PARAMETRIC COST ESTIMATING OF HIGHWAY PROJECTS USING NEURAL NETWORKS

AMR SAMIR AYED

Parametric Cost Estimating of Highway Projects using Neural Networks

by

© Amr S. Ayed

A thesis submitted to the School of Graduate Studies
in the partial fulfillment of the requirements for
the degree of Master of Engineering

Faculty of Engineering & Applied Sciences

July, 1997

St. John's Newfoundland Canada

ABSTRACT

Contractors' experience on previous projects can undoubtedly be considered as an important asset that can help preventing mistakes and also increases the chances of success in similar future encounters. Construction cost data collected from past projects may be used to support cost estimating at different stages of a project's life cycle. At early stages of a project, parametric cost estimate is performed when detailed project information is lacking. The usable historical data at this level pertain to the characteristics of past projects (e.g., location, size, complexity), their construction environment (e.g., market, weather, year), in addition to the associated costs spent. The large number of these factors in addition to other external political, environmental, and technological risks, represent a complex problem in establishing accurate cost estimating models and have thus contributed to the inadequacy of traditional cost estimating techniques.

This thesis uses a non-traditional estimating tool, Neural Networks, to provide an effective cost-data management for highway projects and accordingly develops a realistic cost estimating model. Neural Networks are techniques based on advances in Artificial Intelligence branch of computer science. They have recently been used as a new information management tool in many construction applications to provide an effective cost estimating tool for highway construction cost data. In the present study, the characteristic factors that affect the cost of

highway construction have been identified and actual cases of highway and bridge projects constructed in Newfoundland during the past five years have been used as the source of cost data. The structure of a Neural Network template has been formed on a spreadsheet and three different techniques, Backpropagation training, Simplex Optimization and Genetic Algorithms, have been utilized to determine the optimum Neural Networks model. The resulting optimum model has been coded on Microsoft Excel in a user-friendly program to predict the outcomes for new cases. In addition, the proposed model provides a methodology to account for uncertainty in the user's assessment of project factors by measuring the sensitivity of the model to changes in cost-related parameters. It also enables the user to re-optimize the model on new historical encounters and accordingly adapt the model to new environments. The capabilities and limitations of the developed model have been discussed along with the expected future research in this domain.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 General

Construction estimating is one of the most crucial functions in project management. Cost estimating needs to be done in different manner at different stages of a project. At the early stage where project budgets are to be decided, detailed information is not available and parametric cost estimating techniques are most applicable. With the ever-increasing budget restrictions, accurate estimating becomes crucial to the setting of appropriate project budgets. Despite its great importance, the estimating task is neither simple nor straightforward due to the lack of information at this early project stage and to the existence of many external factors that affect a project including political, site, environmental, and

technological risks. Once budgets have been approved and project scope becomes well defined, detailed cost estimating (rather than parametric) methods become necessary for construction bidding and project control. On the one hand, accurate bid proposals maintain contractor's success and establish his potential profits while inaccurate estimates could result in either significant monetary losses, if the estimates are too low, or no jobs at all if they are too high. On the other hand, estimating realistic costs and schedule baselines is certainly important for efficient job control.

Most parametric costs estimating approaches described in the literature use statistical analysis techniques (Garza and Rouhana, 1995; and Creese and Li, 1995), range from simple graphical curve fitting to multiple correlation analysis. The objective is to use some historical cost data and try to find a functional relationship between changes in cost and the factor(s) upon which the cost depends. Therefore, such approaches produce a mathematical model that describes the cost of a project as a function of one or more independent variables. A major drawback of statistical techniques is that a general mathematical form of the cost estimating relationship has to be defined before any analysis can be applied to best fit the historical cost data. This is extremely difficult in estimating the costs of construction projects since the number of independent variables in the project is huge and no clear relationship exists between the cost and each individual factor. Statistical tools, as such, can only

work for simple systems where cost relationship is known before hand and may not be suited for construction projects. Accordingly, the use of statistical techniques to predict accurate future cost for a construction project is not appropriate as reported in several efforts in the literature (Creese and Li, 1995, and Garza and Rouhana, 1995). This explains the need for a more suitable technique for parametric cost estimation.

Due to the inadequacy of traditional estimating techniques, a new breed of tools has evolved based on the Artificial Intelligence branch of computer science (e.g., Neural Networks) and have recently been used as a new estimating tool in construction (Hegazy et al., 1994b). Some construction examples include the prediction of productivity levels achievable under particular job-site conditions, the assignment of a percentage markup before submitting the bid price, and day-to-day decisions regarding the allocation of resources, minimizing idle times, and solving disputes (Hegazy et al., 1994c). The benefits of Neural Networks stem from their ability to learn from a set of examples (representing historical encounters) to detect by themselves the hidden relationships that link the causal parameters to the correct decisions or outcomes. Moreover, Neural Networks have been demonstrated to have excellent performance in modeling non-linear multi-parameter relationships that involve judgment and experience, even in situations where the data are partially missing or incorrect (Hegazy et al., 1994c). Neural networks, as such, have a great potential for effectively managing

historical cost data to provide adequate budgeting and cost estimating models. Recently, many researchers have started to investigate the potential use of Neural Networks as a tool for cost modeling (Creese and Li, 1995, Garza and Rouhana, 1995; and Mckim, 1993a). The models introduced in these studies demonstrate the applicability of Neural Networks techniques and present several applications for risk assessment, parametric estimating, and competitive bidding.

1.2 Scope and Objectives

This research deals with the problem of estimating accurate costs at early stage of a project (parametric cost estimating). The research has 6 main objectives:

1. Identify the factors that affect the cost of a project including subjective and risk-related factors.
2. Use state-of-the-art techniques, such as Genetic Algorithms and feed-forward Backpropagation, for optimization and training of the Neural Networks to determine the optimum Neural Network model that accommodates the identified parameters.
3. Promote the application of the Neural Networks approach in the construction domain by presenting it in a simple spreadsheet format that is customary to construction practitioners.
4. Compare the results of the Neural Network model with other traditional estimating methods described in the literature.

5. Develop a comprehensive tool for parametric cost estimation using the resulting Neural Network model.
6. Develop methodologies for examining the sensitivity of the developed model to changes in cost-related parameters and for adapting the model to new project environments.

1.3 Research Methodology:

The approach used to arrive at the study objectives can be summarized in the following steps:

1. Design the parametric cost estimating system in a modular architecture with several components (Fig. 1.1).
2. Review the theory and current developments in cost estimation and Neural Networks that relate to the system modules. This helps identify, for each module, the most appropriate procedure applicable to the system.
3. Identify the qualitative factors which need to be considered in cost estimation.
4. Study the applicability of Neural Networks to the problem at hand, accordingly, develop the cost estimation system.

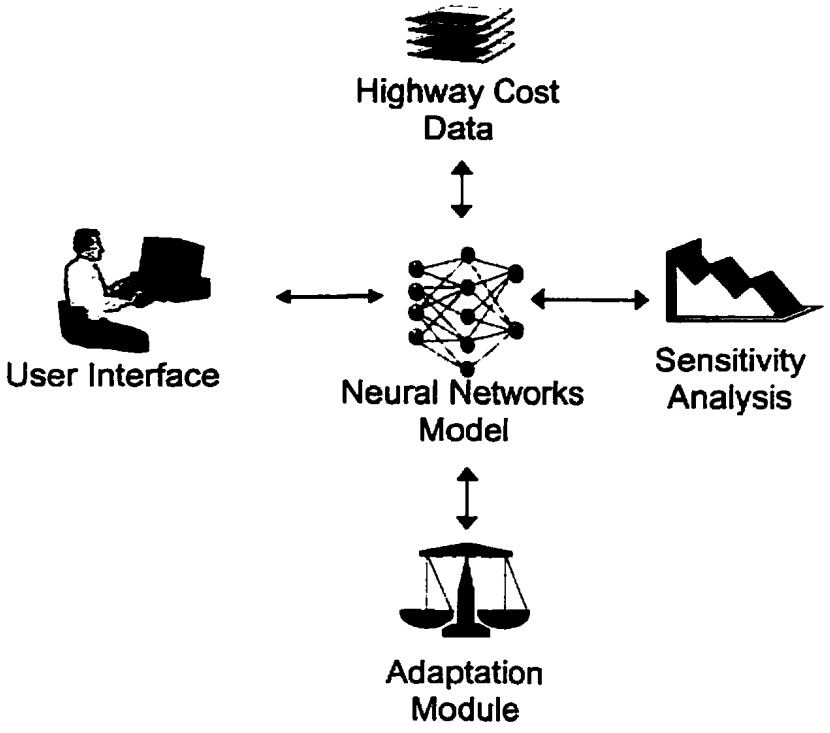


Figure 1.1: Components of a Parametric Cost Estimating System.

1.4 Thesis Organization

Chapter 2 presents a literature review of traditional and state-of-the-art efforts that are related to the parametric cost estimating. In this chapter, Neural Networks are introduced as a new tool for prediction. Neural Networks components, characteristics and training problems are discussed along with Simplex Optimization and Genetic Algorithms as tools for selecting the optimum Neural Network model.

Chapter 3 describes the data-acquisition process of historical highway cost data necessary for the proposed model. The characteristic factors that relate to cost estimating of highway projects are identified and outlined. Details regarding the design, implementation, optimization, and testing of the Neural Network model are described in this chapter.

Chapter 4 describes the development of a complete system for parametric cost estimate of highway projects. The system is coded in a user-friendly software that facilitates storage of previous project encounters, input of project data, prediction of project cost, assessment of project sensitivity and adaptation of the model to the user's environment.

Chapter 5 is the thesis conclusion and description of extensions to current research.

CHAPTER 2

Literature Review

2.1 Introduction

Cost estimation is probably the most crucial function to the success of construction organizations. At all project phases, different types of cost estimating from preliminary to detailed are conducted for different purposes. Figure 2.1 illustrates different types of cost estimate along with their appropriate project phases, associated difficulty levels, and the expected cost-estimate accuracy. These types are briefly described as follows:

1. **Preliminary cost estimate:** This type of estimate is conducted at the early stage where project budgets are to be decided, available information is limited and parametric cost estimating techniques are most applicable. It is made

without working drawings or detailed specifications. The estimator may have to make such an estimate from rough design sketches, without dimensions or details and from an outline specification and schedule of the owner's space requirements. The preliminary cost estimate can serve several purposes, including: 1) it supplements or serves as the owner's feasibility estimate; 2) it aids the Architect/Engineer in designing to a specific budget; and 3) it assists in the establishment of the owner's funding. The most common technique at this stage is the parametric cost estimating approach.

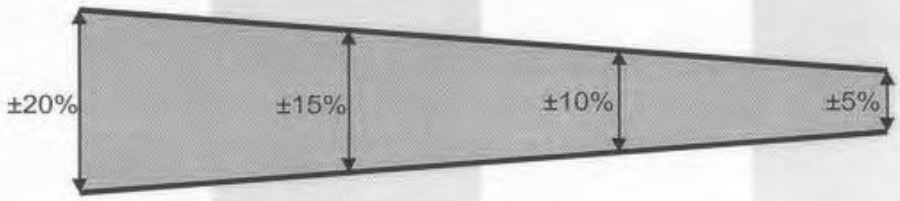
Estimate Type	**Preliminary**	**Elemental**	**Unit price**	**Detailed**
Project Phase	Concept	Preliminary Design	Detailed Design	Constructi
Available Information	Limited →			Detailed
Difficulty Level	High →			Low
Expected Accuracy				

Figure 2.1: Estimate Types Throughout a Project's Life-Cycle.

2. **Elemental (functional analysis) estimate:** This type of estimate is conducted at an intermediate project stage by dividing a project into

convenient functional elements (excavation, foundation work, concreting,.....etc.) and individually pricing each of these elements. Element breakdown can serve several purposes: 1) to reveal the distribution of costs of the constituent elements of the project; 2) to relate the cost of a constituent element to its importance as a part of the whole project; 3) to compare the cost of the same element in different projects; and 4) to enable a determination of how costs could have been all allocated to obtain a better project.

3. **Unit price estimate:** This type of estimate is conducted at a later project stage at which a detailed quantity take-off is possible. The key to quantity take-off is a well-established and identifiable code for work items. The most common type is the 16-division master format code for building projects. For each sub-item, the direct cost (labour, materials and equipment) can be estimated. The purpose of unit price estimate at this stage of the project is to replace previous less accurate estimates in order to keep the project within budget.

4. **Detailed cost estimate:** This type of estimate is conducted at a more detailed and refined level than unit price estimate. This could be done by analyzing each sub-item more accurately considering all factors that affect the sub-item. Usually a separate sheet is used, such as those provided by R.S. Means (R.S.

Means, 1997) for example, to perform this analysis. It should be noted that job planning has a great influence on the detailed estimate. For example, "cost per cubic yard" for excavation can be calculated more accurately than a unit price estimate, by considering site condition, weather condition, dewatering, available excavators, drivers and so on. As such, a detailed cost estimate is feasible only after performing a detailed planning and scheduling study for the project.

At all estimating levels, the difficulties involved often result in many mistakes and errors in judgment. However, as shown in Figure 2.1, preliminary (parametric) estimating exhibits the lowest accuracy level due to the lack of project information and the high level of uncertainty at this early stage of a project. In addition, the political, site, environmental, and technological risks that are extremely difficult to predict. A brief description of parametric cost estimating techniques is described in the following subsection along with their advantages and limitations.

2.2 Parametric Cost Estimating

2.2.1 Traditional Techniques

General forecasting techniques are basically quantitative approaches that have been in use for the past half century. They can be either deterministic or stochastic (Savin and Kumar, 1993). Examples of deterministic methods are regression methods (linear regression and multiple regression), econometric models, moving average methods and exponential smoothing methods. Examples of stochastic methods are the maximum likelihood method, Box-Jankins models, and probability weighted moment (L-moment).

In construction, deterministic cost estimation models (referred to as parametric models) have traditionally been in wide use by many researchers due to their simple formulations. In these techniques, historical data are used to develop cost relationships based solely on statistical analysis. They have been used to estimate one characteristic of a system, usually its cost, from other performance characteristics of the system. The term "Parametric Estimating", however, may mean different things to different researchers (Black, 1982). For instance, it may mean the determination of the life cycle cost of a system from a mathematical model containing a number of parameters and based on case histories of similar projects. On the other hand, it could mean the cost estimation of any system, made up of aggregated components, by means of mathematical models containing parameters.

In parametric analysis, the main parametric equation has to be defined with its associated independent variables before any analysis can be performed. The variables in the parametric equation could be identified by highlighting those characteristics of the system under study that are most directly related to its cost. Then, a mathematical relationship for correlating data could be used to express this relationship. The most popular mathematical models are linearized forms of the arithmetic, logarithmic and semi-logarithmic equations (Black, 1982). In the parametric cost estimating technique, there is one dependent variable which is the cost and two or more independent variables like size, location, capacity, time....etc. The equations generally take one of the following three forms:

1) Linear relationships:

$$Cost=a+bX_1+cX_2+.....$$

2) Logarithmic relationships:

$$Log(Cost)=a+b\ Log\ X_1+\ c\ log\ X_2+.....$$

3) Exponential relationships:

$$Cost=a+\ bX_1^{\ c}+dX_2^{\ e}\ +.....$$

Where a, b, c, d and e are constant and X_1, X_2, X_3, X_4,....., X_n are the performance characteristics of a system.

The best criterion for choosing a form of the cost estimating relationship is a good understanding of how costs vary with changes in the independent parameters. This is a difficult task that is based on the experience of the estimators involved and, as such, has been performed mainly on a trial and error basis.

A major disadvantage of the traditional techniques for parametric estimating is that the mathematical form has to be defined before any analysis can be performed to determine the actual cost function that best fits the historical data (Creese and Li, 1995). Also, modeling the cost of a system as a function of a member of independent variables is not an easy task. This is due to the large number of variables present in the system under evaluation and the numerous interactions among them. Another drawback is the use of a single cost estimation relationship to all cost variables involved which often have different mathematical correlation with the cost of that system. These problems may explain the low accuracy and limited use of parametric estimating techniques based on mathematical analysis in the construction industry and the need for a more accurate technique to solve the cost estimation problem.

Recently, several research efforts have developed for parametric cost estimation models based on traditional estimating techniques. Some of these efforts assist in generating cost indexes or production rates in addition to traditional parametric

analysis. Examples of these efforts that are of interest to the present study include: (Al-Bani 1994, Lopez 1993, Pantzeter 1993, Akeel 1989, Ellis 1989; and Uhlik 1984). Al-Bani (1994) developed a concrete cost estimate model for small residential buildings. The research studied the inter-relationships between the different physical elements of a concrete structure such as footings and columns using mathematical expressions and formulas. In a different approach, Lopez (1993) conducted a study on forecasting construction costs in hyper-inflated economies. In this study, Mexican economic indicators were compared with those from the United States. As a result a new Mexican cost index was developed using Box and Jenkins models. Multiple regression analysis has also been used by Pantzeter (1993) to develop a methodology for modeling the cost and duration of concrete highway bridges. In this research, different projects were divided into five work categories and the cost of each category was modeled by applying statistical techniques. A similar effort was also conducted by Akeel (1989) in developing a database tool for statistically based construction cost estimating. This research utilized the multiple regression analysis to develop the cost estimating relationship. Another effort using statistical analysis was performed by Ellis (1989) to produce a predictive model for construction production rates and examine their variability. To optimize cost estimation in uncertain situations, Uhlik (1984) developed a combined stochastic and deterministic model for highway projects. His developments accounted for the uncertainties associated with quantity of rock in cut areas to estimate fleet

production and determine the optimum distribution of material in cut and fill areas. These research efforts, while constrained by the limitations of traditional tools, provide insights into the elements that need to be considered in the development of effective parametric models.

With the advantage of computers in the 1980's, commercial software systems for cost estimation have proliferated. Several surveys have been conducted to identify the commercial cost estimation software (Fayek et al., 1994; and Arditi and Riad, 1988). The first survey has listed recent systems and compared their capabilities with respect to estimating features, tendering features, reporting features, type of project and other factors. Table 2.1 illustrates some of these capabilities for each estimating system.

Table 2.1: Current Available Cost Estimation Systems (Fayek et al., 1994).

Estimating Systems	Types of Projects			Methods of Estimating			Methods of Reporting		
	Building	Civil	Other	Detailed	Unit Rate	Other	Screen	Printer	File
ACE	Y	Y	Y	Y	Y	N	Y	Y	Y
Computer Gold	Y	Y	Y	Y	Y	Y	Y	Y	Y
Estimator II	Y	Y	Y	Y	Y	N	Y	Y	Y
Everest	Y	Y	Y	Y	Y	Y	Y	Y	Y
Expert Estimation	N	Y	Y	Y	N	N	Y	Y	Y
G2 Estimator	Y	Y	Y	Y	Y	Y	Y	Y	Y
ISCE	Y	Y	Y	Y	Y	N	Y	Y	Y
L.B.E.S.T.	Y	Y	Y	Y	Y	N	Y	Y	Y
Leader	Y	Y	Y	Y	Y	N	Y	Y	Y
Lodex Build	Y	Y	Y	N	Y	Y	Y	Y	N
MAPAS	Y	Y	Y	N	Y	N	Y	Y	N
Paydirt	N	Y	Y	Y	N	N	Y	Y	Y
Probid 8088	Y	Y	Y	Y	Y	N	Y	Y	Y
SUCCESS	Y	Y	Y	Y	Y	Y	Y	Y	Y
Timberline	Y	Y	Y	Y	Y	N	Y	Y	Y
TRACE	N	Y	Y	Y	N	N	Some	Y	Y
Wessex	Y	Y	Y	Y	Y	Y	Y	Y	Y

Note: Y= yes. N= no

Some of these systems, such as *Timberline*, are fully integrated systems that allow estimating module to exchange data with other modules, such as CAD, scheduling and project control as shown in Figure 2.2.

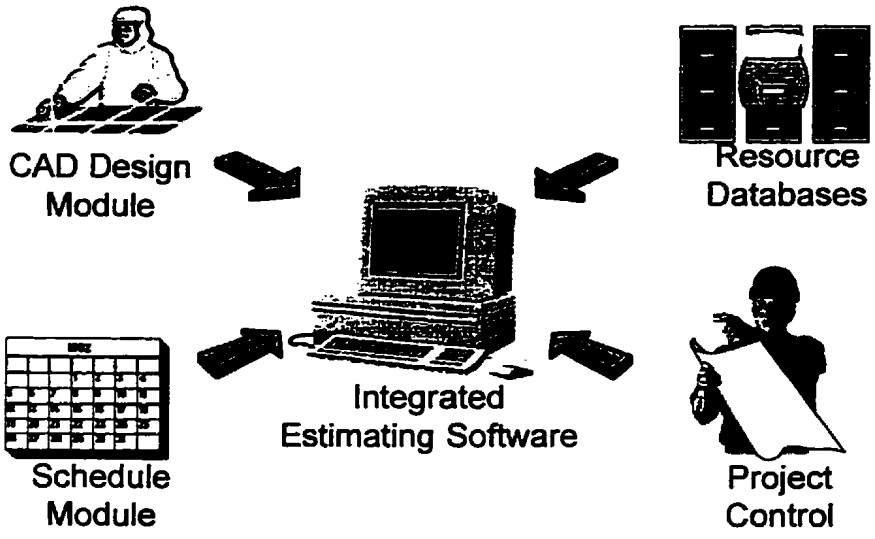


Figure 2.2: A Complete Integrated System for Cost Estimation.

The majority of listed systems with the exception of (*Success*, 1995) are used for cost control and detailed cost estimation. *Success* system and another commercial software, (*Design 4/Cost*, 1995), are used at the early stage of a project when data is not sufficient to perform a detailed cost estimate. On the one hand, *Success* helps in providing a parametric as well as a detailed estimate of a project based on independent variables associated with each item of work. The cost generated, however, is only applicable to the United States and is difficult to be adjusted to other countries. Also the software still needs user identification of how to correlate parameters affecting an item to its cost which is a difficult task. On the other hand, *Design 4/Cost* uses a historical database of previous projects executed in The United States. The software uses algorithms

and mathematical relationships for regional adjustment and cost escalation inside the United States for up to the year 2000. All estimates, however, are generated using the square feet building area of the project as the main parameter. The accuracy of the estimates, however, degrades if the new building area is not within a ±20% range, relative to the historical database.

Several attempts were made in the literature to introduce parametric models for cost estimation based on various computerized techniques. Yau (1992) implemented an object-oriented methodology to develop a model for integrated scheduling and a cost estimate at the preliminary design stage of mid-rise building projects. Another study by Lee (1992) developed a cost estimating model that can provide detailed cost information on design alternatives at all stages of design for reinforced concrete buildings. A recent effort by Hollman (1994) introduced a parametric cost estimating system for buildings developed by the capital estimating department at Kodak Park. The system can give a rough cost estimate with a ±5% accuracy for the total cost of buildings. The advantage of this system over other existing square feet systems is the elimination of the need for detail or assembly level cost data. Also, The U.S. Army Corps of Engineering (COE) has developed a software program that can be used for preparing a quantitative estimate from parametric models. The program is known as Control Estimate Generator. This program was used by Melin (1994) to demonstrate various aspects of parametric estimating and

standard estimating software. The parametric models are designed to generate a detailed estimate from historical or past projects. The program uses a parametric approach to adjust the original quantities by parameters to generate new quantities for use in the proposed estimate.

As opposed to the deterministic approaches described earlier, probabilistic techniques (e.g., Monte Carlo simulation) have recently been used for estimating. The benefit of such techniques is their ability to account for risks and model the cost components with high variability. Smith (1989) implemented a simulation model that uses a cost breakdown structure to assess the uncertainty and risk involved in project cost estimate. The simulation considers statistical distribution for each cost variable under estimation and produces a value for the total cost estimate and its probability of occurrence. Despite the probabilistic advantage of the Monte Carlo simulation, it has two major limitations. First, statistical distributions for various cost components need to be established before hand. Second, if the cost variables are not independent, their correlation should be accounted for.

It is clear from the previous discussion that computers are very fast and accurate when carrying out the extensive computations needed for cost estimating. They are, however, traditionally poor at the intuitive aspects that require the integration of experience and making decisions that can not be clearly defined in

mathematical form. Therefore, they fail to adequately model the essential part of estimating that is based on experience and analogy with previous situations. This has motivated the application of non-traditional estimating techniques based on artificial intelligence (AI) in this domain. Among the several AI areas, the Neural Networks technique presents itself as a new approach of computation and decision making that may potentially resolve some of the major drawbacks of traditional estimating techniques. It holds a great promise for rendering the parametric method of cost estimating a reliable and reasonably accurate way to prepare cost estimates.

2.2.2 Artificial Intelligence-Based Techniques

Artificial Intelligence (AI) has been a rapidly growing field of computer science that has direct applications in the construction industry. The term (AI) is applied to those fields of computer science attempt to simulate human intelligence. Expert Systems and Neural Networks are among current Artificial Intelligence research areas. On the one hand, Expert Systems attempt to model the intelligent reasoning and problem-solving capabilities of the human brain. They use rules in the form of (IF.... THEN) to explain how they arrived at reliable decisions (Moselhi et al., 1990). Mohan (1990) listed 37 expert-system applications in the field of construction engineering and management. Of these, two are specialized in cost estimating at early stage of the project as shown in Table 2.2. Also Salamh (1989) implemented a knowledge-based expert system

for the conceptual design and cost estimating. The research focused on determining the essential rules and heuristics used by experts in making conceptual cost estimates for the construction of buildings. Neural Networks, on the other hand, take a different approach to AI than traditional techniques such as Expert Systems. The technique attempts to replicate the mechanism by which the human brain manipulates data and reaches decisions (Mckim, 1993b).

Table 2.2: Expert-Systems Used for Preliminary Cost Estimate.

Expert -system	System inputs	System outputs
-*HI-Cost*: cost estimating from preliminary design	Preliminary design alternatives	Cost estimate based on preliminary design
-Predicting time and cost of construction during initial design	Activity details and resources	Time and cost of activities

2.3 Neural Networks

Neural Networks are particularly effective for complex estimating problems where the relationship between the variables can not be expressed by a simple mathematical relationship. They are computer programs simulating the biological structure of the human brain which consists of tens of thousands of highly interconnected computing units called neurons. An Artificial Neural Network can

be constructed to simulate the action of a human expert in a complicated decision situation.

2.3.1 Structure and Functionality

A Neural Network is constructed by arranging several processing units in a number of layers (Fig. 2.3). The output of a single layer provides the input to the subsequent layer and the strength of the output is determined by the connection weights between the processing units of two adjacent layers. One of the most important characteristics of Neural Networks is their ability to learn from a set of examples to detect by themselves the relationships that link inputs to outputs. During training, both the inputs (representing problem parameters) and outputs (representing the solutions) are presented to the network normally for thousands of cycles. At the end of each cycle, or iteration, the network evaluates the error between the desired output and actual output, then uses this error to modify the connection weights according to the training algorithms used. One of the most common training algorithms is Backpropagation (Rumelhart et al., 1986). For a certain number of learning cycles the weights are continuously adjusted until the deviations from the desired outputs are minimized. After training, the network can be used to predict the solution for a new case not used in training. For background information regarding Neural Network formulations, mathematics and potential applications in constructions, the reader is referred to several

publications (Hegazy et al., 1994a; and Moselhi et al., 1990) and will be dealt with in chapter 3.

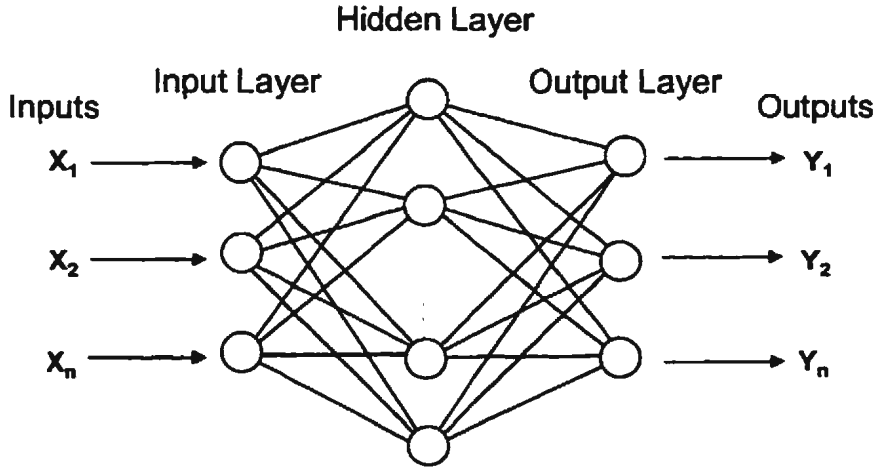


Figure 2.3: General Structural of a Neural Network.

2.3.2 Applications in Cost Estimation

Neural Networks have been applied by several researchers to develop parametric cost models. A summary of these efforts is presented in table 2.3.

Table 2.3: Previous Applications in Parametric Cost Estimating using Neural Networks Technique.

Publisher	Description	NNs Inputs	NNs Outputs	Comparison Module(s)	Results	Comments
Graza and Rouhana (1995)	Estimating the actual cost of Carbon steel pipes.	- Pipe diameter. -Number of elbows. -Flange rating.	Cost per 100 ft of pipe.	Multiple regression(linear form and non-linear form).	The mean square error using NNs was less than mean square error using multiple regression.	Back-propagation which is used in this study could not be suitable for other applications.
Creese and Li (1995)	Estimating the cost of timber bridges.	-Volume of the web. -Volume of the decks. -Weight of the steel used in the bridge.	Actual cost of the bridge.	Linear regression analysis.	In all possible input alternatives, the *r*-square values using NNs were greater than those using regression analysis.	A comparison was made between all possible alternatives for input variables and it was found that the models with 3 input variables are giving the least errors.
Williams (1994)	Estimating the changes in cost indexes.	-The prime lending rate. -# of housing starts for the month. -The month of the year and other variables.	Changes in cost index.	-Exponential smoothing. -Linear regression.	NNs produced a poor prediction for changes in cost index.	An additional effort should be done to identify variables affecting the construction cost indexes better than those used in the past.
Mckim (1993a)	Estimating the actual cost of pumps.	-Pump flow. -Pump head.	Cost of the pump.	-Exponent scale method. -Best- fit exponent method. -Best-fit equation method.	The *r*-square value from NNs was greater than other estimating methods.	The accuracy of the NNs prediction was affected by the quality of the available historical data.
Mckim (1993b)	Predicting the % of change of the final costs from estimated costs.	-Contractor. -Architect. -Location. -Size.	Percentage of change in the cost.	Mean overrun method.	The variance of NNs results was less than the variance of mean overrun results.	NNs is producing good results only in case of rough cost estimation and it is not recommended for detailed estimate.

Graza and Rouhana (1995) compared the results of Neural Networks with those of regression models for predicting the material cost of carbon steel pipes. Ten sets of cost data were used to train the Neural Networks and six sets were used in testing. The costs for these 6 sets were estimated using three methods (linear regression, non-linear regression and Neural Networks). The mean square error was calculated for each model. It was found that Neural Networks produced the lowest mean square error compared with the other two models. The accuracy level was between 66.8% and 77.96%. This study proves that the Neural Networks approach could be used to resolve some of the major drawbacks of the regression-based parametric estimation.

A similar study was conducted by Creese and Li (1995). In this study, three models were developed to estimate the cost of timber bridges. A set of actual cases of 12 timber bridges was collected from West Virginia department of highways. Three variables were used as the main factors affecting the total cost: cost of the web, cost of the deck, and weight of the steel used. The study experimented with three Neural Network models to identify the optimum one. The three models consider either one input variable, two input variables, or three input variables, respectively, to predict the total cost. It was observed that the overall training accuracy increased as more input variables were used and as such, the model with three input variables was the best. The initial training of the Neural Networks used all available bridges' data and 1,500 training cycles were

used. The standard linear regression approach was used to predict the actual cost for the three models using the same variables and *r*-square values (coefficient of determination) were calculated to evaluate the three models. The study also compared the Neural Networks approach to linear regression analysis and concluded that the estimation accuracy of Neural Networks approach is better than linear regression analysis in timber bridges' types. Another important conclusion of the study is that the cost prediction ability of Neural Networks to improves when more independent variables are introduced in training.

Williams (1994) developed two Backpropagation Neural Network models to predict change in ENR construction cost index for one month and six months ahead. The models used many different factors such as the prime lending rate and the month of the year) as input variables. A training set of 215 cases and a test set of 63 cases were used for the one-month model while a training set of 207 cases and test set of 66 cases were used for the six-month model. Two other simple models were used to predict the change in construction cost indexes, an exponential smoothing model and a linear regression model. When comparing the predicted results from the Neural Network models with those predicted from the other two models, it was found that the Neural Networks produced close but poorer predictions of changes in construction cost indexes. These poor predictions were attributed to the extrapolation, rather than estimation, nature of the application. Another study was conducted by Mckim

(1993a). In this study, a Neural Network model was developed to predict the cost of pumps. The input variables to the Neural Network model were the pump flow and head. A set of 23 pumps with their associated flow, head and actual prices was used as training cases. The Neural Network model was trained for 50,000 cycles. The results were compared to other industrial methods commonly used in practice for predicting pumps' cost. These methods were: 1) 0.6 exponent scaling method, 2) Best-fit exponent scaling method, and 3) Empirical best-fit equation method. The standard deviation error and the coefficient of determination were calculated for each method. Similar to the previous study by Creece, it was concluded that the Neural Networks method provides a more accurate estimate than the other regression methods. A similar conclusion has also been arrived at by the same author (Mckim, 1993b) in another study.

2.3.3 Problems with Neural Networks

Despite the good performance of Neural Networks in the previous studies, the process of developing and implementing Neural Networks to parametric cost estimation has a number of problems associated with it. First, designing the network architecture and setting its parameters is not a straight-forward approach; it actually requires some trial and error process. A large amount of time must be spent determining the best network architecture and the network parameters that best fit the application under consideration. Second, the learning algorithms, such as Backpropagation, require optimization of the network training

in order to achieve adequate generalization. Also, there is no explicit set of rules to determine whether a given learning algorithm is suitable for a particular application or not. In addition, the little user control over training and the final status of network weights have contributed to their black box perception. Also, it should be noted that Neural Networks do not do perform well with applications where precise numerical computations are required, like detailed estimating and cost control.

2.3.4 Genetic Algorithms

With Backpropagation being the most applicable training algorithms in cost estimation domain, achieving good generalization performance becomes an essential issue. One of the most promising tools that can be used to optimize generalization capabilities of Backpropagation is the use of Genetic Algorithms (GAs). In general, GAs is optimization procedures inspired by biological systems' improved fitness through evolution (Hegazy et al., 1994b). GAs seeks to solve optimization problems using the method of evolution, specifically survival of the fittest. The theory of GAs is that a population of a certain species will, after many generations, adapt to live better in its environment. For example, if the species is an animal that lives mainly in a swampy area, it may eventually evolve with webbed feet. GAs solves optimization problem in the same fashion. It will create a population of possible solutions to the problem. It solves the problem by allowing the less fit individuals in the population to die and selectively breeding

the fittest individuals (the ones that solve the problem best) of the population. This process is called selection, as in selection of the fittest. GAs takes two fit individuals and mates them (a process called crossover). The offspring of the mated pair will receive some of the characteristics of the mother, and some of the father. Offspring often have some slight abnormalities called mutations. After GAs mates fit individuals and mutates some, the population undergoes a generation change. The population will then consist of offsprings plus a few of the old individuals which was allowed to survive to the next generation because they are the most fit in the population and it is wanted to keep them breeding. The fittest individuals are called elite individuals. After dozen or even hundreds of generations, a population eventually emerges wherein the individuals will solve the problem very well. In fact, the fittest elite individual will be an optimum or close to optimum solution.

2.4 Conclusion

Cost estimating is the act of appraising and evaluating the cost of a project before implementing it. There are many existing techniques for parametric cost estimation. These techniques could be classified into two categories: 1) Traditional techniques, and 2) Artificial Intelligence-based techniques. In this chapter, the two categories have been reviewed with emphasis on parametric cost estimation and the feasibility of applying the technique in this domain. Several conclusions are derived based on this literature survey:

1. Backpropagation is the most widely used Neural Network training algorithms for parametric cost estimation.
2. Neural Networks are proved to outperform traditional techniques including regression analysis in this domain.
3. Special attention has to be focused on identifying the problem attributes and accordingly the relevant independent factors.
4. Special attention has to be focused on improving Neural Networks generalization either by using appropriate testing and verification or the use of optimization technique such as GAs.
5. Neural Networks have been perceived as black box and this has contributed to their slow applicability in construction.

CHAPTER 3

Cost Data Modeling using Neural Networks

3.1 Introduction

The case study being dealt with in this study is the developing of a parametric cost estimation system for highway projects. A structured methodology for Neural Networks development (Hegazy et al., 1994b) has been used to model the problem at hand. The methodology incorporates three main phases: 1) Conceptual analysis, 2) Neural Network design; and 3) Neural Network modeling and implementation as shown in Fig. 3.1.

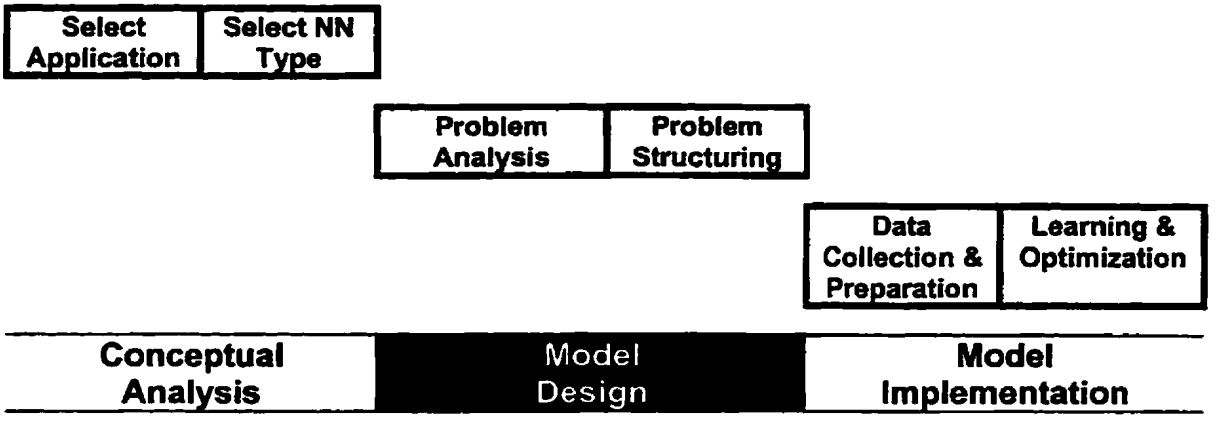


Figure 3.1: Neural Network Development Methodology.

3.2 Conceptual Analysis

At the conceptual analysis stage, a Neural Network paradigm has to be selected as a suitable environment for developing the application. It can be done based on a comparison of the application requirements against Neural Network paradigm capabilities. Based on the literature review of Chapter 2, the Neural Network type deemed suitable for cost estimation has been identified as feed-forward pattern-recognition type (Backpropagation) to suit the desired interpolative and predictive performance of the model. Other suitable types include the General Regression Neural Network (GRNN) which uses a multidimensional regression algorithm that trains fast on sparse data. GRNN applications are sometimes able to outperform Backpropagation (*NeuroShell2* Reference Manual, 1995). For the present study, a comparison has been conducted between the results of these two common

Neural Network architectures (Backpropagation and GRNN) and those of multiple regression analysis.

3.3 Neural Network Design

Model design consists of two main tasks: 1) Problem analysis; and 2) Problem structuring. Problem analysis, on the one hand, is the identification of the independent (non-correlated) factor(s) that fully describes the problem and that are expected to be easily obtainable for training the network at a later stage. For the present study, ten major factors describing a highway project and affecting its cost have been identified. These factors include descriptors of project size, year of construction, and project location so as to have a generic estimating model accounting for time, capacity and other uncertainty-related factors. Problem structuring, on the other hand, entails the arrangement and representation of the descriptive factors and their associated results (cost of tasks) in the form of inputs and outputs, as required by Neural Network modeling. With the ten inputs readily identified, the outputs describing the cost of a highway project can be modeled in different ways and thus two Neural Networks were constructed as shown in Fig. 3.2 (Hegazy and Ayed, 1997). The inputs and outputs in each network are as follows:

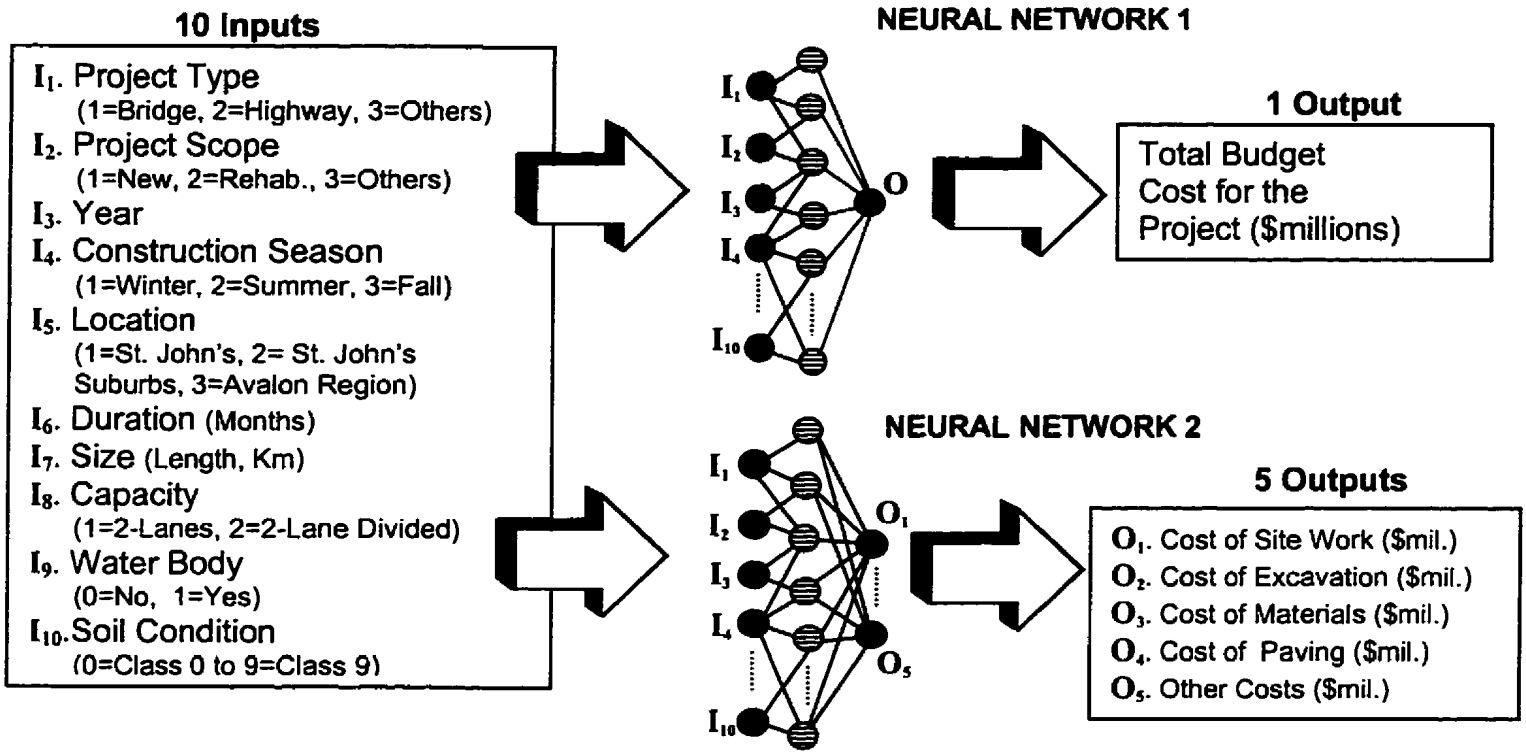


Figure 3.2: Description of Neural Network Inputs and Outputs.

Neural Network 1 (NN1): 10 inputs (project descriptors); and 1 output (total project cost). Since this network produces the total project cost as a single output, it is usable at an early project stage for budgetary estimates before detailed design data is available.

Neural Network 2 (NN2): 10 inputs (project descriptors); and 5 major outputs (cost items). Project cost in this case is modeled as five values corresponding to the five major work items in highway projects.

Having the inputs and outputs of the two Neural Networks defined, relevant data can then be documented and collected for each project. Both the input and output data are utilized during model development. Afterwards, only inputs are needed for the actual use of the trained model to estimate the cost of new projects.

3.4 Neural Network Implementation: Data Collection and Preparation

Once there is a clear idea about some feasible structures and the information needed to be elicited, the implementation phase starts with knowledge acquisition and data preparation. Collecting cost data and information related to cost estimating problems is a difficult task because such information is the property of each construction firm. Construction firms usually do not like to share their approaches or their experience and cost data with other competing

construction firms. Moreover, most firms believe that such information usually makes a difference in being more competitive in the market. There is a need, therefore, for a methodology for collecting the experience-based practices of contractors and effectively utilizing it to develop accurate parametric cost estimating models that adequately account for a project environment and establish accurate cost estimates accordingly.

To collect cost data for Neural Network training, personal contacts were made with contractors, engineering consultants and government offices across Canada (such as Institute for Research in Construction, NRC). As a result of these efforts, the Department of Public Works, Services and Transportation, St. John's, Newfoundland, Canada indicated interest in providing such information. Accordingly, the data used in this study was collected from eighteen bids submitted to their office in the past five years. All eighteen projects were unit price jobs with the itemized prices submitted by non-revealed bidders. Detailed tender documents, however, were deemed confidential by the Department of Public Works and thus mandated additional independent data collection. Some contractors were contacted to provide project-specific information to be used in the model. Other data, such as soil type, has been identified based on the geographic location as classified by St. John's Department of Agriculture.

The data pertaining to the input parameters for all projects was then entered into

a spreadsheet program (*Microsoft Excel*) and further transformed into numerical values according to the representation of Fig. 3.2. The values for the "Construction season" parameter, for example, have been transformed into an integer from 1 to 3 for Winter, Spring/Summer, and Fall, respectively. With respect to the model outputs, on the other hand, the cost estimate of each project was available in the form of detailed unit prices. These were grouped together and five major cost items were used in further experiments with Neural Network 2, as shown in Fig. 3.2.

3.5 Neural Network Implementation: Spreadsheet Simulation

In an effort to develop a more realistic parametric cost model, this study attempts to overcome some of Neural Network drawbacks described early in Chapter 2 and presents it as a simple and transparent approach for use in construction. Accordingly, a three-layer Neural Network was simulated on a spreadsheet format that is easy to use, transparent, and customary to many practitioners in construction. The simulation of Neural Networks on a spreadsheet format presents its underlying mathematical formulas in a simple and fully controllable form.

A typical Neural Network consists of a group of processing elements organized into a sequence of layers with usually full connection between successive layers. Fig. 3.3 shows a simple three-layer network. The input nodes accept the data

that is presented to the network (representing model parameters) while the output nodes produce the network outputs (representing the decision associated with the parameters). The hidden nodes internally represent the relationships in the data and their values usually determined in a trial and error manner. Each processing element in the network performs a simple sum product of its inputs by the corresponding weight value. Also, a bias node is usually added to the input and hidden layers to facilitate network processing (Hegazy et al., 1994a). Those bias nodes are treated as regular nodes having inputs of 1.0 and are connected to successive layers as shown in Fig.3.3. The accuracy of the developed model depends on weight values. If optimum weights are reached, the weights and bias values encode the network's state of knowledge. Afterwards, using that network on new cases is merely a matter of simple mathematical manipulation of these values (Fig. 3.3).

A spreadsheet simulation of a three-layer Neural Network was implemented on *Microsoft Excel* (Fig. 3.4). It represents a template for a one hidden-layer network, which is suitable for most application (Hegazy et al., 1994b), with one output node. The processing of the template incorporates seven steps, following the widely known Backpropagation formulation (Rumelhart, 1986). The general structure and forward computations of this type of Neural Networks are presented in Fig. 3.3. These seven steps are as follows:

Step 1: Data Organization; as a preliminary stage to Neural Network modeling, the problem at hand needs to be thoroughly analyzed. Through this process, the independent factors affecting the problem are identified and considered as (N) input parameters represented by nodes at the input buffer of a Neural Network (Fig. 3.3). Similarly, the numbers of associated outputs or conclusions (O) are represented by nodes at the output layer. Once input and output parameters are identified, their relevant data is collected from the historical examples (P) available for later training of the Neural Networks.

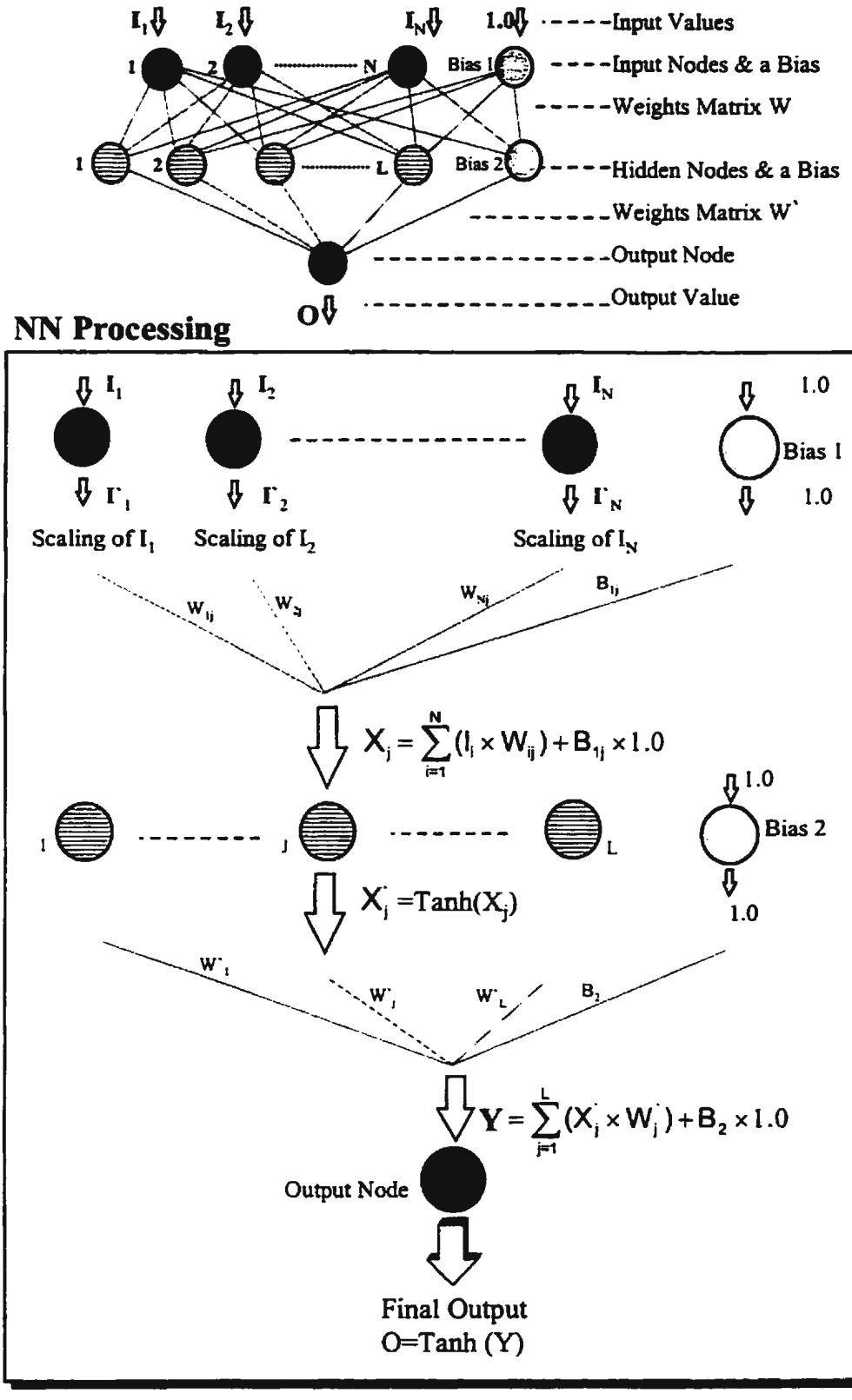


Fig. 3.3: Schematic Diagram of a Neural Network.

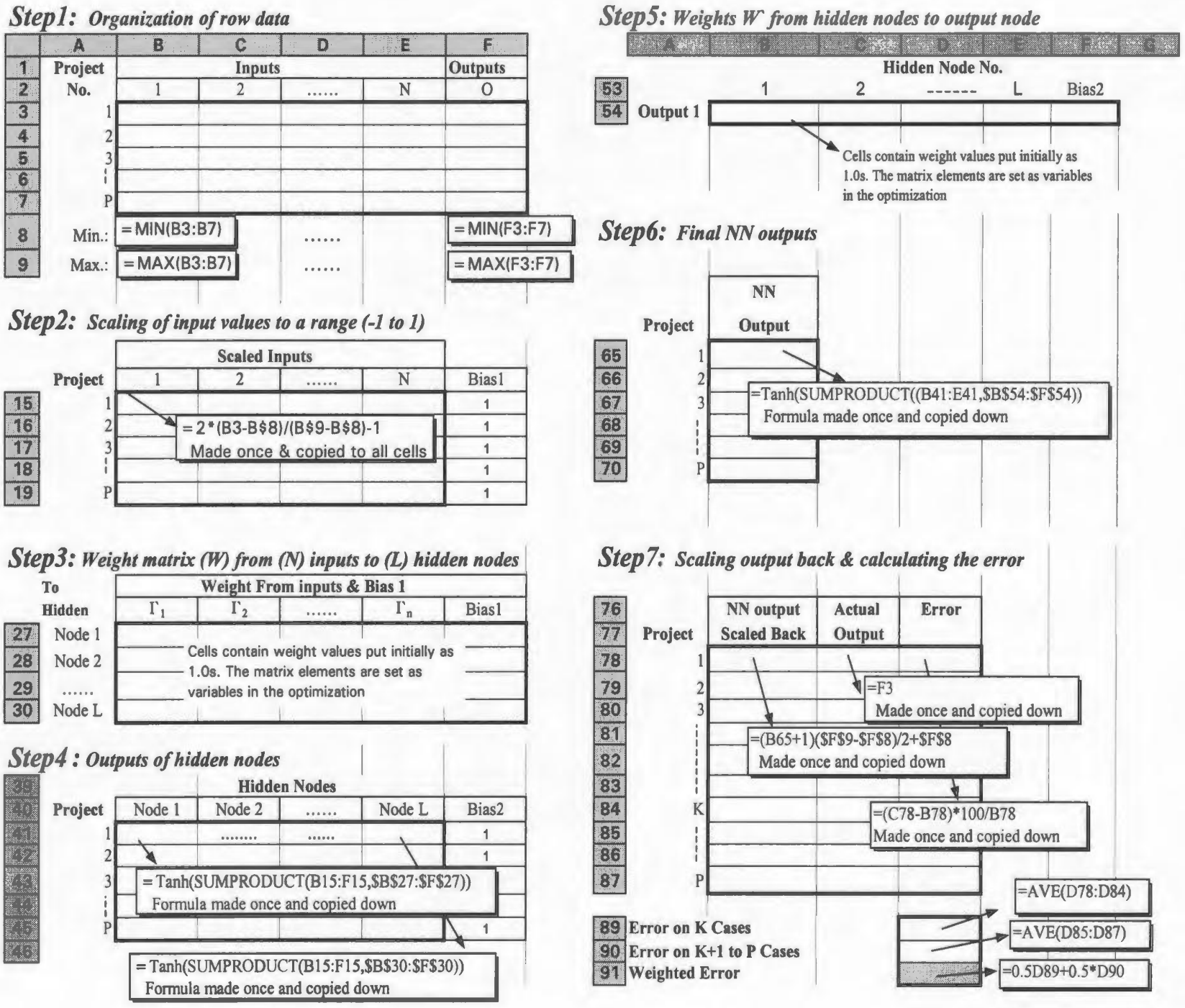


Fig. 3.4: Spreadsheet Simulation of a Three-Layer Neural Network.

To implement this step on *Excel* spreadsheet, the data is first transformed into numerical values and stored in a data-list which is a matrix of (N+O) columns and (P) rows (Fig. 3.4). The numerical transformation of textual data can be done in either a continuous or a binary manner. In continuous transformation, the value of a parameter called "Season", for example, can be an integer 0 to 3 for Winter, Spring, Summer and Fall, respectively. Alternatively, in binary transformation, four parameters are used to represent the four seasons and only one of them is assigned a value of 1.0 while the others are zeroes. Depending on the type of transformation used, the number of Neural Network nodes will be determined and accordingly, the size of the spreadsheet data-list. As shown in Fig. 3.4, two rows of formulas were made at the bottom of the row data-list with the minimum and maximum values of each column (input parameter).

Step 2: Data Scaling; In this step, the input data part of the first matrix (N columns by P rows) is scaled to a range [-1 to 1] to suit Neural Networks processing. This is done by constructing a second matrix with a linear formula for scaling the row value of each cell, as follows:

$$\text{Scaled Value} = \frac{2 \times (\text{Unscaled Value} - \text{Column Min.})}{(\text{Column Max.} - \text{Column Min.})} - 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (1)$$

This scaling formula is written in only one cell (B15, for example), and then copied to all cells in the scaling matrix. To the right of this matrix, a new column is added with unit values associated with the bias node, as illustrated in Fig. 3.4.

Step 3: Weight Matrix (W) from (N) Inputs to (L) Hidden Nodes; the third step is to construct and initialize the weight matrix between the inputs and the hidden layer. All inputs (1 to N) and a bias node were fully connected to the hidden nodes. The number of hidden nodes (L) has been set as one half of the total of input and output nodes, as heuristically suggested in the literature (Hegazy et al., 1994a). All the values in the weight matrix are considered variables to be determined in Neural Network modeling. Through preliminary experimentation, it was found that setting the initial weight values to a range of (0.5 to 1) is appropriate for inputs scaled to a range of (-1 to 1).

Step 4: Output of Hidden Nodes; the fourth step is to allow the hidden nodes to process the input data and produce values to be forwarded to the next layer. According to Neural Networks processing (Fig. 3.3), each hidden node *j* receives an activation X_j which is the sum product of weights by scaled inputs. Accordingly, the hidden node produces an output X'_i that is a function of its activation as shown in Fig. 3.3.

$$X_j = \sum_{i=1}^{N} (I_i \times W_{ij}) + B_{1j} \times 1.0 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2)$$

$$X'_j = \mathrm{Tanh}(X_j) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3)$$

Experimenting with different activation functions such as Linear, Logistic, Sine and Tanh has shown that the Tanh function is producing the best results. As

shown in step 4 of Fig.3.4, a formula was written for the first row of all hidden nodes and then copied to the down cells.

Step 5: Weight Matrix (W`) from Hidden Nodes (L) to Output Node; similar to the weight matrix constructed in step 3, a second matrix was constructed to connect (L) hidden nodes and a bias to output nodes. These weights are additional variables in the model and were initialized as previously described.

STEP 6: FINAL NNs Output; similar to step 4, the output of Neural Network (O) is computed by calculating the sum product (Y) of each output from hidden node by its connected weight and processing this value through the Tanh activation function producing (O) output as follows:

$$Y = \sum_{j=1}^{L} (X_j \times W_{j1}) + B_2 \times 1.0 \quad \ldots\ldots (4)$$

$$O = \text{Tanh}(Y) \quad \ldots\ldots (5)$$

Step 7: Scaling Output Back and Calculating the Error; in the seventh step, the Neural Networks output (O) is scaled back to the original range of values using the reverse of formula (1) as follows:

$$\text{Output Scaledback} = \frac{(\text{Output value} + 1)(\text{Max. Output} - \text{Min. Output})}{2} + \text{Min. Output} \quad \ldots\ldots (6)$$

To calculate a measure of the Neural Networks performance, a column is

constructed for determining the error between the actual output and Neural Networks output as follows:

$$\text{Estimating Error (\%)} = \frac{(\text{NNs output} - \text{Actual output})}{\text{Actual output}} \times 100 \quad \ldots\ldots\ldots\ldots (7)$$

It is also possible in the Neural Networks simulation to use some cases for training and others for testing by splitting the data into two groups, training and testing sets. The average error of each group of cases can be calculated in a different cell and then combined in a cell that calculates the performance measure of the Neural Networks, for example:

Weighted Error (%) = 0.5(test set average error) + 0.5(training set average error)

$\ldots\ldots\ldots\ldots\ldots\ldots (8)$

where weights of 0.5 and 0.5 were assumed for illustration. This approach gives high weight on the test cases (which are usually of small number as compared to training cases) to ensure good generalization performance and avoid overtraining. It is noted that the spreadsheet simulation of (Fig. 3.4) was constructed using only one output node for simplicity. However, the user may easily extend it to more than one output node to suit other applications.

3.6 Neural Network Implementation: Determining NN Weights

Since all formulas in the spreadsheet template are functions of the weights, the next step was to determine the Neural Network weight values that optimize network performance. Three approaches were used: 1) Neural Network training (using *NeuroShell2*, 1995); 2) Simplex Optimization (using *Microsoft Excel Solver*); and 3) Genetic Algorithms (using *GeneHunter*, 1995). Neural Networks training is one of the most common methods for determining Neural Networks weight values. The results of Neural Networks training, therefore, are considered as a baseline to which other approaches were compared. A Neural Network training program, *NeuroShell2*, was used as a standalone environment for Neural Networks development and training and later the two optimization approaches were utilized to arrive at the best Neural Network weights using the spreadsheet simulation. The details of each approach are provided in the following subsections.

3.6.1 Neural Network Training

For the problem at hand, *NeuroShell2* has been used for its ease of use, speed of training, and for its host of Neural Network architectures including Backpropagation and GRNN, with flexible user-optimization of training parameters. *NeuroShell2* includes a simplified set of procedures for building and executing a complete, powerful Neural Networks application. The user has the flexibility to specify his own learning rate, momentum, activation functions, and

initial weight range on a layer basis in the design module. It also has multiple criteria for stopping training in addition to different methods for handling missing data; pattern selection; and viewing weight and neuron values during training.

In addition to the brain-like structure of Neural Networks, the major property that deems Neural Networks superior to algorithmic and other AI-based systems is their ability to be trained. Training is the act of continuously adjusting the connection weights until they reach unique values that allow the network to produce outputs that are close enough to the desired outputs. As such, Neural Networks can generalize well on new cases if over-training is avoided (Hegazy et al., 1994c). To achieve good generalization, a special feature (NetPerfect) of *NeuroShell2* was used. NetPerfect optimizes training by exposing the network to the amount of training that minimizes the average error between actual and predicted results for a group of test cases. Using NetPerfect, two Neural Network architectures in *NeuroShell2* were tested: 1) Traditional Backpropagation (BP); and 2) General Regression Neural Network (GRNN). Figure. 3.5 shows the *NeuroShell2* Advanced Options screen and displays the independent modules that may be used to create a Neural Network application.

Traditionally, Backpropagation training is one of the most common methods for training Neural Networks on historical data. In essence, Backpropagation training adopts a gradient-descent approach of adjusting the Neural Network weights.

During training, a Neural Network is presented with the data thousands of times (called cycles). After each cycle, the error between the Neural Network outputs and actual outputs are propagated backwards to adjust the weights in a manner that is mathematically guaranteed to converge (Rumelhart et al., 1986). As opposed to the gradient-descent nature of Backpropagation, GRNN uses a multidimensional regression algorithm (*NeuroShell2* Reference Manual, 1995) that is able to train fast on sparse data. Using the Backpropagation architecture with NetPerfect, on the one hand, the network was saved whenever a new minimum average error is reached. The testing interval, which is how often the test cases are evaluated, was set to 50 epochs (training cycle). Using GRNN Networks, on the other hand, a regression smoothing factor (0 to 1) was optimized using NetPerfect.

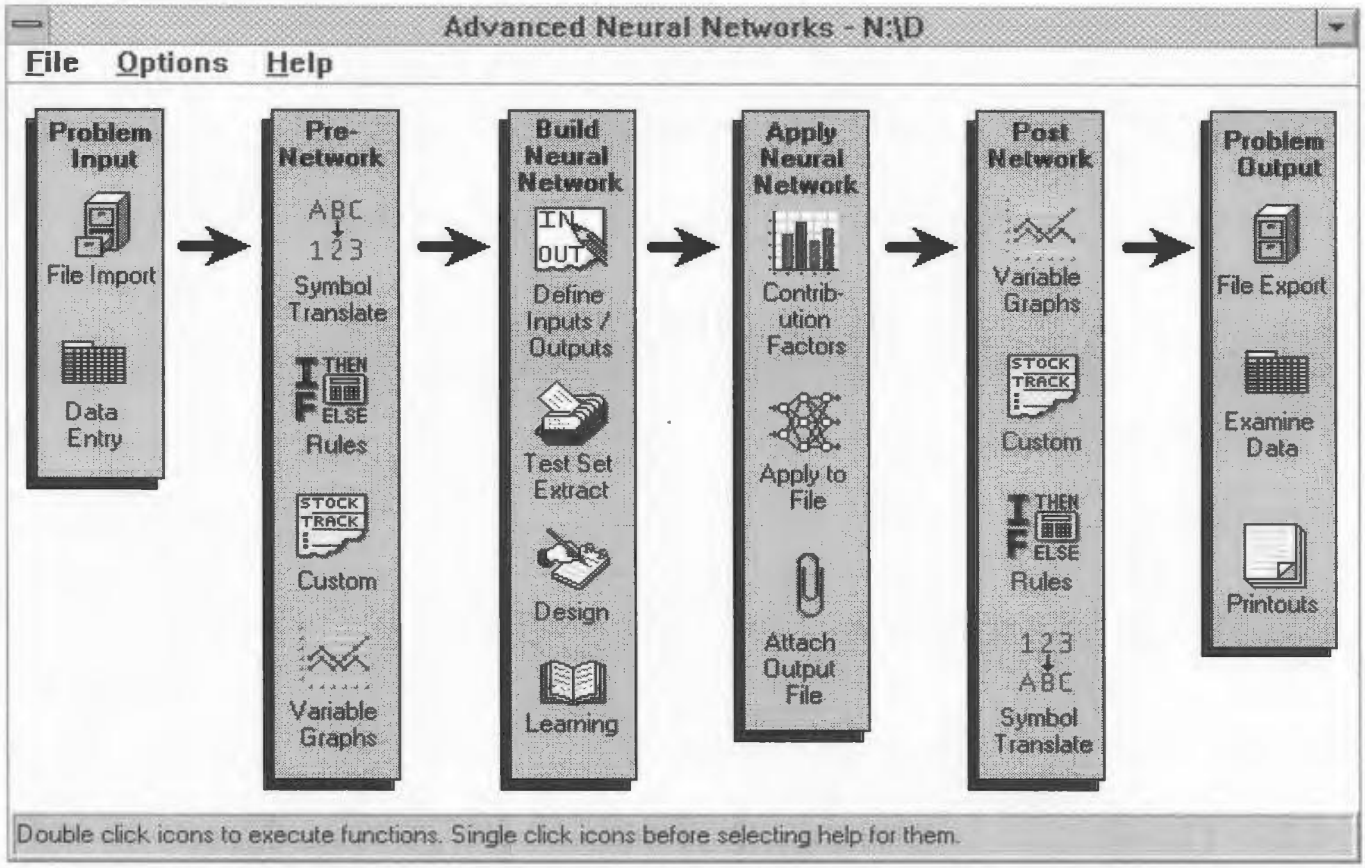


Figure 3.5: *NeuroShell2* Advanced Options Screen.

To implement the Neural Network training on *NeuroShell2*, the data of the 18 projects was first entered into a spreadsheet. Each row represents a case study from the 18 projects and each column represents one of the 10 input parameters. To the right of these columns, five columns were constructed to enter the 5 cost items for each project (as required for NN2 modeling) and an additional column for the total cost for each project (as required for NN1 modeling). By selecting the "File Import" option from *NeuroShell2* screen (Fig.3.5), this option was used to import the spreadsheet file from other systems into the *NeuroShell2*. Then, the "Data Entry" option was used to view and save the data file into *NeuroShell2*.

Afterwards, the "Define Inputs/Outputs" option can be used to choose which variables is to be used as network inputs and outputs, and to compute the minimum and maximum value for each variable. This option creates an .MMX file which is required to either train a network or to process a file through the trained network. For the case study at hand, the first 10 columns were selected as the input variables and the column contains the total cost as the output in case of NN1 modeling (five columns in case of NN2 modeling).

Since network training is as dependent on the quantity of training data as on how the data is presented to the network, several data-representation experiments and network architectures were conducted during training to arrive at the best-trained Neural Network. In these experiments, network parameters such as number of hidden layers, number of hidden nodes, network connections, and transfer functions were tested and the best result was documented. The "Design" option in *NeuroShell2* provides the user with various types of network architecture design to choose from. In case of a three layer network, for example, the user has the flexibility to experiment the network with different types of scaling functions, number of neurons on each layer (slab), learning rate, initial weights and momentum as shown in Figure 3.6.

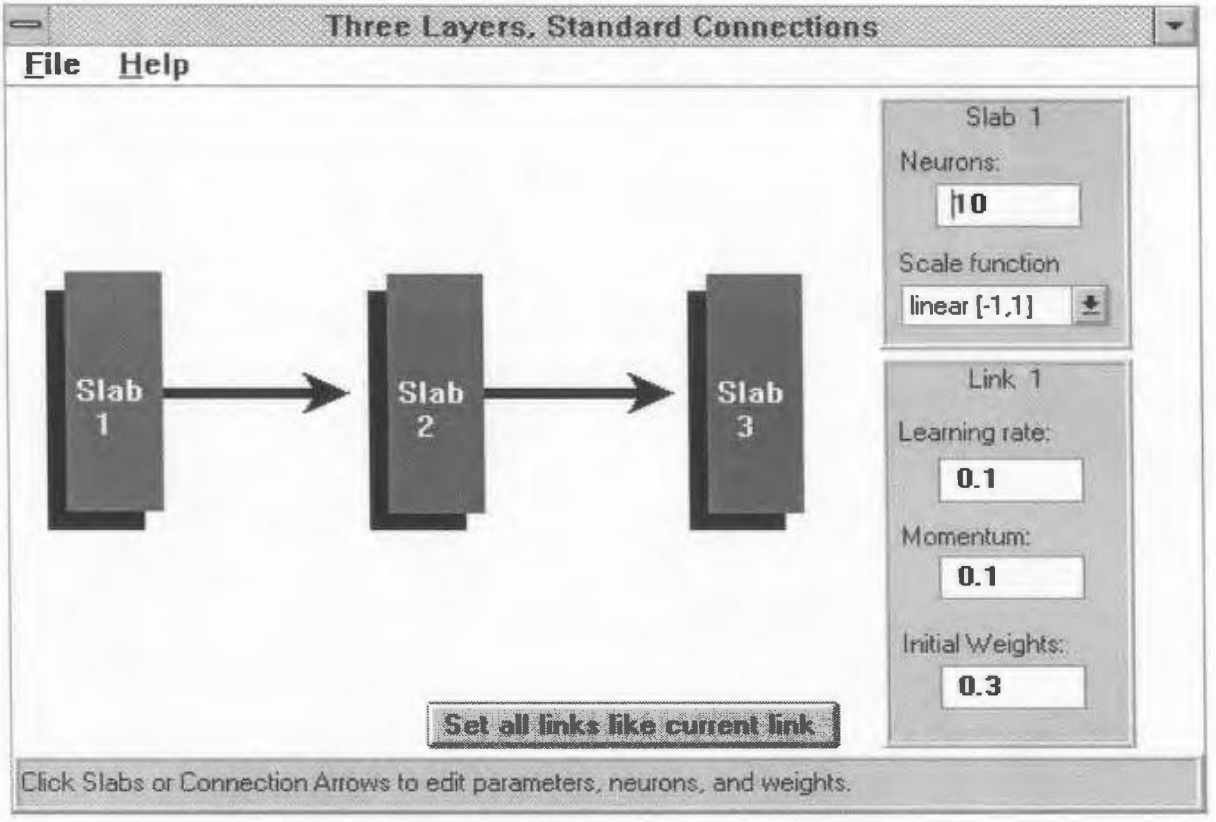


Figure 3.6: Design Parameters for 3-Layer Network.

With respect to data presentation, the "Test Set Extract" option was used to extract a test set from the data as shown in Figure 3.7. First, top fourteen cases were used in training and the last four cases used for testing (L4). As such, the third option in the "Extract" menu was selected with N=14, and M is blanked. Second, fourteen randomly selected cases were used in training and the other four for testing by the software according to its NetPerfect optimization feature (R4). Accordingly, the first option in the "Extract" menu was selected with N=25, and M is blanked. Table 3.1 shows the characteristics of the eight training experiments of this study; four for each Neural Network model.

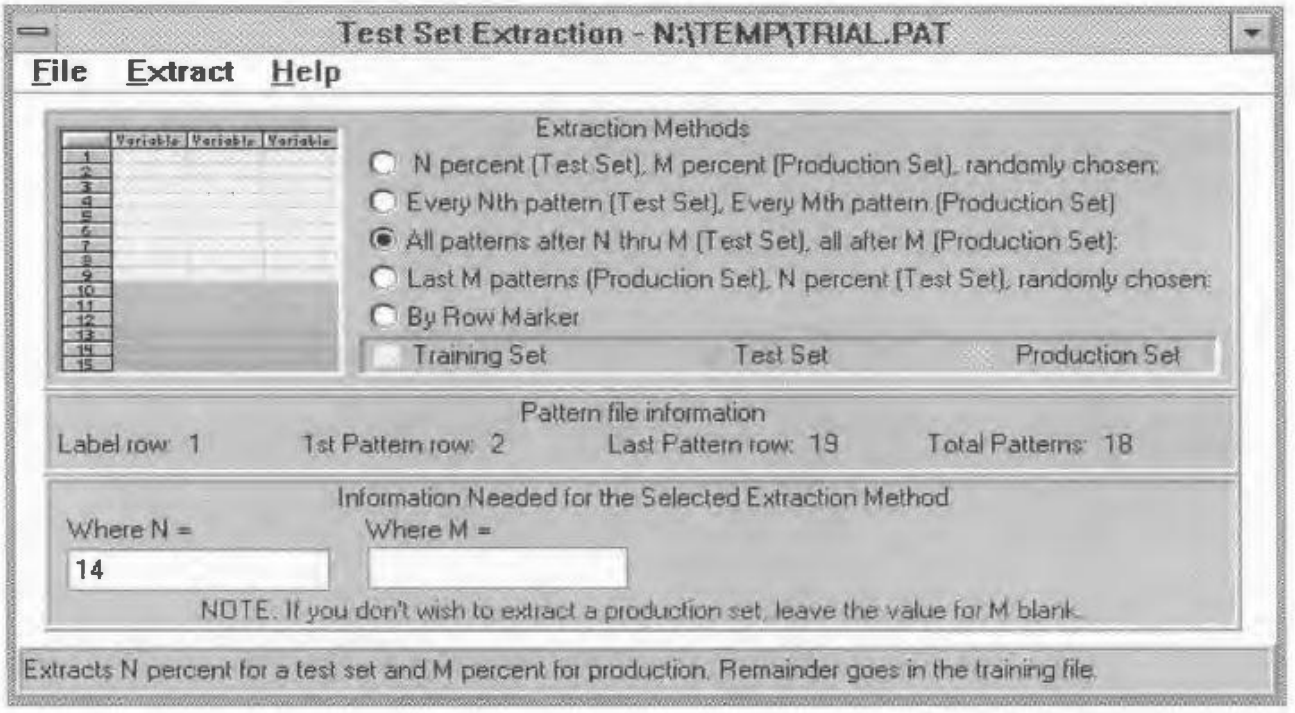


Figure 3.7: Test Set Extraction Screen.

Table 3.1: Neural Network Training Experiments

Experiment	Model	Architecture	Test Cases
1	NN1	BP	Last 4 cases (L4)
2	NN1	BP	4 cases at random (R4)
3	NN1	GRNN	Last 4 cases (L4)
4	NN1	GRNN	4 cases at random (R4)
5	NN2	BP	Last 4 cases (L4)
6	NN2	BP	4 cases at random (R4)
7	NN2	GRNN	Last 4 cases (L4)
8	NN2	GRNN	4 cases at random (R4)

BP = Backpropagation
GRNN = General Regression Neural Network

Once the network architecture was selected, each of the 8 Neural Networks outlined in table 3.1 was trained using the NetPerfect feature and the learning module of *NeuroShell2*. This module calls different learning subprograms depending upon the paradigm and architecture selected earlier. Figure 3.8 illustrates the learning options screen during network training. While other default settings were satisfactorily used for all GRNN networks, they had to be changed for BP networks and the optimum number of layers for each network was decided in a trial-and-error manner.

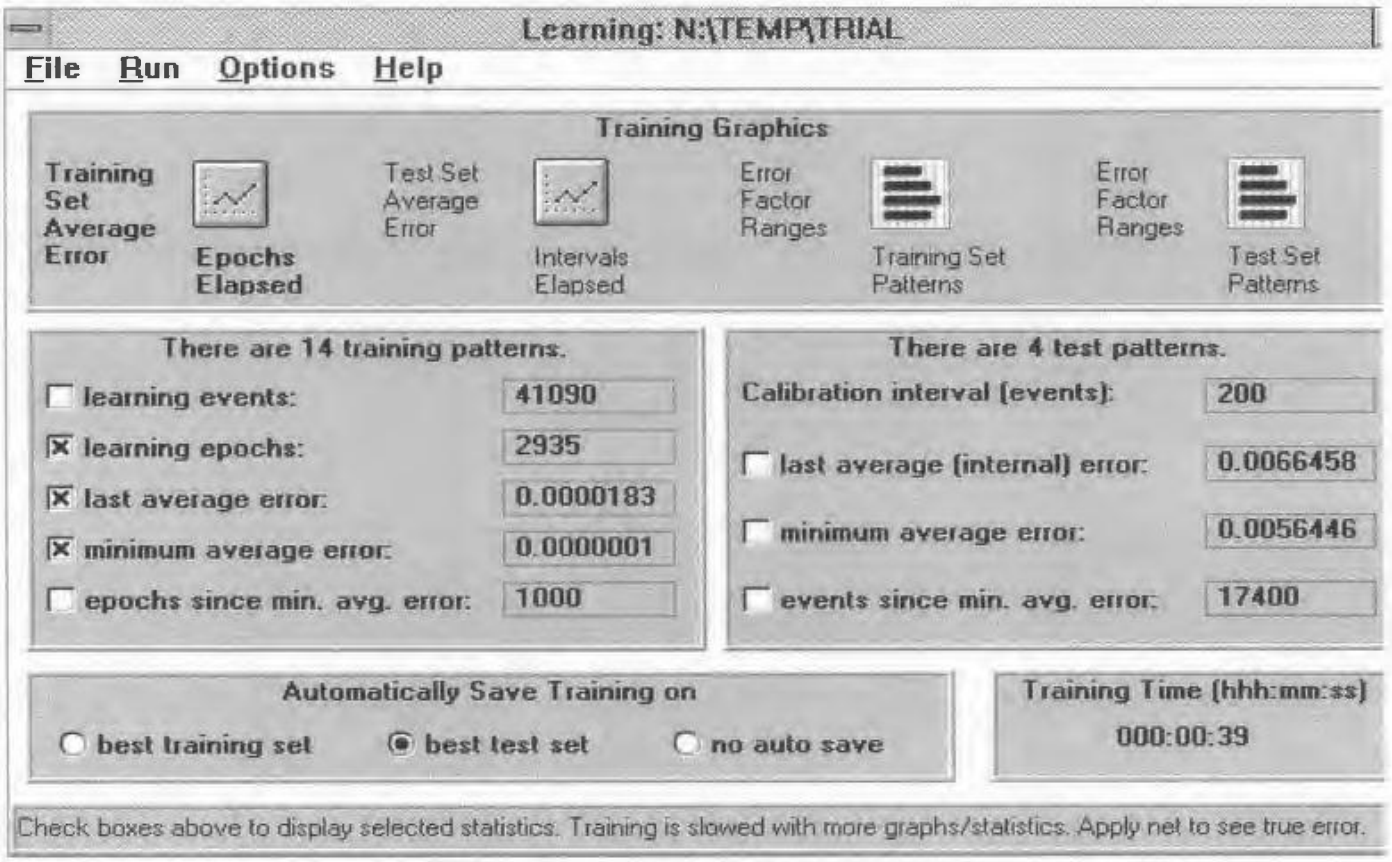


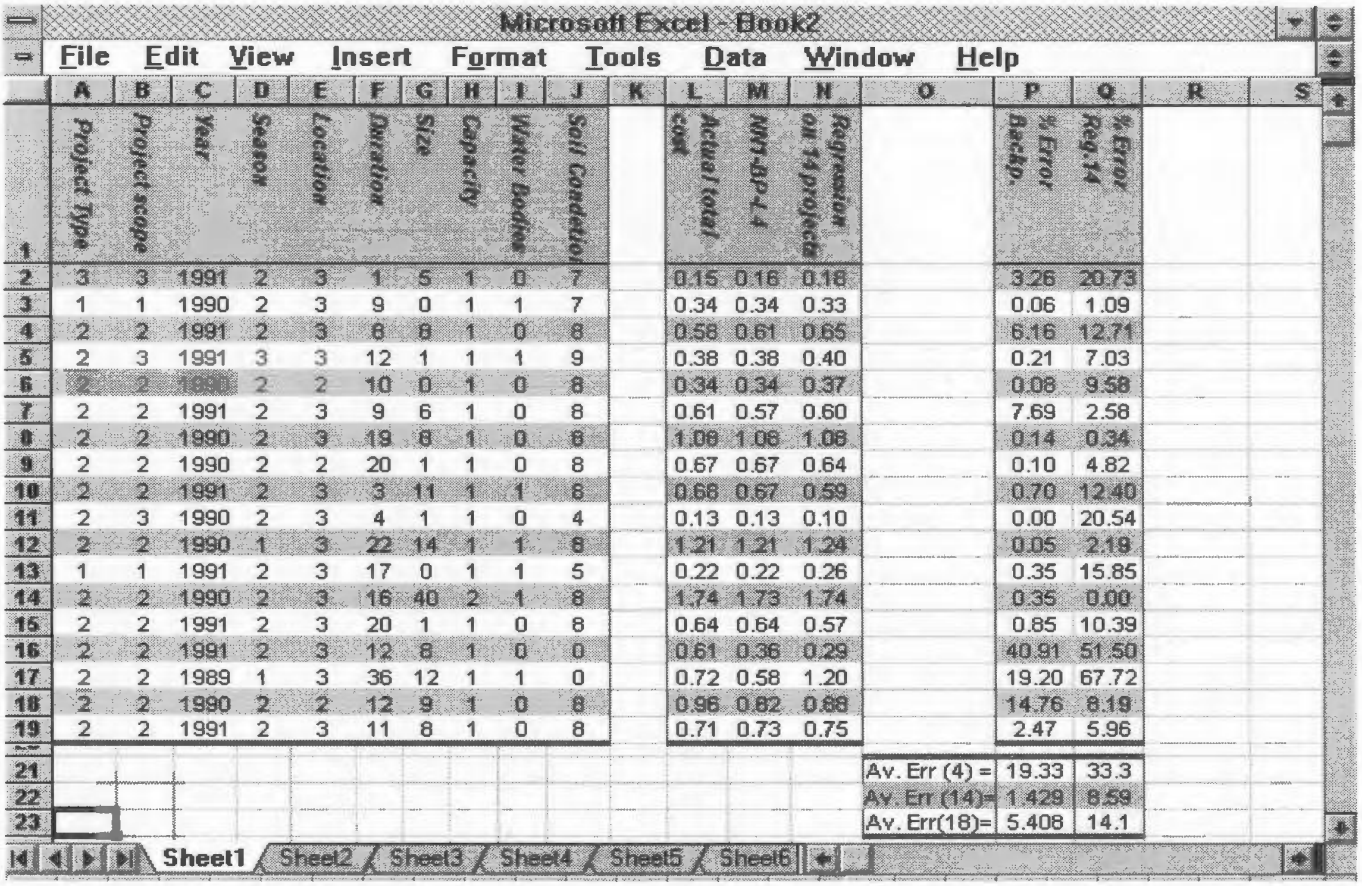
Figure 3.8: *NeuroShell2* Learning Screen.

Once the network was trained, the "Apply to File" option was used to process a data file through the trained Neural Network accordingly, produce the network's classifications or predictions for each pattern in the file (one value in NN1 and five items in NN2). A file of outputs (the .OUT file) was produced which can be activated from *Microsoft Excel*. As such, a spreadsheet was constructed on *Microsoft Excel* to compare among the Neural Network results.

In order to evaluate the performance of the Neural Network models as compared with traditional techniques, two multiple regression analysis experiments (experiment 9 and 10) were also conducted. One experiment used the input data of fourteen cases to predict project detailed cost, while the other used all the eighteen cases in the analysis. The regression analysis was implemented using the same spreadsheet on which the comparison among all results was conducted. Figure 3.9, for example, shows the comparison among Neural Network experiment results experiment 1, 9, and the actual cost.

Predicted versus actual costs were then plotted against those produced by regression analysis. Figure 3.10, for example, shows the performance of NN1 models (BP and GRNN) as compared with actual cost and regression analysis. It is noted that with the random ordering of projects presented in Fig. 3.10, the lines connecting the points serve no purpose other than better visual intelligibility. The percentage error in cost prediction of each model was calculated on the training

cases (14 projects) separately from the test cases (4 projects). The combined error on all eighteen projects is also presented in Table 3.2.



Microsoft Excel - Book2

File Edit View Insert Format Tools Data Window Help

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	Project Type	Project scope	Year	Season	Location	Duration	Size	Capacity	Water Bodies	Soil Condition		Actual total cost	NN1-BP-4-4	Regression on 14 projects		% Error Backp.	% Error Reg.14		
2	3	3	1991	2	3	1	5	1	0	7		0.15	0.16	0.18		3.26	20.73		
3	1	1	1990	2	3	9	0	1	1	7		0.34	0.34	0.33		0.06	1.09		
4	2	2	1991	2	3	8	8	1	0	8		0.58	0.61	0.65		6.16	12.71		
5	2	3	1991	3	3	12	1	1	1	9		0.38	0.38	0.40		0.21	7.03		
6	2	2	1990	2	2	10	0	1	0	8		0.34	0.34	0.37		0.08	9.58		
7	2	2	1991	2	3	9	6	1	0	8		0.61	0.57	0.60		7.69	2.58		
8	2	2	1990	2	3	19	8	1	0	8		1.08	1.08	1.08		0.14	0.34		
9	2	2	1990	2	2	20	1	1	0	8		0.67	0.67	0.64		0.10	4.82		
10	2	2	1991	2	3	3	11	1	1	8		0.68	0.67	0.59		0.70	12.40		
11	2	3	1990	2	3	4	1	1	0	4		0.13	0.13	0.10		0.00	20.54		
12	2	2	1990	1	3	22	14	1	1	8		1.21	1.21	1.24		0.05	2.19		
13	1	1	1991	2	3	17	0	1	1	5		0.22	0.22	0.26		0.35	15.85		
14	2	2	1990	2	3	16	40	2	1	8		1.74	1.73	1.74		0.35	0.00		
15	2	2	1991	2	3	20	1	1	0	8		0.64	0.64	0.57		0.85	10.39		
16	2	2	1991	2	3	12	8	1	0	0		0.61	0.36	0.29		40.91	51.50		
17	2	2	1989	1	3	36	12	1	1	0		0.72	0.58	1.20		19.20	67.72		
18	2	2	1990	2	2	12	9	1	0	8		0.96	0.82	0.88		14.76	8.19		
19	2	2	1991	2	3	11	8	1	0	8		0.71	0.73	0.75		2.47	5.96		
21															Av. Err (4) =	19.33	33.3		
22															Av. Err (14)=	1.429	8.59		
23															Av. Err(18)=	5.408	14.1		

Sheet1 Sheet2 Sheet3 Sheet4 Sheet5 Sheet6

Figure 3.9: Spreadsheet for Comparison among Different Results.

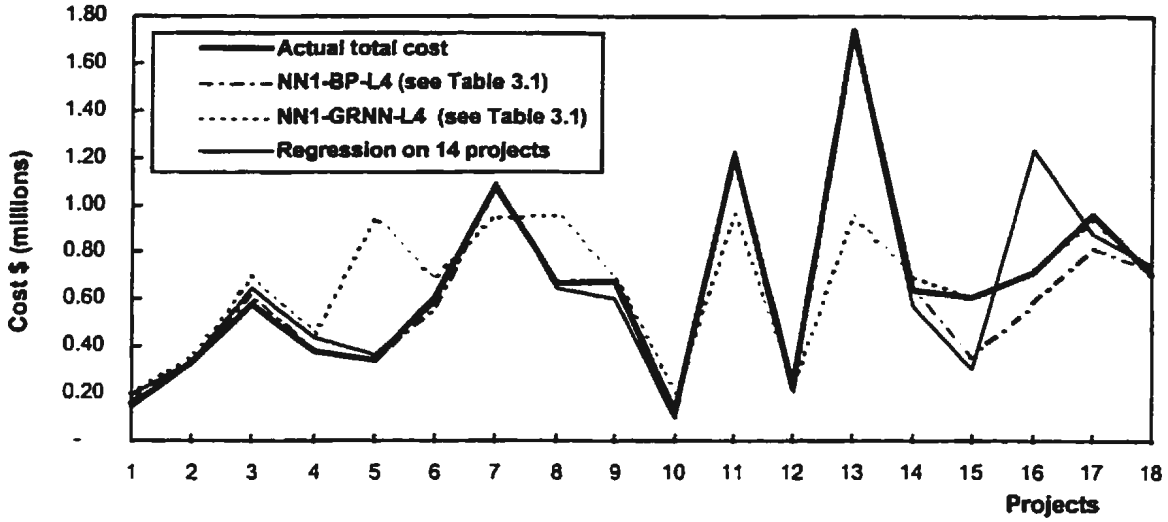


Figure 3.10: Actual versus Estimated Cost for NN1.

Table 3.2: Models' Performance Evaluation

Experiment	Model	Network *	Features	Error in Estimating the Total Project Cost(%) ***, Averaged on:		
				14 Training Projects	4 Test Projects	All 18 Projects
1	NN1	BP-L4	3-layer network	1.43	19.33	5.41
2	NN1	BP-R4	4-layer network	N/A	N/A	3.76
3	NN1	GRNN-L4	Default	20.28	1.59	16.12
4	NN1	GRNN-R4	Default	N/A	N/A	8.19
5	NN2	BP-L4	3-layer network	3.50	32.9	10.04
6	NN2	BP-R4	4-layer network	N/A	N/A	22.05
7	NN2	GRNN-L4	Default	9.97	17.11	11.56
8	NN2	GRNN-R4	Default	N/A	N/A	16.90
9	RA**	N/A	RA on 14 projects	8.59	33.34	14.10
10	RA	N/A	RA on 18 projects	N/A	N/A	13.82

* Refer to Table 1 regarding BP, GRNN, L4, and R4 ** Regression Analysis
*** % = Absolute [100 x (Actual Total Cost - Predicted Total Cost) / Actual Total Cost]

3.6.2 Simplex Optimization

As a first alternative to Neural Network training, a simplex optimization was implemented using *Solver*, an *Excel* add-in program. The implementation, therefore, was conducted directly on the Neural Networks spreadsheet of Fig. 3.4. *Solver* is a powerful and easy to use optimization tool that is highly integrated into *Excel*. It optimizes linear and integer problems using the simplex and branch-and-bound methods. *Solver* can find the optimum set of values for some variables so as to maximize or minimize a target cell (or objective function) that is linked by formulas to the variables, under a set of user-specified constraints. It proceeds by first finding a feasible solution, and then seeking to improve upon it; changing the variables to move from one feasible solution to another until the objective function has reached its maximum or minimum.

Using the described procedure of simulating a Neural Network on *Excel*, the data of the 18 project were then entered into *Excel* as shown in Fig. 3.11, with the textual values transformed into numbers according to the notations used in Fig. 3.2. All ranges and matrix dimension have been modified according to the numbers of inputs, outputs, and historical examples of the case study (i.e., =10, O=1, P=18, and L=5).

Step 1: Original Unscaled Inputs

		Project Type	Project scope	Year	Season	Location	Duration	Size	Capacity	Water Bodies	Soil Condetion	Actual Output
Training Cases	1	3	3	1991.00	2	3	1	4.75	1	0	7	0.151
	2	1	1	1990.00	2	3	9	0.1	1	1	7	0.335
	3	2	2	1991.00	2	3	8	7.9	1	0	8	0.579
	4	2	3	1991.00	3	3	12	1	1	1	9	0.376
	5	2	2	1990.00	2	2	10	0.4	1	0	8	0.339
	6	2	2	1991.00	2	3	9	6.4	1	0	8	0.612
	7	2	2	1990.00	2	3	19	7.9	1	0	8	1.080
	8	2	2	1990.00	2	2	20	0.87	1	0	8	0.673
	9	2	2	1991.00	2	3	3	11.2	1	1	8	0.678
	10	2	3	1990.00	2	3	4	1	1	0	4	0.129
	11	2	2	1990.00	1	3	22	13.5	1	1	8	1.210
	12	1	1	1991.00	2	3	17	0.1	1	1	5	0.220
	13	2	2	1990.00	2	3	16	40	2	1	8	1.738
	14	2	2	1991.00	2	3	20	0.9	1	0	8	0.637
Test Cases	15	2	2	1991.00	2	3	12	7.55	1	0	0	0.608
	16	2	2	1989.00	1	3	36	11.6	1	1	0	0.717
	17	2	2	1990.00	2	2	12	9	1	0	8	0.959
	18	2	2	1991.00	2	3	11	8.4	1	0	8	0.710
Min. Value		1	1	1989	1	2	1	0	1	0	0	0.13
Max.Value		3	3	1991	3	3	36	40	2	1	9	1.74

=2*(B3-B$21)/(B$22-B$21)-1
Made once and copied to all cells

Step 2: Scaled Inputs

Project		Project Type	Project Scope	Year	Season	Location	Duration	Size	Capacity	Water Bodies	Soil Condetion	bias1
	1	1	1	1	0	1	-1.000	-0.767	-1	-1	0.556	1
	2	-1	-1	0	0	1	-0.543	-1.000	-1	1	0.556	1
	3	0	0	1	0	1	-0.600	-0.609	-1	-1	0.778	1
	4	0	1	1	1	1	-0.371	-0.955	-1	1	1.000	1
	5	0	0	0	0	-1	-0.486	-0.985	-1	-1	0.778	1
	6	0	0	1	0	1	-0.543	-0.684	-1	-1	0.778	1
	7	0	0	0	0	1	0.029	-0.609	-1	-1	0.778	1
	8	0	0	0	0	-1	0.086	-0.961	-1	-1	0.778	1
	9	0	0	1	0	1	-0.886	-0.444	-1	1	0.778	1
	10	0	1	0	0	1	-0.829	-0.955	-1	-1	-0.111	1
	11	0	0	0	-1	1	0.200	-0.328	-1	1	0.778	1
	12	-1	-1	1	0	1	-0.086	-1.000	-1	1	0.111	1
	13	0	0	0	0	1	-0.143	1.000	1	1	0.778	1
	14	0	0	1	0	1	0.086	-0.960	-1	-1	0.778	1
	15	0	0	1	0	1	-0.371	-0.627	-1	-1	-1.000	1
	16	0	0	-1	-1	1	1.000	-0.424	-1	1	-1.000	1
	17	0	0	0	0	-1	-0.371	-0.554	-1	-1	0.778	1
	18	0	0	1	0	1	-0.429	-0.584	-1	-1	0.778	1

Step 3:

Weights of links from 10 inputs and a bias to 5 hidden neurons

1	3.319	0.058	0.088	-2.983	-5.712	1.787	-1.948	-0.647	0.270	3.462	3.834
2	2.177	-1.698	0.793	1.056	4.066	2.040	4.353	5.600	3.244	3.015	-4.187
3	0.609	0.541	1.672	4.177	-0.814	-0.654	-1.988	0.851	1.548	2.757	-3.453
4	-1.033	3.721	-2.741	-5.071	-1.888	1.090	-0.452	-0.586	-2.883	4.050	1.349
5	2.059	-1.118	-3.117	4.059	-2.592	-0.749	1.768	3.075	1.051	-0.180	-2.452

Fig. 3.11: NN Simulation for a Case Study of 18 Highway Projects.

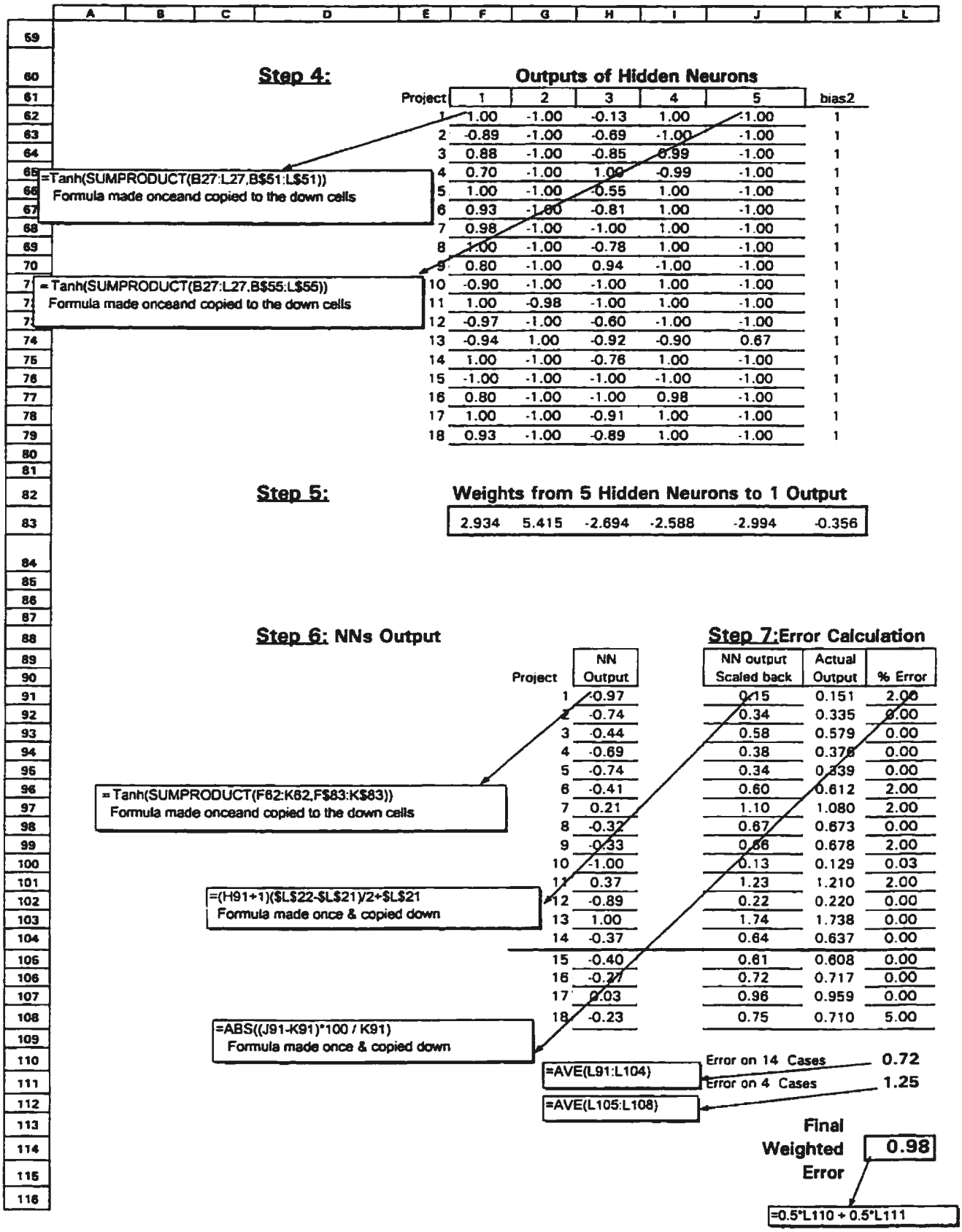

Step 4:
Outputs of Hidden Neurons
Project 1 2 3 4 5 bias2
1 1.00 -1.00 -0.13 1.00 -1.00 1
2 -0.89 -1.00 -0.69 -1.00 -1.00 1
3 0.88 -1.00 -0.85 0.99 -1.00 1
4 0.70 -1.00 1.00 -0.99 -1.00 1
5 1.00 -1.00 -0.55 1.00 -1.00 1
6 0.93 -1.00 -0.81 1.00 -1.00 1
7 0.98 -1.00 -1.00 1.00 -1.00 1
8 1.00 -1.00 -0.78 1.00 -1.00 1
9 0.80 -1.00 0.94 -1.00 -1.00 1
10 -0.90 -1.00 -1.00 1.00 -1.00 1
11 1.00 -0.98 -1.00 1.00 -1.00 1
12 -0.97 -1.00 -0.60 -1.00 -1.00 1
13 -0.94 1.00 -0.92 -0.90 0.67 1
14 1.00 -1.00 -0.76 1.00 -1.00 1
15 -1.00 -1.00 -1.00 -1.00 -1.00 1
16 0.80 -1.00 -1.00 0.98 -1.00 1
17 1.00 -1.00 -0.91 1.00 -1.00 1
18 0.93 -1.00 -0.89 1.00 -1.00 1
=Tanh(SUMPRODUCT(B27:L27,B$51:L$51))
Formula made onceand copied to the down cells
=Tanh(SUMPRODUCT(B27:L27,B$55:L$55))
Formula made onceand copied to the down cells
Step 5:
Weights from 5 Hidden Neurons to 1 Output
2.934 5.415 -2.694 -2.588 -2.994 -0.356
Step 6: NNs Output
Step 7:Error Calculation
Project NN Output NN output Scaled back Actual Output % Error
1 -0.97 0.15 0.151 2.00
2 -0.74 0.34 0.335 0.00
3 -0.44 0.58 0.579 0.00
4 -0.69 0.38 0.378 0.00
5 -0.74 0.34 0.339 0.00
6 -0.41 0.60 0.612 2.00
7 0.21 1.10 1.080 2.00
8 -0.32 0.67 0.673 0.00
9 -0.33 0.66 0.678 2.00
10 -1.00 0.13 0.129 0.03
11 0.37 1.23 1.210 2.00
12 -0.89 0.22 0.220 0.00
13 1.00 1.74 1.738 0.00
14 -0.37 0.64 0.637 0.00
15 -0.40 0.61 0.608 0.00
16 -0.27 0.72 0.717 0.00
17 0.03 0.96 0.959 0.00
18 -0.23 0.75 0.710 5.00
=Tanh(SUMPRODUCT(F62:K62,F$83:K$83))
Formula made onceand copied to the down cells
=(H91+1)(L22-L21)/2+L21
Formula made once & copied down
=ABS((J91-K91)*100 / K91)
Formula made once & copied down
=AVE(L91:L104)
Error on 14 Cases 0.72
Error on 4 Cases 1.25
=AVE(L105:L108)
Final Weighted Error 0.98
=0.5*L110 + 0.5*L111

Fig. 3.11 (Cont.)

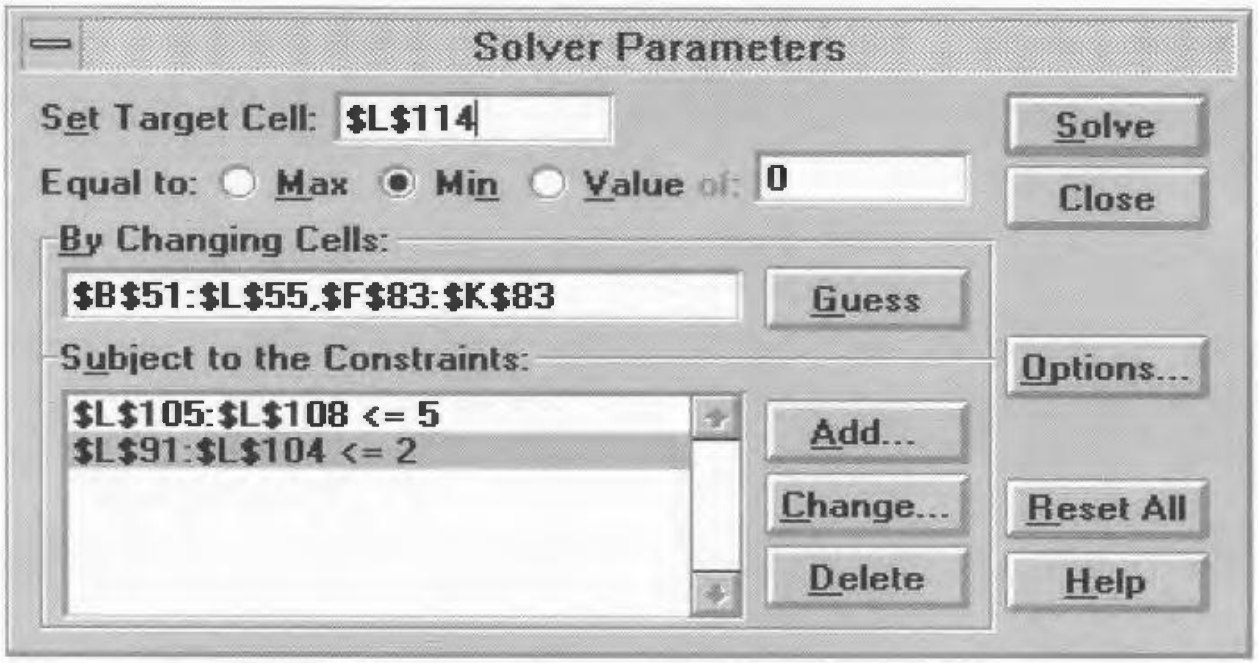


Figure 3.12: *Solver* Optimization Screen.

When *Solver* was activated, its optimization options were set as shown in Fig.3.12. The optimization objective is to minimize the target cell containing the Neural Networks weighted error (cell L114 of Fig. 3.11). Also, the optimization variables, representing the adjustable cells, are the weights from inputs to hidden nodes and from hidden nodes to outputs. In order to avoid erroneous network results on individual training cases, optimization constraints were set so as to limit the percentage error on the training and test cases to 2% and 5% (or lower) respectively. Cell references for the optimization variables and the constraints are shown in Fig. 3.12.

Once optimization parameters were input into *Solver* screen, the optimization process was started. Experimenting with this approach, it was found that the results are often sensitive to the initial values of the variables and some manipulation of *Solver* options may become necessary to arrive at the optimum solution. Using the suggested (0 to 1.0) range for the weights can be a good start. Also, when optimization is not improved over long period of time, it can be manually stopped and then continued after re-initializing some of the weight values. Generally, the time taken by *Solver* to solve a problem varies significantly depending on the size and complexity of the model. Solutions to all linear models can be found much faster than nonlinear models of equivalent size. In non-linear situation such as the present case study, the optimization process may need to be frequently interrupted to change *Solver* options which are fully described in *Excel* documentation (*Excel* Reference Manual, 1995).

Using *Solver* and experimenting with its various options on a trial and error basis, experimentation took 2 hours on a Pentium 120 MHz machine. An optimum solution was reached with a weighted error of **0.98%** (optimization target) with **0.71%** and **1.25%** error on training and test sets, respectively. The resulting network weights and errors are shown in the Neural Networks spreadsheet of Fig. 3.11.

3.6.3 Genetic Algorithms (GAs)

This technique is another optimization method that is fundamentally different from traditional simplex-based algorithms such as the one used by *Excel Solver*. It uses the method of evolution, specifically survival of the fittest. First, a population of possible solutions to the problem is created. Individuals in the population are then allowed to randomly breed, a process called crossover, until the fittest offspring is generated (Hegazy et al., 1994b). After a large number of generations, a population eventually emerges wherein the individuals will provide an optimum or close to optimum solution.

For the case study at hand, a Genetic Algorithm software (*GeneHunter*, 1995) was used to find the optimum weights of the model. *GeneHunter* is an *Excel* add-in tool that can replace *Excel Solver* for optimizing complicated problems. It also includes a Dynamic Link Library (DLL) of genetic algorithm functions that may be called from programming languages such as *Microsoft Visual Basic or C*.

The *GeneHunter* screen is shown in Fig. 3.13 with all the cell references to the optimization objective function and constraints. The objective, or fitness, function specifies the location of the cell to be optimized. Similar to *Solver* optimization, cell L114, representing the Neural Networks weighted error, was selected to be minimized. The adjustable cells containing the optimization variables (called chromosomes in GAs terminology) were also specified as the two weight

matrices. Optimization constraints were then set. These constraints limit the range of weight values within which *GeneHunter* will search for a solution, thus reducing processing time. In addition, the constraints add restrictions or subgoals to the original fitness function so as to limit the percentage error on training and test sets to 2% and 5%, respectively.

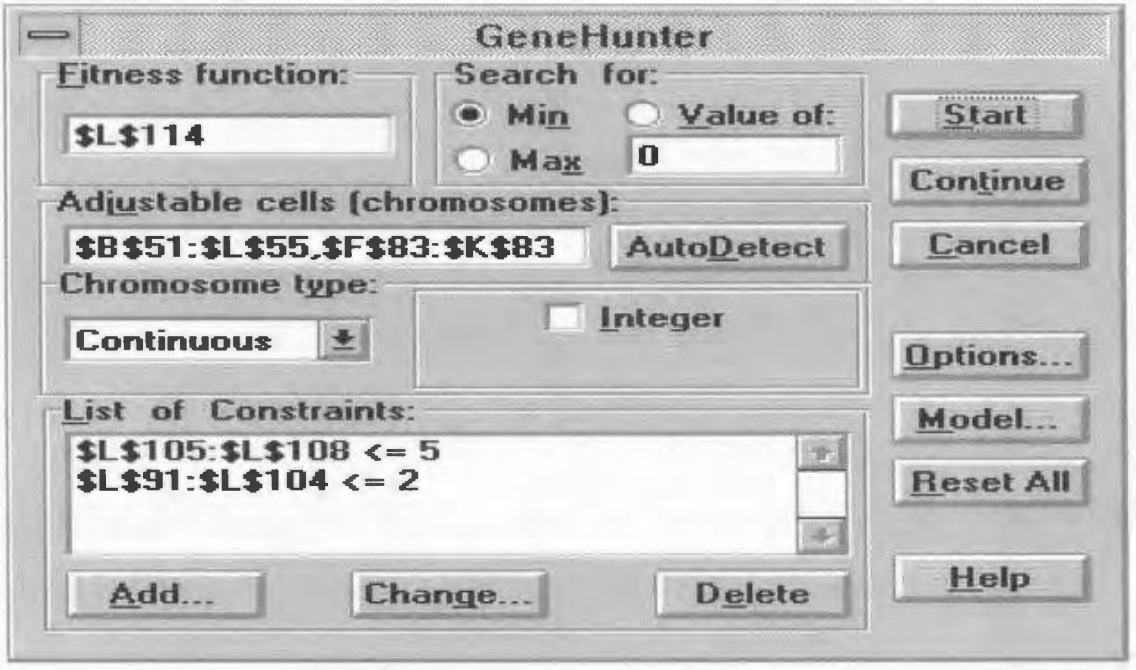


Figure 3.13: *GeneHunter* Optimization Screen.

During the Genetic Algorithm optimization, *GeneHunter* options can be used to enhance the results during optimization of the problem. For example, "Population Size" affects processing time since the fitness function must be calculated for every individual in every generation. A population of size 50 was found as a good number to start with. This number can be increased later during the optimization process. Chromosome Length presents the level of accuracy needed for the

adjustable cells. More bits mean high precision answers. One possible strategy for problem solving is to first start with 8 bit chromosomes and then increased to 16 bit when little progress is being made towards a better solution. "Crossover Rate" which is the probability that the crossover operator will be applied to a particular chromosome during generation must be set fairly high (0.9 is usually good). "Mutation Rate" which is the probability that the mutation operator will be applied to a particular chromosome during generation must be set fairly low (0.01 is fairly good). When using the Elitism Strategy, the individuals who live are the fittest individuals. It should almost always be on so that the most desirable genetic characteristics are available for breeding in subsequent generation. A good strategy for problem solving is to start Elitism option off to allow the population to evolve without significant selective pressure and then after a while, it may be turned on in order to concentrate the optimization process around the best solution. Diversity option is also used to produce individuals that are slightly changed during mutation. This option can make a major contribution to evolutionary progress and should be turned on during optimization process. With the above *GeneHunter* settings and on a trial and error basis during the optimization process, *GeneHunter* was able to come up with an over all weighted error of **21.8** % with **22.48** % error on training set and **21.11** % on the test set.

3.7 Discussion of Results

3.7.1 Neural Networks Training versus Regression Analysis

Several interesting observations are drawn from the results of Table 3.2. As expected, Backpropagation networks, as in experiments 1 and 5, have small errors on the training cases and higher errors on test cases. GRNN networks, on the other hand, did not exhibit a similar consistent behavior. The best overall model for NN1 can be identified as Backpropagation (experiments 1 and 2), having excellent performance on the training cases and less than 20% error on new cases (often acceptable at the budgetary level). It is noted that while experiment 3 (GRNN) produced a network with small errors on the test cases, it behaved relatively poorly on the training cases which is reflected on its overall error (16.12%). For the NN2 model, on the other hand, Backpropagation (experiment 5) and GRNN (experiment 7) produced reasonable results. Despite the slightly higher overall errors of experiment 7, it exhibits a more consistent performance than experiment 5. It is concluded, therefore, that for the NN1 model, the networks of experiments 1 and 2 are most suited while the networks of experiments 5 and 7 are most suited for the NN2 model. In using these networks, the estimates of the two selected networks may indicate any erroneous results and also may be averaged to arrive at a final cost estimate. It can be inferred from the results obtained that the NN1 model, in general, performs better than the NN2 model, probably due to the larger size of NN2 (5 outputs) and accordingly requires a larger number of training cases. Accordingly, NN2 model

was not used with further experimentation using *Solver* and *GeneHunter* optimizations. Furthermore, it is demonstrated that the BP and GRNN networks consistently performed better than Regression Analysis on new cases and as such provide a better predictive model.

3.7.2 Neural Networks Training versus Optimization

Since the Backpropagation network of experiment 1 performed better than other networks, it can be considered as a baseline to which the experimented optimization approaches will be compared. The comparison of results, as shown in Fig. 3.14, determines the optimum Neural Network weights. The best overall model is the one produced by *Excel Solver*, having excellent performance on both the training and test cases. While Backpropagation training produced a network with small errors on the training cases, its learning performance was slightly inadequate on the test cases which was reflected on its overall error (10.4%). For the Genetic Algorithms, on the other hand, the model did not produce reasonable results probably due to the random selection of the generated population. Although The model performed uniformly over training and test sets, it produced an overall error (21.8%) which is slightly above the parametric estimating level of accuracy (20%). It is concluded, therefore, that the network optimized by Simplex Optimization is the most appropriate network to the present case study.

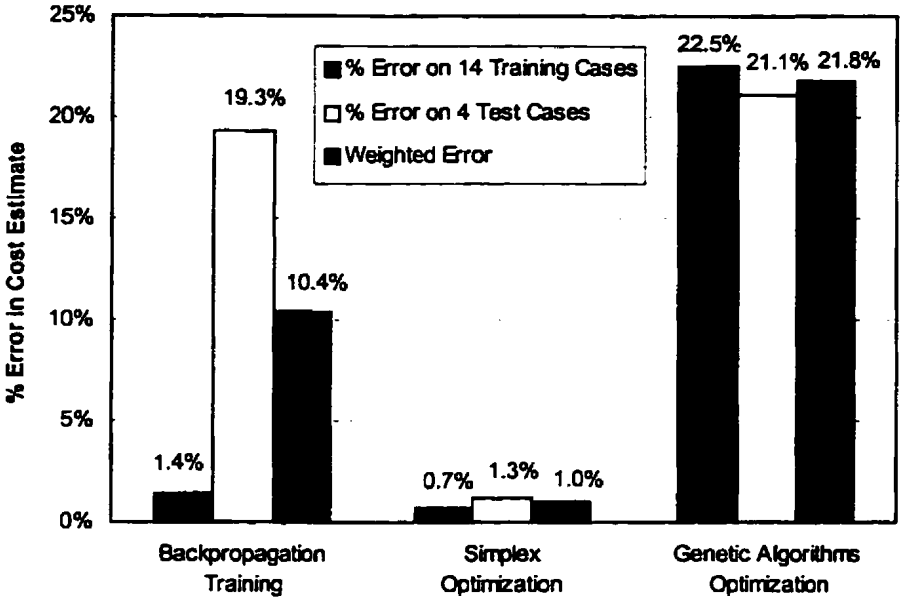


Figure 3.14: Comparison among Weight-Determining Methods.

3.8 Conclusion

In this chapter, the data of 18 highway projects constructed in Newfoundland during the past five years were used as training examples. Accordingly, two Neural Network models were trained to predict the cost of new jobs using one output and 5 outputs, respectively. Issues regarding the modeling approach of Neural Networks were discussed, including data preparation, model formation, network configuration and Neural Networks implementation. To validate the developed models, a comparison was conducted between the results of two common Neural Network architectures (Backpropagation and GRNN) and those of multiple regression analysis. The results signify that Neural Networks have captured the relations embedded in the trained data and in turn indicate better

predictive models than traditional models. It was also concluded that the model with one output was more accurate than the one with 5 outputs.

As an alternative to traditional Neural Network training, the structure of a simple Neural Network was simulated on *Microsoft Excel* program. Two approaches, Simplex optimization and Genetic Algorithms were then applied on the spreadsheet simulation to find the optimum weights of the Neural Networks. Based on the obtained results, it was found that Simplex Optimization, rather than Backpropagation or Genetic Algorithm optimization, determined the optimum Neural Network weights. The resulting optimum model and the spreadsheet simulation will be utilized to implement a comprehensive system for parametric cost estimating of highway projects. The system is fully described in the next chapter.

CHAPTER 4

A Comprehensive System for Parametric Cost Estimating

4.1 Introduction

In parametric cost estimating models, it is always difficult to decide which parameters to be measured and how to evaluate their significance to the model. This is due to the lack of information at the early stage of project life cycle at which the parametric cost estimating is usually performed. Thus, it is desirable to find the key parameters which give the relevant information and measure the level of importance and uncertainty for each parameter. Such uncertainty can be accounted for by conducting a sensitivity analysis. Also, additional modules need to be integrated with the system to facilitate the storage of the data and adapt the developed model to new environments.

Once the optimum Neural Network Model was selected, as discussed in chapter 3, other modules have been integrated with the optimum Neural Network to develop a comprehensive parametric estimating system (Fig. 1.1, chapter 1). The system consists of: 1) A user interface; 2) A sensitivity analysis module to determine the sensitivity of the predicted cost to changes in cost-related parameters; 3) A database module to store new highway projects into the historical cases; and 4) An adaptation module to re-optimize the Neural Network model on new historical cases and accordingly adapt model's performance to new environments. The development of these modules and the operation of the whole system is described in the following subsections.

4.2 The User Interface

A user-friendly interface to the parametric estimating system was developed on *Microsoft Excel* using its macro tool. *Excel* macros can be used to automate repetitious tasks and make an application easier to use. It can be created by turning on the macro recorder, and then manually work through the process that needs to be automated, and accordingly a macro code will be automatically written. Such macro can then be assigned to a button on the screen. The macro code can also be edited and modified to suit the application or to correct mistakes.

Also, before recording the macro, it should be worked through the process so that the procedure can be organized. There are two ways to begin recording a macro: Select the Tools menu, then select Record Macro and Record New Macro. The other way is by clicking on the Record Macro button on the Visual Basic toolbar as shown in Fig. 4.1.

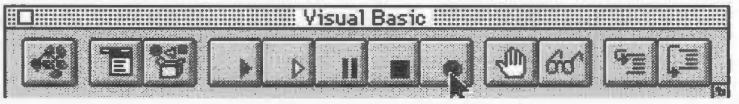


Figure 4.1. Visual Basic Toolbar

A dialogue box will appear that will ask to name the new macro, and also give a brief description of its function. After clicking OK, every operation was performed will be recorded in this macro (from entering data, formatting cells, opening a file,.....etc.) until the macro is turned off. The macro can be turned off by clicking on the Stop macro button, either the one on the Visual Basic toolbar or the freestanding one that will appear somewhere on the screen when the macro is recording (*Excel* Reference Manual, 1996). Using *Excel* macros, the user interface was developed to facilitate the operation of all modules.

4.3 Sensitivity Analysis Module

Once the optimum Neural Network model has been selected as described early in Chapter 3, it can be put to actual use in predicting the budget cost for new highway projects. In this case, the Neural Network is presented with the user's best judgment concerning the ten characteristic factors that describe the project. However, at this early project stage in which the model is applicable, project characteristics might not be decided for certain. A practical model, therefore, has to assess the sensitivity of the model's predictions to variation in the project characteristics. Therefore, a sensitivity analysis module has been incorporated into the present model. This module was coded in *Microsoft Excel* macros and was linked to the Neural Networks spreadsheet. The sensitivity analysis screen is shown in Fig. 4.2, with two steps for performing the analysis. First, the user inputs the project data by clicking on "Enter Project Data" button. A user-friendly data-input screen (see Fig. 4.3) was activated and the data for a sample project was input. Second, the user selects the parameter or combination of parameters in which he/she lacks confidence in the initial judgment. As shown in Fig. 4.2, three parameters were selected for demonstration: project type, location, and duration. Accordingly, a plot comparing the Neural Networks initial cost prediction with those of a group of 20 project scenarios generated with little random variation to the initial project data.

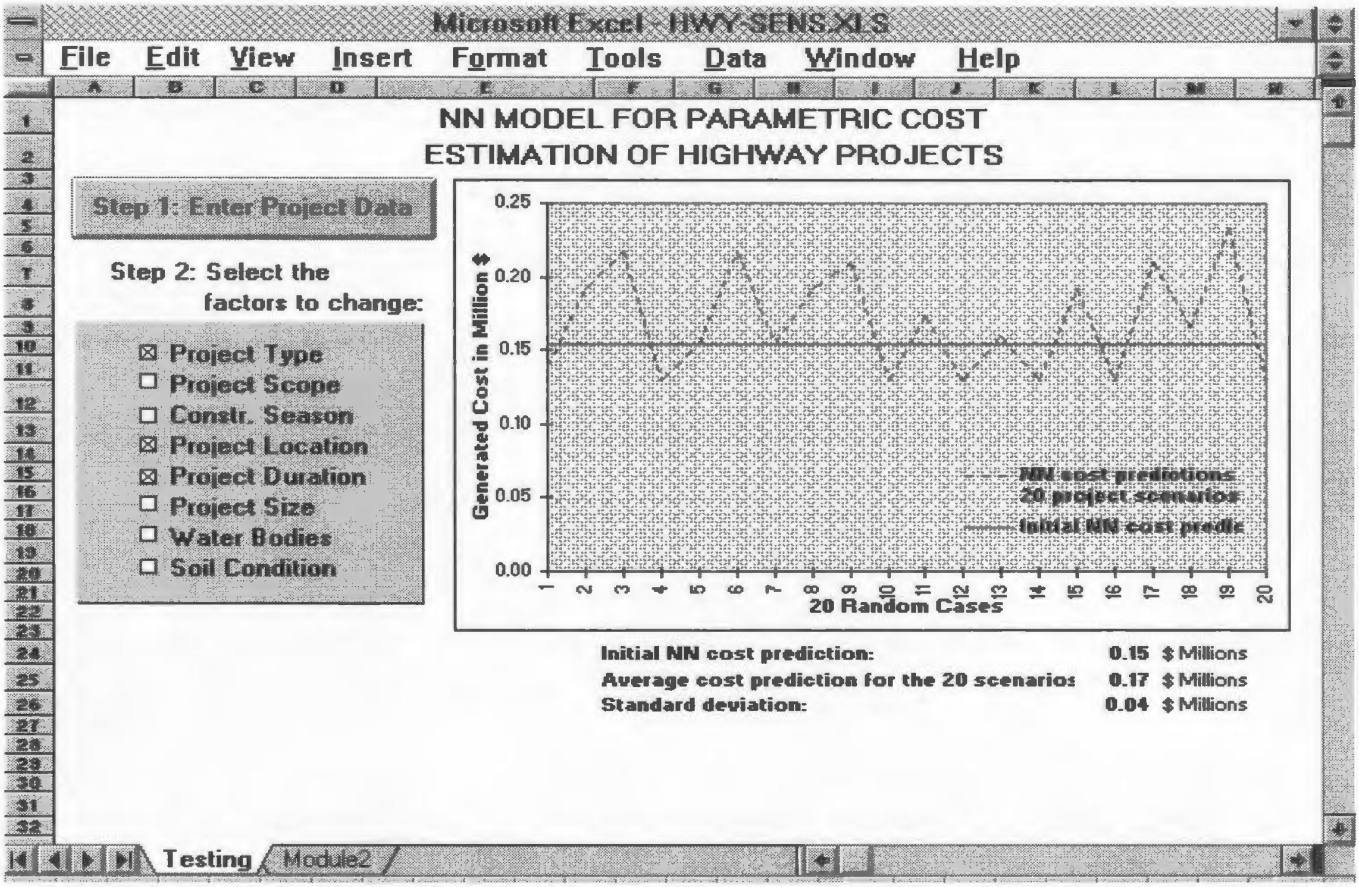


Figure 4.2: Sensitivity Analysis Screen.

The random variations were introduced only in the parameters that the user has selected and the sensitivity analysis was conducted in a manner similar to traditional Monte Carlo simulation. In order not to introduce impractical scenarios into the sensitivity analysis, random variations were limited to a range that is $\forall$ 25% of the initial parameter value, and bounded by the maximum and minimum limits. Other parameters that have discrete values, such as "project scope", were dealt with as integer values. The calculations underlying the sensitivity analysis was done in a spreadsheet similar to that of Fig. 3.11 with only the first matrix changed with formulas to calculate the values for the different scenarios. Neural

Network predictions, as such, are instantaneous and are plotted automatically as shown in Fig. 4.2. The macro executing the sensitivity analysis is shown in appendix A. The mean and the standard deviation of the predicted costs for the 20 scenarios can be compared to the initially estimated cost. The mean value represents the expected change in predicted cost, considering variations in the selected parameters. This approach can be used to determine the relative significance of each input parameter in the model. This can be done by using the same initial data and then conducting the sensitivity several times with a single parameter selected in each, then comparing the resulting mean and standard deviation values.

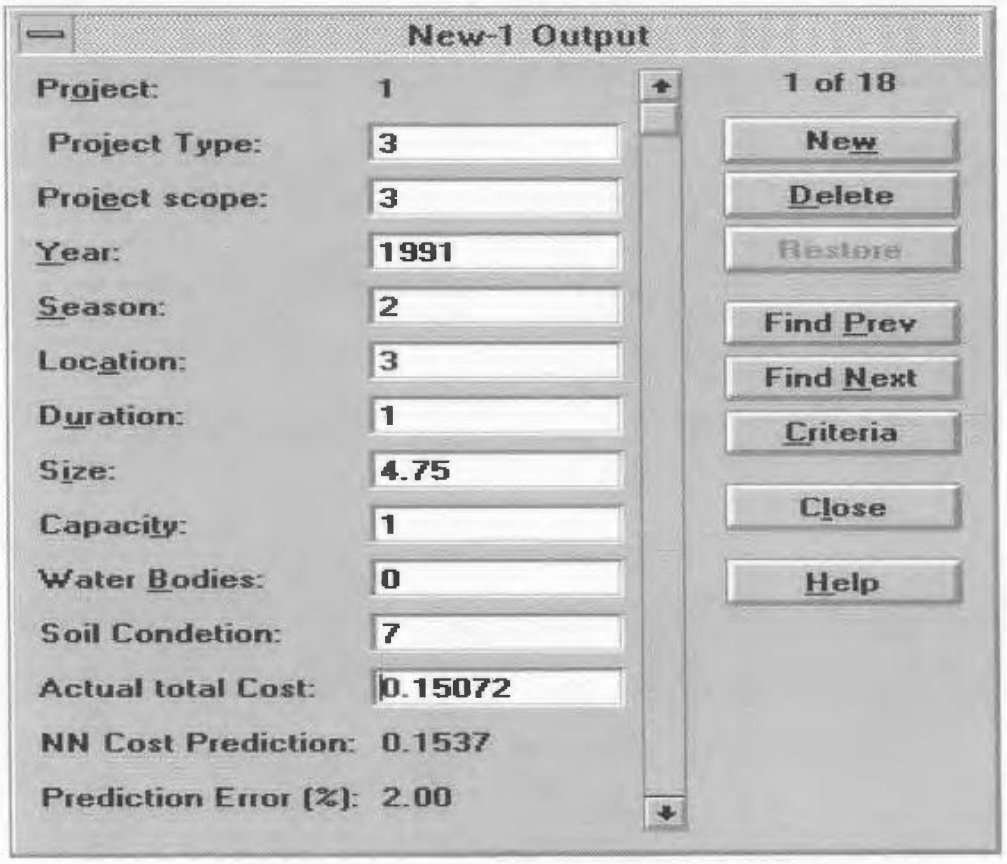


Figure 4.3: Project Data-Input Screen.

4.4 Historical Database and Model-Adaptation Modules

One important aspect of a practical model is to adapt it to new project situations. This enables it to adjust its contractor-independent nature to become more suited to the user's own work environment. It also enables the build-up experience and incorporates new experiences into the model. In the present study, therefore, an adaptation module has been added to the model. This module uses the user's own historical project cases as a representative of his work environment. It then re-optimizes the Neural Networks of the model on those cases. Fig.4.4 shows the adaptation screen, developed using *Excel* macros. The macro for integrating the adaptation module with the optimum model is shown in appendix B. Initially, the user's historical projects need to be entered into the model by selecting "Add a New Project" button. The user will be prompted for project data and accordingly the cell ranges in the Neural Networks spreadsheet (Fig. 3.11) will be increased and the formulas copied to accommodate for an additional example. Once all data is entered, the "Review and Modify Data" button can be used to browse and modify the projects' data in a manner similar to Fig.4.3. To re-optimize the Neural Networks, the user has the flexibility to re-optimize on the total number of projects in the historical database by clicking the "Re-Optimize on All Cases" button. Alternatively, the user can re-optimize the Neural Networks using only the newly added projects by clicking the "Re-Optimize on New Cases Only" button. In each of the two optimization

options, cell references and *Solver* optimization parameters are adjusted automatically and the user is prompted out when the process is completed.

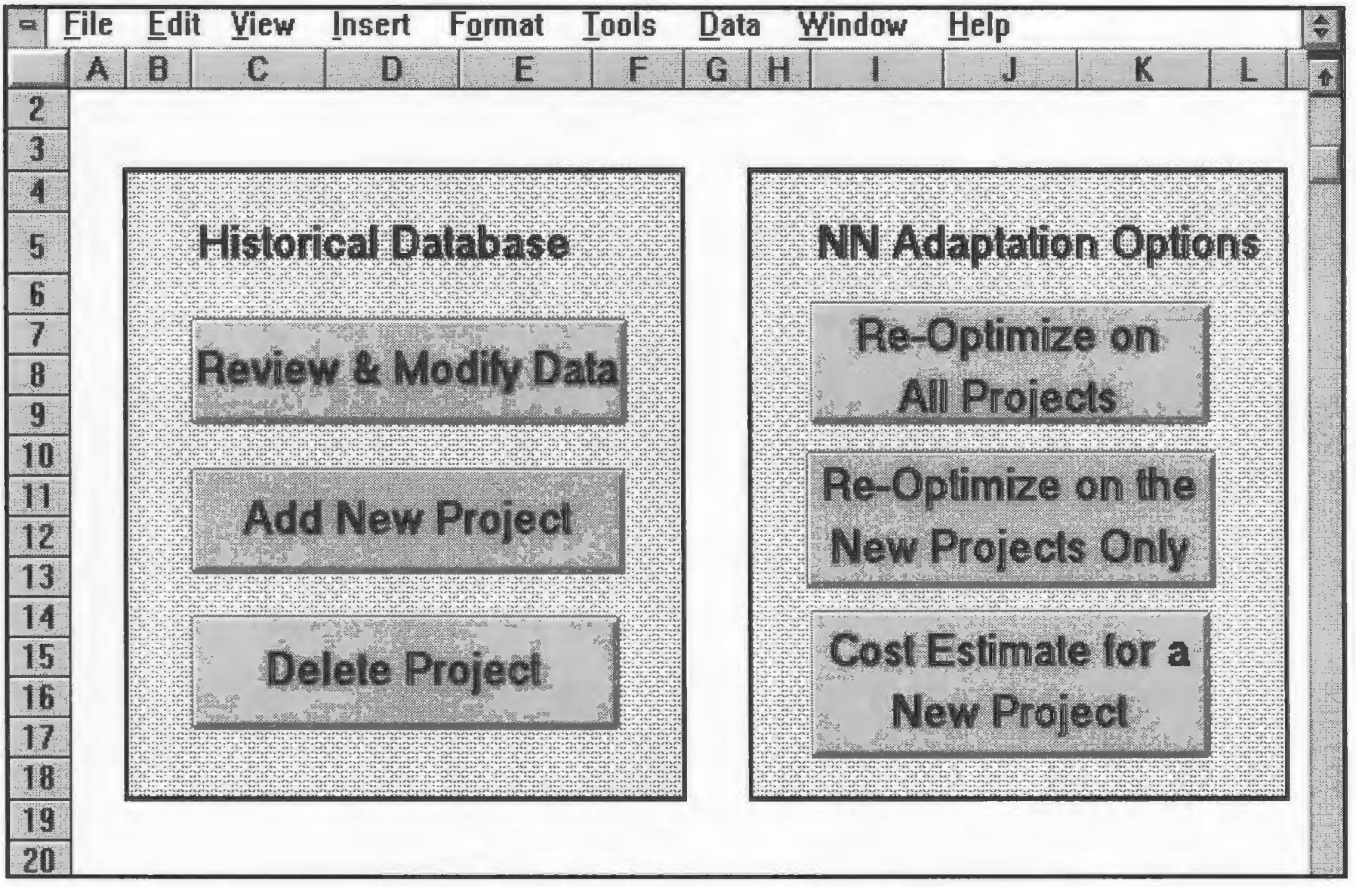


Figure 4.4: Model Adaptation Screen.

Once the model is re-optimized, it can be used to estimate the cost of new projects by selecting the "Estimate New Project" button. It should be noted that model adaptation permits the user to add new historical data to the model, without introducing changes to the structure of the model itself such as the number of inputs and hidden nodes, which had been fixed at an early stage.

4.5 Conclusion

In this chapter, a complete parametric estimating system is presented and coded in a user-friendly software using *Microsoft Excel*. The software uses Neural Networks for parametric cost estimation that derive solutions for new highway projects. A sensitivity analysis is included to account for the uncertainty in the model parameters. The objective is to study the sensitivity of the model output (predicted cost) to changes in cost-related parameters. The sensitivity analysis is carried out by changing each cost-related parameter independently and studying how the model output is being influenced by these changes. In addition, the software provides a tool that facilitates storage of previous project encounters, efficient data editing, model adaptation to the user's environments, and necessary computations.

CHAPTER 5

Conclusion

5.1 Comments on Present Developments

In this study, the Neural Networks technique was employed to develop a parametric cost estimating model for highway projects. The study demonstrates the benefits of using Neural Network technique for effective management and re-use of historical cost data. The data of 18 highway projects constructed in Newfoundland during the past five years were used to train the Neural Networks. The structure of a simple Neural Network was simulated on *Microsoft Excel* program with its main components and elements to determine the optimum Neural Network model. Three different approaches, Neural Network training Algorithms, Simplex optimization and Genetic Algorithms were utilized to find the

optimum weights of the Neural Networks. These approaches greatly benefit from the simulation of Neural Network on a spreadsheet. Issues regarding the modeling approach of Neural Networks were discussed, including data preparation, model formation, network configuration and Neural Networks implementation.

A comparison was conducted between the results of two common Neural Network training Algorithms (Backpropagation and GRNN) and that of multiple regression analysis. The results signify that Neural Networks have captured the relations embedded in the trained data and in turn indicate better predictive models than traditional ones. It was concluded, therefore, that the performance of Backpropagation network model with the total cost as the only model output is performing better than other training techniques.

Next, the Simplex optimization and Genetic Algorithms were applied to optimize the model using the Neural Network spreadsheet simulation and a comparison among Backpropagation, Simplex Optimization and Genetic Algorithms was conducted. Based on this experimentation, the Simplex Optimization produced the optimum Neural Network. *Excel* macros were then utilized to encode the optimum model in a user-friendly software to facilitate user input of cost-related parameters for new projects and accordingly, predict their budget costs. For practicality, the developed model provides a module to study the sensitivity of the

model to changes in cost-related parameters and to determine their relative significance. In addition, the present model incorporates an adaptation module that allows the user to input new project cases and utilize them to re-optimize the model to his/her particular environment. The developments made in this research demonstrated the practicality of using spreadsheet programs in developing adequate Neural Network models for use in construction.

Since their introduction in early 1980's, spreadsheet programs have been among the most easy-to-use software programs that include powerful data management capabilities. The use of spreadsheets in construction has, therefore, been customary to many practitioners and several applications, particularly in cost estimation, were developed in their familiar spreadsheet format (Pickard 1997; and Compton 1987). In the present developments, the use of a spreadsheet program has brought several benefits to the development process and presumably to the end user. It was possible to simulate the Neural Networks process in a transparent form, and further optimize it using spreadsheet tools. This presents Neural Networks as a viable tool for use in construction by adjusting the developed template to other applications. Also, spreadsheets incorporate many powerful features including formula computations and unlimited customization tools that are easy to use. The user, therefore, does not have to program any routines for creating reports and printing results. In addition, recent versions of spreadsheet programs have included powerful data

management techniques and scenario-management capabilities. They have also included links to the Internet to present information and allow the sharing of files among work groups. Their capabilities offer many general-purpose features that can be used to develop modules to integrate with existing ones to form global solutions. Users can also select among many add-in modules available on the market to extend spreadsheet capabilities.

The research conclusions and limitations can be summarized as follows:

- Neural Networks has demonstrated to be a promising tool for use in the initial stages of construction projects when typically only a limited or incomplete data set is available for cost analysis.

- Compared with traditional approaches, this technique is simple and flexible, requires no complicated mathematical modeling and makes no assumptions about the relationship between the cost related parameters.

- The developed model makes significant advances in traditional highway estimating methods by means of using state-of-the-art techniques in the analysis of cost and provides a methodology to adapt the model to new construction environments.

- With this model, the effects of cost-related parameters on the total cost of construction projects can be investigated through its sensitivity analysis procedure.

- This model was developed only using data collected from St. John's and surrounding area, and accordingly the model can not be used to predict construction costs for other areas. However, new historical data can be fed into the model and re-optimized to develop a new estimating model. In such cases, it is recommended to re-optimize the data using only the new historical data which presents the new environments.

- Although many efforts were made to collect as much information as possible, the presented model for St. John's area can be enhanced by feeding the model with more training examples. In this case, it is recommended to eliminate redundant and distorted examples and keep only cases with realistic relationship between inputs and outputs.

- The model developed in this study can only be used at the preliminary design stage at which the acceptable level of accuracy is within ±20% range.

5.2 Future Research

The current research is being expanded to investigate the ability of applying the developed model to The Canadian Strategic Highway Research Program (C-SHRP) and the Canadian Long Term Pavement Performance (C-LTPP) Project. The C-LTPP program documents a comprehensive database of field test results to evaluate the performance of a wide range of pavement types, over a 20-year time frame. The study involves over 2800 test sites located on primary highways in North America. The advantages of expanding the present study to this program are to improve the performance of highways by relating the life cycle cost of highway construction to its service environment and type of pavement construction. This reduces not only the initial costs of construction and enables the estimation of realistic construction cost but also supports the selection of appropriate pavement type and maintenance activities to suit a particular highway condition.

REFERENCES

Akeel, N. (1989). "A database tool for statistically-based construction cost estimate.", Ph.D. thesis, University of Colorado at Boulder, USA.

Al-Bani, M. (1994). "Developing a concrete cost estimate model for small residential buildings.", Ph.D. thesis, King Fahd University, Saudi Arabia.

Arditi, D., and Riad, N. (1988). "Commercially available cost estimating software systems.", *Proj. Management Journal.* 19(2), 65-70.

Black, J. (1982). "*Cost engineering planning techniques for management.*", M. Dekker Inc., New York, N.Y.

Compton, J. (1987). *Construction management using Lotus 1-2-3.* Management Information Resource Inc., Portland, OR.

Creese, R. and Li, L. (1995). "Cost estimation of timber bridges using Neural Networks." *Cost Engineering*, AACE International, 37(5),17-22.

Design 4/Cost User's Guide (1995), DC&D Technologies Inc., 8602 N. 40th st., Tampa, Florida.

Ellis, M. (1989). "A model for prediction of highway construction production rates.", Ph.D. thesis, University of Florida, USA.

Excel 97 reference manual (1997). Microsoft Corporation, One Microsoft Way, Redmond, WA.

Fayek, A., Duffield, C., and Young, D. (1994). "A review of commercially available cost-estimating software systems for construction industry." *Engineering Management Journal.* 6(4), 23-33.

Garza, J. and Rouhana, K. (1995). "Neural Network versus parameter-based application." *Cost Engineering*, AACE International, 37(2), 14-18.

GeneHunter reference manual (1995). Ward Systems Group, Inc., Frederick, MD.

Hegazy, T., Moselhi, O., and Fazio, P. (1993). "Managing construction knowledge in patterns: A Neural Network approach." *Transactions of First International Conference in the Management of Information Technology for Construction*, CIB, Singapore,pp 331-343.

Hegazy, T., Fazio, P., and Moselhi, O. (1994a). "Developing practical Neural Network applications using Back-propagation." *Microcomputers in Civil Engineering*, Blackwell Publishers, 9(2), 145-159.

Hegazy, T., Moselhi, O., and Fazio, P. (1994b). "Analogy-based solution to markup estimation problem." *J. Comp. in Civil Eng.* , ASCE, 8(1), 72-87.

Hegazy, T., Moselhi, O., and Fazio, P. (1994c). "A Neural Network approach for representing implicit knowledge in construction. " *The International Journal of Construction Information Technology*, University of Salford Press, U.K., 1(3), pp.73-86.

Hegazy, T., and Ayed, A. (1997). "A Neural Network approach for effective management of highway cost data."*Preceding of the First European Conference on Intelligent Management Systems in Operations*. Salford, England, pp. 215-222.

Hollman, J. (1994). "A parametric building cost estimating system.", *AACE Transactions*, AACE International, pp. EST.4.1-EST.4.7.

Lee, H. (1992). "Automated interactive cost estimating system for reinforced concrete building structures.", Ph.D. thesis, The University of Michigan, USA.

Lopez, O. (1993). "Forecasting construction costs in hyper-inflated economies.", Ph.D. thesis, The University of Texas at Austin, USA.

Mckim, R. (1993a). "Neural network application to cost engineering." *Cost engineering*, AACE International, 35(7), 31-35.

Mckim, R. (1993b). "Neural networks and identification and estimation of risk.", *AACE Transactions*, pp. P.5.1-P.5.10.

Melin, J. (1994). "Parametric estimation.", *Cost engineering*, AACE International, 36(1), 19-23.

Mohan, S. (1990). "Expert system applications in construction management engineering," *J. Constr. Engrg. Mgmt.*, ASCE, 116(1), 87-99.

Moselhi, O., Hegazy, T. and Fazio P. (1990). "Neural Networks as tools in construction." *J. Constr. Engrg. Mgmt.*, ASCE, 117(4), 606-625.

Moselhi, O. Hegazy, T. and Fazio P.(1994). "DBID: Analogy-Based DSS for Bidding in Construction." *J. Constr. Engrg. Mgmt.*, ASCE, 119(3), 466-479.

NeuroShell2 Reference Manual (1995). Ward Systems Group Inc., Frederick, MD.

Pantzeter, A. (1993) "A methodology for modeling the cost and duration of concrete highway bridges.", Ph.D. thesis, Purdue University, USA.

Pickard, S. (1997). "Integrated spreadsheets." *Civil Engineering*, ASCE, 67(6), 44-45.

R.S. Means (1997). "*Building Construction Cost Data.*", R.S. Means Company Inc., Kingston, MA.

Rumelhart, D., Hinton, G. and Williams, R. (1986). "*Parallel Distributed Processing.*", Vol 1: Foundations, MIT Press, Cambridge, MA.

Salamh, A. (1989). "A knowledge-based expert system for conceptual estimates and designs.", Ph.D. thesis, University of Colorado at Boulder, USA.

Savin, D., and Kumar, S. (1993). "*A review of forecasting techniques and their applications to building economics.*" Univ. of Concordia, Montreal, Canada.

Smith, A. (1989). " Accuracy in progress Measurements.", AACE Transactions, *AACE International*, pp. D.4.1-D.4.5.

Success Tutorial (1995), U.S. Cost Incorporated, 5605 Glenridge Drive, Atlanta, Georgia.

Uhlik, M. (1984). "Optimizing earthwork estimating for highway construction.", Ph.D. thesis, The Pennsylvania State University, USA.

Williams, T. (1994). "Predicting changes in construction cost indexes using neural networks.", *J. Constr. Engrg. Mgmt.*, ASCE, 120(2), 306-320.

Yau, N. (1992). "An object-oriented project model for integrating building design, construction scheduling, and cost estimating for mid-rise construction.", Ph.D. thesis, University of Illinois at Urbana-Champaign, USA.

APPENDICES

Appendix A

Macros of Sensitivity Analysis Module

```
Sub macro2_Click()
 'new project

   itemNum = Application.InputBox(Chr$(13) & "Enter Project Type (1, 2 or 3)", Type:=1)
   Range("T2").Value = itemNum
   itemNum = Application.InputBox(Chr$(13) & "Enter Project Scope(1, 2 or 3) ", Type:=1)
   Range("U2").Value = itemNum
   itemNum = Application.InputBox(Chr$(13) & "Enter Year of construction, e.g 1991 ", Type:=1)
   Range("V2").Value = itemNum
   itemNum = Application.InputBox(Chr$(13) & "Enter Construction Season(1=Winter, 2=Summer or 3=Fall) ", Type:=1)
   Range("W2").Value = itemNum
   itemNum = Application.InputBox(Chr$(13) & "Enter Location (1=St.John's, 2=St.John's suberb or 3=Avalon Reagion) ",
Type:=1)
   Range("X2").Value = itemNum
   itemNum = Application.InputBox(Chr$(13) & "Enter Duration in Months ", Type:=1)
   Range("Y2").Value = itemNum
   itemNum = Application.InputBox(Chr$(13) & "Enter Size in Km ", Type:=1)
   Range("Z2").Value = itemNum
   itemNum = Application.InputBox(Chr$(13) & "Enter Capacity, 1=2-lanes, 2=2-lanes devided " ,Type:=1)
   Range("AA2").Value = itemNum
   itemNum = Application.InputBox(Chr$(13) & "Is there any waterbody 0=No, 1=Yes ", Type:=1)
   Range("AB2").Value = itemNum
   itemNum = Application.InputBox(Chr$(13) & "Enter Soil Classes according to AGR 0 to 9 ", Type:=1)
   Range("AC2").Value = itemNum
   X = MsgBox("NN Cost Estimate is   $" & Int(Range("Ae2").Value * 1000000) & "  Dollars", vbOKOnly)
   X = MsgBox("Average of 20 Scenarios   $" & Int(Range("Ag2").Value * 1000000) & "  Dollars", vbOKOnly)
   X = MsgBox("Standerd Devision of 20 Scenarios = " & (Range("Ah2").Value), vbOKOnly)

End Sub

Sub Macro3()
   Range("AH132:AJ152").Select
   ActiveSheet.ChartObjects.Add(1400, 1930, 437, 281).Select
   Application.CutCopyMode = False
   ActiveChart.ChartWizard Source:=Range("AH132:AJ152"), Gallery:= _
      xlLine, Format:=2, PlotBy:=xlColumns, CategoryLabels:=1, _
      SeriesLabels:=1, HasLegend:=1, Title:="", CategoryTitle:= _
      "20 Random Cases", ValueTitle:="Generated Cost in Million $", _
      ExtraTitle:=""
   ActiveWindow.SmallScroll Down:=4
   Selection.Width = 648.75
   ActiveWindow.SmallScroll ToRight:=-5
End Sub
```

Appendix B

Macros of Model Adaptation Module

```
Sub Macro1()
    Worksheets("New-1 Output").Activate
    Application.Goto reference:="unscaled"
    Selection.End(xlDown).Select
    ActiveCell.Offset(rowOffset:=1, columnOffset:=0).Activate
    Selection.EntireRow.Insert
    ActiveCell.FormulaR1C1 = "=R[-1]C+1"
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    item1 = Application.InputBox(Chr$(13) & "Enter Project Type(1=Bridge; 2=Highway):", Type:=1)
    ActiveCell.Value = item1
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    item2 = Application.InputBox(Chr$(13) & "Enter the Scope for Project (1=New; 2=Rehab.; 3=Other):", Type:=1)
    ActiveCell.Value = item2
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    item3 = Application.InputBox(Chr$(13) & "Enter Year of Construction for the Project(e.g., 1991):", Type:=1)
    ActiveCell.Value = item3
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    item4 = Application.InputBox(Chr$(13) & "Enter the Construction Season (1=Winter; 2=Summer; 3=Fall): ", Type:=1)
    ActiveCell.Value = item4
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    item5 = Application.InputBox(Chr$(13) & "Enter Project Location (1=St. John's; 2=Suburbs; 3=Avalon):", Type:=1)
    ActiveCell.Value = item5
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    item6 = Application.InputBox(Chr$(13) & "Enter the Contract's Duration (months): ", Type:=1)
    ActiveCell.Value = item6
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    item7 = Application.InputBox(Chr$(13) & "Enter Project length in kilometers: ", Type:=1)
    ActiveCell.Value = item7
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    item8 = Application.InputBox(Chr$(13) & "Enter the Highway Capacity (1=2-Lanes; 2=2-Lanes divided): ", Type:=1)
    ActiveCell.Value = item8
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    item9 = Application.InputBox(Chr$(13) & "Is there any crossing water-bodies (0=No; 1=Yes)? ", Type:=1)
    ActiveCell.Value = item9
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    item10 = Application.InputBox(Chr$(13) & "Enter Soil classification according to AGR. Canada (0 to 9): ", Type:=1)
    ActiveCell.Value = item10
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    item11 = Application.InputBox(Chr$(13) & "Enter Actual Cost in Million $: ", Type:=1)
    ActiveCell.Value = item11

    Application.Goto reference:="scaled"
    Selection.End(xlDown).Select
    ActiveCell.Offset(rowOffset:=1, columnOffset:=0).Activate
    Selection.EntireRow.Insert
    ActiveCell.FormulaR1C1 = "=R[-1]C+1"
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    Range("B" & Trim(Str(ActiveCell.Row - 1)) & ":L" & Trim(Str(ActiveCell.Row - 1))).Select
    Selection.Copy
    ActiveCell.Offset(rowOffset:=1, columnOffset:=0).Activate
    ActiveSheet.Paste
    Application.CutCopyMode = False

    Application.Goto reference:="outofhidden"
    Selection.End(xlDown).Select
    ActiveCell.Offset(rowOffset:=1, columnOffset:=0).Activate
    Selection.EntireRow.Insert
    ActiveCell.FormulaR1C1 = "=R[-1]C+1"
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    Range("F" & Trim(Str(ActiveCell.Row - 1)) & ":K" & Trim(Str(ActiveCell.Row - 1))).Select
```

```
    Selection.Copy
    ActiveCell.Offset(rowOffset:=1, columnOffset:=0).Activate
    ActiveSheet.Paste
    Application.CutCopyMode = False

    Application.Goto reference:="nnoutput"
    Selection.End(xlDown).Select
    ActiveCell.Offset(rowOffset:=1, columnOffset:=0).Activate
    Selection.EntireRow.Insert
    ActiveCell.FormulaR1C1 = "=R[-1]C+1"
    ActiveCell.Offset(rowOffset:=0, columnOffset:=1).Activate
    Range("H" & Trim(Str(ActiveCell.Row - 1)) & ":L" & Trim(Str(ActiveCell.Row - 1))).Select
    Selection.Copy
    ActiveCell.Offset(rowOffset:=1, columnOffset:=0).Activate
    ActiveSheet.Paste
    Application.CutCopyMode = False
    ActiveCell.Offset(rowOffset:=0, columnOffset:=2).Activate
    Application.SendKeys "{ESC}^{HOME}", True
    Range("M1").Select
    Selection.End(xlDown).Select
    Selection.Copy
    ActiveCell.Offset(1, 0).Activate
    ActiveSheet.Paste
    Range("N1").Select
    Selection.End(xlDown).Select
    Selection.Copy
    ActiveCell.Offset(1, 0).Activate
    ActiveSheet.Paste
    Application.SendKeys "^{HOME}", True
    Worksheets("Main").Activate
End Sub
'

Sub macro2()
    Worksheets("New-1 Output").Activate
    Range("A1").Activate
    Application.SendKeys "%do", True
    Application.SendKeys "^{HOME}", True
    'Worksheets("Main").Activate
End Sub


Sub Macro3()
    Worksheets("New-1 Output").Activate
    Application.ExecuteExcel4Macro String:= _
        "[SOLVER.XLA]SOLVER!SOLVER.OK(!target1,2,0,)"
    Application.ExecuteExcel4Macro String:= _
        "[SOLVER.XLA]SOLVER!SOLVER.RESET()"
    Application.ExecuteExcel4Macro String:= _
        "[SOLVER.XLA]SOLVER!SOLVER.ADD(!train,1,""=2"")"
    Application.ExecuteExcel4Macro String:= _
        "[SOLVER.XLA]SOLVER!SOLVER.ADD(!test,1,""=5"")"
    Application.ExecuteExcel4Macro String:= _
        "[SOLVER.XLA]SOLVER!SOLVER.OK(!target1,2,0,(!moved_weights))"
    Application.ExecuteExcel4Macro String:= _
        "[SOLVER.XLA]SOLVER!SOLVER.SOLVE()"
    Worksheets("Main").Activate
End Sub
Sub Macro4()
    Worksheets("New-1 Output").Activate
    Application.ExecuteExcel4Macro String:= _
        "[SOLVER.XLA]SOLVER!SOLVER.OK(!target1,2,0,)"
    Application.ExecuteExcel4Macro String:= _
        "[SOLVER.XLA]SOLVER!SOLVER.RESET()"
    Application.ExecuteExcel4Macro String:= _
        "[SOLVER.XLA]SOLVER!SOLVER.ADD(!newcases,1,""=5"")"
```

```
    Application.ExecuteExcel4Macro String:= _
        "[SOLVER.XLA]SOLVER!SOLVER.OK(!target2,2,0,(!moved_weights))"
    Application.ExecuteExcel4Macro String:= _
        "[SOLVER.XLA]SOLVER!SOLVER.SOLVE()"
    Worksheets("Main").Activate
End Sub

Sub macro5_Click()
  'new project
    Worksheets("New-1 Output").Activate
    itemNum = Application.InputBox(Chr$(13) & "Enter Project Type (1, 2 or 3)", Type:=1)
    Range("T2").Value = itemNum
    itemNum = Application.InputBox(Chr$(13) & "Enter Project Scope(1, 2 or 3) ", Type:=1)
    Range("U2").Value = itemNum
    itemNum = Application.InputBox(Chr$(13) & "Enter Year of construction, e.g 1991 ", Type:=1)
    Range("V2").Value = itemNum
    itemNum = Application.InputBox(Chr$(13) & "Enter Construction Season(1=Winter, 2=Summer or 3=Fall) ", Type:=1)
    Range("W2").Value = itemNum
    itemNum = Application.InputBox(Chr$(13) & "Enter Location (1=St.John's, 2=St.John's suberb or 3=Avalon Reagion) ",
Type:=1)
    Range("X2").Value = itemNum
    itemNum = Application.InputBox(Chr$(13) & "Enter Duration in Months ", Type:=1)
    Range("Y2").Value = itemNum
    itemNum = Application.InputBox(Chr$(13) & "Enter Size in Km ", Type:=1)
    Range("Z2").Value = itemNum
    itemNum = Application.InputBox(Chr$(13) & "Enter Capacity, 1=2-lanes, 2=2-lanes devided ", Type:=1)
    Range("AA2").Value = itemNum
    itemNum = Application.InputBox(Chr$(13) & "Is there any waterbody 0=No, 1=Yes ", Type:=1)
    Range("AB2").Value = itemNum
    itemNum = Application.InputBox(Chr$(13) & "Enter Soil Classes according to AGR 0 to 9 ", Type:=1)
    Range("AC2").Value = itemNum
    X = MsgBox("NN Cost Estimate is   $" & Int(Range("Ae2").Value * 1000000) & "  Dollars", vbOKOnly)
    Worksheets("Main").Activate
End Sub

Sub Macro9() 'delete
    Worksheets("New-1 Output").Activate
    Range("A1").Select
    delitem = Application.InputBox(Chr$(13) & "Enter Project Number to Delete( > 18):", Type:=1)
    If delitem <= 18 Then Worksheets("Main").Activate: Exit Sub
    Application.Goto reference:="unscaled"
    ActiveCell.Offset(rowOffset:=18, columnOffset:=0).Activate
    done = 0
    Do
     ActiveCell.Offset(rowOffset:=1, columnOffset:=0).Activate
     If ActiveCell.Value = 0 Then Worksheets("Main").Activate: Exit Sub
     If ActiveCell.Value = delitem Then done = 1: Selection.EntireRow.Delete
    Loop While done = 0

    Application.Goto reference:="scaled"
    ActiveCell.Offset(rowOffset:=18, columnOffset:=0).Activate
    done = 0
    Do
     ActiveCell.Offset(rowOffset:=1, columnOffset:=0).Activate
     If ActiveCell.Value = 0 Then Worksheets("Main").Activate: Exit Sub
     If ActiveCell.Value = delitem Then done = 1: Selection.EntireRow.Delete
    Loop While done = 0

    Application.Goto reference:="outofhidden"
    ActiveCell.Offset(rowOffset:=18, columnOffset:=0).Activate
    done = 0
    Do
     ActiveCell.Offset(rowOffset:=1, columnOffset:=0).Activate
     If ActiveCell.Value = 0 Then Worksheets("Main").Activate: Exit Sub
     If ActiveCell.Value = delitem Then done = 1: Selection.EntireRow.Delete
    Loop While done = 0
```

```
    Application.Goto reference:="nnoutput"
    ActiveCell.Offset(rowOffset:=18, columnOffset:=0).Activate
    done = 0
    Do
     ActiveCell.Offset(rowOffset:=1, columnOffset:=0).Activate
     If ActiveCell.Value = 0 Then Worksheets("Main").Activate: Exit Sub
     If ActiveCell.Value = delitem Then done = 1: Selection.EntireRow.Delete
    Loop While done = 0

    Application.SendKeys "^{HOME}", True
    Worksheets("Main").Activate
End Sub
```