2002

Monitoring process and assessing uncertainty for ANFIS time series forecasting

Yan K. Cathey Deng
West Virginia University

Follow this and additional works at: <https://researchrepository.wvu.edu/etd>

Recommended Citation

Deng, Yan K. Cathey, "Monitoring process and assessing uncertainty for ANFIS time series forecasting" (2002). *Graduate Theses, Dissertations, and Problem Reports*. 2437.
<https://researchrepository.wvu.edu/etd/2437>

This Dissertation is protected by copyright and/or related rights. It has been brought to you by the The Research Repository @ WVU with permission from the rights-holder(s). You are free to use this Dissertation in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you must obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/ or on the work itself. This Dissertation has been accepted for inclusion in WVU Graduate Theses, Dissertations, and Problem Reports collection by an authorized administrator of The Research Repository @ WVU. For more information, please contact researchrepository@mail.wvu.edu.

**Monitoring Process and Assessing Uncertainty
for ANFIS Time Series Forecasting**

Yan Deng

**Dissertation submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in Partial fulfillment of the requirements
for the degree of**

**Doctor of Philosophy
In
Industrial and Management Systems Engineering**

Powsiri Klinkhachorn, Ph.D.

Shahab Mohaghegh, Ph.D.

Alan McKendall, Ph.D.

Wafik H. Iskander, Ph.D., P.E.

Majid Jaraiedi, Ph.D., Chair

Department of Industrial and Management Systems Engineering

Morgantown, West Virginia

2002

Keywords: Time Series Forecasting, ANFIS, Tracking Signals, Bootstrapping

Copyright 2002 Yan Deng

ABSTRACT

Monitoring Process and Assessing Uncertainty for ANFIS Time Series Forecasting

Yan Deng

Although intelligent tools such as neural network, fuzzy logic and neuro-fuzzy methods have been applied in time series forecasting for some time, problems of monitoring forecasting processes and assessing uncertainty for the forecasts represent a major challenge that need to be fully investigated. In this research, we use statistical methods to analyze nonstationary time series forecasting where forecasts are accrued from a neuro-fuzzy ANFIS model. The main focus is to monitor the process and assess the uncertainty of the forecasts.

Single-step-ahead forecasts and multiple-step-ahead forecasts have been investigated by using three nonstationary time series data sets. It is shown that the tracking signals test provides an effective way to detect the nonrandom change in the forecasting process; while prediction intervals give good indication of the uncertainty and risk associated with the forecasts. The difficulty with building prediction intervals here lies in the derivation of the forecast error variance due to the underlying model structure. A bootstrapping technique has been used for forecast error variance estimation. Different numbers of bootstrapping replications B have been tested; the results showed that $B = 20$ is usually informative for neuro-fuzzy model forecasts resampling. Further, the tracking signal tests for the monitoring process and bootstrapping method for estimating forecast error variance could also be extended to other intelligent forecasting applications.

A comparison between the results from the ARIMA model and from the ANFIS forecast has also been provided. It is observed that, for single-step-ahead forecast, ARIMA model performs better, given the precondition that the model parameters were specified appropriately. ANFIS model in general performs well; and therefore it is better for multiple-step-ahead forecast since the current forecast does not depend on the previous forecasting values.

ACKNOWLEDGEMENTS

This dissertation would not have been possible without the support of many people to whom I will always be grateful. First, I would like to thank Dr. Majid Jaraiedi, an excellent teacher and advisor, for his continuous support and mentorship during my time at WVU. I have substantially benefited from him not only in academic area but also in many other aspects, and I will always remember his wisdom and humor.

I would also like to thank all my committee members, Dr. Powsiri Klinkhachorn, Dr. Shahab Mohaghegh, Dr. Alan McKendall, and Dr. Wafik H. Iskander, for their time and many suggestions. Dr. Iskander's invaluable suggestion in my research and his guidance as the graduate program advisor is gratefully acknowledged. My sincere appreciation also goes to Dr. Harsinder Singh in the Department of Statistics, who gave me very helpful advice for accomplishing this research.

I am very grateful to my parents, who always provide me with their whole-hearted love and support in my life. I am indebted to them for all their sacrifices and all those days and nights that they missed their children. I am also very fortunate to have my sister and my brother, who have been an immeasurable source of love and joy, which are always around me. Also, I extend special thanks to my friends, for their support and encouragement.

And last but not least, I would like to thank the Department of Industrial and Management Systems Engineering, West Virginia University, for giving me the opportunity to pursue my education and also for the consistent support I received during my graduate studies there.

TABLE OF CONTENTS

ABSTRACT	ii
ACKNOWLEDGEMENTS	iii
LIST OF TABLES	viii
LIST OF FIGURES	ix
1 Introduction and Research Objectives	1
1.1 Background	1
1.2 Methods of Forecasting.....	2
1.3 Neuro-Fuzzy and Soft Computing.....	5
1.3.1 Neural Networks	5
1.3.2 Fuzzy Logic	8
1.3.3 Soft Computing.....	11
1.4 Intelligent Forecasting	12
1.5 Need for Research.....	14
1.6 Research Objectives and Organization	18
1.6.1 Research Objectives.....	18
1.6.2 Organization.....	20
2 Background and Literature Review.....	22
2.1 Fuzzy Logic	22
2.1.1 Fuzzy Sets	23
2.1.2 Membership Functions.....	24
2.1.3 Fuzzy If-Then Rules	25
2.1.4 Fuzzy Reasoning.....	26

2.1.5	Fuzzy Inference Systems	27
2.1.6	Fuzzy Modeling	30
2.2	Neural Networks	31
2.2.1	Supervised Learning	31
2.2.2	Unsupervised Learning	34
2.3	Neuro-Fuzzy Systems	36
2.3.1	General Neuro-Fuzzy Architecture	38
2.3.2	ANFIS Architecture	39
2.3.3	Hybrid Learning	42
2.4	Comparison of Three Intelligent Forecasting Methods	43
2.4.1	Fuzzy Forecasting	44
2.4.2	Neural Network Forecasting	45
2.4.3	Neuro-Fuzzy Forecasting	48
2.5	Tracking Signals	49
2.5.1	Simple CUSUM Tracking Signals	50
2.5.2	Smoothed Error Tracking Signals	50
2.5.3	Backward CUSUM Control Systems	51
2.5.4	Autocorrelation Tracking Signals	52
2.6	Variance Estimation	52
2.6.1	Overview	53
2.6.2	The Bootstrap: General Ideas	54
2.6.3	Interval Estimation	57
3	Overview of Methodology	59
3.1	Generating forecasts	59
3.2	Monitoring forecasting process	60
3.3	Estimating forecast error variance	61
4	Generating Forecasts Using ANFIS	64

4.1	The ANFIS Model	64
4.1.1	Network Structure	65
4.1.2	Learning Algorithm	66
4.2	Forecasting.....	73
4.2.1	Data Clustering	73
4.2.2	Forecasting Procedures	75
4.3	The Data.....	78
4.3.1	Case I	78
4.3.2	Case II	79
4.3.3	Case III.....	80
5	Monitoring The Forecasting Process	81
5.1	Tracking Signals Test	81
5.2	Procedures.....	83
5.3	Data Description	85
5.3.1	Case I	86
5.3.2	Case II	95
5.3.3	Case III.....	99
6	Building Prediction Intervals.....	104
6.1	Residual Analysis.....	105
6.2	Estimating Forecast Error Variance.....	109
6.2.1	Bootstrap Residual Algorithm	109
6.2.2	Bootstrap Replications	110
6.3	Prediction Intervals	119
7	Computational Experience.....	121
7.1	Using ARIMA Model	121
7.1.1	Case I	122

7.1.2	Case II	132
7.1.3	Case III.....	137
7.2	Using ANFIS Model.....	140
7.2.1	Case I	140
7.2.2	Case II.....	144
7.2.3	Case III.....	148
7.3	Comparison of Forecasting Performance.....	152
8	Conclusions and Future Research Recommendations	156
8.1	Summary of Research	156
8.2	Contributions of Research.....	158
8.3	Recommendations for Future Research	159
	Bibliography	160
	Appendices.....	170
	Appendix I Basic Concepts and Terminology of Membership Functions.....	171
	Appendix II Four Commonly Used Membership Functions	172
	Appendix III Summary Statistics of Fitting Distributions for Residuals from Area	176
	Appendix IV Example S-PLUS Language Code to Generate Bootstrapping Replicates for Residuals When $B = 20$	179
	Appendix V Forecasting Example for Weekly Sales of a Cutting Tool Data	180
	Appendix VI Forecasting Example for Quarterly S&P 500 Index Data	183
	Appendix VII Forecasting Example for Quarterly Beer Production Data in Australia	190

LIST OF TABLES

Table 7.1	Performance of the ARIMA model for Case I	153
Table 7.2	Performance of the ARIMA model for Case II	153
Table 7.3	Performance of the ARIMA model for Case III	154
Table 7.4	Performance of the ANFIS model for Case I	154
Table 7.5	Performance of the ANFIS model for Case II.....	154
Table 7.6	Performance of the ANFIS model for Case III	155

LIST OF FIGURES

Figure 2.1	Cores, supports, boundaries, crossover points of membership function	23
Figure 2.2	Node j of a backpropagation MLP	32
Figure 2.3	A backpropagation multilayer perceptron	33
Figure 2.4	Reducing neighborhoods around node x	35
Figure 2.5	General neuro-fuzzy architecture	38
Figure 2.6	A two-input first-order Sugeno fuzzy model	40
Figure 2.7	Equivalent ANFIS architecture	41
Figure 2.8	Neural network forecasting	46
Figure 4.1	ANFIS forecasting structure	66
Figure 4.2	Structure of updating p_1 for MF A_1	71
Figure 4.3	Flowchart for training process	76
Figure 4.4	Flowchart for testing process	77
Figure 4.5	Weekly sales of a cutting tool	78
Figure 4.6	Quarterly S&P 500 index (1900-1996)	79
Figure 4.7	Quarterly beer production in Australia (Mar 1956 - Jun 1994)	80
Figure 5.1	Flowchart for tracking signals test	84
Figure 5.2	Data import and training preparation (cutting tool data)	87
Figure 5.3	Training set partition and MFs clustering (cutting tool data)	88
Figure 5.4	Training results for three-step-ahead forecasts (cutting tool data)	89
Figure 5.5	Plot of residuals for training set (cutting tool data)	90
Figure 5.6	Three-step forecasting results without tracking signals test (cutting tool data)	91
Figure 5.7	Plot of residuals for testing set (cutting tool data)	92
Figure 5.8	Indication of forecasting out of control (cutting tool data)	93
Figure 5.9	Plot of residuals for testing set after tracking signals applied (cutting tool data)	94
Figure 5.10	Training stage for three-step-ahead forecasts (S&P 500 index data)	96

Figure 5.11 Plot of residuals for training set (S&P 500 index data)	97
Figure 5.12 Plot of residuals for testing set (S&P 500 index data)	98
Figure 5.13 Plot of residuals for testing set after tracking signals test (S&P 500 index data)	99
Figure 5.14 Training stage for four-step-ahead forecasts (beer production data)	100
Figure 5.15 Plot of residuals for training set (beer production data)	101
Figure 5.16 Plot of residuals for testing set (beer production data)	102
Figure 5.17 Plot of residuals for testing set after tracking signals test (beer production data)	102
Figure 6.1 Distribution fitting for cutting tool data training errors: Normal (-0.042, 2.51)	107
Figure 6.2 Distribution fitting for S&P index data training errors: Normal (0.0109, 0.183)	108
Figure 6.3 Distribution fitting for beer production data training errors: Normal (-0.149, 16.2)	108
Figure 6.4 Bootstrap replications of mean for cutting tool data training errors	111
Figure 6.5 QQ plot of bootstrap mean for cutting tool data training errors	112
Figure 6.6 Bootstrap replications of variance for cutting tool data training error.....	113
Figure 6.7 QQ plot of bootstrap variance for cutting tool data training errors	113
Figure 6.8 Bootstrap replicates of residuals for cutting tool training data	114
Figure 6.9 Bootstrap replicates of training set for cutting tool training data	115
Figure 6.10 Original training set for cutting tool data	117
Figure 6.11 One bootstrap replicate of training set for cutting tool data	117
Figure 7.1 Autocorrelation plots for cutting tool data	123
Figure 7.2 White noise and stationarity test for cutting tool data	124
Figure 7.3 Data series after the first difference taken for cutting tool data	125
Figure 7.4 Autocorrelations for the first difference of the cutting tool data	126
Figure 7.5 White noise and stationarity test for the first difference of the cutting tool data	127

Figure 7.6	One-step-ahead forecast using ARIMA (2, 1, 0) (cutting tool data)	129
Figure 7.7	Prediction intervals of one-step-ahead forecast using ARIMA (2, 1, 0) (cutting tool data)	129
Figure 7.8	Residuals of one-step-ahead forecast using ARIMA (2, 1, 0) (cutting tool data)	130
Figure 7.9	Three-step-ahead forecast using ARIMA (2, 1, 0) (cutting tool data)	131
Figure 7.10	Residuals of three-step-ahead forecast using ARIMA (2, 1, 0) (cutting tool data)	132
Figure 7.11	Autocorrelation plots for S&P 500 index data	133
Figure 7.12	Data series after the first difference taken for S&P 500 index data	134
Figure 7.13	One-step-ahead forecast and the prediction intervals using ARIMA (2, 1, 0) (S&P 500 index data)	135
Figure 7.14	Residuals of one-step-ahead forecast using ARIMA (2, 1, 0) (S&P 500 index data)	135
Figure 7.15	Three-step-ahead forecast using ARIMA (2, 1, 0) (S&P 500 index data)	136
Figure 7.16	Residuals of three-step-ahead forecast using ARIMA (2, 1, 0) (S&P 500 index data)	136
Figure 7.17	One-step-ahead forecast and the prediction intervals using ARIMA (2, 1, 2) (beer production data)	138
Figure 7.18	Residuals of one-step-ahead forecast using ARIMA (2, 1, 2) (beer production data)	138
Figure 7.19	Four-step-ahead forecast using ARIMA (2, 1, 2) (beer production data)	139
Figure 7.20	Residuals of four-step-ahead forecast using ARIMA (2, 1, 2) (beer production data)	139
Figure 7.21	One-step-ahead forecast using ANFIS (3, 2, 1) (cutting tool data)	141
Figure 7.22	Residuals of one-step-ahead forecast using ANFIS (3, 2, 1) (cutting tool data)	141

Figure 7.23	Three-step-ahead forecast using ANFIS (3, 2, 3) (cutting tool data)	142
Figure 7.24	Prediction intervals for three-step-ahead forecast using ANFIS (3, 2, 3) When $B = 20$, $\alpha = 5\%$ (cutting tool data)	143
Figure 7.25	Prediction intervals for three-step-ahead forecast using ANFIS (3, 2, 3) When $B = 30$, $\alpha = 5\%$ (cutting tool data)	143
Figure 7.26	Residuals of three-step-ahead forecast using ANFIS (3, 2, 3) (cutting tool data)	144
Figure 7.27	One-step-ahead forecast using ANFIS (3, 2, 1) (S&P 500 index data)	145
Figure 7.28	Residuals of one-step-ahead forecast using ANFIS (3, 2, 1) (S&P 500 index data)	146
Figure 7.29	Three-step-ahead forecast using ANFIS (3, 2, 3) (S&P 500 index data)	146
Figure 7.30	Residuals of three-step-ahead forecast using ANFIS (3, 2, 3) (S&P 500 index data)	147
Figure 7.31	Prediction intervals for three-step-ahead forecast using ANFIS (3, 2, 3) When $B = 20$, $\alpha = 5\%$ (S&P 500 index data)	147
Figure 7.32	Prediction intervals for three-step-ahead forecast using ANFIS (3, 2, 3) When $B = 30$, $\alpha = 5\%$ (S&P 500 index data)	148
Figure 7.33	One-step-ahead forecast using ANFIS (3, 2, 1) (beer production data)	149
Figure 7.34	Residuals of one-step-ahead forecast using ANFIS (3, 2, 1) (beer production data)	149
Figure 7.35	Four-step-ahead forecast using ANFIS (3, 2, 4) (beer production data)	150
Figure 7.36	Residuals of four-step-ahead forecast using ANFIS (3, 2, 4) (beer production data)	150
Figure 7.37	Prediction intervals for four-step-ahead forecast using ANFIS (3, 2, 4) When $B = 20$, $\alpha = 5\%$ (beer production data)	151

Figure 7.38	Prediction intervals for four-step-ahead forecast using ANFIS (3, 2, 4)	
	When $B = 30$, $\alpha = 5\%$ (beer production data)	151

Chapter 1

Introduction and Research Objectives

Forecasting is concerned with the process of predicting the unknown. Realistically, forecasting plays an important role in many aspects of our lives and has been found to be an essential function for government and business enterprises. From the scientific point of view, forecasting is a process of understanding the past and predicting the future. As an introduction, we first discuss certain aspects of forecasting in this chapter, i.e., the background of forecasting, and the methods and new techniques related to this field. We then present the need for the research, based on the literature survey, and explain the objectives of this research.

1.1 Background

Forecasting is defined as estimating in “unknown” situations (Armstrong, 2000). Although “Forecasting” and “Predicting” are typically used interchangeably, forecasting usually connotes an estimation of the future. In fact, forecasting is a subset of futures research which encompasses “any activity that improves understanding about the future consequences of present developments and choices” (Amara and Salanik, 1972). These authors offer the following progression in the process of generating a forecast:

- a statement about the future;
- a probabilistic statement about the future;

- a probabilistic, reasonably definite statement about the future;
- a probabilistic, reasonably definite statement about the future, based on an evaluation of alternative possibilities.

Martino (1983) states that technological forecasting includes four elements: the time of the forecast or the future date when the forecast is to be realized, the technology being forecast, the characteristics of the technology or the functional capabilities of the technology, and a probabilistic statement about the forecast.

Forecasting is a difficult task “beset with hazards” (Ayers, 1969). These hazards include: “the uncertainty and unreliability of data, the complexity of ‘real world’ feedback interactions, the temptation of wishful or emotional thinking, the fatal attraction of ideology, (and) the dangers of forcing soft and somewhat pliable ‘facts’ into a preconceived pattern.”

Forecasting plays an important role in our real life in spite of the fact that inaccuracy is an inherent part of any forecasting procedure. It helps people to plan for the future, to make rational decisions and to reduce the risk in an atmosphere of uncertainty. As a matter of fact, the magnitude of forecasting errors can be decreased and forecasting accuracy can be improved depending on the forecasting system used.

1.2 Methods of Forecasting

Forecasting procedure can be broadly classified as qualitative or quantitative depending upon the extent to which mathematical and statistical methods are used (Montgomery, Johnson and Gardiner, 1990). Qualitative methods are primarily based on

judgment, which involves subjective estimation through the opinions of experts, while quantitative methods are based on statistical techniques. Statistical methods involve examination of historical data to determine the underlying process and use this knowledge to extrapolate the process into the future. The two basic types of statistical methods are time series analysis and causal models.

A time series is a time-ordered sequence of observations of a variable (Montgomery, Johnson and Gardiner, 1990). Time series analysis utilizes the history of the variable being forecast to predict its future values. As an example, the Gross National Product (GNP) might be expressed as a time series model as follows:

$$\text{GNP}_{t+1} = f(\text{GNP}_t, \text{GNP}_{t-1}, \text{GNP}_{t-2}, \text{GNP}_{t-3}, \text{GNP}_{t-4}, \dots, \text{error})$$

This model states that, the GNP of next period will depend upon the GNP of the previous period and the periods before. On the other hand, causal models assume that the variable to be forecasted (dependent variable) exhibits an explanatory relationship with one or more independent variables. Likewise, the GNP could be described as a causal model as follows:

$$\text{GNP} = f(\text{monetary and fiscal policies, inflation, capital spending, imports, exports}) + \text{error}$$

This means that the GNP value will be affected by monetary and fiscal policies, the inflation rate, the capital spending amount, the imports and exports, and even some changes which cannot be modeled. Notice that the relationship is not exact; hence, unpredictable factors might be included in the “error” term.

Various methods have been developed to cater to different kinds of forecasting problems. These methods include: regression analysis, moving averages, exponential smoothing, direct smoothing, and Box-Jenkins methods. Traditional forecasting methods are appropriate if the problem is simple or both the dependent variable and decision variable are well defined. However, these methods suffer from several deficiencies and limitations. First, forecasting environments have become more and more complex; the relationships between the forecasting output and the decision factors that influence it cannot always be expressed by a mathematical model. Second, today's forecasting environment is constantly changing, thus causing the decision boundaries to shift. Conventional forecasting methods lack the mechanisms to deal with these changes and therefore cannot learn from these changes and adapt to them (Li, Ang, Gary, 1999).

Recently, there have been many attempts to address these shortcomings by the application of artificial intelligence (AI) techniques. Tools such as neural network (NN), fuzzy logic (Zadeh, 1965), and genetic algorithm (GA), etc., have been developed in this field. The strengths of AI techniques accrue from the fact that they do not make *a priori* assumption in the models and from the fact that they have the capability to infer complex underlying relationships. From the statisticians' point of view, these tools are essentially statistical devices for performing inductive inference and are analogous to non-parametric, non-linear regression models (Refenes, 1995; Smith, 1993; Deboeck, 1994). It has been shown that nonlinear domain can be described more accurately with these technologies than with the linear statistical method that has been the mainstay of technical analysis throughout the past decades.

1.3 Neuro-Fuzzy and Soft Computing

Analysis of real world problems requires intelligent systems. Soft Computing (SC) is an innovative approach to constructing computationally intelligent systems (Jang, Sun and Mizutani, 1997). These intelligent systems, which combine knowledge, techniques, and methodologies from various sources, are supposed to possess human-like expertise within a specific domain, adapt themselves and learn to do better in changing environments, and explain how they make decisions or take actions. In confronting complex real-world computing problems, it is frequently advantageous to use several computing techniques synergistically rather than exclusively, resulting in the construction of complementary hybrid intelligent systems. One of the most successful of this kind of intelligent systems design is neuro-fuzzy computing: neural networks recognize patterns and learn from examples; fuzzy inference systems incorporate human knowledge and perform inferencing. In the following section, a brief description of these emerging fields is provided.

1.3.1 Neural Networks

A neural network is a parallel, distributed information processing structure consisting of processing elements (which can possess a local memory and can carry out localized information processing operations) interconnected via unidirectional signal channels called connections. Each processing element has a single output connection that branches (“fans out”) into as many collateral connections as desired; each carries the same signal – the processing element output signal. The

processing element output signal can be of any mathematical type desired. The information processing that goes on within each processing element can be defined arbitrarily with the restriction that it must be completely local; that is, it must depend only on the current values of the input signal arriving at the processing element's local memory (Hecht-Nielsen, 1990).

Clearly, neural networks are models based on the working mechanism of the human brain; they are composed of individual interconnected processing elements (PEs), which are analogous to neurons in the brain and utilize a distributed processing approach to computation.

More specifically, anything that can be represented as a number might be fed into a neural network. Each PE sends/receives data to/from other PEs. For each individual PE in standard model, input data ($X_0 \dots X_n$) are multiplied by the weights ($W_0 \dots W_n$) associated with the connection to the PE. Each PE applies a nonlinear activation function to its sum of weighted input signals to determine its output signal. The output from a given PE is multiplied by another separate weight and fed into the next processing element. If the processing element is in the output layer, then the output from the processing element is not multiplied by a weight and instead is an output of the network itself.

The origin of the neural network field began in the 1940s with the work of McCulloch and Pitts (1943), who showed that networks of artificial neurons could, in principle, compute any arithmetic or logical function. They also showed that any arbitrary logical function could be configured by a neural network of interconnected digital

neurons, which introduced the idea of the step threshold used in many neural network models. The first practical application of artificial neural networks was presented by Rosenblatt in the late 1950s. In his book published in 1962, *Principles of Neurodynamics*, he introduced a learning algorithm by which the weights can be changed, and he demonstrated the ability to perform pattern recognition in a perceptron network. At about the same time, Widrow and Hoff introduced a new learning algorithm in 1960 and used it to train adaptive linear neural networks, which were similar in structure and capability to Rosenblatt's perceptron.

Unfortunately, both Rosenblatt's and Widrow's networks suffered from the same inherent limitations as pointed out in the book *Perceptrons* by Minsky and Papert, published in 1969. They showed that single-layer systems were limited and expressed pessimism over multilayer systems. Interest in neural networks dwindled from late 1960s to early 1980s.

The breakthrough of neural network came in the 1980s when the most influential method of training a multilayer neural network, known as the backpropagation (BP) algorithm was developed by Parker (1982) and Rumelhart & McClelland (1986). About the same time, new types of neural net with dynamic behavior, such as Hopfield neural net (Hopfield, 1982; 1984) and the Kohonen self-organizing neural net (Kohonen, 1982; 1984), were introduced. These new developments reinvigorated the field of neural networks.

Neural networks are capable of solving a wide range of problems by "learning", "generalizing" and "abstracting". They can modify their behavior in response to their

environment and once trained, the network's response can be tolerant to minor variations to its input. As a matter of fact, neural networks have been widely used in a broad range of areas such as image processing, signal processing, pattern recognition, speech recognition, industrial control, aerospace, manufacturing, medicine, business, finance, and even literature. The success in application of neural networks is mostly because of their applicability to complex nonlinear systems and multivariable systems.

1.3.2 Fuzzy Logic

We need a radically different kind of mathematics, the mathematics of fuzzy or cloudy quantities which are not described in terms of probability distributions. Indeed, the need for such mathematics is becoming increasingly apparent... for in most practical cases the a priori data as well as the criteria by which the performance of a man-made system is judged are far from being precisely specified or having accurately known probability distributions (Zadeh, 1961).

Fuzzy set theory, originally introduced by Lotfi Zadeh in the 1960's, resembles human reasoning in its use of approximate information and uncertainty to generate decisions. It was specifically designed to mathematically represent uncertainty and vagueness and provide formalized tools for dealing with the imprecision intrinsic to many problems.

Zadeh's idea of membership grade is the backbone of fuzzy set theory. In 1965, the publication of his seminal paper on fuzzy sets declared the birth of fuzzy logic technology. Narrowly speaking, fuzzy logic refers to a logical system that generalizes

classical two-valued logic for reasoning under uncertainty. Broadly speaking, fuzzy logic refers to all of the theories and technologies that employ fuzzy sets, which are classes with unsharp boundaries (Yen and Langari, 1999).

Even though the concept of fuzzy sets encountered sharp criticism from the academic community at the beginning, many researchers around the world still kept stepping into this field. During the first decade (1965-1975), Zadeh continued to broaden the foundation of fuzzy set theory. He introduced fuzzy multistage decision-making, fuzzy similarity relations, fuzzy restrictions, and linguistic hedges. Mamdani and Assilian (1975) developed the first fuzzy logic controller to control a steam generator in 1974. In 1976, the first industrial application of fuzzy logic was developed by Blue Circle Cement and SIRA in Denmark. Another successful application is a fuzzy logic based automatic train operation control system in Sendai city's subway system developed by Yasunobu and his colleagues at Hitachi in 1987. Researchers in Japan made many important contributions to the theory as well as to the applications. In 1980s, Takagi and Sugeno developed the first approach for constructing fuzzy rules based on the training data. This important work did not gain much immediate attention, but it built the foundation for fuzzy model identification.

The fuzzy boom in Japan triggered a broad interest in the world. Fuzzy logic is now being widely used in aerospace, defense, automobile, consumer products, industry, manufacturing, business and finance. The main reason for its popularity is that it utilizes concepts and knowledge that do not have well-defined, sharp boundaries; therefore, it can

alleviate the difficulties encountered by conventional mathematical tools in developing and analyzing complex systems.

Fuzzy set theory implements classes or groupings of data with boundaries that are not sharply defined. Any methodology or theory implementing “crisp” definitions such as classical set theory, arithmetic, and programming, may be “fuzzified” by generalizing the concept of a crisp set to a fuzzy set with blurred boundaries. The benefit of extending crisp theory and analysis methods to fuzzy techniques is the strength in solving real world problems, which inevitably entail some degree of imprecision and noise in the variables and parameters measured and processed. Fuzzy logic comprises of fuzzy sets and fuzzy rules which combine numerical and linguistic data. Linguistic variables are a critical aspect of some fuzzy logic application, where general terms such as “large”, “medium”, and “small” could be used to capture a range of numerical values. Such terms are not precise and cannot be represented in normal set theory. While similar to conventional quantization, fuzzy logic allows these stratified sets to overlap and allows members to be partial members as well as the normal multi-set membership.

Since fuzzy logic can handle approximate information in a systematic way, it is ideal for dealing with nonlinear systems and for modeling complex systems where no exact model exists or systems where ambiguity or vagueness is common.

1.3.3 Soft Computing

Soft computing is an emerging approach to computing which parallels the remarkable ability of the human mind to reason and learn in an environment of uncertainty and imprecision (Zadeh, 1992).

Soft computing consists of several computing paradigms, including neural networks, fuzzy set theory, approximate reasoning, and derivative-free optimization methods such as genetic algorithms and simulated annealing. As for the major part of these constituent methodologies, neural network has the strength of learning and adaptation, fuzzy logic has the strength of knowledge representation via fuzzy if-then rules, and genetic algorithm is suitable for systematic random search.

Although fuzzy logic and neural network emphasize different strengths, these two innovative modeling approaches share some common characteristics: they assume parallel operations; they are well known for their fault tolerance capabilities; and they have the ability of model-free learning, i.e. the ability to construct models using only target system sample data. Despite these similarities, they stem from very different origins. Primarily, fuzzy logic modeling is based on fuzzy sets and fuzzy if-then rules proposed by Zadeh, which are closely related to psychology and cognitive sciences, while neural network modeling is based on artificial neural networks which are motivated by biological neural systems (Jang, 1992). Because of their very origins, the respective philosophies and methodologies underlying their problem solving approaches are quite different and, in general, complementary. Therefore, they can be integrated to generate hybrid models that can take advantage of the strong points of both.

1.4 Intelligent Forecasting

Conventional forecasting approaches rely heavily on mathematical tools which emphasize the precision and exact description of each quantity involved. The use of these mathematical tools (such as differential or difference equations, transfer functions, etc.) is appropriate when the system is simple. However, as the system under consideration grows larger and more complicated, mathematical tools become less effective or even inappropriate. This is due to the fact that either the mathematical expressions themselves are too complicated to be tractable, or the relationships between variables in the system are unclear or known with uncertainty.

To overcome the problem confronted by conventional forecasting methods, fuzzy logic modeling and neural network modeling have been proposed as viable alternatives. In recent years, some research has been conducted using fuzzy logic, neural networks or both in different domains of forecasting.

Neural network has been used in forecasting since Widrow introduced his Adaline (Adaptive linear neuron) and Madaline (Many adaptive linear neuron) learning (Widrow, 1985). He applied this structure to forecast the weather. To date, neural network technology has been widely accepted in forecasting field. An extensive search of the literature showed that the main activity in neural network forecasting is focused on financial and business area due to today's complex trading environment.

Foster, Collopy, and Ungar (1992) studied short, noisy time series using neural network. They pointed out that the use of neural networks as function approximators resulted in forecasts that were generally less accurate than linear regression. They

suggested combining neural networks with traditional forecasts, which gave small but significant performance improvements.

Devika and Achenie (1994) presented a feed forward neural network to implement forecasting by using experience. They use some statistical features such as standard deviation to train the network and found that the network produces better predictions compared to the traditional methods.

Thiesing and Vornberger (1997) studied neural networks trained with backpropagation algorithm as applied to sales forecasting in a supermarket. They considered the indicators of price, advertising campaigns and holidays into the model. Refenes and Zapranis (1999) introduced a methodology for neural model identification which facilitates hypothesis testing at two levels: model adequacy and variable significance.

Fuzzy forecasting is another branch in the intelligent forecasting area. Wu (1999) proposed a procedure to detect the fuzzy change period of a time series based on the value of the presumption level, α . He showed that the approach to detect structure change of fuzzy time series is practical, but he left the problem for multivariate time series and seasonal time series open. Kim and Lee (1999) presented a time series prediction method using fuzzy rule-based system. They suggested a new learning method utilizing the difference of consecutive values to handle non-stationary data whose long-term mean is floating.

Fiordaliso (1998) presented a nonlinear forecasts combination method based on Sugeno fuzzy systems. Palit and Popovic (1999) proposed an algorithm to generate the

fuzzy rules from time series data and used a lookup table to build a combined fuzzy rule base. They concluded that the problem of optimal number of selected membership functions and the extent of their overlap for obtaining a high forecasting accuracy still remains to be solved.

Meanwhile, we have observed that a new trend has emerged recently which combines neural network forecasting with fuzzy logic forecasting.

Thammano (1999) employed neural network and fuzzy system to build neuro-fuzzy architecture in planning. In his research, neural network is used to determine output while fuzzy-inference evaluates the performance by analyzing the error term. Li, Ang and Gray (1999) presented a multilayer fuzzy rule-based neural network which integrates a traditional fuzzy logic inference into a neural network structure. They showed that the forecast results outperform the other two business forecasters.

1.5 Need for Research

Forecasting is a key element of management decision making for the reason that it is closely related to strategic planning. In recent years, forecasting has gained more and more attention because of its critical role in today's competitive market, and hence the impact on the price revenue management (PRM). As an example, in manufacturing industry, forecasting is an activity that goes through supply and demand channels, starting with planning and scheduling of production, the usage of materials, inventory control, and ending with the future sales of products.

Practically, inaccuracy is an inherent part of any forecasting procedure; therefore, uncertainty or risk is always associated with the forecast, and stating the uncertainty associated with a forecast conveys useful information to the decision maker. Indeed, a simple point forecast is no longer sufficient for many forecasting models which need to take explicit account of risk and uncertainty; an interval forecast is usually more informative than point estimate alone. The importance of giving interval forecasts had been clearly stated by Chatfield (1993), which can be summarized as:

- 1) Assess future uncertainty.
- 2) Enable different strategies to be planned for the range of possible outcomes indicated by the interval forecast.
- 3) Compare forecasts from different methods more thoroughly and explore different scenarios based on different assumptions.

However, he also added that, “given the above importance, it is perhaps surprising and rather regrettable that many companies do not regularly produce PI’s (Prediction Intervals) and that most economic predictions are given as a single value.”

Needless to say, forecast decision is always related to an allowance for forecast error. Forecast accuracy can be quantitatively described by the variance of the forecast error. Usually, expressing forecast takes one of the following forms:

- 1) An estimate of the expected value of a variable plus an estimate of the standard deviation of the forecast error;
- 2) An interval that has a stated probability of containing the actual future value.

Prediction intervals are very useful in describing the uncertainty associated with the forecast. As mentioned before, different forecasting methods have been applied in the forecasting field for several decades. The calculation of prediction intervals for these models has been researched by a number of authors. Yar and Chatfield (1990) derived the prediction interval formulae for the Holt-Winters forecasting procedure with an additive seasonal effect. Chatfield (1993) summarized several general approaches to calculating interval forecasts. Masarotto (1990) and McCullough (1994) applied a bootstrap technique to obtain interval forecasts for an autoregressive time series model. Koehler, Snyder, and Ord (2001) showed the forecasting models and prediction intervals for the multiplicative Holt-Winters method with seasonality effect. Clements and Taylor (2001) explored methods of improving the coverage of Box-Jenkins prediction intervals for linear autoregressive models.

As the attention to forecasting process continues to grow, new techniques such as intelligent forecasting continues to be developed. However, the literature survey revealed that although the application of neural network, fuzzy logic or both have been growing in the forecasting area, most of these research works are mainly focused on the forecasting procedure; they merely try to generate a single point forecast value, few of them go so far as to perform any statistical analysis of the results to describe the uncertainty or risk associated with the forecast, which could be informatively described by using prediction intervals. From a forecasting professional's point of view, these analyses are indispensable and essential to time series forecasting system but remain relatively less addressed and weakly recognized in areas of neural network forecasting, fuzzy

forecasting and neuro-fuzzy forecasting. The difficulty with them is mainly because these models are generally nonlinear, while the traditional methods are mostly used for linear models.

Gaps in research related to constructing prediction intervals in neural network forecasting have been pointed out by a few researchers. In a special section of the International Journal of Forecasting under the title of *neural network in forecasting*, Gorr (1994) discussed the research prospective on neural network forecasting. He stated “the non-linear neural network model form does not possess classical statistical properties so the confidence intervals and hypothesis testing are not available”. Taylor and Bunn (1999) investigated the accuracy of prediction intervals for combinations of forecasts; however, in their summary, they left the question for neural network models open. Also, we observed that only a few attempts have been made to find prediction intervals for neural network models. Heskes (1997) proposed a method to compute prediction intervals for a feed-forward neural network for regression tasks. Bartkiewicz (2000) computed the prediction intervals for a standard feed-forward neural forecasting model by using local linearization of the network. Other than these, very little research has been done on the construction of prediction intervals in intelligent forecasting, especially in neuro-fuzzy modeling.

Another important prospective of forecasting is monitoring the process. Monitoring is an essential aspect of any forecasting situation (Makridakis and Wheelwright, 1989). Usually, there are two types of situations in connection with forecasting: (1) the continuation of the established patterns or relationships, and (2)

nonrandom (systematic) changes from established patterns or relationships as time goes by.

Different monitoring approaches have been proposed in the forecasting area; among them, tracking signals test is not only a useful but also an effective method developed for identifying the occurrence of nonrandom changes. It can automatically detect changes in the forecasting errors when the forecast is misbehaving. Tracking signals provide quality control for forecasting (McClain, 1988).

Tracking signals test plays an even more important role when it comes to time series forecasting. This is because time series is a time-ordered sequence of observations of a variable; hence it implies that the underlying system is dynamic. When the time horizon lengthens, it is critical to monitor the process and determine if the previous pattern or relationship has been changed or not. When a nonrandom change occurs, a correspondent action needs to be taken to ensure that the forecasting process is brought back under control. However, search of the literature showed that the tracking signals test, for monitoring the time series forecasting process, has not been investigated in neural network forecasting, fuzzy forecasting, or neuro-fuzzy forecasting, which obviously points to a gap in this research area that needs to be addressed.

1.6 Research Objectives and Organization

1.6.1 Research Objectives

As mentioned earlier, quantitative forecasting methods include causal models and time series analysis. Likewise, intelligent forecasting can be implemented

correspondingly based on the above two categories. For the causal models, one needs to find the influential factors that affect the variable to be forecasted, then use the mechanism of neural networks and/or fuzzy logic to find the relationships between the response variable and the decision variables. For the time series analysis, learning and inferencing are from its past values and past errors, hence the forecast is generated from its own behavior.

In this research, we will focus on time series forecasting. The objective here is to provide a complete set of decision making tools to analyze times series and address uncertainty problems involved in the forecasting system comprehensively, i.e., we will generate forecasts by using neuro-fuzzy method, which is a suitable paradigm for uncertain modeling. We will also assess forecasts uncertainty by building prediction intervals. Although our forecasts will be generated using neuro-fuzzy technique, the scope of this research will not be merely focused on generating forecasts for a specific problem. Keeping in mind that time series analysis is a dynamic problem where the process is changing over time; we will develop a tracking signals test for monitoring the forecasting process, which will complement the neuro-fuzzy paradigm where the re-learning and re-inferencing are enforced based on statistical criteria. The neuro-fuzzy paradigm employed in this research will be the Adaptive Neuro Fuzzy Inference System (ANFIS) architecture, as it possesses the features and advantages of soft computing techniques. Details of the basic ANFIS will be given in Chapter 2. Three sets of nonstationary time series data will be used in the processes of generating forecasts, analyzing the outputs and comparison with traditional forecasting methods. To

implement the system, Microsoft Visual Basic will be used as the development environment.

Specifically, the major objectives of this research are:

1. Generate forecasts by using neuro-fuzzy paradigm, ANFIS model, to implement intelligent time series forecasting.
2. Incorporate tracking signals test into the ANFIS model and monitor the process dynamically. This will be a major contribution which, for the first time, applies statistical tracking signals test to neuro-fuzzy techniques for time series forecasting.
3. Devise a procedure to estimate the forecast error variance for ANFIS model, and build prediction intervals for forecasts based on the derived variance. This has not been done before and will be another main contribution of this research.

1.6.2 Organization

The scope of this research is to analyze a time series forecasting system in which the forecasts are accrued from a neuro-fuzzy ANFIS model; the main focus is to monitor the process and assess the uncertainty associated with the forecasts. Obviously, this task is the combination of different domains and therefore achieving the goal calls for different techniques to be involved. These techniques include neural network, fuzzy logic, neuro-fuzzy ANFIS model, tracking signals test, and bootstrap sampling approach.

This dissertation will be organized as follows: Chapter 2 gives some systematic literature reviews of the topics related to the research. Chapter 3 outlines the

methodology to be used in this research. Chapter 4 presents the forecasting model structure, the hybrid learning and inferencing mechanism, and the forecasting procedure. Chapter 5 describes the tracking signals test developed and applied to three sets of data used in this research for illustration purposes. Chapter 6 shows the comprehensive analysis of the training residuals and derives the forecast error variance to be used in building prediction intervals by applying a bootstrap technique. Chapter 7 gives the numerical results of the comparison between the developed model and its counterpart, the ARIMA model. Chapter 8 draws some conclusions from this research and provides recommendations for future research directions.

Chapter 2

Background and Literature Review

To monitor the forecasting process in an intelligent time series system and assess uncertainty associated with the forecasts, we will first generate forecasts using neuro-fuzzy mechanism, then monitor the process and construct the prediction intervals. We review several main topics in this Chapter, which will be referred to frequently in our research. These topics are:

- Fuzzy Logic
- Neural Networks
- Neuro-Fuzzy systems
- Tracking Signals
- Variance Estimation

2.1 Fuzzy Logic

Fuzzy logic was developed for representing uncertain and imprecise knowledge. It provides an approximate but effective means of describing the behavior of systems that are too complex, ill-defined, or not easily analyzed mathematically. “Inferencing” is the key to any fuzzy system. A typical fuzzy inference system consists of membership functions, a rule base and an inference procedure.

2.1.1 Fuzzy Sets

A classical set is a set with crisp boundary. In contrast, a fuzzy set is a set with smooth boundary. Let X be a space of objects and x be a generic element of X . In classical set theory, a subset A of the universe X is defined by its binary (0 or 1) characteristic function $\mu_A(x): X \rightarrow [0,1]$ such that $\mu_A(x) = 1$ if $x \in A$ and $\mu_A(x) = 0$ if $x \notin A$. Unlike conventional sets, the characteristic function of a fuzzy set is allowed to have values between 0 and 1, where A is called a fuzzy set and μ_A is called the membership function of A (Zadeh, 1965). Fuzzy sets can either be discrete or continuous. The construction of a fuzzy set depends on two things: the identification of a suitable universe and the specification of an appropriate membership function.

Figure 2.1 Cores, supports, boundaries, crossover points of membership function

A fuzzy set is uniquely characterized by its membership function. The basic features about membership functions are graphically shown in Figure 2.1. A description of the terms used is included in Appendix I.

Classical sets have three basic operations: union, intersection, and complement. Likewise, fuzzy sets have similar operations, which were initially defined in Zadeh's seminal paper (Zadeh, 1965). Suppose A and B are fuzzy sets of the universe X with membership functions μ_A and μ_B . The union $A \cup B$, intersection $A \cap B$ and complement $\bar{A}$ are also fuzzy sets whose membership functions are related to those of A and B. They are defined as:

$$\text{Union} \quad \mu_{A \cup B}(x) = \max(\mu_A(x), \mu_B(x)) = \mu_A(x) \vee \mu_B(x) \quad 2.1$$

$$\text{Intersection} \quad \mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x)) = \mu_A(x) \wedge \mu_B(x) \quad 2.2$$

$$\text{Complement} \quad \mu_{\bar{A}}(x) = 1 - \mu_A(x) \quad 2.3$$

Also, other operators have been introduced to extend the classical set-theoretic operations (Fodor and Roubens, 1994). These operators are referred to as T-norm (Dubois and Prade, 1980) for the intersection, T-conorm or S-norm (Dubois and Prade, 1980) for the union, and negation for the complement. The fuzzy extension of the classical modus ponens principle allows for the construction of fuzzy inference systems (Dubois and Prade, 1991).

2.1.2 Membership Functions

A fuzzy set can be defined by enumerating membership values of the elements in the set if it is discrete or by defining the membership function mathematically if it is

continuous. Although there exist numerous types of membership functions, the most commonly used in practice are triangles, trapezoids, Gaussian, and bell curves. Detailed descriptions of these membership functions can be found in Appendix II.

Triangular MFs and trapezoidal MFs have been widely used due to their simplicity and computational efficiency (Yen and Langari, 1999). However, since they are composed of straight line segments, these MFs are not smooth at the corner points specified by the parameters. In some cases, the derivatives of membership functions with respect to their inputs and parameters are very important for fine-tuning a fuzzy inference system to achieve a desired input/output mapping, thus a smooth and nonlinear membership function is needed (Fiordaliso, 1998; Palit and Popovic, 1999). On the other hand, most membership functions are determined subjectively; the human-determined membership functions, however, may not be precise enough in certain applications. Therefore, it is always advisable to apply optimization techniques to fine-tune parameterized membership functions for better performance. Because of these reasons, Gaussian and bell curve MFs are becoming more popular for specifying fuzzy sets.

2.1.3 Fuzzy If-Then Rules

Fuzzy if-then rules are a knowledge representation scheme for capturing knowledge (typically human knowledge) that is imprecise and inexact by nature. Generally, this is achieved by using linguistic variables (Zadeh, 1971; Zadeh, 1975) to describe elastic conditions (i.e., conditions that can be satisfied to a degree) in the “if part” of fuzzy rules and to perform inference under partial matching.

A fuzzy if-then rule takes the form:

$$\text{IF } x \text{ is } A_k \text{ THEN } y \text{ is } B_k(x)$$

where A_k and B_k are linguistic values defined by fuzzy sets on universes X and Y respectively. Often, the “if part” is called antecedent or premise, while the “then part” is called consequence or conclusion. The fuzzy sets in a rule’s antecedent define a fuzzy region of the input space covered by the rule (i.e., the input situations that fit the rule’s condition completely or partially), whereas the fuzzy sets in a rule’s consequent describe the vagueness of the rule’s conclusion.

The consequent of fuzzy rules can be classified into three categories (Yen and Langari, 1999):

1. Crisp Consequent: IF... THEN $y = a$.

where a is a nonfuzzy numeric value or a symbolic value.

2. Fuzzy Consequent: IF... THEN y is A .

where A is a fuzzy set.

3. Functional Consequent: IF x_1 is A_1 , x_2 is A_2 , ... and x_n is A_n THEN $y = a_0 +$

$$\sum_{i=1}^n a_i * x_i .$$

where $a_0, a_1 \dots a_n$ are constants.

2.1.4 Fuzzy Reasoning

Fuzzy reasoning, also called approximate reasoning, is an inference procedure that derives conclusions from a set of fuzzy if-then rules and known facts.

Definition Fuzzy reasoning (Approximate reasoning)

Let A , A' , and B be fuzzy sets of X , X , and Y , respectively. Assume that the fuzzy implication $A \rightarrow B$ is expressed as a fuzzy relation R between X and Y , then the fuzzy set B' induced by “ x is A' ” and the fuzzy rule “if x is A then y is B ” is defined by

$$\mu_{B'}(y) = \max_x \min[\mu_{A'}(x), \mu_R(x, y)] \quad 2.4$$

The process of fuzzy reasoning can be divided into four steps (Jang, Sun and Mizutani, 1997):

1. Degrees of compatibility: Compare the known facts with the antecedents of fuzzy rules to find the degrees of compatibility with respect to each antecedent MF.
2. Firing strength: Combine degrees of compatibility with respect to antecedent MFs in a rule using fuzzy AND or OR operators to form a firing strength that indicates the degree to which the antecedent part of the rule is satisfied.
3. Qualified (induced) consequent MFs: Apply the firing strength to the consequent MF of a rule to generate a qualified consequent MF.
4. Overall output MF: Aggregate all the qualified consequent MFs to obtain an overall output MF.

2.1.5 Fuzzy Inference Systems

Fuzzy inference systems are the most important modeling tool based on fuzzy set theory, whereas fuzzy rules and fuzzy reasoning are the backbone of fuzzy inference

systems. The basic structure of a fuzzy inference system consists of three conceptual components:

- a rule base, which contains a selection of fuzzy rules;
- a database, which defines the membership functions used in the fuzzy rules;
- a reasoning mechanism, which performs the inference procedure upon the rules and given facts to derive a reasonable output or conclusion.

The inputs of fuzzy inference system can either be fuzzy sets or crisp values (which are viewed as fuzzy singletons). If the system produces fuzzy sets as output, while a crisp output is needed, then a method of defuzzification is required to extract a crisp value that best represents the fuzzy set. In general, there are several different methods for defuzzifying a fuzzy set. They are: centroid of area, bisector of area, mean of maximum, smallest of maximum, largest of maximum, and height methods (Jang, Sun and Mizutani, 1997; Yen and Langari, 1999). Other more flexible defuzzification methods can be found in Pfluger, Yen and Langari (1992), Runkler and Glesner (1994), and Runkler (1997). With crisp inputs and outputs, a fuzzy inference system implements a nonlinear mapping from its input space to output space.

Depending on the types of fuzzy reasoning and fuzzy if-then rules employed, most fuzzy inference systems can be classified into three types:

- Mamdani fuzzy model

Mamdani fuzzy model was proposed to control a steam engine and boiler combination by a set of linguistic control rules (Mamdani and Assillian, 1975). The fuzzy rule in this model is in the form of:

IF x_1 is A_{i1} ...and x_n is A_{in} THEN y is C_i .

where x_j ($j=1, 2 \dots n$) are the input variables, y is the output variable, A_{ij} and C_i are fuzzy sets for x_j and y respectively.

- Sugeno fuzzy model

Sugeno fuzzy model (also known as TSK model) was proposed to develop a systematic approach to generating fuzzy rules from a given input-output data set (Takagi and Sugeno, 1985; Sugeno and Kang, 1988). For a two-input system, the fuzzy rule in this model is in the form of:

IF x_1 is A and x_2 is B THEN $y = f(x_1, x_2)$.

where A and B are fuzzy sets in the antecedent, $y = f(x_1, x_2)$ is a crisp function in the consequent. Usually, $f(x_1, x_2)$ is a polynomial function of the input variables x_1 and x_2 , but it can be any function as long as it can appropriately describe the output of the model within the fuzzy region specified by the antecedent of the rule. When $f(x_1, x_2)$ is a first-order polynomial function, the resulting fuzzy inference system is called a first-order Sugeno fuzzy model. When $f(x_1, x_2)$ is a constant, the system is referred as a zero-order Sugeno fuzzy model.

Without the time consuming and mathematically intractable defuzzification operation, the Sugeno model is by far the most popular candidate for sample-based fuzzy modeling.

- Tsukamoto fuzzy model

Tsukamoto fuzzy model (Tsukamoto, 1979) was proposed as another approach to fuzzy reasoning method. The fuzzy rule in this model is in the form of:

IF x is A_i THEN y is C_i .

where x is the input variable, y is the output variable, A_i is a fuzzy set with a monotonical MF, C_i is a crisp value induced by rule's firing strength.

2.1.6 Fuzzy Modeling

In general, the process to construct a fuzzy inference system is called fuzzy modeling. Theoretically, fuzzy modeling can be accomplished in two stages. The first stage is identification of surface structure, which includes the following tasks (Jang, Sun and Mizutani, 1997):

1. Select relevant input and output variables.
2. Choose a specific type of fuzzy inference system.
3. Determine the number of linguistic terms associated with each input and output variable. For Sugeno model, determine the order of consequent equations.
4. Design a collection of fuzzy rules.

The second stage is the identification of deep structure, which means:

1. Choose appropriate parameters of membership functions.
2. Refine the parameters of the MFs using regression and optimization techniques.

2.2 Neural Networks

“Learning” is the central strength point of artificial neural networks. Accordingly, neural network models can be classified as supervised learning vs. unsupervised learning networks. For a supervised network, a teacher is required to specify the desired output; while for an unsupervised network, internal models are constructed to capture regularities in input signals (Vemuri, 1988).

2.2.1 Supervised Learning

The goal of supervised learning is to shape the input-output mappings of the network based on a given training data set. As the term suggests, first, the desired input-output data sets must be known; then the resulting networks have adjustable parameters that are updated by a supervised learning rule. The adjustable parameters are often referred to as weights.

Backpropagation (BP), also known as back error propagation or the generalized delta rule (GDR), is an effective supervised learning method for training multilayer perceptrons (Rumelhart, Hinton and Williams, 1986). The process involves two steps, a forward propagating step and a backward propagating step.

In the forward pass, the training input data is presented to the input layer. The data propagates on through the hidden layers, until it reaches the output layer, where it is displayed as the output pattern. In the backward pass, the error term is calculated and propagated back to change the assigned weights of the inputs. The magnitude of the error value indicates how large an adjustment needs to be made and the sign of the error value

gives the direction of the change. Figure 2.2 shows a node j of a backpropagation multilayer perceptrons (MLP). Usually, the node is a composite of the weighted sum and a differentiable nonlinear activation function, which is often assumed to be sigmoid logistic function as

$$f(x) = \frac{1}{1 + e^{-x}} \quad 2.5$$

Figure 2.3 shows a multilayer backpropagation with n inputs and k outputs. Note that there may be several hidden layers with different number of neurons between input layer and output layer.

Figure 2.2 Node j of a backpropagation MLP

Figure 2.3 A backpropagation multilayer perceptron

Backpropagation training algorithm is an iterative gradient algorithm designed to minimize the mean square error between the actual output of a multilayer feed-forward perceptron and the desired output (Lippmann, 1987). The error is defined as

$$E = \sum_p E_p \quad 2.6$$

where E_p is the error for one input pattern, described as:

$$E_p = \frac{1}{2} \sum_j (T_j - O_j)^2 \quad 2.7$$

and T_j is the desired output (target) while O_j is the actual output.

From the backpropagation learning rule of Rumelhart et al. (1986):

$$W(t+1) = W(t) - \eta \frac{\partial E}{\partial W} \quad 2.8$$

The weight change corresponding to the gradient of the error is

$$\Delta W_{ji} = \eta \delta_j X_i \quad 2.9$$

where η is a learning rate that affects the convergence speed and stability of the weights during learning; W_{ji} is the weight associated with the connection from node i to j ; δ_j is an error term associated with node j ; and X_i is the input to node j .

A recursive method is used to adjust weights starting at the output nodes and working back to the first hidden layer by

$$W_{ji}(t+1) = W_{ji}(t) + \eta \delta_j X_i(t) \quad 2.10$$

2.2.2 Unsupervised Learning

Unsupervised learning is learning without supervision, i.e., no information available regarding the desired outputs; the network updates weights only on the basis of the input patterns. Since the learning system detects or categorizes persistent features without any feedback from environment, it is frequently employed for data clustering, feature extraction, and similarity detection.

Kohonen self-organizing maps (Kohonen, 1982; 1984), also known as Kohonen self-organizing feature maps, are one of three common unsupervised learning paradigms. Self-organizing implies the ability to acquire knowledge through a trial and error learning process involving organizing and reorganizing in response to external stimuli. Namely, the networks impose a neighborhood constraint on the output units, such that a certain topological property in the input data is reflected in the output unit's weights.

During the learning process, the weights of winning units and the weights in a neighborhood around the winning units are adjusted based on the similarity measure or dissimilarity measure. If the similarity measure of inner product is selected, the winning unit is considered to be the one with the largest activation level; if the dissimilarity measure of Euclidean distance is selected, the winning unit is considered to be the one with the smallest activation level. Initially, all of the nodes in the network are included in the neighborhood of the winner; however, as learning proceeds, the size of the neighborhood is decreased linearly until it includes only the winner itself. Figure 2.4 shows the neighborhoods reducing around a winning unit with each iteration.

Specifically, if the Euclidean distance is chosen as the dissimilarity measure, the winning unit c satisfies the following equation:

Figure 2.4 Reducing neighborhoods around node x

$$\|X - W_c\| = \min_i \|X - W_i\| \quad 2.11$$

where c refers to the winning unit, while i refers to all output units.

Then the weights of the winner and its neighborhood units are updated by

$$\Delta W_i = \eta(X - W_i) \quad 2.12$$

where η is small positive learning rate ($0 < \eta < 1$), i belongs to a set of indices corresponding to a neighborhood around the winner c . Usually, Gaussian function $\varphi_c(i)$ is used as the neighborhood function when defining the neighborhood of a winning unit c :

$$\varphi_c(i) = \exp\left[\frac{-\|p_i - p_c\|^2}{2\sigma^2}\right] \quad 2.13$$

where p_i and p_c are the positions of the output units i and c , respectively, σ reflects the scope of the neighborhood, and the weights updating formula can be rewritten as:

$$\Delta W_i = \eta \varphi_c(i) (X - W_i) \quad 2.14$$

2.3 Neuro-Fuzzy Systems

Neural networks and fuzzy logic are two complementary technologies. This is because neural networks, which are viewed as a “Black Box” model, have the learning ability which can learn knowledge using training examples, while fuzzy inference systems can deduce knowledge from the given fuzzy rules. Therefore, the combination of these two outperforms either neural network or fuzzy logic method used exclusively and becomes an ideal partner in control area, medicine, time series forecasting, and decision making (Nie and Linkens, 1995; Jain and Martin, 1999; Deboeck, 1994).

A fuzzy inference system can utilize human expertise by storing its essential components in rule base and database, and perform fuzzy reasoning to infer the overall output value. The derivation of fuzzy rules corresponding to membership functions depends heavily on the *a priori* knowledge of the system under consideration. However, there is no systematic way to transform experiences to the knowledge base of a fuzzy inference system and no adaptability or learning algorithms to tune the membership functions so as to minimize the discrepancy between model output and desired output of the system. On the other hand, a neural network employs a given training data set and a learning procedure to evolve a set of parameters (i.e. weights) such that the required functional behavior is achieved, whereas it has difficulty to develop an insight about the meaning associated with each neuron and each weight. Therefore, these two approaches can be integrated to generate hybrid models that can take advantage of the strong points of both.

Different structures were found in literature that demonstrated the advantages of these two methods when they are combined. These paradigms might be classified into three categories as follows:

1. Neural Fuzzy System: the use of neural networks as a tool in fuzzy modeling.
2. Fuzzy Neural System: fuzzification of traditional neural network model.
3. Neuro-Fuzzy Hybrid System: integrating neural networks and fuzzy logic into a hybrid system.

2.3.1 General Neuro-Fuzzy Architecture

The general neuro-fuzzy hybrid system is basically a multi-layered fuzzy rule-based neural network which integrates the basic elements and functions of a traditional fuzzy logic inference into a neural network structure (Li, Ang and Gray, 1999). With the input and output membership functions, the system indicates that neural nets could have more crisp and meaningful inputs and thus improve the overall output quality when compared with the standard neural network, where the output values are ranging between 0 and 1 by nonlinear transform functions.

Figure 2.5 General neuro-fuzzy architecture

As shown in Figure 2.5, the general neuro-fuzzy structure is a five-layer fuzzy rule-based neural network consisting of nodes in each layer. Input variables are assigned

to layer 1, from where the input values transmit to layer 2 directly. Layer 2 works as a fuzzifier, where the outputs represent the membership grade of the corresponding inputs. The nodes in layer 3 performs fuzzy AND operations on their inputs, and the output indicates whether the rule “fires or not”, which consequently determines the activation level of layer 4, the output membership functions layer. Finally, layer 5 performs defuzzification of the overall output.

However, the calculation for defuzzification operation is a time-consuming and intractable task. Further, most of the defuzzification operations being used are based on experimental results, hence are not easily subject to rigorous mathematical analysis. This leads to the consideration of systems that do not need defuzzification operations.

2.3.2 ANFIS Architecture

ANFIS, which stands for Adaptive Neuro Fuzzy Inference Systems, is an efficient and transparent neuro-fuzzy paradigm first proposed by Jang (1992; 1993; 1996). As aforementioned, Sugeno fuzzy model is a suitable choice for the requirement of not using defuzzification operation and is widely accepted in sample-based fuzzy modeling. Assume that the fuzzy inference system under consideration has two inputs x_1 and x_2 and one output y . For a first-order Sugeno model, a common rule set with two fuzzy if-then rules is the following:

Rule 1: If x_1 is A_1 and x_2 is B_1 , then $f_1 = a_1 x_1 + b_1 x_2 + c_1$.

Rule 2: If x_1 is A_2 and x_2 is B_2 , then $f_2 = a_2 x_1 + b_2 x_2 + c_2$.

where A_1, B_1, A_2, B_2 are fuzzy sets, a_i, b_i and c_i ($i = 1, 2$) are the coefficients of the first-order polynomial linear functions. Also, it is possible to assign a different weight to each rule based on the structure of the system. Figure 2.6 shows the structure of a two-input first-order Sugeno fuzzy model with two rules, where weights w_1 and w_2 are assigned to rules 1 and 2 respectively.

Figure 2.6 A two-input first-order Sugeno fuzzy model

Figure 2.7 shows the equivalent ANFIS architecture, where nodes of the same layer have similar functions. Note that $O_{j,i}$ is the output of the i th node in layer j .

Layer 1: Each node output in this layer is membership grade of a fuzzy set corresponding to each input. The membership function for this fuzzy set can be any

appropriate parameterized membership function, such as generalized bell function or Gaussian function. The parameters of the membership function are the premise parameters of the system.

$$\begin{aligned} O_{1,i} &= \mu_{A_i}(x_1) & i = 1, 2 \text{ or} \\ O_{1,i} &= \mu_{B_{i-2}}(x_2) & i = 3, 4 \end{aligned} \quad 2.15$$

where x_1 and x_2 is the input to node i ($i = 1, 2$ for x_1 and $i = 3, 4$ for x_2).

Figure 2.7 Equivalent ANFIS architecture

Layer 2: Each node output in this layer represents the firing strength of a rule, which performs fuzzy AND operation. The output could be the product of all coming signals or the minimum value of all coming signals or other T-norm operation.

$$O_{2,i} = W_i = \mu_{A_i}(x_1) \mu_{B_i}(x_2) \quad i = 1, 2 \quad 2.16$$

Layer 3: Each node output in this layer is the normalized value of layer 2, i.e., the normalized firing strengths.

$$O_{3,i} = \bar{W}_i = \frac{W_i}{W_1 + W_2} \quad i = 1, 2 \quad 2.17$$

Layer 4: Each node output in this layer is the normalized value of each fuzzy rule. The coefficients of the polynomial linear equation of each rule are the consequent parameters in the system.

$$O_{4,i} = \bar{W}_i f_i = \bar{W}_i (a_i x_1 + b_i x_2 + c_i) \quad i = 1, 2 \quad 2.18$$

Layer 5: The node output in this layer is the overall output of the system, which is the summation of all coming signals.

$$Y = \sum_{i=1}^2 \bar{W}_i f_i = \frac{\sum_{i=1}^2 W_i f_i}{\sum_{i=1}^2 W_i} \quad 2.19$$

2.3.3 Hybrid Learning

To identify the parameters in the nonlinear neuro-fuzzy model, the gradient descent method in conjunction with error backpropagation process could be used for neural network learning. However, this optimization method usually takes a long time to converge. On the other hand, least squares estimation is a powerful and well-developed tool that is widely employed in areas such as adaptive control, signal processing and statistics. It has been proven that least squares method is essential and indispensable for constructing linear mathematical models, and its fundamental concepts can also be

extended to nonlinear models. In fact, the gradient descent method (GDM) and least-square estimator (LSE) provide the most basic and important mathematical foundation for solving neuro-fuzzy modeling problems in soft computing techniques; their combination leads to a hybrid learning rule for fast identification of parameters (Jang, Sun and Mizutani, 1997).

The theory behind hybrid learning is, the set of model parameters, which denoted as S , could be divided into two subsets S_1 and S_2 where only the elements in S_2 are linear parameters. For given fixed values in S_1 , the parameters in S_2 can be obtained by least squares method with the objective function of minimizing the sum of squared errors, and therefore are guaranteed to be the global optimum point. When the parameter values in S_2 are fixed, the parameters in S_1 will be updated by gradient descent method.

In summary, by employing the sample-based Sugeno fuzzy model as the inferencing system, combining the least squares estimator and gradient descent method into the hybrid learning rule, ANFIS structure is suitable for nonlinear neuro-fuzzy modeling, which is also appropriate for nonlinear domain of time series forecasting.

2.4 Comparison of Three Intelligent Forecasting Methods

The application of different soft computing paradigms into different forecasting problems leads to some intelligent forecasting schemes including fuzzy forecasting, neural network forecasting and neuro-fuzzy forecasting. These schemes have their own features and different implementations; but they also share things in common, this is especially true when they are applied to time series forecasting.

2.4.1 Fuzzy Forecasting

The general framework of fuzzy forecasting system is basically a fuzzy rule-based system. Usually, a k-step ahead fuzzy time series forecasting can be described as:

$$x(t+k) = F\{x(t), x(t-1), \dots, x(t-m)\} \quad 2.20$$

where F represents a mapping from a set of time series $\{x(t), x(t-1), \dots, x(t-m)\}$ onto k-step data $x(t+k)$. This mapping can be represented by fuzzy rules as:

Rule: IF $x(t)$ is $T_{x(t)}$ and $x(t-1)$ is $T_{x(t-1)}$... and $x(t-m)$ is $T_{x(t-m)}$, THEN $x(t+k)$ is $T_{x(t+k)}$.

The crisp input-output values need to be fuzzified when employing this rule.

The difficulty with the fuzzy forecasting system is to generate appropriate fuzzy rules by mapping of inputs to the respective output. Palit and Popovic (1999) proposed an algorithm to build a combined fuzzy rule base by using a lookup table and applied it in forecasting of a 6-step ahead Mackey-Glass chaotic time series with 4 inputs. They employed Gaussian membership functions and specified 500 fuzzy rules in the system. Four different systems with 17, 27, 37, 51 membership functions were tested, the results showed the system with 51 membership functions gave the best performance, where $SSE = 0.0164$, $RMSE = 0.0081$ and $MAE = 0.0062$. Kim and Lee (1999) also presented a method to predict the Mackey-Glass chaotic time series by modifying the rule set as the difference of consecutive values in a time series. They used 700 data points for training and 300 data points for testing. For a 9-input system with 29 membership functions, the best RMSE of the system was 0.0156. The fuzzy inference systems of the above two are both Mamdani fuzzy model.

It is observed that Sugeno fuzzy model has been widely used by many researchers. Also, parameterized membership function, i.e., the nonlinear membership grade, is more preferable due to the fine-tuning ability of the parameters. Fiordaliso (1998) presented a nonlinear forecasts combination method based on Sugeno fuzzy systems. He showed the parametric tuning and the structural tuning of the system and also indicated that using Gaussian membership functions ensured the system to be complete, i.e. for every input, at least one of the fuzzy rules is activated, while using triangular or trapezoidal membership functions could not always guarantee the completeness. Fiordaliso compared the system with the traditional models. One test example was a 200-observation time series taken from Anderson's *Time Series Analysis and Forecasting*, published in 1976. The results showed that the proposed system gave smaller forecast errors and outperformed its counterparts with $MSE = 13.19$, $MAE = 3.01$ and $UTHEIL = 0.76$. Palit and Popovic also showed that using Gaussian membership functions instead of triangular membership functions could significantly reduce the manual efforts in handling large data sets and the number of membership functions.

2.4.2 Neural Network Forecasting

In a review of forecasting with artificial neural networks by Zhang, Patuwo and Hu (1998), it has been stated that artificial neural networks (ANNs) have been investigated as an effective tool for time series forecasting. The class of models exclusively used in this area is the feed-forward multilayer network trained by backpropagation learning rule. For a univariate time-series forecasting problem, the

inputs of the network are the past, lagged observations and the output is the predicted value. A single output multilayer perceptrons (MLP) performs the following mapping from the inputs to the output:

$$x(t+k) = F\{x(t), x(t-1), \dots, x(t-m)\} \quad 2.21$$

where $x(t)$ is the observation at time t , m is the dimension of the input vector or the number of past observations used to predict the future, and F in general is a nonlinear function determined by the MLP structure and the data. The general structure of neural network forecasting is shown in Figure 2.8.

Figure 2.8 Neural network forecasting

The most difficult problem in neural network forecasting is to develop a network of appropriate size for capturing the underlying patterns in the training data set (Zhang,

Patuwo and Hu, 2001). The size of an MLP largely depends on the number of input nodes and the number of hidden nodes. Input nodes are used to discover the underlying patterns and /or autocorrelation structures in a time series, while hidden nodes are used to capture the nonlinear structures in a time series.

Balkin and Ord (2000) presented a method to develop an automatic procedure for selecting the architecture of an artificial neural network for forecasting purpose. A numerical example from the M-3 Competition data set, which is a quarterly microeconomic series with 37 observations, has been applied to the method. The result showed that the preferred model under their procedure gave a better fit for the series.

Tkacz (2001) used a multivariate, single hidden layer neural network model, with the number of hidden units allowed to vary from one to four to forecast the short-term (one-quarter) and long-term (four-quarter) real growth rate of GDP in Canada. The model parameters were initially estimated using data from 1968q1 to 1988q4, data between 1989q1 and 1999q2 were used to generate forecasts and update the model. For the one-quarter model, the neural network model that contained three financial variables performed better than other time series models and linear models based on the MSE criterion; but these variables did not make much difference in the short-run for statistical models. For the four-quarter model, the neural network forecast error of the yearly output growth was about 0.25% lower each quarter than the linear models with $MSE = 3.19$ and $MAE = 1.46$.

Dougherty and Cobbett (1997) proposed a technique for stepwise reduction in the network size when using BP neural network to make short-term forecasts of inter-urban

traffic flow. They demonstrated their proposed technique using a test example with 820 data points by root mean square error proportional indicator (RMSEP) statistical measure. They concluded that their method exhibited some promise but did not outperform naïve predictors for this problem.

2.4.3 Neuro-Fuzzy Forecasting

The application of neuro-fuzzy technique to forecasting area is a recent and fast growing area. Different paradigms could be constructed to combine the fuzzy inferencing and neural network learning. One typical example of neuro-fuzzy forecasting system is based on the architecture described in section 2.3.1. Li, Ang and Gray (1999) developed an intelligent business forecaster using the structure with three inputs and gave two numerical examples. They first applied the system to forecast the rate of exchange between US dollar and Singapore dollar. A total of 69 data sets from statistical reports were available, of which 50 samples (from 01/1990 to 02/1994) were used for training and the rest for testing. The error measures for the 19 sets of test data were found to be: $MSE = 0.00046$, $UTHEIL = 1.4536$ and $MAPE = 1.184$. They also applied the system to forecast the electricity consumption in Singapore. A total of 37 data sets were used, of which the first 25 data sets were used for training and the rest for testing. The error measures were found to be: $MSE = 2.6$, $UTHEIL = 1.0988$ and $MAPE = 8.5792$.

Jang (1997) demonstrated the forecasting of 6-step Mackey-Glass chaotic time series with 4 inputs using ANFIS architecture. 1000 input-output data pairs were extracted, of which the first 500 pairs were used for training and the rest were used for

testing. The number of membership functions assigned to each input was set to two. They used the root mean squared error (RMSE) as the error measure. The results showed that after about 500 epochs, the RMSE for training was 0.0016 and for testing was 0.0015, which are much better than the results of other approaches. They also compared their results with those obtained from the autoregressive (AR) model with 45 parameters and concluded that nonlinear ANFIS outperformed the linear AR model but took a longer time to generate the forecasts. Comparisons with other models were also given in the conclusion, which showed the advantages of the ANFIS structure.

2.5 Tracking Signals

Monitoring is applicable to a wide range of situations. Approaches to monitoring can be classified into judgmental procedures and quantitative procedures. Tracking signals test is a quantitative procedure which is used to automatically determine if a pattern or relationship has changed or not and more importantly, when it has changed; furthermore, it leads to the action to bring the process back in control. Gardner (1983) classified monitoring procedures into four categories: (1) simple CUSUM (CUMulative SUM) tracking signals, (2) smoothed error tracking signals, (3) backward CUSUM control system, and (4) autocorrelation tracking signals. The first two are used most often.

The tracking signals in general are ratios. The numerator is a weighted or unweighted sum of forecast errors, while the denominator is usually the mean absolute deviation (MAD), which is defined as:

$$MAD = \frac{1}{n} \sum_{i=1}^n |e_i| \quad 2.22$$

The numerator has an expected value of zero when the forecast is in control, and would move away from zero if the forecast goes out of control. The more the forecasting errors vary from zero in some systematic way, the greater the chances that the established pattern or relationship being monitored has changed.

2.5.1 Simple CUSUM Tracking Signals

Simple CUSUM tracking signals was first introduced by Brown (1959). This approach uses an unweighted sum of forecast errors as a measurement. It compares the cumulative sum (CUSUM) of the errors at the end of each time period with a smoothed value of the mean absolute deviation (MAD). If the ratio CUSUM/MAD exceeds a prespecified limit, the forecasting approach is reexamined to see whether the pattern has changed and thus whether some action needs to be taken.

2.5.2 Smoothed Error Tracking Signals

Trig (1964) pointed out two disadvantages of the Brown's method and introduced the smoothed error tracking signals test. The main problem with Brown's method is the required long memory. Rather than taking the ratio of the actual CUSUM to the smoothed MAD, Trig's method uses an exponentially weighted smoothed value of the forecast error divided by a smoothed value of the MAD. If the ratio exceeds a

prespecified limit, it signals that probably a change in pattern or relationship has occurred.

The smoothed error tracking signal has been widely used in practice. It consists of two smoothing equations and the tracking signal T . It requires the continued updating and storage of two values, E_t and M_t . The updating and tracking signal equations are as follows:

$$E_t = \alpha e_t + (1 - \alpha)E_{t-1} \quad 2.23$$

$$M_t = \alpha |e_t| + (1 - \alpha)M_{t-1} \quad 2.24$$

$$T_t = \left| \frac{E_t}{M_t} \right| \quad 2.25$$

where $e_t = X_t - F_t$, and α is a smoothing parameter whose value is between 0 and 1.

Once a forecasting system is in operation, e_t is known, E_t and M_t will be computed, thus the tracking signal ratio T_t can be obtained. If the value of T_t is close to 0, i.e., M_t is much larger than E_t , it implies that the errors are random. When some systematic change occurs, the forecast errors will be consistently positive or negative, increasing the value of E_t , which will cause the ratio T_t to increase. If a certain limit is reached, it implies that the errors have ceased to be random; therefore the forecasting procedure needs to be reexamined.

2.5.3 Backward CUSUM Control Systems

The backward CUSUM control systems approach was suggested by Harrison and Davies (1963). It is based on the notion that if a past change in a time series could be

guessed, then the sum of all errors since the change would be the best tracking signal available. Since there is no way of knowing the number of past periods to sum in advance, it is necessary to store the information of all possible CUSUMs. The first one is the last error, the second is the sum of the last two errors, and so on. However, the number of such CUSUMs could quickly get out of hand.

2.5.4 Autocorrelation Tracking Signals

For models other than exponential smoothing, the first-order autocorrelation of adjacent errors (referred to as a first-order regression on successive errors) can be used to develop a tracking signal. The tracking signal is found as:

$$Cov_t = e_t e_{t-1} + (1 - \alpha)Cov_{t-1} \quad 2.26$$

$$MSE_t = e_{t-1}^2 + (1 - \alpha)MSE_{t-1} \quad 2.27$$

$$T_t = \left| \frac{Cov_t}{MSE_t} \right| \quad 2.28$$

where α is smoothing parameter for the error, Cov_t is covariance at time t, MSE_t is mean squared error at time t, and T_t is the tracking signal based on the weighted autocorrelation and the mean squared error.

2.6 Variance Estimation

A typical problem in applied statistics involves the estimation of an unknown parameter, wherein two main questions are concerned, namely, what the estimator should

be and how accurate it might be. Variance estimation is used for answering the second question. The estimation of variance, or, the standard error, is relatively easy for linear models; however, for nonlinear models, it is usually intractable.

2.6.1 Overview

There are several general methodologies for estimating the standard error for nonlinear models; these methods include delta method, jackknife and bootstrap techniques.

The delta method is a technique for deriving a variance, and is applicable to an estimator that is a function of statistics. The idea behind the delta method is the local linearization of the nonlinear function by using the first-order Taylor series expansion. However, this method becomes less efficient and mathematically difficult when it comes to a multivariate version, where the matrix notations are needed; or in the case where the nonlinear function is not available or very complicated. Unfortunately, both these two cases are present in neuro-fuzzy model.

Jackknife is another technique for estimating standard error of an estimate. The jackknife method predates the bootstrap and bears similarities to it. The difference between them is that every time a jackknife sample is created, one observation is eliminated from the data set. The jackknife often works well if the statistic is “smooth”, i.e., the statistic under study does not change drastically upon small changes in the data. When the statistic lacks smoothness, the jackknife estimation procedures may not be useful.

The bootstrap technique is a widely accepted approach for estimating the standard error and the determination of confidence intervals. It is a widely applicable computer-intensive statistical tool that may yield estimates which in other ways would be difficult to obtain (Wehrens, Putter, and Buydens, 2000).

2.6.2 The Bootstrap: General Ideas

Literally, bootstrap is a loop of leather, cloth, or synthetic material that is sewn at the side or the top rear of a boot to help in pulling the boot on. More interestingly, the name “bootstrap” is also a reference to the famous story of Baron Von Munchhausen, who pulled himself up by his bootstraps out of swamp (Wehrens, Putter, and Buydens, 2000). For statistician, bootstrap is a procedure which represents a class of methods that resample from the original data set with replacement; in a sense, it works as allowing the sample to act as if it were the population.

The bootstrap was first introduced by Efron in 1979 as a computer-based method for estimating the standard error for an estimator. It requires no theoretical calculations, and is available no matter how mathematically complicated the estimator may be (Efron & Tibshirani, 1993).

Efron’s bootstrap is defined as follows: Given a sample of n independent identically distributed random vectors $X (x_1, x_2, \dots, x_n)$ and a real-valued estimator $\hat{\theta} (x_1, x_2, \dots, x_n)$ of the distribution parameter θ , a procedure (the bootstrap) to assess the accuracy of $\hat{\theta}$ is defined in terms of the empirical distribution function $\hat{F}$. This

empirical distribution function assigns probability mass $1/n$ to each of the observed values x_i , $i=1, 2, \dots, n$ (Chernick, 1999).

The bootstrap method depends on the notion of bootstrap samples, X^* . A bootstrap sample is defined as a random sample of size n drawn from the unknown underlying distribution $\hat{F}$, i.e.,

$$\hat{F} \rightarrow X^*(x_1^*, x_2^*, \dots, x_n^*) \quad 2.29$$

Let s denote the function to obtain the estimator $\hat{\theta}$ based on X , i.e., $\hat{\theta} = s(X)$. For each bootstrap sample X^* , there is a bootstrap replication of $\hat{\theta}$, where the value of $\hat{\theta}$ is obtained by using the bootstrap sample in place of the original sample and applying same s function to X^* .

$$\hat{\theta}^* = s(X^*) \quad 2.30$$

The idea remains the same when applied to neural network model, where it works by creating many pseudoreplicates (“bootstrap samples”) of the training set and then estimating model parameters on each bootstrap sample (Tibshirani, 1996).

Statistically, some aspect of a probability distribution F can be estimated by the corresponding aspect of its empirical distribution $\hat{F}$. This is called “plug-in principle.” The bootstrap technique is based on the application of the plug-in principle to bootstrap data sets X^* . The bootstrap estimate can be obtained from the bootstrap sampling based on the Monte Carlo approximation, which is, using random numbers to approximate some deterministic outcomes.

The procedure of bootstrapping for estimating standard errors for an estimator $\hat{\theta}$ is as follows:

1. Generate a bootstrap sample of size n (where n is the original sample size) from the empirical distribution $\hat{F}$.
2. Compute the bootstrap replication of $\hat{\theta}^*$ corresponding to each bootstrap sample based on the same function for estimating $\hat{\theta}$.
3. Repeat steps 1 & 2 B times, where B is the specified number of replications.
4. Estimate the standard errors of $\hat{\theta}$ by the standard deviation of the B replications.

By repeating steps 1 and 2 B times, a Monte Carlo approximation to the distribution of $\hat{\theta}^*$ is obtained. The standard deviation of this Monte Carlo distribution of $\hat{\theta}^*$ is the Monte Carlo approximation to the bootstrap estimate of the standard error for $\hat{\theta}$.

Efron's bootstrap is basically a nonparametric method. There also exists a parametric bootstrap version, where the underlying distribution F is estimated from the data by a parametric model, not from the empirical distribution. In practice, bootstrap is commonly used to approximate the probability distribution of a statistic where the analytical form of the distribution is unknown or in many cases where the analytical forms rest on assumptions about populations are rarely certainly true, therefore in most applications the nonparametric bootstrap is used.

With improved computing resources, bootstrapping has become a mainstream and a standard statistical technique (Chernick, 1999).

2.6.3 Interval Estimation

In general, a confidence interval for θ is more informative than a point estimate for θ alone. The construction of confidence interval is one of the areas where the bootstrap has achieved major success (Wehrens, Putter, and Buydens, 2000). Several bootstrap methods have been proposed to obtain intervals with approximately correct coverage probabilities.

One of the most often used methods is the basic bootstrap confidence intervals, wherein the confidence interval for the parameter θ is usually obtained from the estimator $\hat{\theta}$ by considering the probability distribution of $\hat{\theta} - \theta$. Let S_α denotes the α -percentile of the distribution of $\hat{\theta} - \theta$, a confidence interval for θ is based on the probability statement

$$P(S_{\alpha/2} \leq \hat{\theta} - \theta \leq S_{1-\alpha/2}) = 1 - \alpha \quad 2.31$$

This leads the interval estimate to be written as follows:

$$\hat{\theta} - S_{1-\alpha/2} \leq \theta \leq \hat{\theta} - S_{\alpha/2} \quad 2.32$$

It states that the coverage probability, i.e., the probability of containing the true unknown value of θ , is $1 - \alpha$.

The basic bootstrap confidence interval is based on the same rule that also underlies bootstrap standard error: the distribution of $\hat{\theta} - \theta$ is approximated by that of $\hat{\theta}^* - \hat{\theta}$. This means that the quantiles of the distribution of $\hat{\theta} - \theta$ are approximated by those of the bootstrap distribution of $\hat{\theta}^* - \hat{\theta}$. Thus, the percentiles of S_α of the

distribution of $\hat{\theta} - \theta$ that are used in the above confidence interval (2.32) can be replaced by the appropriate quantiles of the bootstrap approximation, leading to the following bootstrap confidence interval

$$\hat{\theta} - S_{1-\alpha/2}^* \leq \theta \leq \hat{\theta} - S_{\alpha/2}^* \quad 2.33$$

Another method for interval estimation is the percentile bootstrap confidence intervals, where the intervals are estimated directly from the bootstrap samples of $\hat{\theta}$. Let G^* be the distribution function of the bootstrap replications $\hat{\theta}^*$, the $1 - \alpha$ percentile interval is defined by the $\alpha/2$ and $1 - \alpha/2$ percentiles of G^* .

$$\hat{G}_{\alpha/2}^{-1} \leq \hat{\theta} \leq \hat{G}_{1-\alpha/2}^{-1} \quad 2.34$$

By definition, $\hat{G}_{\alpha/2}^{-1} = \hat{\theta}_{\alpha/2}^*$, the $1 - \alpha$ bootstrap percentile confidence interval can be written as:

$$\hat{\theta}_{\alpha/2}^* \leq \hat{\theta} \leq \hat{\theta}_{1-\alpha/2}^* \quad 2.35$$

Chapter 3

Overview of Methodology

Having stated the scope of this research and reviewed some background and several related topics, we now describe the methodologies used to achieve the specific objectives outlined in Chapter 1.

3.1 Generating forecasts

In order to use ANFIS structure to generate forecasts, the first step is to initialize membership functions for input variables. The Gaussian MFs with two parameters, namely, center and width, are used here for describing the inputs. Because of the unequally distributed data points, the membership functions will not be equally spaced. The process will begin with data clustering to group data around the respective center, wherein the centered value of the each membership function might be shifted as each input data point being presented. After the centered values have been obtained, the width of membership functions will be determined by computing the distance between the x_j and the center of its associated group.

The next step is to apply the above initialized membership functions to the ANFIS forecasting structure. First-order Sugeno fuzzy model will be used for inferencing, i.e., to represent a fuzzy mapping from the inputs to output; and the backpropagation algorithm will be used for learning, i.e., to fine-tune the membership functions based on the training

errors. More specifically, the linear parameters of Sugeno fuzzy model will be obtained by recursive least squares method in the forward pass; while the nonlinear parameters of membership functions will be updated through error propagation in the backward pass. The updating is based on the learning algorithm of Rumelhart et al (1986):

$$R(t+1) = R(t) - \eta \frac{\partial E}{\partial R} \quad 3.1$$

where R is the adjustable parameter in membership functions, E is the error function and η is the learning rate.

The fuzzy inferencing will be reinforced and the linear parameter will be reobtained each time after the nonlinear parameters have been updated. The training process, i.e., inferencing $\rightarrow$ learning $\rightarrow$ inferencing... will be iterated till the specified stopping criterion is reached. Once the model parameters have been identified, the system is ready for generating forecasts.

3.2 Monitoring forecasting process

Although neuro-fuzzy structure can learn and infer from the historical data information, the model is static once the training is over and therefore must be explicitly updated by adding more recent information into the training set or dropping some outdated examples and retraining the network in order to keep it up-to-date (Berry and Linoff, 1997). On the other hand, time series is a time-ordered sequence, indicating the subject under study is a dynamic system. The dynamic feature of the problem requires the knowledge base of fuzzy inferencing system and the learning environment of neural

network be updated as time period lengthens. To deal with this, the smoothed-error tracking signals test developed by Trig (1964) will be used to monitor the process to detect if there are some nonrandom changes occurred and more importantly, to determine when the system needs to be retrained.

Trig's tracking signal ratio is based on the smoothed error and the smoothed mean absolute deviation (MAD). It will work as follows: the fuzzy rule base is built from the historical data information through the training process; whereas in testing process, for each input data pair being fed to the network, the forecast will be generated. When the actual observation becomes available, the tracking signal ratio will be computed based on the forecast error and the specified smoothed parameters. Once the tracking signals detect that the process is off the specified limit, the fuzzy rules and the model parameters, i.e., the knowledge base, need to be updated. By incorporating the most recently available data, the system will be retrained, which is, to recompute the new consequent parameters by using least squares method in the forward pass and to readjust the nonlinear parameters in the backward pass; thus the model parameters will be updated and used for generating future forecasts.

3.3 Estimating forecast error variance

There is no doubt that forecast error is inherent in any forecasting procedure. Uncertainties associated with the forecasts can often be described by prediction intervals. The difficulty with building prediction interval is how to derive the forecast error variance. For traditional methods, mathematical models are used to find the forecast error

variance and provide prediction intervals. However, examining the underlying ANFIS structure indicates that two types of parameters need to be considered for the model, these two types of parameters are: nonlinear parameters for membership functions (S_1) and linear parameters for Sugeno fuzzy model (S_2). The identification of these parameters is based on the ANFIS hybrid learning algorithm. In the forward pass, for fixed nonlinear parameters, the least squares method is used to obtain the linear parameters; while in the backward pass, when linear parameters are fixed, the gradient descent method is used to update the nonlinear parameters. Obviously, the computations of S_1 and S_2 are conditioned on each other; therefore, it is very difficult and intractable to analyze the error variance by using mathematical functions. As a consequence, theoretical forecast error variance expressions cannot be easily derived for the underlying problem. In reviewing the prediction interval literature, Chatfield (1993) observed that, when the theoretical formulae are not available or there are doubts about model assumptions, the use of empirically based methods should be considered as a general purpose alternative. The bootstrapping method introduced by Efron (1979) belongs to this category.

In fact, the bootstrap technique is a very suitable method for the neuro-fuzzy model being studied here due to the fact that theoretical formulae of error variance are not available for the underlying nonlinear domain, and furthermore, no assumptions have been made for our model. Other approaches such as delta method and jackknife could also be used for estimating the variance, however, as stated in Chapter 2, these methods are either intractable or inappropriate to use in some cases. Therefore, we will use the bootstrap method for variance estimation in this research.

Different ways of bootstrapping could be used when applying this technique. Usually, ‘bootstrapping cases’ and ‘bootstrapping residuals’ are two most often used approaches. ‘Bootstrapping cases’ is to consider training data as sampling units, sample with replacement from the training set to create a bootstrap sample; while ‘bootstrapping residuals’ is to treat the model residuals as the sampling units, and create a bootstrap sample by adding residuals to the model fit value.

Considering that the problem being studied is time series forecasting, whereas using bootstrapping cases could cause the problem of samples being disordered, therefore we will use ‘bootstrapping residuals’ for the purpose of obtaining bootstrap replications. The more detailed procedure is as follows: First, calculate the residuals from the training process and generate B bootstrap samples from the empirical distribution of the residual set; for each bootstrap sample of residuals, there is a corresponding bootstrap replicate of training data set, which are obtained from the original trained values by adding the bootstrapped residuals. The bootstrapped training data sets are used to reevaluate the model parameters and hence will generate B bootstrap replications of forecasts for each period. These B forecasts will therefore be used for estimating the variation from the forecast mean, while the error variance for the forecast will be derived after including the random variation of the process.

Chapter 4

Generating Forecasts Using ANFIS

The goal of time series forecasting is to use past values of the time series up to time t to predict the value at some point in the future $t + k$; therefore the common method for this type of forecasting is to create a mapping from previous m points to a predicted future value.

4.1 The ANFIS Model

The ANFIS model we used in this research, depicted in Figure 4.1, is a multi-inputs, univariate output system. More specifically, we use three past values from the time series to predict a future value. The time series are grouped to form input sets and the corresponding outputs; each input is composed of a set with 3-tuples with the form $(x_{t-2\tau}, x_{t-\tau}, x_t)$ and the output as $x_{t+\tau}$, which is the output y in Figure 4.1. Thus, if we want to predict a one-step ahead future value, the inputs are represented by the 3-tuples as (x_1, x_2, x_3) , (x_2, x_3, x_4) , ..., and the output will be $x_4, x_5, \dots$; if we want to forecast a three-step ahead future value, the inputs are represented by the 3-tuples as (x_1, x_4, x_7) , (x_2, x_5, x_8) , ..., and the output will be $x_{10}, x_{11}, \dots$, in other words, the time series is spaced $\Delta=3$ apart in this case.

The number of membership functions (MFs) assigned to each input of the ANFIS is set to two; therefore the number of fuzzy rules is $2^3 = 8$. The MFs are described by

Gaussian membership function with two parameters, center p and width q . The fuzzy model in layer 4 is a first-order Sugeno model with three inputs where the polynomial linear function is represented by $f = a_i x_1 + b_i x_2 + c_i x_3 + d_i$, where $i = 1, 2, \dots, 8$. By using the above settings, the ANFIS forecasting structure proposed here contains 12 premise (nonlinear) parameters, which equals number of inputs (3) $\times$ number of MFs (2) $\times$ number of parameters for MFs (2); and 32 consequent (linear) parameters, which is number of coefficients (4) $\times$ number of rules (8).

Although, in practice, model uncertainty could likely be one of the typical main sources of uncertainty in any forecasting problem, as pointed by Chatfield (1995), we believe that the ability of ANFIS structure along with the advantage of Sugeno fuzzy model, which arise from the sample-based learning and inferencing, is capable of handling model uncertainty situation. We, therefore, assume that the underlying model is an appropriate structure for the problems being studied in this research, and thus model uncertainty issue will not be addressed in the following.

4.1.1 Network Structure

The network structure of the ANFIS model described above is shown in Figure 4.1, where the inputs are fed to the network from the left end and propagated through five layers till the forecast output is generated at the right end.

Figure 4.1 ANFIS forecasting structure

4.1.2 Learning Algorithm

As reviewed in Chapter 2, hybrid learning is the most strong and attractive point of ANFIS structure. Besides, Sugeno fuzzy model provides a systematic way to generate fuzzy rules from a given input-output data set. The fuzzy rules being used in the ANFIS model showed in Figure 4.1 are as follows:

Rule 1: If x_1 is A_1 , x_2 is B_1 , and x_3 is C_1 , then $f_1 = a_1x_1 + b_1x_2 + c_1x_3 + d_1$

Rule 2: If x_1 is A_1 , x_2 is B_1 , and x_3 is C_2 , then $f_2 = a_2x_1 + b_2x_2 + c_2x_3 + d_2$

Rule 3: If x_1 is A_1 , x_2 is B_2 , and x_3 is C_1 , then $f_3 = a_3x_1 + b_3x_2 + c_3x_3 + d_3$

Rule 4: If x_1 is A_1 , x_2 is B_2 , and x_3 is C_2 , then $f_4 = a_4x_1 + b_4x_2 + c_4x_3 + d_4$

Rule 5: If x_1 is A_2 , x_2 is B_1 , and x_3 is C_1 , then $f_5 = a_5x_1 + b_5x_2 + c_5x_3 + d_5$

Rule 6: If x_1 is A_2 , x_2 is B_1 , and x_3 is C_2 , then $f_6 = a_6x_1 + b_6x_2 + c_6x_3 + d_6$

Rule 7: If x_1 is A_2 , x_2 is B_2 , and x_3 is C_1 , then $f_7 = a_7x_1 + b_7x_2 + c_7x_3 + d_7$

Rule 8: If x_1 is A_2 , x_2 is B_2 , and x_3 is C_2 , then $f_8 = a_8x_1 + b_8x_2 + c_8x_3 + d_8$

where A_i , B_i , and C_i , $i = 1, 2$, are Gaussian membership functions with center p_j and width q_j , $j = 1, 2, \dots, 6$.

The linear parameters (a_i , b_i , c_i , d_i , $i = 1, 2, \dots, 8$) in the rules set are determined in the forward pass, wherein the output of each layer is propagated forward layer by layer, till it reaches layer 4 in which the linear parameters (coefficients for polynomial functions) are identified through the input-output data pairs by using recursive least square method; and in layer 5, the overall output is obtained by summing all the normalized nodes output. The relationships among the layers are specified formally as:

Layer 1 (membership functions):

$$\begin{aligned} \mu_{A1} &= e^{-\frac{1}{2}(\frac{x_1-p_1}{q_1})^2} & \mu_{A2} &= e^{-\frac{1}{2}(\frac{x_1-p_2}{q_2})^2} \\ \mu_{B1} &= e^{-\frac{1}{2}(\frac{x_2-p_3}{q_3})^2} & \mu_{B2} &= e^{-\frac{1}{2}(\frac{x_2-p_4}{q_4})^2} \\ \mu_{C1} &= e^{-\frac{1}{2}(\frac{x_3-p_5}{q_5})^2} & \mu_{C2} &= e^{-\frac{1}{2}(\frac{x_3-p_6}{q_6})^2} \end{aligned} \tag{4.1}$$

Layer 2 (fuzzy weights):

4.2

$$W_1 = \mu_{A1} * \mu_{B1} * \mu_{C1} = e^{-\frac{1}{2}\left(\frac{x_1-p_1}{q_1}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_2-p_3}{q_3}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_3-p_5}{q_5}\right)^2}$$

$$W_2 = \mu_{A1} * \mu_{B1} * \mu_{C2} = e^{-\frac{1}{2}\left(\frac{x_1-p_1}{q_1}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_2-p_3}{q_3}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_3-p_6}{q_6}\right)^2}$$

$$W_3 = \mu_{A1} * \mu_{B2} * \mu_{C1} = e^{-\frac{1}{2}\left(\frac{x_1-p_1}{q_1}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_2-p_4}{q_4}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_3-p_5}{q_5}\right)^2}$$

$$W_4 = \mu_{A1} * \mu_{B2} * \mu_{C2} = e^{-\frac{1}{2}\left(\frac{x_1-p_1}{q_1}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_2-p_4}{q_4}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_3-p_6}{q_6}\right)^2}$$

$$W_5 = \mu_{A2} * \mu_{B1} * \mu_{C1} = e^{-\frac{1}{2}\left(\frac{x_1-p_2}{q_2}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_2-p_3}{q_3}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_3-p_5}{q_5}\right)^2}$$

$$W_6 = \mu_{A2} * \mu_{B1} * \mu_{C2} = e^{-\frac{1}{2}\left(\frac{x_1-p_2}{q_2}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_2-p_3}{q_3}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_3-p_6}{q_6}\right)^2}$$

$$W_7 = \mu_{A2} * \mu_{B2} * \mu_{C1} = e^{-\frac{1}{2}\left(\frac{x_1-p_2}{q_2}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_2-p_4}{q_4}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_3-p_5}{q_5}\right)^2}$$

$$W_8 = \mu_{A2} * \mu_{B2} * \mu_{C2} = e^{-\frac{1}{2}\left(\frac{x_1-p_2}{q_2}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_2-p_4}{q_4}\right)^2} * e^{-\frac{1}{2}\left(\frac{x_3-p_6}{q_6}\right)^2}$$

Layer 3 (normalized fuzzy weights):

4.3

$$\bar{W}_i = \frac{W_i}{\sum_{j=1}^8 W_j} \quad i=1, 2, 3, \dots, 8$$

Layer 4 (normalized value for fuzzy rules):

$$\bar{W}_i f_i = \frac{W_i}{\sum_{j=1}^8 W_j} (a_i x_1 + b_i x_2 + c_i x_3 + d_i) \quad i=1, 2, 3, \dots, 8 \quad 4.4$$

Layer 5 (Overall output):

$$\begin{aligned} Y &= \bar{W}_1 f_1 + \bar{W}_2 f_2 + \bar{W}_3 f_3 + \bar{W}_4 f_4 + \bar{W}_5 f_5 + \bar{W}_6 f_6 + \bar{W}_7 f_7 + \bar{W}_8 f_8 \\ &= \sum_{i=1}^8 \bar{W}_i a_i x_1 + \sum_{i=1}^8 \bar{W}_i b_i x_2 + \sum_{i=1}^8 \bar{W}_i c_i x_3 + \sum_{i=1}^8 \bar{W}_i d_i \end{aligned} \quad 4.5$$

Obviously, the overall output is a nonlinear function of the inputs. In the training process, once the predicted value is obtained, the error will be calculated based on the deviation from the actual value and thereafter be propagated backward to update the nonlinear parameters (parameters for membership functions) by gradient decent method, where the goal is to minimize the error function E as defined by equation (4.6) below:

$$E = \frac{1}{2} [y(t) - \hat{y}(t)]^2 \quad 4.6$$

where $y(t)$ is the actual output and $\hat{y}(t)$ is the predicted output.

From the backpropagation algorithm of Rumelhart (1986), we have:

$$r(t+1) = r(t) - \eta \frac{\partial E}{\partial r} \quad 4.7$$

where r is the adjustable parameter of each membership function, and η is the assigned learning rate. Thus, the nonlinear parameters (center p_i and width q_i) can be fine-tuned as:

$$p_i(t+1) = p_i(t) - \eta_1 \frac{\partial E}{\partial p_i} \quad 4.8$$

$$q_i(t+1) = q_i(t) - \eta_2 \frac{\partial E}{\partial q_i} \quad 4.9$$

where η_1 is the learning rate for p_i and η_2 is the learning rate for q_i . Take updating parameter p_1 as an example, we illustrate the step-by-step backward propagation in the following.

By using chain rule to take partial differentiation sequentially, we have

$$\frac{\partial E}{\partial p_1} = \frac{\partial E}{\partial y(t)} \cdot \frac{\partial y(t)}{\partial p_1} \quad 4.10$$

$$\frac{\partial E}{\partial y(t)} = -[y(t) - \hat{y}(t)] = -e \quad 4.11$$

$$\frac{\partial y(t)}{\partial p_1} = \frac{\partial O_5}{\partial p_1} = \frac{\partial O_5}{\partial O_4} \cdot \frac{\partial O_4}{\partial O_3} \cdot \frac{\partial O_3}{\partial O_2} \cdot \frac{\partial O_2}{\partial O_1} = \frac{\partial O_5}{\partial O_4} \cdot \frac{\partial O_4}{\partial O_3} \cdot \frac{\partial O_3}{\partial O_2} \cdot \frac{\partial O_2}{\partial \mu_{A1}} \cdot \frac{\partial \mu_{A1}}{\partial p_1} \quad 4.12$$

$$\frac{\partial O_5}{\partial O_4} = 1 \quad 4.13$$

$$\frac{\partial O_4}{\partial O_3} = f_1 = a_1 x_1 + b_1 x_2 + c_1 x_3 + d_4 \quad 4.14$$

$$\frac{\partial O_3}{\partial O_2} = \frac{\partial \bar{W}_1}{\partial W_1} = \frac{\partial \left(\frac{W_1}{\sum_{i=1}^8 W_i} \right)}{\partial W_1} = \frac{1 \cdot (\sum_{i=1}^8 W_i) - W_1 \cdot 1}{(\sum_{i=1}^8 W_i)^2} = \frac{W_{sum} - W_1}{W_{sum}^2} \quad 4.15$$

$$(\text{Note: } W_{sum} = \sum_{i=1}^8 W_i)$$

$$\frac{\partial O_2}{\partial O_1} = \frac{\partial O_2}{\partial \mu_{A1}} = \mu_{B1} * \mu_{C1} \quad 4.16$$

$$\frac{\partial \mu_{A1}}{\partial p_1} = e^{-\frac{1}{2}(\frac{x_1-p_1}{q_1})^2} \cdot \frac{(x_1-p_1)}{q_1^2} \quad 4.17$$

$$\frac{\partial \mu_{A1}}{\partial q_1} = e^{-\frac{1}{2}(\frac{x_1-p_1}{q_1})^2} \cdot \frac{(x_1-p_1)^2}{q_1^3} \quad 4.18$$

Therefore,

$$\begin{aligned} \Delta p_{11} &= \eta_1 \cdot e \cdot (a_1 x_1 + b_1 x_2 + c_1 x_3 + d_4) \cdot \frac{W_{sum} - W_1}{W_{sum}^2} \cdot e^{-\frac{1}{2}(\frac{x_1-p_1}{q_1})^2} * e^{-\frac{1}{2}(\frac{x_2-p_3}{q_3})^2} * e^{-\frac{1}{2}(\frac{x_3-p_5}{q_5})^2} \cdot \frac{x_1-p_1}{p_1^2} \\ &= \eta_1 \cdot e \cdot f_1 \cdot \frac{W_{sum} - W_1}{W_{sum}^2} \cdot W_1 \cdot \frac{x_1-p_1}{p_1^2} \end{aligned} \quad 4.19$$

where Δp_{11} is the change of parameter p_1 as the derivative of error measure with respect to the output of MF A_1 in layer 1 corresponding to the first fuzzy rule in layer 4. The structure of updating parameter p_1 for membership function A_1 is depicted in Figure 4.2.

Figure 4.2 Structure of updating p_1 for MF A_1

Similarly,

$$\Delta p_{12} = \eta_1 \cdot e \cdot f_2 \cdot \frac{W_{sum} - W_2}{W_{sum}^2} \cdot W_2 \cdot \frac{x_1 - p_1}{p_1^2} \quad 4.20$$

$$\Delta p_{13} = \eta_1 \cdot e \cdot f_3 \cdot \frac{W_{sum} - W_3}{W_{sum}^2} \cdot W_3 \cdot \frac{x_1 - p_1}{p_1^2} \quad 4.21$$

$$\Delta p_{14} = \eta_1 \cdot e \cdot f_4 \cdot \frac{W_{sum} - W_4}{W_{sum}^2} \cdot W_4 \cdot \frac{x_1 - p_1}{p_1^2} \quad 4.22$$

Therefore,

$$\Delta p_1 = \Delta p_{11} + \Delta p_{12} + \Delta p_{13} + \Delta p_{14} \quad 4.23$$

And, p_1 is fine-tuned by

$$p_1(t+1) = p_1(t) + \Delta p_1 \quad 4.24$$

Likewise, using same procedure to fine-tune parameter q_1 , we have

$$\begin{aligned} \Delta q_{11} &= \eta_2 \cdot e \cdot (a_1 x_1 + b_1 x_2 + c_1 x_3 + d_4) \cdot \frac{W_{sum} - W_1}{W_{sum}^2} \cdot e^{\frac{1}{2}(\frac{x_1 - p_1}{q_1})^2} * e^{\frac{1}{2}(\frac{x_2 - p_3}{q_3})^2} * e^{\frac{1}{2}(\frac{x_3 - p_5}{q_5})^2} \cdot \frac{(x_1 - p_1)^2}{q_1^3} \\ &= \eta_2 \cdot e \cdot f_1 \cdot \frac{W_{sum} - W_1}{W_{sum}^2} \cdot W_1 \cdot \frac{(x_1 - p_1)^2}{q_1^3} \end{aligned} \quad 4.25$$

$$\Delta q_{12} = \eta_2 \cdot e \cdot f_2 \cdot \frac{W_{sum} - W_2}{W_{sum}^2} \cdot W_2 \cdot \frac{(x_1 - p_1)^2}{q_1^3} \quad 4.26$$

$$\Delta q_{13} = \eta_2 \cdot e \cdot f_3 \cdot \frac{W_{sum} - W_3}{W_{sum}^2} \cdot W_3 \cdot \frac{(x_1 - p_1)^2}{q_1^3} \quad 4.27$$

$$\Delta q_{14} = \eta_2 \cdot e \cdot f_4 \cdot \frac{W_{sum} - W_4}{W_{sum}^2} \cdot W_4 \cdot \frac{(x_1 - p_1)^2}{q_1^3} \quad 4.28$$

Hence,

$$\Delta q_1 = \Delta q_{11} + \Delta q_{12} + \Delta q_{13} + \Delta q_{14} \quad 4.29$$

And, q_1 is updated by

$$q_1(t+1) = q_1(t) + \Delta q_1 \quad 4.30$$

Other premise parameters (nonlinear parameters $p_i, q_i, i = 2 \dots 6$) could also be obtained in the same way as the above. The learning process is iterated and therefore the model parameters are kept being updated until the specified epoch numbers have been reached.

4.2 Forecasting

In order to use the ANFIS structure in Figure 4.1 to generate forecasts, the first step is to initialize membership functions for input variables. Because of the unequally distributed data points, the membership functions will not be equally spaced. Thus the process will begin with data clustering to group data around the respective centers, calculate the widths and therefore determine the membership functions.

4.2.1 Data Clustering

At the start of the process, initial values $p_1, p_2, \dots p_k$ are assigned arbitrarily as k centered values to the training data set $(x_1, x_2, \dots x_n)$, where

$$\min(x_1, x_2, \dots x_n) < p_i < \max(x_1, x_2, \dots x_n) \quad 1 \leq i \leq k$$

Then the data are grouped around the initial centers according to

$$|x_j - p_c| = \min_i \{|x_j - p_i|\} \quad 1 \leq i \leq k, 1 \leq j \leq n \quad 4.31$$

where p_c is the center associated with x_j . Let $x_j(t)$ be an input and $p_c(t)$ the value of p_c at iteration t . If x_j belongs to the grouping of p_c , then

$$p_c(t+1) = p_c(t) + \eta[x_j(t) - p_c(t)] \quad 4.32$$

where $\eta(0 < \eta < 1)$ is a small positive learning rate.

If x_j does not belongs to the grouping of p_c , then

$$p_c(t+1) = p_c(t) \quad 4.33$$

The centered values of the membership function will be changed with iterations. After the centers have been decided, the widths of membership functions will be determined by computing the discrepancy of the x_j and the center of its associated group. We adopt the equation for computing width from Li, Ang and Gray (1999), which can be written as:

$$q_i = \frac{1}{R} \sqrt{\frac{1}{g_i} \sum_{j=1}^{g_i} (x_j - p_i)^2} \quad 1 \leq i \leq k \quad 4.34$$

where

q_i = width of membership function i

p_i = center of membership function i

x_j = observed data sample

k = total number of membership function nodes

g_i = total number of data samples in i th membership function group

R = overlap parameter

4.2.2 Forecasting Procedures

The underlying forecasting procedure is a two-stage iterative procedure, in which the first stage is to identify the model parameters through the training process, while the second stage uses those estimated parameters to generate forecasts and at the same time performs diagnostics checks to determine the adequacy of the model parameters in the testing process. The flowcharts of these two processes are shown in Figure 4.3 and Figure 4.4, respectively.

The intelligent mechanism of the forecasting system being studied functions in the training process. The learning and inferencing ability of the ANFIS model are gained through the iterations of training data sets. The premise parameters (nonlinear parameters) of the model are first initialized based on data clustering of input variables; then the consequent parameters (linear parameters) are determined in the forward propagation by using recursive least squares method; however, these linear parameters, in turn, impact the updating of the nonlinear parameters in the backward propagation.

The model parameters identified in the first stage will then be used in the second stage to generate forecasts through forward propagation. Meanwhile, the diagnostic checking will be implemented by using tracking signals in this stage. If at some point, the forecast errors exhibit some structure, then all the latest available information before that point will be incorporated into the training set and the network will be retrained. This process will continue until all the tracking signal ratios are within the specified limit, which leads the overall residuals of the forecasts to resemble a random pattern.

Figure 4.3 Flowchart for training process

Figure 4.4 Flowchart for testing process

4.3 The Data

Three univariate nonstationary time series data sets are used for testing the proposed forecasting structure. For each time series, the training set is required to be larger than the testing set; as a default setting for the system, the data are split such that half are used for training and half used for testing.

4.3.1 Case I

The first case represents weekly sales of a cutting tool with 100 observations. The data are taken from *Forecasting & Time Series Analysis* (Montgomery, Johnson and Gardiner, 1990), and are listed in Appendix V. Figure 4.5 shows the data set which, obviously, is a nonstationary time series with no natural mean.

Figure 4.5 Weekly sales of a cutting tool

4.3.2 Case II

The second case is the quarterly Standard & Poor 500 index with 388 observations (from 1900 to 1996). The data set is taken from the Time Series Data Library website by Hyndman & Akram who have used the book of Makridakis, Wheelwright and Hyndman (1998) as their source. These data can also be found in Appendix VI. As shown in Figure 4.6, the observations of any local segment are away from the average value (1.0708), which indicates that the time series is also nonstationary in the mean.

Although forecasting for stock market is indeed not an easy task, a lot of different methods have been attempted by academia and practitioners. We use the S&P 500 data set in this research for the purpose of testing the proposed method.

Figure 4.6 Quarterly S&P 500 index (1900-1996)

4.3.3 Case III

The third case is the quarterly beer production in Australia with 154 observations (from March 1956 to June 1994). This data set is also taken from the Time Series Data Library website by Hyndman & Akram, and is shown in Appendix VII. The plot in Figure 4.7 reveals the nonstationarity of the data, wherein the seasonal effects and some trend can be easily observed.

Figure 4.7 Quarterly beer production in Australia (Mar 1956 - Jun 1994)

Chapter 5

Monitoring The Forecasting Process

Monitoring is an essential aspect of any forecasting system. This is especially important for a time series forecasting system since there is no guarantee that the past behaviors and characteristics of the system continue in the future. When a nonrandom (systematic) change occurs, some corrective actions need to be taken to ensure that the system is adjusted in a timely fashion. The role of monitoring is to detect and identify the occurrence of nonrandom changes from existing patterns or relationships.

5.1 Tracking Signals Test

Tracking signals test is one of most important approaches to monitoring a forecasting system. It uses quantitative procedures to automatically keep track of forecasting errors, and determines when a pattern or relationship has changed. Primarily, tracking signals are based on the concept of statistical process control, where the objective is to discover as soon as possible when a process or product has deviated in a nonrandom manner from its normal value. Tracking signals provide quality control for a forecasting system.

Common tracking signals are expressed in the form of ratios. The numerator is a weighted or unweighted sum of forecast errors, while the denominator is usually the mean absolute deviation (MAD). The numerator has an expected value of zero when the

forecast is in control, and would move away from zero if the forecast goes out of control. The more the forecasting errors vary from zero in some systematic way, the greater the chances that the established pattern or relationship being monitored has changed. The denominator is the sum of forecast errors; without regard to sign, it is weighted by exponential smoothing so that more recent errors count more heavily than the older ones. In general, tracking signal is chosen by determining the weights to use in the numerator and the denominator. Most of the research in this area has been focused on the numerator.

We apply Trig's smoothed error tracking signals test into the ANFIS forecasting structure described in Chapter 4. As reviewed in Chapter 2, smoothed error E_t (equation 2.23) and mean absolute deviation M_t (equation 2.24) need to be updated when a new forecast becomes available, and the ratio of these two i.e., tracking signals T_t (equation 2.25), will determine whether or not the model parameters need to be reevaluated. Usually, same smoothing parameter is used for both the numerator and denominator. McClain (1988) pointed out that the purpose of the denominator (MAD) is to measure the long-run average variability of forecast errors in order to standardize the numerator, hence he suggested to use a separate smoothing parameter for the denominator of the smoothed error tracking signal. Equations (2.23), (2.24), and (2.25) can thus be modified as follows:

$$\text{Smoothed Error:} \quad E_t = \alpha_e e_t + (1 - \alpha_e) E_{t-1} \quad 5.1$$

where α_e = Error smoothing parameter

$$\text{Mean Absolute Deviation:} \quad M_t = \alpha_m |e_t| + (1 - \alpha_m) M_{t-1} \quad 5.2$$

where α_m = MAD smoothing parameter

$$\text{Smoothed Error Tracking Signal: } T_t = \left| \frac{E_t}{M_t} \right| \quad 5.3$$

A control limit must be pre-specified to be compared with the tracking signal T_t . The selection of the control limit needs to take into consideration values chosen for the smoothing parameters (α_e and α_m). An important finding is that the selection of error smoothing parameter α_e and the MAD smoothing parameter α_m is data-specific; different parameter values may be chosen to suit different situations. When there is a dramatic change in the system, the smoothing parameters need to be large, which means, more weight is placed on the current observation to ensure that the change will be quickly captured; on the other hand, if the system is relatively stable, the smoothing parameters need to be relatively small, i.e., the weight given to historical data decays more slowly.

5.2 Procedures

Since the purpose of tracking signals test is to monitor the forecasting process, i.e., to detect if any nonrandom changes have occurred when generating forecasts; therefore, in this research, tracking signals test is only applied to the testing stage. As a matter of fact, from an overall point of view, the neuro-fuzzy hybrid learning algorithm employed in training stage provides a powerful approach to learning and inferencing from historical information while in the testing stage, tracking signals test gives a good sense about when the set of historical information need to be updated by incorporating new observations. The flowchart for implementing tracking signals test is given in Figure 5.1.

Figure 5.1 Flowchart for tracking signals test

5.3 Data Description

Usually, it is preferred that a big pool of data be used in training stage, which would provide a large amount of information as defaults when applying model parameters to the testing stage. However, given the fact that sometimes there might not be enough data available currently; or, as time goes by, more recent data becomes available while some old historical observations might distort the current structure of the system; or, some nonrandom changes might occur, which would result in the forecasts not to be unbiased. Therefore, detecting the fundamental changes in the system and keeping the system updated as soon as possible are not only necessary but play a very important role in time series forecasting.

In the following, we use the three time series data sets as described in Chapter 4 to show the effect of incorporation of tracking signals test into the ANFIS model. We will generate multiple-step-ahead forecasts at current time from three previous values and graphically show how the tracking signals test will improve the forecast accuracy.

Implementation of the ANFIS forecasting system proposed in this research is accomplished by a program written in Microsoft Visual Basic. At the start, source data set is imported to the system, and then it is split into a training set and a testing set. Parameters such as the number of periods ahead to be forecasted also need to be specified. With these settings, the training inputs and the desired output mappings are obtained. The membership functions are initialized by the operation of submenu under *Train*, and can be graphically displayed by clicking on the input variable of interest from the *View* menu. At the end of the ANFIS training stage, forecasts are generated by

clicking the *Test* menu, after which the test data set will pop up and the forecasts will be displayed. When there is an indication that a forecast is out of control, training process will restart using newly obtained training set. To facilitate data manipulation and graphing, all the results from training stage through testing stage can be exported to Excel files.

5.3.1 Case I

The first data set contains 100 observations of a nonstationary time series representing weekly sales of a cutting tool, as shown in Figure 4.5. By examining the time series, it is not difficult to observe that over time, the sales value exhibits some dramatic changes which appear to be a number of ramp variations. Although trend, seasonal, and cyclic effects are usually considered separately in traditional time series forecasting, they will not be distinguished in the underlying intelligent forecasting system.

Data

Time Period	Demand
1	100.0
2	94.0
3	90.0
4	96.0
5	91.0
6	94.0
7	95.0
8	99.0
9	95.0
10	98.0
11	100.0
12	97.0
13	99.0
14	98.0
15	96.0
16	98.0
17	96.0

Total Data:

Please Input/Select the Following:

Number of Inputs:

Number of MF:

Periods_ahead:

Training Data Number

☒ Default:

☐ Alternative:

Testing Data Number

☒ Default:

☐ Alternative:

Figure 5.2 Data import and training preparation (cutting tool data)

Of the total 100 observations, the first 50 observations are used for training and the remaining 50 observations are used for testing. The three-step ahead forecasting training set is partitioned as three inputs X_1 , X_2 , and X_3 , and each with two membership functions, as shown in Figure 5.2. The training set can thus be mapped as: use the first input data (x_1 , x_4 , x_7) to forecast x_{10} , the second input data (x_2 , x_5 , x_8) to forecast x_{11} , ...and so on, see Figure 5.3. The membership function for each input is obtained by data

clustering, which can be viewed respectively by selecting the corresponding input. As an example, the membership functions of X_1 are shown in Figure 5.3.

Figure 5.3 Training set partitioning and MFs clustering (cutting tool data)

After the initialization of the membership functions, the training process takes place through the inferencing and learning procedure: the model parameters are determined and the predicted values are generated. Meanwhile, we observe that the membership functions have been slightly shifted after learning process. The error terms (residuals) are calculated from the desired value (actual) and trained value (predicted) as

Figure 5.4 Training results for three-step ahead forecasts (cutting tool data)

defined by $e_i = Y_i - \hat{Y}_i$, $i = 1, 2, \dots, n$, where Y_i is an actual observation and $\hat{Y}_i$ is the predicted value. The residuals plot of the training set is shown in Figure 5.5. It can be easily seen that the residuals fall within a horizontal band centered around zero with constant variance. It displays no systematic tendencies to be either positive or negative, and therefore presents a desired pattern.

Figure 5.5 Plot of residuals for training set (cutting tool data)

Nevertheless, when applying the model parameters obtained from the training stage to the holdout testing data set to generate forecasts, the results in Figure 5.6 are not that desirable; this can also be verified from the residuals plot of Figure 5.7.

Testing Data Set						
Time Period	X1	X2	X3	Actual	Forecast	Error
1	122.0	116.0	118.0	124.0	120.6	3.4
2	118.0	118.0	120.0	123.0	121.6	1.4
3	120.0	119.0	123.0	121.0	120.9	.1
4	116.0	118.0	124.0	126.0	122.9	3.1
5	118.0	120.0	123.0	128.0	122.1	5.9
6	119.0	123.0	121.0	124.0	120.8	3.2
7	118.0	124.0	126.0	122.0	122.5	-.5
8	120.0	123.0	128.0	123.0	121.2	1.8
9	123.0	121.0	124.0	119.0	119.2	-.2
10	124.0	126.0	122.0	120.0	118.7	1.3
11	123.0	128.0	123.0	125.0	119.1	5.9
12	121.0	124.0	119.0	126.0	119.5	6.5
13	126.0	122.0	120.0	129.0	118.4	10.6
14	128.0	123.0	125.0	132.0	115.9	16.1
15	124.0	119.0	126.0	131.0	117.3	13.7
16	122.0	120.0	129.0	130.0	118.5	11.5
17	123.0	125.0	132.0	133.0	118.6	14.4
18	119.0	126.0	131.0	134.0	121.8	12.2
19	120.0	129.0	130.0	133.0	121.1	11.9

Figure 5.6 Three-step ahead forecasting results without tracking signals test (cutting tool data)

Obviously, the results in Figure 5.6 strongly indicate that the forecast error are increasing over time after a certain period and displaying a systematic tendency of being positive. The residuals plot in Figure 5.7 confirms that the residuals do not scatter randomly around zero; instead, it shows some systematic pattern of being positive, which actually exhibits that autocorrelation exists in forecast errors. The nonrandom changes in the testing set, i.e., systematic deviation from the established desired pattern of training set, call for the employment of tracking signals test, which would be used to detect when

the existing relationship has changed and result in a reevaluation of the model parameters.

Figure 5.7 Plot of residuals for testing set (cutting tool data)

Having observed that the cutting tool training data is a nonstationary series with big jumps and increases over time, we choose relatively large smoothing parameters as:

Error Smoothing Parameter $\alpha_e = 0.4$;

Mean Absolute Deviation Smoothing Parameter $\alpha_m = 0.2$;

Tracking Signal Limit $T_{\text{limt}} = 0.4$.

By following the tracking signals test procedures summarized in Figure 5.1, it was noticed that after the 11th week of prediction, the tracking signal detected that forecast

was out of control as shown in Figure 5.8; this can also be confirmed by the residuals plot of Figure 5.7.

Time Period	X1	X2	X3	Actual	Forecast	Error
1	122.0	116.0	118.0	124.0	120.6	3.4
2	118.0	118.0	120.0	123.0	121.6	1.4
3	120.0	119.0	123.0	121.0	120.9	.1
4	116.0	118.0	124.0	126.0	122.9	3.1
5	118.0	120.0	123.0	128.0	122.1	5.9
6	119.0	123.0	121.0	124.0	120.8	3.2
7	118.0	124.0	126.0	122.0	122.5	-.5
8	120.0	123.0	128.0	123.0	121.2	1.8
9	123.0	121.0	124.0	119.0	119.2	-.2
10	124.0	126.0	122.0	120.0	118.7	1.3
11	123.0	128.0	123.0	125.0	119.1	5.9
12	121.0	124.0	119.0	126.0		
13	126.0	122.0	120.0	129.0		
14	128.0	123.0	125.0	132.0		
15	124.0	119.0	126.0	131.0		
16	122.0	120.0	129.0	130.0		
17	123.0	125.0	132.0	133.0		
18	119.0	126.0	131.0	134.0		

Figure 5.8 Indication of forecasting out of control (cutting tool data)

Next, the 11 new observations are added to the training set, and at the same time the first 11 observations of the original training data are discarded. With the new training set, the model parameters will be reevaluated by the retraining process and forecasts will be generated starting from the previous stopped point (12th week). The process of training →

forecasting → retraining → forecasting will continue until all the forecasts are in control with the residuals plot exhibiting no structure or no systematic pattern.

Figure 5.9 Plot of residuals for testing set after tracking signals applied (cutting tool data)

Figure 5.9 shows the residuals plot after the application of tracking signals test. The plot reveals that the residuals display no positive tendency but present a structureless, random pattern. Also, the residuals are smaller than when the tracking signals test was not being used, which indicates that forecasting accuracy has been largely improved after the employment of tracking signals test. In summary, the example of nonstationary time series of the weekly sales of cutting tool data demonstrates that tracking signals test can

indeed quickly detect the nonrandom changes in the system, update the model parameters based on the latest information and bring the forecasts back under control.

5.3.2 Case II

The second example data set is the quarterly S&P 500 index, a nonstationary time series with 388 observations as shown in Figure 4.6, of which the first 194 observations are used as training set and the remaining 194 observations are used as holdout testing set. We will again generate a three-step-ahead forecast from three previous values in this case, in which the training data are partitioned as three inputs X_1 , X_2 , X_3 , each spaced 3 periods apart. The membership functions of inputs, graphical display of MFs for X_2 , and the training results are shown in Figure 5.10.

The residuals plot of training stage is presented in Figure 5.11. It indicates that no special pattern exists; the residuals mean is centered around zero and the points seem to be randomly scattered.

Figure 5.10 Training stage for three-step ahead forecasts (S&P 500 index data)

Figure 5.11 Plot of residuals for training set (S&P 500 index data)

The forecasts could thus be generated and the residuals be calculated by applying the model parameters obtained from training stage to the holdout testing set data. The plot of residuals in Figure 5.12 reveals that there is no strong indication of nonrandomness, i.e., the established pattern or relationship from the training stage has not undergone significant change as time goes by. However, it can be shown that the forecast accuracy can still be improved by using tracking signals test.

Figure 5.12 Plot of residuals for testing set (S&P index data)

For the time series where the established relationship or pattern does not change dramatically, we choose relatively small smoothing parameters when applying tracking signals test. Specifically, in this case, the parameters are set as follows:

Error Smoothing Parameter $\alpha_e = 0.1$;

Mean Absolute Deviation Smoothing Parameter $\alpha_m = 0.05$;

Tracking Signal Limit $T_{\text{limt}} = 0.57$.

The residuals plot after using tracking signals test is shown below in Figure 5.13. It can be seen that the residuals exhibit more desirable features, falling within a narrower horizontal band centered around zero with constant variance, and displaying randomness. This actually has been verified by the sum of squared errors (SSE) statistics. For the

testing set data after tracking signals test $SSE = 5.25$, which is smaller compared with the results before tracking signals test, where the $SSE = 5.70$.

Figure 5.13 Plot of residuals for testing set after tracking signals test (S&P index data)

5.3.3 Case III

The third data case is a 154 observations nonstationary time series, which represents the quarterly beer production in Australia from March 1956 to June 1994, as shown in Figure 4.7. The first half of the data is used for training and the remaining half for testing. We will generate a four-step-ahead forecast from three previous values for the purpose of investigating the seasonal effect in this case, where the three inputs X_1 , X_2 , and X_3 are spaced 4 periods apart. The results from training stage are shown in Figure

5.14, where the graphical display of the membership functions for input X_3 is taken as an example. The error terms have been plotted in Figure 5.15. It can be observed that the residuals exhibit a desired random pattern where the mean of the errors is centered around 0.

Figure 5.14 Training stage for four-step ahead forecasts (beer production data)

Figure 5.15 Plot of residuals for training set (beer production data)

The smoothing parameters and control limit for tracking signals test are selected as follows:

Error Smoothing Parameter $\alpha_e = 0.4$;

Mean Absolute Deviation Smoothing Parameter $\alpha_m = 0.15$;

Tracking Signal Limit $T_{\text{limt}} = 0.4$.

The residuals plots for testing set before and after tracking signals test are shown in Figures 5.16 and 5.17, respectively, wherein the mean squared error (MSE) statistics was found to decrease from 1,117.8 to 929.4. This confirms that the forecasting accuracy has been improved again by applying tracking signals test.

Figure 5.16 Plot of residuals for testing set (beer production data)

Figure 5.17 Plot of residuals for testing set after tracking signals test (beer production data)

In summary, tracking signals test is a very important and useful tool for monitoring forecasting process and improving forecasts accuracy. However, care should be taken in the selection of smoothing parameters and setting control limit, as it is a data-specific exercise.

Chapter 6

Building Prediction Intervals

As mentioned before, giving interval forecasts is more important than just giving point forecasts. An interval forecast gives the best guess of a prediction and tells how far in error that guess might be. Interval forecasts can help assess future uncertainty and allow alternative scenarios to be explored. Notice that in some literatures, an interval forecast associated with a prescribed probability is called ‘confidence interval’, but we prefer the description ‘prediction interval’, both because it is more descriptive and because the term ‘confidence interval’ is usually applied to interval estimates of model parameters.

Technically, a prediction interval is an upper and lower limit together with a probability that the future value will lie between the two. If error distribution could be obtained, point estimate can be converted into an interval forecast with the stated probability by adding and subtracting a multiple of the standard deviation of forecast error. Usually, prediction intervals are calculated based on the point forecasts and the respective error standard deviation. However, the difficulty with constructing a prediction interval is to find the forecast error variance and the error distribution. Since the actual observation is not available at the time of generating a forecast, in most cases the variance of forecast error would be estimated from the data in training process. As reviewed in Chapter 2, the delta method, bootstrap, and jackknife are three common

methods used to estimate the standard error for nonlinear models. The bootstrap technique will be used here because of the intractableness of the other two methods.

In order to use the bootstrap method to estimate the forecast error variance, one important issue is to determine what to bootstrap. Considering that the underlying problem is time series forecasting, we will use the bootstrap residuals approach and thereby obtain the bootstrap replicates for the training data indirectly. This is because the time series data imply that observations are a time-ordered sequence, which means if bootstrapping is performed on the observations directly, the replicates will be disordered because of random resampling. We first analyze the residuals of the model for the purpose of finding the empirical distribution for the samples to be bootstrapped, then bootstrap residuals based on the derived distribution to obtain the corresponding training pseudoreplicates B times. These resampled data sets are then used to reevaluate the model parameters and, hence, regenerate forecasts. The forecast error variance will then be computed from the $B+1$ forecasts and prediction intervals will then be constructed based on the forecast standard error.

6.1 Residual Analysis

The examination of residuals usually gives a good indication of the underlying forecasting model. A residual is defined as the difference between the observed value and the corresponding predicted value, the i th residual is denoted as $e_i = Y_i - \hat{Y}_i$. In general, residual analysis is very important for prediction problems due to the following reasons. First, residuals are useful for investigating the model appropriateness; second, residual

analysis is a necessity for determining the empirical distribution before the bootstrapping residual pseudoreplicates can be generated. Finally, the diagnostics for the forecasts are usually carried out indirectly through the examination of residuals. In the following, we analyze the residuals obtained from the training stage to derive the corresponding empirical distribution, which will then be used for creating bootstrap samples.

Graphical methods and quantitative methods are two commonly used techniques in residuals analysis. In practice, the mixture of these two techniques is typically used. The analysis usually starts with some graphical methods such as residuals plot, box plot, and QQ plot, etc., and then followed by the confirmatory statistics of quantitative methods, where hypotheses tests are often applied. Although different statistical tests are available for testing distribution adequacy, Kolmogorov-Smirnov (K-S) test and Chi-Square test are most often used to decide if a sample comes from a population with a specific distribution, and therefore have been embedded in many statistical software packages. In general, Kolmogorov-Smirnov test measures the similarity of two cumulative distributions based on their “maximum difference”; it works by comparing an observed distribution to a known (theoretical) distribution of a continuous random variable. Chi-Square test measures the deviation of expected cell frequencies from observed frequencies and can be applied to either a discrete distribution or a continuous distribution, but requires a sufficient sample size. These two tests for fitting distributions are both referred to as the “Goodness-of-Fit” tests. The preliminary examination of training residuals by using graphical methods such as QQ plot indicates some normality.

In the following, we apply the Rockwell Software Arena Input Analyzer for the purpose of obtaining the empirical distribution of training residuals.

For the residuals of weekly sales data (Case I), it can be seen that the Normal distribution fits best, where the Kolmogorov-Smirnov test statistic is 0.0726 and the corresponding P-value is 0.15. More detailed summary statistics are listed in Appendix III. The graphical display of the distribution is shown in Figure 6.1. For the residuals of S&P index data (Case II), it was observed that both Beta and Normal distributions fit. However, since the Normal distribution is more commonly used when dealing with residuals, we conclude normality in this case, where the Kolmogorov-Smirnov test statistic is computed as 0.057 and the corresponding P-value is 0.15. Figure 6.2 indicates that the normal distribution fits the data well. The summary statistics can also be found in Appendix III. For the residuals of beer production data (Case III), the Kolmogorov-Smirnov test statistic is given as 0.0617 and the corresponding P-value is 0.15, which also leads to the conclusion that the residuals follow Normal distribution, as shown in Figure 6.3.

Figure 6.1 Distribution fitting for cutting tool data training errors: Normal (-0.042, 2.51)

Figure 6.2 Distribution fitting for S&P index data training errors: Normal (0.0109, 0.183)

Figure 6.3 Distribution fitting for beer production data training errors: Normal (-0.149, 16.2)

Keep in mind that we did not make any assumptions about the model parameters and residuals pattern before prediction; whereas the residuals pattern obtained from the underlying ANFIS structure appears to be desirable for the time series forecasting problem at hand.

6.2 Estimating Forecast Error Variance

Typically, forecast error variance is composed of variance of process and variance of forecast. The variances of forecasts for the problem being studied are estimated by bootstrapping predicted values, which essentially, are generated from bootstrap residual replicates; where the residual replicates are drawn from their empirical distribution. In the following, we first present the algorithm of bootstrap residuals sampling; then illustrate the procedure of bootstrapping replicates using the cutting tool data; and finally present the overall estimates of the forecast error variances and the prediction intervals.

6.2.1 Bootstrap Residual Algorithm

The algorithm of bootstrapping residuals works by first creating pseudoreplicates (“bootstrap samples”) of the residuals, from which to generate the bootstrap replications of the training data set indirectly. For each bootstrap sample, the model parameters will be reestimated and a forecast will be regenerated. The variance of the forecast will thus be obtained from the B+1 forecasts, where B is the number of bootstrapping replications. Specifically, the procedure of the algorithm is summarized as follows:

1. Estimate model parameter set $\hat{S}$ from the training data and let $r_i = y_i - y(x_i; \hat{S})$, $i = 1, 2, \dots, n$.

2. Generate B samples, each one with size n drawn from the empirical distribution

$\hat{F}$ of $(r_1, r_2, \dots, r_n)$. Denote the b^{th} sample by $r_1^{*b}, r_2^{*b}, \dots, r_n^{*b}$ and let

$$y_i^{*b} = y(x_i; \hat{S}) + r_i^{*b}.$$

3. For each bootstrap sample of training set, $b = 1, 2, \dots, B$, reestimate model parameters and obtain $\hat{S}^{*b}$.
4. For each model parameters set $\hat{S}^{*b}$, $b = 1, 2, \dots, B$, generate the corresponding bootstrap predicted value $y(x_i; \hat{S}^{*b})$, $i = n+1, n+2, \dots, m$.
5. Estimate the variance of the j th bootstrap predicted value by

$$\frac{1}{B} \sum_{b=1}^B [y(x_i; \hat{S}^{*b}) - \bar{y}(x_i)]^2 \quad \text{where} \quad \bar{y}(x_i) = \sum_{b=1}^B y(x_i; \hat{S}^{*b}) / (B+1).$$

6.2.2 Bootstrap Replications

Bootstrap methods depend on bootstrap samples; each bootstrap sample is an independent random sample of size n from the empirical distribution $\hat{F}$. From the residual analysis in Section 6.1, it was concluded that the training errors from ANFIS forecasting fit normal distribution. In the following, S-PLUS language will be used to implement the bootstrapping residual procedure. An example S-PLUS language code written to create bootstrap replicates is shown in Appendix IV.

In order to generate the pseudoreplicates for the residuals, we first bootstrap the statistics for the empirical distribution $\hat{F}$, which is, to obtain the mean and variance for the normal distribution. For bootstrapping statistics, the number of replications B often takes relatively large value; in this case, the number of replications is 1,000. We use the cutting tool data as an example in the following to walk through the procedure and

generate the bootstrap replicates. The bootstrap replications of mean for training errors from S-PLUS are graphically displayed in Figure 6.4.

The summarized statistics from S-PLUS output showed that the bootstrap mean of the training errors is -0.037, while the observed mean from the original training errors is -0.042, with the bias, i.e., the difference between the bootstrap mean and the observed mean, around 0.00459, and the standard error about 0.34. The normal probability plot (QQ plot) of the mean is graphed in Figure 6.5, which follows a straight line and indicates a desirable normality feature.

Figure 6.4 Bootstrap replications of mean for cutting tool data training errors

Figure 6.5 QQ plot of bootstrap mean for cutting tool data training errors

The bootstrap replications of variance for training errors are shown in Figure 6.6. The summary statistics can be also obtained from S-PLUS output, where the mean values of the bootstrapped variance is 6.356, the observed variance from the original training errors is 6.452, and the difference between these two i.e., the bias, is -0.09663 . The standard error of the bootstrapped variance is given as 1.263. Figure 6.7 displays the QQ plot of replicates' variance.

Figure 6.6 Bootstrap replications of variance for cutting tool data training errors

Figure 6.7 QQ plot of bootstrap variance for cutting tool data training errors

Once the mean and variance have been estimated, the empirical normal distribution $\hat{F}$ can be determined and, therefore B bootstrap residual replicates of size n can be generated by sampling from $\hat{F}$. For the problem being studied, the obtained bootstrap sample is a set of residuals, $(r_1^{*b}, r_2^{*b}, \dots, r_i^{*b}, \dots, r_n^{*b})$, where $b = 1, 2, \dots, B$. Ideally, the number of bootstrap replications, i.e., B, takes the value $\rightarrow \infty$; however, the typical B is in the range of $20 \leq B \leq 200$ (Tibshirani, 1996). An examination of the

	1	2	3	4	5	6	7	8	9	10	11
	Periods	err.trn	resam.i								
1	1.00	0.40	3.56	-1.56	0.21	0.81	3.14	2.88	-3.64	2.59	-0.90
2	2.00	0.00	-2.82	3.33	-1.65	2.39	-4.97	-0.13	-7.07	1.63	0.04
3	3.00	2.20	0.35	-2.77	-1.59	2.10	1.23	1.98	-4.30	1.24	-1.13
4	4.00	-2.20	-3.68	-0.75	0.98	-2.23	1.98	2.28	-1.30	-4.20	-3.53
5	5.00	-3.80	0.65	-0.93	-0.32	-4.20	-2.31	0.44	0.61	3.61	-2.26
6	6.00	-2.00	-6.26	-0.61	-0.16	-0.93	4.88	-1.54	0.39	-2.94	-4.48
7	7.00	-2.90	-4.41	1.11	-3.82	-1.00	0.37	1.17	0.88	3.94	0.49
8	8.00	-2.50	0.44	1.26	-2.17	2.35	-0.60	0.06	0.26	2.67	2.74
9	9.00	2.30	0.92	-1.08	-1.71	4.24	-2.16	1.96	-4.44	-0.06	2.60
10	10.00	-2.40	-3.87	-1.74	-3.28	-1.63	-3.63	1.02	-0.41	0.86	-0.04
11	11.00	1.50	1.95	-2.01	0.14	3.54	0.28	2.03	2.70	-2.21	0.34
12	12.00	-1.80	0.03	1.38	-2.88	-2.88	1.08	-2.95	-1.77	-2.48	-0.46
13	13.00	1.40	4.82	-0.19	1.44	-2.71	-1.50	0.32	2.87	0.77	6.06
14	14.00	0.00	0.58	0.04	2.83	2.52	-0.05	5.20	-0.15	-1.42	0.99
15	15.00	6.00	-2.90	-4.67	-3.56	2.38	-1.50	-6.18	-1.79	1.90	-3.23
16	16.00	-1.10	1.24	0.87	-3.29	-1.08	4.05	3.51	3.44	0.87	0.36
17	17.00	4.30	-1.30	-1.65	2.31	-1.24	-0.67	-5.48	-2.35	-3.31	3.12
18	18.00	-2.40	-0.29	0.78	2.91	-4.84	-4.00	-0.54	1.81	1.67	5.24
19	19.00	3.10	-4.40	2.82	-4.15	-0.03	-0.47	1.62	0.16	0.86	2.55
20	20.00	-1.80	0.06	4.68	-2.21	-2.54	3.72	2.97	-4.01	-0.76	0.11
21	21.00	0.00	-2.47	-1.66	1.53	-0.87	3.47	1.10	-4.56	-3.30	-2.94
22	22.00	3.40	-3.25	0.87	-2.31	5.82	-0.88	2.19	1.88	1.37	-2.19

Figure 6.8 Bootstrap replicates of residuals for cutting tool training data

literature and research done in this area showed that even a not very large number of bootstrap replications, e.g., $B = 25$, could usually be informative (Efron and Tibshirani, 1993); this is also reasonable for complicated models such as the neural network or neuro-fuzzy system. We use two values of B for this problem, viz., $B = 20$ and 30 bootstrap replications. The output data frame of the cutting tool data for $B = 20$, implemented in S-PLUS environment with a spreadsheet format, is shown in Figure 6.8.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Time Period	Original	1	2	3	4	5	6	7	8	9	10	11	12	13
2	1	98	101.16	96.04	97.81	98.41	100.74	100.48	93.96	100.19	96.70	98.53	96.84	98.11	95.36
3	2	100	97.18	103.33	98.35	102.39	95.03	99.67	92.93	101.63	100.04	97.54	97.96	98.27	104.61
4	3	97	95.15	92.03	93.21	96.90	96.03	96.78	90.50	96.04	93.67	93.49	100.23	91.04	99.00
5	4	99	97.52	100.45	102.18	98.97	103.18	103.48	99.90	97.00	97.67	105.23	101.49	98.63	98.43
6	5	98	102.45	100.87	101.48	97.60	99.49	102.24	102.41	105.41	99.54	101.71	103.98	102.27	100.33
7	6	96	91.74	97.39	97.84	97.07	102.88	96.46	98.39	95.06	93.52	96.35	99.76	96.76	99.97
8	7	98	96.49	102.01	97.08	99.90	101.27	102.07	101.78	104.84	101.39	101.15	105.19	100.18	103.36
9	8	96	98.94	99.76	96.33	100.85	97.90	98.56	98.76	101.17	101.24	97.90	97.60	102.47	94.72
10	9	99	97.62	95.62	94.99	100.94	94.54	98.66	92.26	96.64	99.30	94.63	97.22	102.93	94.84
11	10	97	95.53	97.66	96.12	97.77	95.77	100.42	98.99	100.26	99.36	98.48	96.76	99.25	97.17
12	11	98	98.45	94.49	96.64	100.04	96.78	98.53	99.20	94.29	96.84	93.76	97.83	98.94	100.04
13	12	101	102.83	104.18	99.92	99.92	103.88	99.85	101.03	100.32	102.34	100.53	101.98	107.70	102.00
14	13	100	103.42	98.41	100.04	95.89	97.10	98.92	101.47	99.37	104.66	100.57	98.83	98.32	98.26
15	14	102	102.58	102.04	104.83	104.52	101.95	107.20	101.85	100.58	102.99	102.77	98.86	97.79	101.37
16	15	110	101.10	99.33	100.44	106.38	102.50	97.82	102.21	105.90	100.77	97.90	106.49	103.62	103.03
17	16	104	106.34	105.97	101.81	104.02	109.15	108.61	108.54	105.97	105.46	106.51	112.86	107.68	101.86
18	17	110	104.40	104.05	108.01	104.46	105.03	100.22	103.35	102.39	108.82	102.43	103.59	105.21	106.26
19	18	113	115.11	116.18	118.31	110.56	111.40	114.86	117.21	117.07	120.64	118.58	113.70	112.38	119.79
20	19	112	104.50	111.72	104.75	108.87	108.43	110.52	109.06	109.76	111.45	105.49	114.60	109.68	109.76
21	20	113	114.86	119.48	112.59	112.26	118.52	117.77	110.79	114.04	114.91	115.52	114.31	111.48	113.65
22	21	120	117.53	118.34	121.53	119.13	123.47	121.10	115.44	116.70	117.06	118.93	121.96	121.93	117.65
23	22	121	114.35	118.47	115.29	123.42	116.72	119.79	119.48	118.97	115.41	116.27	118.43	121.80	120.87
24	23	118	120.74	121.51	119.10	119.73	120.62	117.28	119.00	123.03	119.18	118.23	122.23	117.37	116.56
25	24	123	125.36	122.07	123.75	123.21	123.60	123.57	120.82	123.40	123.84	125.71	126.22	125.99	118.30
26	25	124	127.50	122.71	124.75	126.44	122.46	124.59	126.31	122.27	125.32	125.41	123.99	122.54	121.85

Figure 6.9 Bootstrap replicates of training set for cutting tool data

Column 1 indicates the time periods n , where $n = 50$ in this case; column 2 displays the original training error terms and each of the remaining columns represents a typical bootstrap sample of training residuals $(r_1^{*b}, r_2^{*b}, \dots, r_i^{*b}, \dots, r_{50}^{*b})$.

The corresponding bootstrap replicate for training data can thus be represented as:

$$(y_1^{*b}, y_2^{*b}, \dots, y_i^{*b}, \dots, y_{50}^{*b})$$

where $y_i^{*b} = \hat{y}_i^0 + r_i^{*b}$ is the i th element of the b th bootstrap replication and $\hat{y}_i^0$ ($i = 1, 2, \dots, 50$) is the i th predicted value generated from the original training data set. From the bootstrapped residuals of Figure 6.8 and the original predicted values in the training stage, the bootstrap replicates for training set of cutting tool data can be obtained using Excel spreadsheet, as shown in Figure 6.9. Investigation of the bootstrap replicates reveals that the resampling data in Figure 6.11 display trend similar to what can be observed from the original data set in Figure 6.10.

For each bootstrap replicate $(y_1^{*b}, y_2^{*b}, \dots, y_i^{*b}, \dots, y_{50}^{*b})$, $b = 1 \dots 20$, the same training procedure is applied to reevaluate the model parameters S and consequently, to generate the corresponding bootstrap future values $\hat{y}_i^b$ ($i = 51, 52, \dots, 100$). For each period i , the total number of forecasts is the number of bootstrapping replicates B . Differences among the forecasts $\hat{y}_i^b$ ($b = 1, 2, \dots, 20$) exist due to the variability of parameters estimation.

Forecast error is defined as

$$e_i = y_i - \hat{y}_i \quad 6.1$$

where y_i is the actual value and $\hat{y}_i$ is the predicted value.

Figure 6.10 Original training set for cutting tool data

Figure 6.11 One bootstrap replicate of training set for cutting tool data

Assuming the model specification is appropriate, equation (6.1) states two sources of uncertainty in a forecasting system. They are: (1) random error variation in the process, and (2) variability due to parameter estimation. These two sources of variation, which for simplicity are assumed to be independent of each other, contribute to the forecast error variance, σ^2 , that can be expressed as:

$$\sigma^2 = \sigma_{\varepsilon}^2 + \sigma_s^2 \quad 6.2$$

More specifically, σ_{ε}^2 describes the uncertainty related to the random variation in the process and σ_s^2 represents the uncertainty about the estimation of the model parameters. The examination of training residuals in Section 6.1 reveals that, the random variation of the process follows normal distribution with mean 0 and variance σ_{ε}^2 , wherein the variance is assumed to be constant. The uncertainty associated with the model parameters σ_s^2 can be estimated by bootstrap predicted values and their mean through

$$\sigma_s^2 = \frac{1}{B} \sum_0^B [y(x_i; \hat{S}^{*b}) - \bar{y}(x_i)]^2 \quad 6.3$$

$$\text{where } \bar{y}(x_i) = \sum_0^B y(x_i; \hat{S}^{*b}) / (B + 1);$$

Therefore, the forecast error variance can be estimated as

$$\sigma^2 = \sigma_{\varepsilon}^2 + \frac{1}{B} \sum_0^B [y(x_i; \hat{S}^{*b}) - \bar{y}(x_i)]^2 \quad 6.4$$

6.3 Prediction Intervals

Having developed the expressions for forecast error variance, we may now turn to the problem of using this information to make probability statements about forecast errors. Since the training residuals have been found to be normally distributed for the underlying model, we assume that the errors from the forecasting process also follow normal distribution. Without loss of generality, we use Z_α to represent a value of the standard normal random variable such that

$$\int_{-\infty}^{Z_\alpha} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2} dt = \alpha \quad 6.5$$

where Z_α denotes the 100α percentile of the standard normal distribution. The prediction interval would contain the actual value in period i with probability $100(1 - \alpha)\%$ can be stated as:

$$P\left\{Z_{\alpha/2}\sigma < y_i - \hat{y}_i < -Z_{\alpha/2}\sigma\right\} = 1 - \alpha \quad 6.6$$

Thus, the $100(1 - \alpha)\%$ prediction interval for y_i can be given by:

$$\hat{y}_i \pm Z_{\alpha/2}\sigma \quad 6.7$$

which after substituting for σ , we get:

$$\hat{y}_i \pm Z_{\alpha/2} \sqrt{\frac{1}{B} \sum_0^B [y(x_i; \hat{S}^{*b}) - \bar{y}(x_i)]^2 + \sigma_\varepsilon^2} \quad 6.8$$

Equation (6.8) is valid when $n \rightarrow \infty$, but is only an approximation for finite samples. When n is finite, the better approximation is the Student's t distribution. By using this approximation, the prediction interval can be calculated as:

$$\hat{y}_i \pm t_{\alpha/2, n-1} \sqrt{\frac{1}{B} \sum_0^B [y(x_i; \hat{S}^{*b}) - \bar{y}(x_i)]^2 + \sigma_\varepsilon^2} \quad 6.9$$

where $t_{\alpha/2, n-1}$ denotes the $\alpha/2^{\text{th}}$ percentile of the t distribution with $n - 1$ degrees of freedom. Note that failure to include the process variation σ_ε^2 will result in a $100(1 - \alpha)\%$ interval for the mean of the conditional prediction y_i , not an interval for a particular future observation on y_i .

Good prediction intervals are those upper and lower limits which contain the actual observations in a relatively tight band for a given probability; a wide prediction interval is usually of little practical use. In the next chapter, we will use equation (6.9) to calculate the prediction intervals of the three data sets being studied, examine different cases for bootstrapping size. We will also show the forecast results from ARIMA model and give the numerical comparison of these two methods.

Chapter 7

Computational Experience

For comparison purposes, we first apply ARIMA models to the three cases being studied and investigate the forecasts and prediction intervals using SAS Time Series Forecasting System. We will then use the ANFIS model with our proposed structure, procedures and methodologies to generate forecasts, monitor process and construct prediction intervals for the same data sets.

7.1 Using ARIMA Model

When successive observations are not statistically independent, a time series can be represented in the form of a linear filter model as

$$x_t = \mu + \psi_0 \varepsilon_t + \psi_1 \varepsilon_{t-1} + \psi_2 \varepsilon_{t-2} + \cdots \quad 7.1$$

Also, time series forecasting involves the use of past values and past errors to predict the future, therefore, a time series model which takes advantage of the dependency in the observations is superior to the models which do not. In practice, many different time series models could be generated for the linear filter model. This category of statistical forecasting models were unified and presented by Box and Jenkins (1976), and are usually called the ARIMA model.

ARIMA, which stands for AutoRegressive Integrated Moving Average, is a general model capable of representing a wide class of nonstationary time series. It is

usually denoted as ARIMA (p, d, q), indicating that the model represents the dth difference of the original series as a process containing p autoregressive and q moving average parameters. A typical example is ARIMA (1, 1, 1), which can be expressed as

$$x_t = (1 + \phi_1)x_{t-1} - x_{t-2} + \varepsilon_t - \theta_1\varepsilon_{t-1} \quad 7.2$$

where ϕ_1 is autoregressive parameter, θ_1 is moving average parameter, ε_t and ε_{t-1} are error terms.

7.1.1 Case I

Building an ARIMA model involves the steps of identifying model, estimating parameters, and making predictions. Statistical tools such as autocorrelation function, partial autocorrelation function, white noise tests, unit root tests, etc., could be used to analyze historical data in the identification process. Autocorrelation function is the primary tool used, as it can provide a good indication whether the time series is stationary or not. If the time series is nonstationary, the autocorrelation function will die down very slowly since the observations will tend to be on the same side of the sample mean for many periods, therefore produce long lags. Figure 7.1 shows the autocorrelations of the cutting tool data, which indicates that the time series is nonstationary. Moreover, the unit root test in Figure 7.2 also confirms the nonstationarity. The usual approach to dealing with nonstationarity is to take the difference of the series. Figure 7.3 shows the plots after the first difference of the series is taken. Figure 7.4 depicts the autocorrelations of the first difference and Figure 7.5 gives the white noise and stationary tests, which both indicate that the stationary behavior is achieved.

Figure 7.1 Autocorrelation plots for cutting tool data

Figure 7.2 White noise and stationarity test for cutting tool data

Figure 7.3 Data series after the first difference taken for cutting tool data

Figure 7.4 Autocorrelations for the first difference of the cutting tool data

Figure 7.5 White noise and stationarity test for the first difference of the cutting tool data

Having determined that only the first difference needs to be taken for the time series to become stationary, the parameter d of the ARIMA (p, d, q) process is thus set equal to 1. Also, the autocorrelation plots for the first difference series in Figure 7.4 show that, the autocorrelation function tails off while the partial autocorrelation function cuts

off after two lags. This is the indicative of AR (2) process, i.e., the first difference of the original data series is a second-order autoregressive process and the parameter q of the MA process is zero. The ARIMA predictive model is thus identified as the process of ARIMA (2, 1, 0)

$$(1 - \phi_1 B^1 - \phi_2 B^2) \nabla x_t = \varepsilon_t \quad 7.3$$

which can also be written as

$$x_t = x_{t-1} + \phi_1 (x_{t-1} - x_{t-2}) + \phi_2 (x_{t-2} - x_{t-3}) + \varepsilon_t \quad 7.4$$

$$\text{and } x_t = (1 + \phi_1)x_{t-1} + (\phi_2 - \phi_1)x_{t-2} - \phi_2 x_{t-3} + \varepsilon_t \quad 7.5$$

Note that x_t is the predicted value while x_{t-1} , x_{t-2} , x_{t-3} are the actual observations.

Model (7.5) states that the forecasts are based on previous three consecutive periods values and the current error.

To estimate the unknown parameters, the data set is divided into two halves, the first 50 observations are used as fitting data set and the remaining 50 observations are used as the evaluation data set. With this splitting, the model parameters estimated by SAS are:

$$\phi_1 = -0.44555$$

$$\phi_2 = -0.15389$$

and the model variance is 12.1670 (sigma squared). The mean squared error (MSE) is chosen as the evaluation criterion for the model, which has the value of 9.668. The forecasts vs. the actual values are plotted in Figure 7.6 and the prediction intervals are depicted in Figure 7.7. The residuals plot is graphed in Figure 7.8.

Figure 7.6 One-step-ahead forecast using ARIMA (2, 1, 0) (cutting tool data)

Figure 7.7 Prediction intervals of one-step-ahead forecast using ARIMA (2, 1, 0) (cutting tool data)

Figure 7.8 Residuals of one-step-ahead forecast using ARIMA (2, 1, 0)
(cutting tool data)

Predictions generated by the SAS forecasting procedure are one-step-ahead forecasts, namely, given the observations up to period t , predict the value for next period $(t+1)$. However, in many cases, we may be interested in finding multiple-step-ahead forecasts, in other words, knowing the information up to period t , make prediction for period $(t+\tau)$ where $\tau > 1$. In general, multiple-step-ahead forecast is less accurate than one-step-ahead forecast, as it shows more uncertainty about the predictions when time lengthens.

In statistical models, forecast for period $(t+\tau)$ is built up successively from the forecasts for periods t , $(t+1)$, $(t+2)$, ... $(t+\tau-1)$. If the actual value of period $(t+j)$ has not occurred at the time of generating forecasts, it will be replaced by the forecast value for that period; if the error term for period $(t+j)$ has not occurred at time, it will be replaced

by zero. Therefore, given that x_t is the actual value observed, the three-step-ahead forecast generating at time t using ARIMA (2,1,0) model would be:

$$\hat{x}_{t+3} = (1 + \phi_1) \hat{x}_{t+2} + (\phi_2 - \phi_1) \hat{x}_{t+1} - \phi_2 x_t \quad 7.6$$

Note that, $\hat{x}_{t+2}$ is a two-step-ahead forecast generated at time t as

$$\hat{x}_{t+2} = (1 + \phi_1) \hat{x}_{t+1} + (\phi_2 - \phi_1) x_t - \phi_2 x_{t-1} \quad 7.7$$

and $\hat{x}_{t+1}$ is a one-step-ahead forecast generated at time t with the value of

$$\hat{x}_{t+1} = (1 + \phi_1) x_t + (\phi_2 - \phi_1) x_{t-1} - \phi_2 x_{t-2} \quad 7.8$$

The forecasts starting from period 51 are obtained from the above models and are shown in Figure 7.9. The residuals are plotted in Figure 7.10.

Figure 7.9 Three-step-ahead forecast using ARIMA (2, 1, 0) (cutting tool data)

Figure 7.10 Residuals of three-step-ahead forecast using ARIMA (2, 1, 0) (cutting tool data)

7.1.2 Case II

Same identification process and estimation procedure could be used for constructing an ARIMA model for the S&P 500 index data. The autocorrelations plot in Figure 7.11 reveals that the series exhibits nonstationarity; whereas the stationarity can be achieved by taking the first difference for the original data as shown in Figure 7.12.

Likewise, the predictive model is formed as ARIMA (2, 1, 0). The first 194 observations are used for fitting the model, while the rest are set aside for evaluation. The model parameters are estimated as:

$$\phi_1 = 0.19252$$

$$\phi_2 = -0.10752$$

Figure 7.11 Autocorrelation plots for S&P 500 index data

Figure 7.12 Data series after the first difference taken for S&P 500 index data

The model variance given by SAS is 0.01396 (sigma squared). The mean squared error criterion is used for model evaluation, which returns the value of 0.00394. The one-step-ahead forecasts with the prediction intervals from the SAS forecasting system can be shown in Figure 7.13. The residuals are plotted in Figure 7.14.

Figure 7.13 One-step-ahead forecast and the prediction intervals using ARIMA (2, 1, 0) (S&P 500 index data)

Figure 7.14 Residuals of one-step-ahead forecast using ARIMA (2, 1, 0) (S&P 500 index data)

The three-step-ahead forecasts for S&P 500 index data can also be obtained by recursively using model 7.6, model 7.7, and model 7.8, as shown in Figure 7.15. The forecast residuals are plotted in Figure 7.16.

Figure 7.15 Three-step-ahead forecast using ARIMA (2, 1, 0) (S&P 500 index data)

Figure 7.16 Residuals of three-step-ahead forecast using ARIMA (2, 1, 0) (S&P 500 index data)

7.1.3 Case III

For the beer production data, the first 77 observations are used to identify the model while the remaining data are used for evaluation. From SAS forecasting system, it was determined that the ARIMA (2, 1, 2), i.e., a second-order autoregressive and second-order moving average process of the first difference of the original data series, outperforms other models examined. The model can be expressed as:

$$x_t = (1 + \phi_1)x_{t-1} + (\phi_2 - \phi_1)x_{t-2} - \phi_2x_{t-3} + \varepsilon_t - \theta_1\varepsilon_{t-1} - \theta_2\varepsilon_{t-2} \quad 7.9$$

In other words, this model uses three previous values and three previous errors to generate forecast; however, it does not consider the seasonal effect.

With the mean squared error (MSE) being used as the model evaluation criterion, the model parameters are estimated as:

$$\phi_1 = 0.98463 \quad \theta_1 = -0.01158$$

$$\phi_2 = -0.84452 \quad \theta_2 = -0.93268$$

The one-step-ahead forecasts with the prediction intervals are shown in Figure 7.17 and the residuals plot is graph in Figure 7.18. The MSE value is computed as 1,314.

The four-step-ahead forecast model could be developed sequentially based on model in the equation (7.9) as below:

$$\hat{x}_{t+4} = (1 + \phi_1)\hat{x}_{t+3} + (\phi_2 - \phi_1)\hat{x}_{t+2} - \phi_2\hat{x}_{t+1} + \varepsilon_{t+4} - \theta_1\varepsilon_{t+3} - \theta_2\varepsilon_{t+2} \quad 7.10$$

where

$$\hat{x}_{t+3} = (1 + \phi_1)\hat{x}_{t+2} + (\phi_2 - \phi_1)\hat{x}_{t+1} - \phi_2x_t + \varepsilon_{t+3} - \theta_1\varepsilon_{t+2} - \theta_2\varepsilon_{t+1} \quad 7.11$$

$$\hat{x}_{t+2} = (1 + \phi_1)\hat{x}_{t+1} + (\phi_2 - \phi_1)x_t - \phi_2x_{t-1} + \varepsilon_{t+2} - \theta_1\varepsilon_{t+1} - \theta_2\varepsilon_t \quad 7.12$$

Figure 7.17 One-step-ahead forecast and the prediction intervals using ARIMA (2, 1, 2) (beer production data)

Figure 7.18 Residuals of one-step-ahead forecast using ARIMA (2, 1, 2) (beer production data)

$$\hat{x}_{t+1} = (1 + \varphi_1)x_t + (\varphi_2 - \varphi_1)x_{t-1} - \varphi_2x_{t-2} + \varepsilon_{t+1} - \theta_1\varepsilon_t - \theta_2\varepsilon_{t-1} \quad 7.13$$

The forecasts generated from the above models are shown in Figure 7.19, below. The residuals are plotted in Figure 7.20.

Figure 7.19 Four-step-ahead forecast using ARIMA (2, 1, 2) (beer production data)

Figure 7.20 Residuals of four-step-ahead forecast using ARIMA (2, 1, 2) (beer production data)

7.2 Using ANFIS Model

Having proposed the forecasting system using ANFIS model, the procedure of monitoring the process and the methodology of building the prediction intervals, we will now apply them to the two time series being studied. Further, we adopt the notation ANFIS (p, q, r) to represent the structure of ANFIS for time series forecasting, where p is the number of previous periods, q is the number of membership functions, and r is the periods-ahead to be forecasted.

Unlike the ARIMA model, ANFIS forecasting does not attempt to examine the original series to find any indication of trends or seasonality, etc.; instead, the historical data are fed into the model without any prior information and assumption. Also, multiple-step-ahead forecasts are generated by specifying the training parameter which determines the training data partitioning, without depending on preceding forecasts.

To examine the outputs from the ANFIS forecasting system, we first implement the one-step-ahead forecasts; and then generate the three-step-ahead forecasts and estimate the forecast error variances in order to build prediction intervals, and finally show the prediction intervals constructed.

7.2.1 Case I

One-step-ahead forecasts for cutting tool data are depicted below in Figure 7.21 and the residual plot is given in Figure 7.22. The sum of squared errors (SSE) is computed as 867.96, which is larger than the computed SSE for the ARIMA model in this case.

Figure 7.21 One-step-ahead forecast using ANFIS (3, 2, 1)
(cutting tool data)

Figure 7.22 Residuals of one-step-ahead forecast using ANFIS (3, 2, 1)
(cutting tool data)

Figure 7.23 Three-step-ahead forecast using ANFIS (3, 2, 3) (cutting tool data)

The three-step-ahead forecasts are generated likewise but with different input parameter setting. The graphical output of forecasts is presented in Figure 7.23. Prediction intervals for the forecasts are obtained through the process of forecast error variance estimation, wherein bootstrapping approach is used. Two different bootstrapping size, i.e., $B = 20, 30$, have been tested respectively for this data set.

Experience has shown that as the number of bootstrapping replications B increase, the estimated error variance in general gets smaller. However, the number of bootstrapping replications $B = 20$ is usually informative and gives a good estimate of the forecast error variance. Figure 7.24 shows the graph of the 95% prediction intervals for the forecasts calculated from formula (6.9) when the number of bootstrapping replications $B = 20$, and the process variation is assumed to be the same as the training

stage, which is, $\sigma_\varepsilon^2 = 6.45$. Similarly, Figure 7.25 provides the 95% prediction intervals for the forecasts when $B = 30$.

Figure 7.24 Prediction intervals for three-step-ahead forecast using ANFIS (3, 2, 3) when $B = 20$, $\alpha = 5\%$ (cutting tool data)

Figure 7.25 Prediction intervals for three-step-ahead forecast using ANFIS (3, 2, 3) when $B = 30$, $\alpha = 5\%$ (cutting tool data)

One thing worth noticing is that, in general, forecast errors in testing stage tend to be larger than the training errors; this means that the random variation in testing stage would be underestimated when their actual values turned out to be large, therefore, the prediction intervals might be narrower for some periods. This can be observed from the graphs in the previous page. It shows, under the stated probability (95%), 1 to 2 upper limits are underestimated.

The numerical results of three-step-ahead forecast are given in Appendix V. The residual plot is shown in Figure 7.26. The sum of squared errors is 891.52, which appears very close to the results of one-step-ahead forecast where $SSE = 867.96$.

Figure 7.26 Residuals of three-step-ahead forecast using ANFIS (3, 2, 3) (cutting tool data)

7.2.2 Case II

For S&P 500 index data, the one-step-ahead forecast is examined similarly. The forecasts vs. the actual values are graphed in Figure 7.27 and the residuals are plotted in

Figure 7.28. The sum of squared errors is calculated as 0.882, which does not outperform the results from the ARIMA (2, 1, 0) model.

The three-step-ahead forecast is also investigated; the results are shown in Appendix VI. The forecasts plotted in Figure 7.29 are obviously less accurate than the one-step-ahead forecast. Figure 7.30 also verifies this and indicates that the residuals have become larger. To build the prediction intervals for the forecasts, two different bootstrapping size, i.e., $B = 20, 30$, were tested. The 95% prediction intervals for the forecasts when $B = 20$ are displayed in Figure 7.31 and Figure 7.32 shows the 95% prediction intervals for the forecasts when $B = 30$. The results confirm that, the number of bootstrapping replications $B = 20$ is informative.

Figure 7.27 One-step-ahead forecast using ANFIS (3, 2, 1)
(S&P 500 index data)

Figure 7.28 Residuals of one-step-ahead forecast using ANFIS (3, 2, 1)
(S&P 500 index data)

Figure 7.29 Three-step-ahead forecast using ANFIS (3, 2, 3)
(S&P 500 index data)

Figure 7.30 Residuals of three-step-ahead forecast using ANFIS (3, 2, 3)
(S&P 500 index data)

Figure 7.31 Prediction intervals for three-step-ahead forecast using ANFIS (3, 2, 3)
when $B = 20$, $\alpha = 5\%$ (S&P 500 index data)

Figure 7.32 Prediction intervals for three-step-ahead forecast using ANFIS (3, 2, 3) when $B = 30$, $\alpha = 5\%$ (S&P 500 index data)

7.2.3 Case III

For beer production data, the one-step-ahead forecasts are generated likewise. Figure 7.33 shows the forecasts along with the actual values and Figure 7.34 depicts the residuals plot. The mean squared error statistics indicates that the results are less accurate than the single-step-ahead forecasts from the ARIMA (2, 1, 2) model.

The four-step-ahead forecast has been examined for this data case, where the results are given in Appendix VII. Because of the seasonal effect, it can be easily observed that the four-step-ahead forecast performs much better than the one-step-ahead forecast, as shown in Figure 7.35. The residuals are plotted in Figure 7.36. The 95%

prediction intervals with two different bootstrapping size, viz., $B = 20$ and 30 , have been constructed, as displayed in Figure 7.37 and Figure 7.38, respectively.

Figure 7.33 One-step-ahead forecast using ANFIS (3, 2, 1)
(beer production data)

Figure 7.34 Residuals of one-step-ahead forecast using ANFIS (3, 2, 1)
(beer production data)

Figure 7.35 Four-step-ahead forecast using ANFIS (3, 2, 4)
(beer production data)

Figure 7.36 Residuals of Four-step-ahead forecast using ANFIS (3, 2, 4)
(beer production data)

Figure 7.37 Prediction intervals for Four-step-ahead forecast using ANFIS (3, 2, 4) when $B = 20$, $\alpha = 5\%$ (beer production data)

Figure 7.38 Prediction intervals for Four-step-ahead forecast using ANFIS (3, 2, 4) when $B = 30$, $\alpha = 5\%$ (beer production data)

7.3 Comparison of Forecasting Performance

Although there are many criteria that can be used to evaluate a forecasting method, the accuracy of forecasts is usually the basis for selecting appropriate method. Error measures for forecasting accuracy have been studied by many researchers for investigating the appropriateness of various forecasting methods (Makrodakis et al., 1998; Armstrong and Collopy, 1992). Choice of the best performance measure for the calculation of forecasting accuracy is still an open question; however, it is widely accepted that no single accuracy measure can capture all the differences among various methods.

In view of this, three performance measures are examined in this research. They are: Mean Squared Error (MSE), Mean Absolute Percentage Error (MAPE) and Theil's-U statistic. For n periods of forecasts, these measures can be expressed as:

$$MSE = \frac{\sum_{i=1}^n (X_i - F_i)^2}{n} \quad 7.14$$

$$MAPE = \frac{\sum_{i=1}^n \frac{|X_i - F_i|}{X_i} * 100}{n} \quad 7.15$$

$$\text{Theil's U - statistic} = \sqrt{\frac{\sum_{i=1}^{n-1} \left(\frac{F_{i+\tau} - X_{i+\tau}}{X_i} \right)^2}{\sum_{i=1}^{n-1} \left(\frac{X_{i+\tau} - X_i}{X_i} \right)^2}} \quad 7.16$$

where X_i = actual observation for time period i

F_i = forecast value for time period i

$F_{i+\tau}$ = forecast generated for time period $i + \tau$ at period i

τ = time periods ahead

We also need to consider the forecasting lead time when comparing various models studied. In this research, we investigate single period ahead forecasts as well as multiple periods ahead forecasts, i.e., three-step-ahead and four-step-ahead forecasts are examined using the above three accuracy measures. The results for each case are summarized in Table 7.1 through Table 7.4, as listed below.

Table 7.1 Performance of the ARIMA model for Case I

	MSE	MAPE (%)	Theil's U-statistic
One-step-ahead	9.668	1.843	0.988
Three-step-ahead	17.962	2.722	0.987
Four-step-ahead	22.435	3.055	0.985

Table 7.2 Performance of the ARIMA model for Case II

	MSE	MAPE (%)	Theil's U-statistic
One-step-ahead	0.004	4.282	0.956
Three-step-ahead	0.019	10.034	0.994
Four-step-ahead	0.026	11.665	1.120

Table 7.3 Performance of the ARIMA model for Case III

	MSE	MAPE (%)	Theil's U-statistic
One-step-ahead	1314.03	6.05	0.475
Three-step-ahead	23220.6	30.63	2.151
Four-step-ahead	26033.7	27.53	7.385

Table 7.4 Performance of the ANFIS model for Case I

	MSE	MAPE (%)	Theil's U-statistic
One-step-ahead	17.359	2.496	1.319
Three-step-ahead	17.086	2.630	0.960
Four-step-ahead	16.757	2.504	0.843

Table 7.5 Performance of the ANFIS model for Case II

	MSE	MAPE (%)	Theil's U-statistic
One-step-ahead	0.004	4.578	1.024
Three-step-ahead	0.023	10.793	1.073
Four-step-ahead	0.029	12.095	1.090

Table 7.6 Performance of the ANFIS model for Case III

	MSE	MAPE (%)	Theil's U-statistic
One-step-ahead	1889.27	7.05	0.584
Three-step-ahead	2683.36	9.12	0.686
Four-step-ahead	929.39	4.96	1.430

It can be observed that ARIMA model performs well for single period ahead forecasting when appropriate model parameters have been determined. For multiple periods ahead forecasting, since future forecasts are obtained from previous forecasts, the accuracy of a forecast depends on the performance of the previous forecasts. When the previous forecasts do not perform well, the errors for further forecasts may get very large. In neuro-fuzzy ANFIS model due to the fact that learning and inferencing are based on different segments of data, which are partitioned according to the requirements of the number of time periods ahead, the forecasting system performs well, especially for long term horizon.

Chapter 8

Conclusions and Future Research Recommendations

Time series analysis is a very important and widely used method in forecasting field. The goal of time series forecasting is to discover patterns in the historical data and extrapolate that pattern into the future. Given the fact that artificial intelligence tools such as neural network, fuzzy logic, are capable of learning and inferencing from the past to capture the patterns that exist in the data, this research addresses intelligent time series forecasting where forecasts are generated from a neuro-fuzzy structure. However, this study was mainly focused on combining statistical techniques with neuro-fuzzy methods to study the time series forecasting problem comprehensively.

8.1 Summary of Research

In view of the literature on time series forecasting using artificial intelligent methods, the objectives of this research were outlined as: to use a neuro-fuzzy paradigm ANFIS model to implement an intelligent time series forecasting; to monitor the forecasting process by tracking signals test; to estimate forecast error variance and build prediction intervals for the forecasts. By accomplishing the above, we have been assessed uncertainties associated with a time series forecasting problem thoroughly from the model building to the analysis of the results.

The main focus of this research was nonstationary, univariate time series. An ANFIS forecasting structure was built as a three-input, one-output system with two Gaussian membership functions for each input and first-order Sugeno fuzzy model used for generating the fuzzy rules. No assumptions were made for model building and forecast generating. Three example data sets were investigated. The first data set was weekly sales of a cutting tool with 100 observations, the second data set was the quarterly S&P 500 index with 388 observations, and the third data set was the quarterly beer production in Australia with 154 observations. Each data set was split into a training set and a testing set, where training data are used for identifying two sets of model parameters, namely, the parameters for Sugeno fuzzy models and the parameters for the membership functions. Single-step-ahead forecast along with multiple-step-ahead forecast can be obtained by specifying the input parameter.

Trig's smoothed error tracking signals test was used for monitoring the forecasting process and as a statistical measure for keeping the forecasting model up-to-date. The results showed that, tracking signals test is an effective way to detect any nonrandom change in the existing pattern and serves as a quality control procedure for the forecasting process. However, the selection of the smoothing parameters and the control limit is data-specific.

The uncertainty associated with the forecasts can be quantified by the forecast error variance, and is measured by prediction intervals. Considering the complexity of the model structure and also the procedures for identification of the model parameters, bootstrapping resampling technique was used to estimate the forecast error variance in

this research. More specifically, we used bootstrapping residuals approach by first analyzing the training residuals and deriving their empirical distribution, then obtained the bootstrapped training set, reevaluated the model parameters, and regenerated forecasts. Different number of bootstrapping replications B were tested, the results revealed that $B = 20$ is usually informative. Another finding is that when training errors are relatively smaller than the variation in testing errors, the prediction intervals appear to be narrower for some periods.

ARIMA model was also investigated for the purpose of comparison with the ANFIS structure. Forecasts were generated using the SAS time series forecasting procedure for the ARIMA models. It was observed that, for one-step-ahead forecast, ARIMA model performs better if the model parameters were specified appropriately. ANFIS model in general performs well; it is better suited for use in multiple-step-ahead forecast generating, as the current forecast does not count on its previous forecasting values.

8.2 Contributions of Research

The extensive literature review showed that although some research have been conducted on time series forecasting using neural network, fuzzy logic, and neuro-fuzzy structure, this dissertation addressed the problem of monitoring forecasting process for the first time. Also, the derivation of forecast error variance and the construction of prediction intervals for neuro-fuzzy models is a contribution of the present research. The

use of tracking signals test and bootstrapping technique applied in this research provided a cutting edge tool in the intelligent forecasting field.

8.3 Recommendations for Future Research

Future research work could be conducted in the following areas:

- Data might need to be pre-processed before being fed into the network

For the time series being studied, data used for training were purely raw data. Future research could investigate if data pre-processing, such as taking difference of original data series, deseasonalizing the original series, etc., would improve the system performance.

- Multivariate time series need to be studied

This research only examined univariate time series. Future work could explore multivariate time series where one time series depends on some other time series.

- Procedures for setting the parameters from statistical point of view need to be more researched

The setting of parameters influences the forecasting performance, and therefore needs further research.

Bibliography

1. Amara, R. and Salanik, G. (1972). Forecasting: From conjectural art toward science. *Technological Forecasting and Social Change*, 3(3): 415-426.
2. Armstrong, J. S. (2000). The forecasting dictionary. Full text at <http://www-marketing.wharton.upenn.edu/forecast/dictionary.pdf>.
3. Armstrong, J. S. and Collopy, F. (1992). Error measures for generalizing about forecasting methods: Empirical comparisons. *International Journal of Forecasting*, 8(1): 69-80.
4. Ayers, R. (1969). *Technological forecasting and long-range planning*. McGraw-Hill Book Company, New York.
5. Balkin, D. S. and Ord, J. K. (2000). Automatic neural network modeling for univariate time series. *International Journal of Forecasting*, 16: 509-515.
6. Bartkiewicz, W. (2000). Confidence intervals prediction for the short-term electrical load neural forecasting models. 8-12.
7. Berry, M. and Linoff, G. (1997). *Data Mining Techniques*. John Wiley & Sons, Inc.
8. Box, G. and Jenkins, G. (1976). *Time Series Analysis, Forecasting and Control*, Revised Edition, Holden-Day, San Francisco, CA.
9. Brown, G., (1959). *Statistical Forecasting for Inventory Control*. McGraw-Hill, New York.

10. Chatfield, C. (1993). Calculating interval forecasts. *Journal of Business & Economic Statistics*, 11(2): 121-135.
11. Chatfield, C. (1995). Model uncertainty, data mining and statistical inference. *Journal of Royal Statistical Society*. 158 (3): 419-466.
12. Chernick, M. (1999). *Bootstrap Methods: A Practitioner's Guide*. John Wiley & Sons, Inc.
13. Clements, M. and Taylor, N. (2001). Bootstrapping prediction intervals for autoregressive models. *International Journal of Forecasting*, 17: 247-267.
14. Deboeck, G. J. (1994). *Trading on the Edge: Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets*. John Wiley & Sons, Inc.
15. Devika, P. and Achenie, L. (1994). Experience with using a neural network for forecasting time series. *World Congress on Neural Networks-San Diego. 1994 International Neural Network Society Annual Meeting*. Vol. 2: 315-320.
16. Dougherty, M. and Cobbett, M. (1997). Short-term inter-urban traffic forecasts using neural network. *International Journal of Forecasting*, 13: 21-31.
17. Dubois, D. and Prade, H. (1980). *Fuzzy sets and systems: theory and applications*. Academic press, New York.
18. Dubois, D. and Prade, H. (1991). Fuzzy sets in approximate reasoning. *Fuzzy sets and systems*, 40: 143-202
19. Efron, B. and Tibshirani, R. (1986). Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Statistical Science*. 1(1): 54-77.

20. Efron, B. and Tibshirani, R. (1993). An Introduction to the Bootstrap. Chapman & Hall.
21. Efron, B., (1979). Bootstrap methods: another look at the jackknife. *Annals of Statistics*, 7: 1-26.
22. Efron, B., (1979). Computers and the theory of statistics: thinking the unthinkable. *SIAM Rev.* 21: 460-480.
23. Fiordaliso, A. (1998). A nonlinear forecasts combination method based on Takagi-Sugeno fuzzy systems. *International journal of forecasting*, 14:367-379.
24. Fodor, J. and Roubens, M. (1994). Fuzzy preference modeling and multicriteria decision support, Kluwer Academic Publishers, Boston.
25. Foster, W. Collopy, F. and Ungar, L. (1992). Neural network forecasting of short, noisy time series. *Computers & Chemical Engineering*. Vol. 16 (4): 293-297.
26. Gardner, S., (1983). Automatic monitoring of forecast errors. *Journal of Forecasting*, 2(1): 1-22.
27. Gardner, S., (1988), A simple method of computing prediction intervals for time-series forecasts. *Management Sciences*, 34: 541-546.
28. Gorr, W., (1994). Research prospective on neural network forecasting. Neural networks in forecasting: Special section. *International Journal of Forecasting*. 10: 1-4.
29. Harrison, P. J. and Davies, L. O. (1963). The use of cumulative sum (CUSUM) techniques for the control of routine forecasts of product demand. *Operations Research*, 12: 325-333.

30. Hecht-Nielsen, R. (1990). Neurocomputing. Addison-Wesley.
31. Heskes, T. (1997). Practical confidence and prediction intervals. In Mozer, M., Jordan, M. and Petsche, T. editors, Advances in Neural Information Processing Systems 9, 466-472. Cambridge, MIT Press.
32. Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. In proceedings of the National Academy of Science, 2254-2258, USA.
33. Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. In proceedings of the National Academy of Science, 3088-3092, USA.
34. Hyndman, R. (2002). Time Series Data Library website at: <http://www-personal.buseco.monash.edu.au/~hyndman/TSDL/index.htm>.
35. Jain, L.C. and Martin, N. M. (1999). Fusion of Neural Networks, Fuzzy Sets, and Genetic Algorithms: Industrial Applications. CRC Press.
36. Jang, J. -S. R, Sun, C. T. and Mizutani, E. (1997). Neuro-Fuzzy and Soft Computing. Prentice Hall.
37. Jang, J. -S. R. (1992). Neuro-Fuzzy Modeling: Architecture, Analyses and Applications. Ph.D. dissertation, Department of Electrical Engineering and Computer Science, University of California, Berkeley.
38. Jang, J. -S. R. (1993). ANFIS: Adaptive-network-based fuzzy inference systems. IEEE Transactions on Systems, Man, and Cybernetics, Vol. 23: 665-685.

39. Jang, J. -S. R. (1996). Input selection for ANFIS learning. In proceedings of 5th IEEE International Conference on Fuzzy Systems. 1493-1499, New Orleans.
40. Kim, I. and Lee, S. R. (1999). A fuzzy time series prediction method based on consecutive values. In proceedings of IEEE International Fuzzy Systems, FUZZ-IEEE'99, Vol. 2: 703-707.
41. Klinkhachorn, P. (1999). CpE 321: Applied Fuzzy Logic. Class Slides at: <http://www.csee.wvu.edu/classes/cpe321/>, West Virginia University.
42. Klinkhachorn, P. (2000). CpE 320: Application of Neural Networks. Class Slides at: <http://www.csee.wvu.edu/classes/cpe320/>, West Virginia University.
43. Koehler, A., Snyder, R., and Ord, J. (2001). Forecasting models and prediction intervals for the multiplicative Holt-Winters method. *International Journal of Forecasting*, 17: 269-286.
44. Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 66:59-69.
45. Kohonen, T. (1984). Self-organization and associate memory. Springer-Verlag, London, 3rd edition.
46. Li, X., Ang, C. and Gray, R. (1999). An intelligent business forecaster for strategic business planning. *Journal of forecasting*, 18:181-204.
47. Lippmann, R. P. (1987). An Introduction to Computing with Neural Nets. *Artificial Neural Networks: Theoretical Concepts*. Computer Society Press of the IEEE.

48. Makridakis, S., and Wheelwright, S. (1989). *Forecasting Methods for Management*. John Wiley & Sons, Inc.
49. Makridakis, S., Wheelwright, S. and Hyndman, R. (1998). *Forecasting: Methods and Applications*. John Wiley & Sons, Inc, 3rd edition.
50. Mamdani, E. H. and Assillian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7(1): 1-13.
51. Martino, J. (1983). *Technological forecasting for decision making*. Elsevier Science Publishing Company, New York.
52. Masarotto, G. (1990). Bootstrap prediction intervals for autoregressions. *International Journal of Forecasting*, 6: 229-239.
53. McClain, J., (1988). Dominant tracking signals. *International Journal of Forecasting*. 4: 563-572.
54. McCulloch, W. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, Vol. 5: 115-133.
55. McCullough, B. (1994). Bootstrapping forecast intervals: an application to AR(p) models. *Journal of Forecasting*, 13: 51-66.
56. Minsky, M. and Papert, S. (1969). *Perceptrons*. MIT Press, Cambridge.
57. Montgomery, D. C., Johnson, L. A. and Gardiner, J. S. (1990). *Forecasting & Time Series Analysis*. McGraw-Hill, Inc., 2nd edition.
58. Nie, J. and Linkens, D. (1995). *Fuzzy-Neural Control: Principles, Algorithms and Applications*. Prentice Hall.

59. Palit, A. K. and Popovic, D. (1999). Fuzzy logic based automatic rule generation and forecasting of time series. IEEE international fuzzy systems conference proceedings.
60. Parker, D. B. (1982). Learning-logic. Invention Report S81-64, File 1, Office of Technology Licensing, Stanford University, October.
61. Pfluger, N., Yen, J. and Langari, R. (1992). A defuzzification strategy for a fuzzy logic controller employing prohibitive information in command formulation. In proceedings of IEEE international conference on Fuzzy Systems, 717-723, San Diego.
62. Refenes, A. -P. (1995). Neural Networks in the Capital Markets. John Wiley & Sons, Inc.
63. Refenes, A. -P. N. and Zapanis, A. D. (1999). Neural model identification, variable selection and model adequacy. Journal of Forecasting, Vol. 18 (5): 299-332.
64. Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, Vol. 65: 386-408.
65. Rosenblatt, F. (1962). Principles of Neurodynamics. Spartan Books, New York.
66. Rumelhart, D. E. and McClelland, J. L. (1986). Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol.1, MIT Press. Cambridge, MA.
67. Rumelhart, D. E., Hinton, G. E. and Williams, R. J. (1986). Learning Internal Representations by Error Propagation. In D. E. Rumelhart & J. L. McClelland

- (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Vol. 1: Foundations, 318-362. MIT Press.
68. Runkler, T. A. (1997). Selection of Appropriate Defuzzification Methods Using Application Specific Properties. *IEEE Transactions on Fuzzy Systems*, February.
69. Runkler, T. A. and Glesner, M. (1994). Defuzzification and ranking in the context of membership value semantics, rule modality, and measurement theory. In *European Congress on Fuzzy and Intelligent Technologies*, Aachen, September.
70. Smith, M. (1993). *Neural Networks for Statistical Modeling*. Van Nostrand Reinhold, New York.
71. Sugeno, M. and Kang, G. T. (1988). Structure identification of fuzzy model. *Fuzzy Sets and Systems*, 28:15-33.
72. Takagi, T. and Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man and Cybernetics*, 15:116-132.
73. Taylor, J. and Bunn, D. (1999). Investigating improvements in the accuracy of prediction intervals for combinations of forecasts: A simulation study. *International Journal of Forecasting*. 15: 325-339.
74. Thammano, A. (1999). A new forecasting approach with neuro-fuzzy architecture. In *proceedings of IEEE International Conference on Systems, Man, and Cybernetics, IEEE SMC'99*, Vol. 1: 386-389.

75. Thiesing, F. and Vornberger, O. (1997). Sales forecasting using neural networks. In proceedings of IEEE International Conference on Neural Networks, Vol. 4: 2125-2128.
76. Tibshirani, R. (1996). A comparison of some error estimates for neural network models. *Neural Computation*. 8: 152-163.
77. Tkacz, G. (2001). Neural network forecasting of Canadian GDP growth. *International Journal of Forecasting*, 17: 57-69.
78. Trig, W., (1964). Monitoring a forecasting system. *Operations Research Quarterly*, 15: 271-274.
79. Tsukamoto, Y. (1979). An approach to fuzzy reasoning method. *Advances in fuzzy set theory and applications*. 137-149. North-Holland, Amsterdam.
80. Vemuri, V. (1988). *Artificial Neural Networks: An Introduction*. Artificial Neural Networks: Theoretical Concepts. Computer Society Press of the IEEE.
81. Wehrens, R., Putter, H., and Buydens, L. (2000). The bootstrap: a tutorial. *Chemometrics and Intelligent Laboratory Systems*, 54: 35-52.
82. Widrow, B. and Stearns, D. (1985). *Adaptive Signal Processing*. Prentice Hall, Upper Saddle River, NJ.
83. Wu, B. (1999). Applications of fuzzy time series analysis to change periods detection. In proceedings of IEEE International Fuzzy Systems, FUZZ-IEEE'99, Vol. 2: 697-702.
84. Yar, M. and Chatfield, C. (1990). Prediction intervals for the Holt-Winters forecasting procedure. *International Journal of Forecasting*, 6: 127-137.

85. Yasunobu, S. and Miyamoto, S. (1985). Automatic train operation system by predictive fuzzy control. Industrial Applications of Fuzzy Control, in Sugeno, M. ed., North Holland.
86. Yen, J. and Langari, R. (1999). Fuzzy Logic: Intelligence, Control, and Information. Prentice Hall.
87. Zadeh, L. A. (1965). Fuzzy sets. Information and Control, 8:338-353.
88. Zadeh, L. A. (1971). Quantitative fuzzy semantics. Information Sciences, 3:159-176.
89. Zadeh, L. A. (1975). The concept of a linguistic variable and its application to approximate reasoning, Parts 1, 2, 3. Information Sciences, 8:199-249, 8:301-357,9:43-80.
90. Zadeh, L. A. (1992). Fuzzy logic, neural networks and soft computing. Course announcement, the University of California at Berkeley, November.
91. Zhang, G., Patuwo, B. E., and Hu, M. Y. (1998). Forecasting with artificial neural networks: the state of the art. International Journal of Forecasting, 14: 35-62.
92. Zhang, G., Patuwo, B. E., and Hu, M. Y. (2001). A simulation study of artificial neural networks for nonlinear time-series forecasting. Computers & Operations Research, 28: 381-396.

Appendices

Appendix I

Basic Concepts and Terminology of Membership Functions

The basic concepts and terminology of membership functions are as follows:

- Support

The support of a fuzzy set A is the set of all points x in X such that $\mu_A(x) > 0$:

$$\text{Support (A)} = \{x \mid \mu_A(x) > 0\}$$

- Core

The core of a fuzzy set is the set of all points x in X such that $\mu_A(x) = 1$:

$$\text{Core (A)} = \{x \mid \mu_A(x) = 1\}$$

- Boundary

The boundary of a fuzzy set is the set of all points x in X such that $0 < \mu_A(x) < 1$:

$$\text{Boundary (A)} = \{x \mid 0 < \mu_A(x) < 1\}$$

- Normality

A fuzzy set is normal if there is a point $x \in X$ such that $\mu_A(x) = 1$.

- Crossover point

A crossover point of a fuzzy set is a point $x \in X$ at which $\mu_A(x) = 0.5$:

$$\text{Crossover (A)} = \{x \mid \mu_A(x) = 0.5\}$$

- α -cut, strong α -cut

The α -cut of a fuzzy set A is a crisp set defined by

$$A_\alpha = \{x \mid \mu_A(x) \geq \alpha\}$$

Strong α -cut is defined similarly: $A'_\alpha = \{x \mid \mu_A(x) > \alpha\}$

Appendix II

Four Commonly Used Membership Functions

- Triangular Membership Function

A triangular membership function is specified by three parameters (a, b, c) as:

$$\text{Triangle}(x; a, b, c) = \begin{cases} 0 & x \leq a \\ \frac{x-a}{b-a} & a \leq x \leq b \\ \frac{c-x}{c-b} & b \leq x \leq c \\ 0 & c \leq x \end{cases}$$

Figure II-1 Triangular membership function: triangle (x; a, b, c)

- Trapezoidal Membership Function

A trapezoidal membership function is specified by four parameters (a, b, c, d) as:

$$\text{Trapezoid}(x; a, b, c, d) = \begin{cases} 0 & x \leq a \\ \frac{x-a}{b-a} & a \leq x \leq b \\ 1 & b \leq x \leq c \\ \frac{d-x}{d-c} & c \leq x \leq d \\ 0 & d \leq x \end{cases}$$

Figure II-2 Trapezoidal membership function: trapezoid (x; a, b, c, d)

- Gaussian Membership Function

A Gaussian membership function is specified by two parameters (c,σ):

$$\text{Gaussian}(x; c, \sigma) = e^{-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)^2} \quad 2.12$$

where

c — represents the MF center

σ — determines the MF width.

Figure II-3 Gaussian membership function: Gaussian (x; c, σ)

- Generalized bell Membership Function

A generalized bell membership function is specified by three parameters (a, b, c);

$$\text{Bell}(x; a, b, c) = \frac{1}{1 + \left| \frac{x - c}{a} \right|^{2b}}$$

where

a — represents the MFs width

b — controls the slopes at the crossover points

c — varies the MFs center.

Figure II-4 Generalized bell membership function: Bell (x; a, b, c)

Appendix III

Summary Statistics of Fitting Distributions for Residuals from Arena

Case I

Fit All Summary

Function	Sq Error
----------	----------

Normal	0.00739
Triangular	0.00931
Weibull	0.0107
Beta	0.0166
Erlang	0.0199
Gamma	0.0212
Lognormal	0.0439
Uniform	0.0659
Exponential	0.125

Distribution Summary

Distribution: Normal
Expression: NORM(-0.042, 2.51)
Square Error: 0.007385

Chi Square Test

Number of intervals = 5
Degrees of freedom = 2
Test Statistic = 1.82
Corresponding p-value = 0.421

Kolmogorov-Smirnov Test

Test Statistic = 0.0726
Corresponding p-value > 0.15

Data Summary

Number of Data Points = 50
Min Data Value = -5.6
Max Data Value = 6
Sample Mean = -0.042
Sample Std Dev = 2.54

Histogram Summary

Histogram Range = -6 to 6
Number of Intervals = 7

Case II

Fit All Summary

Function	Sq Error

Beta	0.00228
Normal	0.00264
Weibull	0.00568
Gamma	0.00847
Erlang	0.00861
Lognormal	0.0157
Triangular	0.0438
Uniform	0.113
Exponential	0.153

Distribution Summary

Distribution: Normal
Expression: NORM(0.0109, 0.183)
Square Error: 0.002638

Chi Square Test

Number of intervals = 6
Degrees of freedom = 3
Test Statistic = 4.81
Corresponding p-value = 0.201

Kolmogorov-Smirnov Test

Test Statistic = 0.057
Corresponding p-value > 0.15

Data Summary

Number of Data Points = 194
Min Data Value = -0.69
Max Data Value = 0.54
Sample Mean = 0.0109
Sample Std Dev = 0.184

Histogram Summary

Histogram Range = -0.82 to 0.67
Number of Intervals = 13

Case III

Fit All Summary

Function	Sq Error
----------	----------

Normal	0.00324
Beta	0.00758
Weibull	0.0081
Triangular	0.0146
Erlang	0.0291
Gamma	0.0291
Lognormal	0.0643
Uniform	0.088
Exponential	0.144

Distribution Summary

Distribution: Normal
Expression: NORM(-0.149, 16.2)
Square Error: 0.003244

Chi Square Test

Number of intervals = 4
Degrees of freedom = 1
Test Statistic = 0.664
Corresponding p-value = 0.44

Kolmogorov-Smirnov Test

Test Statistic = 0.0617
Corresponding p-value > 0.15

Data Summary

Number of Data Points = 77
Min Data Value = -48.6
Max Data Value = 43.1
Sample Mean = -0.149
Sample Std Dev = 16.3

Histogram Summary

Histogram Range = -49 to 44
Number of Intervals = 8

Appendix IV

Example S-PLUS Language Code to

Generate Bootstrapping Replicates for Residuals When $B = 20$

```
rm(test.df,result.df)
#Define Parameter
m <- 20

import.data(FileName="c:\\Forecasting\\BootStrap\\error1a.txt ",
FileType="ASCII",DataFrame="test.df")

attach(test.df)
test.df$err.trn
names(test.df)

mean(test.df$err.trn)
var(test.df$err.trn)

boot.obj.mean <- bootstrap(test.df$err.trn ,mean, B=1000, seed=0)
summary(boot.obj.mean)
plot(boot.obj.mean)
qqnorm(boot.obj.mean)

boot.obj.var <- bootstrap(test.df$err.trn ,var, B=1000, seed=0)
summary(boot.obj.var)
plot(boot.obj.var)
qqnorm(boot.obj.var)

n <- length(test.df$err.trn)

samp.mean <- boot.obj.mean$estimate$Mean
samp.var <- boot.obj.var$estimate$Mean

result.df <- test.df
for (i in 1:m) {

    resam.i <- rnorm(n,mean=samp.mean,sd=sqrt(samp.var))
    # names(resam.i) < paste("resam",i,sep="")
    # names(resam.i)
    # print(resam.i)
    result.df <- data.frame(result.df, resam.i)
}
result.df
names(result.df)
Edit.data(result.df)

export.data(DataSet = "result.df",
FileName = "c:/Forecasting/BootStrap/error_1a.xls",
FileType = "EXCEL")
```

Appendix V

Case I Forecasting Example for Weekly Sales of a Cutting Tool Data

1. The Data (100 Observations, Read Down from Left)

100	101	115	125	137
94	100	122	126	138
90	102	118	129	140
96	110	120	132	139
91	104	116	131	131
94	110	118	130	129
95	113	119	133	134
99	112	118	134	133
95	113	120	133	133
98	120	123	135	134
100	121	124	138	132
97	118	123	137	126
99	123	121	140	130
98	124	126	138	131
96	123	128	144	130
98	129	124	135	133
96	125	122	138	132
99	119	123	137	129
97	118	119	136	135
98	120	120	136	136

2. Forecasting Results from ANFIS (3, 2, 3) with 95% Prediction Intervals When B = 20

Time Period	X1	X2	X3	Actual	Forecast	Lower	Upper	Error
51	122	116	118	124	120.6	114	127	3.4
52	118	118	120	123	121.6	115	128	1.4
53	120	119	123	121	120.9	114	128	0.1
54	116	118	124	126	122.9	116	130	3.1
55	118	120	123	128	122.1	115	129	5.9
56	119	123	121	124	120.8	114	128	3.2
57	118	124	126	122	122.5	116	129	-0.5
58	120	123	128	123	121.2	114	129	1.8
59	123	121	124	119	119.2	111	127	-0.2
60	124	126	122	120	118.7	110	127	1.3
61	123	128	123	125	119.1	111	127	5.9
62	121	124	119	126	119.5	113	126	6.5
63	126	122	120	129	120.5	113	128	8.5
64	128	123	125	132	123.6	117	130	8.4
65	124	119	126	131	124	118	130	7
66	122	120	129	130	126.5	120	133	3.5
67	123	125	132	133	129.7	123	136	3.3
68	119	126	131	134	130	123	137	4
69	120	129	130	133	129.3	122	136	3.7
70	125	132	133	135	131.5	124	139	3.5
71	126	131	134	138	132	125	139	6
72	129	130	133	137	130.5	123	138	6.5
73	132	133	135	140	132.6	125	140	7.4
74	131	134	138	138	135.6	128	144	2.4
75	130	133	137	144	134.6	127	142	9.4
76	133	135	140	135	137.5	129	146	-2.5
77	134	138	138	138	137	130	144	1
78	133	137	144	137	141.8	132	151	-4.8
79	135	140	135	136	135.6	129	142	0.4
80	138	138	138	136	137.2	130	145	-1.2
81	137	144	137	137	140.7	134	148	-3.7
82	140	135	136	138	134.3	126	142	3.7
83	138	138	136	140	135.5	128	143	4.5
84	144	137	137	139	136.8	129	144	2.2
85	135	136	138	131	136.1	129	144	-5.1
86	138	136	140	129	131.9	123	140	-2.9
87	137	137	139	134	134	127	141	0
88	136	138	131	133	136.4	129	144	-3.4
89	136	140	129	133	138.1	126	150	-5.1
90	137	139	134	134	136.7	130	143	-2.7
91	138	131	133	132	128.2	120	136	3.8
92	140	129	133	126	123.2	111	136	2.8
93	139	134	134	130	131.5	125	138	-1.5
94	131	133	132	131	133.4	127	140	-2.4

95	129	133	126	130	129.7	123	136	0.3
96	134	134	130	133	133.4	127	140	-0.4
97	133	132	131	132	132.2	126	139	-0.2
98	133	126	130	129	125.4	118	133	3.6
99	134	130	133	135	129.4	122	136	5.6
100	132	131	132	136	131.7	125	138	4.3

Appendix VI

Case II Forecasting Example for Quarterly S&P 500 Index Data

1. The Data (388 Observations, Read Down from Left)

1.198938	1.468326	0.656165	1.233377	0.512429	1.490388	1.323799	0.827807
1.29991	1.489383	0.684185	1.276381	0.495691	1.499687	1.263968	0.916464
1.348435	1.467662	0.740614	1.144861	0.516431	1.566702	1.146498	0.9937
1.455093	1.33498	0.734218	1.102134	0.54956	1.546944	1.053527	0.987543
1.552336	1.250519	0.776154	0.787417	0.578644	1.368524	0.853609	0.989349
1.725968	1.233247	0.830696	0.745844	0.614934	1.263307	0.760621	1.119197
1.694321	1.186374	0.83713	0.687328	0.609539	1.29355	0.840847	1.159895
1.681869	1.204348	0.787446	0.82996	0.655938	1.41232	0.925902	1.246875
1.648196	1.163751	0.869476	0.888136	0.698641	1.490175	0.887717	0.984298
1.697809	0.36452	0.878108	0.852686	0.688422	1.507297	0.880443	0.980816
1.736508	0.34745	0.905885	0.765854	0.702849	1.545423	0.965462	0.985949
1.620808	1.045744	0.970039	0.823303	0.698998	1.62371	0.968054	0.986264
1.605157	1.120466	1.067984	0.866768	0.715065	1.669205	0.976045	1.002005
1.430414	1.157109	1.136023	0.832399	0.709367	1.710328	0.942943	1.042495
1.261505	1.29339	1.177344	0.733481	0.733726	1.736378	0.916814	1.105811
1.203517	1.181483	1.294396	0.690702	0.729718	1.758499	0.874519	1.18809
1.233163	1.176719	1.330927	0.717497	0.751424	1.761416	0.849037	1.181455
1.227566	1.193553	1.500402	0.665384	0.703638	1.736889	0.797858	1.130283
1.313457	1.263748	1.658972	0.621108	0.68475	1.818369	0.741642	1.154678
1.497786	1.122197	1.69201	0.651099	0.6935	1.795864	0.778478	1.088763
1.621974	1.041095	1.964777	0.583132	0.734557	1.708215	0.806959	1.011819
1.600681	0.936706	1.523377	0.521362	0.798756	1.560893	0.754357	1.111384
1.671367	0.759741	1.499516	0.473402	0.859653	1.513282	0.747306	1.174032
1.720116	0.73845	1.551051	0.494547	0.93184	1.632242	0.737583	1.178627
1.775124	0.712114	1.372924	0.525104	1.005588	1.699226	0.751129	1.166585
1.688127	0.698161	1.10029	0.579584	1.059995	1.731497	0.722683	1.227253
1.73614	0.704654	1.134978	0.632358	1.188118	1.714038	0.729338	1.20779
1.784338	0.677468	1.005005	0.632055	1.211593	1.637534	0.691362	1.214302
1.56487	0.725635	0.931256	0.596595	1.23531	1.726642	0.759021	1.219254

1.395649	0.720057	0.689696	0.609218	1.268968	1.735266	0.792037	1.265997
1.320892	0.699057	0.600878	0.619786	1.287893	1.801338	0.759779	1.261039
1.107306	0.621649	0.410734	0.644014	1.227746	1.698151	0.745409	1.274634
1.194777	0.57267	0.527989	0.645696	1.163887	1.68168	0.685688	1.306225
1.318208	0.523322	0.546046	0.681164	1.204931	1.535972	0.648343	1.292558
1.420211	0.477349	0.513565	0.720807	1.180474	1.507182	0.594531	1.225041
1.517884	0.463622	0.692795	0.734287	1.027435	1.392017	0.582839	1.239066
1.54937	0.464201	0.868333	0.814037	1.037869	1.219974	0.570434	1.208609
1.656639	0.452888	0.790587	0.836564	1.078462	1.190827	0.672484	1.239951
1.739487	0.499671	0.868893	0.851665	1.163697	1.280689	0.71761	1.331009
1.755338	0.539891	0.807528	0.755295	1.266832	1.415506	0.781507	1.411654
1.632925	0.61203	0.714021	0.646392	1.336236	1.462803	0.785447	1.473569
1.550926	0.649351	0.703984	0.643798	1.375894	1.399851	0.777291	1.55946
1.436905	0.668508	0.677533	0.587655	1.396324	1.349871	0.740967	1.585132
1.500498	0.681895	0.727574	0.598795	1.365095	1.458016	0.709248	1.78814
1.512805	0.63943	0.837852	0.570929	1.319374	1.47834	0.720424	1.81748
1.528752	0.590358	0.937242	0.527187	1.304084	1.475108	0.730705	
1.471175	0.598424	1.05792	0.576425	1.285667	1.522089	0.774187	
1.434839	0.632881	1.063064	0.560576	1.26664	1.505316	0.796668	
1.4091	0.610933	1.148668	0.531846	1.410066	1.379191	0.801489	

2. Forecasting Results from ANFIS (3, 2, 3) with 95% Prediction Intervals When B = 20

Time Period	X1	X2	X3	Actual	Forecast	Lower	Upper	Error
195	0.852	0.644	0.571	0.561	0.666	0.273	1.059	-0.105
196	0.755	0.588	0.527	0.532	0.626	0.234	1.018	-0.094
197	0.646	0.599	0.576	0.512	0.61	0.222	0.998	-0.097
198	0.644	0.571	0.561	0.496	0.604	0.215	0.993	-0.108
199	0.588	0.527	0.532	0.516	0.559	0.165	0.953	-0.043
200	0.599	0.576	0.512	0.55	0.527	0.132	0.922	0.022
201	0.571	0.561	0.496	0.579	0.496	0.097	0.895	0.083
202	0.527	0.532	0.516	0.615	0.496	0.095	0.897	0.119
203	0.576	0.512	0.55	0.61	0.577	0.183	0.971	0.032
204	0.561	0.496	0.579	0.656	0.608	0.212	1.004	0.048
205	0.532	0.516	0.615	0.699	0.63	0.234	1.026	0.069
206	0.512	0.55	0.61	0.688	0.597	0.202	0.992	0.091
207	0.496	0.579	0.656	0.703	0.641	0.245	1.037	0.062
208	0.516	0.615	0.699	0.699	0.695	0.302	1.088	0.004
209	0.55	0.61	0.688	0.715	0.695	0.305	1.085	0.021
210	0.579	0.656	0.703	0.709	0.702	0.313	1.091	0.008
211	0.615	0.699	0.699	0.734	0.688	0.300	1.076	0.045
212	0.61	0.688	0.715	0.73	0.711	0.323	1.099	0.018
213	0.656	0.703	0.709	0.751	0.711	0.323	1.099	0.041
214	0.699	0.699	0.734	0.704	0.747	0.360	1.134	-0.044
215	0.688	0.715	0.73	0.685	0.735	0.347	1.123	-0.05
216	0.703	0.709	0.751	0.694	0.763	0.375	1.151	-0.07
217	0.699	0.734	0.704	0.735	0.703	0.315	1.091	0.032
218	0.715	0.73	0.685	0.799	0.689	0.301	1.077	0.109
219	0.709	0.751	0.694	0.86	0.688	0.299	1.077	0.171
220	0.734	0.704	0.735	0.932	0.751	0.363	1.139	0.181
221	0.73	0.685	0.799	1.006	0.826	0.437	1.215	0.18
222	0.751	0.694	0.86	1.06	0.891	0.500	1.282	0.169
223	0.704	0.735	0.932	1.188	0.979	0.587	1.371	0.209
224	0.685	0.799	1.006	1.212	1.063	0.669	1.457	0.148
225	0.694	0.86	1.06	1.235	1.114	0.719	1.509	0.121
226	0.735	0.932	1.188	1.269	1.268	0.866	1.670	0
227	0.799	1.006	1.212	1.288	1.236	0.841	1.631	0.052
228	0.86	1.06	1.235	1.228	1.221	0.828	1.614	0.007
229	0.932	1.188	1.269	1.164	1.193	0.801	1.585	-0.029
230	1.006	1.212	1.288	1.205	1.195	0.800	1.590	0.01
231	1.06	1.235	1.228	1.18	1.108	0.710	1.506	0.073
232	1.188	1.269	1.164	1.027	1.032	0.630	1.434	-0.004
233	1.212	1.288	1.205	1.038	1.071	0.670	1.472	-0.033
234	1.235	1.228	1.18	1.078	1.043	0.642	1.444	0.036
235	1.269	1.164	1.027	1.164	0.895	0.491	1.299	0.269
236	1.288	1.205	1.038	1.267	0.911	0.507	1.315	0.356

237	1.228	1.18	1.078	1.336	0.944	0.541	1.347	0.393
238	1.164	1.027	1.164	1.376	1.049	0.649	1.449	0.327
239	1.205	1.038	1.267	1.396	1.188	0.787	1.589	0.208
240	1.18	1.078	1.336	1.365	1.287	0.885	1.689	0.078
241	1.027	1.164	1.376	1.319	1.334	0.937	1.731	-0.015
242	1.038	1.267	1.396	1.304	1.316	0.923	1.709	-0.012
243	1.078	1.336	1.365	1.286	1.247	0.852	1.642	0.039
244	1.164	1.376	1.319	1.267	1.184	0.786	1.582	0.082
245	1.267	1.396	1.304	1.41	1.17	0.772	1.568	0.24
246	1.336	1.365	1.286	1.49	1.154	0.756	1.552	0.337
247	1.376	1.319	1.267	1.5	1.134	0.735	1.533	0.365
248	1.396	1.304	1.41	1.567	1.33	0.932	1.728	0.236
249	1.365	1.286	1.49	1.547	1.465	1.062	1.868	0.082
250	1.319	1.267	1.5	1.369	1.482	1.079	1.885	-0.113
251	1.304	1.41	1.567	1.263	1.489	1.087	1.891	-0.226
252	1.286	1.49	1.547	1.294	1.412	1.011	1.813	-0.119
253	1.267	1.5	1.369	1.412	1.227	0.830	1.624	0.185
254	1.41	1.567	1.263	1.49	1.178	0.780	1.576	0.312
255	1.49	1.547	1.294	1.507	1.189	0.795	1.583	0.319
256	1.5	1.369	1.412	1.545	1.33	0.936	1.724	0.216
257	1.567	1.263	1.49	1.624	1.58	1.172	1.988	0.044
258	1.547	1.294	1.507	1.669	1.575	1.175	1.975	0.094
259	1.369	1.412	1.545	1.71	1.471	1.064	1.878	0.24
260	1.263	1.49	1.624	1.736	1.505	1.100	1.910	0.231
261	1.294	1.507	1.669	1.758	1.557	1.143	1.971	0.202
262	1.412	1.545	1.71	1.761	1.606	1.159	2.053	0.156
263	1.49	1.624	1.736	1.737	1.67	1.193	2.147	0.067
264	1.507	1.669	1.758	1.818	1.656	1.161	2.151	0.162
265	1.545	1.71	1.761	1.796	1.648	1.138	2.158	0.148
266	1.624	1.736	1.737	1.708	1.651	1.141	2.161	0.057
267	1.669	1.758	1.818	1.561	1.601	1.031	2.171	-0.041
268	1.71	1.761	1.796	1.513	1.597	1.043	2.151	-0.083
269	1.736	1.737	1.708	1.632	1.622	1.110	2.134	0.01
270	1.758	1.818	1.561	1.699	1.547	0.985	2.109	0.152
271	1.761	1.796	1.513	1.731	1.523	0.974	2.072	0.209
272	1.737	1.708	1.632	1.714	1.617	1.121	2.113	0.097
273	1.818	1.561	1.699	1.638	1.556	1.109	2.003	0.081
274	1.796	1.513	1.731	1.727	1.55	1.107	1.993	0.176
275	1.708	1.632	1.714	1.735	1.637	1.164	2.110	0.098
276	1.561	1.699	1.638	1.801	1.647	1.181	2.113	0.155
277	1.513	1.731	1.727	1.698	1.612	1.124	2.100	0.086
278	1.632	1.714	1.735	1.682	1.66	1.267	2.053	0.022
279	1.699	1.638	1.801	1.536	1.616	1.187	2.045	-0.08
280	1.731	1.727	1.698	1.507	1.628	1.219	2.037	-0.12
281	1.714	1.735	1.682	1.392	1.634	1.224	2.044	-0.242

282	1.638	1.801	1.536	1.22	1.568	1.126	2.010	-0.348
283	1.727	1.698	1.507	1.191	1.543	1.096	1.990	-0.352
284	1.735	1.682	1.392	1.281	1.404	0.963	1.845	-0.123
285	1.801	1.536	1.22	1.416	0.934	0.534	1.334	0.481
286	1.698	1.507	1.191	1.463	1.004	0.608	1.400	0.459
287	1.682	1.392	1.281	1.4	1.061	0.660	1.462	0.339
288	1.536	1.22	1.416	1.35	1.347	0.949	1.745	0.003
289	1.507	1.191	1.463	1.458	1.437	1.039	1.835	0.021
290	1.392	1.281	1.4	1.478	1.424	1.035	1.813	0.054
291	1.22	1.416	1.35	1.475	1.259	0.874	1.644	0.216
292	1.191	1.463	1.458	1.522	1.259	0.871	1.647	0.263
293	1.281	1.4	1.478	1.505	1.478	1.092	1.864	0.027
294	1.416	1.35	1.475	1.379	1.514	1.125	1.903	-0.135
295	1.463	1.458	1.522	1.324	1.57	1.181	1.959	-0.246
296	1.4	1.478	1.505	1.264	1.542	1.154	1.930	-0.278
297	1.35	1.475	1.379	1.146	1.385	0.999	1.771	-0.239
298	1.458	1.522	1.324	1.054	1.372	0.983	1.761	-0.318
299	1.478	1.505	1.264	0.854	1.299	0.910	1.688	-0.445
300	1.475	1.379	1.146	0.761	1.076	0.685	1.467	-0.316
301	1.522	1.324	1.054	0.841	0.862	0.468	1.256	-0.021
302	1.505	1.264	0.854	0.926	0.654	0.261	1.047	0.272
303	1.379	1.146	0.761	0.888	0.663	0.275	1.051	0.225
304	1.324	1.054	0.841	0.88	0.7	0.314	1.086	0.18
305	1.264	0.854	0.926	0.965	0.837	0.452	1.222	0.128
306	1.146	0.761	0.888	0.968	0.889	0.504	1.274	0.079
307	1.054	0.841	0.88	0.976	0.892	0.507	1.277	0.084
308	0.854	0.926	0.965	0.943	0.988	0.603	1.373	-0.045
309	0.761	0.888	0.968	0.917	0.986	0.602	1.370	-0.07
310	0.841	0.88	0.976	0.875	1.034	0.649	1.419	-0.159
311	0.926	0.965	0.943	0.849	0.951	0.566	1.336	-0.101
312	0.888	0.968	0.917	0.798	0.903	0.518	1.288	-0.106
313	0.88	0.976	0.875	0.742	0.844	0.459	1.229	-0.102
314	0.965	0.943	0.849	0.778	0.849	0.463	1.235	-0.071
315	0.968	0.917	0.798	0.807	0.798	0.412	1.184	0.009
316	0.976	0.875	0.742	0.754	0.745	0.360	1.130	0.009
317	0.943	0.849	0.778	0.747	0.787	0.402	1.172	-0.039
318	0.917	0.798	0.807	0.738	0.832	0.447	1.217	-0.094
319	0.875	0.742	0.754	0.751	0.781	0.396	1.166	-0.03
320	0.849	0.778	0.747	0.723	0.763	0.378	1.148	-0.04
321	0.798	0.807	0.738	0.729	0.737	0.352	1.122	-0.008
322	0.742	0.754	0.751	0.691	0.761	0.376	1.146	-0.069
323	0.778	0.747	0.723	0.759	0.735	0.350	1.120	0.024
324	0.807	0.738	0.729	0.792	0.749	0.364	1.134	0.043
325	0.754	0.751	0.691	0.76	0.694	0.309	1.079	0.066
326	0.747	0.723	0.759	0.745	0.783	0.398	1.168	-0.038

327	0.738	0.729	0.792	0.686	0.823	0.438	1.208	-0.138
328	0.751	0.691	0.76	0.648	0.796	0.411	1.181	-0.148
329	0.723	0.759	0.745	0.595	0.748	0.363	1.133	-0.153
330	0.729	0.792	0.686	0.583	0.668	0.283	1.053	-0.085
331	0.691	0.76	0.648	0.57	0.63	0.246	1.014	-0.06
332	0.759	0.745	0.595	0.672	0.602	0.217	0.987	0.071
333	0.792	0.686	0.583	0.718	0.609	0.224	0.994	0.108
334	0.76	0.648	0.57	0.782	0.601	0.216	0.986	0.18
335	0.745	0.595	0.672	0.785	0.72	0.335	1.105	0.066
336	0.686	0.583	0.718	0.777	0.774	0.389	1.159	0.004
337	0.648	0.57	0.782	0.741	0.864	0.480	1.248	-0.123
338	0.595	0.672	0.785	0.709	0.808	0.424	1.192	-0.099
339	0.583	0.718	0.777	0.72	0.767	0.383	1.151	-0.047
340	0.57	0.782	0.741	0.731	0.67	0.286	1.054	0.06
341	0.672	0.785	0.709	0.774	0.677	0.293	1.061	0.097
342	0.718	0.777	0.72	0.797	0.709	0.324	1.094	0.088
343	0.782	0.741	0.731	0.801	0.747	0.362	1.132	0.055
344	0.785	0.709	0.774	0.828	0.812	0.427	1.197	0.015
345	0.777	0.72	0.797	0.916	0.838	0.453	1.223	0.078
346	0.741	0.731	0.801	0.994	0.836	0.451	1.221	0.157
347	0.709	0.774	0.828	0.988	0.845	0.461	1.229	0.143
348	0.72	0.797	0.916	0.989	0.966	0.582	1.350	0.024
349	0.731	0.801	0.994	1.119	1.093	0.708	1.478	0.026
350	0.774	0.828	0.988	1.16	1.075	0.690	1.460	0.085
351	0.797	0.916	0.989	1.247	1.009	0.625	1.393	0.238
352	0.801	0.994	1.119	0.984	1.12	0.735	1.505	-0.136
353	0.828	0.988	1.16	0.981	1.217	0.832	1.602	-0.237
354	0.916	0.989	1.247	0.986	1.416	1.031	1.801	-0.43
355	0.994	1.119	0.984	0.986	0.927	0.542	1.312	0.059
356	0.988	1.16	0.981	1.002	0.884	0.499	1.269	0.118
357	0.989	1.247	0.986	1.042	0.792	0.407	1.177	0.251
358	1.119	0.984	0.986	1.106	0.987	0.602	1.372	0.119
359	1.16	0.981	1.002	1.188	0.99	0.605	1.375	0.198
360	1.247	0.986	1.042	1.181	0.992	0.606	1.378	0.19
361	0.984	0.986	1.106	1.13	1.188	0.803	1.573	-0.058
362	0.981	1.002	1.188	1.155	1.313	0.928	1.698	-0.158
363	0.986	1.042	1.181	1.089	1.262	0.877	1.647	-0.174
364	0.986	1.106	1.13	1.012	1.117	0.733	1.501	-0.106
365	1.002	1.188	1.155	1.111	1.06	0.676	1.444	0.051
366	1.042	1.181	1.089	1.174	1.024	0.639	1.409	0.15
367	1.106	1.13	1.012	1.179	0.993	0.607	1.379	0.185
368	1.188	1.155	1.111	1.167	1.105	0.719	1.491	0.062
369	1.181	1.089	1.174	1.227	1.213	0.827	1.599	0.014
370	1.13	1.012	1.179	1.208	1.272	0.886	1.658	-0.064
371	1.155	1.111	1.167	1.214	1.2	0.814	1.586	0.014

372	1.089	1.174	1.227	1.219	1.238	0.853	1.623	-0.019
373	1.012	1.179	1.208	1.266	1.153	0.768	1.538	0.113
374	1.111	1.167	1.214	1.261	1.235	0.850	1.620	0.026
375	1.174	1.227	1.219	1.275	1.219	0.834	1.604	0.056
376	1.179	1.208	1.266	1.306	1.295	0.910	1.680	0.012
377	1.167	1.214	1.261	1.293	1.281	0.896	1.666	0.012
378	1.227	1.219	1.275	1.225	1.304	0.918	1.690	-0.079
379	1.208	1.266	1.306	1.239	1.321	0.935	1.707	-0.081
380	1.214	1.261	1.293	1.209	1.307	0.921	1.693	-0.099
381	1.219	1.275	1.225	1.24	1.216	0.830	1.602	0.024
382	1.266	1.306	1.239	1.331	1.234	0.848	1.620	0.097
383	1.261	1.293	1.209	1.412	1.199	0.813	1.585	0.213
384	1.275	1.225	1.24	1.474	1.247	0.856	1.638	0.226
385	1.306	1.239	1.331	1.559	1.396	1.011	1.781	0.163
386	1.293	1.209	1.412	1.585	1.471	1.085	1.857	0.114
387	1.225	1.24	1.474	1.788	1.415	1.014	1.816	0.373
388	1.239	1.331	1.559	1.817	1.352	0.945	1.759	0.466

Appendix VII

Case III Forecasting Example for Quarterly Beer Production Data in Australia

1. The Data (154 Observations, Read Down from Left)

284.4	249.8	327	556.1	547.5	416	411
212.8	269.8	442.3	499.6	440.2	440.2	512
226.9	345.7	383.1	451.9	468.7	538.7	449
308.4	293.8	331.6	434.9	574.5	473.8	382
262	254.7	361.4	553.8	492.6	439.9	
227.9	277.5	445.9	510	432.6	446.8	
236.1	363.4	386.6	432.9	479.8	597.5	
320.4	313.4	357.2	453.2	575.7	467.2	
271.9	272.8	373.6	547.6	474.6	439.4	
232.8	300.1	466.2	485.8	405.3	447.4	
237	369.5	409.6	452.6	434.6	568.5	
313.4	330.8	369.8	456.6	535.1	485.9	
261.4	287.8	378.6	565.7	452.6	442.1	
226.8	305.9	487	514.8	429.5	430.5	
249.9	386.1	419.2	464.3	417.2	600	
314.3	335.2	376.7	430.9	551.8	464.5	
286.1	288	392.8	588.3	464	423.6	
226.5	308.3	506.1	503.1	416.6	437	
260.4	402.3	458.4	442.6	422.9	574	
311.4	352.8	387.4	448	553.6	443	
294.7	316.1	426.9	554.5	458.6	410	
232.6	324.9	565	504.5	427.6	420	
257.2	404.8	464.8	427.3	429.2	532	
339.2	393	444.5	473.1	534.2	432	
279.1	318.9	449.5	526.2	481.7	420	

2. Forecasting Results from ANFIS (3, 2, 4) with 95% Prediction Intervals When B = 20

Time Period	X1	X2	X3	Actual	Forecast	Lower	Upper	Error
78	376.7	387.4	444.5	451.9	450.2	371	530	1.7
79	392.8	426.9	449.5	434.9	455.9	414	498	-21
80	506.1	565	556.1	553.8	555.6	446	665	-1.8
81	458.4	464.8	499.6	510	513.4	418	608	-3.4
82	387.4	444.5	451.9	432.9	448.7	363	534	-15.8
83	426.9	449.5	434.9	453.2	460.3	418	503	-7.1
84	565	556.1	553.8	547.6	557.9	427	688	-10.3
85	464.8	499.6	510	485.8	516.9	431	603	-31.1
86	444.5	451.9	432.9	452.6	467.1	405	529	-14.5
87	449.5	434.9	453.2	456.6	482.3	431	533	-25.7
88	556.1	553.8	547.6	565.7	555	444	666	10.7
89	499.6	510	485.8	514.8	513.1	453	573	1.7
90	451.9	432.9	452.6	464.3	482.7	404	561	-18.4
91	434.9	453.2	456.6	430.9	477.2	418	537	-46.3
92	553.8	547.6	565.7	588.3	569	449	689	19.3
93	510	485.8	514.8	503.1	535.4	415	656	-32.3
94	432.9	452.6	464.3	442.6	481.2	325	638	-38.6
95	453.2	456.6	430.9	448	469.2	391	547	-21.2
96	547.6	565.7	588.3	554.5	586.9	471	703	-32.4
97	485.8	514.8	503.1	504.5	519.1	445	594	-14.6
98	452.6	464.3	442.6	427.3	475.4	361	590	-48.1
99	456.6	430.9	448	473.1	481.1	434	528	-8
100	565.7	588.3	554.5	526.2	561.1	480	642	-34.9
101	514.8	503.1	504.5	547.5	527.9	465	591	19.6
102	464.3	442.6	427.3	440.2	470.6	384	558	-30.4
103	430.9	448	473.1	468.7	486.6	450	523	-17.9
104	588.3	554.5	526.2	574.5	525.3	374	676	49.2
105	503.1	504.5	547.5	492.6	554.8	517	593	-62.2
106	442.6	427.3	440.2	432.6	471.3	410	533	-38.7
107	448	473.1	468.7	479.8	487.8	445	530	-8
108	554.5	526.2	574.5	575.7	576.3	507	645	-0.6
109	504.5	547.5	492.6	474.6	514.9	412	618	-40.3
110	427.3	440.2	432.6	405.3	460.1	366	554	-54.8
111	473.1	468.7	479.8	434.6	483.8	434	533	-49.2
112	526.2	574.5	575.7	535.1	562.3	495	629	-27.2
113	547.5	492.6	474.6	452.6	485.4	381	590	-32.8
114	440.2	432.6	405.3	429.5	417.6	332	503	11.9
115	468.7	479.8	434.6	417.2	442.9	351	535	-25.7
116	574.5	575.7	535.1	551.8	539.6	475	604	12.2
117	492.6	474.6	452.6	464	461	401	521	3
118	432.6	405.3	429.5	416.6	440.4	392	489	-23.8
119	479.8	434.6	417.2	422.9	427.6	362	493	-4.7
120	575.7	535.1	551.8	553.6	556.5	470	643	-2.9
121	474.6	452.6	464	458.6	471.2	417	525	-12.6

122	405.3	429.5	416.6	427.6	426.2	347	505	1.4
123	434.6	417.2	422.9	429.2	434.1	383	485	-4.9
124	535.1	551.8	553.6	534.2	547.7	490	605	-13.5
125	452.6	464	458.6	481.7	463.9	414	513	17.8
126	429.5	416.6	427.6	416	438.1	387	489	-22.1
127	417.2	422.9	429.2	440.2	438.2	383	493	2
128	551.8	553.6	534.2	538.7	535.3	481	590	3.4
129	464	458.6	481.7	473.8	484.9	431	539	-11.1
130	416.6	427.6	416	439.9	426.7	345	509	13.2
131	422.9	429.2	440.2	446.8	447.6	389	506	-0.8
132	553.6	534.2	538.7	597.5	541.1	477	605	56.4
133	458.6	481.7	473.8	467.2	476.2	400	553	-9
134	427.6	416	439.9	439.4	448.5	396	501	-9.1
135	429.2	440.2	446.8	447.4	453.1	347	559	-5.7
136	534.2	538.7	597.5	568.5	582.8	484	682	-14.3
137	481.7	473.8	467.2	485.9	473.5	420	527	12.4
138	416	439.9	439.4	442.1	445.6	364	527	-3.5
139	440.2	446.8	447.4	430.5	454.3	407	502	-23.8
140	538.7	597.5	568.5	600	555.8	481	631	44.2
141	473.8	467.2	485.9	464.5	489.1	416	562	-24.6
142	439.9	439.4	442.1	423.6	450.2	399	501	-26.6
143	446.8	447.4	430.5	437	440	348	532	-3
144	597.5	568.5	600	574	596.4	486	707	-22.4
145	467.2	485.9	464.5	443	509.9	435	585	-66.9
146	439.4	442.1	423.6	410	354	260	448	56
147	447.4	430.5	437	420	399.6	324	475	20.4
148	568.5	600	574	532	617.9	544	692	-85.9
149	485.9	464.5	443	432	438.9	292	585	-6.9
150	442.1	423.6	410	420	381	300	462	39
151	430.5	437	420	411	356.4	277	436	54.6
152	600	574	532	512	582.4	468	696	-70.4
153	464.5	443	432	449	388.7	292	485	60.3
154	423.6	410	420	382	377.5	255	500	4.5