University of Alberta

ARTIFICIAL NEURAL NETWORKS AND THEIR APPLICATION TO WATER TREATMENT

by

Qing Zhang ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**

in

ENVIRONMENTAL ENGINEERING

Department of Civil and Environmental Engineering

Edmonton, Alberta

Fall 1996

ISBN 0-612-18342-4

[illegible]

Qing Zhang
Apt 204, 10612 29 Avenue
Edmonton, Alberta
T6J 4H6

Date: Sep 9, 96

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Artificial Neural Networks And Their Application In Water Treatment** submitted by **Qing Zhang** in partial fulfillment of the requirements for the degree of **Master of Science** in **Environmental Engineering**.

[illegible]

Dr. S. J. Stanley

[illegible]

Professor L. R. Plitt

[illegible]

Dr. G. R. Finch

Ian Buchanan

Professor I. Buchanan

Sep. 9, 96

Abstract

The artificial neural network is an emerging artificial intelligence modeling technique which has a great potential in environmental engineering and especially related to treatment processes. In this thesis, an artificial neural network modeling approach was used for two applications in water treatment. First, an artificial neural network model was built to forecast the raw water colour in the North Saskatchewan River. Similar models for the other raw water quality parameters can be established by using the approach outlined in the thesis. This thesis proceeds to provide a survey of recent research applications of neural networks in two process control strategies: adaptive control and internal model control. From the analysis of the benefits and deficiencies of these two control strategies, and the associated engineering knowledge of the water treatment process, a feedforward neural network controller was proposed and built for the Rossdale water treatment plant in Edmonton, Alberta. Upon the success of both models, this thesis also attempts to explain why neural network modeling functions well from the viewpoints of inductive learning and the computational theory. The methodology of making the hypothesis space tractable in order to obtain the optimum neural network model was also demonstrated.

Acknowledgments

Financial support for completion of this research was provided by the City of Edmonton through Dr. Stephen J. Stanley and Mr. David Rector from Rossdale Water Treatment Plant is gratefully acknowledged.

I would sincerely like to thank Dr. Stephen Stanley for his guidance and support throughout this project, Dr. Daniel. W. Smith for his advice on the early proposal of this project, and Dr. Renee Elio from the Computer Science Department at the University of Alberta for her guidance in the research regarding to the artificial intelligence aspects of this thesis.

I also extend my sincerest thanks to the following:

1. Mr. David Rector, the engineers and the staff in the Rossdale Water Treatment Plant for their assistance in source data collection and their precious opinions in the process control design.

2. Dr. Stefan Kremer of the Computer Science Department at the University of Alberta for providing his Ph. D. thesis regarding to the research in inductive learning.

3. Ms. Tanya Armstrong and Ms. Samantha Rizak for their kindly assistance in proof reading my thesis and reports.

Finally, I sincerely thank my family - my parents and sisters for their wonderful constant support.

Table of Contents

CHAPTER I INTRODUCTION TO ARTIFICIAL INTELLIGENCE MODELING APPROACHES

1. Introduction 1

2. Outline 5

References 7

CHAPTER II INTRODUCTION TO ARTIFICIAL NEURAL NETWORK

1. Introduction 9

2. Concepts of ANN 10

2.1 ANN In General 10

2.2 The ANN Structure and the Artificial Neuron 11

2.3 Error Backpropagation 13

2.4 Development of A Backpropagation Neural Network 14

2.4.1 Algorithm Verification 14

2.4.2 Neural Network Architecture Design 16

2.4.3 Training the Network 18

2.4.4 Model Verification 20

3. Time Series Analysis and the Model Rendering Process 21

3.1 The Rendering Model Identification Procedure 22

3.1.1 Stationary Time Series 23

3.1.2 Autocorrelation Coefficient Function (ACF) 24

3.1.3 Partial Autocorrelation Coefficient Function (PACF) 25

3.1.4 The Box-Jenkins Models 25

3.2 Model Rendering 27

4. Conclusion 28

References 30

CHAPTER III FORECASTING THE RAW WATER QUALITY PARAMETERS FOR THE NORTH SASKATCHEWAN RIVER BY NEURAL NETWORK MODELING

1. Introduction 34

2. Project Description 35

3. Artificial Neural Network Overview 37

4 Neural Network Modeling In Water Treatment 40

5 Raw Water Colour Forecasting 42

5.1 Source Data Analysis 42

5.2 System Priming 45

5.3 System Fine-tuning 49

5.4 Model Evaluation 52

5.5 Results 53

6 Conclusion and Recommendations 58

References 59

CHAPTER IV SOLVING ENGINEERING PROBLEMS WITH ARTIFICIAL NEURAL NETWORK IN AN INDUCTIVE HEURIDTIC APPROACH

1. Introduction 61

2. An Indutive ANN Modeling Process 62

3. Why Inductive Learning Works With ANN 65

3.1 Theoretical Limits of the Sample Size 66

3.2 ANN Learning As Search In the Hypothesis Space 72

4. Hypothesis Space of ANN Modeling Approach 74

5. Partitioning the ANN Hypothesis Space 76

6. Prune the ANN Hypothesis Space 80

7. Discussion 91

8. Conclusion 93

References 97

CHAPTER V REAL-TIME WATER TREATMENT PROCESS CONTROL WITH ANN

1. Introduction 99

2. Background 101

3. Neural Network As A Plant Process Model 107

4. A Control Model For Coagulation-Flocculation-Sedimentation Process 112

5. Conclusion 122

References 124

CHAPTER VI CONCLUSIONS

1. Results and Discussions 126

2. Recommended Further Research 131

APPENDIX I BACK PROPAGATION CALCULATIONS 134

APPENDIX II THE CRITERIA TO EVALUATE AN ANN MODELING RUN RESULT 139

APPENDIX III RESIDUAL ANALYSIS CHARTS FOR THE FEEDFORWARD NEURAL NETWORK CONTROLLER 143

Lists of Tables

Table 3-1 The Statistical Data for the Candidate Models 55

Table 4-1 The Factorial Design 85

Table 4-2 The Runs And The Results 86

Table 4-3 The New 2^2 Factorial Design 87

Table 4-4 The New 2^2 Runs and Results 87

Table 4-5 The “Star” Runs and Results 89

Table 4-6 The Second “Star” Runs and Results 89

Table 5-1 The Raw Water Condition 1 118

Table 5-2 The Control Model Results 121

Table 5-3 The Raw Water Condition 2 121

Table 5-4 The Statistical Data for the Candidate Models 121

Lists of Figures

Figure 1-1 The Engineering Modeling Approaches........4

Figure 2-1 A Simple Neural Network........11

Figure 2-2 An Artificial Neuron........12

Figure 2-3 The S-shaped Sigmoid Function........13

Figure 2-4 The ANN Modeling Approach........15

Figure 3-1 The North Saskatchewan River Basin........37

Figure 3-2 A Simple Neural Network........39

Figure 3-3 A Sample Annual Cycle Pattern of Colour in 1994........41

Figure 3-4 A Sample Contribution Factor Graph For All the Input Parameters........47

Figure 3-5 The Time Series of Colour Difference in 1994........50

Figure 3-6 The Training Set Data Prediction Results (1990) for Model A........54

Figure 3-7 The Colour Prediction by Candidate Model A (1994)........56

Figure 3-8 The Residual Versus Time (1990-1993)........57

Figure 3-9 The Residuals Versus True Colour (1990-1993)........57

Figure 4-1 The Diagram for PAC-learning Concept 68

Figure 4-2 The Partitioned Hypothesis Space For ANN Model 70

Figure 4-3 The New Partitioned Hypothesis Space 77

Figure 4-4 The Final Partitioned Hypothesis Space 78

Figure 4-5 The Star Design Testing Pattern 88

Figure 4-6 The Partial Response Surface Measuring The Success Of ANN Models Respecting To The Variance In Neuron Ratio And Total Number Of Connection Weights 89

Figure 4-7 The Complete Response Surface Measuring The Success Of ANN Models Respecting To The Variance In Neuron Ratio And Total Number Of Connection Weights 91

Figure 5-1 Rossdale Water Treatment Plant Layout 102

Figure 5-2 The Daily Record of Clarifier Effluent of 1993 105

Figure 5-3 The IMC Structure 108

Figure 5-4 The Modified Direct IMC 111

Figure 5-5 The Proposed Process Control Model For Coagulation-Flocculation-Sedimentation 113

Figure 5-6 The Alum Dose Optimization Process 116

Figure 5-7 The Model Prediction Versus the Testing Data for The Plant Reference Model 119

Figure 5-8 The Model Prediction Versus the Testing Data for The Plant Inverse Model 120

List of Symbols

Symbol	Definition
AD_1	Alum Dose as an input to the process reference model.
AD_2	Alum Dose as an output from the inverse process model.
C	The inverse process model.
C_n	The colour time series.
$C_n^{\cdot}$	The colour time series predicted by neural network.
ΔC_n	The colour difference time series.
$\Delta C_n^{\cdot}$	The colour difference time series predicted by Neural Network.
ET_1	Effluent Turbidity as an output from the process reference model.
e	The desired operation state of the plant.
e_t	The white noise which is unknown and expected to be zero in the moving average model.
e_{t-1}, e_{t-2}	The past white noise (forecast error).
$f(x)$	The true function of an inductive learning example.
g_m	The conjecture series of an inductive learning when the inductive inference learns more and more about the inductive problem.
H	The number of hypothesis in the hypothesis space.
H_{bad}	The portion of the hypothesis space that contains the bad hypothesizes.
h(x)	The approximate function of f through inductive learning.
h_b	The bad hypothesis.

I_1	An inductive inference method.
i	The input vector to the process control model.
L(h,s)	The function introduced by L. G. Valiant to calculate the number of examples needed of an inductive problem, where h is any real number greater than 1 and s is a positive integer.
M	The process reference model.
M^{-1}	The inverse process model, same as C.
PD_1	The PAC dose as an input to the process reference model.
m	The limits on the training example needed in an inductive learning procedure.
P(x)	The probability function of an even.
r_k	The autocorrelation coefficient for k time period lags apart.
r_{kj}	The partial autocorrelation coefficient for k lags apart when the effect of j intervening lags has been removed;
t	The time.
Δt	The time interval.
u	The manipulated variable by the controller.
u^n	u at time interval n.
y^m	The output of the neural network plant model.
Z_t	The stationary time series.
Z_{t+k}	The data k time period ahead.
$\bar{Z}$	The mean of the stationary time series.

μ	The constant of the moving average model.
Θ_1, Θ_2	The coefficients of the moving average model.
δ, ϕ_1, ϕ_2	The coefficients of the autoregression model.
ε	The prediction error.
ε_t	The random forecast error series of the autoregression model.
δ	A boundary of the probability function P(x) which is a small number.

Chapter I

Introduction to the Artificial Intelligence Modeling Approaches

1. Introduction

As a result of steady progress in research related to the field of artificial intelligence (AI) during last decade and recent advancements of computer technology, AI techniques are gaining acceptance in the environmental engineering field in general and in water and wastewater treatment industry in particular. Although the available AI techniques are diversified in various research disciplines, they can be loosely classified into three major approaches: 1) the knowledge-based expert system; 2) the fuzzy logic engineering, and 3) the artificial neural network (ANN).

Expert systems have been defined as man and machine systems with specialized problem solving expertise (Hushon, 1990). An expert system normally consists of a database of knowledge about a particular subject area; an ability to understand the problems addressed within that subject area; and a skill pool for solving these problems. While expert system technology has existed for more than twenty-five years, environmental expert systems have only started to appear since the mid eighties. Nonetheless, its development has been rapid. As identified by Hushon, in December of 1987, there were 51 environmental expert systems. By the end of 1989, the number had risen to 80 (Hushon, 1990). The complexity of the expert systems in the environmental field has also broadened from the early narrow applications in the problem diagnosis and planning, into the area of interpretation, design and process control where a significant amount of knowledge collection and engineering is required. In recent years, the number of environmental systems has grown and diversified such that it has become difficult to track them all. In water and wastewater treatment alone, there were already 53 expert systems by 1992 dealing with process design, pumping stations, water distribution and wastewater transportation networks and plant operations, etc. (Boger, 1992).

Fuzzy logic is a technique that encodes the ambiguity of entry information into input data, composes inferences in the fuzzy logic procedure, and produces outputs labeled with a certain degree of belief that it is true. With fuzzy logic, a fact need not be true or false, but can be true to a certain degree (Russell, et al, 1994). Two key features of systems based on fuzzy logic are: 1) there is no need for equations to solve the problem and 2) human experience and behavior can be expressed as they are, that is always with ambiguity. Although fuzzy logic has been very successful in commercial applications, such as automatic transmissions, trains, video cameras and electric shavers (Russell, et al, 1994), a relatively small number of applications are sited in environmental engineering (Boger, 1992). Several applications are found in the areas of water treatment plant process control, (Baba, et al 1990 and Enbutsu, et al, 1991), water demand forecasting (Kawamura, et al, 1990) and activated sludge treatment (Tong, et al, 1980).

ANN is an AI technique simulating some biological functions of the human brain. The simple arithmetic computing units of this AI technique corresponds to neurons which are cells that perform information processing in the brain. The network as a whole corresponds to a collection of interconnected neurons. For these reasons, the networks are called neural networks (Russell, et al, 1994). The concept of ANN was initiated in the fifties and went into decline in the sixties because its complex algorithm could not be matched by the available computing power at that time. Only in the mid eighties, did its true potential become evident. With recent advances in both computer hardware, which results in the improved learning speed and software to provide easier access to the ANN technique, the ANN modeling approach is utilized in more and more industrial applications. Many promising applications of neural networks in water and wastewater treatment industries have been identified in the literature (Boger, 1992 and Schmuller, 1990).

With all these available sophisticated techniques, questions arise over which kind of modeling technique is more suitable for the problem encountered and what are the prospects associated with that technique for future advances. Although there are not simple answers to these questions, one way to describe the selection process of the AI-based modeling techniques can be presented as a decision tree shown in Figure 1-1 (Baba, et al, 1990).

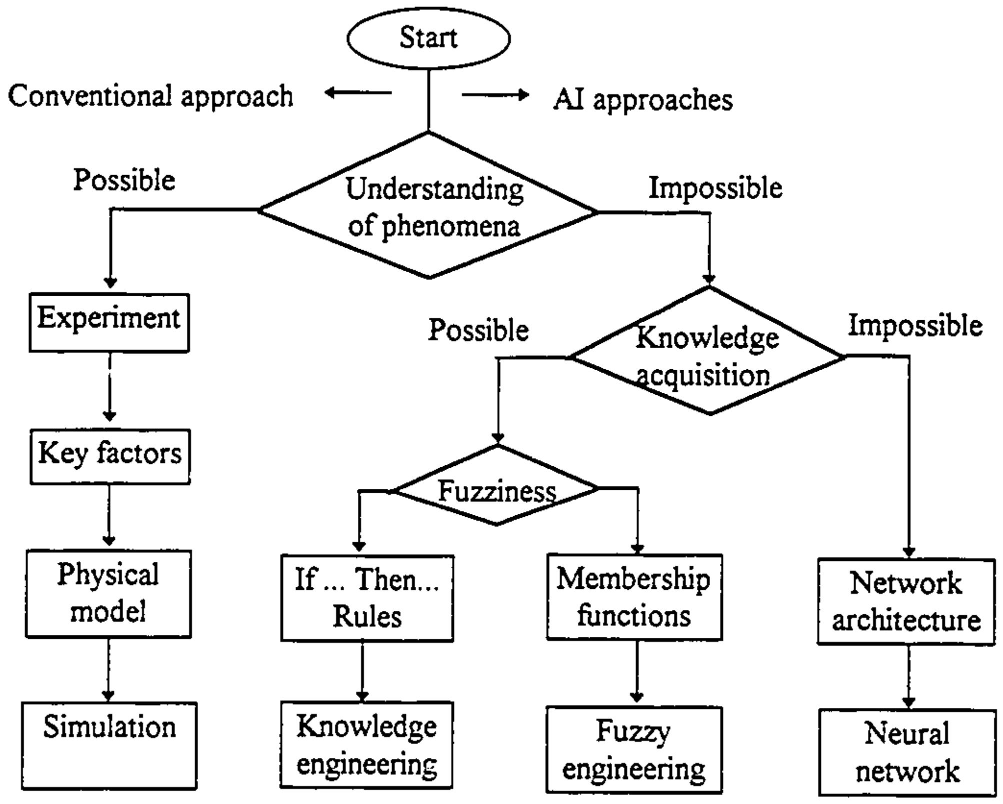


Figure 1-1 The Engineering Modeling Approaches

When applied to the environmental systems, it is often found that conventional mathematical models are hard to formulate because the experimental requirement for establishing the model parameters is expensive and difficult to perform. Compared to the conventional approach, expert system rules can be obtained more readily, but frequently the number of rules and their generality are too rudimentary for the description of complex phenomena. There is also a question of the availability of experts on certain systems and sometimes even experts may not agree on the proposed rules. The use of a fuzzy logic based system is one method to overcome the problem of rule generation from experts.

With the aid of the statistical tools, fuzzy logic systems are used to formulate the rules from historical data. Once a reliable model is generated, the analysis of its behavior might provide the insights needed for the rule elucidation.

However, a neural network which can learn from examples are believed to be a better solution than a fuzzy logic based system for extracting concepts from historical data and for many additional problems encountered in the water and wastewater plant operations in particular. By learning from plant historical data automatically, there is no human subjective interference in the learning process, no mathematical formulae are needed and the plant specific behaviors can be incorporated into the model naturally. The resulting network should be robust against process noise or instrument bias. The disadvantage of a neural network is that it requires a large data set which should describe the complete characteristics of the environmental system and it does not guarantee the success for every system because the theory behind this approach is still incomplete. Overall, there is a foreseeable potential associated with this approach and further investigation of its use in environmental applications is warranted. Investigation of the use of ANN in water treatment becomes the primary objective of this thesis.

2. Outline

Specific components of the work involved in meeting the primary objective include:

- Chapter II provides an overview of the ANN modeling approach, especially for the water and wastewater treatment research. Since much of the data found in water and wastewater treatment are time series, some key concepts related to the time series are briefly reviewed and a time series data rendering technique is subsequently introduced.
- Chapter III demonstrates the process of building an ANN model to forecast the water quality in the North Saskatchewan River from an engineering perspective. This process includes four typical steps: source data analysis, system priming, system fine-tuning and model evaluation.
- Chapter IV examines the theoretical aspects of ANN from the viewpoints of inductive learning in artificial intelligence. The computational theory and the concept of hypothesis state space are used to show that theoretically it is possible to find a probably approximately correct answer with reasonable time and space complexity. Two statistical analysis techniques—factorial design and response surface methodology —are illustrated to show this.
- Chapter V proposes and builds an ANN real-time multi-variable process control model for the Rossdale water treatment plant in Edmonton, Alberta.
- Chapter VI relates the above studies to each other, concludes the findings and suggests the directions for the further research in the applications of ANN.

References

Baba, K., Enbutsu, I., Matuzaki, H. and Nogita, S. (1990), "Intelligent Support System for Water and Sewage Treatment Plants Which Include a Past History Learning Function— Coagulation Injection Guidance System Using Neural Net Algorithm", *Proceedings of the 5th International Association on Water Pollution Research and Control Workshop*, Yokohama and Kyoto, Japan, pp. 227, Pergammon press.

Boger, Z. (1992), "Applications of Neural Networks to Water and Wastewater Treatment Plant Operation", *ISA Transactions*, Vol. 31, No. 1, pp. 25.

Enbutsu, I., Baba, K. and Hara, N. (1991), "Fuzzy Rule Extraction from a Multilayered Neural Network", *Proceedings of the International Joint Conference on Neural Networks*, Seattle. Vol. 2, pp. 461.

Hushon, J. M. (1990), "Overview of Environmental Expert Systems", Expert Systems For Environmental Applications, Edited by Judith M. Hushon, ACS Symposium Series 431, pp. 1.

Kawamura, Y. and Tsukamoto, T. (1990), "Water Demand Prediction by Fuzzy Logic", Water Supply, Vol. 10, pp. 727.

Russell, S.J. and Norvig, P. (1994), Artificial Intelligence: A Modern Approach, Prentice-Hall Inc., Englewood Cliffs, New Jersey, pp. 932.

Schmuller, J. (1990), "Neural Networks and Environmental Applications", Expert Systems For Environmental Applications, Edited by Judith M. Hushon, ACS Symposium Series 431, pp. 1.

Tong, R.M., Beck, M.B. and Latten, A. (1980), "Fuzzy Control of the Activated Sludge Wastewater Treatment Process", *Automatica*, Vol. 16, No. 6, pp. 695.

Chapter II

Introduction to Artificial Neural Network

1. Introduction

Artificial neural network (ANN) is an emerging modeling technique with many potential applications in various research fields. This chapter describes the general concepts of ANN, such as the origin of ANN, the special structural components of ANN, how ANN works and the capacity of ANN. As well, a general modeling approach by ANN for applications in environmental engineering is presented.

Since much of the raw data encountered in environmental engineering are time series data, which are usually nonlinear and complicated, applying the ANN modeling approach directly to the raw data may not obtain the best result. Sometimes, a transformation on the raw data will be necessary. Several key concepts involved in these processes are the stationary time series, the autocorrelation coefficient function and the partial auto-correlation coefficient function. These concepts are used to identify the characteristics of a time series, the type of the transformations and the proper selection of a model rendering process. Due to the importance of time series in the data, a general discussion of time series as it applies to ANN will also be presented.

2. Concepts of ANN

2.1 ANN In General

Within recent years, neural networks have experienced a resurgence of interests in the field of civil engineering (Garrett, 1992; Schmuller, 1990; Zikto, 1991) especially in the area of the forecasting capacity of neural networks (Bacha, 1992; Caire, 1992; Feldkamp, 1992; Villareal, 1992). A neural networks, as its name applies, is a piece of hardware or software (or combination of the two) that simulates what we think we know about how the brain works (Schmuller, 1990). This is a fairly loose analogy of ANN to the function of the brain. In more scientific terms, a neural network is constructed from a set of simple processing units, each capable of only a few computations such as summation and threshold logic (Garrett, 1992). The power of a neural network is due to the fact that there are many of these processing units and that each processing unit is connected to many others, in much the same way that the neurons in our brain are highly interconnected.

There are many distinct characteristics of ANN. The most exciting one is its capability of self-organization and learning. One does not program a neural network as one programs a computer. Rather, one presents a neural network with examples of the concepts to be captured. The network then internally organizes itself to be able to reconstruct the presented concepts. Two other interesting and valuable characteristics are: 1) their ability to produce correct, or nearly correct responses when presented with partially incorrect or incomplete stimuli; and 2) their ability to generalize rules from the cases on which they are trained and apply these rules to new stimuli (Garrett, 1992). Both of these latter characteristics stem from the fact that a neural network, through self-organization, develops an internal set of features that it uses to classify the stimuli presented to it and returns the expected response. By dissecting a simple network, we can learn how a neural network actually performs its task.

2.2 The ANN Structure and the Artificial Neuron

As mentioned before, a neural network is a highly interconnected network of many simple processors as depicted in Figure 2-1. In a typical neural network, these simple processors, named nodes or artificial neurons, are organized in a three-layer fashion: an input layer, a hidden layer (or layers) and an output layer. The input layer takes a pattern of stimulation from the outside world. This input pattern is then processed through the hidden layer(s) which is the major computation unit of the model. Finally, an output pattern is expressed in the neuron network's own version, as its understanding to stimulation from the outside world is generated. This output pattern is then presented back to the outside world through the output layer.

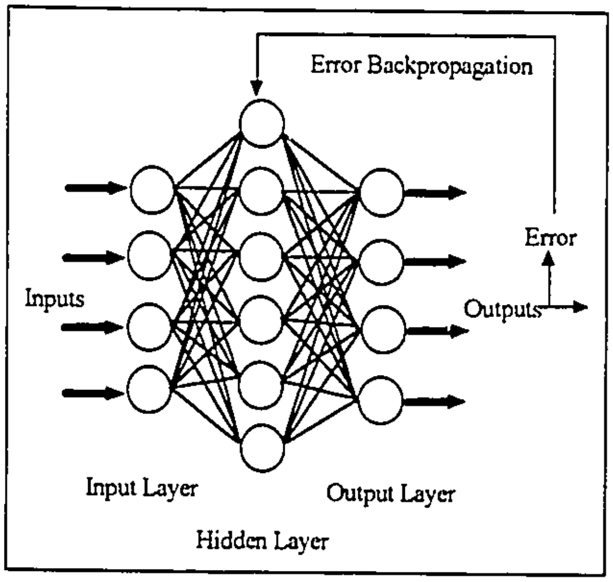


Figure 2-1 A Simple Neural Network

Being a basic component of the neural network, the artificial neuron consists of a processing unit and some input and output ports (Figure 2-2). The processing unit receives information or stimulus from the other artificial neurons through the input ports and exports the processed information through the output ports to the next connected neurons.

The two components of the processing unit are the weight (or the activation level) and the transfer function. The weight is a number between 0 and 1 which is also called the activation level. The transfer function defines how the input received by a unit is combined with its current level of activation to compute a new level of activation which is stored as the new weight of the neuron. The transfer function is usually expressed mathematically as follows:

$$a_{i\text{-new}} = F_i\,(a_{i\text{-old}}, N_i) \qquad (1)$$

where $a_{i\text{-new}}$ = the new activation level of the ith neuron;

$a_{i\text{-old}}$ = the old activation level of the ith neuron;

N_i = the inputs to the neurons.

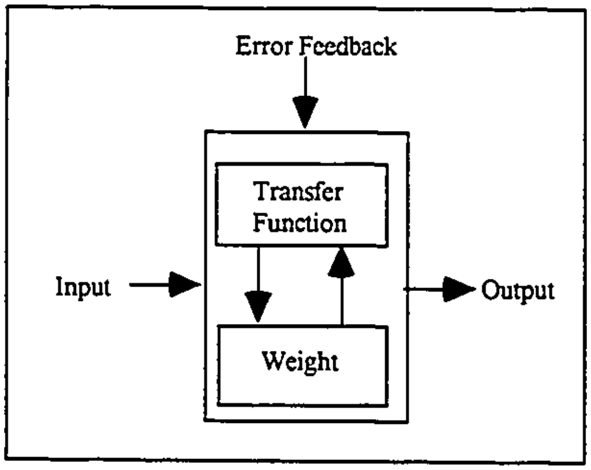


Figure 2-2 An Artificial Neuron

The most common transfer function used is the sigmoid function (Figure 2-3). The sigmoid function, which is continuous and differentiable (an important property for use in backpropagation learning), and is used to maintain the values of activation for a neuron within the bounds of 0 and 1.

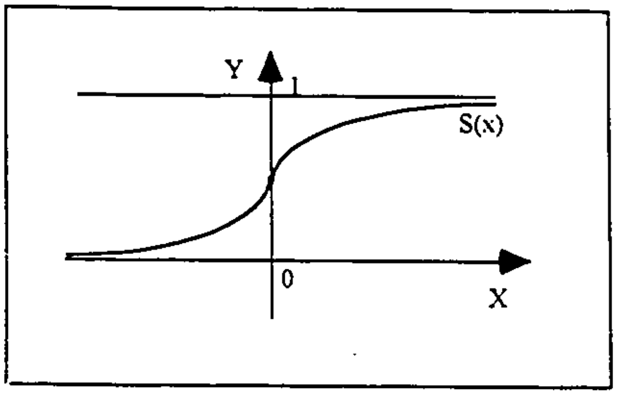


Figure 2-3 The S-shaped Sigmoid Function

2.3 Error Backpropagation

Error backpropagation is a method to train a neural network. In other words, it adjusts the weight in each artificial neuron for better results. More specifically, only the weights of the hidden layer neurons are adjusted. This process is mainly composed of two steps. First the differences between the observed outputs from the output-layer neurons and the desired outputs are calculated. Second these differences are propagated back to the neurons which send output values to them and the weights in those neurons are reset. These two steps are repeated until the difference between the neural network output and the desired output is negligible. The weight adjustment is complex and involves the differentiation of the transfer function in each neuron. More details are discussed in Appendix I.

2.4 Development of a Backpropagation Neural Network

The development of a backpropagation neural network involves five steps (Garrett, 1992): 1) the verification of the suitability of the backpropagation network; 2) the collection of a large set of historical data or a complete representative set of data for a pattern; 3) the selection of the architectural details; 4) the selection of the training set data and the actual network training; and 5) the adjustment of the architecture of the network for optimum results. Usually there is a control and feedback iteration process among step 3 to step 5. This process is illustrated in Figure 2-4. Step 6 and 7 illustrated in Figure 2-4 are the post-neural network modeling processes. Rendering the forecasting result from the neuron network is necessary when the magnitude of the continuous data are highly fluctuating such as random peaks in the water parameters in time series data. After the rendering process, the model is checked one more time to ensure the accuracy of the rendered model.

2.4.1 Algorithm Verification

The verification of the suitability of the backpropagation neural network approach to the problem is extremely important as this approach is not generally applicable to all types of problems. The backpropagation neural network in general is best used to recognize features within a input pattern and it uses these features to classify the pattern into one of a set of predefined classes.

Successful ANN applications have the following characteristics (Bowen, 1991): 1) the algorithm to solve the problem is unknown or expensive to discover; 2) heuristics or rules to solve the problem are unknown or perhaps difficult to enunciate; and 3) the application is data intensive and a variety of data describing the subjects are available. It should also be determined that: 1) conventional computer technology is unsuitable or inadequate; 2) the application requires qualitative or complex quantitative reasoning; 3) the solution is derived from inter-dependent or correlated factors which are difficult or impossible to quantify; and 4) the data are available and the corresponding known solution can be derived.

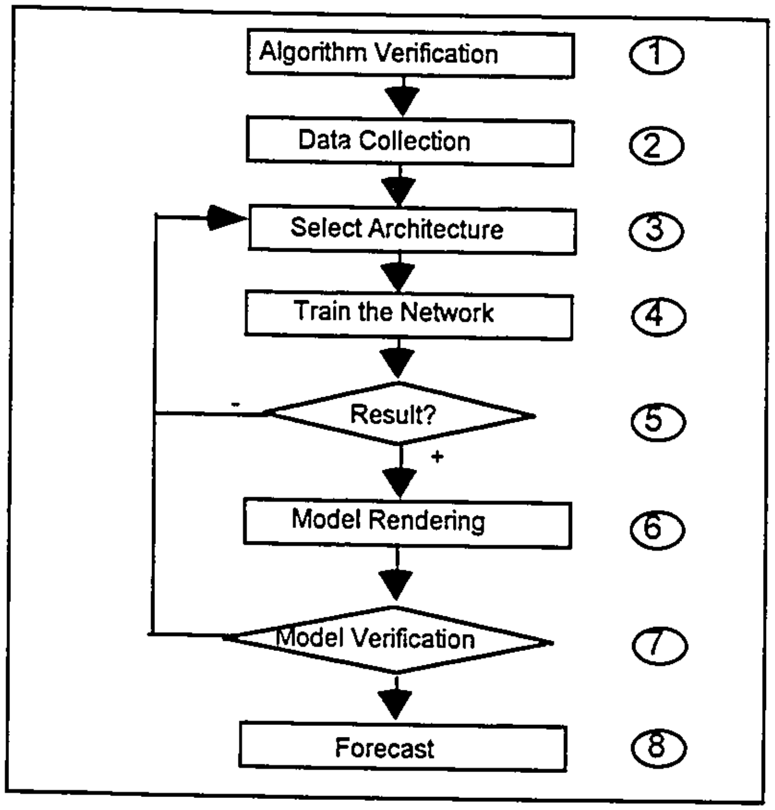


Figure 2-4 The ANN Modeling Approach

However, ANN will not be suitable for the cases in which precise mathematical computations are required, computational procedures must be explained, or adequate representative data are not available.

2.4.2 Neural Network Architecture Design

The selection of an architecture of the neural network is very much dependent on the problem that the neural network is intended to solve. Typically, it involves the selection of the number of hidden layers, the number of the neurons in each layer and the transfer function for the neurons. Although the selection of the neural network architecture is nearly a trial and error process, there are some guidelines available[1]. Based on experience and the literature (Schmuller, 1990; Garrett 1992; Villareal 1992; Tang 1991), the architecture of a network is dictated by the representation chosen from the input and the output of the problem. It is generally recommended that for a relatively simple domain, the number of the neurons in the hidden layer should not exceed three times the number of the neurons in the input layer. Generally, ANN with two hidden layers can extract much more complicated features of the study domain. However, the number of the hidden layers seldom exceeds two. These ideas are further explained as follows.

As mentioned above, the architecture of a neural network is determined, in large part, by the format selected for its input and output. The remaining two questions are: (1) what, if any, transformations should be performed on the source data before it is presented to the network; and (2) how many hidden layers should be used in the network and of what sizes should they be.

The answer to the first question can be obtained from a close examination of the transfer function (Figure 2-3) in each neuron. Values on the y axis represent node output resulting from the corresponding input indicated on the x axis. Note that the sigmoid function is asymptotic about the x axis and the range of y values are between zero and one. This means that as the x values become very small, the corresponding y values will approach

[1] A detailed ANN architecture searching procedure will be explained in chapter 4.

zero. Similarly, as the x values approach infinity, the y values approach one. Because the learning function depends on the first derivative of this transfer function, x values much above 5 or below -5 will have derivatives close to zero and the network will not learn. In fact, the best learning takes place when the y values falls between 0.2 to 0.8. To take advantage of this fact, all the input data should be normalized between 0.2 to 0.8. Usually normalization is performed using calculations with the mean and standard deviation of the data set.

The question of choosing the right configuration for the hidden layer is somewhat more difficult because the relationship between network performance, the number and the size of hidden layers is not well understood. There are, however, some principles which can be used as a guide (Villarreal, 1992). The first principle is generalization versus convergence. These are two aspects of network behavior which often work against one another. Generalization is the ability of the network to produce reasonable results for the unknown situation once the training process has been completed. Convergence is simply the ability of the network to learn the training data within the error tolerance specified for the problem. In general, the more hidden neurons presented, the greater the likelihood tnat the network will converge. However, if too many hidden units are used, the network will generalize poorly, "memorizing" the training data rather than focusing on its significant features. The goal is then to use as many hidden neurons as needed to ensure convergence without using so many as to inhibit generalization (Tang, 1991).

The second principle is the increase in learning time versus the number of the hidden layers. Networks tend to learn backwards, meaning that the learning cascades down from the output layer to the input layer. The more layers which are sandwiched between the input and output layers, the longer such cascading has to travel and the slower the network will learn. Consequently, it is often best to start with network designs which have few hidden layers (one is usually a good first try). One exception to this rule is when experience indicates otherwise.

2.4.3 Training the Network

There are three important concepts for the neural network training. First, when a new network is to be trained, its initial activation level in each neuron must be randomized. If they are set to the same number, the error backpropagation will not work[2]. Second, after the weights are randomized, each training set is presented to the network in cycles. The network readjusts itself by the error backpropagation process introduced in Section 2.3 in each cycle. This cycle of presenting the same set of training data to the network will not stop until a specified error tolerance is obtained. Third, there is an important parameter--the learning rate--which must be set before the training. The learning rate is the amount the weights may be modified in any given training cycle. It affects the speed of convergence. If the learning rate is greater than 0.5, the weight will be changed more drastically and this may cause the optimum combination of weights to be overshot resulting in oscillations about the optimum (Tang 1991). If the learning rate is small ($<$ 0.2), the weights will change in smaller increments thus causing the system to converge more slowly but with little oscillation. Usually in the preliminary run, the learning rate is set to be large to speed up the process. An error trend can be found from the preliminary run and thus allow the selection of a more appropriate learning rate in the following runs.

Once a network architecture has been chosen, the next step is to select the typical cases and go on to network training. The selection of the training cases for a neural network can be equated with the examples used by an instructor in the course for teaching a subject for which there is no theory, only examples. If these examples are not representative for the

[2] This is due to the delta rule in the error propagation process. The delta rule is explained in Appendix I.

data pattern or are not the complete coverage of the various features and sub-features that the network is likely to encounter, the network will fail in solving the unknown features[3]. There are also two important questions to be answered for the training process: how and in what quantity should the training data set be selected, and how long should the data be trained in the training cycle.

The answer to the first question is the randomization in the data extraction process and the completeness in the data sampling process. The randomized data extraction guarantees that the training set and the testing set will contain full features of the collected data set. The completeness of the sample data can be obtained only if the collected data represent the complete study domain. This is usually difficult to verify.

There are two possible ways to solve this dilemma. First, if the distribution of the sample data is known, some statistical tests are avaliable to justify the completeness of the data. Second, if the distribution is unknown, which occurs most of the time, then expert knowledge must be relied upon. Usually, the expert knowledge is qualitative. For example, in the development of NETtalk (Karayiannis, 1993), a neural network was developed which learned the relationship between the English language and phonetics, and 1000 of most commonly used words in English language were selected as a training set. The expert knowledge indicated that a 1000 word set was the basic requirement to provide a sufficient range of English to phonetic translation to cover a large percentage of the rules necessary to read or pronounce a word. Thus, NETtalk was not allowed to select a small set of words, say 50 words, to prevent the undergeneralization of the network which would result in the system being unable to pronounce untrained words correctly. Neither

[3] Notice there is a significant difference between the unknown feature and the unknown situation for the neuron network problem solving. An unknown situation means a case has never been used to train the neuron network but it is within the pattern of the data and the neural network will be able to solve it within a acceptable margin of error. An unknown feature means the irrelevant set of data to the designated data pattern and the prediction from the network will fail with certainty.

was NETtalk allowed to handle a training set with more than 20,000 words, which could then cause difficulty for NETtalk to make distinctions and fail to adjust its weights correctly. In short, the objective is to pick a large enough training set to provide a thorough representation of the feature space without overwhelming the network. The expert knowledge can also be quantitative if the previous experience with the problem is extensive and can be represented by a learning curve. An excellent review of the learning curve approach is given by Russell (1994).

Today, there is still no clear answer to the second question. Knowing when to stop is the biggest challenge in producing the best ANN model. If the training time is too little, the network will not learn enough. If the training time is too much, the network will overgeneralize, in which it will either learn the noise or memorize the training patterns. The network simply becomes a lookup table for the patterns it learned and has little power to recognize a new pattern that it has never seen in the training process. Typically, the problem is that the criterion for stopping the training is case dependent. In this thesis, a systematic approach was developed to handle this dilemma. The detailed information about the appraoch and the theory behind are described in Chapter 4

2.4.4 Model Verification

In validating a trained neural network, there are two crucial processes. The first step is to determine how well the network performs on input patterns for which it was not trained. The second step is to use statistical tools to determine whether the trained network is a suitable model for the problem. This step has usually been ignored by the many researchers in this field. The most common way to check the model is to examine the residuals (Box, 1976).

The first step is basically a test to see how well the network can handle the unknown situations. One must be careful not to present input which contains unknown features. Otherwise, the result may be discouraging. Again there is a conceptual difference between the unknown situation and the unknown features. This step is completed by applying the testing set data through the trained network and comparing the actual values with the predicted results from the trained network. If the overall prediction error is less than a specified number, the model is accepted.

Model diagnostic checking is performed to test the adequacy and closeness of fit of the model to the data. The most common method used is residual analysis. With the plots of the residuals versus the dependent variable and the independent variables, the model adequacy or the lack of fit can be detected from the graph intuitively. Three aspects of the residuals which must be checked are: whether the mean of the residual is zero, whether the residual variance is constant and whether the shape of the residual plot suggests lack of fit. Usually, these questions can be answered by simply plotting the residual graphs which include residual versus dependent variable, residual versus time variable and residual versus independent variables. Also, other statistical tests such as the t test can be used to test the significance and the relationships of the model parameters. If any of the tests are unacceptable, the model must be respecified and the previous steps such as neural network design must be repeated.

3. Time Series Analysis and the Model Rendering Process

For a clearly defined and closed domain such as the case studied in Chapter 5, the ANN modeling technique is quite adequate and the rendering process may not be necessary. However, if the study domain is big, open and loosely defined such as the case studied in Chapter 3, the ANN model may not be as efficient as it could be in a small closed domain because the completeness of the sample data is sacrificed by trying to find a small representable set of parameters to describe an open domain. In this case, the statisitical model rendering process may serve as a post-optimzation process for the ANN model.

The principle behind this statistical rendering process is to enhance the local variance of time series data. The statistical model keeps tracking the variance of the time series data in a small interval, which is called the local variance. If this variance is under a certain threshold, the model does little to change the local variance. If the variance is significantly over the threshold, the model tends to increase the variance by adding a correction component to each time series value within the interval. The net result of this process is that the model tends to smooth out the time series data in the low local variance intervals and increases the peak values when the local variances are significant.

The statistical model used for the rendering process is the Box-Jenkins Model. Several literature reviews (Vemuri, 1994; Farmer, 1987; Zaknich, 1991; Deppish 1991) have reported that both the Box-Jenkins model and the ANN are the major time series analysis tools with reasonable complexity and reliability. However, none of these papers combined these two methods together to enhance the modeling quality. A case study illustrating this rendering process can be found in author's thesis proposal which described ANN research activity in Quebec City. A detailed approach to select the time series model for the rendering process is explained in the following sections.

3.1 The Rendering Model Identification Procedure

The selection of Box-Jenkins model consists of several steps (Gaynor, 1994; Bowerwan 1979). To properly identify the suitable model, several characteristics of the time series data are very helpful: the type of time series (stationary or nonstationary), the autocorrelation coefficient and the partial autocorrelation coefficient. Usually two phases are involved in selection. First, it is necessary to identify the type of the time series. Usually only stationary time series can be modeled and sometimes transformation of the data into a stationary time series may be necessary. The next phase is to determine the tentative model by analyzing the autocorrelation and partial autocorrelation functions of the time series data.

3.1.1 Stationary Time Series

A stationary time series is one that does not contain a trend: that is, it fluctuates around a constant mean. Two methods can be used to identify a stationary time series: plotting the data or calculating the autocorrelation coefficients. Theoretically, a nonseasonal time series is stationary if the autocorrelations are all zero (indicating random error) or if they differ from zero only for the first few lags (Gaynor, 1994). Because sample autocorrelations are dealt with and there are sampling errors involved, these small lags are interpreted as "statistically" equal to zero.

If the original series does contain a trend, it can be transformed into a series without a trend by taking first or second differences of the data (Box, 1976). The method of taking first differences of the data is to simply subtract the values of two adjacent observations in the series:

$$Z_t = \Delta Y_t = Y_t - Y_{t-1} \tag{2}$$

where Y is the original data series. If the Z_t is still not a stationary time series, then it is necessary to take the difference again. These results are then called the second difference.

3.1.2 Autocorrelation Coefficient Function

Once the stationary series has been obtained, the next step is to determine a tentative ARIMA model. It is necessary to analyze the behavior of the autocorrelation and partial autocorrelation functions at this stage.

The autocorrelation coefficient measures the relationship, or correlation, between a set of observations and a lagged set of observations in a time series. Given the time series (such as the Z series in Equation 2), the autocorrelation between Z_t and Z_{t+k} measures the correlation between the pairs (Z_1, Z_{1+k}), (Z_2, Z_{2+k}), ...,(Z_n, Z_{n+k}). The sample autocorrelation (r_k) is computed by:

$$r_k = \frac{\Sigma(Z_t - \overline{Z})(Z_{t+k} - \overline{Z})}{\Sigma(Z_t - \overline{Z})^2} \tag{3}$$

where $\overline{Z}$ = the mean of the stationary data;

Z_t = the data from the stationary time series;

Z_{t+k} = the data k time period ahead;

r_k = the measure of the relationship between the two sets of data.

If the autocorrelation coefficient function (ACF) has been obtained, the next step is to classify its behavior. The autocorrelations for a stationary, nonseasonal time series usually differs from zero only for the first few lags. Their behaviors can then be categorized into three series (Gaynor, 1994). First, the ACFs for all lags can be zero indicating random errors. This is a *"no trend"* category. Second, only the ACFs for first 1, 2 and/or 3 are quite large and significant. These large coefficients are known as spikes. This category is named the *"cut off."* Third, if an ACF does not cut off, but rather decreases to zero in a quick, steady fashion, it belongs to the category of *"dying down."* These three categories can be also applied to the partial autocorrelation function introduced next.

3.1.3 Partial Autocorrelation Coefficient Function (PACF)

A partial correlation coefficient is the measure of the relationship between two variables when the effect of other variables has been removed or held constant. This adjustment is made to see if the correlation between Z_t and Z_{t+k} is due to the intervening variables or if indeed there is something else causing the relationship. The behavior of the PACF is used along with the ACF to identify a tentative ARIMA model (see next section).

The sample PACF can be computed using the following formula:

$$r_{kk} = \frac{r_k - \sum (r_{k-1,j})(r_{k-j})}{1 - \sum (r_{k-1,j})(r_j)} \qquad (4)$$

where r_k = the autocorrelation coefficient for k lags apart;

r_{kj} = the partial autocorrelation coefficient for k lags apart when the effect of j intervening lags has been removed; calculated by $r_{kj} = r_{k-1,j} - (r_{kk})(r_{k-1,k-j})$

3.1.4 The Box-Jenkins Models

Box-Jenkins models can only describe or represent stationary series or series that have been made stationary by differencing. The models fall into one of the three following categories.

(1) Moving average model (MA). By this model, the present stationary observation (Z_t) is a linear function of a mean, a present forecast error and a set of past forecast errors. A

moving average model tries to estimate the forecast error by a weighted average of the most recent forecast errors:

$$Z_t = \mu + \varepsilon_t - \Theta_1\varepsilon_{t-1} - \Theta_2\varepsilon_{t-2} - \ldots - \Theta_q\varepsilon_{t-q} \tag{5}$$

where Z_t	= the present stationary observations;
ε_t	= the white noise which is unknown and expected to be zero;
ε_{t-1}, ε_{t-2}	= past forecast errors;
μ, Θ_1, Θ_2	= the constant and moving average coefficients.

(2) Autoregressive model (AR). By this model, Z_t is a linear function of past stationary observations Z_{t-1},Z_{t-2},.... Since the trend has been removed in a stationary series, errors or residuals are what is left for modeling. By applying regression analysis to lagged values Z_{t-1},Z_{t-2},... of the stationary time series, we obtained an autoregressive model:

$$Z_t = \delta + \phi_1 Z_{t-1} + \phi_2 Z_{t-2} + \ldots + \varepsilon_t \tag{6}$$

where Z_t	= the present stationary observation;
Z_{t-1}, Z_{t-2}	= past stationary observations (usually no more than two);
δ, ϕ_1, ϕ_2	= the parameters from the regression analysis;
ε_t	= the random forecast error for the present time period (expected to be zero).

(3) Mixed model. By this model, Z_t is a combination of an autoregressive and a moving average model, that is, it is a linear function of past stationary observations and present and past forecasting errors:

$$Z = \delta + \phi Z + \phi Z + \ldots + \varepsilon - \Theta \varepsilon - \Theta \varepsilon - \ldots \tag{7}$$

where the notations are the same as in equation (5) and (6).

To select an appropriate model, in theory (Box, 1976) the following generalizations can be made: if the ACF cuts off and the PACF dies down, this is a MA model; if the ACF dies down and PACF cuts off, this is an AR model; and if both ACF and PACF die down, it is a mixed model. An example demonstrating this procedure can be found in the author's thesis proposal.

3.2 Model Rendering

The purpose of the rendering process is to magnify the peaks (both positive and negative) in the data series. Once a stationary time series is obtained by applying the first difference, the trend in the original time series data is eliminated. To magnify the peaks, the AR model is usually the first choice because this model is very sensitive to the local variance of the residuals. This peak magnification process depends on the rate of change in variance in the neighborhood. Thus, when the preliminary predictions from the neural network are fed into the autoregressive model, the local variances are magnified and the peaks are generated.

Usually the rendering process must be performed repeatedly on the data for several times to yield a good result. There are three general rules for how many times the rendering process should be applied to the data set. First, the rendering process should stop when the rendered peaks hit a maximum value (a data ceiling) of the data set. This ceiling value can be obtained from a statistical analysis of the historical data. Usually the value of the 95th percentile is a good estimation of the ceiling value. Second, similar to the ceiling value, a floor value of the data should also be considered. Usually, it is the 5th percentile of the historical data. However, in the water treatment process for example, extreme low colour values are not significant. Thus, in the rendering process, if the floor value is hit by means of a negative peak, the negative peak is replaced by the floor value and the next round of the rendering process is continued. Third, sometimes, neither the floor value nor the ceiling value is of concern and it is the percentage of the peak which is important.

Thus, the rendering process will continue until a certain percentage of the peaks are obtained.

4. Conclusion

In this chapter, a detailed review of two important concepts --ANN and Box-Jenkin time series models -- were completed. ANN is a different modeling approach compared to the traditional modeling approaches. It is a biological paradigm to the function of the human brain. To simulate the human brain, a network of artificial neurons are proposed which results in complex nature of the model structure and the algorithm of ANN. Thus, the whole modeling process is achieved through computer simulation. Nevertheless, this computer simulation process must be supervised in order to achieve the purpose of the modeling. Therefore, step-by-step ANN modeling procedures were presented in this chapter. To illustrate the capacity of ANN modeling, the main features of ANN are summarized below: 1) it has capacity of self-learning to solve a problem; 2) problem solving is based on sample data and learning mechanisms; 3) it does not require computer knowledge representation, logical inferencing schemes, or statistical algorithm to develop a solution; 4) it can learn on-line real-time or be trained off-line by a sample data set; and 5) it does require an appropriate architecture with sufficient capacity and paradigmatic learning/training scheme.

In this chapter, an interesting new application for the Box-Jenkin time series analysis was also introduced. Time series analysis is integrated into the ANN modeling process as a pre-model or post-model rendering process. The important characters of the autoregressive model -- the ability to keep the overall trend of the data, the sensitivity to the local variances and the capacity to magnify the local variances -- makes it a perfect tool for generating the data peaks. Although this rendering step is optional in the whole

ANN modeling approach, it does provide a safeguard to the quality of the ANN modeling whenever necessary and makes the ANN modeling approach more promising.

References

Bacha, H. and Meyer, W. (1992), "A Neural Network Architecture for Load Forecasting", *Proceedings of the IEEE International Conference on Neural Networks*, volume II, pp. 442.

Bowen, B.A. (1991), A Neural Network design Methodology: Considerations and Issues for Design and Project Management, *AGARD Lecture Series 179 - Artificial Neural Network Approaches in Guidance and Control*, 7 Rue Ancelle 92200 Neuilly Sur Seine, France, pp.3-1 -pp.3-20.

Box, G.E. and Jenkins, G.M. (1976), Time Series Analysis: Forecasting and Control, revised edition, Holder-Day, Inc. San Francisco, pp. 575.

Bowerwan, B.L. and O'Connell, R.T. (1979), Time Series and Forecasting, Duxbury Press, A division of Wadsworth, Inc., pp. 481.

Caire, P. Hatabian, G. and Muller, C. (1992), "Progress in Forecasting by Neural Networks", *Proceedings of the IEEE International Conference on Neural Networks*, volume IV, pp. 304.

Chakraborty, K. (1992), "Forecasting the Behavior of Multivariate Time Series Using Neural Networks", *Neural Networks*, Vol.5, No.6, 1992, pp.961-970.

Deppisch, J., Bauer, H.U.. and Geisel, T. (1991), "Hierarchical Training of Neural Networks and Prediction of Chaotic Time Series", *Physics Letters A*, Vol. 158, Nos.1 and 2, 1991, pp. 57-62.

Farmer, J.D. and Sidorowich, J.J. (1987), "Predicting Chaotic Time Series", *Physical Review Letters*, Vol.59, No.8, 1987, pp. 845-848.

Feldkamp, L.A., Puskorius, G.V., Davis, L.I. and Yuan F. (1992), "Strategies and Issues in Applications of Neural Networks", *Proceedings of the IEEE International Conference on Neural Networks*, IV-540.

Garrett, J.H.Jr.,Ghaboussi, J. and Wu, X. (1992), "Neural Networks", in *Expert Systems for Civil Engineers: Knowledge Representation*, Edited by Robert H. Allen, American Society of Civil Engineering, New York, pp. 12.

Garrett, J.H.,Jr., Ranjitham S. and Eheart J. W. (1992), "Application of Neural Network to Groundwater Remediation", in *Expert Systems for Civil Engineers: Knowledge Representation*, Edited by Robert H. Allen, American Society of Civil Engineering, New York, pp. 57.

Gaynor, P.E. and Kirkpatrick, R.C. (1994), Introduction to Time Series Modeling and Forecasting in Business and Economics, MaGraw-Hill Inc., pp. 625.

Jayawardena, A. W. and Lai, F. (1989), "Time Series Analysis of Water Quality Data In Pearl River, China", Journal of Environmental Engineering, Vol 115, No.3 June, 1989, ASCE.

Karayiannis, A.B. and venetsannpoulos, A.N. (1993), Artificial Neural Nets, Learning Algorithms, Performance Evaluations and Application, Kluwer Academic Publisher, Boston, pp. 439.

Lowe, D. and Webb, A.R. (1991), "Time Series Prediction by Adaptive Networks: A Dynamical Systems Perspective", *IEEE Proceedings-F*, Vol. 138, No.1, 1991, pp17-25.

Nielson, R.H. (1991), "Theory of the Backpropagation Neural network", *Proceedings of the IEEE International Conference on Neural Networks*, volume I, pp. 593.

Ossenbruggen, P.J. (1985),"Time Series Models for Treatment of Surface Waters", *Journal of the Environmental Engineering*, Vol. 111, No.1, February, 1985, ASCE.

Park, D.C. (1991), "Electric Load Forecasting Using An Artificial Neural Networks", *IEEE Trans. Power Systems*, Vol. 6, No.2, 1991, pp.442-449.

Rogers, R.D. and Vemuri V. (1994), "Time Series and the Forecasting Problem", *Time Series Vs Neural Network in Forecasting*, IEEE Proceedings, 1994.

Russell, S.J. and Norvig, P. (1994), Artificial Intelligence: A Modern Approach, Prentice-Hall Inc., Englewood Cliffs, New Jersey, pp. 932.

Sawyer, C.N., MaCarty, P.L. and Parka, G.F. (1994), Chemistry for Environmental Engineering, 4th Edition, Mc-Grow Hill Inc., pp659.

Schmuller, J. (1990), "Neural Networks and Environmental Applications", in *Expert Systems for Environmental Applications*, edited by Judith M. Hushon, ACS Symposium Series 431, American Chemical Society, Washington, DC, pp. 235.

Tang, Z., Almeida, C.D. and Fishwick, P.A. (1991), "Time Series Forecasting Using Neural Networks vs. Box-Jenkins Methodology", *Simulation*, Vol.57, No.5, 1991, pp303-310.

Villareal, J. and Baffes, P. (1992), "Time Series Prediction Using Neural Networks", in *Expert Systems for Civil Engineers: Knowledge Representation*, Edited by Robert H. Allen, American Society of Civil Engineering, New York.

Weigend, A.S., Rumelhart, D.E. and Huberman, B.A. (1991), "Generalization by Weight-Elimination with Application to Forecasting", *Advances in Neural Information Processing Systems 3*, 1991, pp. 875-882.

Zaknich, A., deSilva, C.J.S. and Attikiouzel, Y. (1991), "A Modified Probabilistic Neural Network for Nonlinear Time Series Analysis", *Proc. IJCNN*, 1991, pp. 1530-1535.

Zikto, V. (1991), "Prediction of Biodegradability of Organic Chemicals by An Artificial Neural Network", *Chemosphere*, Vol. 23, No.3 , pp. 305-312.

Chapter III

Forecasting Raw Water Quality Parameters for the North Saskatchewan River By Neural Network Modeling

1. Introduction

Many community water supplies are obtained from surface water sources. To produce high quality drinking water from surface water, the contaminants in the raw water such as sediments, colour-causing organic substances, other chemical contaminants and microorganisms must be removed by the water treatment process. To accomplish this purpose, modern water treatment plant consists of a series of physical and chemical treatment processes. The performance of these processes is highly related to the characteristics of the raw water entering the plant. To optimize the treatment processes and thus provide a good quality potable water in an economical manner, the ability to predict the raw water quality over time is desired by the water treatment industry. This would allow advance warning of changes in raw water quality, which require alternation of process conditions. Rather than rely on the traditional reactionary approach where process changes are made once process performance begins to deteriorate, advance warning of changes would allow planning and optimization of process conditions to meet expected changes in raw water quality. In this paper, a modeling technique called artificial neural network (ANN) modeling approach is introduced to forecast the quality of the raw water parameters in a river.

The main purpose of this paper is to demonstrate a powerful modeling technique through a case study for the Rossdale Water Treatment Plant in Edmonton, Alberta, Canada. The case study not only highlights the use of this modeling technique for predicting raw water quality in a river which is the first step in developing the overall process model, but also shows the general application of ANN modeling for water treatment area. Some optimization issues involved in the model development are also discussed.

2. Project Description

The source water for the Rossdale Water Treatment Plant (WTP) is the North Saskatchewan River. The North Saskatchewan River is a major tributary in the Saskatchewan-Nelson River system (Figure 3-1). A large proportion of the upstream watershed is uninhabited mountains and forests, with little municipal or industrial development. From Drayton Valley downstream, the percentage of agricultural land use increases. The major source of upstream contamination is runoff from forested and agricultural lands. The overall water quality is considered good (Milne, 1991).

The Rossdale WTP is a conventional surface water treatment facility consisting of alum coagulation and flocculation, sedimentation, lime softening, disinfection and filtration. The process most impacted by rapid changes in raw water quality is the coagulation and flocculation process because it is the first treatment process and must contend with the full range of variation in the raw water quality. The subsequent downstream processes tend to be buffered by the upstream processes. A review of literature suggests that the coagulation

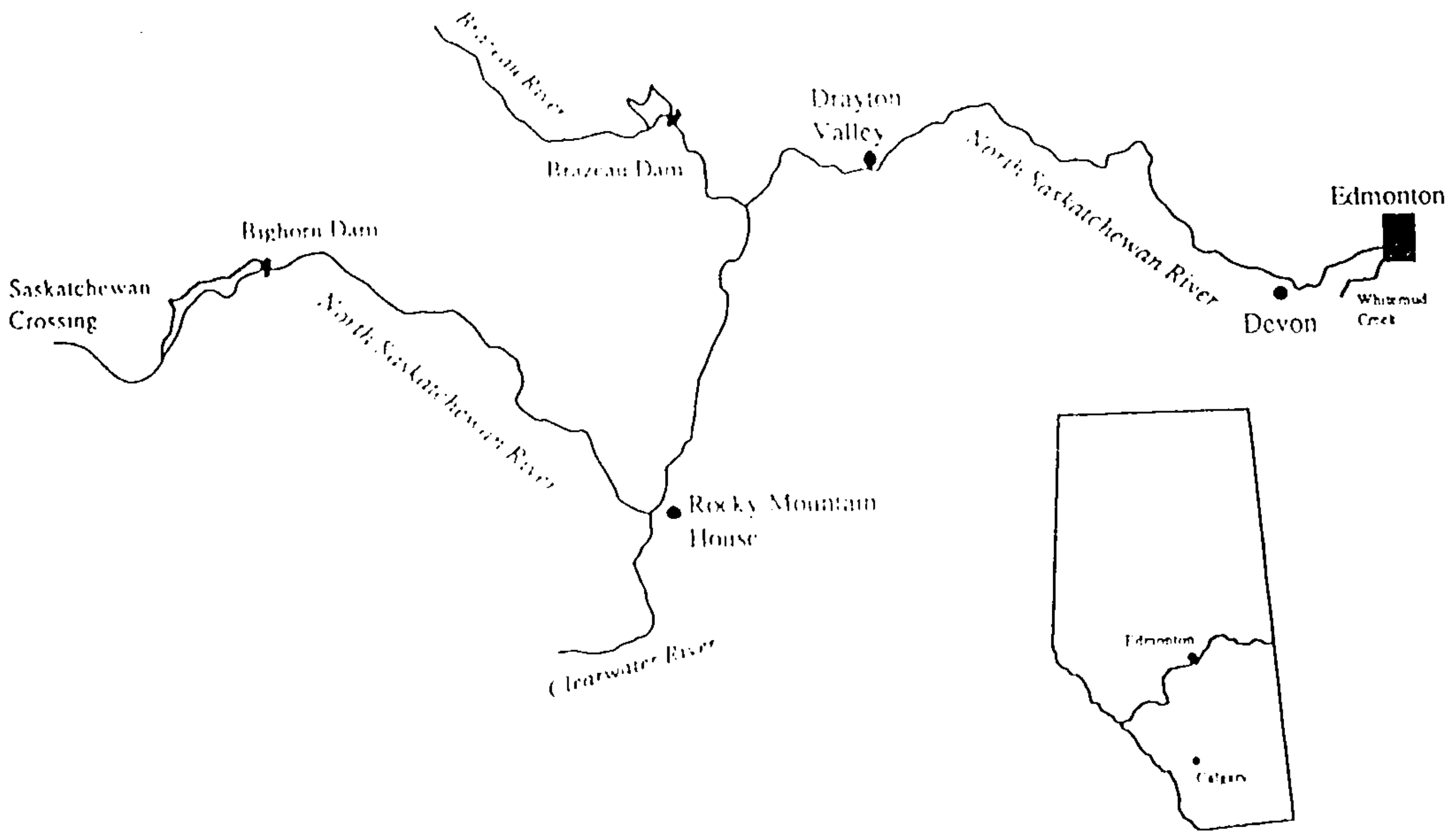


Figure 3-1 The North Saskatchewan River Upstream of Edmonton

process in the water treatment plant is significantly affected by the raw water colour and turbidity[1]. Several models have been developed that link these key raw water parameters to the alum dosage (Ossenbruggen, 1985). It was also found at the Rossdale WTP that based on past experience, operators control the alum dosage mainly by the value of the raw water colour from the previous day. However, the large fluctuation of the raw water colour especially in the early spring and mid-summer renders this approach impractical. Thus, a need to predict the raw water colour to aid Rossdale WTP personnel in making process decisions is the primary objective of the present study. Colour is also an important process monitoring parameter in water treatment, as it is used as a surrogate parameter for the organic content of the water. Singer et al. (1981) also found that colour could be used as a surrogate parameter for trihalomathane formation potential. Given the increasing concern and regulation associated with disinfection by-products, knowledge of the organic content of the raw water and optimization of treatment processes to remove these organics is a significant challenge to the water treatment industry.

3. Artificial Neural Network Overview

A number of modeling techniques could be used to accomplish the primary objective of this study. However, due to the complexity of the many factors which can affect river water quality and the availability of long term water quality records at the Rossdale WTP, the artificial neural network approach was chosen.

[1] Raw water colour is measured in the Rossdale WTP by the Spectrophotometric Method illustrated in Standard Methods for the Examination of Water and Wastewater. The raw water is first filtered to remove particles greater than 0.45 µm in size, then the colour value is measured at the wavelength of 400 nm and the pH of 7.6. Raw water turbidity is measured by the Nephelometric Method from *Standard Methods*.

Artificial neural network, as its name applies, is a modeling technique that simulates the human brain's problem solving process. Just as humans apply knowledge gained from past experience to new problems or situations, a neural network takes previously solved examples, looks for patterns in these examples, learns these patterns, and develops the ability to correctly classify new patterns. By this way, the network internally organizes itself to learn the presented concepts. It does not require either programming, logical inference schemes, or statistical algorithm to develop a solution. However, it does require expert knowledge of the parameters which govern the process.

An ANN is a highly interconnected network of many simple processors (Figure 3-2). These simple processors, named artificial neurons, are organized into an input layer, a hidden layer (or layers) and an output layer. The neurons are connected by weights. The input layer takes a pattern of stimulation from the outside world and passes the pattern to the hidden layer(s) where it is processed. Finally an output pattern is generated and presented by the output layer.

A backpropagation neural network[2] learns by adjusting the interconnection weights between layers. The answers which the network produces are repeatedly compared with the correct answers, and each time the connecting weights are adjusted slightly in the direction of the correct answers. Eventually, if the problem can be learned, a stable set of

[2] Backpropagation network is one of the many different ANN schemes and is the one used by this application. For more information on backpropagation network, see the Neural Network Paradigms by P.K. Simpson (1991).

weights adaptively evolves and will produce good answers for all of the sample decisions or predictions.

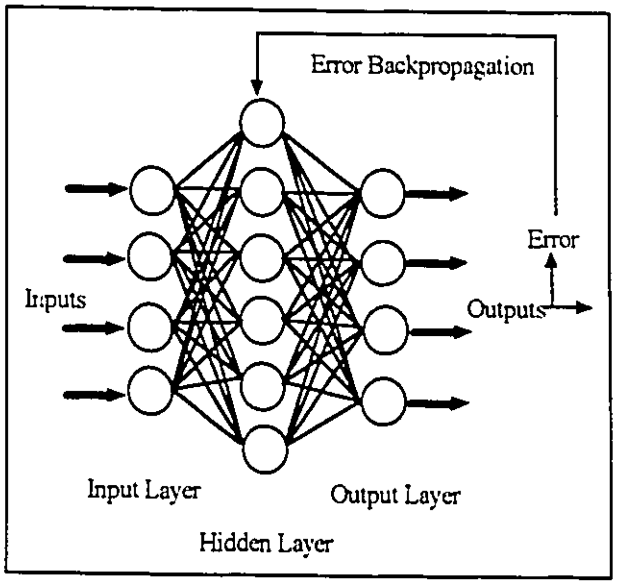


Figure 3-2 A Simple Neural Network

Successful ANN applications have the following characteristics (Bowen, 1991): 1) the algorithm to solve the problem is unknown or expensive to discover; 2) heuristics or rules to solve the problem are unknown or perhaps difficult to enunciate; and 3) the application is data intensive and a variety of data describing the subjects are available. On the other hand, ANN will not be suitable for the cases in which precise mathematical computations are required, computational procedures must be explained, or adequate representative data are not available.

4 Neural Network Modeling In Water Treatment

Compared to the conventional modeling approaches in water treatment, ANN modeling has a number of distinct advantages. First, no mathematical algorithm is required to build the model. The network simply learns from the sample data and generates a black box type relationship. The major requirement is the knowledge of all important factors governing the process. In water treatment processes, many uncertainties exist because of the micro-scale physical and chemical reactions involved. Conventional modeling requires mathematical algorithms to describe these uncertainties, where a neural network simply learns the process based on past experience shown through the data. In a sense, it performs like an operator who learns the best process conditions for a given raw water quality based on past experience.

Second, the ANN modeling approach is fast and flexible. Since there is no need to build a physical model and generally no special lab tests are required, the scale-up process is eliminated. Even a complicated neural network model can be completed relatively quickly once the data are collected. In addition, if some changes in the treatment process are necessary, the network can be quickly adjusted to the new processes through model retraining, in which the new data describing the new processes are added to the network's learning procedure. With conventional models, process changes can render the mathematical algorithms used invalid.

Third, the ANN can handle non-linear relationship well due to its inherent non-linear data structure and computation process (Simpson, 1991). The time series of many water quality parameters vary in a non-linear fashion. One such example is the daily variation of the raw water colour at the Rossdale Plant intake (Figure 3-3). The capacity for modeling the non-linear relationship makes the neural network modeling well suited for water quality modeling and forecasting.

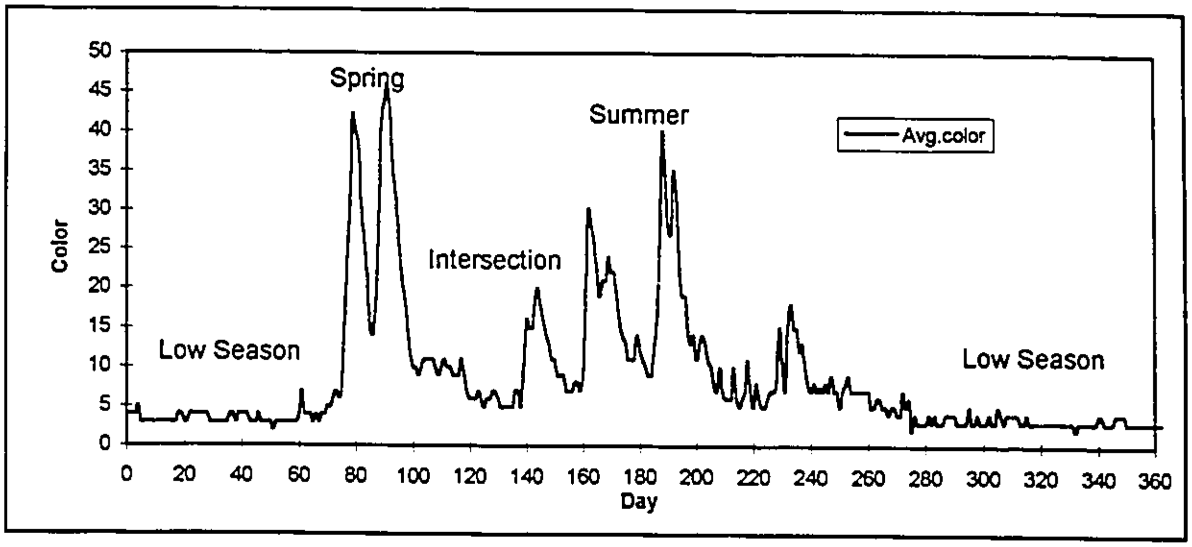


Figure 3-3 A Sample Annual Cycle Pattern of Colour in 1994

Fourth, neural networks tend to be inherently fault-tolerant, as their data structure is loosely organized and there is no boundary limit on the input parameters. Depending on the significance of the parameters in the neural network architecture, the impact of the false input can be either blocked within the data structure or reduced in magnitude due to the non-linear calculations inside the data structure. This fault-tolerance feature is critical for the real-time process control.

Fifth, the water treatment industry has traditionally kept excellent records of process parameters. By using this existing information to train the ANN model, the experiences of the operators and the valuable lab tests are incorporated into the model automatically.

The possible applications of ANN modeling in water treatment is fairly large. ANN modeling can also provide solutions to problems such as both short term and long term raw water quality forecasting, process optimization, automatic process control, daily water demand forecast, management decision making, etc.

5 Raw Water Colour Forecasting

Although the actual process to develop a successful network is quite complex, a simplified version includes four major phases: source data analysis, system priming, system fine-tuning and model evaluation. To demonstrate the process of building an ANN model for raw water colour forecasting, these four steps are discussed separately with some optimization issues.

5.1 Source Data Analysis

The major purpose of source data analysis is to select a proper architecture for the ANN model. During the analysis of the source data collected from the North Saskatchewan River, two distinct features were found. First, the daily raw water colour values are strongly autocorrelated. Second, an annual cyclic pattern exists for the data.

A strong autocorrelation in the data series means that the raw water colour value of the present day is closely related to the colour values of the previous days, especially to the colour value of the day before. A recurrent type neural network model will be immediately suggested by this feature. However, this type of model actually failed. Instead of predicting tomorrow's colour value from today's data, it returned a value very close to the present day's colour value. Further research found that it was the strong autocorrelation that actually confused the network and caused the network model to put greater weights on the trend of the data rather than the detail individual differences. Thus, the model lost the ability to distinguish the small individual difference between today's colour value and yesterday's colour value.

To solve this problem, the first difference is applied to the original time series to weaken (but not eliminate) the effect of autocorrelation[3]. By this approach, the architecture of the neural network model was adjusted to the semi-recurrent type and the one-day-lag problem was solved. The new forecasting process is: first, the difference of today's raw water colour from tomorrow's raw water colour is predicted by the neural network model; then, the predicted difference is added back to today's colour to obtain the whole prediction value of tomorrow's colour.

As the result of this new approach, the accuracy of the forecasting is improved significantly. This can be illustrated through an example:

[3] The theoretical base of this approach is explained in *Time Series Analysis: Forecasting and Control* by G.E. Box and G. M. Jenkins (1976).

Assume a raw water colour series C = {12, 11, 14, 16}.

Apply first difference on C $\Rightarrow$ A new series ΔC, in which $\Delta C_n = C_n - C_{n-1}$ ={-1, 3, 2}.

Assume the predicted results from the neural network model for ΔC is $\Delta C'$ = {-0.8, 2.7, 1.9}.

Then, the average error in predicting $\Delta C = \dfrac{\sum(\Delta C_n - \Delta C_n')/\Delta C_n \times 100\%}{n} = 12\%$. (1)

Apply the formula $C_n' = C_{n-1} + \Delta C'$ on C $\Rightarrow$ the predicted colour series C' = {12, 11.2, 13.7, 15.9}.

The average error in predicting $C = \dfrac{\sum(C_n - C_n')/C_n \times 100\%}{n} = 1.5\%$. (2)

Since $C_n' = C_{n-1} + \Delta C' = C_{n-1} + \Delta C + \varepsilon$ (where ε is the prediction error), Equation 1 can be rewritten as:

The average error in predicting $\Delta C = \dfrac{\sum \frac{\varepsilon}{\Delta C_n} \times 100\%}{n}$. (3)

and Equation 2 can be rewritten as:

The average error in predicting $C = \dfrac{\sum \frac{\varepsilon}{(C_{n-1} + \Delta C_n)} \times 100\%}{n}$ (4)

From Equation 3 and 4, it can be seen that $C_{n-1} + \Delta C_n >> \Delta C_n > \varepsilon$. Although there is an average of 12% error in predicting the colour difference, the average error in predicting the whole colour value is only 1.5%. The error in prediction is greatly reduced. Notice that this approach is only suitable for the strongly autocorrelated time series otherwise the condition of $C_{n-1} + \Delta C_n >> \Delta C_n$ can not be guaranteed.

The second important feature of the original colour series is that there are four distinctive "seasons" that exist in the annual cycle pattern (Figure 3-3). Despite the differences in the starting time of these seasons over the five year sampling period, the length and the distinct characteristics of these seasons are quite consistent. Thus, it is possible to separate the annual data into four seasons: (1) the early spring high colour period which lasts for about a month and a half; (2) the summer high colour period which usually starts in early June and finishes at early August; (3) the spring and summer intersection with the colour values in the medium range; and (4) the rest of time in which the colour values are considered low. The annual cycle pattern of the colour values is a key factor which influences many aspects of the model design, especially in the design of system indexes.

5.2 System Priming

The main task of the system priming is to find a set of input parameters which will allow the model to converge. Usually two kinds of input parameters are included in the design of the ANN model: the cause-effect parameters and the system indexes. The cause-effect parameters provide the clues to identify the potential results. The system indexes describe the distinctive features of the particular system to be modeled and provide additional constraints or conditions to further improve the accuracy of the model. Since most of the river water quality parameters show a strong autocorrelation, a third type of parameter, the lag N day time series, is also added to the input parameters for the raw water colour forecasting model.

The cause-effect parameters included in the colour forecasting model are the turbidity, the river volume and the rain precipitation upstream. The derivatives of the time series of these parameters also belong to this category. For example, the change rate of river volume used in the model is the first derivatives of the original river volume time series.

Turbidity is the most important parameter in describing the river colour. Literature reviews suggest that natural colour exists in water primarily as negatively charged colloidal particles, which adsorb significant amount of organic substances such as tannins, humic acid and humates on their surfaces and cause the colour of the water (Sawyer, et al, 1994). Turbidity provides a measurement of the colloidal particles in the river water. The actual modeling results also prove this point. In calculating the contribution factor of each input parameter, the turbidity is the most dominant single factor which contributes more than 55% of the time in predicting each colour value (column 5 and 6 in Figure 3-4). Thus, turbidity is the primary input parameter.

The rain precipitation and the river volume are two factors that either directly influence the value of turbidity or interact with the value of the turbidity. During summer, high intensity rainfalls will wash organic substances into stream and river resulting in high turbidity and high colour downstream. Usually, river volume and turbidity are positively related. However, in case of point source pollution, high river volume will actually dilute the pollution and reduce the turbidity and colour. Both rainfall and river volume are included in the model as the supplemental parameters to turbidity.

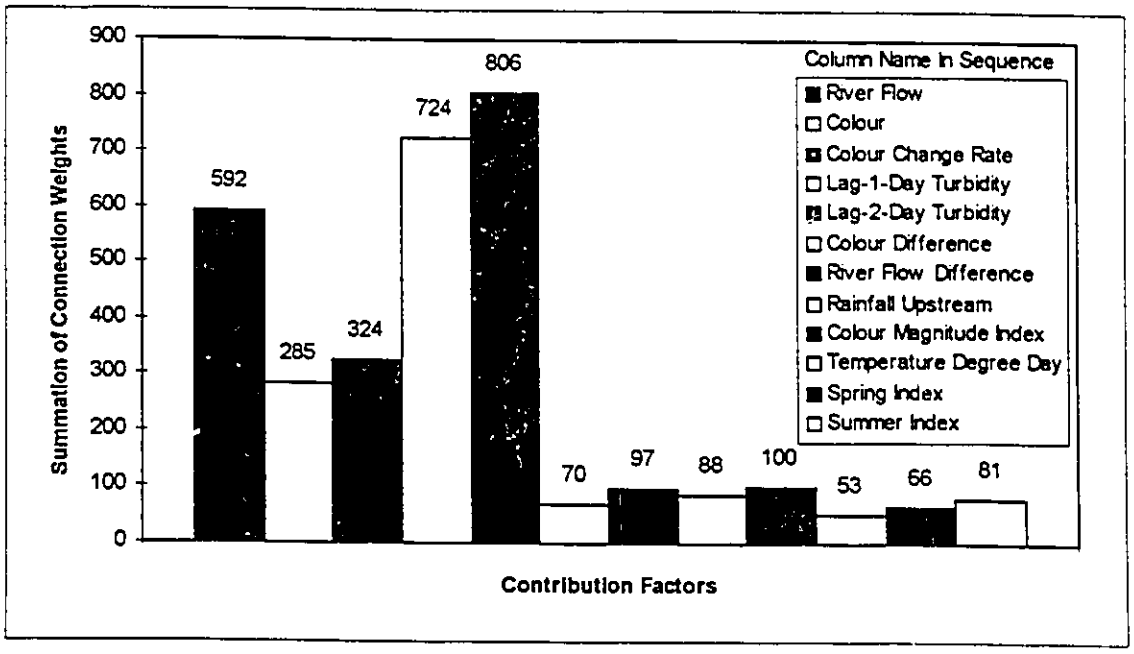


Figure 3-4 A Sample Contribution Factor Graph for all the Input Parameters

Other physical factors such as pH and alkalinity of the river water were also considered but excluded from the model. For example, an increase in pH value of the river water will generally result in an increase in colour intensity. But since the variance of pH of the North Saskatchewan River is small, it does not serve as an effective parameter. The same argument applies to alkalinity when its time series is normalized.

Several system indices were designed to describe the annual pattern of the colour values: the spring index, the summer index, the colour magnitude index and the temperature degree day. The first three indices are the general indices summarized from the sample data collected to indicate the peak colour seasons in a year. Their values are designed to reflect the concept of non-seasonal days, "gray" seasonal days and the typical seasonal days. Non-seasonal days are assigned a value of 0. The strong seasonal days are assigned a value of 1. Those gray seasonal days are assigned values between 0 and 1 according to their grayness. These indices are assigned to every day of a year. For example, the typical summer peak colour season occurs from early June to mid July, thus the summer index for this period is 1. The two weeks immediately before and the four weeks immediately after this typical summer colour season are the "gray" summer days. They are assigned a value between 0 and 1 statistically. Of course, the summer index in the rest of the year is 0.

Most of the colour seasons except spring have relatively fixed initial and ending days. In order to detect the initial day of the spring season, the temperature degree day index was used. From the data analysis, it was found that the high colour in early spring is closely associated with the snow melt. It is the different snow melting period in each year (shifts around approximately 20 - 30 days over the years analyzed) that results in the fluctuation of the initial day of the spring colour season. The temperature degree days were used to detect the initial day of the snow melting time.

Due to the strong autocorrelation of the raw water quality parameters in this case and the semi-recurrent architectural design of the ANN model, the colour values and the turbidity values in the previous days should be considered as the potential input parameters as well. Care must be taken in determining how many previous days' values should be used. If there are too many input parameters having the similar values, they will interact with each other and become noise. Normally, it is recommended that an autocorrelation table should be calculated for up to a lag of ten days for these parameters. Then, the number of previous days' values to be used can be determined according to the magnitude of the autocorrelation coefficients. The change rate of the time series of these parameters is also worth exploring. In this case, the change rate of the colour time series was selected as the input parameter.

To summarize, in order to predict the colour difference at N day, the twelve input parameters in the final colour forecasting model are: the river volume at Goldbar Sampling Station at N-1 day, the change rate of river volume at N-1 day, the raw water colour value at the treatment plant intake at N-1 and N-2 day, the change rate of colour at N-1 day, the turbidity difference at the plant intake at N-1 day and at N-2 day, the rain precipitation upstream in Rocky Mountain House[4] at N-3 day, the temperature degree day at Edmonton International Airport at N-1 day, the colour magnitude index, the spring index and the summer index.

5.3 System Fine-tuning

The primary tasks in this phase are: find a set of representative data for the process; split the data into a training set and a testing set; lead the model to learn from the training data; and finally, use the testing data to measure the success of the model. The optimization issues in this stage are pattern recognition and pattern selection.

As mentioned in the source data analysis, the most favourable representation of the source data for prediction is the colour difference which can be obtained by applying the first

[4] The Rocky Mountain House locates about 2 days upstream with respect to the river velocity.

difference[5] to the original colour time series. One typical plot of the colour difference is shown in Figure 3-5. It can be seen that the trend in the original colour time series (Figure 3-3) is eliminated and the new time series fluctuates around the mean of zero. These fluctuation components are called the random prediction errors or the white noise in the Box-Jenkins models (Box, et al, 1976). However, in this case, the time series of colour difference contain not only the random errors but also the distinct features of the system presented as the peaks in Figure 3-5.

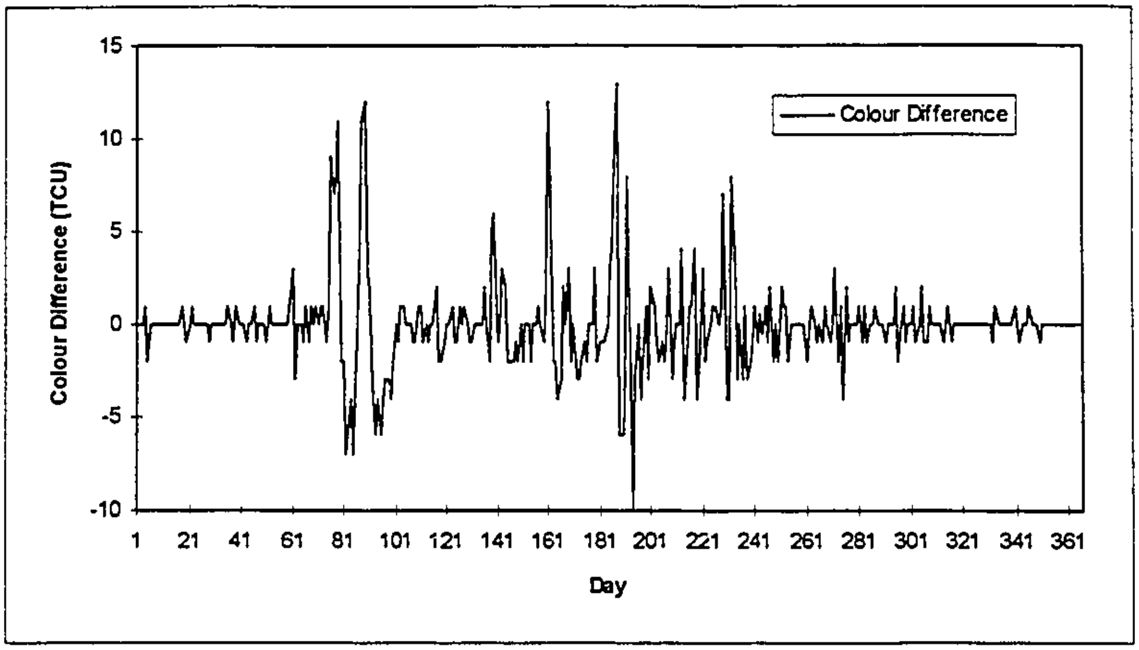


Figure 3-5 The Time Series of Colour Difference in 1994

Theoretically, in order to improve the generalization, it is necessary for the ANN model to have the capacity of recognizing both the random error and the distinct features of the

[5] The method of taking first differences of the data is to simply subtract the values of two adjacent observations in the time series.

system. However, the random error is also considered as the noise in the training process due to its randomness in pattern matching. If too many random error patterns[6] are introduced in the training process, it will be difficult to obtain an optimized network because of the interference. Conversely, too few random noise patterns may result in the incompleteness of the model to recognize the true random noise patterns and subsequently fail to predict it. Two approaches to address this issue have been tested.

The first approach is to treat every pattern in the training data set as an individual distinct feature of the system and use them to train the model. There is no difference between random noise or any other feature in this approach and the error tolerance of the model design is also fully tested. The second approach is to select a proper amount of random noise patterns from the training data in an iterative way. This approach follows three steps: 1) include only the feature patterns in the training set and build the first ANN model; 2) use this model to predict the rest of the random noise patterns in the training data. If the error of prediction for a random noise pattern is over a threshold value, then that random noise pattern will be considered as a necessary feature pattern to be included in the training data set. A new and bigger training data set is then generated and a new ANN model is subsequently trained; and 3) repeat Step 2 until the model is sufficiently accurate over the whole training data set. By this approach, the number of training patterns is increased iteratively and hopefully will reach an optimum number on which the optimum model can be built.

[6] A pattern includes all the input parameters and the corresponding output parameter.

For this project, daily data over five years were collected. The general training set includes the data from 1990 to 1993 and the testing set included the data in 1994. Although the criteria to distinguish the feature patterns from the random errors is system dependent, it is recommended to use the accuracy of instrument measurement as in this case. Since the accuracy in raw water colour measurement is 1 TCU, the error thresholds are ±1. Any noise pattern with a prediction error over this range is considered as a new feature pattern and should be included in the training data set.

Both approaches have advantages and disadvantages. If the assumption that every individual pattern is a distinct pattern is true, then the first approach will be preferred because the more patterns that are added to the training set, the more experience the ANN model can learn and generally a higher accuracy of the prediction can be expected. However, the training time increases exponentially with additional patterns in the training set. If the assumption is false, the second approach will be a better alternative, as it only extracts the real distinctive features of the system to form a training set. This will save time, and provide a more generalized solution by filtering out the true random noise.

5.4 Model Evaluation

Prediction error assessment includes two aspects: the error assessment within the model and the error assessment across the model. The purpose of the error assessment within the model is to evaluate how closely the neural network model can recognize the patterns presented in the training data set and its ability to reproduce them. The main statistical tools are the root mean square error[7] and the R square.

The error assessment across the models are more complicated. Although sets of minimum training errors and testing errors are available for the candidate models, they do not represent every aspect of the models as they are average values calculated over the whole data set. For example, in this project, special attention is put on the colour value of 10 units (TCU) because once the colour value exceeds 10 units, the treatment process requires adjustment. Thus, the error in predicting the peak values will be more significant. Some candidate models could have relatively good minimum training and testing errors but may have large peak prediction error. One partial solution to this problem is the graphical analysis of the final candidates.

5.5 Results

A total of four candidate models were developed for the raw water colour forecasting. Table 3-1 shows the results for the four models tested. Model A was built by the first approach discussed in the system fine-tuning section. Model B, C and D were constructed by the second approach with different iteration numbers. The minimum training error and the minimum testing error in Table 3-1 are the root mean square errors calculated for the

[7] Root mean square error is a statistical measure of the differences between the actual output values and the output values predicted by the network model. It is a mean over the whole data set of the square of the actual value minus the predicted value. The errors are squared to penalize the large errors and to cancel the effect of the positive and negative values of the differences.

normalized[8] training and testing data. These error parameters represent how well the models fit the training and testing data.

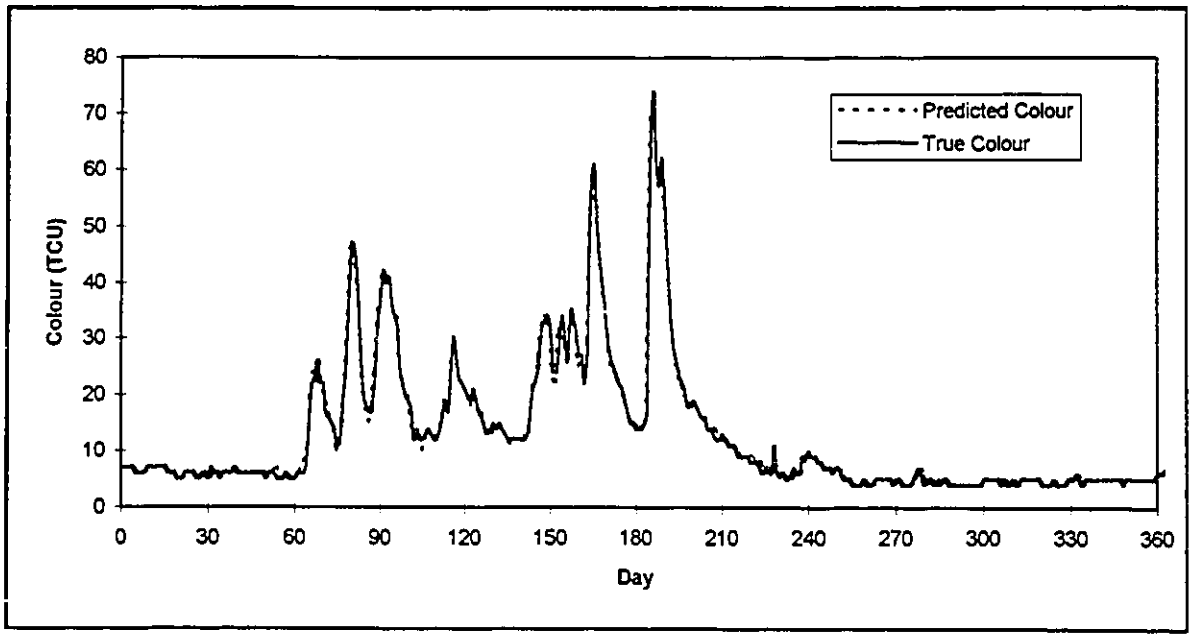


Figure 3-6 The Training Set Data Prediction Results (1990) For Model A

For all models, it was found that when the minimum training error is close to 1.5E-4, the prediction for the training data is satisfactory as shown in Figure 3-6. This figure (a section of colour prediction for the training set data by Model A) highlights that the accuracy in prediction is good, as the measured and the predicted values are very close. The model was able to both predict the peak colour values adequately and exhibited, no lag phenomenon with all significant colour jumps being predicted one day ahead.

8 Normalization is the process to scale the input data for each input parameter into the range of (0,1) linearly.

Table 3-1 The Statistical Data for the Candidate Models

Candidate Model	A	B	C	D
Building Method	Approach 1	Approach 2	Approach 2	Approach 2
Minimum Training Error	0.0001457	0.0002560	0.0001340	0.0001628
Minimum Testing Error	0.0010005	0.0009612	0.0012186	0.0010717
R Squared	0.896	0.876	0.935	0.921
Root Mean Squared error(TCU	1.0	1.3	0.9	1.0
Mean Absolute Error (TCU)	0.6	0.8	0.6	0.7
Correlation Coefficient r	0.946	0.938	0.967	0.960

Next the trained models were used to predict the 1994 data, the testing data set. Since the data in the testing set were never seen by the model, the predictions on these testing data can test the adequacy and the potential of the model. In Table 3-1, although the minimum testing errors of all the candidate models are approximate 10 times higher than their corresponding minimum training error in magnitude, they still predict the trend of the colour variance. As an example, the prediction by Model A is presented in Figure 3-7. Model A reasonably predicts all the peaks and only two small lags are detected at Day 185 and Day 233. It is suspected that these two small lags may be the new feature patterns which emerged in 1994. Thus, the neural network model did not recognize them. Overall, Model A recognizes 355 patterns out of 365 patterns for 1994 which it had never encountered before. This indicates that although there were a few instances where model predictions did not match actual results, overall the model predicted raw water quality successfully.

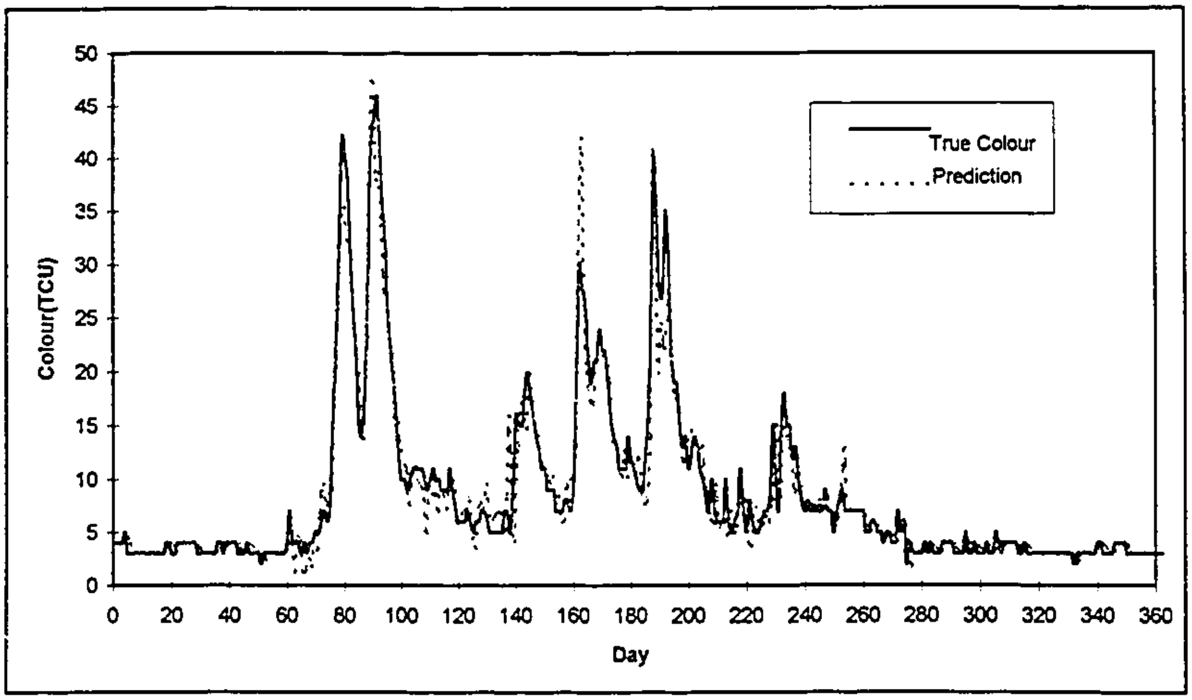


Figure 3-7 The Colour Predictions by Candidate Model A (1994)

In order to check the goodness of fit of candidate model A, a residual analysis was performed. The plot of the residual versus time is shown in Figure 3-8. The plot of the residual versus true color value is shown in Figure 3-9. In both figures, it is evident that the mean of the residuals is approximately zero. This is further proved by the value of standard deviation of the residuals which is about 0.3 colour unit. In both graphs, no special trend occurs for the distribution pattern other than the random scatter pattern.

Although the density of the residuals are quite different in some regions (Figure 3-8 and 9), it is due to the different number of sampling data available in that region and not because of the bias. Overall, it can be concluded that model A is adequate, though the variance of the residuals in the middle of the year tends to be higher than that of those at the end (Figure 3-8). This is due to the high fluctuation in colour values in the middle of the year, which is much more difficult to predict.

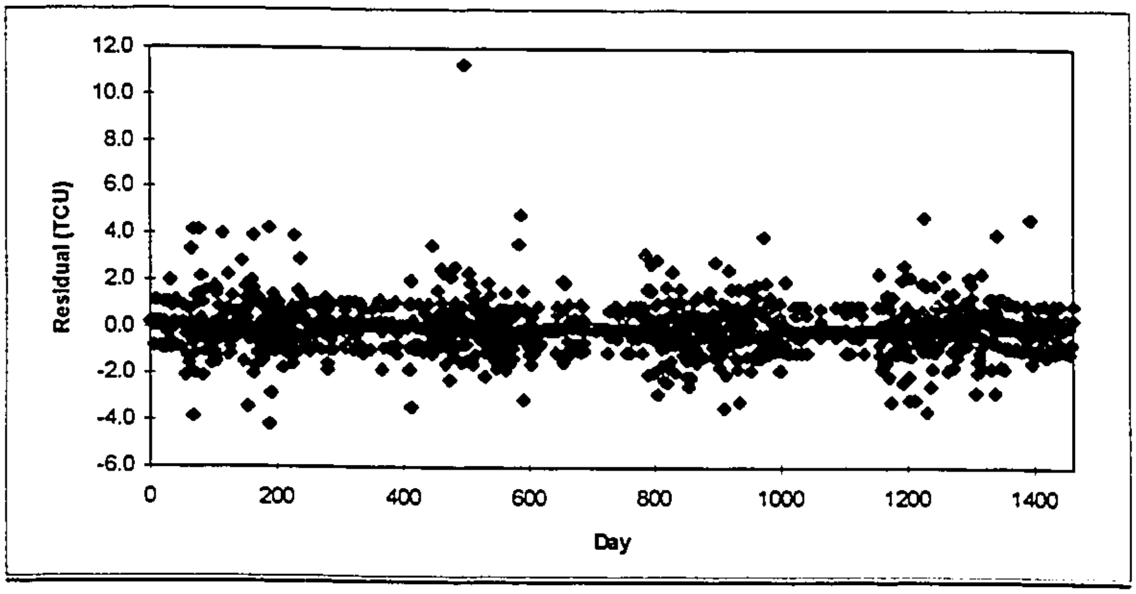


Figure 3-8 The Residual Versus Time (1990-1993)

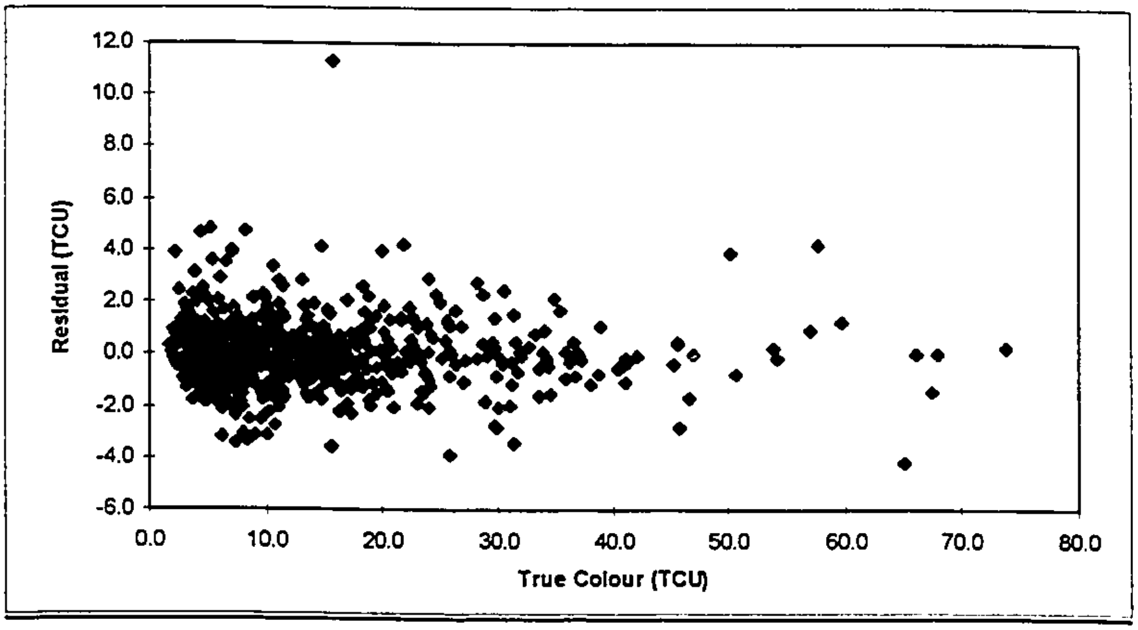


Figure 3-9 The Residuals Versus True Colour (1990-1993)

6. Conclusion And Recommendations

With the detail graphical analysis and the final evaluations of the significant factors listed in Table 3-1, Model A is recommended as the best model tested. However, it should be pointed out that Model A may not be the best possible model. Each of the four candidate models could be a local optimum. To find the global optimum, more trials are needed. It is suggested that the new trials must be done by the factorial design approach to allow the effective construction of the error surface and the final search for the global optimum.

Although the technique introduced here is still in its preliminary stage, the ANN modeling approach seems promising. It is a fast and flexible way to incorporate multiple input parameters and target parameters into one model. It is this aggregation of multiple information sources that provide the main power for this approach to gain many new capacities and a significant improvement in the accuracy in modeling over other conventional approaches.

Currently, several potential optimization techniques for this ANN modeling approach are being studied. The results of this study will provide a solid foundation for the further research in establishing models for the real time operations such as the computerized coagulation dosing control. The future development of computer technology will make this approach even more reliable and readily accessible for a wider range of applications in the water treatment and water quality control area.

References

Bowen, B.A. (1991), "A Neural Network Design Methodology: Considerations and Issues for Design and Project Management", *AGARD Lecture Series 179 - Artificial Neural Network Approaches in Guidance and Control*, 7 Rue Ancelle 92200 Neuilly Sur Seine, France, pp.3-1 -pp.3-20.

Box, G.E. and Jenkins, G.M. (1976), Time Series Analysis: Forecasting and Control, revised edition, Holder-Day Inc., San Francisco, pp. 575.

Milne, G. D. (1991), Chlorine Decay in a Large River, M.Sc. Thesis for Department of Civil Engineering, University of Alberta. pp. 190.

Ossenbruggen, P.J. (1985), "Time Series Models for Treatment of Surface Waters", *Journal of Environmental Engineering*, ASCE, Vol. 111, No.1, pp 27- 44.

Sawyer, C.N., MaCarty, P.L. and Parka, G.F. (1994), Chemistry for Environmental Engineering, 4th Edition, Mc-Grow Hill Inc., pp. 659.

Simpson, P.K. (1991), "Neural Network Paradigms", *AGARD Lecture Series 179 - Artificial Neural Network Approaches in Guidance and Control*, 7 Rue Ancelle 92200 Neuilly Sur Seine, France, pp.2-1 -pp.3-33.

Singer, P.C., Barry III, J.J., Palen, G.M. and Scrivner, A.E. (1981), "Trihalomethane Formation in North Carolina Drinking Waters", *Journal American Water Works Association*, Vol. 72, No.8, pp 392-402.

Chapter IV

Solving Engineering Problems With Artificial Neural Network In An Inductive Heuristic Approach

1. Introduction

In recent years, the applications of the artificial neural network (ANN) modeling approach have flourished in various engineering research fields. However, this promising modeling technique has been limited by a lack of solid theoretical explanation behind the apparent successes. In this paper, an examination of the ANN modeling process from the aspects of the inductive learning process is performed as an attempt to provide a possible explanation to these successes. Based on this explanation, a heuristic approach of finding a best possible neural network solution for a closed domain engineering problem was developed.

This paper will begin with the concept of inductive learning and two key ideas associated with the general paradigm of inductive inference established by Gold in his fundamental paper (Gold, 1967): identification in the limit and identification by enumeration. These two ideas serve as fundamental concepts in examining the ANN modeling approach. The idea of identification in the limit will be further expanded by the computational theory developed in the recent years to show that an inductive learning process, such as the ANN modeling approach, can be possible if sufficient learning samples are provided. With the idea of identification by enumeration, a successful neural network model is considered as an enumerated rule which agrees with all the learning data, and this rule exists somewhere in a rule hypothesis space which is the summation of all possible rules. The process to discover the best neural network model is then equivalent to a search for the best rule in the rule hypothesis space. Within this search paradigm, this paper illustrates the ideas on how to partition the ANN hypothesis space by the architecture of the ANN model, how to prune the hypothesis space by the engineering knowledge of the problem domain, and how to search for the optimum solution with the statistical analysis tools —factorial design and response surface methods. The summation of these techniques forms a methodology to find a best possible ANN model for the problem studied. Finally, an example is provided to demonstrate the modeling process.

2. An Inductive ANN Modeling Process

Learning a function from examples of its inputs and outputs is called inductive learning. More formally, an example is a pair (x, f(x)), where x is the input and f(x) is the output of the function applied to x. The task of pure inductive inference (or induction) is this: given a collection of examples of f, return a function h that approximates f. The function h is called a hypothesis.

The process of completing a picture puzzle is an excellent analogy to the process of learning by the inductive inference. If all the pieces of the puzzle are available, with time and effort the pieces can be placed properly to determine what the picture is. During the process, some rules may be discovered which will help to either guide the direction of or increase the speed of building the puzzle. In terms of the definition of inductive learning, these pieces of the puzzle are the learning data set, the picture is the target function f and the rules or experience of building the puzzle is the learning algorithm.

The puzzle building process is just one simple example illustrating the process of inductive learning. Much research has been done on the more complicated issues of the inductive inference problems with various types of representations for the sample inputs and the inference procedures[1]. Nevertheless, there are several distinct characteristics of inductive learning.

As summarized by Anglium, et al (1983) in their survey of the existing inductive inference processes, an inductive inference problem includes five items: 1) the class of rules being considered, such as a class of functions or languages; 2) the hypothesis space, that is, a set of descriptions such that each rule in the class has at least one description in the hypothesis space; 3) for each rule, a set of examples, and the sequences of examples that constitute admissible presentations of the rules; 4) the class of inference method under consideration; and 5) the criteria for a successful inference. To evaluate the ANN modeling approach from the views of inductive learning, it is necessary to examine the ANN problem solving process with these five characteristics of the inductive learning procedure.

[1] For more information on connectionist network representations, please check S. Kremer's "A theory of grammatical induction in the connectionist paradigm".

For the ANN modeling approach, the first element of an inductive problem -- the class of rules -- is interpreted as the class of nonlinear functions. In general, every ANN model is a black box capable of establishing relationships between inputs and outputs. Although the relationships can be either linear and nonlinear, the major interests in ANN modeling focus on the nonlinear relationship, especially in the fields of engineering. Thus, the class of rules for ANN in this paper is interpreted as a class of nonlinear functions and every ANN model is a rule which describes one specific nonlinear function.

Once the class of rules for ANN is identified, the summation of the descriptions of each rule in the class forms the second element of inductive problem — the hypothesis space. A detailed discussion of hypothesis space of ANN modeling approach is further illustrated in the next section. The third element of an inductive problem is to collect a set of admissible examples. Problems which can be described adequately by a set of attributes should fit this requirement well. Each set of attributes describes a problem scenario and the summation of the example sets is an admissible presentation of the problem. An additional restriction on admissible presentation is to limit the example sets to those that are computable with algorithm(s) of reasonable time complexity and space complexity. Another common restriction is to present the examples in a fixed sequence determined by the domain to reduce the computational complexity (Kremer, 1995).

With the ANN modeling approach, the class of inference method will be the combination of the computing algorithms which adjust the elements of the connectionist network.

Depending on the knowledge of the study domain, the inference method may be different. For example, the inference method in a feedforward ANN, such as the error backpropagation, is different to that of a recurrent network, such as the Hopfield network[2]. Finally, the last requirement of an inductive problem is also satisfied with the ANN model approach. To judge the convergence of an ANN model, empirical criteria are usually developed to determine the fitness of the trained neural network.

3. Why Inductive Learning Works With ANN

In solving a practical problem by inductive learning, there is a great possibility that the complete data describing every aspect of the problem may not be available, due to many constraining factors such as the limit in knowledge, economical considerations or deficiencies in the sampling method and process. Therefore, it is a challenge for the inductive learning to unveil the complete picture with limited information. This situation is exactly like the case when some pieces of the puzzle are missing or some pieces do not fit into the picture. Under these scenarios, the success of the puzzle building depends on whether it is possible to guess an approximately right answer from the incomplete picture.

The situations encountered in the puzzle building process are common to all the inductive learning processes including ANN modeling approach. For successful inductive learning, there are two key points that must be clarified. That is how much source data are needed to learn a correct concept and how does the inductive procedure learn? These two points

[2] Hopfield networks are probably the best-understood class of recurrent networks. They use bidirectional connections with symmetric weights. All units are both input and output units. The activation function is the sign function and the activation levels can only be ± 1.

can be accessed based on the fundamental research of inductive learning by Gold and the recent developments of the computational learning theory.

In his fundamental paper of "Language identification in the limit" (Gold, 1967), Gold established two important concepts for inductive inference: identification in the limit and identification by enumeration. These two concepts provide the principle answers to the two questions just raised. First, the concept of identification in the limit and the computational learning theory will be used to illustrate that the theoretical limits of the size of source data required for learning a correct concept from them does exist. Second, the concept of identification by enumeration will be used to demonstrate how the inductive procedure learns with a searching paradigm.

3.1 Theoretical Limits of the Sample Size

The concept of identification in the limit views inductive inference as an infinite process. However, Gold believed that there must exist some eventual or limiting behaviors of an inductive inference method which can be used to define the success of a learning process after certain examples are learned. An excellent description was provided by Angluin, *et al* (1983) to further clarify this point. Suppose that I_1 is an inductive inference method that is attempting to describe correctly some unknown rule R. If I_1 is run repeatedly on larger and larger collections of examples of R, an infinite sequence of I_1's conjectures is generated as g_1, g_2, g_3, ... During this process, I_1 may be viewed as learning more and more about the unknown rule R and successively modifying its conjectures about R. If there exists some number m such that g_m is a correct description of R and $g_m = g_{m+1} = g_{m+2} = \ldots$, then I_1 is said to identify R correctly in the limit of sample size m on this sequence of examples.

When Gold introduced the concept of identification in the limit, he more or less focused on the application of this concept in the aspect of artificial grammars. With the recent development of computational theory, this concept is further extended for the rest of inductive learning procedures. The key concept in the computational learning theory is the Probably Approximately Correct-learning (PAC-learning) introduced by Valiant (1984). The underlying principle is that any hypothesis which is seriously wrong will almost certainly be "found out" with high probability after a small number of examples, because it will make an incorrect prediction.

There are subtleties in this definition. In the inductive learning process, it is common to divide the available data set into a training set and a testing set. For the learning process to be successful, the hypothesis must not only be correct on the training set, but also approximately correct on the testing set. To satisfy this requirement, a key assumption called the stationary assumption was introduced by Valiant which states that both the training and the testing sets are drawn randomly from the same population of examples using the same probability distribution. Under this stationary assumption, the error of a hypothesis h with respect to the true function f can be defined as the probability that $h(x) \neq f(x)$. This can be rewritten as

$\text{Error}(h) = P(h(x) \neq f(x) \mid x \text{ drawn from a probability distribution})$.

With the definition of error(h), the PAC-learning concept can be formally defined as illustrated in Figure 4-1. Since a hypothesis space H which contains the true function f could be infinite, a complete search to find the true function f becomes impractical. Thus, in order to increase the probability of obtaining an answer from the learning process, an hypothesis h that is approximately correct (if error(h) $\leq \varepsilon$, where ε is a small constant) can be accepted. With Figure 4-1, hypothesis h lies inside what is called the ε-ball around the true function f. The space outside the ε-ball is called H_{Bad}.

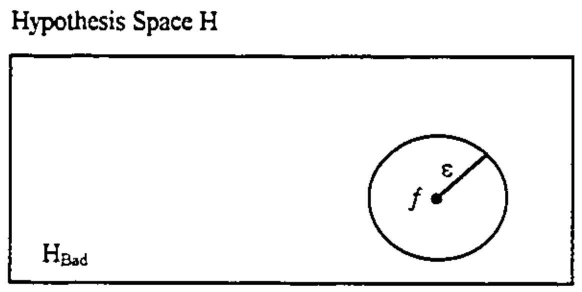


Figure 4-1 The Diagram for PAC-learning Concept

The remaining question is what is the upper limit m of the size of the training data set such that after m examples, with high probability, all consistent hypotheses will be approximately correct. There are two methods of providing the theoretical learning limits for the concept of the PAC-learning.

The first method was introduced by Valiant (1984). In order to examine the learnability of the problems, Valiant introduced a probability function L to establish the combinational bound of the learning theory. The function L has two parameters written as L(h,s), in which h is any real number greater than 1 and s is a positive integer. The purpose of the function is defined as: for any legitimate h and s, function L(h,s) will return the smallest integer m such that in m independent Bernoulli trials, each with probability of at least h^{-1} of success, the probability of having fewer than s successes is less than h^{-1}. Valiant further proved (Valiant, 1984) that:

$$m = L(h,s) \leq 2h(s+\log_e{}^{h}) \qquad (1)$$

By Equation 1 and function L, the upper limit m of the size of the learning samples can be estimated. The challenge is then to find the suitable values for s and h. It is easier to assign the value of ε^{-1} for h than to assign a reasonable approximation for s. One way around this problem is to imagine s as the number of the approximately correct hypotheses inside the ε-ball with the radius of ε while the whole hypothesis space is a ball with the radius of 1. Thus,

$$s = H \times [(4\pi\varepsilon^3/3) / (4\pi 1^3/3)] = H\varepsilon^3 \qquad (2)$$

Therefore the upper bound of the training set size is:

$$m = L(h,s) = L(\varepsilon^{-1}, H\varepsilon^3) \leq 2\varepsilon^{-1}(H\times\varepsilon^3 + \ln \varepsilon^{-1}) = 2H\times\varepsilon^2 + 2\varepsilon^{-1} \ln \varepsilon^{-1} \qquad (3)$$

Since $2\varepsilon^{-1} \ln \varepsilon^{-1}$ is small compared to H, Equation 3 can be rewritten as:

$$m \leq 2H\varepsilon^2 \qquad (4)$$

The second method is directly derived from the definition of PAC-learning (Russell, et al, 1994) to estimate the lower bound of m. The probability of a seriously wrong hypothesis,

h_b, which is consistent with the first m examples can be calculated as follows. Since error(h_b) > ε, the probability that h_b agrees with any given example is less than or equal to 1-ε. The bound for m examples is

$$P(h_b \text{ agrees with m example}) \leq (1-\varepsilon)^m \qquad (5)$$

For the hypothesis space H_{Bad} (Figure 4-2) to contain a consistent hypothesis, at least one of the hypotheses in H_{Bad} must be consistent. The probability of this occurring is bounded by the sum of the individual probabilities:

$$P(H_{Bad} \text{ contains a consistent hypothesis}) \leq H_{Bad}(1-\varepsilon)^m \leq H(1-\varepsilon)^m \qquad (6)$$

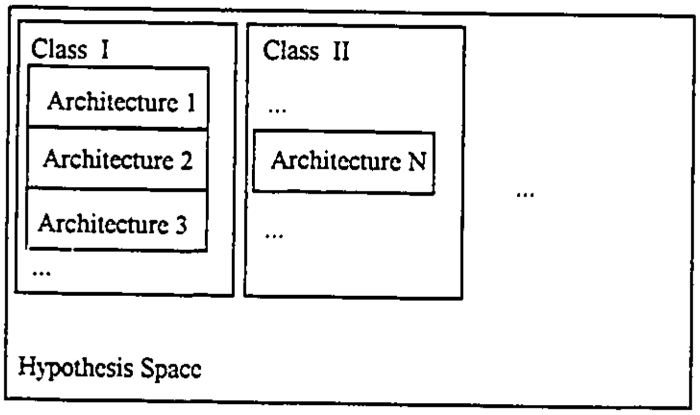


Figure 4-2 The Partitioned Hypothesis Space For ANN Model

In order to obtain a meaningful result, it is preferred to reduce this probability to below some small number δ, thus $P(H_{Bad}$ contains a consistent hypothesis$) \leq H(1-\varepsilon)^m \leq \delta$, which implies

$$(1-\varepsilon)^m \leq \delta/H \qquad (7)$$

Additionally, since function $f(\varepsilon) = \ln(1-\varepsilon) + \varepsilon$ is monotonically decreasing with an upper bound of zero in the interval of $\varepsilon \subseteq [0,1)$[3],

$$\ln(1-\varepsilon) + \varepsilon \leq 0, \Rightarrow \ln(1-\varepsilon) \leq -\varepsilon \tag{8}$$

By knowing that function $f(x) = \ln(x)$ and function $f(x) = x^m$ (m is a positive integer) are both monotonically increasing, from Equation 8, it can be derived that

$$(1-\varepsilon)^m \leq e^{-\varepsilon m} \tag{9}$$

Combining Equation 9 and Equation 7 together, we obtained

$$m \geq \varepsilon^{-1} [\ln(\delta^{-1}) + \ln(H)] \tag{10}$$

From Equation 4 and 10, it is possible to estimate the lower bound and the upper bound of m. The key question then is the size of the hypothesis space H. It was shown by Russell and Norvig (1994) that even with the simplest Boolean function with n attributes, the size of the hypothesis space grows as 2^n and becomes intractable. The dilemma is that if there are not any restrictions on the space of functions that the algorithm can consider, the problem becomes unlearnable; but if there are some restrictions imposed on the hypothesis space, the true function may be eliminated altogether.

The answer to this dilemma is obvious. The restrictions on the hypothesis space are always needed. It is preferred to have an approximately right answer rather than not to try it at all.

[3] The monotonically decreasing character of $f(\varepsilon)$ can be obtained by proving that the first derivative of $f(\varepsilon)$ is always less than zero in the interval of $[0,1)$. ε can not be 1 because $\ln(1-\varepsilon)$ will be meaningless.

The difficulty lies in how to place the reasonable restrictions on the hypothesis space. In Section 4 and 5, a methodology is proposed to construct a restricted hypothesis space.

3.2 ANN Learning As Search In the Hypothesis Space

Once there is a known limit for the learning examples needed, the next step is to find a way to explore the answer. This process is called the search in the hypothesis space and was illustrated by Gold in his second key concept of inductive learning: identification by enumeration.

Identification by enumeration is an abstraction of the process of searching the space of possible rules until one is found that agrees with all the data given. Suppose that a particular domain of rules is specified, and there is an enumeration of descriptions: d_1, d_2, ..., such that each rule in the domain has at least one descriptions in this enumeration. The method of identification by enumeration goes down the list to find the first description that is compatible with the given examples. For the enumeration method to find the limit and to be computable, the following conditions must be satisfied: 1) a correct hypothesis is always compatible with the examples given; 2) any incorrect hypothesis is incompatible with some sufficiently large collection of examples and with all larger collections; 3) all the enumeration d_n must be computable.

A common search paradigm consists of four steps. First, define a hypothesis space containing all possible configurations of the relevant objects. Second, identify a decision procedure for determining whether an arbitrary point in the hypothesis space constitutes the goal of the task. Third, define a procedure for traversing the hypothesis space until a satisfactory goal state is discovered. And, fourth, determine a state representation for describing a point in the hypothesis space passed by the search procedure. In short, the search paradigm consists of a hypothesis space, a goal tester, a state generator, a state representation and a initial state to start the search.

For the ANN modeling approach, the hypothesis space is defined by the different classes of network architectures. The goal tester is the criteria to measure the ability of the ANN model to converge and generalize. The representation of a state in the hypothesis is a combination of the values of the connection weights; and the state generator is the learning algorithm used such as the error backpropagation method to move from state to state. For some learning algorithms, the initial state could be different from one another. More often, the initial state of an ANN model is a combination of randomly assigned small connection weights.

The method of identification by enumeration is very general and powerful but also rather impractical because the search through the hypothesis space is typically exponential in time and space (Angluin, *et al*, 1983). As mentioned previously, it appears that the key question in the search for the optimal solution is how to restrict the size of the hypothesis space. There are two possible ways. One is to place some reasonable restrictions on the hypothesis space by partitioning the large space into smaller ones. The other is to use prior knowledge of the study domain and some statistical tools to prune the partitioned hypothesis space. Before introducing these two techniques, it is necessary to examine the hypothesis space of the ANN modeling approach.

4. Hypothesis Space of ANN Modeling Approach

The rule hypothesis space is a key concept in examining the ANN modeling approach from the aspect of inductive learning. Here, the rule hypothesis space is a summation of all the possible ANN models. Each model is a rule that describes a concept inducted from the learning examples. Since each ANN model is a black box as it appears, it is a challenge to describe and organize the hypothesis space precisely with this presentation. Therefore, one must look inside the black box and examine the architecture of ANN model.

From a conceptual view, a connectionist network architecture consists of input vector, output vector, connection weights and their associated computing algorithms. From a practical view of problem solving with ANN, many elements of the connectionist network architecture, for example the computing algorithms, can be determined by the prior knowledge of the problem to be solved[4] (Kremer, 1995). Thus, when designing the architecture of an ANN model, more attention is directed to the selection of input vectors, output vectors and connection weights. Among these three elements, within a closed study domain, the possible choices of the input and output vectors are limited because sufficient

[4] Kremer developed a table for determining the suitable computing algorithm for each type of problem.

knowledge can be obtained in a closed domain to clarify most of the selection. Under this scenario, the selection of input and output vectors is fairly established. On the other hand, the design of hidden layer(s) is highly variable. It is the most dynamic factor that directly influences the quality of the ANN architecture design.

Usually the design of the hidden layers involves three factors: the number of the hidden layers, the number of the neurons in each hidden layer, and the ratio of the neurons in different hidden layers if there are multiple hidden layers. As to the number of the hidden layers, although the frequent choices are either one layer or two layers, the type of problems they can handle are quite different and usually one out-performs the other. Thus, the number of hidden layers is a key factor. Although the architecture of the hidden layer can be specified by the number of the neurons in the hidden layer, it is the total number of the connection weights in the hidden layer that is important as they are the true components of the model for describing the concept the ANN model learned. In the case of noisy data, the convergence ability of the ANN model becomes an important issue. To force convergence in a two-hidden-layer architecture, the neurons in the first hidden layer[5] are designed to be greater in number than that in the second hidden layer[6]. The ratio of the number of the neurons in these two hidden layers then becomes another key factor.

With these three key description factors plus the input, output vectors and their computing algorithms, different types of ANN architecture can be determined. In other words, the

[5] The hidden layer directly connects to the input layer.
[6] The hidden layer directly connects to the output layer.

difference of the rule classes can be distinguished. The hypothesis space of ANN modeling can then be redefined as the summation of different classes of ANN architectures.

Within a class of architectures, further specifications of design factors such as the determination of the number of neurons in the hidden layer will establish a detail architecture or a detail rule in a rule class. Each architecture further forms a subset of the hypothesis space called the architecture space, which consists of all the ANN models with the same architecture design but different combination in the neuron connection weights. Since the connection weights are real numbers, the number of the models within each architecture space is infinite.

Nevertheless, there is an implied benefit with this description of hypothesis space. Despite the overall hypothesis space being infinite, this approach allows the hypothesis space to be partitioned exclusively into many small pieces as shown in Figure 4-2, in which each rectangle represents an exclusive rule space associated with each architecture. This partitioned hypothesis space makes the systematic study on a certain part of the hypothesis space possible without losing the integrity of the whole study.

5. Partitioning the ANN Hypothesis Space

In this section, this concept of partitioning the ANN hypothesis space is illustrated by introducing two methods which divide the hypothesis space of the ANN modeling process. Remember that the architecture of hidden layers is determined by three key factors. Among them, the number of hidden layers is the factor with the most limited selection of choices: one layer, two layers, and three or more layers. Therefore, the first approach is to use the number of the hidden layers to categorize ANN architectures.

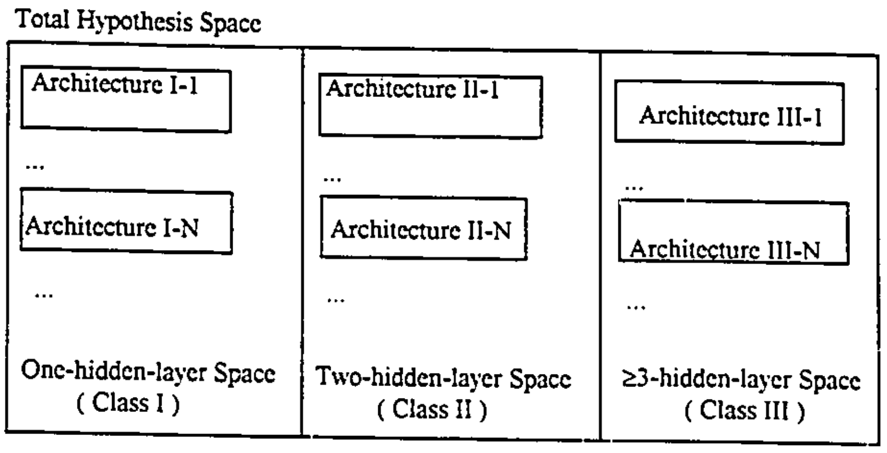


Figure 4-3 The New Partitioned Hypothesis Space

As shown in Figure 4-3, the hypothesis space is divided into three classes. The ANN architectures with one or two hidden layer(s) are most commonly encountered in engineering applications. Thus, each forms a category itself. Although the ANN architectures with three or more hidden layers are seldom encountered, for the sake of completeness of the partitioning process, the third hypothesis category is added to contain the architectures with three or more hidden layers.

The second approach is to use the learning algorithm to shrink the infinite weight space of each architecture into a finite search space. As shown in Figure 4-4, the shaded ellipse within each architecture model represents the new finite weight space. Given a predetermined architecture, the number of the possible combinations of the weights is infinite. In order to regroup this infinite space into a finite search space, a new concept of associative weight space is introduced.

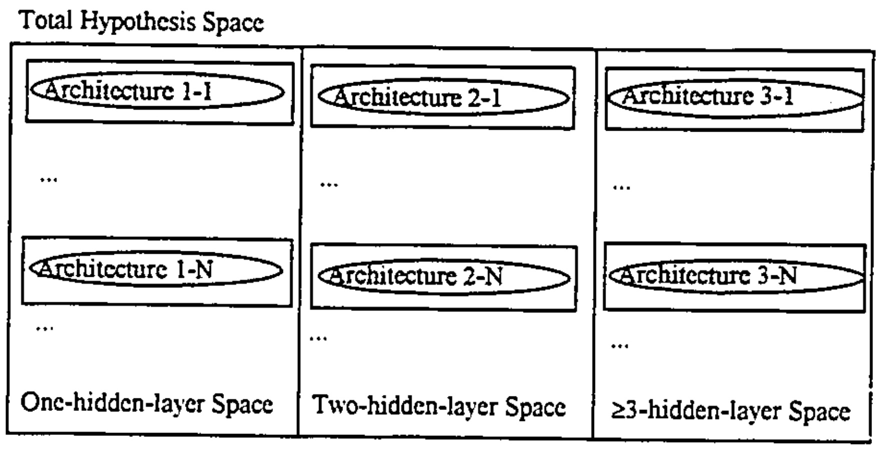


Figure 4-4 The Final Partitioned Hypothesis Space

For each modeling case, it is assumed that a PAC-answer exists in the weight space of a particular architecture. Furthermore, this combination of the weights can not be obtained randomly. It must be found by a learning algorithm which traverses the hypothesis space and experiences many different combinations of the weights before finally reaching the answer. Thus, those combinations traversed are related to the final combination by the learning algorithm. They are the predecessors of the final PAC-answer. All these connection weight combinations form a tractable weight space called the associative weight space of an architecture.

Another way to interpret the associative weight space of an architecture is to view it as a finite collection of the pathways traversed by learning algorithms. Two key points in this interpretation are: first, there are finite elements in each pathway and second, there is a finite number of pathways. Each pathway starts from some initial point in the hypothesis space and moves toward the target ε ball. Since the initial weights are usually assigned with small values in practice, they are in a fairly restricted starting area of the hypothesis space, for example the origin of a cardinal space, which can simply be considered as one point if compared to the infinite hypothesis space. These pathways will either hit the ε ball or end up in or around some local extremes. Since each pathway has both start and end points with the learning algorithm to calculate each step in between, the total number of the weight combinations in each pathway is finite.

The total number of pathways with an architecture is determined by the learning algorithm and the presentation order of the learning examples. The selection of the learning algorithm is limited due to the availability and the suitability of the learning algorithms for the study domain. Although the total number of the ordering sequence of n learning examples is n!, which means there are n! potential pathways, it is still a finite number. Therefore, the original infinite hypothesis space is now reduced to finite subspaces.

Even after adding all these restrictions on the hypothesis space, the solution subspace is still exponentially large because of n!. The next question is how to find the solution with a reasonable time complexity and space complexity. The partial solution to this dilemma is the application of the prior knowledge and the usage of statistical analysis tools.

6. Pruning the ANN Hypothesis Space

With the hypothesis space shown in Figure 4-4, there are three possible ways to reduce the size of hypothesis space. First, prune the class of architecture as a whole. Selecting one particular hypothesis class will effectively prune two third of the hypothesis space. Second, prune the type of architecture space within a class. Even a partial determination of the elements of ANN model architecture will reduce the selection of architecture type significantly. Third, prune the associative weight space within an ANN architecture space by reducing the selection of the pathways and the length of the pathway. Within each approach, the requirement of prior knowledge of the study domain is extensive.

The process to prune the hypothesis class depends on how the class of architecture is designed. Any prior knowledge related to the design will help to select the appropriate class of architecture. For the hypothesis space shown in Figure 4-4, the useful knowledge is the sample data analysis and the information gathered from the preliminary runs.

Typical sample data analysis includes pattern analysis and data noise analysis. Pattern analysis of the input and output vectors focuses on whether there is any trend of data development, any correlationship(s) and autocorrelationship(s) between input and output vectors and among themselves. The noisiness of the data can be examined in the data scatter plot or series plot against time or any other independent parameters. From experience, the appearance of frequent irregular data extremes, nonlinear variance, strong autocorrelation and high degree of randomness in the source data indicates the complexity of the study domain and usually the two-hidden-layer design is required for a better generalization of the model. If the pattern of the source data is easy to recognize and the input vectors are fairly independent, the better choice for architectural design will be one hidden layer. Although these rules of thumb help the selection of the right class, it is recommended that some preliminary runs should be performed to compare the time complexity and preliminary accuracy of the model.

Due to the complexity in the discussion of pruning architecture space, the process of pruning the associative weight space within a selected architecture space is introduced first. The possible techniques are weight fixing and proper selection of the sequence of learning examples (Kremer ,1995). As pointed out by Kremer, prior knowledge can be used to choose or hard-wire (fixing weights and not allowing them to be trained) the initial weights prior to training. This is equivalent to a short cut in the pathway. It reduces the time and space complexity of search by rejecting some part of the associative weight pathway. However, there are some limitations. First, there is some cost involved for knowing the short cut and it may not always be affordable. Second, fixing some weights restricts the degrees of freedom available to a learning algorithm. It is also possible for the trained weights in the network to overpower the contributions of the fixed weights. Third, the local extreme may render the method of choosing initial weights invalid because the global extreme will not be subsequently explored. Thus, it is a delicate art in selecting proper weight fixing.

Input ordering is another technique to reduce the associative weight space. As pointed out by Kremer (1995), for an input ordering to be advantageous, two criteria must be met: (1) the input sequencing should provide additional information of the study domain other than showing the learning examples and (2) the learning system should be informed and use it to reduce the search space. One such example is to present the simple learning examples first then the complicate ones. This approach has been used mostly in the artificial grammar learning as mentioned by Kremer. Its usefulness in the study domain where no typical sequence exists in learning examples may need to be further explored.

The process of pruning architectural space within a class is the most complicated issue in the subject of pruning hypothesis space since there are many architectural elements to be considered at a time. Such elements include the number of input and output vectors, the number of neurons in the hidden layer, etc. Typically a rough number of the input and output vector can be determined by the prior knowledge of the study domain. However, in many cases, the prior knowledge is insufficient for an optimal answer. For example, some input parameters may be correlated and form a dependent relationship with a degree of significance. Thus an optimal number of these parameters must be selected into the model for a built-in redundancy. Also if the mechanism of a problem under study is not completely understood, it can be difficult to determine whether the higher derivatives of source parameters should be included in the input parameters. The existence of multiple interactive parameters complicates the process of pruning the architectural space. Another typical problem is what is the optimal number of neurons in the hidden layer(s). A partial answer to these questions can be obtained by the use of statistical tools of factorial design and the response surface methods.

To perform a general factorial design, an investigator selects a fixed number of "levels" for each of a number of factors and then runs experiments with all the possible combinations. If there are l_1 levels for the first factor, l_2 for the second, ..., and l_k for the kth, the complete arrangement of $l_1 \times l_2 \times \ldots \times l_k$ experimental runs is called an $l_1 \times l_2 \times \ldots \times l_k$ factorial design (Box, et al, 1978). For example, a $2 \times 2 \times 2 \times 2$ factorial design includes four variables each with two levels and requires 16 experimental runs.

The factorial design is of importance for a number of reasons (Box, et al, 1978): 1) although the initial experiment from factorial design may not be able to explore a wide region in the factor space, they can indicate major trends and so determine a promising direction for further examination; 2) if a more thorough local exploration is needed, they can be suitably augmented to form composite designs; and 3) the factorial design looks over a large number of factors superficially rather than a small number thoroughly at the early stage of an investigation. For these reasons, this systematic approach is of great value in pruning the architectural hypothesis space.

The power of this method can be illustrated through an example. In this example, the ANN modeling approach was used to build a model for the coagulation and flocculation process in a water treatment plant. From the data analysis and the preliminary runs, several guiding points were found. First, since the data pattern is fairly noisy, the architecture with one hidden layer can not perform the task no matter how large the number of the neurons in the hidden layer is. In addition, the time complexity with the three-hidden-layer design is simply intolerable. Thus, the proper choice is two-hidden-layer architecture. Second, in the preliminary runs, the models tend to converge better with increasing number of connection neurons in the network. Third, the converging time increased significantly as the number of the weights in the network increased. Therefore, an optimal answer must be a balanced choice between the second and the third point. A set of evaluation criteria[7] was then developed to measure the overall goodness of each run in the factorial design with respect to convergence and generalization. These numercial measurements are called run effects. By these run effects, the model runs become comparable.

The preliminary analysis identified the class of architecture and some key influencing parameters related to the architectural design of ANN models. The next step is to search for the optimal architecture by factorial design. In the following example, a factorial design process is illustrated to determine the significance of four factors in the optimal design of the ANN model. Two of the four factors are the potential input parameters: the change rate of the overflow in the sedimentation tank and the system index[8]. The available

[7] The more information about the evaluaiton criteria is presented in Appendix II.
[8] The system index is a parameter designed to help recognize the data pattern.

engineering knowledge can not determine whether the system index should be included in the input and there is a slight preference to exclude it from the input because a system index design may bring in additional noise to the input pattern. On the other hand, the engineering knowledge concludes that the change rate of the overflow should be excluded from the input. The reason to put this factor in the factorial design is to demonstrate the capacity of the factorial design to conclude that the effect of this parameter is insignificant to the optimal ANN model. The other two factors are the total number of the weights in the model and the ratio of the number of the neurons in two hidden layers.

Considering the time complexity, it was decided to run a half-fraction factorial design first. It was believed these four factors were independent to each other because there is not any cause effect relationship among them. The only caution is that there may be an interaction between the ratio of hidden neurons and the total number of the weights. A resolution IV design will be able to justify this doubt. With this partial factorial design, if it does show some interactions interfering with the main effect, the other half of the factorial design can be completed to determine all main effects and interactions. The detailed factorial design is shown in Table 4-1. The generator used is I=1234.

Table 4-1 The Factorial Design

Factors	+ Level	- Level
Change Rate of Overflow	Not an input parameter	Is an input parameter
Ratio of Neurons	2:1	1:1
Total Weights in the Network	2500	1600
The System Index	Is an input parameter	Not an input parameter

Table 4-2 shows the results of the factorial design. By comparing the main effect of each factor to the mean run effect and to each other, it can be concluded that the total weight number is the most significant factor influencing the selection of the architecture. On the other hand, the system index and the change of the overflow rate are much less significant. This confirmed our initial estimations. The significance of the main effect of the neuron ratio is not as clear as that of the total weight number and should be explored further. In addition, it is interesting to find that when both the neuron ratio and the total weight number are at the positive level, the effects of the runs are much better than those of the other combinations. This suggests there must be a positive interaction between these two factors which improves the run effect. Further exploration of the optimal architecture should focus on the proper selection of the total number of weights and the neuron ratio. The statistical approach used for accomplishing this task is the response surface method.

Table 4-2 The Runs And The Results

Run ID	ΔOverflow	Neuron Ratio	Total Weights	System Index	Run Effects
4	+	+	+	+	0.59
2	-	+	+	-	0.48
7	+	-	+	-	0.25
3	-	-	+	+	0.25
1	+	+	-	-	0.135
6	-	+	-	+	0
5	+	-	-	+	0.15
8	-	-	-	-	0.05
Main Effect	0.086	0.126	0.308	0.025	0.238

Response surface methodology consists of a group of techniques used in the empirical study of relationships between a measured response (or output) and a number of input variables. In this case, the responses are the run effects and the input variables are the total number of weights and the neuron ratio. The response surface method is an effective approach to measure the joint effects of the input variables on the response if the input variables do interact with each other.

As pointed out in the analysis on the first half-fraction, the next step in searching for the better architecture should be at the region of higher neuron ratio and larger weight connections. Thus, the design was started with a 2^2 factorial design with a center point. The design and the run effects are listed in Table 4-3 and Table 4-4.

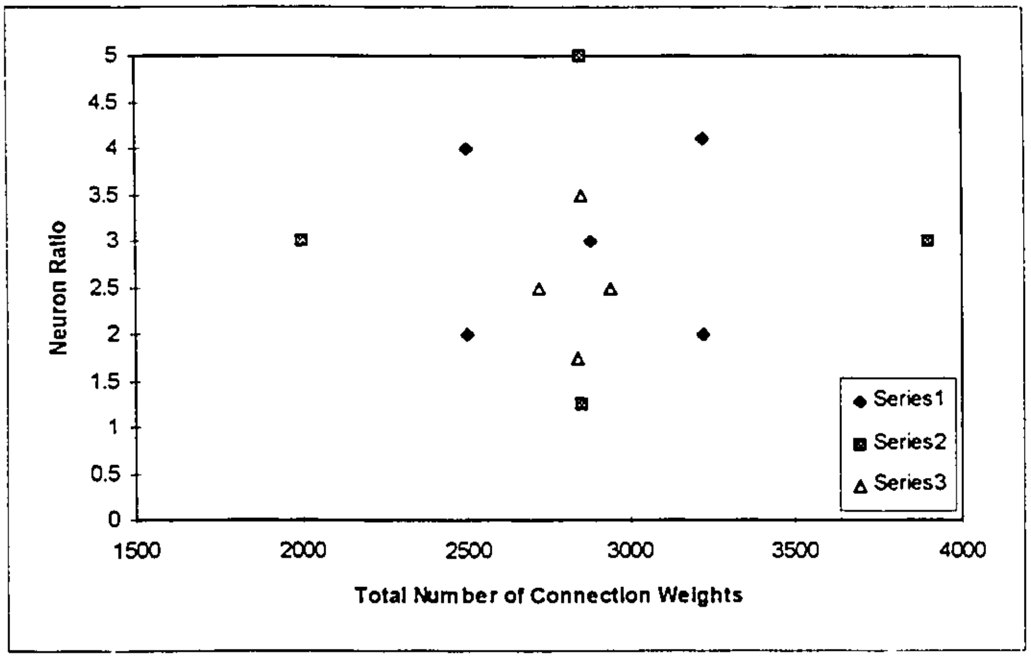
Table 4-3 The New 2^2 Factorial Design

Factors	+ Level	0 Level	- Level
Ratio of Neurons	4:1	3:1	2:1
Total Weights in the Net	3200	2880	2500

Table 4-4 The New 2^2 Runs and Results

Run ID	Neuron Ratio	The No. of Total Weight	Run Effects
4	+1	+1	0.15
2	-1	+1	0.325
3	+1	-1	0.225
1	-1	-1	0.15
5	0	0	0.455

Apparently the response function will not be first-order because the highest effect is in the center. Since the peak is at the center of the run, it is easy to have a perception that the optimum value should be around the center point and does not consider the possibility that there may be a ridge cross the middle of the design region. For this consideration, a second group of 4 additional runs in a "star" format which is shown in Figure 4-5 was attempted. The design and the run results are shown in Table 4-5. The center point was not rerun as the model will return the same run result if the input, output and the architecture of the neural network are the same.



Note: Series 1 is the first group of 5 runs;
Series 2 is the second group of 4 runs;
Series 3 is the third group of 4 runs.

Figure 4-5 The Star Design Testing Pattern

Table 4-5 The "Star" Runs and Results

Run ID	Neuron Ratio	The No. of Total Weights	Run Effects
3	5:1	2850	0.15
2	3:1	2000	0.05
4	3:1	3900	0.15
1	1.25:1	2850	0.25

Table 4-6 The Second "Star" Runs and Results

Run ID	Neuron Ratio	The No. of Total Weights	Run Effects
1	2.5:1	2700	0.15
2	1.75:1	2840	0.525
3	2.5:1	2920	0.405
4	3.5:1	2850	0.225

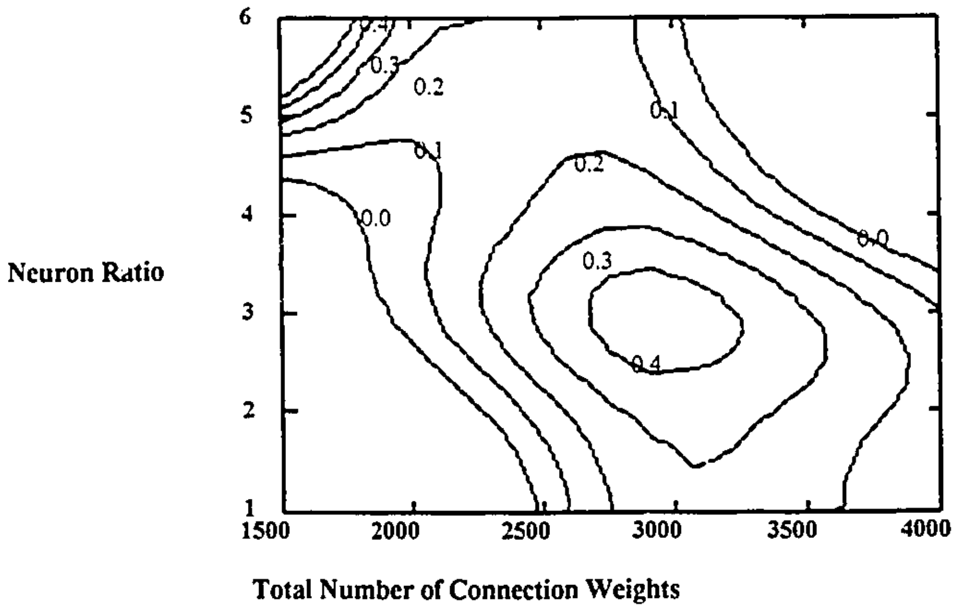


Figure 4-6 The Partial Response Surface Measuring the Success of ANN Models Repecting to the Variance In Neuron Ratio And Total Number of Connection Weights

In order to obtain the full picture of the run effects associated with these nine runs, a run effect contour was plotted as shown in Figure 4-6. It is interesting to see that there are two potential areas where the optimum solution could exist: the center of the graph or the upper left corner of the graph. One additional test was done inside the upper left corner region and the result proved that the contour lines in the upper left corner result from an imperfect interpretation of the source data. Now, it is clear that the optimum architecture must exist in the center of the design region and another "star" format was run. This time the "star" format was placed tight around the center (Table 4-6) and the optimum result was subsequently found in the region where the neuron ratio is about 1.9 and the total number of connection weights is about 3000 as shown in Figure 4-7. With this two key parameters, the architecture of the optimum model was finally determined.

It is important to examine the contour map cautiously when interpreting the potential optimum solution region. First, the sampling points should cover as much as possible the study domain to obtain the complete picture. Otherwise, it is possible to misinterpret a section of a ridge as a peak. Second, the sampling density around the potential optimum regions (or the suspicious regions) should be dense enough to allow several closed contour loops. Third, the contour interval should be chosen carefully. The best approach is to select the contour interval by the sensitivity of the measured response of the test. Finally, select your sampling region carefully to avoid mistakening the local extreme as the global extreme.

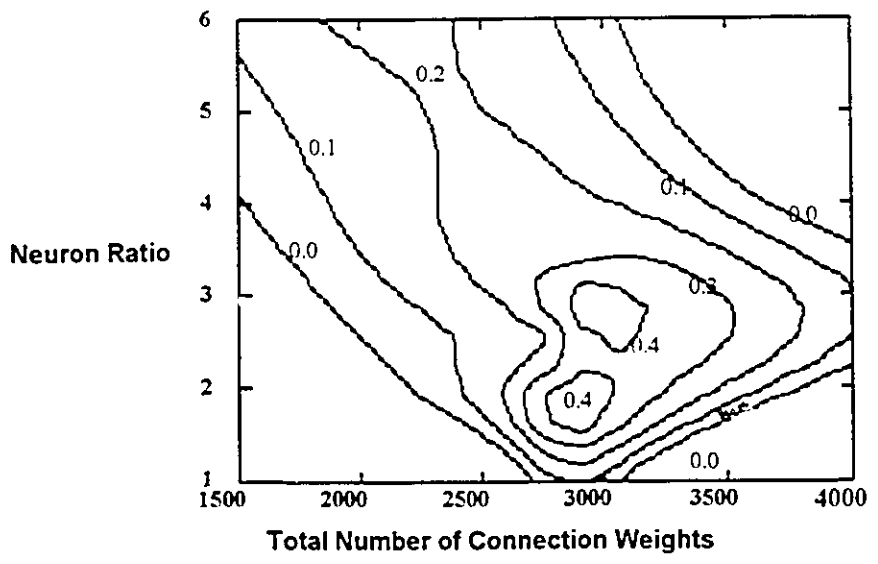


Figure 4-7 The Complete Response Surface Measuring the Success of ANN Models Respecting to the Variance In Neural Ratio And Total Number of Connection Weights

7. Discussion

Like many other modeling approaches, the ANN modeling approach has been demonstrated by scientists and engineers to be applicable to many research fields while for other problems this technique has been found to be inadequate. Much of the controversy with this technique is that there is not a solid theoretical support for the ANN models. In the attempt to explain why this modeling technique works, there is a delicate balance to achieve.

On one hand, there is an urgent need for a theoretical guidance in order to reduce the blindness and randomness in the usage of this technique. For this purpose, in an attempt to establish such a theoretical foundation, this paper borrows concepts of the inductive learning in the symbolist approach and combines them with the connectionist network representation to establish a big picture of this technique. Several fundamental concepts such as defining hypothesis space, learning by enumeration and inductive learning as searching in hypothesis space are thereby introduced.

On the other hand, it should be reminded that an exact theoretical explanation is still not possible because we lack a complete understanding of how human brain functions, from which this modeling technique is originated. To accomodate such a complex situation and be able to materialize the abstract concepts just mentioned into a practical modeling guide, some reasonable assumptions and limitations must be made. Just as humans learn concepts from limited examples and are expected to make satisfactory judgments, so does this technique. There is no need for a machine to learn infinite examples to achieve perfect learning. There must exist a reasonable limit on learning such that its judgement is satifactory. Therefore, in theory, the concept of PAC-answer is derived to establish the learning limit for the machine by which users can establish their own goal in machine learning if needed.

Since the concept of PAC-answer is loosely defined, there is inherent difficulty involved in determining the learning limit by ε and δ. However, there is a practical approach which is used effectively to detemine the size of m. The main idea of the approach is to restrict the study domain such that every major mapping feature between the inputs and outputs is [illegible] and included in the training data. This kind of study domain is called a closed domain. By this approach, the basic learning limit of the problem is the total number of the mapping feature between inputs and outputs of the system. The training result is a conceptual model capable to draw general conclusions and provide general direction for the further study. Moreover, if further detailed information other than the major feature is available for learning, a better model in prediction will result. This may explain why ANN modeling achieves its goals successfully in most instances with limited training examples. Obviously, this approach will not work well for an open domain and subsequent unsatisfactory results can be expected.

There is also an interesting argument which can be made on the limit m. Does every example in the sample group have to be distinct? If not, what is the effect of repeated examples on the limit m? Consider the case of neural networks with backpropagation computing algorithm. With the backpropagation algorithm, the whole training data set is presented as a batch again and again to prune the connection weight space. Within each batch, the training data are distinct. Among the batches, they are identical. Notice that even the training data in each batch are identical both in values and presentation sequence, they are used to prune different combinations of weights in the different stage of the weight space. That is say, the repeated batch presentation of a same data set is as effective as a complete distinct data set in pruning the continuous weight space. In addition, this repeated batch presentation will have the benefit of reinforcing the learning process in some particular direction and thus converging faster than that of the strictly distinct training examples. Nevertheless, this approach has the weakness of possibly converging to the local extremes. Notice that this weakness is not unexcelled for a closed study domain in which inputs and outputs are well understood.

8. Conclusion

With the increased use of artificial neural networks in engineering applications, many questions are being raised on how this technique will work for a given application and how one would make it work better? In this paper, a heuristic approach to tailor the specific components of ANN modeling technique for a closed study domain was illustrated.

This systematic approach is based on a foundamental concept in inductive learning called the PAC-answer in a hypothesis space. Under the guidance of this concept, this approach starts with redefining the hypothesis space of the ANN modeling approach by the architectural representation of the connectionist network. The hypothesis space is partitioned exclusively into a hierarchical structure of subspaces such as class space, architectural space and the associative weight space by the characteristics of the ANN architecture. This hierarchical structure and the concept of the associative weight space are designed to make each subspace tractable by placing a restriction on them. This restriction was the learning algorithm used in traversing the weight space for the PAC-answer.

Nevertheless, the number of these subspaces are still large. Thus, various kinds of the prior knowledge are introduced to prune these subspaces. The focus was on two major kinds of prior knowledge: the background information about the domain under study and a statistical search algorithm which combines the power of factorial design and the response surface methodology.

The background information provides an early logical screening of the input parameters and thus reduces the size of the hypothesis space to be searched later. The decisions at this stage are usually supported by hard evidence to minimize the risk of losing the true target model. Then, the uncertainties in buidling the model are examined by the experimental runs and the statistical analysis tools. Factorial design allows the preliminary screening of many types of ANN architecture with limited test runs. The systematic analysis of the run effects will be able to direct the further testings to the region where the optimal result is expected. This allows the subsequent detailed analysis to be concentrated on the selected influential factors. It is possible that there are interactions among the influential factors. The response surface method examines these interactions in terms of the potential location of the local extremes and the global extreme through the data obtained from the detailed factorial design and finally pinpoints the optimal result. By this way, the systematic searching through the hypothesis space becomes available within reasonable time complexity. This subspace pruning process is demonstrated through a case study and the results are very promising.

However, the techniques used to prune the hypothesis space still heavily rely on the background information of the study domain. For example, prior knowledge is needed in order to determine which factors should be included in factorial design and what levels of the factors should be analyzed. There is also a difficulty in providing an objective and complete evaluation criteria to measure each run result in factorial design since these criteria are determined both by the subjective human experience and the degree of noise in the input data. That is why for the successful usage of this technique, the recommended study domain is limited to the closed ones, in which more information can be obtained about the input vector and the output vector even when the cause-effect relationships between input and output vectors are very fussy and complicated.

Nevertheless, this paper provides a new perspective into the theoretical analysis of the ANN application. Further research in the area of how to prune the hypothesis space will definitely reduce the blindness in the ANN modeling process and guide the practical application of the ANN.

References

Angluin, D. and Smith, C.H. (1983), "Inductive Inference: Theory and Methods", *Computing Survey*, Vol. 15, No. 3, pp. 237-269.

Box, G.E. and Jenkins, G.M. (1976), Time Series Analysis: Forecasting and Control, revised edition, Holder-Day Inc., San Francisco, pp. 575.

Gold, E.M. (1967), "Language Identification in the Limit", *Information and Control*, Vol. 10, pp. 447-474.

Hinton, G.E. (1990), "Connectionist Learning Procedures", Artificial Neural Networks Concept Learning, Edited by Joachim Diederich, IEEE Computer Society Press, pp. 11-47.

Kremer, S.C. (1995), A Theory of Grammatical Induction in the Connectionist Paradigm, Ph.D. thesis, Department of Computer Science, University of Alberta, pp. 250.

Russell, S.J. and Norvig, P. (1994), Artificial Intelligence: A Modern Approach, Prentice-Hall Inc., Englewood Cliffs, New Jersey, pp. 932.

Valiant, L.G. (1984), "A Theory of the Learnable", Communications of ACM, Vol. 27, pp. 1134-1142.

Chapter V

Real-time Water Treatment Process Control With Artificial Neural Networks

1. Introduction

Since water treatment consists of a sequence of complex physical and chemical processes, currently in many water treatment plants (WTP), process control is generally accomplished through examining the quality of the product water and adjusting the processes through operator's experience. This practice is inefficient and slow in control response. It is also the source for many control problems such as selecting optimum dosing of chemicals. With more stringent requirements being placed on water treatment performance, operators need new tools to meet these requirements more efficiently. In this paper, one such tool is presented, which is a reliable artificial neural network (ANN) process models that can be incorporated into process control systems.

Much of the difficulty in modeling water treatment processes can be related to complex interactions among many influential water quality factors and many micro-scale chemical and physical reactions. Although the chemical or physical reactions involved are generally well known in principle, the ability to describe these reactions numerically is limited.

Historically, engineers and scientists have tried to fit mathematical formulas to describe the processes, but this approach has generally resulted in poor performance and typically, limited success in control of the processes.

With advances in computing science and modeling techniques, there are now new approaches to this problem. In this paper, the principal concepts of using the artificial neural network in the water treatment modeling and process control are introduced. This approach offers a number of distinct advantages over the traditional methods. To model water treatment processes, one of the major challenges is the evaluation of the non-linear relationships between inputs and outputs of each process. To solve this problem, the neural network approach focuses on finding a repeated, recognizable and predictable pattern(s) between the causes and the effects from the past operation data records, and bypasses the modeling of actual micro-scale reactions. The model itself does not require a description of how the processes occurs in either the micro or macro environments, only knowledge of important factors which govern the process. This situation makes the ANN modeling approach a rational choice for process modeling and control in water treatment.

With the development of a reliable process model, it can be integrated into the process control architecture. Using neural networks for control requires neural networks to go beyond classifying the control input signals to actually monitoring and influencing them. In this paper, the concept of a neural network as a process model in the internal model control strategy (IMC) will also be examined.

Finally, based on these concepts, a project was initiated to study the potential capacity of ANN process control in water treatment. The project was conducted at the Rossdale Water Treatment Plant in Edmonton, Alberta, Canada. The objective of this project was to select a section of the treatment process, collect real operational data, and build and test the ANN control model. The treatment process chosen was the coagulation, flocculation and clarification process.

2. Background

The Rossdale WTP is a conventional treatment facility consisting of alum coagulation, flocculation, lime softening, disinfection and filtration (Figure 5-1). Its daily production capacity is approximately 360 ML. Although it consistently produces good quality drinking water, plant process control could be improved. Thus the project was started to investigate ANN process control for the coagulation, flocculation and sedimentation processes at the Rossdale WTP. Since all of these processes are completed in a cross-flow sedimentation tank, all will be studied as a control unit.

In developing a process control system, prior knowledge of the control unit is very important. First, prior knowledge can be used to identify key input and output parameters. Second, a clear understanding of the process mechanism will help to qualify the direct relationships among the inputs and outputs and the indirect relationship such as the time series correlation. With the knowledge of these two aspects, a suitable control model can

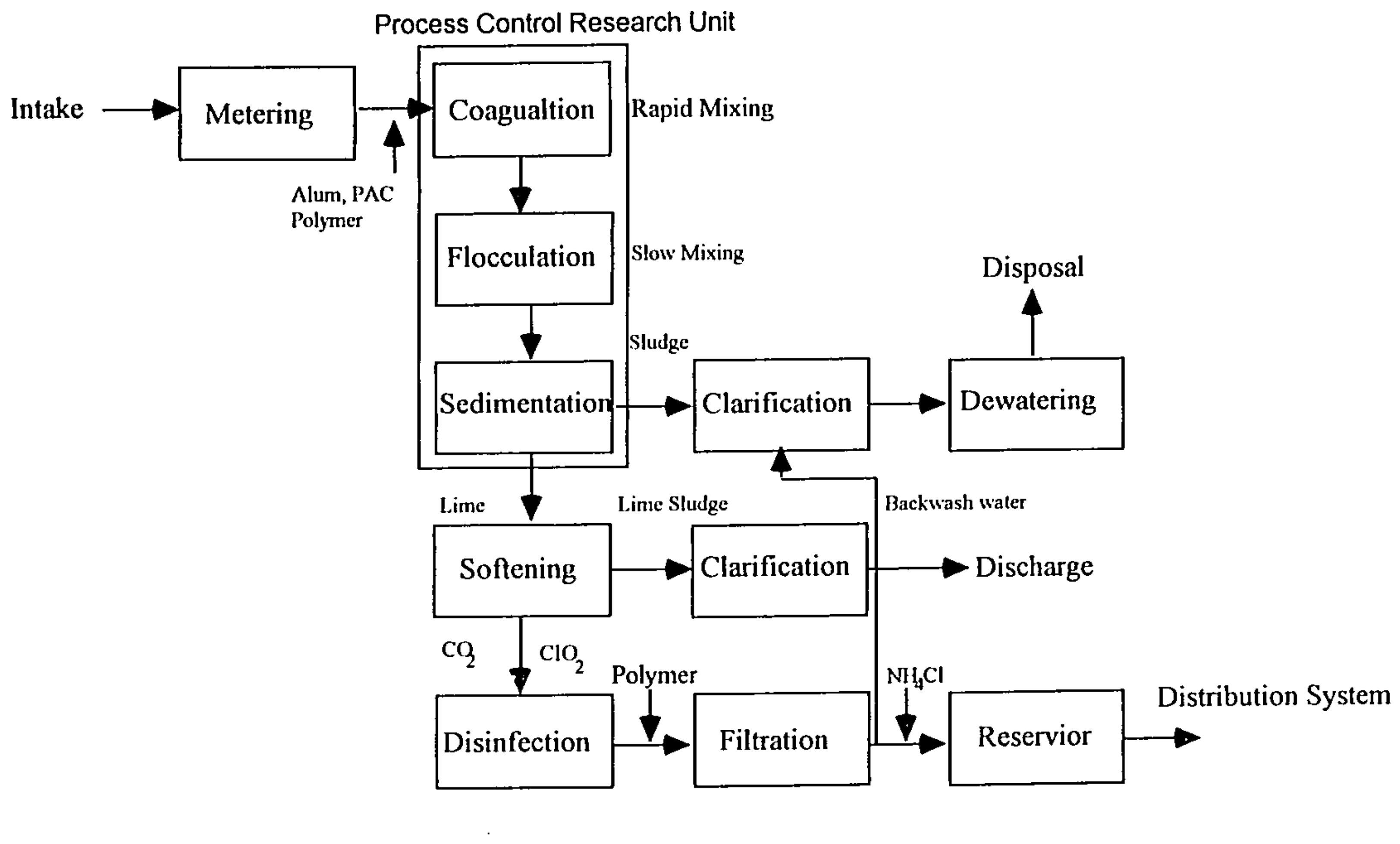


Figure 1 Rossdale Water Treatment Plant Layout

then be selected. A brief review of the process mechanisms for the case study are presented.

The coagulation process in water treatment refers to the addition of a chemical to the water to destabilize colloidal particles, allowing them to form larger agglomeraties in the following flocculation step. In the crossflow clarifier, flocculation occurs at the upstream end of the clarifier and the formed flocs settle to the bottom of the clarifier while the settled water is collected in the effluent troughs at the down stream end of the clarifier.

There are several factors which have various impacts on these processes. Identification of these factors are necessary if ANN is successfully going to be applied. At the Rossdale WTP, the following factors were considered to be important and historical data was available for them. pH is the significant factor that determines the charge of the dominant coagulant species and the concentration of hydroxide precipitates. Alkalinity of the raw water acts as a buffer to the pH system and therefore are closely related. Low turbidity may result in too few particles for good flocculation. However, since the main mechanism of coagulation is sweep flocculation in the Rossdale WTP, the impact of low turbidity should be limited. The raw water colour is a measure of organic content of the water and the powdered activated carbon (PAC) dosage influences the removal of organic taste and odour compounds. PAC is used whenever the raw water colour is high. Low temperature hinders the processes by slowing down the reaction rates and increasing the viscosity. Finally, the overflow rate through the clarifier can impact performance. High overflow rates not only reduces the effectiveness of sedimentation, but also increases the operation burden on the subsequent treatment units. However, a low overflow rate may not be economical.

The eight parameters can be further classified into two categories: those that can be used for monitoring and those that can be used for operational control. The first category includes mainly the water quality parameters such as pH, turbidity, colour, alkalinity[1] and temperature. Typically operators at the facility have little control over these parameters. Of these parameters, temperature is the only one which is adjusted. Throughout winter, raw water at Rossdale is constantly heated to typically 5°C from 0°C. The second category, used for process control, includes chemical doses of alum and PAC and the parameters of the hydraulic control such as the overflow rate. These parameters are adjusted by operators to meet performance criteria.

Once the inputs are identified, the next step is to find a set of additional parameters that measure process performance. Two parameters, the colour and the turbidity of the clarifier effluent, were original selected as the control outputs. A partial time series of the daily effluent colour and turbidity are presented in Figure 5-2. Further study found that the time series of the colour of the clarifier effluent is fairly linear and consistently below 5 colour units against any variances of the input parameters. The mean of the time series is 2.0 with a standard deviation of 0.8. On the contrary, the clarifier effluent turbidity fluctuates

[1] The alkalinity value used in this case study is the total alkalinity measured by the Titration Method.

significantly and is nonlinear. Its time series has a mean of 2.5 and a variance of 2.25. Either effluent quality parameter can be used as the control output depending on the type of control model chosen. However, results for turbidity will be presented.

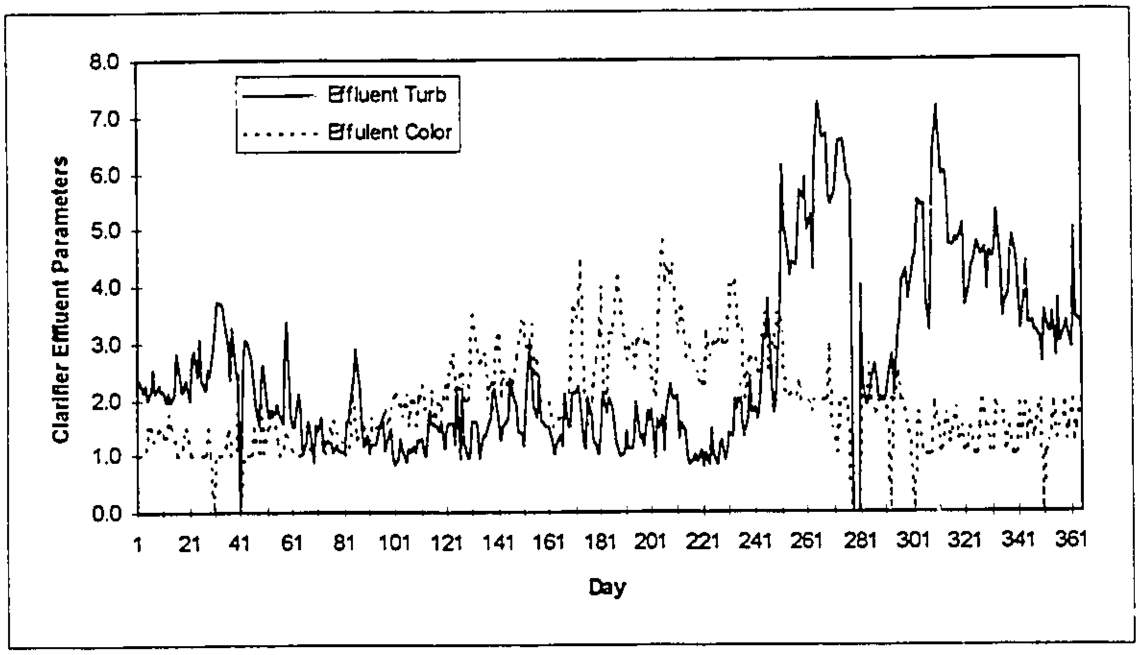


Figure 5-2 The Daily Record of Clarifier Effluent of 1993

Before knowledge of the process unit is applied to construct a neural network control model, it is necessary to identify the characteristics of process control in water treatment processes. There is a significant difference in the control precision requirements for process control between water treatment plant and commercial chemical plants, for which substantial research on process control has been completed. In a commercial chemical plant, normally the chemical reactions are controlled at a constant state. One example is the reactions to maximize the yield of an end product. In this kind of precise control style, the control system must be able to react to the disturbance quickly and adjust the process back into its normal operating state. Therefore, the robustness and the convergence are the key issues of the control system design.

Compared to a commercial chemical plant, the process control in the water treatment plant is much less stringent. Several factors contribute to this relaxed control. First, the concentrations of the chemicals in the water treatment reactions are relatively low and the reactions are fairly rapid. Thus, there is little need to control the process to a specific state for a maximum yield. Second, the resident time in most of the processes in the water treatment plant is determined by the physical aspects of the process and is typically long. For example, the resident time in the control unit to be studied in this project is about four hours. Within these four hours, the coagulation process only takes less than 10 minutes. Most of the residence time occurs in the flocculation and sedimentation processes. Third, the concept of multiple barriers in drinking water treatment design also helps to relax the control practice in the front units. For example, the temporary spike of turbidity in the sedimentation tank effluent can be overcome by the operation of the filtration process. In addition there are usually large reservoirs at the end of the treatment process, short term and minor deviation from the optimum operation will not significantly change the quality of the final product. However, the process control of processes close to the end should be tighter as additional barriers may not be presented. Finally, a significant difference between process control requirements for water treatment and chemical plants results from the variation in raw water quality that can occur. In most chemical plants, feed stock used in the processes normally have a consistent quality. In contrast, raw water quality parameters may vary several orders of magnitude over short time periods.

In the following sections, the use of neural network models in the Internal Model Control strategy and its various control schemes will be investigated for the real-time automatic process control. The capacity of neural network controllers, the methodologies required to develop them and the appropriate application of each control scheme will be discussed.

3. Neural Network As A Plant Process Model

IMC is a model based control strategy. A process model is explicitly used to predict future process behaviour. The same process model is also inverted to calculate the control action. With this strategy, the mechanism of the process must be well understood and can be expressed in equations. Although a similar concept was encountered in late seventies (Rivera, et al., 1986), the IMC for the single input-single output (SISO) linear process was first introduced by Garcia and Morari (1982) and later extended for multivariable systems (Garcia, 1985a and 1985b). A further expansion of IMC for nonlinear systems was illustrated by other independent researchers (Economou and Morari, 1986; and Li and Biegler, 1988).

As shown in Figure 5-3, IMC consists of three parts: (1) an internal model to predict the effect of the manipulated variables on the output; (2) a filter to achieve a desired degree of robustness; and (3) a control algorithm to compute future values of the manipulated variable such that the process follows a desired trajectory. This structure gives IMC a highly intuitive appeal. A model based prediction of the output is attractive in the presence of output constraints. Any constraint violations can be anticipated and corrective action can be taken. In the water treatment industry, in order to produce high quality drinking water, a set of output constraints are placed on every major process. For example, the current maximum clarifier effluent turbidity allowable in the Rossdale WTP under normal situations should be less than 5 NTU and the process goal is less than 2 NTU.

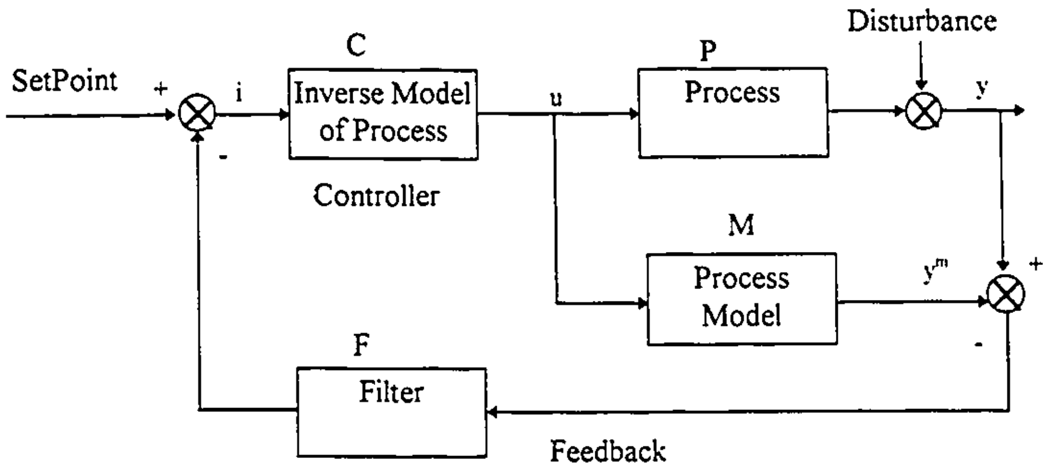


Figure 5-3 The IMC Structure

Although the control theory for the IMC model is developed extensively for the linear system, the application of the IMC to a nonlinear system is still incomplete (Ungar, 1990). The major problem is in establishing the inverse process model for nonlinear systems. Even if the mechanism for the forward process is known, theoretically, the perfect inversion of the forward process exists only for limited classes of nonlinear functions (Economou and Morari, 1986). For the rest of the nonlinear functions, it is still not clear whether the theoretical inversion of the process exists. For these nonlinear functions, the neural network approach could be an interesting solution as the inverse model can be built from the past records.

The IMC with neural networks as process models is divided into direct and indirect IMC (Psichogios and Ungar, 1991). In the direct IMC, the inverse model of the process is replaced by a neural network model. In this case, given the current state of the dynamic system and the target state for the next sampling instant, the network is trained to produce the control action that drives the system to this target state. The resulting network can then be used as a controller in a feedforward fashion. In the indirect IMC, the process model is replaced by a neural network model. The neural network is trained with input-output data from the dynamic system to represent the forward dynamics. Thus, the network will be able to predict the process output given a set of input parameters. This prediction can then be used by the control algorithm to calculate the control action.

For a system with nonlinear inputs and outputs, both the forward and the inverse process models can be replaced by neural network models. This results in a hybrid IMC system. Due to its flexibility to accommodate the nonlinear systems, the hybrid IMC system was first built and tested by Bhat and McAvoy (1990) and Ydstie (1990). In their studies, both neural networks were trained off-line. It was found that using a neural network as a controller (the inverse plant model) has the advantage that the control action is calculated explicitly and hence quickly. However, if the process under study is well understood and the exact equations are available (for example, a chemical reaction in CSTR studied by Bhat and McAvoy (1990), the traditional IMC controller performed better than the neural network IMC controller (Bhat and McAvoy, 1990 and Psichogios and Ungar, 1991).

There are two explanations to this observation. First, depending on the completeness of the data and the process to train the neural network model, the best available model may not always be obtained and approximation errors exist. The magnitude of this approximation error will be directly related to the accuracy of the model. However, if the equations describing the process are available an[illegible]sable, the exact control action can be calculated for the traditional IMC approach. In this case, the error will be much less than that of the neural network model. Second, in the hybrid system, there are two neural network models. If the precision of the models are sacrificed in the training process, it is possible for them to generate a combined error which is intolerable. In a delicate chemical process, a combined total error of 5% from both neural network models will be sufficient to destroy the effectiveness of the controller (Ydstie, 1990).

To increase the control precision in the hybrid system, several control schemes were suggested (Psichogios and Ungar, 1991). Since the main problem is the potentially large combined error if two neural network models are used together, the intuitive solution is to use only one neural network model in the control scheme. Under this direction of thinking, Psichogios and Ungar (1991) suggested two new control schemes. The first scheme they proposed was a direct IMC with the process model being substituted by a feedforward delay (Figure 5-4). The reason for the use of feedforward delay was based on the assumption that if the inverse process model C in Figure 5-3 was a perfect inverse of the plant model M, then C×M=1. Thus the input at time t to C would be the output of M at time t + Δt where Δt was the length of sampling interval. In other words, y^m at t + Δt is equal to the input i to C at time t. In control, this relationship can be simply replaced by a delay.

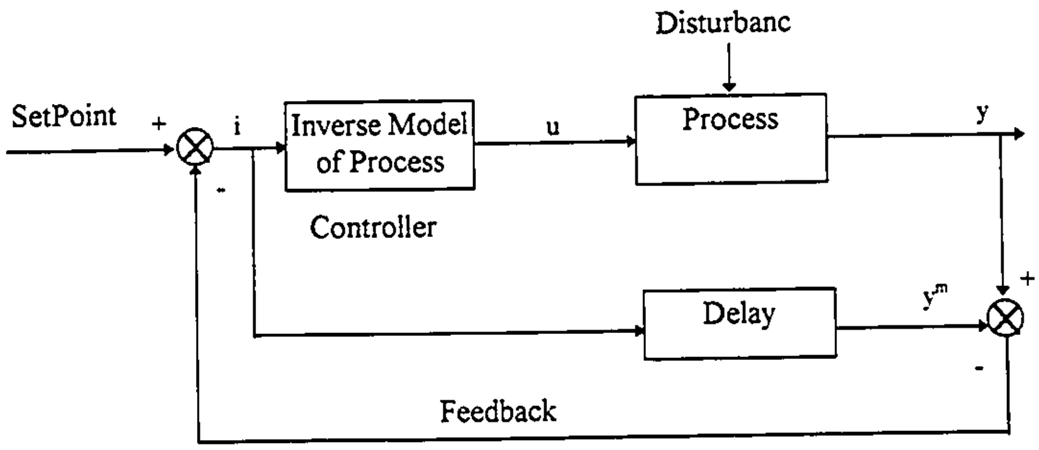


Figure 5-4 The Modified Direct IMC

The second scheme is a modified indirect IMC in which the inverse model of the process is replaced by a conventional controller whose control signals are calculated from the direct inverse of the process model outputs. By considering that the network representing the forward model implicitly contains the Jacobian of the process, Psichogios and Ungar (1991) derived Equation 1 for calculating the control action in a recursive fashion:

$$u^{n+1} = u^n - \frac{y^m - e}{\partial y^m / \partial u^n} \qquad (1)$$

where u = the manipulated variable by the controller;

e = the desired operation state of the plant;

y^m = the output of the neural network plant model.

With Psichogios and Ungar's study (1991), it was found that the modified direct model had difficulty in converging to the desired state. Instead, the system had a constant offset from the desired state. Obviously, this approach has some severe drawbacks. For the modified indirect model, the performance was considerably better than that of the modified direct model. However, the partial derivatives of both y^m and u in Equation 1 have to be calculated by the approaches of numerical analysis. Solving such a nonlinear equation is not a trivial task. Sometimes, the solution may not be found due to a poor initial guesses. The lengthy computation time is also a major concern for on-line operation. Finally, both control schemes were designed for SISO systems, especially for the indirect model as Equation 1 will only accept one input variable.

4. A Control Model For Coagulation-Flocculation-Sedimentation Process

Upon the completion of the analysis of several neural network control schemes, an open-route neural network control model for the coagulation-flocculation-sedimentation process was proposed for the Rossdale WTP (Figure 5-5). The design was a feedforward control model with a feedback line as an optimal reference point. The feedforward controller consists of three major parts in series: 1) a neural network plant reference model to predict the process output from the input variables; 2) a control information filter for calculating, selecting and passing the preferred control information from sources such as the process standards, the feedback if appropriate, and the prediction from the previous plant reference model, to the next unit; and 3) a neural network inverse model of the plant which accepts the predicted process output and uses it to generate the proper chemical doses.

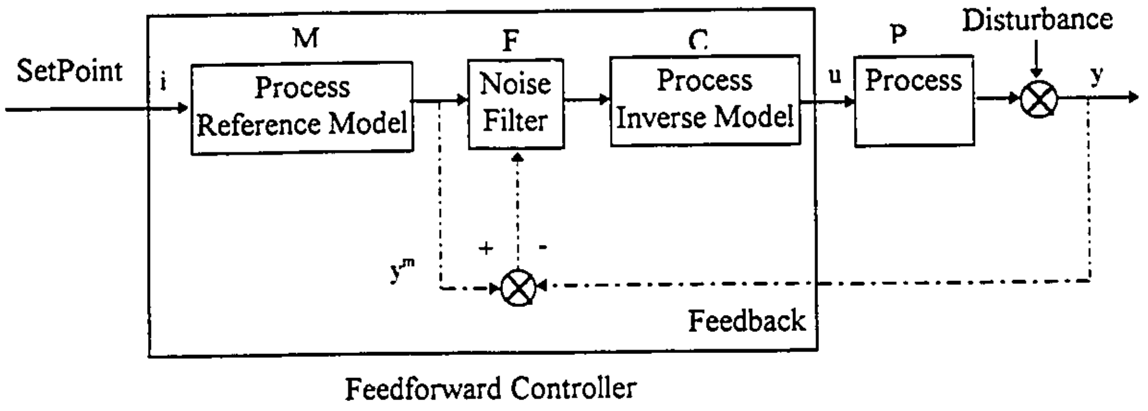


Figure 5-5 The Proposed Process Control Model For Coagulation-Flocculation-Sedimentation

There are several concerns with the design of this model. The first major consideration is the length of the overall process time. As mentioned in Section 3, the coagulation-flocculation-sedimentation is a continuous and lengthy process (four hours in average), and the intermediate state of the process is difficult to assess. This means the value of the process feedback for the constant control actions is limited as the feedback available at the current time was the result of the control setting approximately four hours prior. However, if the raw water quality is consistent within the residence time, the feedback can be used as a reference point. This is the case for the Rossdale WTP during the winter time as its raw water source, the North Saskatchewan River, is covered with ice and the variation in river water quality is minimal. However, this is not the case in other seasons. Therefore, the feedback loop in Figure 5-5 is an option (shown as the dashed lines).

Under this feedforward control design with an optional feedback, the selection of the proper control scheme is another key issue. Since the exact mathematical description for the processes under study is not available, the strict concept of IMC may not be appropriate because of the consideration of the inversability of the forward model. Furthermore, since many raw water quality parameters directly influence the control actions, they must be incorporated into the process model. This requires the process model to be able to deal with multiple inputs. Under all these constraints, the only possible solution is the neural network approach. With the flexibility of neural networks to learn the forward and inverse plant model directly from the historical data without knowing the exact mechanism of the process, and with the design concepts from the traditional IMC strategy, a design of a three-part neural network feedforward controller was created.

The functions of each component of the feedforward controller and the function as a whole are explained as follows. The first part of the controller is the neural network plant reference model. This reference model is built to take the necessary water quality parameters plus the dosages of alum and PAC to predict the clarifier effluent turbidity under a scenario. Thus, given a set of water quality parameters which are fixed conditions, with this reference model, it is possible to make some initial guess on the dosage of alum and PAC for the control action and run them as the input data to the reference model to obtain the effluent turbidity for each guess of the dosage of alum and PAC. The neural network inverse plant model does the job in a reverse way. It takes the raw water quality parameters, the assumed dosage of PAC and the assumed effluent turbidity whose value is desired as the inputs and predicts the dosage of alum for the control action in order to reach that assumed effluent turbidity.

There is a slight difference in the functional requirements for each model. This results in different training process for each model. Since the plant reference model must be able to predict the effluent turbidity with all kinds of dose scenarios of alum and PAC, the reference model is not only trained with good operation cases, but also the cases in which either underdosed or overdosed occurred. In comparison, the reverse model is built with the purpose of being able to predict the best possible alum dose in order to achieve the desired effluent turbidity. Therefore the inverse model is trained with good cases only.

With both the forward plant reference model M and the inverse plant model $C = M^{-1}$, theoretically the product of $M \times C$ should be the identity under the assumption that C is the perfect inverse of M. That is if the alum dose AD_1 is the correct dosage and set to be the input to M, M will subsequently predict the effluent turbidity ET_1 as the result of adding AD_1 to the process. If this ET_1 is then used as the input to C, then the output of C should be AD_1. With this relationship in mind, an iterative numerical analysis process is designed as shown in Figure 5-6 to calculate the optimum alum dose and PAC dose for each set of raw water quality.

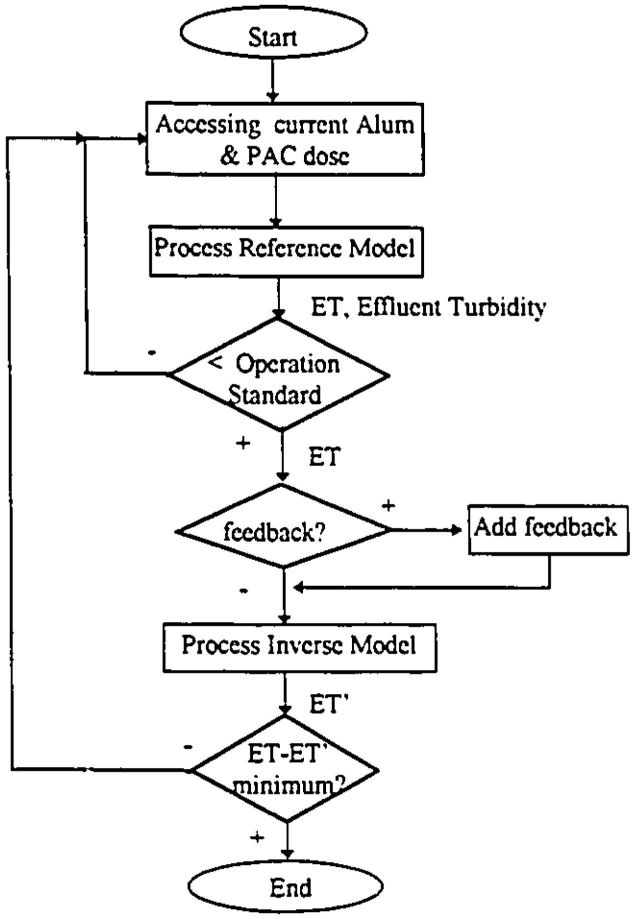


Figure 5-6 The Alum Dose Optimization Process

Both the plant reference model and the plant inverse model are the integrated parts of the iteration and should be used as a whole for control. When the numerical analysis tool is used to search for a solution inside a multiple-input nonlinear relationship through an iterative approach, the solution may not be unique because multiple inputs results in a solution surface with multiple dimensions. Depending on the analytical method used, it is possible for the numerical analysis to converge on many local extremes on the multiple-dimension solution surface with different initial settings of the inputs. Usually, many of these local extremes may not be the best solutions. Consider the following example in the coagulation dosage control. With a poor initial guess of alum dose for AD_1 the process reference model could falsely predicts an acceptable effluent turbidity of ET_1. Without the safeguard of the process inverse model, the alum dose of AD_1 would be used. However, if the inverse process model is used to generate an alum dose AD_2 by ET_1. By comparing how close AD_1 is to AD_2, the goodness of alum dose AD_1 will be tested by the principle of perfect inverse as stated in the previous paragraph, which states $AD_1 \times M \times C$ should be equal to AD_1. By this approach, the control model should generate the good answers only. From these good answers, the operator can then select a best possible solution.

As illustrated in Figure 5-6, the detailed control process to treat the raw water is explained as follows. Since the water treatment process is a continuous process, samples of the process are taken from time to time. Thus to assess whether the alum dose AD_1 and PAC dose PD_1 in the previous sampling time are good for the current condition, their values can be inputted into the process reference model with the raw water conditions to predict the effluent turbidity ET_1. ET_1 is then passed to the filter component of the controller. If ET_1 is greater than the effluent turbidity limit, then the filter will determine that the initial guess of alum dose AD_1 was poor and will initiate another guess. If ET_1 is under the limit, then the filter will pass the value to the inverse plant model and will generate the alum dose prediction AD_2 for matching that effluent turbidity. Ideally, if the inverse model is perfect, then $AD_1 = AD_2$ and the usage of the alum dose, AD_1, for the control action is justified because it is proved by both forward and reverse model. However, as mentioned, the plant reference model and the inverse plant model are purposely built with slight differences. Thus, numerically there could be a difference between AD_1 and AD_2 predicted by these two models. The next step is then to adjust the alum dose in the vicinity of the AD_1 in a iterative process to optimize this difference to a minimum. This task can be easily done by the conventional numerical analysis. Since the range of the practical alum dose is limited (about 15 mg/L to 125 mg/L), the computer run time should be short.

Overall, the accuracy of this control algorithm relies on the accuracy of the two neural network models. As discussed in the previous section, if the search for the optimum model is incomplete, the combined error of the two models could jeopardize the whole control model. To obtain the best possible neural network models, factorial design concepts were used. Also, in order to ensure the training data set is representative of the study domain, a total of 1978 cases of input-output data were screened from the real plant operation data from 1992 to 1994. The evaluations of the best models obtained from this data set are presented in Table 5-1.

Table 5-1 The Statistical Data for the Candidate Models

Candidate Model	Plant reference model	Plant inverse model
R square for the whole data set	0.873	0.975
R square for the Training data	0.974	0.98
R squared for the test data	0.237	0.946
Root mean squared error	0.2 NTU	7 mg/L
Mean Absolute Error	0.2 NTU	2 mg/L
Correlation Coefficient r	0.935	0.988

It is obvious that the accuracy of the plant reference model is far less than that of the plant inverse model. This is due to the fact that it must have the capacity to predict the poor operational conditions and as a result, the plant reference model has to learn all scenarios which include cases of overdosing and underdosing. Therefore, it is much more difficult for the neural network model to learn as the learning data contain more noise.

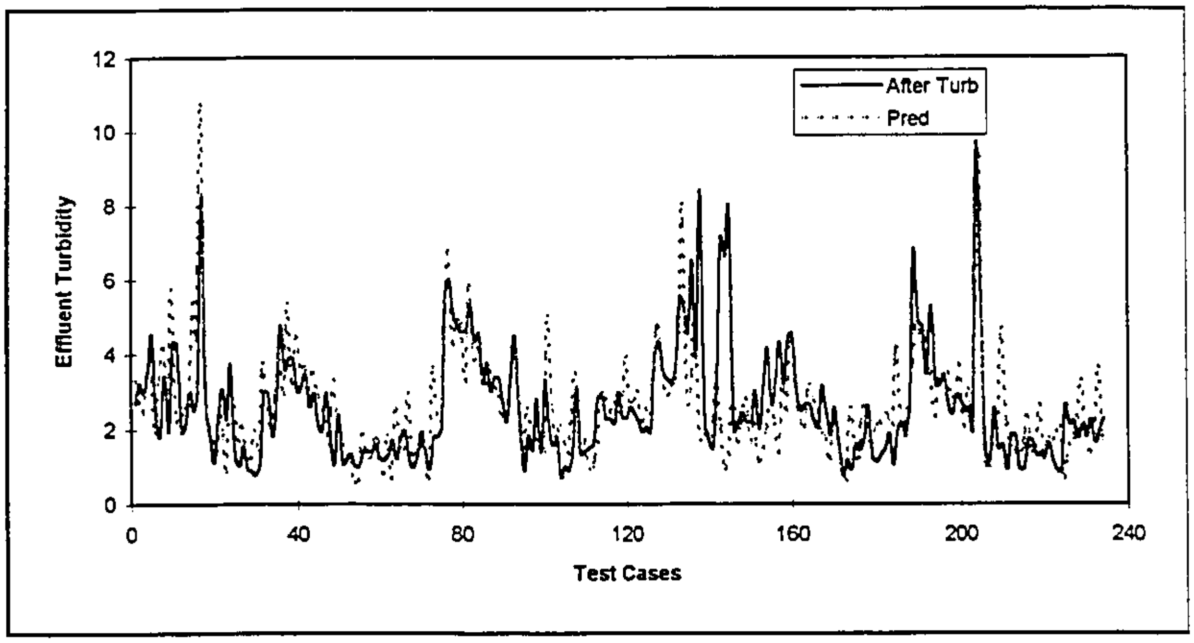


Figure 5-7 The Model Prediction Veruse the Testing Data for the Plant Reference Model

Overall, both models still converge and generalize well. This can be found from the graph of the model prediction veruse the testing data set as shown in Figure 5-7. Figure 5-8 is the similar plot for the plant inverse model. The quality of the model can also be evaluated by the mean absolute error as listed in Table 5-1. For the plant reference model, the mean absolute error is 0.2 NTU which is less than 10% of the mean effluent turbidity of 2.5 NTU. For the plant inverse model, the percentage of the mean absolute error (2 mg/L) to the mean of source data (30 mg/L) is even lower— 6%. For a relaxed feedforward control process, these results are acceptable.

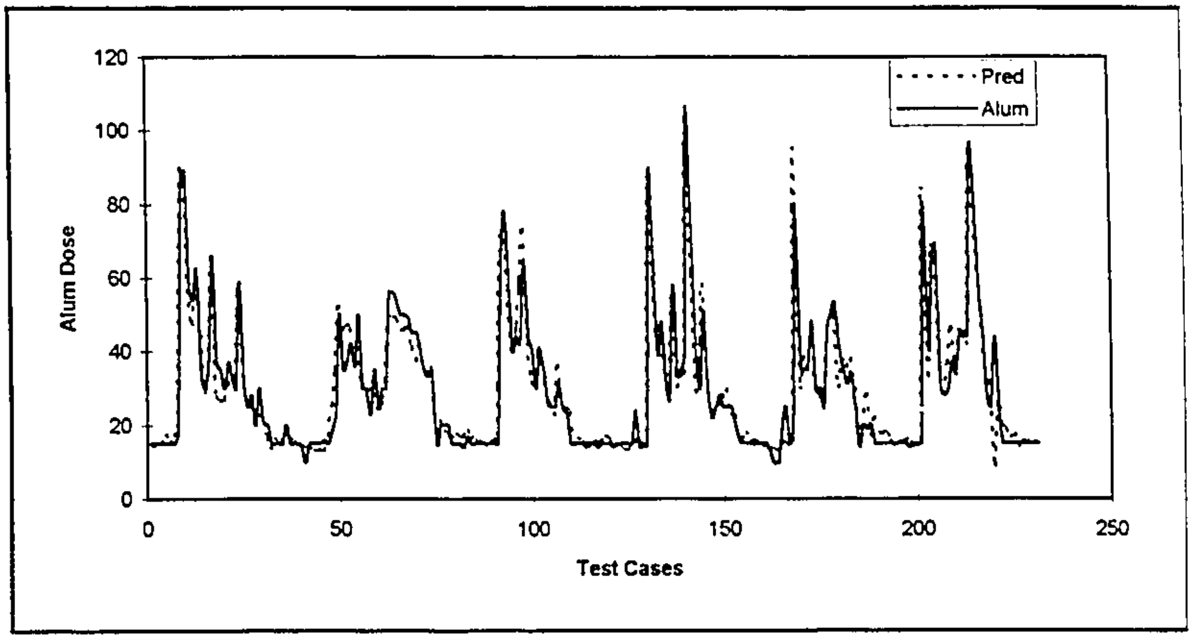


Figure 5-8 The Model Prediction Veruse the Testing Data for the Plant Model Inverse

With this ANN control model, multiple test runs were conducted and two results are presented in Table 5-2 to Table 5-4. Table 5-2 shows one case of raw water conditions in late October. The actual operation actions were alum dose 23 mg/L, PAC dose 19 mg/L and the effluent turbidity 8.5 NTU. The solutions obtained from the reference model are listed in Table 5-3. By investigating various alum and PAC doses, a total of four solutions were found. In every situation, the plant would perform better than the actual operation if the specified alum dose and PAC dose were used. The results also indicate that there is no feasible solution for effluent turbidity which is less than 3 NTU. For the case shown in Table 5-4, the actual operation actions were alum dose 66 mg/L, PAC dose 10 mg/L and effluent turbidity 8.5 NTU. The control model predicts an effluent turbidity of 1.5 NTU if the alum dose is 48.5 mg/L and PAC dose is 5 mg/L.

Table 5-2 The Raw Water Condition 1

pH	Raw water Temperature	Alkalinity	A0-A1	Average Turbidity	T0-T1	Average color	C0-C1	Overflow rate
8	6	153	-2	37	28	7	3	36

Note: A0-A1 = the difference of current alkalinity to the alkalinity in the previous time interval;

T0-T1 = the difference of current turbidity to the turbidity in the previous time interval;

C0-C1 = the difference of current colour to the colour in the previous time interval.

Table 5-3 The Control Model Solutions

Solution	Alum dose (mg/L)	PAC dose(mg/L)	Effluent Turbidity (NTU)
1	15	5	3.6
2	15	10	3.7
3	17	15	3.6
4	18	20	3.5

Table 5-4 The Raw Water Condition 2

pH	Raw water Temperature	Alkalinity	A0-A1	Average Turbidity	T0-T1	Average color	C0-C1	Overflow rate
8.2	1	139	8	121	20	4	0	48

For the situation where the plant feedback is applicable, such as during the winter in the Rossdale WTP, the feedback is calculated as the difference between the value predicted by the reference model and the actual plant process result. If the actual value is greater than the reference value, the absolute value of the difference will be then subtracted from the desired effluent turbidity. Otherwise, the difference will be added to the desired effluent turbidity. This process is completed by the filter and the new desired value is passed to the inverse plant model in another iterative process to find the optimum. By this method, although there is no guarantee to achieve a stable operation state, the feedback information is utilized to modify the desired effluent turbidity to achieve the goal of process control. Once the optimum alum dose and the PAC dose are determined, the control signals could be sent to the actuators which will subsequently adjust the chemical feed accordingly.

5. Conclusion

The research of neural network in process control has only recently started. However, the ability of neural network to learn directly from historical process data and then represent it as a process model has a promising future for application to process control of the nonlinear plant. In the water treatment industry, because of the complex chemical and physical reactions, many of the treatment processes exhibit nonlinear behaviour and it is difficult to establish effective and economical process control by conventional approaches. In this paper, the neural network modeling approach was studied for its potential in the water treatment process control.

Upon comparison of each type of neural network control model suggested by researchers and the thorough analysis of the mechanism of the treatment process, a feedforward neural network control scheme was proposed for the coagulation-flocculation-sedimentation process in the Rossdale WTP in Edmonton, Alberta, Canada. With near 2000 process control data, a neural network plant forward reference model and a neural network plant inverse model were built and form the major components of a software feedforward controller.

This software controller takes the multiple inputs describing the raw water quality and then calculates the necessary alum dose and PAC dose for the control actions. When the raw water quality is consistent within the normal residence time of the process, feedback from the actual plant performance can also be included in the model to form a closed-loop control. Another benefit of this controller is that it can either be integrated with the actuator to operate on-line independently or be used off-line by the operator to predict the optimum chemical doses as a reference to the actual chemical dosing. Although this controller performs well on the data learned, the final remarks on the generalization of this neural network controller will be verified with real on-line operations at the Rossdale WTP.

References

Bhat, M. and McAvoy, T.J. (1990), "Use of Neural Nets for Dynamic Modeling and Control of Chemical Process Systems", *Computers and Chemical Engineering*, Vol. 14, No. 4/5, pp. 573.

Canadian Water and Wastewater Association (1993), Guidelines for Canadian Drinking Water Quality: Water Treatment Principles and Applications —A Manual for the Production of Drinking Water, Minister of Supply and Services Canada Cat. No. H46-1/28-1993E.

Economou, C.G. and Morari, M. (1986), "Internal Model Control: 5. Extension to Nonlinear Systems", *Ind. Eng. Chem. Process Des. Dev.*, Vol. 25, pp. 403.

Garcia, G.E. and Morari, M. (1982), "Internal Model Control: 1. A Unifying Review and Some New Results", *Ind. Eng. Chem. Process Des. Dev.*, Vol. 21, pp. 308.

Garcia, G.E. and Morari, M. (1985a), "Internal Model Control: 2. Design Procedure for Multivariable Systems", *Ind. Eng. Chem. Process Des. Dev.*, Vol. 24, pp. 472.

Garcia, G.E. and Morari, M. (1985b), "Internal Model Control: 3. Multivariable Control Law Computation and Tuning Guidelines", *Ind. Eng. Chem. Process Des. Dev.*, Vol. 24, pp. 485.

Li, W.C. and Biegler, L.T. (1988), "Process Control Strategies for Constrained Nonlinear Systems", *Ind. Eng. Chem. Res.*, Vol. 27, pp. 1421.

Psichogios, D.C., and Ungar, L.H. (1991), "Direct and Indirect Model Based Control Using Artificial Neural Networks", *Ind. Eng. Chem. Res.*, Vol. 30, pp. 2564.

Rivera, D. E., Morari, M. and Skogestad, S. (1986), "Internal Model Control: 4. PID Controller Design", *Ind. Eng. Chem. Process Des. Dev.*, Vol. 25, pp. 252.

Ungar, L.H. (1990), "A Bioreactor Benchmark for Adaptive Network-based Process Control", ", Neural Network For Control, Edited by W. T. Miller, R. S. Sutton and P. J. Werbos, The MIT Press, Cambridge, Massachusetts, pp. 387.

Ydstie, B.E. (1990), "Forecasting and Control Using Adaptive Connectionist Network", *Computers and Chemical Engineering*, Vol. 14, No. 4/5, pp. 583, 1990.

Chapter VI

Conclusions

1. Results and Discussions

In this thesis, unique research into the application of the artificial neural network (ANN) modeling approach in the water treatment area is provided. In chapter 1, recent developments of several artificial intelligence (AI) modeling approaches are examined. By comparing the potential capacity of each approach for the water treatment research, the ANN modeling approach is selected for further exploration.

In Chapter 2, to obtain a better understanding of this relatively new AI modeling approach, a full review of ANN with its recent developments was conducted. First, the biological origin of this AI approach was identified, then this biological paradigm was used to explain how the basic functions of a human neuron are simulated and subsequently organized into a complex architecture called artificial neural network. Since this AI approach has a distinctive ability to learn directly from examples and generalize a concept out of these examples, the chapter further explained how the learning algorithm functions and how the learning process should be conducted to obtain an optimum solution. Several time series concepts were also introduced in Chapter 2 to establish the theoretical foundations for a time series data rendering technique which was used in the ANN research in the subsequent chapter.

With knowledge of ANN approach, Chapter 3 and Chapter 5 illustrate a step by step procedure of how the ANN modeling technique can be used to provide a better process control and plant management in water treatment industry to achieve the goal of producing high quality drinking water in an efficient and economical way without sacrificing the production standards.

Normally, a conventional water treatment plant consists of a sequence of chemical and physical processes to remove the micro particles, the metal contaminants and the disease causing agents from the source water. Since most of these processes occur at the micro-scale and the process mechanisms are inherently nonlinear, it has been difficult for conventional modelling approaches to provide a reliable model to monitor and control the process. Therefore, the current control practice in water treatment plants still heavily depends on operator's personal experience. Obviously, this control style is subjective and passive.

To overcome these problems, two potential approaches were investigated. First, in the water treatment plant, the quality of water produced highly depends on the characteristics of raw water. Sudden changes in raw water quality, such as the colour peaks in the early spring and the high turbidity resulting from the intensive rainfall in mid summer, can negatively impact finished water quality. Therefore, if the quality of raw water can be predicted ahead of time, a series of control measures can be prepared to prevent and reduce the negative impacts on finished water quality. Second, since the resident time for each process is long, typically hours, if the chemical dosing is adjusted depending on the plant output, there will be a time delay which could render the adjustment incorrect. Thus, a real-time reactive model which consists of a monitoring system to constantly sample the raw water quality and a chemical dosing model to predict the proper chemical dose is desired. In this thesis, the neural network modeling approach was used to achieve these two tasks. Chapter 3 explains how the first task was achieved. Chapter 5 illustrates the procedure to complete the second task.

In Chapter 3, the objective is to assess the capacity of neural network to predict the water quality of the North Saskatchewan River one day ahead. The raw water color is selected as the parameter to forecast. In this chapter, the main focus is on how engineering knowledge can be used in the model building process.

Since a neural network model is a black box type relationship, the analysis of the source data and the input-output design becomes an important aspect related to the accuracy of the model. Raw water colour is influenced by many factors in a large watershed. The challenge is then to narrow down the scope of this study from a vast array of information of a large river basin to a set of real-time measurable and representative parameters. Based on knowledge of system and the available information, the parameters selected were the river flow rate, the turbidity, temperature degree day, the rainfall upstream, and the raw water colour itself. With the analysis of the source data, it was found that most of the parameters studied had a very strong autocorrelation. This means that the current values of the time series parameters were significantly influenced by its values in the near past. Hence, the time series of these parameters were also incorporated into the inputs. Further study suggested that if the pure time series were used as inputs, there was a lot of data noise incurred by the high fluctuation of the original time series. To reduce this kind of data noise, the Box-Jenkins time series analysis tools were used to render the noisy time series into the stationary time series. As the result, the forecasting accuracy was significantly improved. Overall, the neural model was able to predict with a mean error of 0.6 TCU. Compared to the time series modeling approach which has a mean of 9.5 TCU and a standard deviation of 9 TCU, the result is satisfactory. Based on the experience obtained in this case study, a protocol was developed to ease the process of building similar neural network models for the other raw water quality parameters.

In Chapter 5, the goal was to build a neural network real-time process model which can be used on-line real time to determine the proper dosage for alum and PAC. The chapter began with a general survey of the neural network research in internal model control. By comparing the methodology of several control schemes by assessing their strengths and weakness, a neural network feedforward controller was proposed for the coagulation-flocculation-sedimentation unit at the Rossdale Water Treatment Plant.

The main idea of the design originated from the internal model control strategy which benefits from using both a forward plant model and an inverse plant model to provide the stability and robustness needed for control process. The design also considered that the resident time in this particular process unit is long and there is no intermediate state of the process which can be measured effectively. Therefore, the control model used was an open-loop feedforward process. A data filter was encoded into the controller to handle an optional feedback loop which will be used only if the raw water quality is stable throughout the resident time.

With this design concept, two neural network models were subsequently built from approximate 2000 cases selected from the historical control data. These two models were then translated into the spreadsheet functions which can be called within the spreadsheet solver to calculate the required alum dose and PAC dose for the different raw water qualities. The average calculation time was found to be about 10 seconds. With this capacity, this software controller can either be integrated with an actuator as an on-line chemical feeder or used in the central control computer to provide guidance for the operator to monitor the ongoing process. Further on-line tests are needed to verify the generalization of the model.

Although both applications are considered successful, many questions about the neural network modeling remain unanswered. This is largely due to the nature of the black box type model associated with neural networks. In Chapter 4, an attempt was made to answer some of these questions. By treating the neural network approach as an inductive learning process, the question of why a functional model can be built by learning from actual examples was explained. A detail discussion of the theoretical size of the learning examples was presented from the viewpoints of the inductive learning and the computational theory. However, since the theoretical limits for the size of the learning example involves the size of a hypothesis space which is infinite in nature, the discussion was continued on how to partition the hypothesis space and how to place reasonable restrictions to reduce the size of the hypothesis to be searched. Finally, two statistical analysis tools of factorial design and response surface methodology were used to make the hypothesis space tractable and eventually find the best available model associated with the problem.

2. Recommended Further Research

Based on the results of this study, a full plant automated process control system can be developed. The proposed automatic process control system will utilize neural network techniques developed in this thesis, along with the computer data communication technology to create a powerful concept for process control in modern water treatment processes. It is a system capable of real time process monitoring, real time process optimization, warning of potential hazardous situation and management decision making. It is also a system capable of constant learning and fine-tuning.

As illustrated in Figure 6-1, a complete expert system will consist of several independent modules. Its basic functionality can be briefly described as follows. The expert system is the control center. It receives or retrieves information from the other modules such as the sensor array and the database, makes the decision and sends out the commands to the modules such as the chemical dose control and the hydraulic system. All of the information flow will be handled by a tele-communication unit. The system also communicates with the human operator through the user interface for the tasks such as receiving commands, learning the management decisions and presenting monitoring and control information.

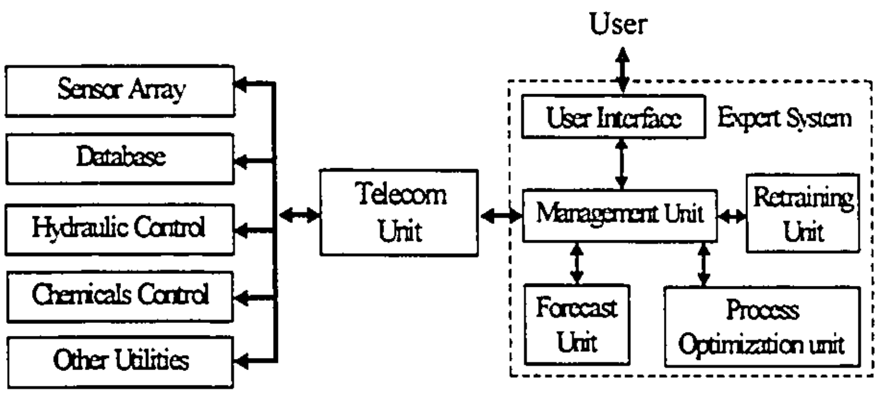


Figure 6-1 The Automatic Control System

The focus of this recommended research is to develop the expert system shown in Figure 6-1. Inside the expert system module are the functional units similar to those described in section 5. They are mainly a management shell, a forecast unit, a process optimization unit and a network retraining unit. The management shell will send its requests to the forecast unit to obtain the information such as daily water demand, future raw water quality, etc.

Additional information, together with the forecast values, will then be passed to the process optimization unit to obtain the process control data. The process control data are then converted into control signals and sent out to the responding modules through the telecommunication unit. The retraining unit is used for learning both new management decisions and new or altered processes that are added in the future. All these function units will be supported by the neural network technology.

Since the whole system is modularized, each module can be developed separately. This modularization will also allow some existing systems such as automatic data sampling and collection and hydraulic system central control to be integrated into the system to save money and time. The recommended study will focus on the development of the expert system module and the necessary computer organization to bring all these modules together to form the automatic control system.

Appendix I

Back Propagation Calculations

Backpropagation Calculations[1]

Back propagation is certainly the most popular learning algorithm used by ANN developers. A back-propagation neural network can be defined as a multilayered network for which the error between the desired output and the actual output is reduced as successive pairs of input-output data are shown (Figure 1). In other words, for each example presented, the network propagates the input forward through the hidden layer to the output layer, calculates a global error at the output layer, and finally propagates the errors back to the input layer. This learning process allows the network to adjust its connection weights until convergence is reached.

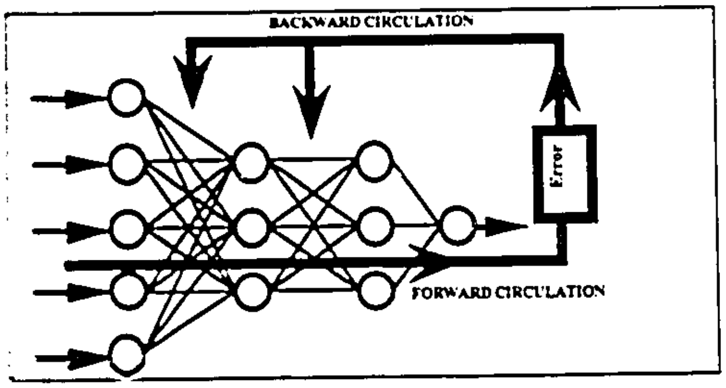


Fig. 1. Back-propagation neural network

Given the network illustrated in Figure 2 the global error can be calculated as follows:

$$E = 0.5 \bullet \sum_{p=1}^{n} (d_p - x_p^{[out]}) \qquad (1)$$

where $x_p^{[out]}$ is the response of the pth neuron of the output layer calculated by the network, d_p is the desired value that corresponds to this output neuron, and n is the

[1] The description of the backpropagation algorithm is modified with permission from Manuel Rodriguez's Ph.D thesis completed in the department of Civil Engineering in the University of Laval.

number of neurons in the output layer. Because the aim of the learning process is to minimize this global error, a local error for each output neuron has to be defined.

For example, the local error, e_k[out], for the kth neuron of the output layer can be expressed as follows:

$$e_k^{[out]} = \frac{\partial E}{\partial I_k^{[out]}} \qquad (2)$$

where, $$I_k^{[out]} = \sum_{q=1}^{m} (w_{kq}^{[out]} \bullet x_q^{[L]})$$

(3)

w_{kq}[out] is the weight of the connection linking the qth neuron in layer L and kth neuron in the output layer, x_q[L] is the current output state of qth neuron in layer L, and m is the number of neurons in the layer L.

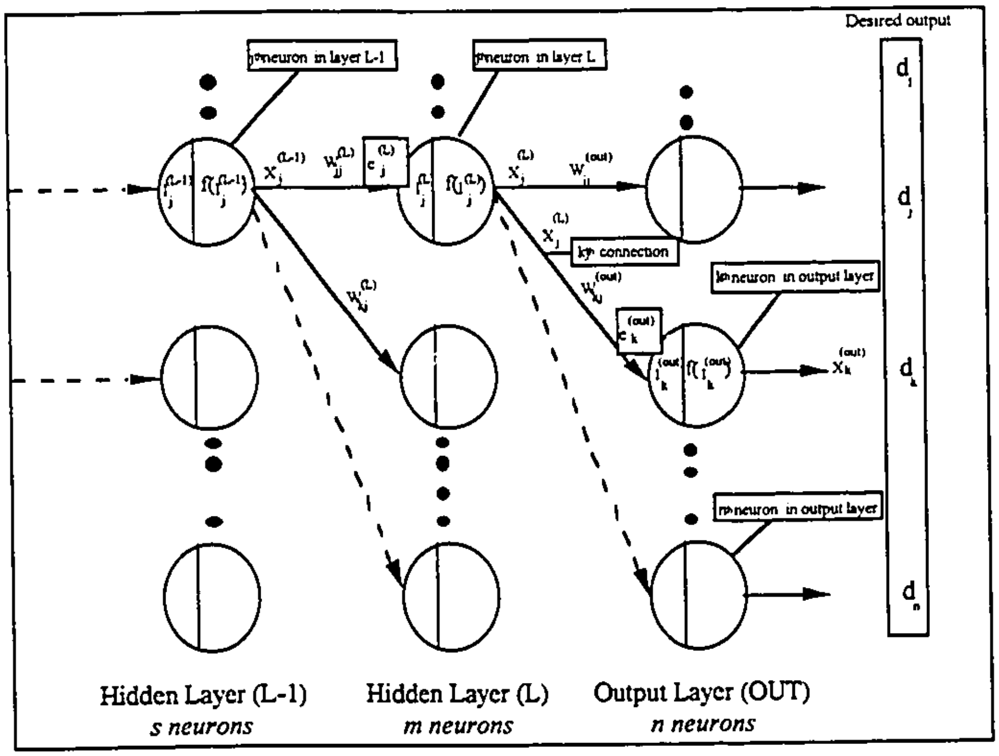


Fig. 2. Example of network using back-propagation algorithm

Equation 2 can be written as follows:

$$e_k^{[out]} = \frac{\partial E}{\partial x_k^{[out]}} \bullet \frac{\partial x_k^{[out]}}{\partial I_k^{[out]}} \tag{4}$$

On one hand, because $x_k^{[out]} = f(I_k^{[out]})$ then $\frac{\partial x_x^{[out]}}{\partial I_k^{[out]}} = f'(I_k^{[out]})$ (5)

On the other hand, $\frac{\partial E}{\partial x_k^{[out]}} = (d_k - x_k^{[out]})$ (6)

Replacing terms from equations 5 and 6 in equation 4, gives:

$$e_k^{[out]} = f'(I_k^{[out]}) \bullet (d_k - x_k^{[out]}) \tag{7}$$

If the transfer function $f(\psi)$ is the sigmoide, then:

$$f(\psi) = \frac{1}{\left(1 + e^{-\psi}\right)} \tag{8}$$

and $f'(\psi) = f(\psi) \bullet (1 - f(\psi))$ (9)

The weight adjustment (incremental or decremental) of each (k_j)th connection reaching the output network is done using a gradient descent rule, the *delta weight equation*,

$$\Delta w_{kj}^{[out]} = -\alpha \frac{\partial E}{\partial w_{kj}^{[out]}} \tag{10}$$

where α is called the *learning coefficient*;

and $\frac{\partial E}{\partial w_{kj}^{[out]}} = \frac{\partial E}{\partial I_k^{[out]}} \bullet \frac{\partial I_k^{[out]}}{\partial w_{kj}^{[out]}}$ (11)

Again, on one hand, $\frac{\partial E}{\partial I_k^{[out]}} = e_k^{[out]}$

(12)

On the other hand, $\frac{\partial I_k^{[out]}}{\partial w_{kj}^{[out]}} = x_j^{[L]}$

(13)

Replacing terms from equations 12 and 13 in equation 11, gives:

$$\frac{\partial E}{\partial w_{kj}^{[out]}} = e_k^{[out]} \bullet x_j^{[L]} \qquad (14)$$

and from equation 10,

$$\Delta w_{kj}^{[out]} = \alpha \bullet e_k^{[out]} \bullet x_j^{[L]} \qquad (15)$$

In order to consider the weight correction of the immediately previous iteration (*t-1*), the delta weight equation for the iteration *t* is given as:

$$\left[\Delta w_{kj}^{[out]}\right]_{iteration(t)} = \left[\alpha \bullet e_k^{[out]} \bullet x_j^{[L]}\right]_{iteration(t)} + \beta\left[\Delta w_{kj}^{[out]}\right]_{iteration(t-1)} \qquad (16)$$

where β is called the *momentum term* (between 0 and 1).

The same equation is used for all connections reaching the output layer. Adjustment for connections reaching hidden layers uses the same method. However in those cases, the local error is calculated in a different way. For instance, for the layer L, the local error, $e_j^{[L]}$, of the jth neuron is expressed as follows,

$$e_j^{[L]} = f'(I_j^{[L]}) \bullet \sum_{p=1}^{n} \left(e_p^{[out]} \bullet w_{pj}^{[out]}\right) \qquad (17)$$

Since connection weights are corrected using equation 16, a new set of input-output data is shown to the network, and a new global error is computed. This same procedure is used for each set of examples presented, until global error is minimized and convergence is reached.

Appendix II

The Criteria To Evaluate An ANN Modeling Run Result

The design of the ANN model eveluation criteria is very problem specific and as mentioned in the paper, the best way to design this criteria is to combine the results from the preliminary runs and the designer's experience together.

In this following example, four aspects of the run are evaluated. The ability of an ANN model to converge is measured by the R square of the training set. The generalization capacity of an ANN model is measured by the R square of the testing set. The overall ability to both generalize and converge is measured by the R square of the whole data set. The reason to use R square is that in the calculation of R square, the big prediction error is severely penalized. This helps to choose a model with relatively well distributed prediction errors rather than a model with good average prediction error but has some wild prediction errors. Finally, there are some engineering criteria which must be satisfied for this specific problem. The mean absolute prediction error should be less than 0.5 unit.

Depending on what stage of the optimal solution searching is in, the weight assigned to each factor may be different. For example, in this case study, the factorial design is used to explore the factors in the early stage of problem. The focus is on the ability of the candidate model to converge on the training set. Thus, even though the ability of the candidate model to generalize is important, it is assigned less weight. In this example, the ability to converge is assigned a weight of 0.4. The ability to generalize is assigned a weight of 0.3. The overall ability to both generalize and converge is assigned a weight of 0.2 and whether the model satisfies the engineering requirement is assigned a weight of

0.1. The detailed evaluations of the goodness of each ability are presented in Tables 1 to 4. The boundary values in each table only serve as an example and they are not applicable to every ANN problem.

Table 1 Measuring the Goodness of Convergence

R Square of Training Set	Comment	Assigned Value fraction
<0.9	out of consideration	0
0.9 ~ 0.94	marginally acceptable	0.25
0.94 ~ 0.97	acceptable	0.5
0.97 ~ 0.98	good	0.75
>0.98	excellent	1

Table 2 Measuring the Goodness of Generalization

R Square of Testing Set	Comment	Assigned Value fraction
<0.1	out of consideration	0
0.1 ~ 0.35	marginally acceptable	0.25
0.35 ~ 0.5	acceptable	0.5
0.5 ~ 0.7	good	0.75
>0.78	excellent	1

Table 3 Measuring the Goodness of Overall Performance

R Square of Whole Set	Comment	Assigned Value fraction
<0.85	out of consideration	0
0.85 ~ 0.875	acceptable	0.5
0.875 ~ 0.9	good	0.75
>0.9	excellent	1

Table 4 Measuring the Goodness of Engineering Properties

Mean Absolute Error(unit)	Comment	Assigned Value fraction
>0.5	out of consideration	0
0.3 ~ 0.5	acceptable	0.5
0.1 ~ 0.3	good	0.75
<0.1	excellent	1

Appendix III

Residual Analysis Charts For the FeedForward Neural Network Controller

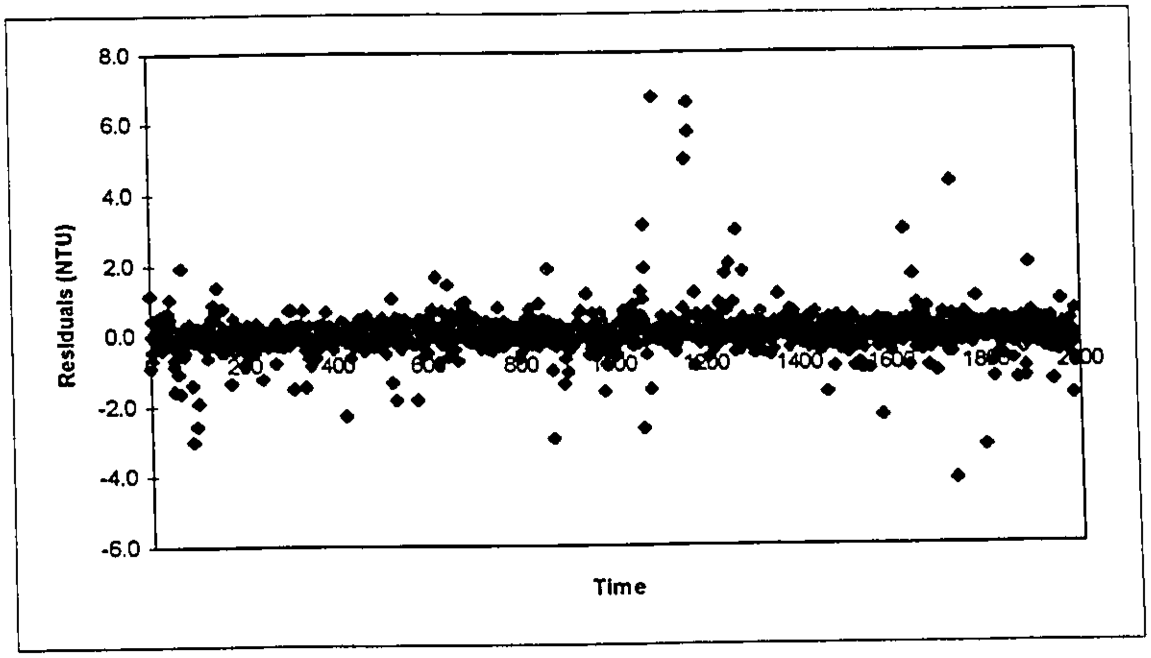


Figure 1 The Time Veruse Residuals For the Plant Reference Model

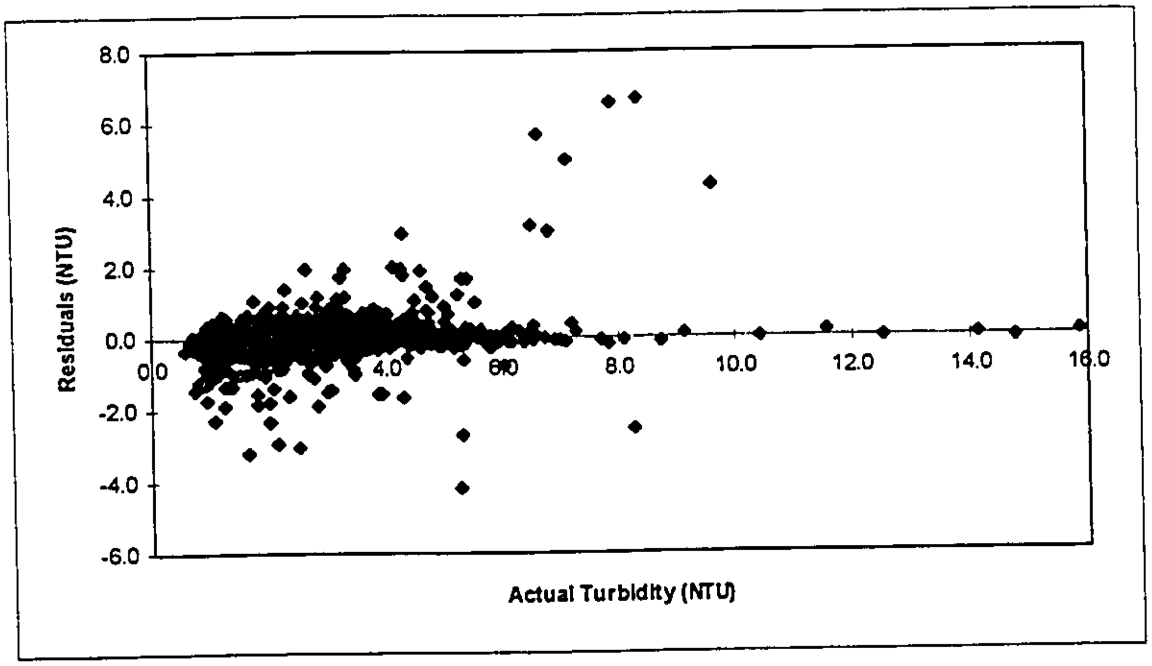


Figure 2 The Actual Turbidity Veruse Residuals For the Plant Reference Model

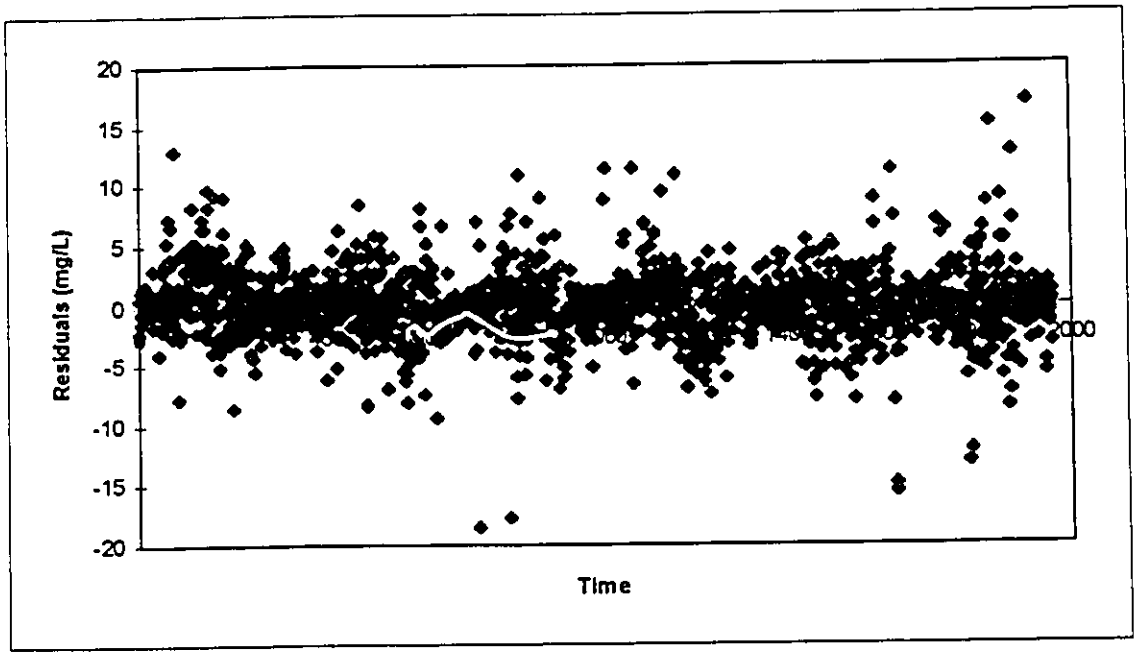


Figure 3 The Time Veruse Residuals For the Plant Inverse Model

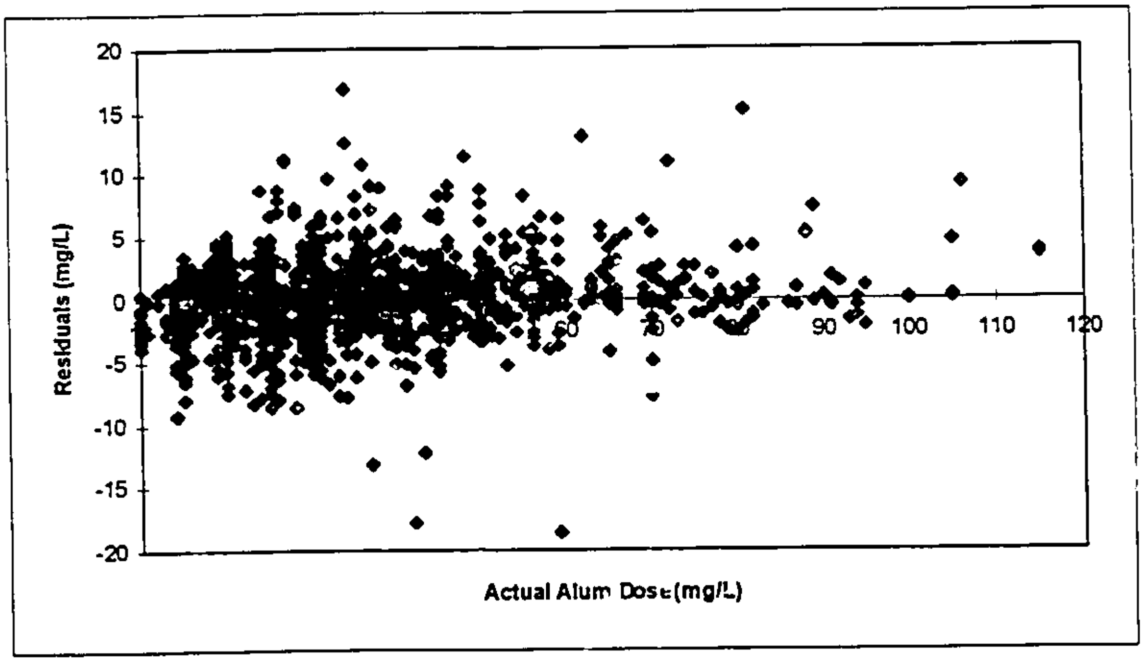


Figure 4 The Actual Alum Dose Veruse Residuals For the Plant Inverse Model