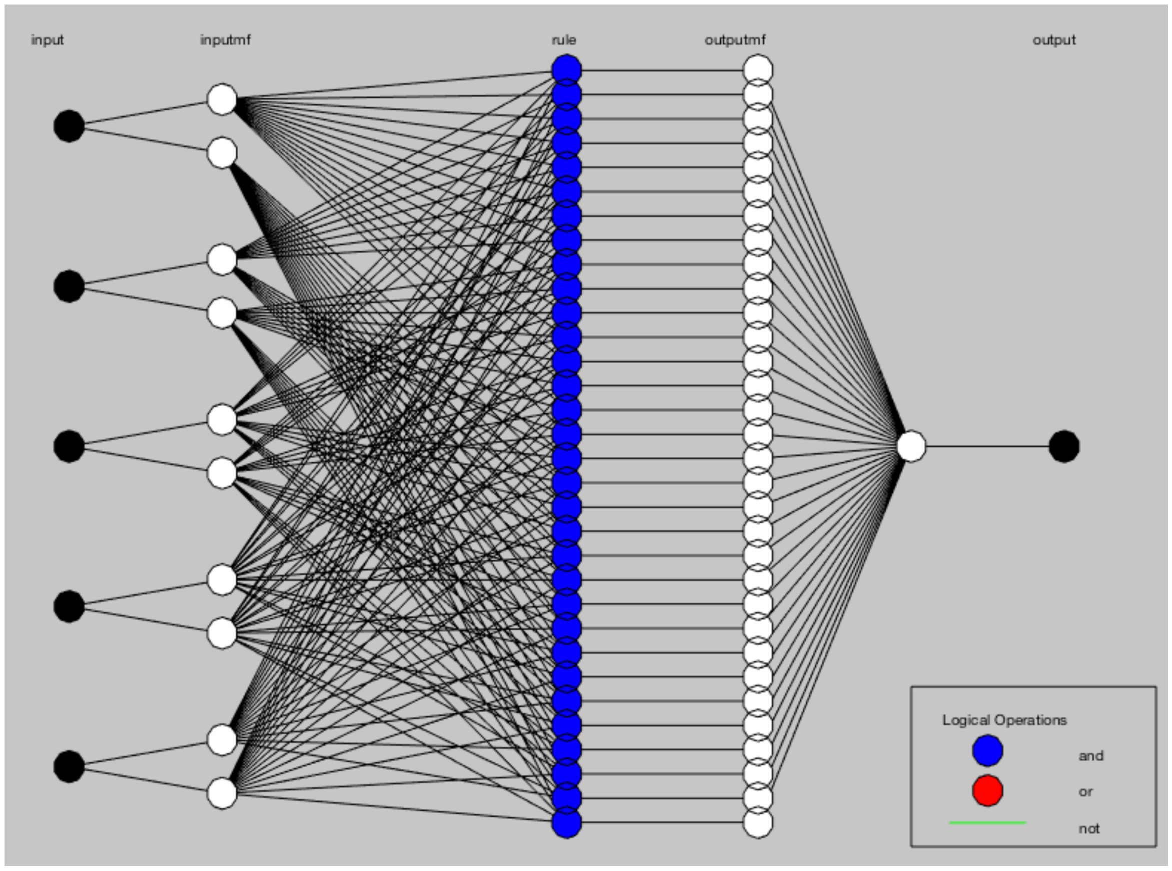


Figure 5.21 – ANFIS structure illustration

Moreover, as seen above, the model has 32 rules. Forecasts generated by this model are shown in Table 5.10 along with the calculated MAPE (MAPE=13.27).

Table 5.10 – ANFIS, grid partitioning model Forecasts, and MAPE

| Actual | Forecast | APE |
|---|---|---|
| 78,642 | 82,505 | 4.91 |
| 90,378 | 89,806 | 0.63 |
| 86,508 | 89,024 | 2.91 |
| 84,847 | 90,035 | 6.11 |
| 82,481 | 76,449 | 7.31 |
| 93,739 | 71,844 | 23.36 |
| 98,568 | 95,466 | 3.15 |
| 95,314 | 92,512 | 2.94 |
| 108,489 | 84,667 | 21.96 |
| 85,880 | 76,643 | 0.75 |
| 104,052 | 71,642 | 31.15 |
| 121,272 | 67,853 | 44.05 |
|  | **MAPE=** | **13.27** |

Based on the results shown above for the two identified models, the FIS generated based on grid partitioning is providing forecasts with smaller MAPE values as compared to those of the sub-clustering method. Thus, this model is selected as the best performing ANFIS model and is further compared against the previously constructed models i.e. the simple linear and ANN models.

## 5.7. Comparisons

This section summarizes the results obtained from the proposed models and compares them based on the specified performance measure, the MAPE. Models investigated in this research include the Simple Linear, Exponential growth, Quadratic, ARMA (1,3,1), Robust ARMA, Multi-layer perceptron neural network, and the ANFIS models. Table 5.11 shows the MAPEs associated with the models and ranks them according to the MAPE value.

Table 5.11 – MAPE for developed models

| **Model** | **MAPE** | **Rank** |
|---|---|---|
| Linear | 15.38 | 4 |
| Exponential growth | 19.07 | 6 |
| Quadratic | 43.97 | 8 |
| ARIMA (1,3,1) | 19.00 | 5 |
| **ARIMA (Robust)** | **9.68** | **2** |
| **ANN** (multi-layer perceptron) | **5.85** | **1** |
| ANFIS.subclustering | 21.91 | 7 |
| **ANFIS.gridpartitioning** | **13.27** | **3** |

As can be seen in Table 5.11, the ANN provided the lowest MAPE value among the investigated models. The Robust ARIMA model is the second best performing model based on the calculated MAPE. Note the significant difference between the two ARIMA methodologies, which is due to the difference between generating 1 time step ahead forecast vs. 12 time step ahead forecasts. The ANFIS model and Linear Regression model are ranked 3 and 4 respectively. Furthermore, according to the previously stated research objective, it is clear that the ANN model provides the most accurate forecast among the investigated models. Figure 5.22 illustrates the time-series alongside the fitted models and their forecasts.

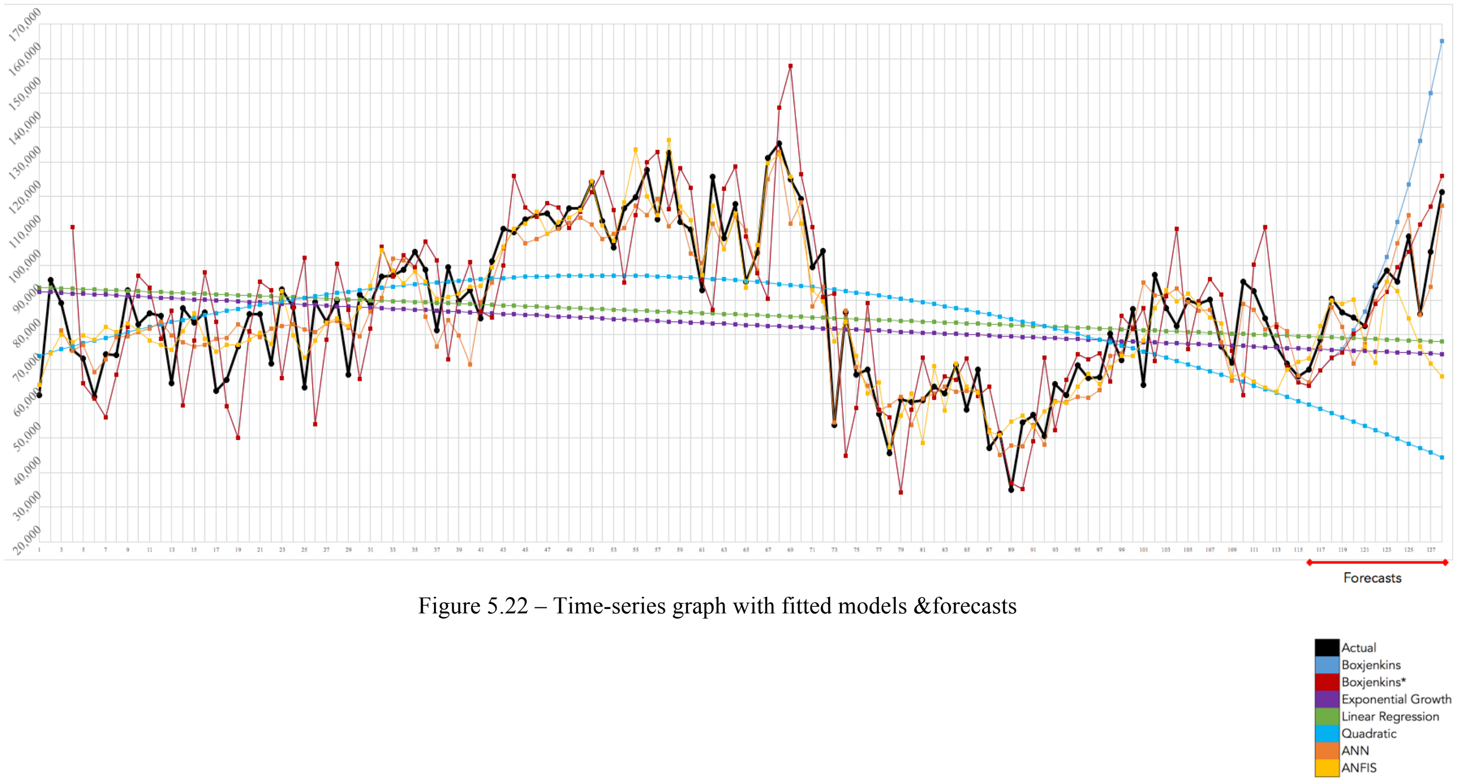


Figure 5.22 – Time-series graph with fitted models &forecasts

It can be seen from Figure 5.22 that the ANN model outperformed other utilized models. Suggestions towards improving the feedforward multi-layer perceptron neural network are discussed in Chapter 6.

# CHAPTER 6 | Conclusion & future work

Automotive industries vastly benefit from the ability to accurately forecast the demand for their products. The increasing demand for transportation and automobiles in Iran in the past half century and the fact that there are no available forecasting models for the demand has made it necessary to provide an accurate forecasting model that can be used reliably to plan for sufficient production to meet the customers' demand. It is clear that domestic automobile manufacturers will benefit significantly from having accurate forecasts for the demand of their products.

Based on the literature survey, the ANN and ANFIS methodologies were expected to outperform preceding methodologies. The obtained results confirm the initial expectation as the ANN model provides the most accurate forecasts compared to the other developed models. Aside from the monthly domestic production values of cars in Iran, a number of other variables such as monthly gold, steel, rubber, and iron ore prices were used as inputs for the ANN and ANFIS models. Note that the utilized time series presented significant variation throughout the 128 monthly data points. Both AI models were less affected by this compared to the other investigated models. Note that the remedial procedures attempted to address this issue. However, further tampering with the data would result in significant deviation from the actual data.

After developing and testing several forecasting models, namely the simple linear and the Box-Jenkins models, the feedforward multi-layer perceptron artificial neural network model with back-propagation and the Levenberg-Marquardt learning algorithm was identified as the best performing forecasting model with regards to MAPE. The hidden layers of this network utilize the Tan-Sig transfer function, while the output layer uses the linear transfer function.

Figure 6.1 shows the forecasts from all utilized models. Based on this graph, it is clear that the ANN methodology is superior compared to other investigated methodologies.

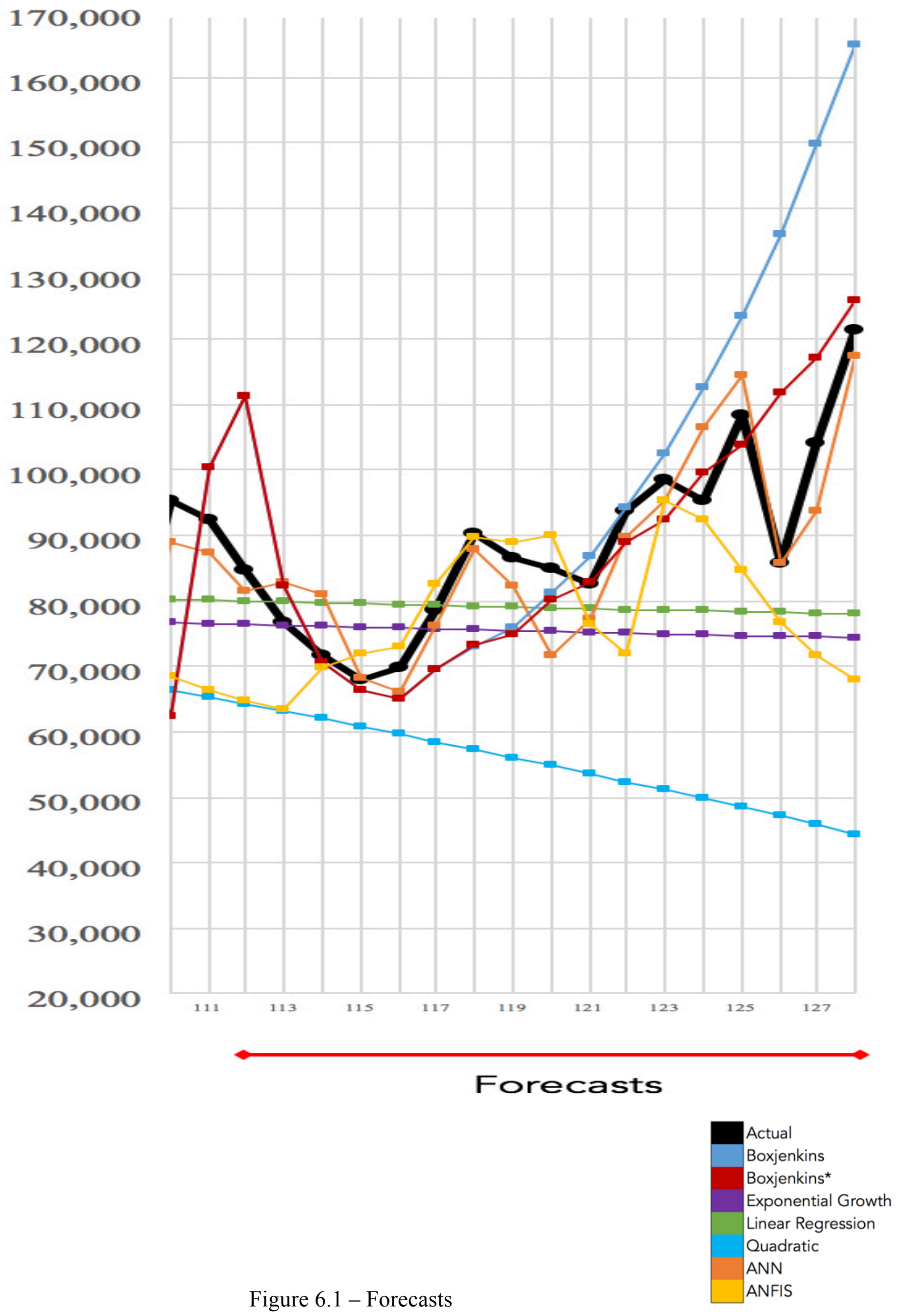


Figure 6.1 – Forecasts

Moreover, the following summarizes recommendations for future work:

- Expanding the data set: This research was based on 128 data points of the nine input/predictor variables and the response. Adding more data points will only help the developed ANN model generate more accurate forecasts. Also, other input/predictor variables should be investigated for possible inclusion in the model.

- Refining/tuning input variables: Due to the scarcity of reliable data in Iran, a limited number of predictors with relations to the response were identified and utilized. However, as the country is progressing, more data will be available, which could be used for the ANN forecasting model. The proposed ANN model – the feedforward multi-layer perceptron network with two input nodes (one for the nine predictors and another for the response), with 13 nodes in one hidden layer can be further tuned to improve the forecasting accuracy.

- Investigating other emerging methodologies: The Water Wave Optimization metaheuristic, which presents aspects of water waves such as propagation, refraction, and breaking can be used to derive effective models to search for feasible solutions in high-dimensional solution spaces. This methodology can be adapted to time series forecasting problems towards obtaining more accurate forecasts.

# References

## Referenced Papers

1. AhmariNejad A., Rajabi H. and Sadeh J., 2005, The Importance of Load Forecasting for Electricity Price Forecasting with dividing of electricity price to separating different components of electrical energy in a competitive environment, Presented at 20th international power system conference.
2. Akhavan Niaki, S.T., Hoseinzade, S., 2013, Forecasting S&P 500 index using artificial neural networks and design of experiments, Journal of Industrial Engineering International, (9)1.
3. Aris Z. and Mohamad D., 2008, Application of Artificial Neural Networks Using Hijri.
4. Azar, A., Afsar, A., 2006, A Modeling of Stock Price Forecasting Using Neuro-Fuzzy System, Iranian Journal of Trade Studies, 33(52).
5. Barth, J.A., 1898, Vorlesungen uber Gastheorie, 23(89).
6. Biggs, J.R., Campion, W.M., 1982, The effect and cost of forecast error bias for multiple stage production-inventory systems, Decision Sciences, 13(4).
7. Catalão J. P. S., Mariano S. J. P. S., Mendes V. M. F., Ferreira, L. A. F. M., 2007, An Artificial Neural Network Approach for Short-Term Electricity Prices Forecasting Engineering Intelligent Systems for Electrical Engineering & Communications, 15(1).
8. Crane, D. B., Eratty, J. R., 1967, A two stage forecasting model: exponential smoothing and multiple regression, Management Science, 13(8).
9. Dickey, D. A., Fuller, W. A., 1979, Distribution and the Estimators for Autoregressive Time Series with a Unit Root, Journal of the American Statistical Association, 74(366).
10. Esmaili, A., Oloumi, M., 2008, Using a Neural Network Approach to Predict the Energy Cost of the Electricity Market of Iran, Presented at 23th International Power System Conference.
11. FarjamNia, I., Naseri, M., Ahmadi, M., 2007, Oil Price Forecasting Using ARIMA Model and Artificial Neural Networks, Iranian Journal of Economic Research, 32(183).
12. Ljung, G.M., Box, P., 1978, On a Measure of Lack of Fit in Time Series Models, Biometrika, 65(2): 297-303
13. Haidar, I., Kulkarni, S., Pan, H., 2008, Forecasting Model for Crude Oil Prices Based on Artificial Neural Networks, Journal of Institute of Electrical and Electronics Engineers, 978(42).
14. Hippert, H. S., Perreira, C. E., Souza, R. C., 2001, Neural Networks for Short-Term Load Forecasting: A Review and Evaluation, IEEE Transactions on Power Systems, 16(44).
15. Holt, C.C., Modigliani, F., Simon, H.A., 1955, A linear decision rule for production and employment scheduling, Management Science, 2(1).
16. Kamruzzaman, J., Sarker, R. A., 2003, Comparing ANN Based Models with ARIMA for Prediction of Forex Rates, Asia Bulletin, 22(2).
17. Lee, T.S., Adam, E.E.Jr, 1986, Forecasting Error Evaluation in Material Requirements Planning (MRP) Production-Inventory Systems, Management Science, 32 (9).
18. Lu, C.N., Wu, N.T., Vemuri, S., 2014, Neural Network Based Short Term Load Forecasting, International Journal of Computer Trends and Technology, 11(6).

19. Mandal, P., Senjyu, T., Urasaki, N., Funabashi, T., Srivastava, A. K., 2007, A Novel Approach to Forecast Electricity Price for PJM Using Neural Network and Similar Days Method, IEEE Transactions On Power Systems, 22(4).
20. Merh, N., Vinod, P. Saxena, Pardasani K., 2010, A Comparison between Hybrid Approaches of ANN and ARIMA for Indian Stock Trend Forecasting, Business Intelligence Journal, 3(2).
21. Mirbagheri, 2010, Fuzzy- Logic and Neural Network Fuzzy Forecasting of Iran GDP Growth, African Journal of Business Management, 4(6).
22. Mirsoltani, S.M., Akhavan Niaki, S.T., 2013, Forecasting energy price and consumption for Iranian industrial sectors using ANN and ANFIS, Iranian Journal of Economic Studies, 2(1).
23. Nagel, S., Singleton, K.J., 2011, Estimation and Evaluation of Conditional Asset Pricing Models, Journal of Finance, 66(3).
24. Nurmi, A., Martin, P., Rossi, M., 2007, Communication patterns over the project life cycle – evidence from a virtual project exercise, Proceeding, HICSS proceedings of the 40th annual Hawaii International Conference on System Sciences, 232(b).
25. Ritzman, L.P., King, B.E., 1993, The relative significance of forecast errors in multistage manufacturing, Journal of Operations Management, 11(51).
26. Roy, Sh. Sh., 2005, Design of Adaptive Neuro-Fuzzy Inference System for Prediction Surface Roughness in Turning Operation, Journal of Scientific and Industrial Research, 64(653).
27. Sadeghi, H., Zolfaghari, M., ElhamiNejad, M., 2011, Performance Comparison of Neural Networks and ARIMA Models in the Modeling and Forecasting of Short-Term Price of OPEC Basket of Crude Oil, Energy Economics Studies Journal, 28(25).
28. Sarafraz, L., Afsar, A., 2005, The Examination of Effective Factors of Gold Price and Forecasting Using Neuro-Fuzzy System, Quarterly Publication of the Economic Researches, 16.
29. Shing, J., Jang, R., 1993, ANFIS: Adaptive-Network-based Fuzzy Inference System, IEEE Transaction on System, Man and Cybernetics, 23(3).
30. Sinaie, H.A., Mortazavi, S., Teymuri, Y., 2005, Tehran Stock Index Forecasting Using Neural Networks, Iranian Accounting, and Auditing Review Journal, 41(59).
31. Szkuta, B.R., Sanabria, L.A., Dillon, T.S., 1999, Electricity Price Short-Term Forecasting Using Artificial Neural Networks, IEEE Transactions on Power Systems, 14(3).
32. Terwiesch, C., Bohn, R.E., 2001, Learning and process improvement during production ramp-up, International Journal of Production Economics, 70(1).
33. Yaghoubi, M., Toutounchi, M. R. A., Bahrepour, M., 2008, Fuzzy Time Series for Currency Price Trend Forecasting, Presented at 13th Computer Society of Iran Conference, Kish Island, Iran.
34. ZaraNejad, M., Hamid, Sh., 2009, Forecasting of Iran Inflation Rate Using Dynamic Artificial Neural Networks, Quarterly Journal of Economic Review, 1(145).
35. Zhang, G., Patuwo, B. E., Hu, M. Y., 1998, Forecasting with Artificial Neural Networks: the state of the art, International Journal of Forecasting 14(35).

**Referenced Books (Manuals & Technical reports included)**

1. Baumgartner, T., Midttun, A., The Politics of Energy Forecasting: A Comparative Study of Energy Forecasting in Western Europe and North America, Oxford University Press, New York, 1987.
2. Bhattacharyya, S.C., Energy Economics: concept, issues, markets and governance, Springer, UK, 2011.
3. Box, G.E.P., Jenkins, G. M., Time Series Analysis: Forecasting and Control. San Francisco: Holden-Day, 1970.
4. Brockwell, P. J., & R. A. Davis, Introduction to Time Series and Forecasting, Springer, New York, 1996.
5. Brown, R. G., Smoothing, Forecasting, and Prediction of Discrete Time Series, Prentice-Hall, New Jersey,1962.
6. Chen, C. T., Linear System Theory and Design, third ed., Oxford University Press, 1999.
7. Donovan T. M., Short Term Forecasting: an introduction to the Box-Jenkins approach, John Wiley and Sons Publishing Co., New York, 1983.
8. Finlay, J., Dix, A., An Introduction to Artificial Intelligence, UCL Press, London, 1996.
9. Holden, K., Peel, D.A., Thompson, J.L., Economic Forecasting: an introduction, Cambridge University Press, New York, 1990.
10. Kartalopouloss, S. V., Understanding Neural Networks and Fuzzy Logic: Basic concepts and applications, IEEE Press, New York, 1996.
11. Lawrence R., Using Neural Networks to Forecast Stock Market Prices, Department of Computer Science University of Manitoba, 1997.
12. Learning MATLAB, Student Guide, Version 6.0, Math Works INC., 2001.
13. Levenbach, H., Cleary, J. p., Forecasting: Practice and process for demand management, Thomson Publishing, U.S.A, 2006
14. Lunar Transaction as Extracted Variables to Predict Stock Trend Direction, Labuan e-Journal of Muamalat and Society, 2(9).
15. Mankiw, N. G., Principles of Macroeconomics, Second Edition, Harcourt College Pub, U.S.A, 2000.
16. Marquez, L., Hill, T., Connor, M., Remus, W., Neural Network Models for Forecast: A Review, the University of Hawaii and New South Wales University, 1992.
17. Nauck, D., Klawonn, F., and Kruse, R., Foundation of Neuro-Fuzzy Systems, John Wiley & Sons Co., New York, 1997.
18. Neural Network Toolbox for use with MATLAB, User's Guide, Version 3.0, Math Work INC., 1998.
19. NIST/SEMATECH e-Handbook of Statistical Methods, http://www.inl.nist.gov/div898/handbook/, April 2012.
20. Pankratz, A., Forecasting with Univariate Box-Jenkins Models: Concepts and Cases, John Wiley & Sons Co, New York, 1983.
21. Picton, Ph., Neural Networks, Second edition, Palgrave Publisher, New York, 2001.
22. Pourkazemi, M. H., Asadi, M., Crude Oil Price Forecasting Using Artificial Neural Networks with Using of OECD Countries Oil Resources, Shahid Beheshti University, Tehran, Iran, 2009.

23. Sandberg, I.W., Lo, J. T., Fancourt, C. L., Principe, L. C., Katagiri, Sh., Haykin, S., Nonlinear Dynamical System: feed forward neural network perspectives, Wiley- Inter-science publication, New York, 2001.
24. Sayaddi, O., Basic Tutorial of Artificial Neural Networks, Sharif University of Technology, Medical Image Processing and Vital Signals Laboratory, 2008.
25. Tarafdar, M., Kashtiban, A. M., Application of Neural Networks in Power System, World Academy of Science, Engineering and Technology, Dept. Elect. Eng., Tabriz University, June 2005.
26. Vollman, T.E., Berry, W.L., Whybark, D.C., Manufacturing Planning and Control Systems, 3rd edition, Richard D. Irwin Corp: Homewood, IL,1992.
27. Wheelwright, S.C., Makridakis, S., Forecasting Methods for Management, fourth edition, John Wiley & Sons Co., New York, 1985.
28. Wolfe, H. D., Business Forecasting Methods, Holt, Rinehart and Winston INC., USA, 1966.
29. Zalloi, M., Forecasting Momentary Price of Crude Oil and Gas Fossil Energies, in the World Markets through Fuzzy Based Modeling, National Iranian Gas Company, 1999.

# Appendix

## Code

## ANN-script:

```
flag = 0;% a flag to stop the calculation when the criteria on line 17 met.
[mm,nn]=size(X1);% get the size of the input
for i=1:3 % the delay interval that you want to test
   for j=10:15 % the hidden layer interval that you wana test
      for k=24:24% the number of the time stps for the test
         for tR=0.7:0.05:0.8 % the interval for training ratio
            for vR=0.1:0.05:0.15% the interval for validation ratio
               for teR=0.1:0.05:0.2% the interval for testing ratio
   [mape,rmse,pred1,actual1,sim,real,Network]=NNrun(i,j,k,X1,Y1,tR,vR,teR);% run the neural net and get
the data
        if mape<0.06&&rmse<7000 % this is the condition line based on RMSE error. change it as you wish
           flag = 1;% if line 17 met, the flag would be on
           rmse1=rmse;
           pred=pred1;%line 17 met, get the predicted data
           actual=actual1;%line 17 met, get the actual data
           sim1=sim;%line 17 met, get the simulated data
           real1=real;%line 17 met, get the real data
           mape_min=mape;
           Final_net=Network;
           Input_Weight=Final_net.IW;% input weights
           Output_Weight=Final_net.LW;%output weights
           Bias=Final_net.b;%% bias
        end

   close all;% it closes all the windows to avoid lack of memory
      end
      end
      end
      end
   end


   if flag == 1% %line 17 met, plot the results!!
      knownOutputTimesteps = 1:(nn-k);%% the elements in (1: (114-PP))
predictOutputTimesteps = (nn-i-(k-1)):nn-i;%%(105:114)
S=predictOutputTimesteps(1,1):predictOutputTimesteps(1,k);% interval of
%timesteps for graphing purpose
      figure
subplot(2,1,1)
 plot(S,actual1,'b',S,pred1,'r');
 legend('Real production','Prediction')
title('Prediction vs. Real Production')
xlabel ('timestep to be predicted');
ylabel ('production')
subplot(2,1,2)
[tt,dd]=size(actual1);
plot(1:nn-i,real1,'b',1:nn-i,sim1,'r')
```

```
title('Simulation vs. Real Production')
legend('Real production','Prediction')
xlabel ('timestep to be predicted');
ylabel ('production')
weights=getwb(Final_net);
break;
end
end
```

**ANFIS – sub-clustering model code:**

```
Inputs = IN;
Targets = OUT;
nData = size(Inputs,1);
pTrain=0.85;
nTrainData=round(pTrain*nData);
TrainInd=PERM(1:nTrainData);
TrainInputs=Inputs(TrainInd,:);
TrainTargets=Targets(TrainInd,:);
pTest=1-pTrain;
nTestData=nData-nTrainData;
TestInd=PERM(nTrainData+1:end);
TestInputs=Inputs(TestInd,:);
TestTargets=Targets(TestInd,:);
        PARAMS=inputdlg(Prompt,Title,1,DefaultValues);
        pause(0.01);
        nCluster=str2num(PARAMS{1});        %#ok
        Exponent=str2num(PARAMS{2});        %#ok
        MaxIt=str2num(PARAMS{3});           %#ok
        MinImprovment=str2num(PARAMS{4});   %#ok
        DisplayInfo=1;
        FCMOptions=[Exponent MaxIt MinImprovment DisplayInfo];
fis=genfis3(TrainInputs,TrainTargets,'sugeno',nCluster,FCMOptions);
end
MaxEpoch=str2num(PARAMS{1});                %#ok
ErrorGoal=str2num(PARAMS{2});               %#ok
InitialStepSize=str2num(PARAMS{3});         %#ok
StepSizeDecreaseRate=str2num(PARAMS{4});    %#ok
StepSizeIncreaseRate=str2num(PARAMS{5});    %#ok
TrainOptions=[MaxEpoch ...
              ErrorGoal ...
              InitialStepSize ...
              StepSizeDecreaseRate ...
              StepSizeIncreaseRate];
DisplayInfo=true;
DisplayError=true;
DisplayStepSize=true;
DisplayFinalResult=true;
DisplayOptions=[DisplayInfo ...
                DisplayError ...
                DisplayStepSize ...
                DisplayFinalResult];
OptimizationMethod=0;
fis=anfis([TrainInputs TrainTargets],fis,TrainOptions,DisplayOptions,[],OptimizationMethod);
Outputs=evalfis(Inputs,fis);
```

```
TrainOutputs=Outputs(TrainInd,:);
TestOutputs=Outputs(TestInd,:);
TrainErrors=TrainTargets-TrainOutputs;
TrainMSE=mean(TrainErrors.^2);
TrainRMSE=sqrt(TrainMSE);
TrainErrorMean=mean(TrainErrors);
TrainErrorSTD=std(TrainErrors);
TestErrors=TestTargets-TestOutputs;
TestMSE=mean(TestErrors.^2);
TestRMSE=sqrt(TestMSE);
TestErrorMean=mean(TestErrors);
TestErrorSTD=std(TestErrors);
End
flag = 0;
n=0;
for i=0.5:0.02:0.6
  for j=1000:3250
[MAPE,R_Corr,PR_Test,AC_Test,PR_Train,AC_Train,PR,AC]=AnfisRun2(i,j,IN,OUT);%%% "i" is the
Radius; "j" is the epoches
      if MAPE<0.3&&R_Corr>0.4
        flag = 1;
        R_CorrMin=R_Corr;
        mape_min=MAPE;
      end
  close all;% it closes all the windows to avoid lack of memory
  n=n+1;
  if flag == 1
  TrainTargets=AC_Train;
  TrainOutputs=PR_Train;
  TestTargets=AC_Test;
  TestOutputs=PR_Test;
  Targets=AC;
  Outputs=PR;
if ~isempty(which('plotregression'))
  figure;
  plotregression(TrainTargets, TrainOutputs, 'Train Data', ...
          TestTargets, TestOutputs, 'Test Data', ...
          Targets, Outputs, 'All Data');
  set(gcf,'Toolbar','figure');
end
```

**ANFIS – grid partitioning model using ‘anfisedit’ MATLAB toolbox:**

```
[System]
Name='anfistrampf1500backprop'
Type='sugeno'
Version=2.0
NumInputs=5
NumOutputs=1
NumRules=32
AndMethod='prod'
OrMethod='probor'
ImpMethod='prod'
AggMethod='sum'
```

```
DefuzzMethod='wtaver'
[Input1]
Name='input1'
Range=[328.208 1780.648]
NumMFs=2
MF1='in1mf1':'trapmf',[-673.6964 -101.1796 757.000687605796 1330.32649586363]
MF2='in1mf2':'trapmf',[755.308389970476 1329.51787044754 2188.8876 2761.4044]
[Input2]
Name='input2'
Range=[105.54 256.24]
NumMFs=2
MF1='in2mf1':'trapmf',[0.0500000000000256 60.33 150.57028695941 212.801911527114]
MF2='in2mf2':'trapmf',[150.581369780447 210.848268606412 301.45 361.73]
[Input3]
Name='input3'
Range=[56.7 280.79]
NumMFs=2
MF1='in3mf1':'trapmf',[-100.163 -10.527 123.969442458127 213.096987192184]
MF2='in3mf2':'trapmf',[123.390423212143 213.604988350438 348.017 437.653]
[Input4]
Name='input4'
Range=[8131.3529 87389.368]
NumMFs=2
MF1='in4mf1':'trapmf',[-47349.25767 -15646.05163 31908.7569485313 63611.9611325557]
MF2='in4mf2':'trapmf',[31908.7551765164 63611.9629885316 111166.77253 142869.97857]
[Input5]
Name='input5'
Range=[33.45 187.18]
NumMFs=2
MF1='in5mf1':'trapmf',[-74.161 -12.669 79.4294633853845 140.604803231249]
MF2='in5mf2':'trapmf',[79.5963737954478 140.921444658791 233.299 294.791]
[Output1]
Name='output'
Range=[34969 135286]
NumMFs=32
MF1='out1mf1':'linear',[1.87889077528655 0.356581017341143 0.254465695399984 9.21533753418034
0.372773520685096 0.00331311373255687]
MF2='out1mf2':'linear',[0.609868412728055 0.267916308944556 0.202915531794686 4.06069905793337
0.18691177982171 0.00144986995907038]
MF3='out1mf3':'linear',[0.02609096144391 0.00365014033449522 0.0031401080839359
0.705959663083553 0.00237259958483968 2.05831794216821e-05]
MF4='out1mf4':'linear',[0.0459706819078436 0.00855543254907792 0.00787410708210968 -
0.531884794173012 0.00535655730237302 4.32171783500657e-05]
MF5='out1mf5':'linear',[0.0726401858247748 0.0369368688612498 0.0285937273648867
1.08210798481671 0.0147210670735954 0.000205324102899381]
MF6='out1mf6':'linear',[0.182882261204246 0.0793423692518182 0.0646329376585627
2.64792078379504 0.0627989491384392 0.000430946231190982]
MF7='out1mf7':'linear',[0.00540188262462596 0.00077647980151837 0.000575884348433063
0.239882314721165 0.000568644610528341 4.33816163579668e-06]
MF8='out1mf8':'linear',[0.0222394112251025 0.00346315855426982 0.00255433019485421
0.943242285519479 0.00245396187423429 1.89833033832824e-05]
MF9='out1mf9':'linear',[0.67294617921145 0.339330602044415 0.229826571867217 7.28433569550985
0.131961153365894 0.0016992685436473]
MF10='out1mf10':'linear',[0.932817778767536 0.433839921731919 0.319721455183053
6.64562736221915 0.322278243018496 0.00226846144574714]
MF11='out1mf11':'linear',[0.0252095377838107 0.00384480862444992 0.00306937731480442
0.705343991638128 0.0026232020826902 2.07692979076366e-05]
MF12='out1mf12':'linear',[0.105723414119836 0.0180393454531394 0.0142201412605527
2.49052398472865 0.012364739788174 9.49770338176192e-05]
MF13='out1mf13':'linear',[0.0803030297153549 0.03858849725462 0.0296856389018248
1.34819214567703 0.0163988748361885 0.000210998744142789]
MF14='out1mf14':'linear',[0.711100043116565 0.330046575713355 0.341377557217355
6.37408981949445 0.250827871436821 0.00137921789474814]
MF15='out1mf15':'linear',[0.00549199987119919 0.000832797758790593 0.000610727344819736
0.22173834840905 0.000601779266005722 4.5171815010598e-06]
MF16='out1mf16':'linear',[0.0271357781308941 0.00434020051023818 0.00320606549602599
1.06776223629671 0.00305025186976674 2.33789525914383e-05]
MF17='out1mf17':'linear',[-0.311491997821817 -0.0618806342391351 -0.046359727353277 -
9.40101895625598 -0.036532400913594 -0.000359503099170684]
MF18='out1mf18':'linear',[-0.0879244396745382 -0.0339456051513988 -0.0288220239432695
0.638210276241297 -0.0201533219933131 -0.000198511528749006]
MF19='out1mf19':'linear',[0.0989006441648279 0.0113225266090018 0.0104109847673351
1.23019622015188 0.00953873862539293 6.29963002711936e-05]
MF20='out1mf20':'linear',[0.280299420125255 0.0347792553166455 0.0258632615523657
0.509617862984191 0.0228719030985239 0.000194386636591565]
MF21='out1mf21':'linear',[-0.0194934151627975 -0.00432319497802231 -0.00343449693889254 -
0.551254476580396 -0.00243519341861299 -2.47217326947794e-05]
MF22='out1mf22':'linear',[-0.0781176083705839 -0.0200053477329905 -0.0174495615875844 -
1.76265733050441 -0.0140471859934594 -0.000101589872147396]
MF23='out1mf23':'linear',[0.0207163176272092 0.00303034176034442 0.00228301103838122
0.931954745857679 0.00210449411711783 1.71674791207252e-05]
MF24='out1mf24':'linear',[0.0609762854628143 0.00933180665162924 0.00694736006129105
2.67646055419617 0.0065609137661387 5.17595867272627e-05]
MF25='out1mf25':'linear',[-0.130751186829373 -0.0292802010915555 -0.0226061570564588 -
3.68154002855208 -0.0161795232185144 -0.000175293988840275]
MF26='out1mf26':'linear',[-0.192012983410237 -0.0739230090159948 -0.0693807520752825 -
1.16977573853289 -0.0512446072460368 -0.00036757057063081]
MF27='out1mf27':'linear',[0.083957105988095 0.0112321437451025 0.00905877903855718
1.59322524118396 0.00845108788347234 6.17863629395432e-05]
MF28='out1mf28':'linear',[0.260081698043151 0.0375830810427272 0.028190726879148
0.845496383260956 0.0251406322237267 0.0002037687938213]
MF29='out1mf29':'linear',[-0.00734939727859272 -0.00237488460641481 -0.00211597562652736 -
0.137520046201311 -0.0012469563513409 -1.40976144710461e-05]
MF30='out1mf30':'linear',[-0.0522325731283927 0.00369321247519974 0.0355605849978808 -
2.24159679756522 0.00135678160839719 -0.000112885483752768]
MF31='out1mf31':'linear',[0.0172425856057266 0.00262658956297608 0.00196146220557136
0.722297014281121 0.00182088621507756 1.45220622665507e-05]
MF32='out1mf32':'linear',[0.0656995883878234 0.0104099815119225 0.00771747845932173
2.68455627500577 0.00730593613708159 5.64901460072267e-05]
[Rules]
1 1 1 1 1, 1 (1) : 1
1 1 1 1 2, 2 (1) : 1
1 1 1 2 1, 3 (1) : 1
1 1 1 2 2, 4 (1) : 1
1 1 2 1 1, 5 (1) : 1
1 1 2 1 2, 6 (1) : 1
1 1 2 2 1, 7 (1) : 1
1 1 2 2 2, 8 (1) : 1
1 2 1 1 1, 9 (1) : 1
1 2 1 1 2, 10 (1) : 1
1 2 1 2 1, 11 (1) : 1
1 2 1 2 2, 12 (1) : 1
```

```
1 2 2 1 1, 13 (1) : 1
1 2 2 1 2, 14 (1) : 1
1 2 2 2 1, 15 (1) : 1
1 2 2 2 2, 16 (1) : 1
2 1 1 1 1, 17 (1) : 1
2 1 1 1 2, 18 (1) : 1
2 1 1 2 1, 19 (1) : 1
2 1 1 2 2, 20 (1) : 1
2 1 2 1 1, 21 (1) : 1
2 1 2 1 2, 22 (1) : 1
2 1 2 2 1, 23 (1) : 1
2 1 2 2 2, 24 (1) : 1
2 2 1 1 1, 25 (1) : 1
2 2 1 1 2, 26 (1) : 1
2 2 1 2 1, 27 (1) : 1
2 2 1 2 2, 28 (1) : 1
2 2 2 1 1, 29 (1) : 1
2 2 2 1 2, 30 (1) : 1
2 2 2 2 1, 31 (1) : 1
2 2 2 2 2, 32 (1) : 1
```