DALHOUSIE UNIVERSITY DALTECH LIBRARY

"AUTHORITY TO DISTRIBUTE MANUSCRIPT THESIS"

Title:

Application of Modified Conditional Simulation and Artificial Neural Networks to Open Pit Optimization

The above library may make availabl̶e or authorize another library to make available individual photo/microfilm copies of this thesis without restrictions.

Name of Author: Peter Kwagyan Achireko

Signature of Author: [redacted]

Date: 5/11/98

0-612-39318-6

Application of Modified Conditional Simulation and Artificial Neural Networks to Open Pit Optimization

by

Peter Kwagyan Achireko

A Thesis Submitted to the
Faculty of Engineering
in Partial Fulfilment of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: Mining Engineering

APPROVED:

[redacted]

Dr. D.H. Zou, Co-Supervisor

[redacted]

Dr. Samuel Frimpong, Co-Supervisor

[redacted]

Dr.M.C. Rockwell

[redacted]

Dr. Gordon Fenton

[redacted]

Dr. Jozef Szymanski, External Examiner,University of Alberta

DALHOUSIE UNIVERSITY DALTECH
Halifax, Nova Scotia 1998

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS AND SYMBOLS

b	Inverse of the range *a*
BLUE	Best linear unbiased estimate
BSi	Back of side of *i* neighbour of a 3-D block in column (*j-1, k-1*)
COMECON or CMEA	Council for Mutual Economic Assistance
CPU	Central processing unit
C(r)	Covariance of TBM field
CS	Conditional simulation
CS/MFNN	Conditional simulation/multi-layer feedforward neural networks
C_{ijk}	Removal cone
$C(\underline{x}_\beta,\underline{x}_\alpha)$	Covariance matrix of the unknown data at $\underline{x}_\beta$ and known data at $\underline{x}_\alpha$
$C(\underline{x}_\alpha,\underline{x}_\gamma)$	Covariance matrix of known data $\underline{x}_\alpha$ and the weight $\underline{x}_\gamma$
c_ω	Circle of radius ω
dω	Differential length on circle c_ω
DC	Direct Costs
du	Differential length or area at the end of vector u
$\overline{E}$	Average error
E_p	Error function for a pattern *p*
E( )	Expected value
EBP	Error is backpropagated
EBV	Economic block value
EPL	Economic pit limits
f(ω)	Radial spectral density function
f_j	Threshold function acting on the weighted sum
$(H_1, H_2, H_3, ..., H_N)$	Hidden layer nodes
I	Income

IC	**Indirect Costs**
$J_0(\ )$	**Bessel function of first kind of order zero**
$[K_\alpha]$	**Covariance matrix between the known sample points**
M	**Number of harmonics**
MFNN	**Multilayer feedforward neural networks**
MRM	**Multiple regression model**
m_β, m_α	**Known means at points $\underline{x}_\beta$ and $\underline{x}_\alpha$ respectively**
$[M_\beta]$	**Covariance matrix between the unknown sample and known sample points**
M_{ij}	**Block column for a given block b_{ijk} in 2-D**
M_{ijk}	**Block column for a given block b_{ijk} in 3-D**
n	**Actual block grades**
$[n_\beta]$	**Weight matrix.**
$n_{\alpha\beta}$	**Set of coefficients.**
$(O_1, O_2, O_3, \ldots\ldots., O_N)$	**Output layer nodes**
P	**$N \times 1$ column vector of dependent variable observations**
P	**Two- or three-dimensional field to be simulated**
o_{pj}	**Actual outputs for pattern *p* on node *j***
P_{ijk}	**Pit value on block (*i,j,k*) with the neighbours in the backward direction**
$P_{Si,j\text{-}1,k}$	**Pit value of one of the nearest neighbour blocks in the column (*j-1, k*)**
$P_{SBS(Si),j\text{-}1,k\text{-}1}$	**Pit value on the block which was the optimum neighbour block in the column (*j-1,k-1*), determined during the computation of** $P_{si,j\text{-}1,k}$. $P_{i\text{-}1,\, j\pm 1,\, k}$, $P_{i,\ j\pm 1,\, k}$ and $P_{i+1,\, j\pm 1,\, k}$ **Neighbouring pit values**
Q	**Maximum tonnage of recovered metal for a given tons of ore mined**
R^n	**Dimensionality of the space**
S	**Upper limit of cluster (surface)**

Si	**Side *i* neighbour of a 3-D block in column (*j-1,k*)**
SBSi	**Side of back of side of *i* neighbour of a 3-D block in column (*j,k-1*)**
SSBSi	**Side of side of back of side of *i*. in column (*j,k-1*)**
$S_1(\omega)$	**Spectral density function of the unidimensional process**
S(x, y)	**2-D slope wall equation**
S(x, y, z)	**3-D slope wall equation**
t_{pj}	**Target outputs for pattern *p* on node *j***
T	**Tons of ore in the pit**
TBM	**Turning bands method**
u	**Direction vector**
UPL	**Ultimate pit limits**
UPD	**Ultimate pit design**
V	**Tons of ore mined**
w_{ij}	**Weight from node *i* to node *j***
$(x_1, x_2, x_3, ..., x_N)$	**Input layer nodes**
X	$N \times k$ **matrix of independent variable observations**
X	**Random function of the grade distribution at a point**
$X_{SC}(\underline{x})$	**Conditioned field**
$X_O(\underline{x})$	**Conditionally simulated field based on observed data using BLUE**
$X_S(\underline{x})$	**Unconditionally simulated field via TBM**
$X_C(\underline{x})$	**Conditionally simulated field using BLUE based on the TBM values at the observed points**
$X_O(\underline{x}_\alpha)$,	$\alpha = 1,2,........,n$ **are the known values of** X **at** $x_1, x_2, ..., x_n$
$X(\underline{x}_\beta)$	**Actual value at point** $\underline{x}_\beta$
$X^*(\underline{x}_\beta)$	**Predicted value at point** $\underline{x}_\beta$
X_i	**Estimated block grades**
X_i^*	**Estimated block grades**

$\mathbf{x}_N$	**Location vector**
X_p	**Presented input pattern**
$z_i(\eta_{Ni})$	**Assigned values from the unidimensional realizations**
θ_i	**Angle formed between a line *i* and a fixed *x* axis**
η	**Coordinate on line *I*.**
$\Gamma\mathbf{x}$	**The set of all successors of x.**
Θ	**Activation Function**
ω	**Frequency (wave number)**
σ^2	**Variance**
β	**Parameter**
ϕ_k	**Independent random angles uniformly distributed between 0 and** 2π
$[-\Omega, +\Omega]$	**Band frequency of the process**
$\Delta\omega$	**Discritization frequency**
$\Delta\omega'$	**Small frequency,**
η	**Gain term or momentum coefficient**
δ_{pj}	**Error term for pattern *p* on node *j***
β	$k \times 1$ **column vector of unknown parameters**
ε	$N \times 1$ **column vector of errors**
σ_e^2	**Variance of the errors**

ACKNOWLEDGMENTS

I would like to thank my supervisor Dr. D. H. Zou and co-supervisor Dr. Samuel Frimpong of University of Alberta for their guidance, assistance and constructive comments throughout this study and in preparation of this manuscript. I wish to express my appreciation to the members of my thesis committee, Dr. M.C. Rockwell, Dr. Gordon A. Fenton and Dr. Jozef Szymanski, external examiner, University of Alberta, for their constructive comments and criticism and contributing their time, effort and knowledge to the process of completing and defending this manuscript.

I would like to acknowledge the assistance and efforts of two random field theory and programming experts, Dr. Gordon A. Fenton of Department of Applied Mathematics, DalTech and Dr. A. F. Thompson of Geosciences & Environmental Technologies Division, Livermore California, who gave me invaluable suggestions, advice and help in setting up the program codes used in the current study. I gratefully acknowledge and appreciate the assistance given to me by Dr. Granville Ansong of St. Mary's University in modelling the mineral prices. My sincere thanks go to my fellow graduate students of the Department of Mining and Metallurgical Engineering, Robert M. Changirwa, C. Ketata, T.G. Lay, L. Li and Y. Huang for their great assistance, encouragement and companionship. The cordial relationship with the departmental secretaries Karen and Heather is highly appreciated.

Without a doubt I owe my greatest debt to my wife Martha, who was and still is a constant source of motivation and encouragement. She made the end, of what often seemed an endless road, seem near. I thank my son Akwasi for his co-operation throughout the preparation of this manuscript. To my parents, who even though illiterates but saw the need to educate all their children, I say thank you.

I thank Dr. Samuel Frimpong for offering me financial support from his National Sciences and Engineering Research Council grant. With regard to the Teaching Assistantships, I am extremely grateful to the Department of Mining and Metallurgical Engineering.

ABSTRACT

Design and optimization of open pit limits are of paramount importance because they provide information for evaluating economic potential of a mineral deposit and for developing short- and long-range mine plans. Many algorithms and their modifications have been used to design and optimize open pit limits. These algorithms have provided mine planning engineers with pertinent information in designing, optimizing and extracting ore reserves by open pit technology. However, they do not address the random field properties associated with the ore grades and reserves and commodity prices, and thus, fail to yield the truly optimized pit limits in any time horizon. Also, in mine design and valuation, commodity price forecasts are required to assess the economic viability of the project. The forecast must cover relevant period to capture the trend and volatility in prices within a mining business cycle. But mineral commodity prices are volatile in the spot and futures markets. They depict unpredictable shifts in the balance between supply and demand. Fluctuations of commodity prices in the spot markets can cause project failure if not handled with care.

1. In this study, a new algorithm, CS/MFNN, which overcomes these limitations is proposed and used to optimize open pit limits. The random field properties of the ore grade and reserves have been modelled using the modified conditional simulation based on the best linear unbiased estimation and turning bands method. Artificial neural networks are used to classify the blocks into classes based on their conditioned values. The error back propagation algorithm, in the neural networks, is used to optimize the pit limits by minimizing the desired and actual outputs error in a multilayer perceptron under the wall slope constraints. The optimized pit value obtained by this algorithm has been compared to that from the Lerchs-Grossman's algorithm using actual mine data. The results are the same, but in random multivariable states, the CS/MFNN algorithm is the most suitable for pit optimization. Comparing the Lerchs-Grossmann's and CS/MFNN algorithms, it can be said that both yield the same optimum pit value in the absence of grid blocks with zero

economic block values. However, in the presence of grid blocks with zero economic block values they may portray different pit outlines. Whereas the Lerchs-Grossmann's algorithm considers any gridded block irrespective of its economic block value in the design process, CS/MFNN algorithm does not consider gridded blocks with zero or negative economic block values unless they happen to lie above any of the gridded blocks in the positive regions delineated by the MFNN. The stochastic gold price is modelled via two main models, namely; multiple regressional model (MRM) and multilayer feedforward neural networks (MFNN) model. The MRM model is used to find the multiple regression equation. The MFNN model is used to predict the average-annual monthly high gold price and average-annual monthly low gold price, which are also independent variables in the multiple regression equation. World annual gold production, annual gold consumption, average-annual monthly high gold price, average-annual monthly low gold price, socio-politico-economic condition, interest rate and inflation rate are identified as the most important factors and/or parameters which determine world average annual gold prices. A data set on these parameters of an identified socio-politico-economic cycle is used to validate the model. Analysis of the results shows that the mineral price model predicts average-annual gold price with negligible error. The actual prices and the predicted ones are almost the same. A realistic forecasting of gold prices can be made with data from an identified socio-politico-economic cycle. The main novelty of this methodology is the solution of the randomness property associated with mineral prices using multiple regression and artificial neural network to reduce the mineral price forecasting error. This is an important contribution to mineral venture evaluation, mine planning and design.

CHAPTER 1.0

INTRODUCTION

1.1 Background

Modern society needs a supply of minerals for its growth and sustenance. Most of these minerals are obtained by means of surface and underground mining technology. Currently, surface mining accounts for a significant proportion of produced minerals. Surface mining operations generally have many advantages in terms of production equipment sizes, preproduction development period, ore recovery and labour requirements compared with underground. It is classified into open pit, strip, alluvial and in-situ mining methods [Hartman, 1987; Nilson, 1982]. Open pit mining layouts consist of concentric shells with near-ellipsoidal cross-sections. These shells decrease in size with increasing depth from the surface as illustrated in Figure 1.1 and Figure 1.2 (plan and longitudinal sections of a typical open pit layout). An open pit layout is bounded by wall slopes whose angles depend on the rock mechanics and geological characteristics of the ore body and host rocks. The challenge to mine planning engineers is to plan, design and optimize the pit layouts to minimize waste removal, ensure safety and maximize the net value of the minerals in the pit.

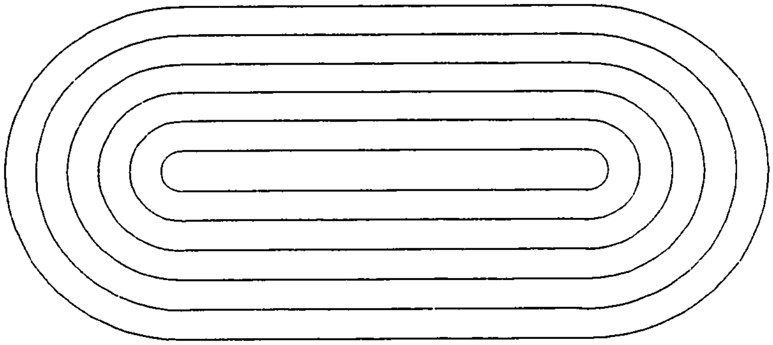


Figure 1.1 Plan View of a Typical Open Pit Layout.

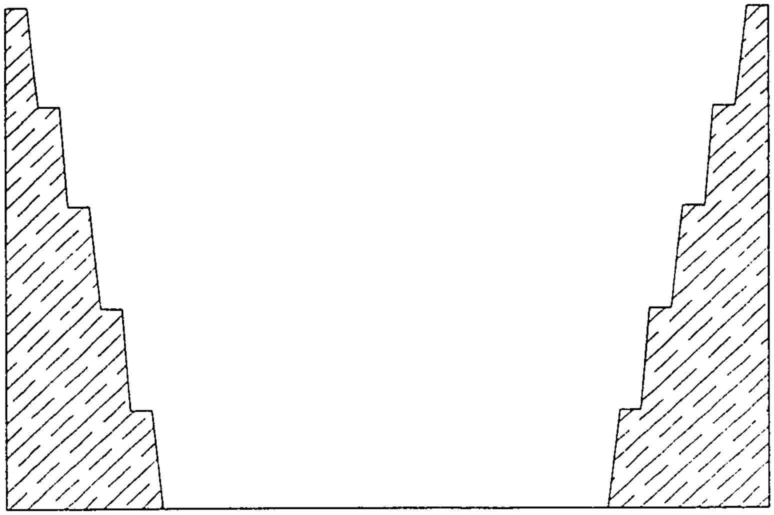


Figure 1.2 Longitudinal Section of a Typical Open Pit Layout.

Determination of optimized pit layouts is one of the most important tasks in the overall open pit mine design process, which has to be solved right at the very beginning of mine planning. These layouts must continuously be readjusted throughout the life of the mine due to changing database of geology, ore grades, etc., [Wilke, 1990]. The optimized pit limits define the size and shape of mineable reserves and the associated waste materials to be excavated based on the technical, economic and safety constraints. They also provide information for evaluating the economic potential of a mineral deposit, and for project acquisition, financing, taxation, regulation and the formulation of long-, intermediate-, and short-range mine plans. Open pit limits are also used to determine the boundaries outside which surface structures, such as, processing plants and mine offices should be located in order to avoid interruption in the long-term mine plans. Lerchs and Groomsmann (1965) published one of the most important mathematical algorithms for optimizing open pit limits based on dynamic programming and graph theory. Many algorithms have been developed to solve the problems associated with the Lerchs-Grossmann's algorithm. However, these problems which include random fields characteristics, comprehension difficulty, programming, variable slopes and roadways incorporation, and long CPU times [Achireko and Frimpong, 1996; Dowd and Onur, 1993] still persist and require continuous research for better solutions.

1.2 Problem Definition

Lerchs and Groomsmann (1965) developed mathematical algorithms for optimizing open pit layouts based on dynamic programming and graph theories, respectively, for the 2-D and 3-D geological sections under constant walls slopes constraints. The Lerchs-Grossmann's algorithm is limited because it does not address the random field properties of the mine design problem. In addition, a number of other problems have been encountered by users in the application of these algorithms to solve mine design and optimization problems. These problems include the incorporation of variable slopes and roadways constraints, long CPU times, difficulties in understanding the method, and algorithm programming concepts [Dowd and Onur, 1993]. Many algorithms, including the moving cone, linear and dynamic programming, graph theory, stimulation, parameterization and heuristic, and their modifications have been developed to solve the problems associated with the Lerchs-Grossmann's algorithm [Braticevic, 1984; Dowd and Onur, 1993; Gauthier and Gray, 1971; Kim, 1978; Koenigsberg, 1982; Lemieux, 1979; Robinson and Prenn, 1973; Shenggui and Starfield, 1985]. These methods have provided mine planning engineers with pertinent information in pit planning, design and optimization. However, none of them has been successful in defining unique optimum pit layouts to solve the mine design problems. Even though there is some success with the application of maximal flow techniques [Dowd and Onur, 1993; Giannini et al., 1991], finding the maximal flow through a network has similar disadvantages to the Lerchs-Grossmann algorithm.

Unique optimum pit layouts must be carried with a recognition of the random field properties of the mine design problem. Optimized pit layouts of a typical open pit mine are the solutions of a problem in random multivariable states. The stochastic processes governing these states significantly affect the optimized pit limits in any given time. The most important state variables include ore reserves and grade, capital and operating costs and commodity prices. Any algorithm that neglects these stochastic processes governing the state variables will yield sub-optimal pit layouts. The randomness associated with the distribution of the ore grades and reserves, have not been taken into consideration in all the algorithms. The omission of these random field properties in the above algorithms poses a limitation on the pit layouts definition, the economic potential of the mineral deposit, and the long-, intermediate-, and short-range mine plans. This might lead to poor investment, inaccurate mine design, development and production decisions.

In mining practice, the pit slopes are determined by the rock mechanics and geological characteristics of the ore body and host rocks. These conditions may result in variable pit slopes all over the entire mine which make the results from algorithms under constant slope constraints unrepresentative of actual field conditions. The Lerchs-Grossmann's 3D algorithm also presents a tight choice between a true optimizing algorithm characterized by large computing requirements and a non-optimizing algorithm with lower computing demands and thus, fails to yield a truly optimized pit [Kennedy, 1990]. The solution of the pit design and optimization problem using these algorithms involves the inversion of matrices with large dimensions and long iterative processes which result in long computing times. Changes in the database and information also require complete rerun of these algorithms which can be long and tedious for the mine production engineers who require just-in-time results for executing production plans based on these results. Commodity price volatility in the markets must also be factored in the mine design and optimization on a continuous basis. Efficient pit design and optimization algorithms must be capable of handling these price volatilities with just-in-time results for executing mine production plans.

Even though it is not necessary for engineers designing a pit to have a detailed knowledge of the mathematics involved in the Lerchs-Grossmann algorithm that is embodied in a verified software, yet its complexity is often advanced as a reason for not using the algorithm [Dowd and Onur, 1993]. The comprehension of the underlying theory for the implementation of the algorithm will help engineers and researchers to factor in and quantify the effects of various combinations of identifiable variables which are regularly encountered by mine engineers, mineral venture planners, evaluators and investors in the evaluation and assessment of economic potential of a mineral deposit.

1.3 Objectives of the Study

The main objectives of this study are to: (i) develop the mathematical and computer models of the conditional simulation/multilayer feedforward neural network (CS/MFNN) algorithm for pit design and optimization; (ii) develop mineral price model, MRM-MFNN, for economic block values calculation in the CS/MFNN algorithm; (iii) verify and validate the CS/MFNN algorithm using actual gold mine data; and (iv) compare the results from the CS/MFNN to that from the Lerchs-Grossmann's algorithm.

In this research, the author develops and applies the CS/MFNN algorithm to solve the problems of random fields, long computing time and variable slopes in open pit design and optimization. In order to ensure that the random property associated with ore grade distribution are accounted for at a minimum cost during the pit optimization, CS model is developed to assign grades and economic block values to all the gridded blocks with few known samples in the field. In this algorithm, the random field properties of the ore grade and reserves are modelled using the conditional simulation (CS) based on the best linear unbiased estimation (BLUE) and the turning bands method (TBM) technique.

In this study, only gridded-blocks' grades are conditionally simulated but not the pit. The author wishes to make it crystal clear that the pit can also be conditionally simulated and a probability distribution constructed using a series of optimum pit values obtained from the pits. In this case, every conditional simulation realization of the gridded-blocks' grades should be used in optimizing the pit, rather than the average values of the number of realizations as done in this study. This would give the probability and confidence intervals of optimum pit values for the evaluation of the economic potential of the ore deposit in question.

After assignment of grades and economic block values to all the gridded blocks, multilayer feedforward neural networks (MFNN) are used to classify the conditioned blocks into classes based on their economic values. A search algorithm, PITSEARCH, developed by the author is then used to search for the optimized pit layouts and the resulting net pit value under the variable slope walls constraints. MRM-MFNN model is used to forecast the volatile mineral price which is used by CS/MFNN model to calculate the economic block values. In the CS/MFNN algorithm, MFNN model is used to search for the optimum pit in few steps.

1.4 Scope and Limits of the Study

This study deals with the design and optimization of open pit using the CS/MFNN algorithm. It is concerned with the definition of the layouts that maximizes the net value, at any given time, of an open pit section under technical, operational, economic and safety constraints. First, a review of the significant literature is included. The review is to provide an analysis of the evolution of optimal open-pit limit design methodologies and their attendant problems, and to identify areas that require further research work. which led to this research.

During this research, the theory, mathematical and computer models of the CS/MFNN, with illustrative examples, are developed and verified using data in 2-D and 3-D geological sections. The computer models are validated using data from a gold mine. The Lerchs-Grossmann's algorithm is also used to solve the Star Gold open pit design and optimization problem. A comparative analysis has been carried out on the results from the CS/MFNN and the Lerchs-Grossmann algorithms.

In the MRM-MFNN price model, inflation and interest rates, socio-politico-economic factors are assumed to influence mineral price but they are embodied in the other factors (such as world annual gold production, annual gold consumption, average-annual monthly low gold prices and average-annual monthly high gold prices), that determine mineral prices used for the price forecast as there are no world parameters for defining them. Also the gold loan transaction in which gold holders lend their gold for a fee is assumed to be known with certainty. These loan transactions are used primarily by gold producers who earn instant cash flow by selling the borrowed gold and repaying the loan at some point in the future out of their gold mine production. The markets usually reacts negatively to news of large loan agreements because more gold is added to the market. Even though these restrictions and assumptions make the underlying information structure easy to handle, they place some limitations on the study.

1.5 Significance of the Study

The CS/MFNN algorithm can be used to evaluate mineral projects involving open pit operations to provide clear estimation of randomly distributed grades and reserves and volatile mineral prices. The possibility of recouping money invested in any mining venture within the shortest possible time is of prime concern to both engineers and investors. Mineral price fluctuation is one of the hurdles to overcome in order to arrive at fair estimation of the return of the investment. Hence, MRM-MFNN price model for predicting mineral prices and CS/MFNN algorithm for optimizing open pits will assist engineers to provide accurate mine planning and design alternatives to boost investors and other stakeholders confidence.

The CS model is appropriate for block modelling in a random field with few data. The random field properties of the ore reserves and grade are modelled using conditional simulation. The CS/MFNN algorithm has significant economic potential as it can bring savings in drilling cost, geological and technological difficulties in obtaining many data to define a potential ore deposit. Existing evaluation models use other stochastic and/or probabilistic concepts to model the mineral price. Such methods demand the individual factors probability distribution, which is almost always an approximation of the actual distribution. CS/MFNN helps in selective mining of profitable blocks since it eliminates all the negative regions which are not economically viable for mining. The blocks with zero economic values are left behind. This will enable engineers to plan well to leave those blocks with zero E.B.Vs and maintain proper production scheduling and planning. This will increase the profit margin of the company. The CS/MFNN algorithm overcomes the problem of overlapping zones and search patterns for optimum pit value in the floating cone algorithm. It obviates the task of both forward and backward passes in the Lerchs-Grossmann's algorithm, delineates the positive regions and incorporates the technical constraints to design the optimum pit. It is applicable to any slope wall constraint in both

2-D and 3-D design. No smoothing is needed. In comparison, smoothing is essential for the floating cone and Lerchs-Grossmann methods in design of slopes in multiples of blocks.

The problem of fluctuating mineral prices in mineral appraisal is solved by the MRM-MFNN model, which uses the most important attribute of multilayer feedforward network, the ability to learn a mapping of any complexity, together with multiple regression model to predict the volatile mineral price. This methodology takes all the factors which influence mineral price simultaneously into consideration in forecasting the price in any time horizon. Also, the MFNN model will require much less time. The minimum CPU time is cost saving and/or profit to the mineral industry. MFNN open pit optimization model is rigorous and much easier to understand than the existing algorithms.

1.6 Research Methodology

The development and analysis of CS/MFNN algorithm for open pit optimization are based on numerous methodological procedures. The analytical survey of the literature on open pit optimization, outlining the evolution and limitations of open pit optimization algorithms, constitutes an essential part of the research and gives the main background for the study.

In the development and analysis of the CS/MFNN algorithm, a model-building approach with specific assumptions is used in the evaluation and assessment of the economic potential of the mineral deposit under consideration. Knowledge in artificial neural networks, random-field mathematics, statistics, mining engineering and mineral economics is used in the CS/MFNN algorithm and MRM-MFNN price model to express the several variables. FORTRAN 77 on the HP main frame is the computer programming language used to translate the mathematical models into computer programs for experimentation and convenient, detailed analysis for the desired objectives. Also neural networks software is used to develop the MFNN model for the pit optimization and mineral price forecast.

The CS/MFNN algorithm is tested and validated using the Star Gold Project data. The same data is also analyzed using the Lerch-Grossmann algorithm as a bench mark to make the proposed algorithm comprehensible and acceptable to potential users in the mineral industry. A thorough analysis of the results from the two algorithms are done to see how these algorithms compare with each other. MRM-MFNN mineral price model is tested with historical data from 1980 to 1994.

1.7 Structure of the Study

Chapter 1 elucidates the background, the concepts and problems associated with the existing open pit optimization techniques. Also included in Chapter 1 are the objectives, significance and the research methodology of the study of the CS/MFNN algorithm. Chapter 2 contains the analytical survey of the literature on open pit optimization algorithms and their limitations. Also, Chapter 2 provides the theory and fundamentals of neural networks, and its application to open pit optimization and mineral price forecasting. Chapter 3 deals with the mathematical modelling of Lerchs-Grossmann and CS/MFNN algorithms. The 2-D and 3-D models of these two algorithms are developed. The theory and modelling of modified conditional simulation, MFNN and MRM-MFNN are also provided in Chapter 3. Chapter 4 deals with the solution procedures and the experiments designed to implement and validate them. The solution algorithms in Chapter 3 and the flow charts of the computer programs used to solve the problems, are presented and described. Large scale validation and experimentation with Star Gold Project data is presented in Chapter 5. The experimental design and procedure, the description of the deposit, nature of ore, extraction technology and processing are included in this chapter. The discussion and analysis of the validation results are in Chapter 6. Chapter 7 has all the conclusions and the recommendations for further research works arising from this research study. All the references are given at the end of the report as sources of various citations in this study. Programs developed in this study that may be helpful in the use of this document for academic and industrial purposes and data are given in appendices.

CHAPTER 2.0

ANALYTICAL SURVEY OF THE LITERATURE

The determination of the optimum pit limit of a mine is considered to be a fundamental problem in open pit mine planning and design because it provides essential information for evaluating the economic potential of a mineral deposit, and formulating long, medium, and short range mine plans. The ultimate pit limits define the size and shape of an open pit at the end of its life based on the technical, economic and ground stability constraints. They also determine the extent of mineable reserves and waste materials to be moved in the mining process. Pit limits on the surface mark the boundaries for locating surface structures, such as processing plants and mine offices. The pit limits would normally delineate the limiting boundary beyond which the open pit mining of a given deposit will be uneconomic. As such the pit limits are commonly referred to as the Economic Pit Limits (EPL), Ultimate Pit Limits (UPL) or Ultimate Pit Design (UPD). In general the manual, computerized and combinations of manual and computerized approaches are used for the design of open pit limits.

A number of mathematical techniques have been proposed to solve this problem, most of them posing considerable computational difficulties. Other problems include the inability to incorporate these factors: (i) the random field properties associated with ore reserves and grades and the mineral prices, (ii) the amount of ore and waste to be mined each year, (iii) the specific blocks to be mined each year, (iv) the mine life which defines the expected time horizon of the operation and (v) the ultimate pit limits which delineate the total reserve to be mined and the final pit shape at the end of the operation. The more elaborate methods use manual, stimulation, linear programming, dynamic programming, graph theory, heuristic and parameterization approaches to the open pit limit design. These methods are limited in their respective approaches to defining the optimum pit limit. Among them, the four rigorous optimizing techniques, that have mathematical proofs, are graph theory [Lerchs and Grossmann, 1965], dynamic programming [Lerchs and Grossmann, 1965], linear programming [Meyer, 1969] and network flow [Johnson, 1968]. Heuristic algorithms [Lemieux, 1968; Marino and Slama, 1972; Phillips, 1972; Korobov, 1974] lack rigorous mathematical proof. In the following sections, various algorithms for open pit design and optimization and their limitations have been described.

2.1 The Manual Approach to Open Pit Design

Manual approach to open pit design is basically a trial and error method in which the analyst uses subjective and objective analyses to define the pit limits. This approach requires (i) vertical sections showing clearly the ore boundaries, the grade distribution within the ore, the overburden and the waste rock portions; (ii) plans for each proposed mine level showing corresponding ore and waste details as in (i); (iii) allowable maximum slope angles for the various rock types; (iv) minimum width at the proposed pit bottom; and (v) relevant stripping ratio curves showing the variation of the stripping ratio with the ore grades and possible selling prices.

The overall, incremental, periodic and break-even stripping ratios are taken into consideration in the design of the pit. These ratios depict the relative proportion of waste material to ore to be mined and provide the necessary profit information. Stripping curves, the behaviour of stripping ratio with grade variation, are normally constructed to cover possible grade ranges at the pit limits as a pre-requisite for a manual pit design exercise [Pana and Davey, 1973].

With the aid of the stripping ratio curves and the cross-sections showing the ore boundaries and grades, the pit limits will be located on each section as follows: (i) Note the grades around the two ends of the ore on a given level and work out a weighted average grade to include the ore up to the top of the grid blocks; (ii) Construct the stripping ratio curve and note the allowable break-even stripping ratio; (iii) Construct trial slope lines such that the ratio of the intercept along waste to that along ore correspond to the break-even stripping ratio; (iv) The procedures in (iii) are repeated for other sections of the deposit; (v) When the pit limit has been located on each vertical section, the intersections are then transferred to the level plans. The total ore reserve and waste volumes and tonnages are determined using the planimetry method.

Even though this trial-and-error method helps in the development of the basic skills required for the geometrical manipulations involved in such areas as haul road design and phase plan development, which is needed with fully computerized pit design approaches, it has some limitations. The manual approach is very time consuming. The assumption that the total recovery during mineral processing is a constant in the derivation of the stripping curves would not necessarily hold true for different ore grades. In fact the effective recovery will invariably depend on the grade of the mill feed. This method can only be used for small and/or geologically simple mineral deposit.

2.2 The Moving Cone Algorithm

The moving cone algorithm is one of the algorithms for optimizing open pit limits [Pana, 1965; Williams, 1970; Lemieux, 1968]. The 2-D moving cone algorithm uses the following optimization criteria, deduced from the definition of the open pit design problem, namely: (i) maximization of total pit economic value, (ii) maximization of value per tonne of saleable product, (iii) maximization of the mine's life, provided the value per tonne does not fall below a certain figure, and (iv) maximization of the metal content within the pit [Wright, 1990]. The maximization of the total economic value is by far the most common criterion of optimization in open pit mine design. Within an allowable slope angle this algorithm ensures that before mining any block, all the blocks above must be removed. The minimum removal cone on a block is the cone bounded by the maximum allowable slope angles in all directions, from the block up to the ground surface. The cone on any block is dependent upon the different slope angles for the different materials overlaying the block in the different directions up to the ground surface.

The basic element for the optimization process is the minimum removal cone. The algorithm can be stated as follows: (i) Start from the surface and search for ore blocks with positive economic block value; (ii) Construct the minimum removal cones on such ore blocks; (iii) If the sum of the economic block values (EBV) of all blocks contained in a given cone, including the ore block in question, is positive, consider the cone removed; (iv) Continue the search until all the ore blocks in the block model have been examined; (v) The ultimate pit is formed by the shape left after the removal of all positive valued cones. The technique is an optimization technique, and as such rather costly to obtain the ultimate pit limit. Furthermore, the technique can miss the optimum pit limit under certain unusual conditions, because it cannot recognize a joint contribution by two ore blocks that are laterally some distance apart. Lemieux (1968) stated that his heuristic algorithm overcame this shortcoming of the moving cone technique. However, the geometrical requirements of slope stability in combination with certain ore blocks, can lead to situations where the technique will fail to yield the pit with the maximum value [Wright, 1990]. It does not solve the problem of overlapping blocks. This problem arises from the fact that if the blocks are considered separately with their individual cones, the sign of the cones may be negative. However, if such blocks are considered together with their cones combined, a positive-valued pit can be obtained.

In the 3-D positive moving cone technique, the search commences at the north-west corner(top left) of the grid block model and proceeds from west to east (left to right) along each level. All blocks with positive value on the first level are examined, before proceeding to the second level and then to the third level until the utmost pit depth. Finally the 3-D block representation of the optimum pit is a combination of the optimum 2-D cross-sections and longitudinal sections. Smoothing is employed to fit the sections into 3-D pits after the optimum pit limits on 2-D sections have already been designed.

2.3 Smoothing Algorithms

Two of the algorithms which are normally employed in smoothing to fit the sections in 3-D moving cone pits are the dynamic programming algorithm by Johnson and Sharp (1971) and the dynamic path approach by Wright (1987). The dynamic programming algorithm is a systematic approach for aligning optimum pit limits on 2-D sections into 3-D pits. The approach is in effect a repeated application of the 2-D algorithm, first of all to the cross-sections and finally at right angles to the original sections. The first stage of the calculation is to compute the optimum level outlines for each section, and thus, estimate the sum of the economic values of blocks within the optimum pit outline on a section, given that one must mine down to the level of location of the block under consideration. This is done using the 2-D algorithm and treating each level in turn as the deepest level of the section. Finally the 3-D block representation of the optimum mine is presented as a combination of the 2-D cross-sections and longitudinal sections. The algorithm has the advantage of taking care of some of the smoothing problems - provided the outlines on the consecutive sections are not very much offset from each other. For simple elongated mineral deposits, the algorithm does yield optimum pits. However, the dynamic programming does not guarantee optimum 3-D pit design and does not completely solve the smoothing problem.

The dynamic path approach (Wright, 1987) guarantees optimum cross-section designs while the moving cone in 3-D automatically takes care of the boundary smoothing aspects in the pit design. The approach involves the application of the 2-D recursion formula in a forward pass from section to section while changing the direction of the analysis along the sections from one section to the next. At the end of the forward pass, the boundary blocks in the consecutive sections then form a path along which minimum removal cones are constructed in the backward pass and the union of the minimum removal cones along the dynamic path thus define the ultimate pit. With the dynamic path approach, the smoothing problem is completely solved, due to the use of minimum removal cones.

However, it cannot guarantee optimal solutions if particular attention is not paid to blocks which are brought into the pit during the backward pass [Wright, 1990].

2.4 Lerchs-Grossmann's Optimum Open Pit Mine Design

Lerchs and Grossmann (1965) have used the graph theory and dynamic programming to formulate an open pit model to define the optimum pit limit. The objective of Lerchs-Grossmann's open-pit model is to design the final pit limit of an open pit mine which maximizes the difference between the total mine value of ore extracted and the total extraction cost. The model is based on the following assumptions: (i) the type of minerals, its mine value and extraction cost are given for each point; (ii) the restrictions on the geometry are specified (surface boundaries and maximum allowable wall slopes); (iii) the objective is to maximize total profit.

Dynamic programming algorithm is used for the two-dimensional pit (or a single vertical section of a mine), and a graph theory algorithm for the general three-dimensional pit. Density functions of mine value of ore per unit volume, extraction cost per unit volume, and profit per unit volume are defined at each point of a three-dimensional space. In developing the 2-D pit model, the whole pit is divided into parallel and vertical sections and each section is considered as a 2-D pit. The technique used to determine the contour of a section consists of moving three straight lines which represent the bottom of the pit and two walls at the slope angles, and evaluating the ore and the extraction cost of materials limited by the three lines. The configuration of lines yielding the best results is then selected. Here, the angle is taken to be constant over the entire pit.

In the 3-D model a graph algorithm is applied. Here the entire pit is divided into a set of volumes defined by a 3-D grid and the objective function is to maximize the profit associated with each volume. The Lerchs-Grossmann algorithm converts the three-dimensional grid of blocks in the orebody model into a directed graph. Each block in the grid is represented by a vertex which is assigned a mass equal to the net revenue value of the corresponding block. The vertices are connected by arcs (arrows) in such a way that the connections leading from a particular vertex to the surface define the set of vertices (blocks) which must be removed if that vertex (block) is to be mined. The mathematical modelling of 2-D and 3-D Lerchs-Grossmann algorithms have been provided in Appendix A.

Lerchs-Grossmann's open-pit optimization algorithm assumes a constant slope angle over the entire pit. In mining practice the pit slopes are determined by the rock mechanics characteristics and the hydrogeological conditions of the mine. These conditions may result in variable pit slopes all over the entire mine. In the two-dimensional model, the whole pit is divided into parallel vertical sections and each section is considered as a two dimensional pit. When the optimum contour of all the vertical sections are assembled, it invariably turns out that they do not fit together. The resulting contour may be far from optimum, thus the solution yields a series of optima. This undermines the reliability and credibility of the model. The problem with the three-dimensional model is that it presents a tight choice between a true optimizing algorithm characterized by large computing requirements and a non-optimizing algorithm with lower demands on computing facilities [Carlson, et al., 1966].

2.5 The Dynamic Programming Algorithm by Koenigsberg

Koenigsberg (1982) and Wilke and Wright (1984) have succeeded in applying dynamic programming directly to solve the 3-D pit design problem. In contrast to the 2-D case in which a given block can have nearest neighbours from only one column in the backward direction, in the 3-D case a block can have neighbours from four columns. These columns were designed by Koenigsberg as follows: ***Si*** in column ***(j-1,k)*** represents side of ***i***, ***BSi*** in column ***(j-1, k-1)*** represents back of side of ***i***, ***SBSi*** in column ***(j,k-1)*** represents side of back of side of ***i***, ***SSBSi*** in column ***(j,k-1)*** represents side of side of back of side of ***i***. The required slope angle, 1:1 in this case, must be satisfied with respect to each of the neighbouring columns. Figure 2.1 shows the 12 nearest neighbours to block $\mathbf{b}_{ijk}$ in the backward direction. With the neighbours in the backward direction identified, the pit value, P_{ijk}, on the block (i,j,k) is the sum of block column value for a given block $\mathbf{b}_{ijk}$ and the maximum pit value on a block among the neighbouring blocks.

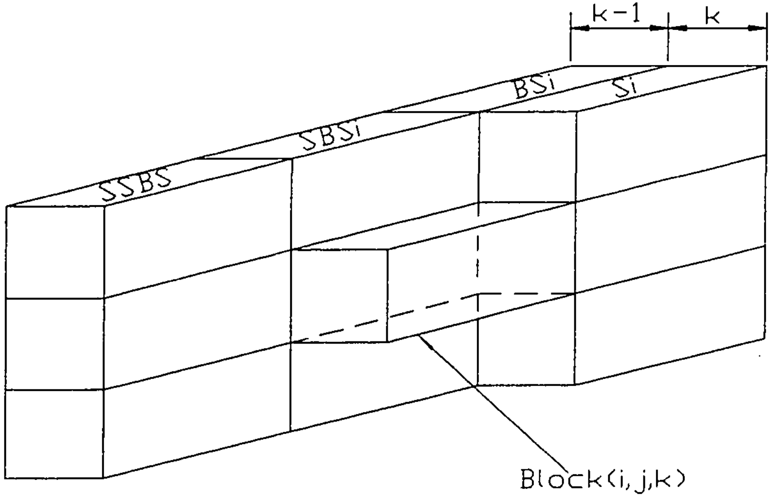


Figure 2.1 The 12 Nearest Neighbours to Block b_{ijk} in the Backward Direction.

P_{ijk} is the optimum value of the pit with the block b_{ijk} as the last block to be analyzed. This is the optimum value on the block b_{ijk}. M_{ijk} is the block column for a given block b_{ijk}. This is computed in the same manner as for M_{ij} in the 2-D case. $P_{Si,j-1,k}$ is the pit value of one of the nearest neighbour blocks in the column $(j-1, k)$. $P_{SBS(Si),j-1,k-1}$ is the pit value on the block which was the optimum neighbour block in the column $(j-1,k-1)$, determined during the computation of $P_{si,j-1,k}$. $P_{BSi,j-1,k-1}$ is the pit value on a block in a column $(j-1,k-1)$ to be selected as the optimum neighbour block, in the event the block $b_{SBS(Si),j-1,k-1}$ turns out to be incompatible to both b_{ijk} and block $b_{BSi,j,k}$. The other terms in the recursion formula are interpreted in a similar manner. The negative terms are corrections. These ensures that the neighbours eventually selected as the optimum neighbours are indeed compatible to each other, and to the block in question, with respect to the slope angle restrictions.

The problem with this algorithm is that it degenerates due to the use of a 2-D increment, M_{ijk}, for the progressive widening of the ultimate pit. This may lead to situations in which blocks selected from the four neighbouring columns end up being incompatible to each other in terms of slope requirements. Further, Shenggui and Starfield (1985) have shown that Koenigsberg algorithm is actually over constrained, and, in some cases, will actually miss the optimum ultimate pit.

2.6 3-D Dynamic Programming Algorithm by Wilke and Wright

The problem of degeneration encountered with the 3-D algorithm by Koenigsberg was overcome by Wilke and Wright (1984) with the use of 3-D increments in the form of minimum removal cones over each block. This necessitates a new formulation of the recursion formula for the block pit value, P_{ijk}. However, the basic procedure of the 2-D dynamic programming case is preserved and remains valid; namely, the pit value, P_{ijk}, on the block, b_{ijk}, can be calculated from the value of the removal cone over and including the block b_{ijk}, added to the best value neighbouring pit compatible with the block b_{ijk}. It must, however, be ensured that no block value is counted more than once. Thus, all the block values, M_{ijk}, which are contained in both the removal cone, C_{ijk}, and in the value of the respective neighbouring pits $P_{i-1,\ j\pm 1,\ k}$, $P_{i,\ j\pm 1,\ k}$ and $P_{i+1,\ j\pm 1,\ k}$, must be deducted from the neighbouring pit values.

2.7 Genetic Algorithms for Open Pit Design and Scheduling

Denby and Schofield (1994) presented a technique based on a self-learning genetic algorithm which provides, simultaneously, a combined pit limit and extraction schedule that aims to maximize the net present value (NPV) of the ore in the pit. In this algorithm, the value of a pit is calculated based on the value of each block when it is mined. The optimization procedures are as follows: (i) generation of random pit population (number of pits) with a higher average value; (ii) assessment of pit fitness value whose calculation is independent of the main optimization process for each of the schedules in the population; (iii) reproduction of pit population during which each schedule either survives to the next generation or is removed altogether. These procedures produce a population with higher average value; (iv) crossover of pits during which selected schedules are randomly combined in pairs on a probabilistic basis. Applying crossover operator to a new population produces a new population of pits combining areas (features) of existing pits to create both higher and lower value pits; (v) mutation and normalization of pits. Mutation is performed on probabilistic basis, approximately 0.1% of the cells in the schedule being modified in a random manner to help maintain genetic diversity and prevents the system from converging to a false optimum. Normalizing functions are then applied to new population. This changes the pits generated into feasible scheduled pits by the application of a number of constraint functions. After iterating through a number of generations an optimum pit and schedule is produced simultaneously. The system slows down gradually as the size of the problem increases, but if this can be minimized for much larger problems, a practical system should be feasible. Scaling-up the system to cope with realistic problems requires further investigation.

2.8 Parameterization of Optimal Design of an Open Pit

Francois-Bongarcon and Guibal (1983) have described the basic principles for parameterizing reserves and their application to open-pit design. The strength of the methodology proposed is based upon the acceptance of the idea that a complete family of optima might exist within a set of technically optimum pit. The process consists of determining a spectrum of technically optimum pits, each containing the maximum tonnage of recovered metal (*Q*) for a given tons of ore mined (*V*) and tons of ore in the pit (*T*). There are usually many different feasible pits that correspond to the same size (*V*,*T*) in a deposit. Among these pits, there is at least one that contains the maximum content of metal. This pit is the technically optimal pit associated with the pair (*V*,*T*). The objective is to find the complete spectrum of optimal pits corresponding to every possible value of (*V*,*T*). Under their hypothesis, the true absolute economic optimum pit will necessarily be among this family of pre-selected pits. The variables *V*,*T*, and *Q* are attached to one point of the abstract 3-D space of points whose coordinates are the values of *Q*,*V*,*T*. So, in this space, all the feasible pits constitute a cluster of points. The best one is the one with the highest *Q* coordinate. The family of technically optimal pits is the family of pits located at the upper limit of the cluster (surface *S*). The problems associated with this method are (i) the difficulty in adapting it accurately to polymetallic deposits; (ii) the initial selection of the best pit being based only on the metal content without the full knowledge of the economic factors; (iii) the assumption that the same profit formula is valid everywhere in the deposit; and (iv) the optimal pit selected without taking time into account. Thus, the results may mislead production planning as well as cash flow analysis.

2.9 Ultimate Pit Limit Design Using Transportation Algorithm

Huttagosol and Cameron (1992) have formulated the ultimate pit limit determination as a large scale transportation problem. The formulation is based on the use of a 3-D block model to represent an orebody and surrounding materials. The solution of the problem is obtained by solving the dual system of the formulated problem by using the adapted simplex algorithm of linear programming. The first constraint of the model represents the mining sequence as defined by pit slope requirements. The problem is formulated such that each ore block (positive) must be equated to an origin node while each waste block (negative) is equivalent to a destination node of the transportation network. A route from an origin to each of its destinations must represent pit slope constraints that is, the waste block must be removed before its underlying ore blocks can be mined. The objective function here is to maximize the value of each block. The problems associated with this algorithm are: (i) the inappropriate way of assigning the unit cost to a route; and (ii) economic factors not duly considered.

2.10 Excess Scaling Algorithm

By mapping the ultimate pit limit problem to equivalent maximum flow problem, Yegulalp and Arias (1992) utilized the "Excess Scaling Algorithm" in solving the maximum flow problem to arrive at the ultimate pit limit. It has a shorter processing time compared to that of Lerch-Grossmann. However, their model does not make room for variable pit slopes which are vital in open pit design.

2.11 An Algorithm to Estimate the Optimal Evolution Pit Mine

Tolwinski and Underwood (1992) have developed an algorithm for the determination of a production schedule for an open pit mine. The algorithm is based on ideas from dynamic programming, artificial intelligence and heuristics search rules. The evolution of the mine is modeled over time as a sequential optimization problem of finding a path with a largest value in a graph. The model allows for net present value maximization and takes into account such constraints as maximum allowable pit slope angle, working space for equipment, and a uniform supply of ore for a processing plant. The working width is defined using the drop cut concept. Here a block is removed as a drop cut if all of its surrounding blocks are at the same level. When a drop cut is made, there is sufficient working space. The angle of inclination of any cone is chosen to correspond to the slope angle for the sides of the pit.

Even though this algorithm tries to include some parameters vital to open pit design, it is still unable to define them in a manner applicable to real mining situation. Definition of working width using drop cut is incompatible with mining practice. It does not incorporate bench height which is a function of the geology and the existing mining equipment and so is rarely changed during the life of the pit unless there is a significant change in economic parameters which consequently alters recoverability criteria. Further work is required to improve this algorithm. This includes techniques like aggregation-disaggregation schemes to speed up learning as well as more sophisticated data structures to make better use of computer memory. Also, other features of scheduling problem may have to be incorporated into the model to accommodate potential users.

2.12 A Graph Theory Algorithm by Kim and Zhao

Zhao and Kim (1992) used an algorithm derived from graph theory for optimum ultimate pit limit design. The algorithm maximizes the total undiscounted net profit for a 3-D block mine model. This algorithm formulates the 3-D block model into a directed graph consisting of many trees. The vertices in the graph are equated to the blocks in the mine model and the imposed arcs in the graph represent pits slope constraints. The algorithm determines a feasible contour of a pit with maximum mass, thus achieving the optimal limit. This algorithm weakly incorporates the pit slope constraints. It uses the value of the block (either + or -) to determine the pit slope. Thus, the boundaries resulting from the selected blocks define the pit outline. It ignores the rock mechanics characteristics and the hydrogeological conditions of the mine which defines the pit slope angles and are vital for good sequencing and optimization.

2.13 An Intelligent Dynamic Search Algorithm

Wang and Sevim (1992) have used intelligent dynamic search solution methodology to simultaneously optimize all principal elements of production scheduling problem in open pit mine without a priori assumption on any of the following four elements. These include: (i) the amount of ore and waste to be mined each year; (ii) the specific blocks to be mined each year, (iii) the mine life which defines the expected time horizon of the operation and (iv) the ultimate pit limits which delineate the total reserve to be mined and the final pit shape at the end of the operation. It takes into account both the interaction among the elements and the impact of the decisions concerning these elements in any period upon the decisions in future periods. The solution of the optimum production planning consists of two major steps: (i) obtaining the technically optimum pit using reserve parameterization or heuristic approach, and (ii) devising a scheme to economically evaluate the technically optimum pits to select the best pit sequence, and consequently best production plan. This algorithm does not guarantee that the pits are the truly maximum-metal pits. First, only the union of cones with the lowest average grades is considered. Due to overlapping among the cones, the average grade of the union of these cones may not be the minimum among all possible unions of the same size. Secondly, a technically optimum pit is the one that has the maximum metal quantity for a given total volume and a given ore tonnage. But in this algorithm, the maximum metal quantity is sought for a given total volume without considering the ore tonnage. Moreover, this algorithm suffers from the gap property: only a limited number of technically optimum pits can be obtained corresponding to some discrete values of total material and ore tonnages, and the sizes of the available pits can not be controlled. As a result, reserve parameterization is, in most cases, unlikely to produce all the pit of the right sizes needed for production planning purposes. Hence there is a definite need to devise an efficient algorithm to eliminate the gap problem. The heuristic algorithm does not guarantee that the pit limits are truly optimized.

2.14 Neural Network Overview

In this study, neural network is used to forecast mineral price and design and optimize pit layouts. Hence, the fundamentals, theory and architectures of neural networks are described in this section. The history of its development and current methodologies used in research are also presented.

The human audio/visual capabilities are superior to any kind of super-computer in mankind's history. The brain is composed of a large number, (10^{11}), of nonlinear processing unit called neurons. Figure 2.2 depicts a biological neuron reported in the literature [Haykin,1994; Hertz et al, 1991; Zurada, 1992].

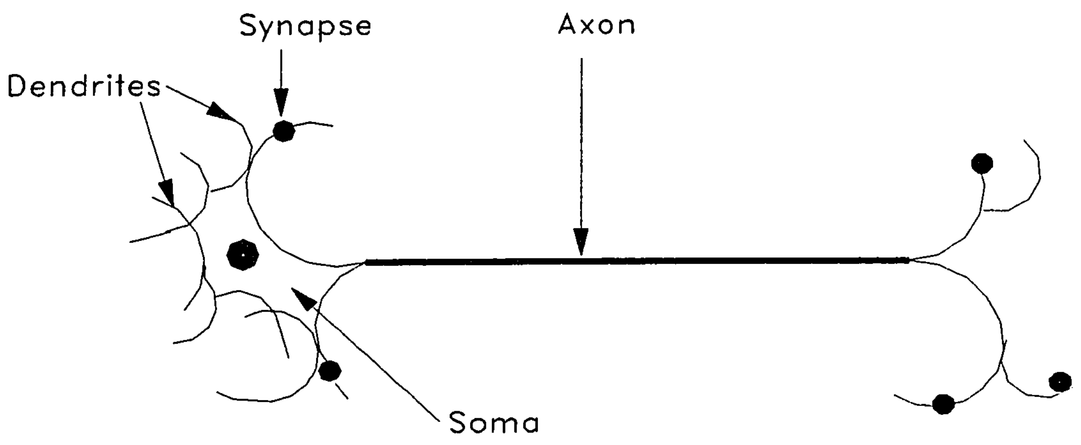


Figure 2.2 A Biological Neuron.

The dendrites are nerve cells that are connected to the neuron body, called the soma. Axons are the long fiber cables extending from the soma. Synapses are located at the end of axons, and are connected to other neurons. An axon is composed of several thousand synapses. Information transmission from one neuron to another takes place at a synapse which is considered to be a complicated chemical process. The goal is to increase or decrease the potential of a neuron that is receiving information. If the potential exceeds a threshold limit set in the receiving neuron, the neuron is triggered to send an information along the axon to the other neurons. The way these neurons are connected to each other makes us capable of recognizing speech or images very easily. Biological neurons are robust and fault tolerant. This means that if one or more neurons die or do not function properly, the overall performance of the brain does not change. They are highly parallel, very small, compact and low powered and they can deal with noisy, inconsistent data with no difficulty. These features motivated the new research field called artificial neural networks. McCulloch and Pitts (1943), in their initial attempt at modeling a neuron, proposed a binary threshold unit as a neuron. Their model is depicted in Figure 2.3 and equations 2.1 and 2.2. The McCulloch-Pitts neuron has proven to be capable of doing universal approximation for appropriately chosen weights as long as a number of them are connected in parallel and working simultaneously.

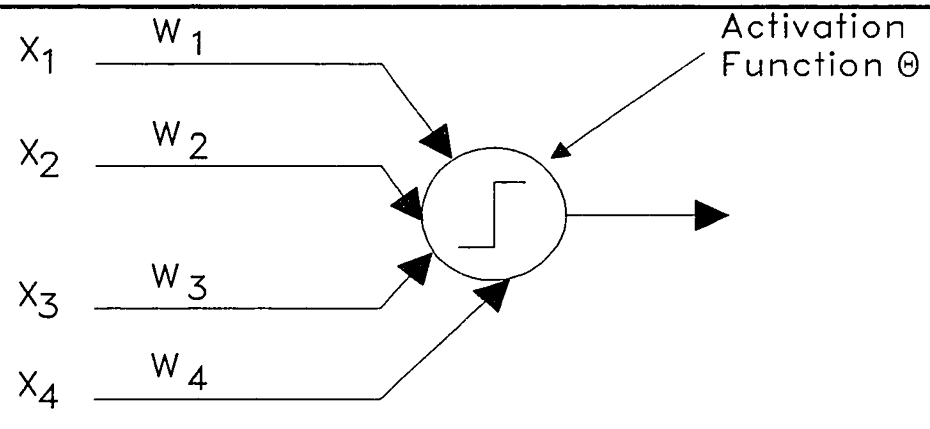


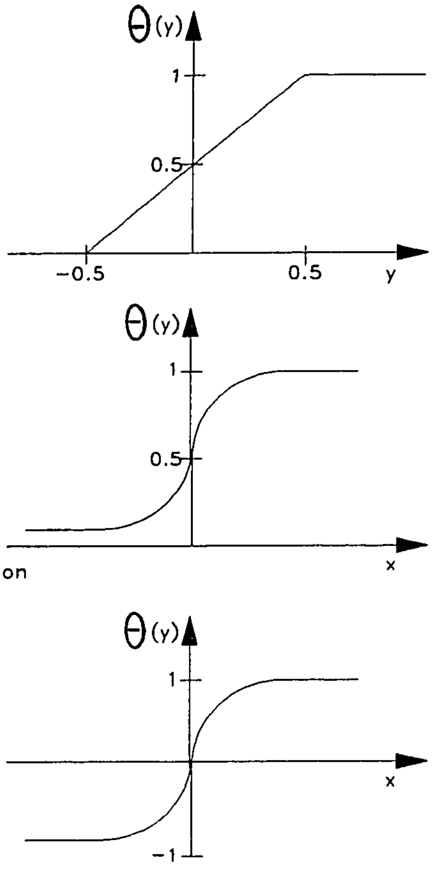
Figure 2.3 McCulloch-Pitts Neuron Model

$$\Theta(y) = \begin{cases} 1, & \text{if} \quad y \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

$$y = \Theta\left(\sum_{i=0}^{M} w_i x_i\right) \tag{2.2}$$

where y is an output unit, x_i is the presented input pattern, x_0 is **-1**, w_0 is considered as the threshold value, and M is the number of inputs. Haykin (1994) has described various activation functions. These activation functions as well as their mathematical representations are shown in Figure 2.4. The first one is piece wise function, and the latter one is sigmoidal function. Both logistic sigmoid function and hyperbolic tangent function **tanh(x)** are considered as sigmoidal functions.

There are various types of neural network architectures. The famous neural network architectures are multilayer feedforward neural network (MFNN) or multi-layer perceptron, Hopfield network and Kohonen network. Each of them is described in the next sections.



$$\Theta(y) = \begin{cases} 1, & \text{if } y > 1/2 \\ y, & \text{if } -1/2 > y > 1/2 \\ 0, & \text{if } y < -1/2 \end{cases}$$

(a) Piece Wise Activation Function

$$\Theta(y) = \frac{1}{1+e^{-x}}$$

(b) Logistic Sigmoidal Activation Function

$$\Theta(y) = \frac{1+e^{-2x}}{1+e^{-2x}}$$

(c) Tanh(x) Sigmoidal Activation Function

Figure 2.4 Mathematical Representation of Activation Functions.

2.14.1 Multilayer Feedforward Neural Network

Multilayer feedforward neural network consists of simple McCullogh-Pitts type neurons in a massively parallel structure (Figure 2.5). It is composed of an input layer, also called sensory unit layer, hidden layer, which could be as many as necessary, and an output layer. The input signal is usually propagated forward layer by layer. These types of artificial neural networks are referred to as multilayer feedforward (perceptrons). These networks were widely used to solve some classification and approximation problems employing a very popular training algorithm named error back propagation (EBP). The EBP is a learning scheme where the error is backpropagated and used to update the weights. The EBP algorithm is presented and explained in chapter 3.

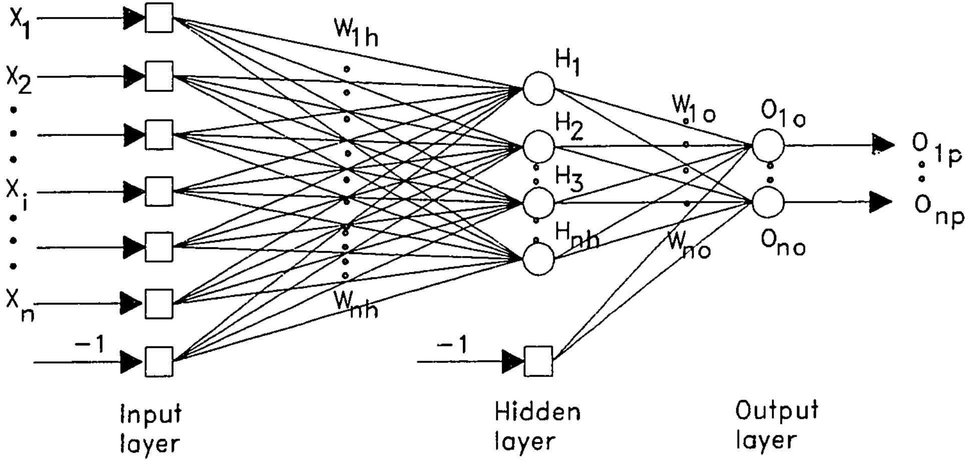


Figure 2.5 Multilayer Feedforward Neural Network

In a multilayer feedforward neural network (Figure 2.5) the nodes are grouped in input, hidden, and output layers, by the network. The output layer declares the computation results. The hidden or intermediate layers follow the input layer. All the nodes within the input layer, ($\mathbf{x_1}$, $\mathbf{x_2}$, $\mathbf{x_3}$, ..., $\mathbf{x_N}$) are connected to all the nodes of the first hidden layer, ($\mathbf{H_1}$, $\mathbf{H_2}$, $\mathbf{H_3}$, ..., $\mathbf{H_N}$), which are subsequently connected to all the nodes of the second hidden layer, and so on, and eventually to the nodes of the output layer, ($\mathbf{O_1}$, $\mathbf{O_2}$, $\mathbf{O_3}$, ..., $\mathbf{O_N}$). Weight, $\mathbf{W_{ih}}$ depicted in Figure 2.6, is associated with each one of the connections. $\mathbf{x_i}$ is the value of the input node. $\mathbf{W_{ih}}$ is the weight associated with the connection between node **i** of the input layer and node **h** of hidden layer. $\mathbf{W_{ho}}$ is the weight associated with the connection between node **h** of the hidden layer and node **o** of the output layer. $\mathbf{O_{ip}}$ is the activation value (network output) at output node **o** for pattern **p** on node **I**. The **-1**'s in Figure 2.5 are biases to the input and hidden layers.

The input nodes perceives the data to be processed by the network and then transmit it to the hidden nodes. Within the hidden layers, the values received from the input layer nodes are multiplied by the connection weights and then each hidden node sums all the weighted values it receives. On the other hand any other type of linear transformation can be used instead of the usual arithmetic summation. Finally a non linear transformation, such as a sigmoid function, is applied to the summation result and the resulting value (the activation level of the node) is then transferred to each of the nodes of the second hidden layer or of the output layer. The output layer plays the same role as the hidden layer however, the linear and non-linear transfer functions within the output layer can be different from those of the hidden layer. The activation level of the output nodes are the network outputs or results.

2.14.2 Hopfield Network

The Hopfield network consists of two layers, an input layer and a Hopfield layer (Figure 2.6). Each node in the input layer is directly connected to only Hopfield layer. The nodes in the latter layer are neuron models previously described with either hard limiting or sigmoidal activation functions [Hopfield, 1982]. The outputs of these nodes are weighted and fed back to the inputs of all of the other nodes.

During training, the network output is often required to be the same as the input. Connection strengths are weakened by reducing the corresponding weight values if the output of a neuron is different from the input, and strengthened when the converse is true. The trained network is used by applying an input pattern to the network. The network outputs are then continually fed back through the weights until a convergence criterion is met, typically when there are no changes at the network output nodes on successive iterations. This is the final network output for the input pattern. Binary input and output values, often represented as + 1 and - 1, are usually used with the Hopfield network.

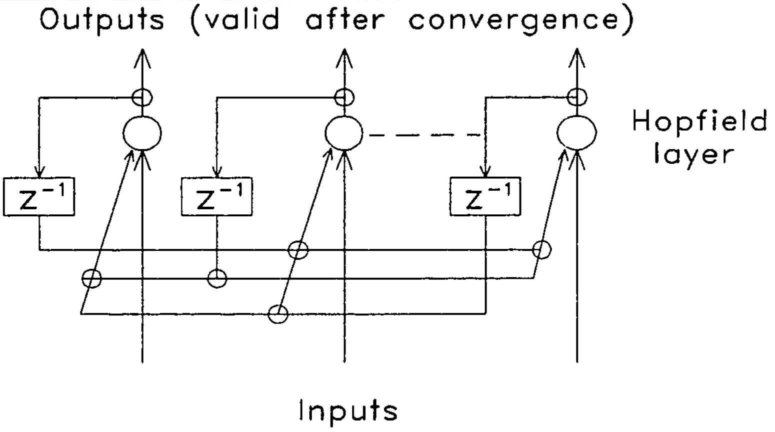


Figure 2.6 Hopfield Network. z^{-1} is the Unit Delay Operator.

2.14.3 Kohonen Network

The main distinguishing feature of this network, from the MFNN and Hopfield networks, is that no output data is required for training [Gomm, Page and Williams, 1993]. A Kohonen network is constructed of a fully interconnected array of neurons (i.e. the output of each neuron is an input to all neurons, including itself) and each neuron receives the input pattern (Figure 2.7) [Kohonen, 1988]. There are two sets of weights: an adaptable set to compute the weighted sum of the external inputs and a fixed set between neurons that controls neuron interactions in the network.

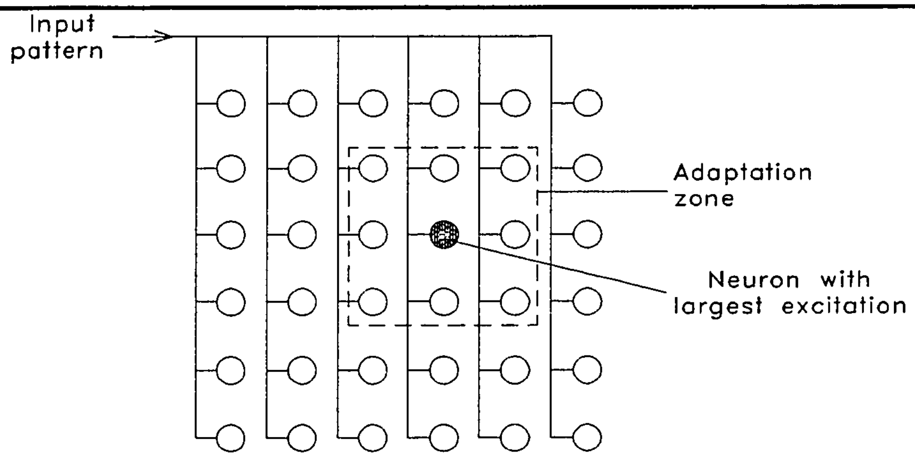


Figure 2.7 Kohonen Network.

Training involves applying an input pattern to the network, consisting of a set of continuous-valued data, and the output of each neuron is computed. The neurons are then allowed to interact with each other and the neuron that responds most to the input stimuli (i.e. the one that has the largest output) is found. Only this neuron and neighbourhood neurons, within a certain distance, are allowed to adjust their weights to become more responsive to the particular input. This form of training has the effect of organizing the "map" of the output nodes such that different areas of the map will respond to different input patterns. Hence, the Kohonen network has self-organizing properties and is capable of recognition.

2.15 Mineral Commodities

In mine design and valuation, mineral commodity price forecasts are important for future cash flow projection and evaluation of the optimum pit layout. The forecast must cover an extended period, the project life. This can be a decade or more years. However, prices of many mineral commodities are volatile, depicting unpredictable shifts in the balance between supply and demand. The significant balance is the one prevailing in the world market but not any national market. There are four main procedures by which mineral prices are established. They include producer price, independent price, negotiated price and commodity-exchange price methods [Strauss, 1992].

Producer prices are fixed by mineral producers, who periodically announces the terms and conditions, as well as the prices, at which sales will be made. In fixing the price, the seller takes into consideration his cost of production, potential markets, the position of his competition and the possibilities of increasing his market share. Independent prices are established periodically by an independent agency, often a trade periodical, that makes regular surveys among both buyers and sellers to avouch the basis at which actual transactions have taken place. Negotiated prices are prices negotiated directly between seller and buyer, with contractual agreement and provision for price adjustment under specific condition, extending over period of several years. Commodity-exchange prices are prices established on or largely influenced by transactions on commodity exchanges e.g. London Metal Exchange and New York Commodity Exchange [Strauss, 1992].

Two approaches for price forecasting are the trend analysis and econometric models. Trend analysis involves the replacement of the actual price-time history with a mathematical representation which can be used for projection into the future. A commodity model is a quantitative representation of a commodity market or industry [Labys, 1977]. Factors which determine mineral prices are very complicated and some of which are interdependent. The spot and future mineral prices are the result of these numerous factors. It is very difficult to bring these factors into traditional mathematical model for calculations. Usually only several main factors, instead of the whole set, are considered as variables to define a function to forecast commodity prices. Therefore, the experimental data are at variance from the predicted result for these kinds of formulas because of ignoring so many factors. Neural networks, a most attractive branch of artificial intelligence, has the potential to handle this problem, considering those governing factors as a whole, without limiting their quantity if a sufficient set of examples are available. Grudnitski and Osburn (1993) used neural networks to forecast monthly gold futures price changes based on past price changes, historical open interest patterns held to represent the beliefs of a majority of the traders in the gold futures market and a barometer of general economic conditions. They adaptively trained 41 networks from different 15-month training sets, applying a similarity-based selection process. Their networks forecast the correct direction of the next month's gold futures price change 61% of the time.

2.16 Summary

Various open pit optimization algorithms have been reviewed. Among them, the Lerchs-Grossmann and moving cone algorithms are widely used for open pit design and optimization. These two algorithms are saddled with some problems. Lerchs-Grossmann's open-pit optimization algorithm is inefficient in incorporating variable slopes in the pit design. In mining practice the pit slopes are determined by the rock mechanics of the mine. These conditions may result in variable pit slopes all over the entire mine. The Lerchs-Grossmann's 3-D algorithm fails to yield a truly optimized pit. The geometrical requirements of slope stability in combination with certain occurrences of ore blocks, can lead to situations where the floating cone will fail to design the maximum valued pit. The floating cone can fail to design true maximum valued pits, depending on the direction of search employed to locate positive valued blocks. Even in cases where the search for ore blocks commences at the top and proceeds to the lower rows, the floating cone cannot guarantee finding the optimum pit. The floating cone can fail to recognize a positive valued pit due to inability to solve the problem of overlapping waste when cones on individual blocks are combined. Due to this, with certain positive block configuration the maximum pit value may be missed.

From the above approaches to open pit limits design, the randomness associated with the distribution of the ore blocks, mining of the various blocks and price of minerals with respect to time have not been taken into consideration. The omission of this randomness property in the above algorithms poses limitations on the pit limit definition, the economic potential of the mineral deposit, and the long-, intermediate-, and short-range mine plans. This might lead to poor investment decision and inaccurate mine design and optimization. An algorithm which takes into account the random field properties of the mineral deposit, its overall economic potential in a strategic plan will aid in the proper design, optimization and subsequent extraction of the deposit.

Hence, in this research the theory of random field is proposed to incorporate the randomness property and the principal elements of production schedule and design parameters to give the best optimum pit limit. CS/MFNN algorithm is proposed for open pit optimization. The CS model is used for the random grades and ore reserves calculation, while the MFNN model is used for the pit optimization.

In this study MRM-MFNN is proposed to rigorously predict the mineral prices at various times by using the factors which influence mineral price determination. The stochastic gold price is modelled via two main models; multilayered feedforward neural networks (MFNN) model and multiple regression model (MRM) within an identified socio-politico-economic cycle.

The description of the CS/MFNN and MRM-MFNN models and their subsequent application and validation are given in details in the following chapters.

CHAPTER 3.0
MATHEMATICAL MODELLING

In open pit optimization, the depth and width of the pit from which the ore is extracted at a profit is defined using an open pit optimization algorithm. The first step involves building a block model and assigning grades to each block in the model using data base of grade samples from drill-holes. An interpolation technique is used to assign grades to all the block. After grade assignment to each block, mineral price is used to estimate the revenue of all the blocks based on mineral recovery. Cost of mining and profit (economic block value) for each block is also calculated. It is after knowing all the economic block values of the blocks that an open pit optimization algorithm is applied to define the optimum pit limit. The maximum allowable slope based on rock mechanics studies of the ore deposit is factored into the optimization algorithm.

In this study, conditional simulation (CS) model is the grade interpolation technique used to assign grades and subsequently economic block value to each block. A mineral price model is used to calculate the mineral price and multilayer feedforward neural network (MFNN) model is the open pit optimization algorithm used to define the ultimate pit limit. In this chapter, the mathematical modelling of 2-D and 3-D CS model, MFNN model and mineral price model are developed. The use of error back propagation algorithm in training MFNN is presented. The optimization of 2-D and 3-D open pit mine design using MFNN model are also developed.

3. 1 Block Model for Pit Design

The block model is the basis for almost all computer supported pit designs. For a block model, a rectangular block large enough to cover the area of interest is placed around the mineral deposit [Wright, 1990]. The large block, as shown in Figure 3.1, is then subdivided into smaller three-dimensional blocks (Figure 3.2). The smaller blocks may be of various different sizes and shapes. The geometrical position of a block is uniquely fixed with reference to any suitable coordinate system. Each block is assigned geological, rock mechanical, processing and economic data pertaining to each type of material contained in the block.

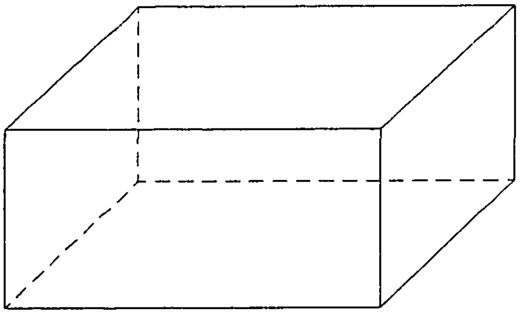


Figure 3.1 Large rectangular box to encompass area of interest

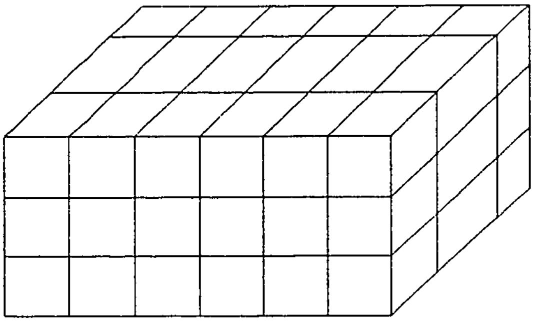


Figure 3.2 Large box is subdivided into smaller three-dimensional blocks

There are several types of block models (Kim, 1978): The regular 3-D fixed block model is the most widely used one in practice. The vertical height of each block is usually the bench height and the horizontal shape will normally be a rectangle or a square. The main characteristics of a 3-D fixed block model is that each block has the same measurements. By means of interpolation techniques such as geostatistics using kriging, inverse distance weighting methods and polygonal method, data are assigned to each block from a given data base.

3.2 The Economic Value of a Block

In open pit design, the criterion for maximizing the total pit value is equivalent to finding that collection of blocks within a pit limit which will give the maximum possible value; subject to mine stability and mining constraints. Hence the economic value of a block is of utmost importance. Each block in a block model can be characterized by:

1. Income ***I*** **:** value of the recoverable and salable part of the block;
2. Direct Costs ***DC*** **:** costs of mining that can be traced directly back to the block; e.g. drilling, blasting, loading and transportation costs.
3. Indirect Costs ***IC*** **:** overall costs which cannot be allocated to individual blocks. Such costs are time-dependent; e.g. depreciation for machinery, etc.

From these economic block value (***EBV***) can be defined as:

$$EBV = I - DC - IC \tag{3.1}$$

Ore blocks and blocks containing both ore and waste (= mixed blocks) will have ***EBV*** less than zero, equal to zero or greater than zero, depending on the amount and quality of the ore contained in such blocks. Waste blocks will always have a negative ***EBV*** since income from waste is zero. The optimization criterion for the pit limit design problem can thus be stated as:

$$\textit{Maximize } Z = \Sigma(EBV)_j \tag{3.2}$$

subject to slope stability and mining constraints.

After the determination of the ***EBV***, the algorithm for combining the blocks which will yield the maximum profit is applied to the gridded blocks. In this study, the conditional simulation is the interpolation technique used to assign grade to each block in the block model. The MRM-MFNN is then used to forecast the mineral price to calculate the EBV of each block.

3.3 Mathematical Modelling of Conditional Simulation

3.3.1 Block Modelling Using Conditional Simulation

In the conditional simulation model, the stochastic property associated with ore reserves and grade is defined using the limited known field data and used to estimate the unknown expected field data using best linear unbiased estimation (BLUE) and turning bands method (TBM) techniques. In the model, the grade assigned to each gridded-block is the average of all the 1000 realizations obtained during the simulation. The objective of the conditioning is to require the simulation model to assume observed data values at spatial locations where the random function of the grade distribution, *X*, has been observed. The flow diagram for the conditional simulation is depicted in Figure 3.3.

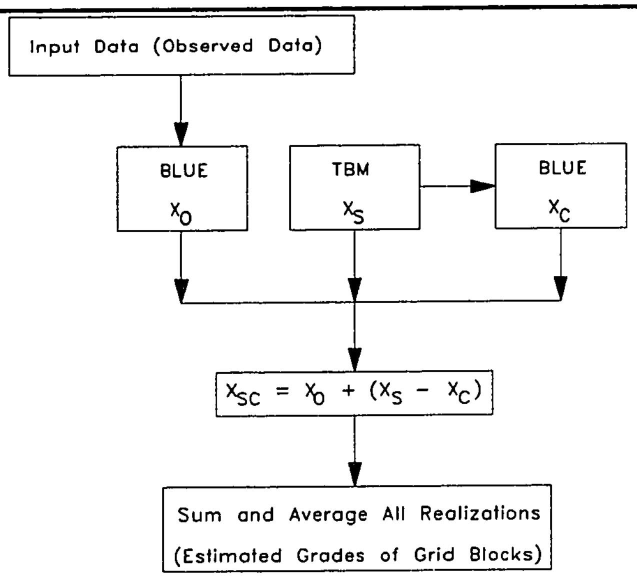

Input Data (Observed Data)
BLUE
X_0
TBM
X_S
BLUE
X_C
$X_{SC} = X_0 + (X_S - X_C)$
Sum and Average All Realizations
(Estimated Grades of Grid Blocks)

Figure 3.3 Flow Diagram of Conditional Simulation.

The assumptions in this model are:

i) the grades in the blocks are spatially and randomly distributed,

ii) the gridded blocks are defined by the coordinates of their centers.

iii) the deposit/field under study is isotropic and homogeneous.

iv) the geometry of the deposit is rectangular in shape for 2-D model and rectangular prism or cuboid for 3-D model.

The conditioned field for estimating a gridded-block grade, $X_{SC}(\underline{x})$, is obtained through a linear combination of three fields as stated in equation (3.3).

$$X_{SC}(\underline{x}) = X_O(\underline{x}) + (X_S(\underline{x}) - X_C(\underline{x})) \tag{3.3}$$

where $X_O(\underline{x})$is the estimated field based on observed data using BLUE; $X_S(\underline{x})$is unconditionally simulated field via TBM; and$X_C(\underline{x})$is the estimated field using BLUE based on the TBM values at the observed data locations. $X_S(\underline{x})$gives the TBM simulated values at both the observed and unknown data locations. The $X_C(\underline{x})$ values at the observed data locations are equal to their values generated by the field $X_S(\underline{x})$.

3.3.1.1 Modelling $X_O(\underline{x}_\beta)$ and $X_C(\underline{x}_\beta)$ Fields using BLUE

If $X_O(\underline{x}_\alpha)$, $\alpha = 1,2,........,n$ are the known values of X at $x_1, x_2, ...,x_n$ then the value of X at another point $\underline{x}_\beta$ can be estimated as

$$X^*(\underline{x}_\beta) = m_\beta + \sum_{\alpha=1}^{n} \eta_{\alpha\beta}(X(\underline{x}_\alpha) - m_\alpha) \tag{3.4}$$

where m_β, m_α are the known means at $\underline{x}_\beta$ and $\underline{x}_\alpha$ respectively; $\eta_{\alpha\beta}$ are the unknown weights for the given location $\underline{x}_\beta$ [Fenton, 1995]. In this model, m_β is the mean of the known drill-hole samples between which the point $\underline{x}_\beta$ lies.

Equation (3.4) is unbiased since

$$E[X^*(\underline{x}_\beta)] = m_\beta, \ (as \ \ E[X(\underline{x}_\alpha)] - m_\alpha = 0 \ ) \tag{3.5}$$

If $C(\underline{x}_i, \underline{x}_j) = E[(X_i(\underline{x}_i) - m_i)(X_j(\underline{x}_j) - m_j)]$ is defined as the covariance between the field processes at $\underline{x}_i$ and $\underline{x}_j$, then the mean square error of predicting the actual data $X(\underline{x}_\beta)$ with the $X^*(\underline{x}_\beta)$ is given by

$$E[\{X^*(\underline{x}_\beta) - X(\underline{x}_\beta)\}^2] = C(\underline{x}_\beta, \underline{x}_\beta) - 2\sum_{\alpha=1}^{n} \eta_{\alpha\beta} C(\underline{x}_\beta, \underline{x}_\alpha) +$$

$$\sum_{\alpha=1}^{n}\sum_{\gamma=1}^{n} \eta_{\alpha\beta}\eta_{\gamma\beta} C(\underline{x}_\alpha, \underline{x}_\gamma) \tag{3.6}$$

where $C(\underline{x}_\beta, \underline{x}_\alpha)$ is the covariance matrix of the unknown data at $\underline{x}_\beta$ and known data at $\underline{x}_\alpha$; $C(\underline{x}_\alpha, \underline{x}_\gamma)$ is the covariance matrix of known data $\underline{x}_\alpha$ and the weight $\underline{x}_\gamma$; $\eta_{\alpha\beta}$ and $\eta_{\gamma\beta}$ are weights.

Minimizing the mean square error in equation (3.6) leads to a system of n equations in n unknowns in equation (3.7)

$$\sum_{\gamma=1}^{n} \eta_{\gamma\beta} C(\underline{x}_\alpha, \underline{x}_\gamma) = C(\underline{x}_\beta, \underline{x}_\alpha) \quad , \ \alpha = 1,2,\ldots,n \tag{3.7}$$

where $n_{\gamma\beta}$ is the weight matrix to be determined.

Defining $n_{\gamma\beta} = \eta_\beta$ for convenience, equation (3.7) can be written in a matrix form as:

$$[\mathbf{K}_\alpha]\ [\eta_\beta] = \mathbf{M}_\beta \tag{3.8}$$

where $[\mathbf{K}_\alpha]$, $[\eta_\beta]$ and $\mathbf{M}_\beta$ are respectively as follows (using $\mathrm{C}_{ij} = C(\underline{x}_\alpha, \underline{x}_\gamma) = \mathrm{Cov}[X(\underline{x}_\alpha), X(\underline{x}_\gamma)]$)

$$[\mathbf{K}_\alpha] = \begin{bmatrix} C_{11} & C_{12} & \cdot & \cdot & \cdot & C_{1n} \\ C_{21} & C_{22} & \cdot & \cdot & \cdot & C_{2n} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ C_{n1} & C_{n2} & \cdot & \cdot & \cdot & C_{nn} \end{bmatrix},\ [\eta_\beta] = \begin{bmatrix} \eta_{1\beta} \\ \eta_{2\beta} \\ \cdot \\ \cdot \\ \cdot \\ \eta_{n\beta} \end{bmatrix},\ \mathbf{M}_\beta = \begin{bmatrix} C_{1\beta} \\ C_{2\beta} \\ \cdot \\ \cdot \\ \cdot \\ C_{n\beta} \end{bmatrix}$$

The solution is thus

$$[\eta_\beta] = [\mathbf{K}_\alpha]^{-1}\,[\mathbf{M}_\beta] \tag{3.9}$$

The elements in $[\mathbf{K}_\alpha]$ are the covariances between the known sample points calculated and/or read from the semi-variogram. The elements in $[\mathbf{M}_\beta]$ are the covariances between the unknown sample and known sample points also calculated and/or read from the semi-variogram.

Equation (3.9) can be solved for n_β via Gaussian elimination. n_β is a set of coefficients. $[\mathbf{K}_\alpha]$ depends only on the set of known points. Once it is inverted, it can be used repeatedly to find the best estimates at different unknown points $\underline{x}_\beta$. Solving equation (3.8) gives $n_{\alpha\beta}$ in equation (3.4) and the best linear unbiased estimate is

$$X^*(\underline{x}_\beta) = m_\beta + \sum_{\alpha=1}^{n} \eta_{\alpha\beta}(X(\underline{x}_\alpha) - m_\alpha) \qquad (3.10)$$

This same procedure can be used to generate $X_C(\underline{x}_\beta)$ using the data generated by turning bands method described below.

3.3.1.2 Modelling $X_S(\underline{x}_\beta)$ Field using TBM

The turning-bands method of Matheron (1973) is described in considerable detail by Journel and Huijbregts (1978) for three dimensions and by Mantoglou and Wilson (1981, 1982) and Bras and Rodriguez-Iturbe (1985) for both two and three dimensions, so only the essentials are presented here.

In this study, TBM is chosen because it is fast and easy to construct using a unidimensional covariance function for any dimension and has been successfully applied in mining geostatistics. Moreover, a lot of researches have been done on its implementation procedure making it easier and reliable to simulate a field of any dimension. This fast method builds a higher-dimensional simulation by averaging the contributions of several independent one-dimensional simulations oriented in several directions in space. The only limitation is that using too few lines results in streaked appearance of the realization and getting rid of the streaks substantially slows the algorithm.

In the simulation using turning bands method, it is assumed that the field is second order stationary and isotropic, at each point the values of the field are normally distributed and have zero mean and the covariance **C(r)** of the field is known. Also should it happen that the field is not normally distributed and have non-zero mean, a Gaussian transformation is made and the mean is subtracted. In the 2-D or 3-D fields, simulation is performed along several lines, using a unidimensional covariance function that corresponds to the given two- or three-dimensional one. Then at each point of the two- or three-dimensional field a weighted sum of the corresponding values of the line process is assigned. Let **P** be the two- or three-dimensional field to be simulated by generating values at discrete points in it. An arbitrary origin **0** in the dimensionality of the space $\mathbf{R}^n$, where $n = 2$ or $n = 3$, and lines are generated such that the corresponding direction vectors **u** are uniformly distributed on the unit circle or sphere in two- or three-dimensions, respectively.

In the two-dimensional case, the angle θ_i formed between a line i and a fixed x axis is uniformly distributed between **0** and 2π. Along each line i, a second-order stationary unidimensional discrete process is generated having zero mean and covariance function $\mathbf{C_1(\eta)}$, where η is the coordinate on line i. Onto line i, orthogonally project those points of the field where the values are to be generated, and assign to them the corresponding values of the one dimensional discrete process. If **N** is a point of the region having location vector $\mathbf{x}_N$, then the assigned value from the line i will be $\mathbf{z}_i(\eta_{Ni})$ where $\eta_{Ni} = \mathbf{x}_N{\cdot}\mathbf{u}_i$ is the projection of the vector $\mathbf{x}_N$ onto line i, $\mathbf{u}_i$ the unit vector on line i, and $\mathbf{x}_N{\cdot}\mathbf{u}_i$ represents the inner product of the vectors $\mathbf{x}_N$ and $\mathbf{u}_i$. Taking L lines such as i, an independent unidimensional realization is generated for each line using $\mathbf{C_1(\eta)}$ as the covariance function. Then at every point **N** of the region, there are L assigned values $\mathbf{z}_i(\eta_{Ni}) = \mathbf{z}_i(\mathbf{x}_N{\cdot}\mathbf{u}_i)$, where $i = 1, \ldots., L$, from the unidimensional realizations. Finally, the value $z_s(\mathbf{x}_N)$ is assigned to the point **N** where

$$\mathbf{z_s(x_N)} = \frac{\mathbf{1}}{\sqrt{\mathbf{L}}}\sum_{\mathbf{i=1}}^{\mathbf{L}} \mathbf{z_i(x_N \cdot u_i)} \qquad \mathbf{(3.11)}$$

is the realization of the two- or three-dimensional random field. The subscript s represents simulated or synthetic. If lines or planes are drawn perpendicular to the line at the ends of each discretized segment, a set of bands is defined as depicted in Figure 3.4. As the lines turn, the bands defined above also turn. Thus the method was given the name ‘turning bands method’ (TBM) by Matheron [Matheron,1973; Mantoglou and Wilson, 1982].

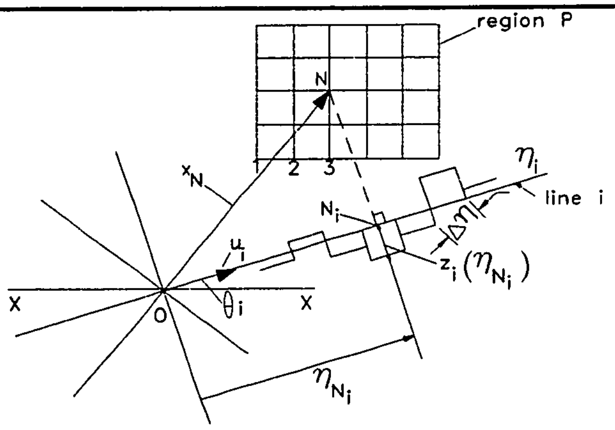


Figure 3.4 Schematic representation of the field **P** and the turning bands lines **I**.

The generation of a second-order stationary unidimensional discrete process having zero mean and covariance function $\mathbf{C_l(\eta)}$ is presented in the following section.

3.3.1.3 Unidimensional Covariance Function $C_l(\eta)$

The form of the unidimensional covariance function $\mathbf{C_l(\eta)}$ must be such that the field defined by equation (3.11) has the imposed two- or three-dimensional covariance function $\mathbf{C(r)}$ where $\mathbf{r} = |\mathbf{h}|$ defined below. For two points of the field having location vectors $\mathbf{x_1}$ and $\mathbf{x_2}$ respectively, the simulated values corresponding to these points are given by equation (3.11) and the covariance function of the simulated field is

$$\begin{aligned} \mathbf{C_s(x_1,x_2)} &= \mathbf{E[Z_s(x_1)Z_s(x_2)]} \\ &= \frac{1}{L}\sum_{i=1}^{L}\sum_{j=1}^{L} \mathbf{E[Z_i(x_1 \cdot u_i)Z_j(x_2 \cdot u_i)} \end{aligned} \qquad \textbf{(3.12)}$$

As the unidimensional realizations along two different lines are independent, the expected value $\mathbf{E[Z_i(x_1 \cdot u_i)Z_j(x_2 \cdot u_j)]}$ is zero unless $i \equiv j$. Thus equation (3.12) reduces to

$$\begin{aligned}\mathbf{C_s(x_1,x_2)} &= \frac{1}{L}\sum_{i=1}^{L}\mathbf{E[Z_i(x_1\cdot u_i)Z_j(x_2\cdot u_i)]} \\ &= \frac{1}{L}\sum_{i=1}^{L}\mathbf{C_1(h\cdot u_i)}\end{aligned} \qquad (3.13)$$

where $\mathbf{h} = \mathbf{x}_1 - \mathbf{x}_2$. The expected value $\mathbf{E[Z_i(x_1\cdot u_i)Z_j(x_2\cdot u_j)]}$ represents the covariance of the one-dimensional process on line *i* between points $\mathbf{x_1\cdot u_i}$ and $\mathbf{x_2\cdot u_j}$, which is written as $\mathbf{C_1(h\cdot u_i) = E[Z_i(x_1\cdot u_i)Z_j(x_2\cdot u_j)]}$, assuming that the unidimensional process is second-order stationary. Because the vector $\mathbf{u}_i$ is uniformly distributed over the unit circle or sphere, the right-hand side of equation (3.13) is only a function of $|\mathbf{h}|$ for large L. This means that the obtained process is wide sense stationary and isotropic, so that we can write

$$\mathbf{C_s(x_1,x_2) = C_s(h) = C_s(r)} = \frac{1}{L}\sum_{i=1}^{L}\mathbf{C_1(h\cdot u_i)} \qquad (3.14)$$

where r = $|\mathbf{h}|$. For L→ ∞ , by using the law of large numbers this becomes

$$\begin{aligned}\mathbf{C_s(r)} = \lim_{L\to\infty}\left\{\frac{1}{L}\sum_{i=1}^{L}\mathbf{C_1(h\cdot u_i)}\right\} &= \mathbf{E[C_1(h.u)]} \\ &= \int_{c}\mathbf{C_1(h.u)f(u)du}\end{aligned} \qquad (3.15)$$

where **c** represents the unit circle or sphere, **f(u)** is the probability density function of **u** which becomes $\frac{1}{2\pi}$ or $\frac{1}{4\pi}$ two- or three-dimensions cases respectively, and **du** is the differential length or area at the end of vector **u**. Equation (3.15) then gives, for the two-dimensional case

$$\mathbf{C_s(r)} = \frac{1}{2\pi}\int_{\text{unit circle}}\mathbf{C_1(h.u)du} \qquad (3.16)$$

while for the three-dimensional field

$$\mathbf{C_s(r) = \frac{1}{4\pi} \int_{\text{unit sphere}} C_1(h.u)du} \tag{3.17}$$

3.3.1.4 Covariance in Two-Dimensional Fields

Define orthogonal axes (x, y) in the plane of the field, with origin as point $\mathbf{x_1}$ and $\mathbf{x_2}$ along the y axis in the direction of the vector $\mathbf{h} = \mathbf{x_1} - \mathbf{x_2}$ as shown in Figure 3.5.

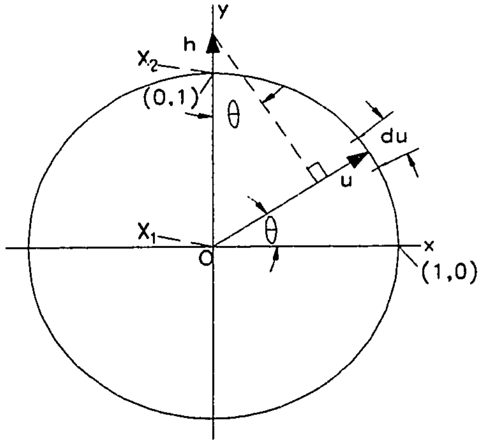


Figure 3.5 Definition sketch for the two-dimensional case, showing unit circle.

In polar coordinate we can write $\mathbf{h \cdot u} = \mathbf{r}\ \mathbf{sin\theta}$ and $\mathbf{du} = \mathbf{d\theta}$. Equation (3.16) then becomes

$$\mathbf{C_s(r) = \frac{1}{2\pi}\int_0^{2\pi} C_1(\theta)d\theta = \frac{2}{\pi}\int_0^r \frac{C_1(\eta)}{(r^2-\eta^2)^{1/2}}d\eta} \tag{3.18}$$

where $\eta = \mathbf{r}\ \mathbf{sin\theta}$ and that $\mathbf{C_1}$ is an even function. Substituting $\mathbf{C_s}$ $\mathbf{(r)} = \mathbf{C_1(r)}$, in order to preserve the known covariance, we get

$$\int_0^r \frac{C_1(\eta)}{(r^2-\eta^2)^{1/2}} d\eta = \frac{\pi}{2} C(r) \tag{3.19}$$

Equation (3.19) relates the two-dimensional covariance function **C(r)** to the corresponding unidimensional **$C_1(\eta)$** along the turning band lines. The line process is generated by using a spectral method of Rice [1954] and Shinozuka and Jan [1972].

3.3.1.5 Covariance in Three-Dimensional Fields

Because of the second-order stationarity and isotropy of the process, without loss of generality we can define orthogonal (*x, y, z*) axes with origin at the point $\mathbf{x}_1$ and with the *z* axis in the direction of the vector $\mathbf{h} = \mathbf{x}_1 - \mathbf{x}_2$, as depicted in Figure 3.6.

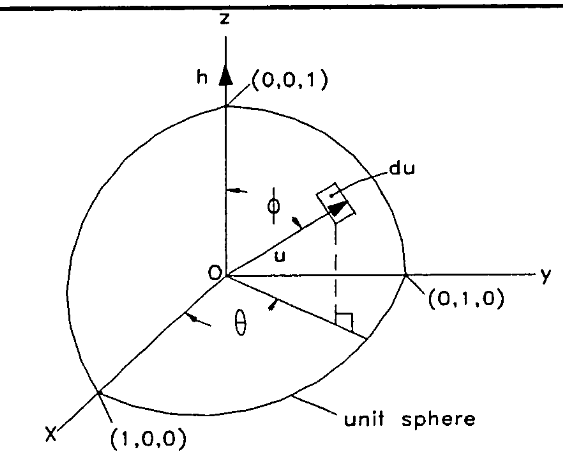


Figure 3.6 Definition sketch for the three-dimensional case, showing unit sphere.

The unit sphere where the vector **u** ends is also shown. In spherical coordinates $\mathbf{h}\cdot\mathbf{u}$ = r cosϕ, where r = |**h**|, and **du** = **sinϕdϕdθ**. The integral equation (3.17) is then written as

$$\begin{aligned} C_s(r) &= \frac{1}{4\pi}\int_0^{2\pi}\int_0^{\pi} C_1(r\cos\phi)\sin\phi \;\; d\phi \;\; d\theta \\ &= \frac{1}{r}\int_0^r C_1(\eta)d\eta \end{aligned} \tag{3.20}$$

taking advantage of the symmetry of $\mathbf{C_1}$ and introducing the line coordinate $\eta = \mathbf{r}\ \mathbf{cos}\ \phi$ oriented in the direction of **u**. Differentiating and changing notation leads to the relationship between one- and three-dimensional covariances:

$$\mathbf{C_1(\eta) = \frac{d}{d\eta}[\eta \quad C(\eta)]} \tag{3.21}$$

where $C_s(\eta) = C(\eta)$ in order to preserve the known three-dimensional covariance during simulation. For an exponential three-dimensional covariance $\mathbf{C(r)} = \boldsymbol{\sigma}^2\mathbf{exp\ (-br)}$, the corresponding one-dimensional covariance is $\mathbf{C_1(\eta)} = \boldsymbol{\sigma}^2\mathbf{(1\text{-}b\eta)exp(\text{-}b\eta)}$, commonly called the hole function. For a double exponential three-dimensional covariance $\mathbf{C(r)} = \boldsymbol{\sigma}^2\mathbf{exp\ (-b^2r^2)}$, the corresponding one-dimensional covariance is $\mathbf{C_1(\eta)} = \boldsymbol{\sigma}^2\mathbf{(1\text{-}2b^2\eta^2)exp\ (\text{-}b^2\eta^2)}$. For a spherical three-dimensional covariance $\mathbf{C(r)} = \boldsymbol{\sigma}^2\ \mathbf{[1\text{-}3br/2 + (br)^3/2]}$ for $\mathbf{0 \leq br \leq 1}$, and $\mathbf{C(r) = 0}$ for $\mathbf{br > 1}$, the corresponding one-dimensional covariance is $\mathbf{C_1(\eta)} = \boldsymbol{\sigma}^2\mathbf{[1\text{-}3b\eta + 2(b\eta)^3]}$ for $\mathbf{0 \leq b\eta \leq 1}$, and $\mathbf{C_1(\eta) = 0}$ for $\mathbf{b\eta > 1}$. σ^2 = variance; $\boldsymbol{b}$ = inverse of the range $\boldsymbol{a}$; $\boldsymbol{A} = \sin(\boldsymbol{\omega/b})$; $\boldsymbol{B} = \cos(\boldsymbol{\omega/b})$; $\boldsymbol{\beta}$ = parameter; arctan = arctangent.

3.3.2 Simulation of the Fields Using the Spectral Line Generation Method

After obtaining the covariance function, the corresponding spectral density function of the unidimensional process must be found in order to generate the process along the turning bands lines.

3.3.2.1 Spectral Representation of the Unidimensional Process

Let $S_1(\omega)$ be the spectral density function of the unidimensional process having covariance function $C_1(\eta)$. Then in one dimension the Fourier representation is

$$\mathbf{C_1 = \int_{-\infty}^{\infty} e^{i\omega\eta} S_1(\omega)d\omega = 2\int_{0}^{\infty} cos(\omega\eta)S_1(\omega)d\omega} \tag{3.22}$$

where the frequency (wave number) ω is a scalar, $S_1(\omega)$ is real, symmetric and positive. For the two-dimensional geometry of Figure 3.5, $\eta = \mathbf{r}\ \mathbf{cos}\ \phi$, since ηis oriented in the same direction as **u**. Substituting this Fourier representation of $C_1(\eta)$ into the TBM equation (3.18) setting $\mathbf{C_s(r) = C(r)}$ and changing the order of integration leads to

$$C(r) = \frac{4}{\pi}\int_0^\infty S_1(\omega)\left\{\int_0^{\pi/2} \cos(\omega r \quad \sin\theta)d\theta\right\}d\omega \tag{3.23}$$

From Gradshteyn-Ryzhik (1980), the integral inside the brackets { } becomes $(\pi/2)J_0(\omega r)$. Thus equation (3.23) can be written as

$$C(r) = 2\int_0^\infty S_1(\omega)J_0(\omega r)d\omega \tag{3.24}$$

Applying a Hankel transform to this equation gives

$$S_1(\omega) = \frac{1}{2}\int_0^\infty C(r)J_0(\omega r)r \quad dr \tag{3.25}$$

The Fourier transform pair for the two-dimensional continuous process, which tends to zero fast as $|\mathbf{h}| \rightarrow \infty$, covariance function **C(h)** can be written as equations (3.26) and (3.27) [Shoenberg,1938; Matern, 1960; Mantoglou and Wilson, 1982].

$$C(r) = \sigma^2\int_0^\infty f(\omega)J_0(\omega r) \quad d\omega \tag{3.26}$$

$$f(\omega) = \frac{\omega}{\sigma^2}\int_0^\infty C(r)J_0(\omega r)r \quad dr \tag{3.27}$$

$\mathbf{r} = |\mathbf{h}|$, $J_0(\ )$ is a Bessel function of first kind of order zero, and $\mathbf{f}(\omega)$ is the radial spectral density function of the two dimensional isotropic process defined by

$$f(\omega) = \frac{1}{\sigma^2}\int_{c_\omega} S(\omega) \quad d\omega = \frac{1}{\sigma^2}S(\omega)\int_{c_w} d\omega = \frac{2\pi\omega S(\omega)}{\sigma^2} \tag{3.28}$$

c_ω is a circle of radius ω and dω is the differential length on circle c_ω.

Comparing equations (3.25) and (3.27), we obtain

$$S_1(\omega) = \frac{\sigma^2}{2} f(\omega) \qquad (3.29)$$

This means that the spectral density function of the unidimensional process along the turning bands lines is simply given by one half of the radial spectral density function of the two-dimensional process multiplied by the variance. The comparison which led to equation (3.29) can be used to derive the spectral density functions of the unidimensional processes for various two- and three-dimensional covariance functions using derived radial spectral density functions such as tabulated by Mantoglou and Wilson, (1982). The unidimensional spectral density function is related to the unidimensional covariance function for a transformed pair of Fourier. The analytic expression of S(ω) for the more widely used models in the unidimensional geostatistical practice is shown in Table 3.1 [Pardo-Iguzquiza et al., 1992].

Table 3.1 Analytic expression of S(ω) for more widely used models

Model	C(h)	S(ω)
Exponential	$\sigma^2 exp(-bh)$	$\frac{\sigma^2}{\pi}\frac{b}{(b^2+\omega^2)}$
Gaussian	$\sigma^2 exp(-b^2h^2)$	$\frac{\sigma^2}{2\sqrt{\pi}b}\exp\left[-\left(\frac{\omega}{2b}\right)^2\right]$
Triangular	$\sigma^2(1-bh)\quad h \le a$ $0 \quad h > a$	$\frac{\sigma^2 b}{\pi\omega^2}\left[1-\cos\left(\frac{\omega}{b}\right)\right]$
Spherical	$\sigma^2\left[1-\frac{bh}{2}+\frac{2b^3h^3}{3}\right] \quad h \le a$ $0 \quad h > a$	$\frac{3\sigma^2 b}{2\pi\omega^3}\left[A+\frac{b}{\omega}(B-1)\right]$
Hole I	$\sigma^2(1-bh)exp(-bh)$	$\frac{\sigma^2 b}{\pi(b^2+\omega^2)}\left[1-\cos\left(2\arctan\left(\frac{\omega}{b}\right)\right)\right]$
Hole II	$\sigma^2 exp(-bh)cos(\beta h)$	$\frac{\sigma^2}{2\pi}+\left[\frac{b}{b^2+(\omega+\beta)^2}+\frac{b}{b^2+(\omega-\beta)^2}\right]$

σ^2 = variance; ***b*** = inverse of the range ***a***; ***A*** = sin(ω/***b***); ***B*** = cos(ω/***b***); β = parameter; arctan = arctangent.

3.3.2.2 Generation of the Line Process

After obtaining the spectral density function of the unidimensional process, the process along the turning bands lines can easily be generated using the classical method proposed by Rice (1954) and modified by Shinozuka and Jan (1972). If the unidimensional covariance function is $C_1(\eta)$ and the corresponding spectral density function is $S_1(\omega)$, then the unidimensional process on line *i* can be generated by

$$z_i(\eta) = 2\sum_{k=1}^{M}[S_1(\omega_k)\Delta\omega]^{1/2}\cos(\omega_k'\eta + \phi_k) \qquad (3.30)$$

ϕ_k are independent random angles uniformly distributed between **0** and 2π, ω_k = **(k-1/2)**$\Delta\omega$, and $\omega_k{}' = \omega_k + \delta\omega$ for **k** = **1, ..., M**. It has been assumed that the spectral density function $S_1(\omega)$ is insignificant outside the region [$-\Omega$, $+\Omega$], the band frequency of the process. The discretization frequency $\Delta\omega$ is defined as $\Delta\omega = \Omega/M$, **M** is the number of harmonics used. The frequency $\delta\omega$ is a small random frequency added in order to avoid periodicities and is uniformly distributed between $-\Delta\omega'/2$ and $+\Delta\omega'/2$, where $\Delta\omega'$ is a small frequency, $\Delta\omega' << \Delta\omega$. It has been proven by Shinozuka and Jan (1972) that the process given by equation (3.30) has zero mean and covariance function $C_1(\eta)$, as $M \rightarrow \infty$, $\Omega \rightarrow \infty$, and $\Delta\omega \rightarrow 0$. This process is strictly ergodic, even for a finite number of lines, and Gaussian. In the turning bands method, equation (3.30) is used to generate values at discrete points. These points are chosen to be the middle points of the segments defined by the bands along each line. The same value is assigned to the entire segment or band.

Equations (3.15), (3.16) and (3.17) are obtained in the limit as the number of lines goes to infinity. The lines are assumed to be randomly oriented, as taken from a uniform distribution on the unit circle or sphere. These TBM equations are also be obtained by spacing the lines evenly on the unit circle or sphere, with prescribed directions. If the same line orientations are maintained during a sequence of simulations, both methods are ergodic, but the simulated covariance converges much faster to the theoretical function for the even spacing approach and hence this is the preferred approach. For two-dimensional fields, it has been observed that 4-16 lines should sufficient [Mantoglou and Wilson, 1982]. In three dimension, experience has shown that a group of 15 lines, joining the midpoints of the opposite edges of a regular icosahedron, is adequate for typical applications [Journal and Huijbregts, 1978].

3.4 MRM-MFNN Mineral Price Model

In this study, the stochastic gold price is modelled via two main models; namely multiple regression model (MRM) and multilayer feedforward neural networks (MFNN) as depicted in Figure 3.7.

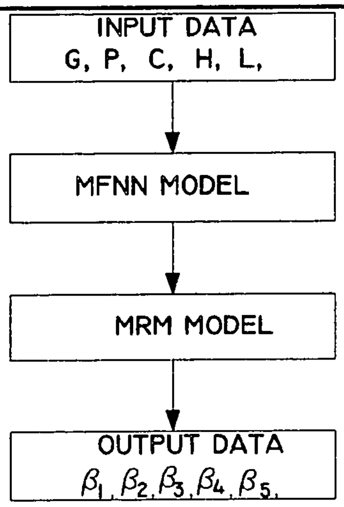


Figure 3.7 Flow Diagram for Mineral Price Model

Due to the ever changing socio-politico-economic factors in the world, mineral price trend and transaction data have undergone some drastic metamorphosis which leave analysts to ponder over what and how to use the data in designing mineral forecast models. Before the collapse of the former Union of Soviet Socialist Republics (U.S.S.R), all mineral transactions were done in the COMECON or CMEA (Council for Mutual Economic Assistance) market. Data was unavailable from the eastern bloc, but the situation changed after the collapse of the communist and socialist eastern bloc. Historical mineral price data must be truncated in order to develop a better price forecast model for sound, meaningful and coherent analysis to reflect economic realities in this modern world. As a result of these problems, data for validating realistic commodity models is presently scanty. In this study, the multiple regression technique is used to develop a mineral price model for predicting future gold prices. The data within a determined socio-politico-economic trend is used as input into the multiple regression model to estimate the coefficients of the respective parameters. These coefficients are used to construct a multiple regression equation for predicting the mineral commodity prices.

In this study, it is assumed that the world inflation and interest rates are reflected in the price and that only data reported for periods with consistent economic and mineral price trend are used to forecast the future prices. Also, the limited data available is assumed to be sufficient in defining the random distribution of the commodity price.

Gold, like any commodity, obeys the law of demand and supply. Gold price is determined by the interaction of supply (production) and demand (consumption). The production of operating and prospective new mines can be estimated from their production schedules. A careful and detailed study of consumption trends in end-use markets can permit enlightened estimates to be made about likely levels of demand. Average annual monthly high and low gold prices depict the range of fluctuation of the average annual gold price in a year. Hence, the variability of the average annual monthly high and low gold prices with average annual gold price is of statistical significance.

Having made the aforementioned assumptions and in accordance with multiple regression theory, gold price is assumed to be linear combination of four variables, which include world (i) annual gold production; (ii) annual gold consumption; (iii) average annual monthly low gold prices; and (iv) average annual monthly high gold prices. It is assumed that these variables exhibit random variability, cross-correlation and autocorrelation characteristics and that they are distributed normally [Pindyck and Rubinfeld, 1991].

3.4.1 Multilayer Feedforward Neural Network Model for Average Annual Monthly High and Low Gold Prices Modelling

The principle of the MFNN model is the same as previously discussed. In this model, the world annual gold production and consumption are the two inputs. The world average annual monthly high and low gold prices are the two outputs. The network details are depicted in Figure 3.8. The generated data is then used to train the MFNN model for prediction of the two outputs.

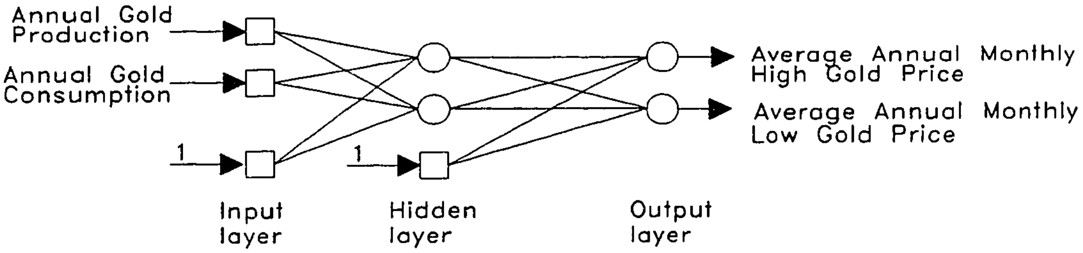


Figure 3.8 MFNN Model for Average Annual Monthly High and Low Gold Prices Modelling.

3.4.2 Multiple Regression Model

According to Pindyck and Rubinfeld (1991), in the multiple regression model, the dependent variable, ***P***, is linearly related to a series of independent variables, X_1, X_2, ..., X_k and an error term ε. With N samples from the dependent and independent variables, the multiple regression model can be written as

$$P_1=\beta_1+\beta_2X_{21}+\beta_3X_{31}+\ldots+\beta_kX_{k1}+\varepsilon_1$$
$$P_2=\beta_1+\beta_2X_{22}+\beta_3X_{32}+\ldots+\beta_kX_{k2}+\varepsilon_2$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$P_N=\beta_1+\beta_2X_{2N}+\beta_3X_{3N}+\ldots+\beta_kX_{kN}+\varepsilon_N \qquad \textbf{(3.31)}$$

Thus, there are **k+1** variables - one dependent variable and **k** independent variables (including the constant term). The corresponding matrix formulation of the model is

$$P=X\beta+\varepsilon \qquad \textbf{(3.32)}$$

$$P=\begin{bmatrix} P_1 \\ P_2 \\ \cdots \\ P_N \end{bmatrix};\ X=\begin{bmatrix} 1 & X_{21} & \cdot & \cdot & \cdot & X_{k1} \\ 1 & X_{22} & \cdot & \cdot & \cdot & X_{k2} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & X_{2N} & \cdot & \cdot & \cdot & X_{kN} \end{bmatrix};\ \beta=\begin{bmatrix} \beta_1 \\ \beta_2 \\ \cdot\cdot \\ \beta_N \end{bmatrix};\ \varepsilon=\begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \cdot\cdot \\ \varepsilon_N \end{bmatrix}$$

where P = $N \times 1$ column vector of dependent variable observations

X = $N \times k$ matrix of independent variable observations

β = $k \times 1$ column vector of unknown parameters

ε = $N \times 1$ column vector of errors

In matrix X, each component X_{ij} has two subscripts, the first signifying the appropriate column (variable) and the second signifying the appropriate row (observation). According to Pindyck and Rubinfeld (1991), the assumptions of the classical linear regression model are that the: (i) model specification is given by equation (3.31); (ii) elements of X are fixed and have finite variance; (iii) variable, X has rank k, which is less than the number of observations N; (iv) variable, X, is normally distributed with $E(\varepsilon) = 0$ and $E(\varepsilon\varepsilon') = \sigma^2 I$, where I is an $N \times N$ identity matrix. ε' is the $1 \times N$ vector transpose of ε. The assumption that X has rank k guarantees that perfect collinearity will not be present. With perfect collinearity, one of the columns of X would be a linear combination of the remaining columns and the rank of X would be less than k. The error assumptions are the strongest possible, since they guarantee the statistical, as well as, the arithmetic properties of the ordinary least-squares estimation process. In addition to normality, it is assumed that each error term has mean zero, all variances are constant, and all covariances are zero. The variance-covariance $\sigma^2 I$ appears as follows:

If G, P, C, H, L are defined as the world annual gold price, annual gold production, annual gold consumption, average-annual monthly high gold price and average-annual monthly low gold price respectively, then the multiple regression model can be stated as

$$G = \beta_1 + \beta_2 P + \beta_3 C + \beta_4 H + \beta_5 L + \varepsilon \tag{3.33}$$

where ε is an error term; β_1, β_2, β_3, β_4 and β_5 are the regression coefficients. β_1 is a constant term. The dependent gold price variable, G, is linearly related to a series of the independent variables, P, C, H, L and an error term ε.

3.5 Multilayer Feedforward Neural Network (MFNN)

3.5.1 The Error Back Propagation (EBP) Algorithm

Figure 3.9 illustrates the MFNN and its components. The network learning is based on repeated representations of the training samples. In a MFNN, the nodes are grouped into input, hidden and output layers by the network. The output layer yields the computation results. The hidden or intermediate layers follow the input layer. $X_p = x_0, x_1, x_2, \ldots, x_{n-1}$ is the presented input pattern. w_{ij} is the weight from node i to node j. o_{pj} is the actual outputs for pattern p on node j.

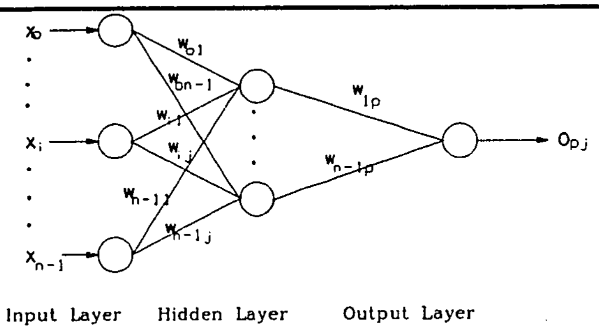


Figure 3.9 Multilayer Feedforward Neural Network Components

The EBP, a training algorithm for the multilayer perceptron, is a learning scheme where the error is backpropagated and used to update the weights. The algorithm is a gradient descent method that minimizes the error between the desired and actual outputs calculated by the multilayer perceptron [Rumelhart et al., 1986; Beale et al., 1990]. In MFNN, EBP allows experiential acquisition of input/output mapping knowledge within the multilayer networks. If a submitted pattern and its classification is determined to be erroneous, the synaptic weights, as well as, the thresholds are adjusted so that the current least mean-square error is reduced. These back propagations and error adjustments continue until all mapping examples from the training set are learned within an acceptable overall error. Figure 3.10 depicts the implementation of EBP algorithm.

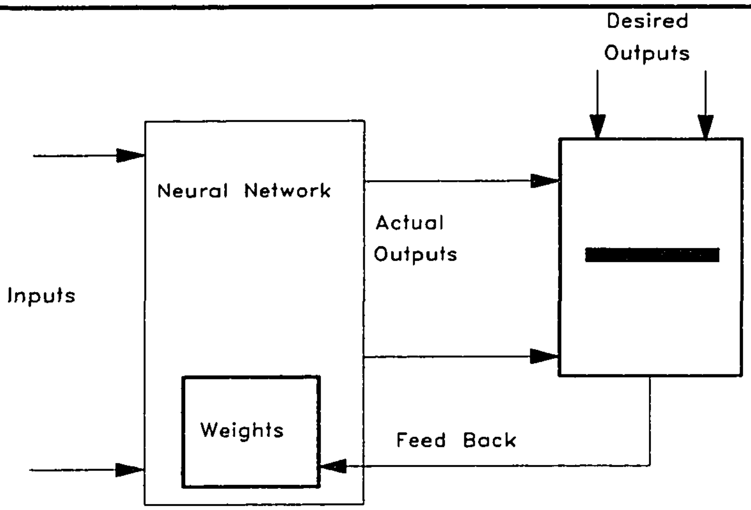


Figure 3.10 Implementation of EBP Algorithm.

If E_p is the error function for a pattern p and proportional to the square of the difference between the actual and desired outputs for all the patterns to be learnt, then

$$E_p = \frac{1}{2}\sum_{j}^{N}(t_{pj} - o_{pj})^2 \tag{3.34}$$

The activation of each unit j, for pattern p is

$$net_{pj} = \sum_{i}^{N} w_{ij} o_{pi} \tag{3.35}$$

The output from each unit *j* is the threshold function f_j acting on the weighted sum. In this multilayer perceptron, f_j is the sigmoid function defined as

$$f(net) = 1/(1+e^{-knet})\ ;\ (0 < f(net) < 1) \qquad \textbf{(3.36)}$$

k is a positive constant that controls the spread of the function. The actual output , o_{pj}, can also be defined as

$$o_{pj} = f_j(net_{pj}) \qquad \textbf{(3.37)}$$

$$\frac{\partial E_p}{\partial \omega_{ij}} = \frac{\partial E_p}{\partial net_{pj}} \frac{\partial net_{pj}}{\partial \omega_{ij}} \quad \textbf{(from the chain rule)} \qquad \textbf{(3.38)}$$

From equation (3.35), the second term on the RHS of equation (3.38) is given by

$$\frac{\partial net_{pj}}{\partial \omega_{ij}} = \frac{\partial}{\partial \omega_{ij}} \sum_k \omega_{kj} o_{pk}$$

$$= \sum_k \frac{\partial \omega_{jk}}{\partial \omega_{ij}} o_{pk} = o_{pi} \qquad \textbf{(3.39)}$$

$\frac{\partial \omega_{jk}}{\partial \omega_{ij}} = 0$ (except when $k = i$ when it equals to 1). Thus, the error can be defined as a function of the change in the net inputs to a unit as

$$-\frac{\partial E_p}{\partial net_{pj}} = \delta_{pj} \qquad \textbf{(3.40)}$$

Substituting equations (3.39) and (3.40) in equation (3.38), we have

$$-\frac{\partial E_p}{\partial w_{ij}} = \delta_{pj} o_{pi} \tag{3.41}$$

From equation (3.41), decreasing E_p implies that the weights changes are proportional to $d_{pj}o_{pj}$, i.e.,

$$\Delta_p \omega_{ij} = \eta \delta_{pj} o_{pi} \tag{3.42}$$

Using equation (3.40) and the chain rule, d_{pj} can be written as

$$\delta_{pj} = -\frac{\partial E_p}{\partial net_{pj}} = -\frac{\partial E_p}{\partial o_{pj}} \cdot \frac{\partial o_{pj}}{\partial net_{pj}} \tag{3.43}$$

From equation (3.34), the second term on the RHS of equation (3.43), is given by

$$\frac{\partial o_{pj}}{\partial net_{ij}} = f_j'(net_{pj}) \tag{3.44}$$

Differentiating E_p in equation (3.40) with respect to o_{pj}, the first term on the RHS in equation (3.45) is

$$\frac{\partial E_p}{\partial o_{pj}} = -(t_{pj} - o_{pj}) \tag{3.45}$$

Substituting equation (3.44) and (3.45) in equation (3.43), we have

$$\delta_{pj} = f_j'(net_{pj})(t_{pj} - o_{pj}) \tag{3.46}$$

Equation (3.46) is the error term for the output units which is back-propagated by the EBP algorithm. Assuming unit **j** is not an output unit, the error term for the hidden units can be formulated by considering equation (3.47).

$$\frac{\partial E_p}{\partial o_{pj}} = \sum_k \frac{\partial E_p}{\partial net_{pk}} \cdot \frac{\partial net_{pk}}{\partial o_{pj}}$$

$$= \sum_k \frac{\partial E_p}{\partial net_{pk}} \cdot \frac{\partial}{\partial o_{pj}} \sum_i \omega_{ik} o_{pi} = -\sum_k \partial_{pk} \omega_{ik} \qquad \textbf{(3.47)}$$

Substituting equation (3.47) in equation (3.43) we have

$$\delta_{pj} = f_j^{'}(net_{pj}) \sum_k \partial_{pk} \omega_{ik} \qquad \textbf{(3.48)}$$

Equation (3.48) represents the change in the error function with respect to the weights in the network. The function is proportional to the error, ∂_{pk}, in subsequent units, so the error has to be calculated in the output units first in equation (3.47) and then passed back through the networks to the earlier units to allow them to alter their connection weights. Equations (3.46) and (3.48) define the training of multilayer networks. The constant k in equation (3.36) also acts as an automatic gain control, since for small input signals the slope is quite steep and so the function is changing quite rapidly, producing a large gain. For large inputs, the slope and thus the gain is much less. This means that the network can accept large inputs and still remain sensitive to small changes.

A major reason for the use of the sigmoid function is that it has a simple derivative, and this makes the implementation of the back-propagation system much easier. The output of a unit, o_{pj} is given by

$$o_{pj} = f(net) = 1/(1+e^{-knet}) \tag{3.49}$$

The derivative with respect to that unit, $f^{'}(net)$, is a simple function of the outputs and is given by

$$\begin{aligned} o_{pj} = f(net) &= ke^{-k \ net} / (1+e^{-k \ net})^2 \\ &= k \quad f(net)(1-f(net)) \\ &= k \quad o_{pj}(1-o_{pj}) \end{aligned} \tag{3.50}$$

3.5.2 Multilayer Perceptron Learning Algorithm

Beale et al. (1990) presented a detailed multilayer perceptron learning algorithm as in the following steps:

(i) initialize weights and thresholds by setting them to small random values; (ii) present input $X_p = x_0, x_1, x_2, \ldots, x_{n-1}$, and desired target output $T_p = t_0, t_1, t_2, \ldots, t_{m-1}$ where n is the number of input nodes and m is the number of output nodes. Set w_0 to be $-\theta$, the bias, and x_0 to be always 1. For pattern association, X_p and T_p represent the patterns to be associated. For classification, T_p is set to zero except for one element set to 1 that corresponds to the class that X_p is in; (iii) calculate actual output. Each layer calculates

$$y_{pj} = f\left[\sum_{i=0}^{n-1} w_i x_i\right] \tag{3.51}$$

and passes that as input to the next layer. The final layer outputs values o_{pj}.; (iv) adapt weights. Start from the output layer, and work backwards.

$$w_{ij}(n+1) = w_{ij}(n) + \eta\delta_{pj} o_{pj} \tag{3.52}$$

$w_{ij}(n)$ represents the weights from node ***i*** to node ***j*** at step ***n***, η is a gain term or momentum coefficient, and δ_{pj} is an error term for pattern ***p*** on node ***j***. The momentum method is for improving the training time of the backpropagation algorithm, while enhancing the stability of the process [Rumelhart, Hinton and Williams, 1986]. This involves adding a momentum coefficient to the weight adjustment that is proportional to the amount of the previous weight change. Once an adjustment is made it is remembered and serves to modify all subsequent weight adjustment. The momentum coefficient is commonly set to around 0.9 [Wasserman, 1989].

For output units

$$\delta_{pj} = ko_{pj}(1 - o_{pj})(t_{pj} - o_{pj}) \tag{3.53}$$

For hidden units

$$\delta_{pj} = ko_{pj}(1 - o_{pj})\sum_{k} \delta_{pk} w_{jk} \tag{3.54}$$

where the sum is over the ***k*** nodes in the layer above node ***j***.

3.5.3 2-D Open Pit Optimization Model Using MFNN

In this study, MFNN is chosen for the optimization process because it can learn and implement arbitrary complex input/output mappings or decision surfaces separating pattern classes. Moreover, it has the ability to classify linearly nonseparable patterns. The ability of the multilayer perceptrons in classifying linearly nonseparable patterns is used to delineate the conditioned blocks into the various regions. MFNN have many identical nodes, with computational features that enable them to transform perceived signals into new transmittable signals. EBP algorithm is chosen to train the MFNN because it is an efficient and exact method for calculating all derivatives of a single target quantity with respect to a large set of input quantities. Also, it performs supervised learning task that allows inputs and outputs to map correctly, and this is directly applicable to the separation of the gridded-blocks based on their EBVs and center coordinates. In the supervised learning, EBP adapts the MFNN so that its actual outputs come close to some target outputs. The final weight obtained after EBP learning process is what is used to draw separation lines (decision lines).

With the 2-D model, MFNN is used to classify and partition the conditionally simulated blocks into classes based on their economic block values (EBVs). Each block is identified by the coordinates of its center of gravity and its EBV. In the 2-D model, the decision surfaces separating the 2-D block model of the gridded blocks are straight lines. The inputs are the 2-D coordinates of the centers of gravity and EBVs of the gridded blocks. The target values of the gridded blocks are numerically **-1**, **0** and **1**, corresponding to negative, zero and positive EBV respectively. The blocks are ranked based on their economic values as illustrated in equation (3.55) and these ranking values become the target values, $\mathbf{t}_{pj}$, for the error back propagation algorithm.

$$R[EBV_{ij}] = t_{pj} = \begin{cases} 1 & \forall \;\; positive \;\; EBV_{ij} \\ 0 & \forall \;\; zero \;\; EBV_{ij} \\ -1 & \forall \;\; negative \;\; EBV_{ij} \end{cases} \tag{3.55}$$

The flow diagram for MFNN model for Pit Optimization is depicted in Figure 3.11.

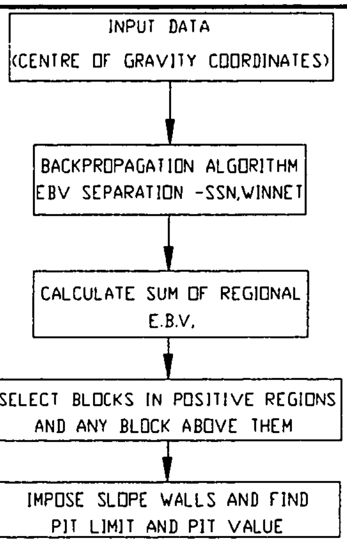

INPUT DATA
(CENTRE OF GRAVITY COORDINATES)
BACKPROPAGATION ALGORITHM
EBV SEPARATION -SSN,WINNET
CALCULATE SUM OF REGIONAL
E.B.V.
SELECT BLOCKS IN POSITIVE REGIONS
AND ANY BLOCK ABOVE THEM
IMPOSE SLOPE WALLS AND FIND
PIT LIMIT AND PIT VALUE

Figure 3.11 MFNN Flow Diagram of Pit Optimization

Using error back propagation algorithm, all the blocks are partitioned and the associated final weights are calculated. These final weights are used to generate decision lines (equations of straight lines) which are used to partition the blocks into regions. The sum of the block E.B.Vs in each region is calculated and is positive, negative or zero. The blocks in the positive regions form the basis for detailed search using search algorithm. The slope wall constraints are then imposed. All the blocks in the positive regions are subjected to the slope constraint as discussed in the next section. This process leads to the selection of group of positive blocks which can be economically and technically mined at a profit. Then any block lying on top of any of the selected positive blocks is selected as mineable. The optimum pit value is computed by summing all the blocks confined by the optimized pit layout. The details of the programs for accomplishing all these steps are discussed in the next chapter.

3.5.4 Imposition of Slope Constraints

In the 2-D model, the slope walls, in the form of straight lines, are imposed on the blocks in the positive regions. The 2-D pit model features are depicted in Figure 3.12.

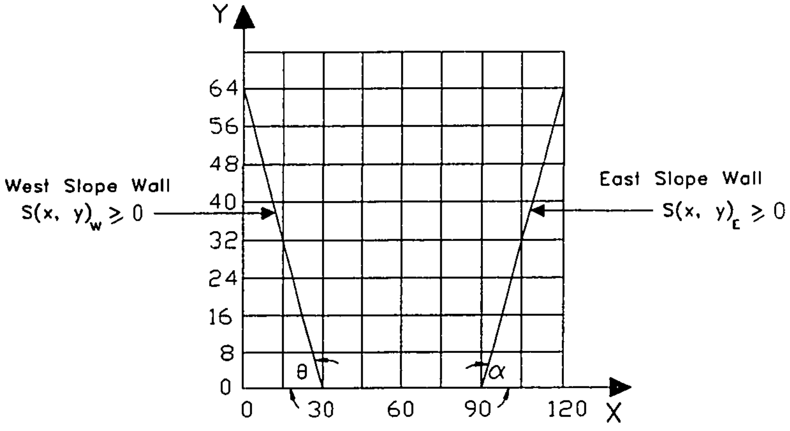


Figure 3.12 Diagram of 2-D Pit Section.

Stating the equation of a straight line as

$$\mathbf{y = ax + c} \tag{3.56}$$

where **(x , y)** is the coordinate of the gridded block center; **c** is the intercept of the slope wall with the referenced y-axis; θ and α are the slope angles of the west and east slope walls respectively; slope wall **a = tan(θ) or tan(α)**. Using algebraic principles in searching for optimum pit blocks, the slope wall equation, **S(x,y)**, can be stated as

$$\mathbf{S(x,y) = y - ax - c} \tag{3.57}$$

The inclusion of a block in the optimum pit is governed by the conditions:
for the west and east slope walls:

$$\begin{cases} S_W(x,y) \geq 0 \ \ \textit{and} \ \ S_E(x,y) \geq 0, & \textit{the block is within the pit.} \\ \textit{Otherwise,} & \textit{the block is outside the pit.} \end{cases} \qquad (3.58)$$

The slope walls are adjusted in the following way in search for the optimum pit:

(i) moving the east slope wall towards the west slope while the west slope wall is stationary or holding west slope wall stationary and moving east slope wall outward. The movement is shown in Figure 3.13. The dotted lines are the new positions of slope wall during movement.

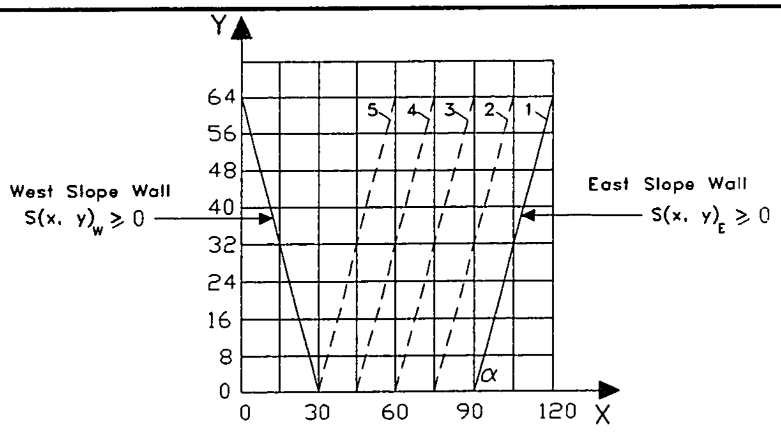


Figure 3.13 Holding West Slope Wall Stationary and Moving East Slope Wall.

(ii) moving the west wall towards the east slope while the east slope wall is stationary or holding east slope wall stationary and moving west slope wall outward. The movement is shown in Figure 3.14.

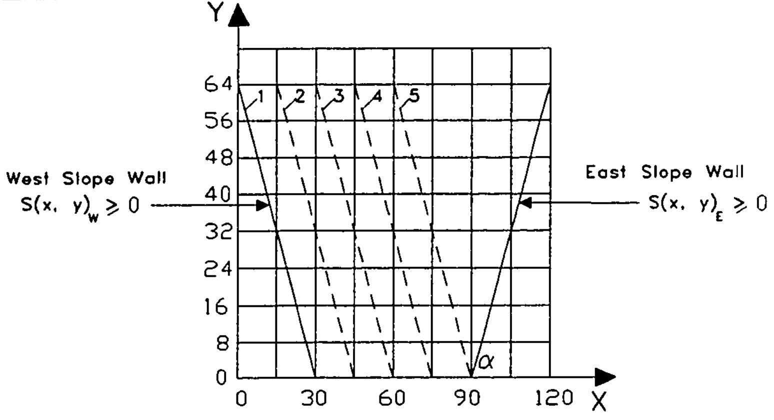


Figure 3.14 Holding East Slope Wall Stationary and Moving West Slope Wall.

(iii) moving both slope walls towards or away from each other is shown in Figure 3.15.

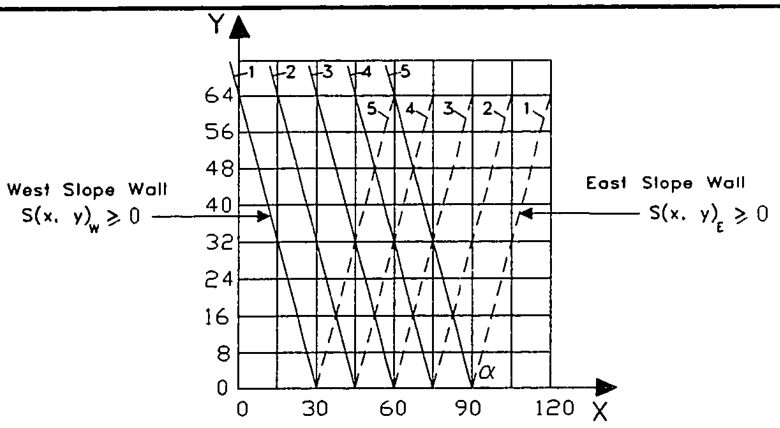


Figure 3.15 Moving both Slope Walls Towards or Away from each other.

The movements are done level by level and along east-west direction in determined sequence e.g. per fraction of block or some distance. This is achieved by varying the intercept c of the equation of the slope wall. At each new position of the slope walls, the total sum of E.B.Vs of blocks confined by the slope walls are calculated and compared with the previous sum. The larger sum is used to compare with the next one. At the end of the slope wall movements, the slope walls define the optimum pit limit and the final sum of all the E.B.Vs of blocks confined by the optimized pit layout is the optimum pit value.

3.5.5 3-D Open Pit Optimization Model Using MFNN

With the 3-D model, MFNN is used to classify and partition the conditionally simulated blocks into classes based on their economic block values (EBVs). Each block is identified by the coordinates of its center of gravity and its EBV. In the 3-D model, the decision surfaces separating the 3-D block model of the gridded blocks are planes. The inputs are the 3-D coordinates of the centers of gravity and EBVs of the gridded blocks. The ranking and target values of the gridded blocks are the same as in the 2-D model. The flow diagram for MFNN 3-D model for Pit Optimization is the same as depicted in Figure 3.11. The 3-D pit model features are depicted in Figure 3.16.

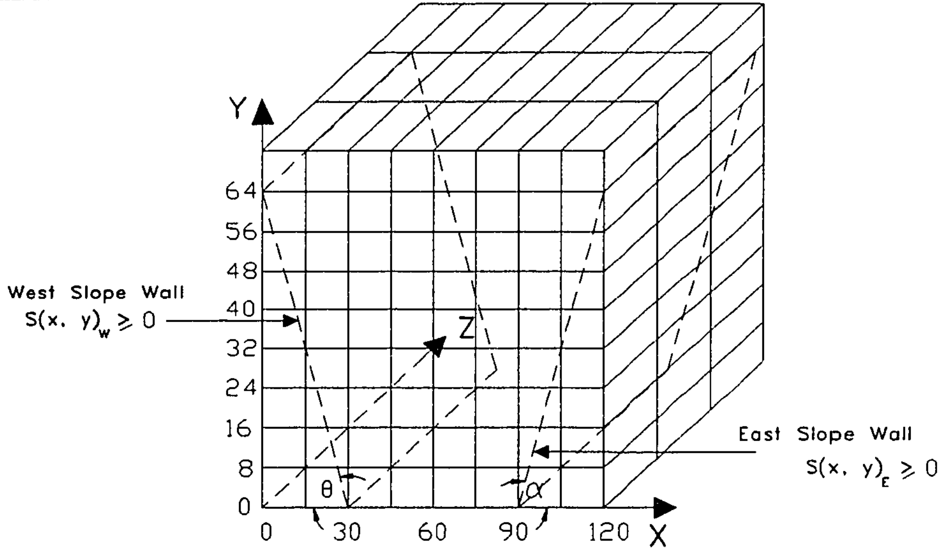


Figure 3.16 Diagram of 3-D Pit Section.

In the 3-D model, the slope wall equation is in the form of an equation of a plane

$$\frac{x}{p}+\frac{y}{q}+\frac{z}{s}=1 \quad \textit{where} \quad p \neq 0, \; q \neq 0, \; s \neq 0, \tag{3.59}$$

where p, q and s are plane intercept coordinates with the **x**, **y** and **z** axes respectively; (**x**, **y**, **z**) is the coordinate of the gridded block center; θ and α are the slope angles of the west and east slope walls respectively. As in the 2-D case, the 3-D slope wall equation **S(x, y, z)**, can be stated as

$$S(x,y,z)=\frac{x}{p}+\frac{y}{q}+\frac{z}{s}-1 \tag{3.60}$$

In the 3-D case, the slope wall does not intercept the **z**-axis (**oz**). Thus the term $\frac{z}{s}$ in equation (3.60) is equal to zero and disappears. Hence, the slope wall equation takes the form of the previous 2-D model. The procedure for searching for the optimized pit layout and optimum pit value is the same as that of the 2-D model. The inclusion of a block in the optimum pit is governed by the conditions:

for the west slope and east slope wall:

$$\begin{cases} S_W(x,y,z) \geq 0 \;\; \textit{and} \;\; S_E(x,y,z) \geq 0, & \textit{the block is within the pit.} \\ \textit{Otherwise,} & \textit{the block is outside the pit.} \end{cases} \tag{3.61}$$

3.5.6 Generalized Flow Chart for the 2-D and 3D Algorithm

The algorithm can be stated as follows:

1. Start from any of the positive regions and search for the ore blocks which are inclusive in the optimized pit. The ore blocks have positive economic block values.
2. Substitute the blocks center coordinates in the slope wall equations.

3. Sum all the positive blocks whose center coordinates satisfy the slope wall equations.
4. Continue the search until all the ore blocks in the positive regions of the block model have been examined.
5. The E.B.V of any block lying on any of the positive blocks in the optimized pit is added to the sum in (3).
6. The ultimate pit is formed by the blocks in the positive region whose center coordinates satisfy the slope wall equations and the blocks lying on any of the positive regional blocks in the optimized pit.

Figure 3.17 depicts the generalized flow chart for the 2-D and 3D Algorithm.

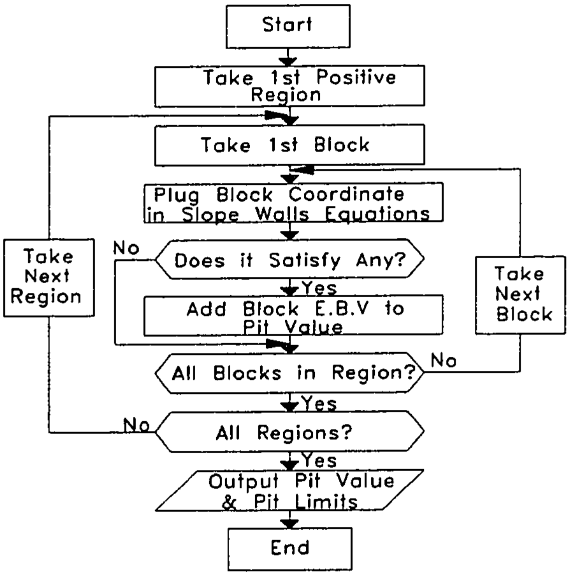
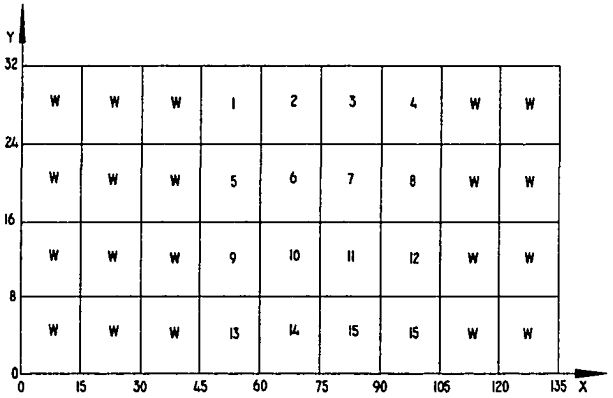


Figure 3.17 Generalized Flow Chart for the 2-D and 3D Algorithm.

3.5.7 Illustrative Example of MFNN Pit Optimization

Optimized pit limits are required for the 2-D section of a gold deposit in Figure 3.18. The EBVs and coordinates of all the blocks at their centers of gravity are provided in Table 3.2. The main problem is to optimize the pit limits using MFNN algorithm.



W -denotes waste

Figure 3.18 2-D Section of Pit

Table 3.2 EBV, Target values and center coordinates of gridded-blocks

Block #	$x = X_1$	$y = X_2$	EBV (Blk) (US\$ $\times 10^3$)	Target (t_{pj})
1	7.5	28	-11.40	-1
2	22.5	28	252.40	1
3	37.5	28	251.00	1
4	52.5	28	-11.40	-1
5	7.5	20	255.20	1
6	22.5	20	238.60	1
7	37.5	20	416.70	1
8	52.5	20	257.90	1
9	7.5	12	253.80	1
10	22.5	12	242.70	1
11	37.5	12	249.60	1
12	52.5	12	273.10	1
13	7.5	4	872.50	1
14	22.5	4	253.80	1
15	37.5	4	240.00	1
16	52.5	4	-11.40	-1

Solution Procedures

Pit Optimization Using Neural Networks: Figure 3.19 illustrates the neural networks for block classification in the pit limits optimization. The block center coordinates and defined targets in Table 3.2 are used as inputs and outputs data respectively in the MFNN shown in Figure 3.19. X_1 and X_2 are the x and y coordinates respectively and are inputs of the network. O_{pj} are the target values which are 1, 0 or -1 for positive, zero and negative economic block values respectively. The input error or tolerance for complete training is 0.01; a_1, a_2, n_1, n_2, n_3, n_4 and n_5 are neurons and b_1 and b_2 are biases equivalent to -1 each. Using WinNN software package [Danon, 1996] and applying EBP training algorithm to the networks in Figure 3.19, the blocks are partitioned and the associated final weights are calculated. The final weights can be any real number.

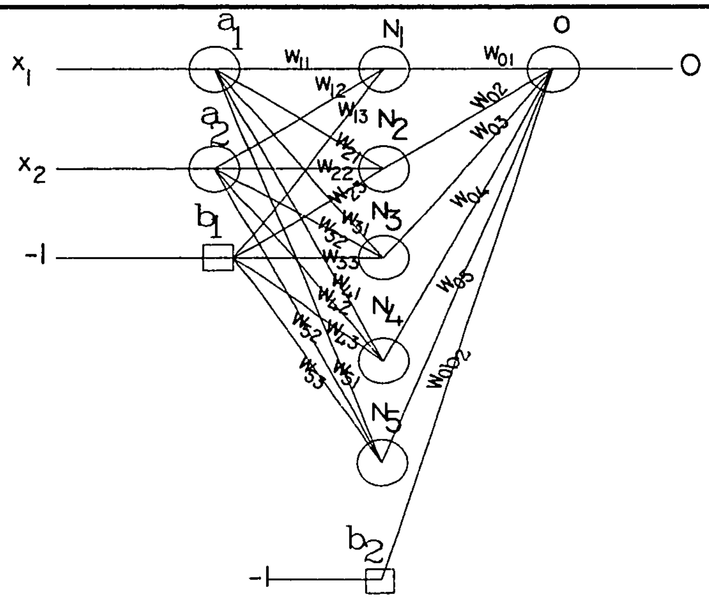


Figure 3.19 Neural Networks for Block Classification.

The final weights between the input and the hidden layer, shown in Table 3.3, are used to write the decision line equations (3.63), (3.64), (3.65), (3.66) and (3.67).

Table 3.3 Weights of input layer to hidden layer

Weight of node\node	X_1	X_2	b_1
N_1	0.01429	0.00000	0.64286
N_2	-0.17142	-0.32142	-1.28571
N_3	-0.01200	- 0.02250	-1.71000
N_4	0.00760	0.0000	0.00943
N_5	-0.01069	- 0.02005	0.88235

Given that the final weights at N_1 are w_{11}, w_{12}, and w_{13} as shown in Figure 3.19, the decision line equation is written as

$$\textbf{DL: } g(X_1, X_2) = w_{11}X_1 + w_{12}X_2 - w_{13} \qquad (3.62)$$

$$\textbf{DL1: } g(X_1, X_2) = 0.01429X_1 - 0.64286 \qquad (3.63)$$

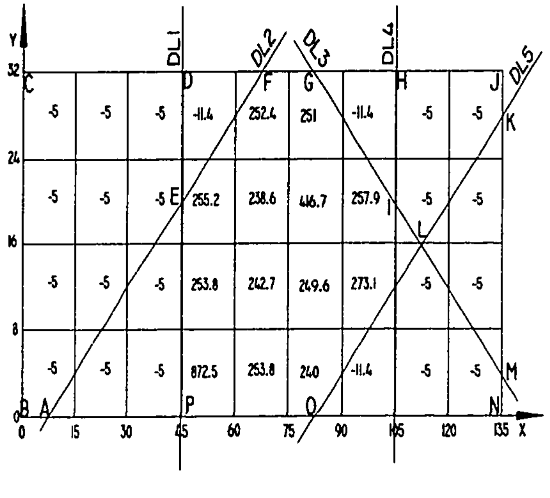
$$\textbf{DL2: } g(X_1, X_2) = -0.17142X_1 - 0.32142X_2 + 1.28571 \qquad (3.64)$$

$$\textbf{DL3: } g(X_1, X_2) = -0.01200X_1 - 0.02250X_2 + 1.71000 \qquad (3.65)$$

$$\textbf{DL4: } g(X_1, X_2) = 0.99057X_1 -0.00943 \qquad (3.66)$$

$$\textbf{DL5: } g(X_1, X_2) = -0.01069X_1 - 0.02005X_2 - 0.88235 \qquad (3.67)$$

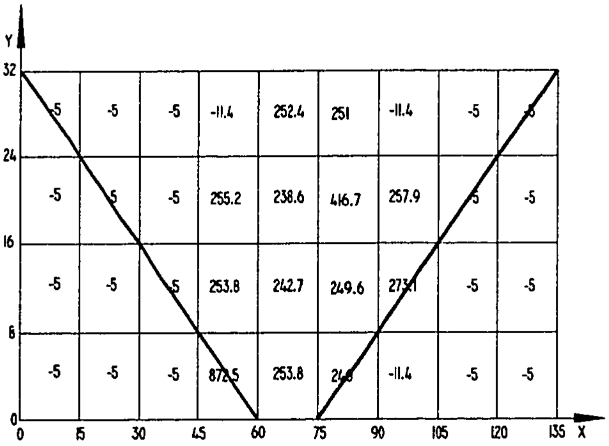
The blocks in Figure 3.18 are divided into regions by the decision lines as illustrated in Figure 3.20. The sum of the block EBVs (US$ × 10^3) in regions ABCDEA, DEFD, AEPA, FGILOPEF, GHIG, HJKLIH, KLMK and LMNOL have EBV of -45, -11.4, -15, 4,057.3, -11.4, -15, -10 and -26.4 respectively.



-5	-5	-5	-11.4	252.4	251	-11.4	-5	-5
-5	-5	-5	255.2	238.6	416.7	257.9	-5	-5
-5	-5	-5	253.8	242.7	249.6	273.1	-5	-5
-5	-5	-5	872.5	253.8	240	-11.4	-5	-5

Figure 3.20 Blocks Delineation by MFNN

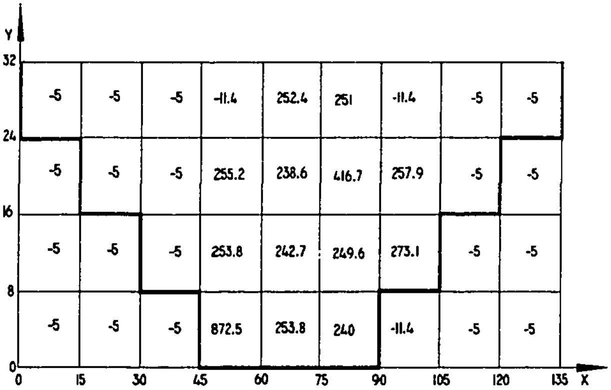
Imposing Slope Constraints: The final slope walls imposed on the mineable blocks are as illustrated in Figure 3.21. Blocks whose centers are enclosed in or lie on the slope walls are included in the final pit outline as illustrated in Figure 3.21.



-5	-5	-5	-11.4	252.4	251	-11.4	-5	-5
-5	-5	-5	255.2	238.6	416.7	257.9	-5	-5
-5	-5	-5	253.8	242.7	249.6	273.1	-5	-5
-5	-5	-5	872.5	253.8	240	-11.4	-5	-5

Figure 3.21 Imposing Slope Constraints

Optimized Pit Value using MFNN Algorithm: The optimized pit value is the sum of all the blocks contained in the pit limits defined in Figure 3.22. Thus, the optimized pit value is equal to US$ 3, 989,400.



-5	-5	-5	-11.4	252.4	251	-11.4	-5	-5
-5	-5	-5	255.2	238.6	416.7	257.9	-5	-5
-5	-5	-5	253.8	242.7	249.6	273.1	-5	-5
-5	-5	-5	872.5	253.8	240	-11.4	-5	-5

Figure 3.22 Optimized Pit Outline using MFNN

3.6 Summary

The CS model for assigning grades to block model, mineral price model MRM-MFNN for calculating economic block values of block model and MFNN model for optimizing open pit have been developed. The solution algorithms and computer programs for implementing all the above models have been described in chapter 4. The modelling validation, test results and illustrations of these models, and discussions are provided in chapter 5 and chapter 6.

CHAPTER 4.0

COMPUTER MODELLING AND EXPERIMENTAL INVESTIGATIONS

The computer modelling, the solution procedures and experiments designed to implement all the mathematical models developed in chapter 3 are discussed. Solution algorithms for the 2-D and 3-D problems are described. In addition, the flow charts for the computer programs used to implement the models, the experimental setup, the procedure and the experimentation process are described.

4.1 Solution to Mathematical Models

The flow diagram for the solution to the mathematical models is illustrated in Figure 4.1. The solution starts with the CS model. The semi-variogram for all the known block grade samples are fitted to determine the variogram model and its parameters.

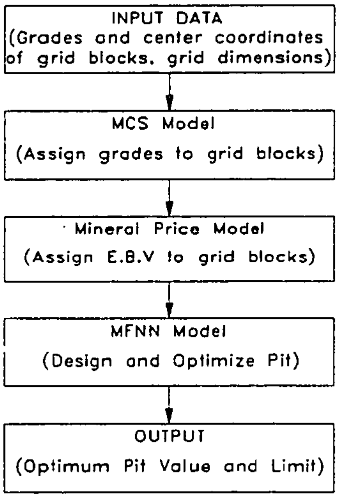

INPUT DATA
(Grades and center coordinates of grid blocks, grid dimensions)
MCS Model
(Assign grades to grid blocks)
Mineral Price Model
(Assign E.B.V to grid blocks)
MFNN Model
(Design and Optimize Pit)
OUTPUT
(Optimum Pit Value and Limit)

Figure 4.1 Flow Diagram of the Experimental Procedure

With the variogram model parameters and the block center coordinates, grade estimates are assigned to all the other unknown blocks. The mineral price model predicts the average annual mineral prices for the project life. Using block grades and the average annual mineral prices, economic block values of all the blocks are calculated. The MFNN model is used to optimize the pit to yield the optimum pit value and layout.

4.1.1 Solution Procedure for CS Model

4.1.1.1 Solution Procedure for BLUE

In equation (3.10), the elements in matrix **[K$_\alpha$]** are the covariances between the known sample points calculated and/or read from the semi-variogram. Rewriting equation (3.10) as equation (4.1), the solution is thus

$$[\eta_\beta] = [K_\alpha]^{-1} [M_\beta] \qquad \textbf{(4.1)}$$

The elements in **M$_\beta$** are the covariances between the unknown sample and known sample points also calculated and/or read from the semi-variogram. Equation (3.10) can be solved via Gaussian elimination for n_β, a set of coefficients. **[K$_\alpha$]** depends only on the set of known points, so it is inverted once and then used repeatedly to find the best estimates at different unknown points $\underline{x}_\beta$.

4.1.1.2 Solution Procedure for TBM

In the simulation of the 2-D or 3-D fields, simulation is performed along several lines, using a unidimensional covariance function that corresponds to the given two- or three-dimensional one. Then at each point of the two- or three-dimensional field a weighted sum of the corresponding values of the line process is assigned. The fields are of zero mean and unit variance.

4.1.2 Solution Algorithm for MFNN Model

The algorithm for the MFNN model is the multilayer perceptron learning algorithm described in chapter 3. The network is trained so that application of a set of inputs produces the desired (or at least consistent) set of outputs. Each such input (or output) set is referred to as a vector. Training is accomplished by sequentially applying input vectors, while adjusting network weights according to a predetermined procedure. During training, the network weights gradually converge to values such that each input vector produces the desired output vector.

4.2 Development of Computer Models

The procedure for the development of computer models is shown in the flow diagram of Figure 4.2. The first stage of the process commences with the development of the technical concepts, based on the problem and objectives of the thesis as outlined in chapter 1.

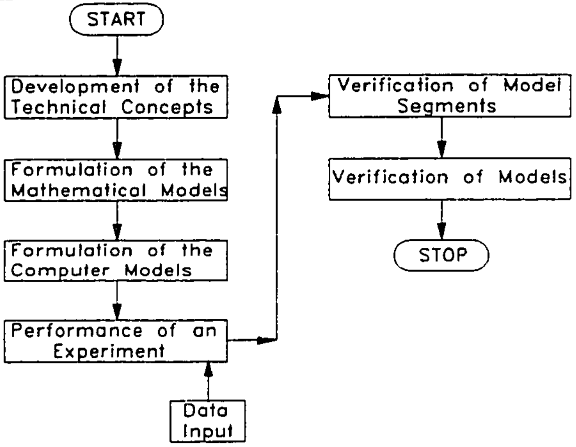

START
Development of the Technical Concepts
Formulation of the Mathematical Models
Formulation of the Computer Models
Performance of an Experiment
Data Input
Verification of Model Segments
Verification of Models
STOP

Figure 4.2 Flow Diagram for the Development of Computer Models

The next stage is the development of the mathematical models which were described in chapter 3. The mathematical equations of the models are then translated into computer models using FORTRAN 77 language. The models are then verified and cross-checked, segment by segment, to ensure that they are producing the required results, and are finally validated using real mine data. The input data is supplied, and an experiment is conducted. The analysis was carried out on the HP main frame at Dal Tech, Dalhousie University.

4.2.1 Computer Flow Charts and Description of Functions and Routines

The CS and PITSEARCH models are programmed using Fortran 77 language. WinNN 97 version neural networks software package is employed to implement the MFNN model. Time Series Processor (T.S.P) software is used in modelling the MRM model. Figure 4.3 depicts a general layout and interaction of the program files and software used in solving the model equations. All the program files are discussed in Appendix B.

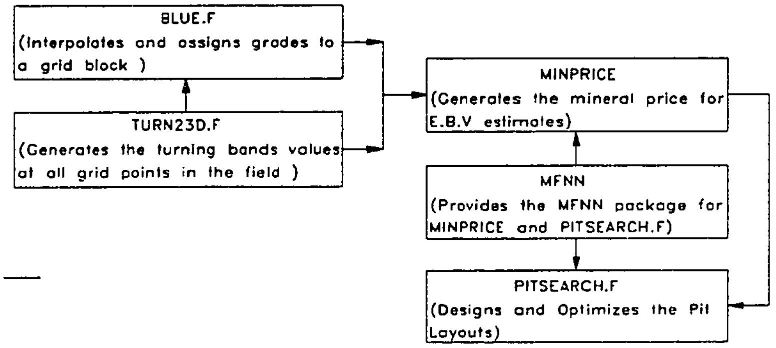

BLUE.F
(Interpolates and assigns grades to a grid block)
TURN23D.F
(Generates the turning bands values at all grid points in the field)
MINPRICE
(Generates the mineral price for E.B.V estimates)
MFNN
(Provides the MFNN package for MINPRICE and PITSEARCH.F)
PITSEARCH.F
(Designs and Optimizes the Pit Layouts)

Figure 4.3 General Layout of the Program Files.

4.2.1.1 TURN23D.F

TURN23D.F is a program for calculating two- and three-dimensions TBM values for the CS model to assign grades to gridded-blocks which have no grade values. This file contains functions and routines for simulating or generating a stationary, Gaussian random field in two- or three- dimensions of zero mean and a unit variance with a known spatial correlation function, spectral density or covariance function as described in chapter 3. It generates the line processes using the standard spectral integration method [Thompson et al, 1989]. The routines and functions as illustrated in Figure 4.4, are same in both cases, with a few modifications to run the algorithm strictly in either two- or three- dimensions. The semi-variograms, which give the covariance functions, were drawn and analyzed using VARIOC [Kuchta, 1993] and VARIOWIN [Pannatier, 1996].

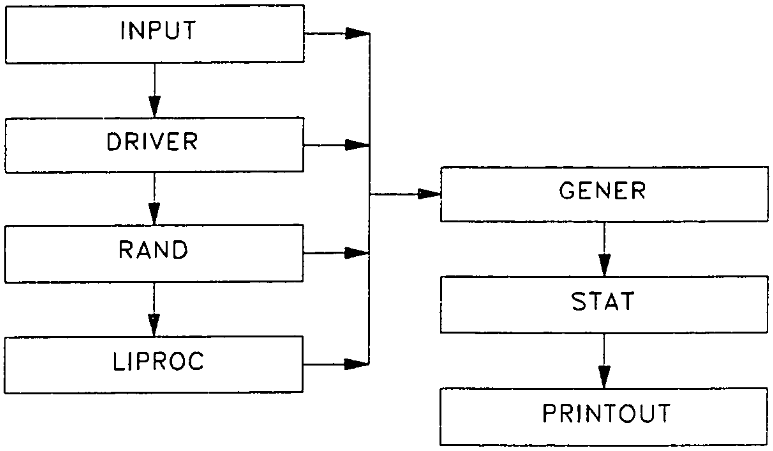


Figure 4.4 Routines and Functions in the TURN23D.

The function of each subroutine is described in the following.

INPUT: This subroutine reads all options and turning band inputs. The required data are read in subroutine INPUT in two groups from two files respectively. The first group contains physical information regarding the size of the grid, and the second contains parametric data related to the simulation itself.

TURN1.DAT: It is the file that contains physical information regarding the size of the grid for the 3-D simulation.

TURN2.DAT: It is the file that contains parametric data related to the simulation itself for the 3-D simulation.

TURN2D1.DAT: This file is analogous to and contains the same information as TURN1.DAT for the 2-D simulation.

TURN2D2.DAT: This file is analogous to and contains the same information as TURN2.DAT for the 2-D simulation

DRIVER: This subroutine sets up the blank common storage and file assignments for the simulation.

GENER: This subroutine generates random field with turning bands method.

LIPROC: This subroutine generates a correlated line process using the methods of Rice (1954); Shinozuka and Jan (1972).

RAND: This function generates pseudo random numbers, uniform (0,1).

STAT: This subroutine calculates sample statistics of the random field.

PRINTOUT: This subroutine prints out the results to unformatted data file(s) which is also used in BLUE.F.

4.2.1.2 BLUE.F

This file, as illustrated in Figure 4.5, contains functions and routines for solving a linear system of equations using Gaussian elimination. The functions and routines are described in the following.

KALPHA.F: This subroutine generates the elements in matrix $[\mathbf{K}_\alpha]$. The elements are the covariances between the known sample points and are calculated given the variogram model (covariance function) of the sample data.

CA.DT: The input file of KALPHA.F and contains known sample points coordinates and grades.

CA.UT: The output file of KALPHA.F and contains the calculated covariances

MBETA.F: This subroutine generates the elements in matrix $[\mathbf{M}_\beta]$. The elements are the covariances between the unknown sample and known sample points and are also calculated given the variogram model (covariance function) of the sample data.

CB.DT: The input file of MBETA.F and contains center coordinates of blocks with unknown grades.

CB.UT: The output file of MBETA.F and contains the calculated covariances

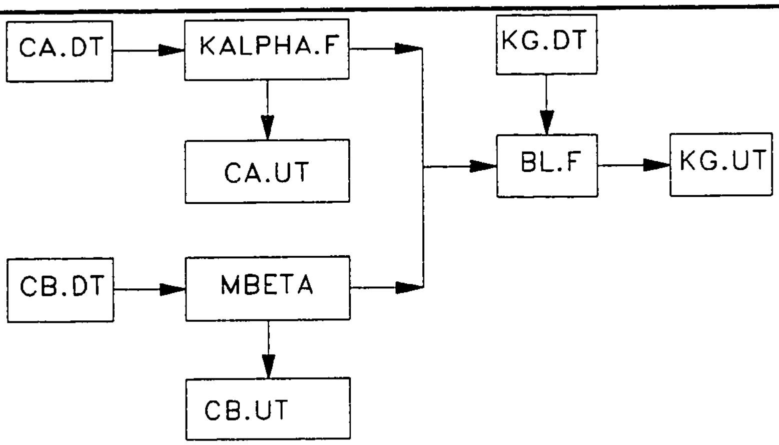


Figure 4.5 Routines and Functions in the BLUE.

4.2.1.3 MINPRICE

In this study, the multiple regression technique is used to develop a mineral price model for predicting future gold prices. The data within a determined socio-politico-economic trend is used as input into the multiple regression model to estimate the coefficients of the respective parameters.

This file contains functions and routines for simulating the mineral price model. The routines and functions are illustrated in Figure 4.6. The stochastic gold price is modelled using a two-stage procedure. Stage one involves using TSP to estimate the coefficients the multiple regression equation for predicting the mineral commodity prices. The data on the mineral price determinants within the socio-politico-economic cycle is what is used to construct the multiple regression equation.. In the second stage, the socio-politico-economic cycle data is fed into multilayer feedforward neural networks (MFNN) to determine the average-annual monthly high and low gold prices for each year during the life of the mine. The world annual gold production, world annual gold consumption, average-annual monthly high price and average-annual monthly low gold price data for each year during the life of the mine is substituted to forecast the mineral prices.

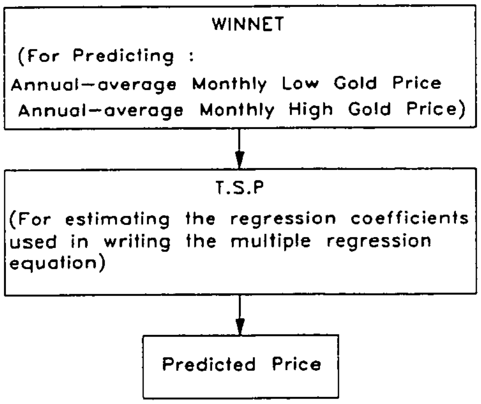

WINNET
(For Predicting :
Annual—average Monthly Low Gold Price
Annual—average Monthly High Gold Price)
T.S.P
(For estimating the regression coefficients used in writing the multiple regression equation)
Predicted Price

Figure 4.6 Routines and Functions in the MINPRICE Model.

T.S.P.: This is the Time Series Processing software with a regression package for multiple regression calculations. A program was written to suit this study.

WINNET: This is WinNN version 0.97, Windows Neural Networks, a neural network package that will train a fully connected feedforward networks with the back propagation algorithm. It has a user friendly interface written in Visual Basic and uses a FORTRAN DLL for fast network calculations. This is used for the multilayer feedforward neural networks calculation.

4.2.1.5 PITSEARCH.F

This is the program for searching and calculating the optimum pit value. It is used in both CS/MFNN and Learchs-Grossmann models. It implements the pit optimization algorithms. The interactions of the programs, input and output files for CS/MFNN model are illustrated in Figure 4.7. This program imposes the slope wall constraints. This is done by MINEBLOCKS.F and LG23D.F for the CS/MFNN and Learchs-Grossmann models respectively. The interactions of the programs and routines for the Learchs-Grossmann model are illustrated in Figure 4.8.

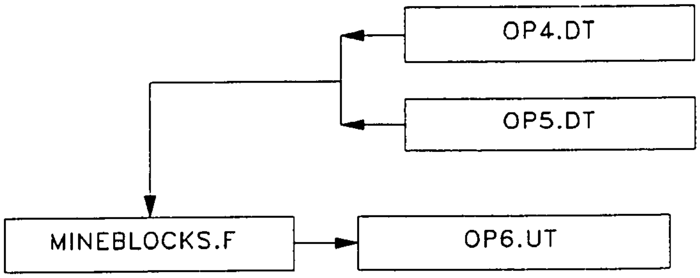


Figure 4.7 Routines and Functions in the PITSEARCH.F for CS/MFNN Model.

The functions and routines in the PITSEARCH.F are described in the following.

MINEBLOCKS.F: This is the routine for the MFNN 2- and 3-D pit optimization algorithm.

LG23D.F: This is the routine for the Learchs-Grossmann 2- and 3-D pit optimization algorithm.

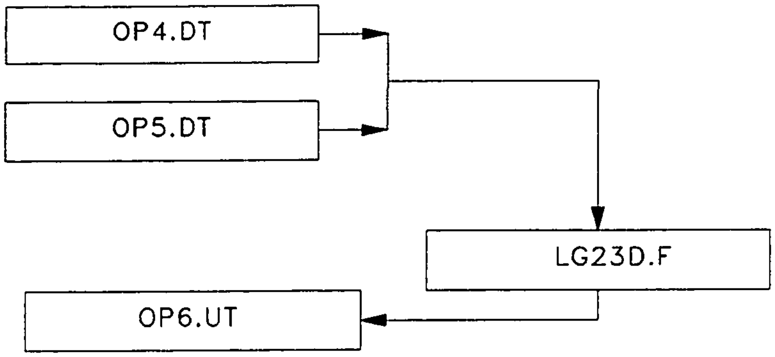


Figure 4.8 Routines and Functions in the PITSEARCH.F Learchs-Grossmann Model.

OP4.DT: This is the input file containing the economic block values of all the grid blocks.

OP5.DT: This is the input file containing the number and sizes of all the grid blocks.

OP6.UT: This is the output file of the results on the optimum pit.

4.2.2 Verification

The programs were verified, segment by segment, using a debugging tool on the HP main frame to ensure that the respective results were accurate. The analytically tractable problems were calculated employing EXCEL 5.0 to ensure that the computer results matched the EXCEL 5.0 results.

4.2.3 Summary

The solution procedure and necessary computing software for this study have been provided. The computer flow charts and the experimentation process are also provided. The programs BLUE.F, PITSERACH.F, MRM TSP and their accompanying routines were written by the author. TURN23D.F was supplied by Dr. A. F. Thompson of Geosciences and Environmental Technologies Division, Livermore California. LG23D.F was adopted from Dr. P.A. Dowd (1994) and revised to suit this study.

CHAPTER 5.0

EXPERIMENTATION AND VALIDATION OF THE MODELS

In this chapter, all the models presented in chapter 3 and computer programs in chapter 4 are validated using data from an actual gold mine. The CS/MFNN and Lerchs-Grossmann algorithms are applied to a 2-D and 3-D sections of Star Gold Project[1] to optimize the pit limits to evaluate its economic potential, and compare the results. Furthermore, the mineral price model is validated using gold price data from the spot markets. Ore grade and gold price are the two most important random variables studied in this experimentation as EBV of gridded-block is dependent on them. Experiment on drill-hole spacing is carried out to see the optimum drill-hole spacing range that will give the best estimate of the gridded-blocks grades, optimum value and limit.

5.1 Background Information on the Star Gold Project

The total tonnage of mineable ore reserves is estimated at around 15 million tonnes at an average grade of about 2 grammes of gold per tonne of ore. The Star gold deposit is within a ridge which is in a basement gneiss along N-S fault. The outcropping mineralized body is essentially contained within two sub-parallel, almost vertical shears varying from about 50 to 100 meter apart. The orebody extends for more than 2 km with a north-ward trend, mainly at or near the crests of the ridge. The gold is made up of 20-25 % free gold and the rest is associated with pyrites. Other associated minerals include chalcopyrite, bornite, calcite, epidote, magnesite, and the main gangue mineral is quartz. Past mining activities have been confined to auriferous pods aligned along stressed-relief fractures running at a few degrees east of north. These pods are discontinuous horizontally and vertically, pitching steeply to the north, and are highly siliceous. Average overburden

[1] The name and location of project cannot be revealed for confidential purposes.

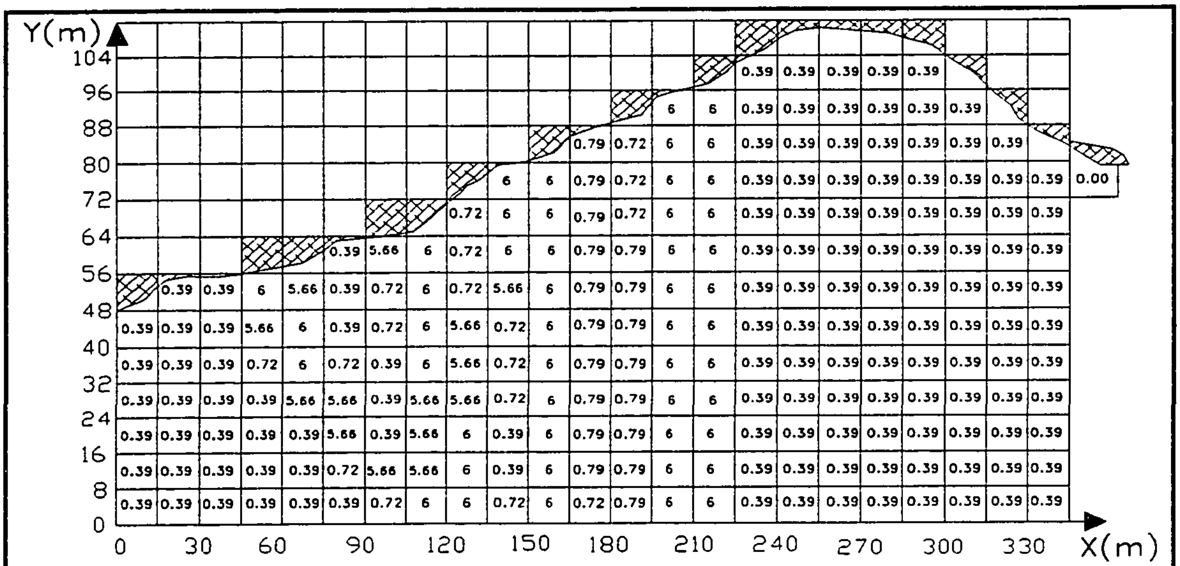
thickness is about 0.5 m and the oxidized zone (with ore amenable to heap leaching) persists to a depth of about 30 m. The bank density of the ore is in the order of 2.76 tonnes/m^3, and the loose density is 1.700 tonnes/m^3. The rock quality designation (RQD) is about 40 %. Metallurgical tests on the ore have proved that heap leaching can be a suitable method for processing the ore. During processing, recoveries of between 60-76% have been reported and further tests are being carried out to identify various strategies to be used to achieve higher recoveries. Three analytical methods: atomic absorption, ketone (DIBK) and fire assay, were used for the analysis to reduce estimation errors and increase the confidence of analysts in the assayed values.

5.2 Data and Information for Validation and Experimentation

Optimized pit limits are required for the 2-D and 3-D sections of Star gold deposit, depicted in Figure 5.1 and Figure 5.2 respectively. The block dimensions are 15m × 8m × 8m and the weighted tonnage factor of material is 2.76 tonne/m^3. The mill recovery efficiency is 95 %. The average cost for mining ore and waste, pit wall slope and administrative overheads are respectively \$4.45 /tonne, 45° and 6% on gross revenues. The coordinates of all the blocks at their centers of gravity and grades are depicted in Table B1 and Table B2. Furthermore, it is assumed that production of gold is going to increase by 3% of 1994 production, consumption will increase by 2% of 1994 consumption. The identified socio-politico-economic cycle data for use in the mineral price forecast model spans from 1980 to 1994 (Table 5.1).

Table 5.1 World Mineral Production, Consumption and Price Data

YEAR	PRODUCTION	CONSUMPTION	MONTHLY	MONTHLY	PRICE
			HIGH	LOW	
	[tonnes]	[tonnes]	[US $]/oz	[US $]/oz	[US $]/oz
1980	960.80	946.00	850.00	481.50	614.38
1981	977.30	1214.00	599.25	391.25	459.22
1982	1025.30	1253.00	481.00	296.75	375.52
1983	1113.70	1218.00	509.25	374.50	423.52
1984	1160.00	1463.00	399.25	307.50	360.63
1985	1230.30	1469.00	340.90	284.25	317.35
1986	1292.50	1686.00	438.10	326.30	367.58
1987	1734.40	1689.00	499.75	390.00	446.66
1988	1908.90	1942.00	483.90	395.30	436.45
1989	2067.60	2342.00	415.80	355.75	381.27
1990	2138.10	2478.00	423.75	345.85	383.72
1991	2167.30	2592.00	403.00	344.25	362.34
1992	2249.90	2890.00	359.60	330.25	343.86
1993	2308.90	2765.00	405.60	326.10	360.06
1994	2296.40	2769.00	396.25	369.65	384.15



															0.39	0.39	0.39	0.39	0.39				
													6	6	0.39	0.39	0.39	0.39	0.39	0.39			
											0.79	0.72	6	6	0.39	0.39	0.39	0.39	0.39	0.39	0.39		
									6	6	0.79	0.72	6	6	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.00
								0.72	6	6	0.79	0.72	6	6	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	
					0.39	5.66	6	0.72	6	6	0.79	0.79	6	6	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	
	0.39	0.39	6	5.66	0.39	0.72	6	0.72	5.66	6	0.79	0.79	6	6	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	
0.39	0.39	0.39	5.66	6	0.39	0.72	6	5.66	0.72	6	0.79	0.79	6	6	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	
0.39	0.39	0.39	0.72	6	0.72	0.39	6	5.66	0.72	6	0.79	0.79	6	6	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	
0.39	0.39	0.39	0.39	5.66	5.66	0.39	5.66	5.66	0.72	6	0.79	0.79	6	6	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	
0.39	0.39	0.39	0.39	0.39	5.66	0.39	5.66	6	0.39	6	0.79	0.79	6	6	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	
0.39	0.39	0.39	0.39	0.39	0.72	5.66	5.66	6	0.39	6	0.79	0.79	6	6	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	
0.39	0.39	0.39	0.39	0.39	0.39	0.72	6	6	0.72	6	0.72	0.79	6	6	0.39	0.39	0.39	0.39	0.39	0.39	0.39	0.39	

Figure 5.1 Blocks Grades for 2-D Section (g / Ton).

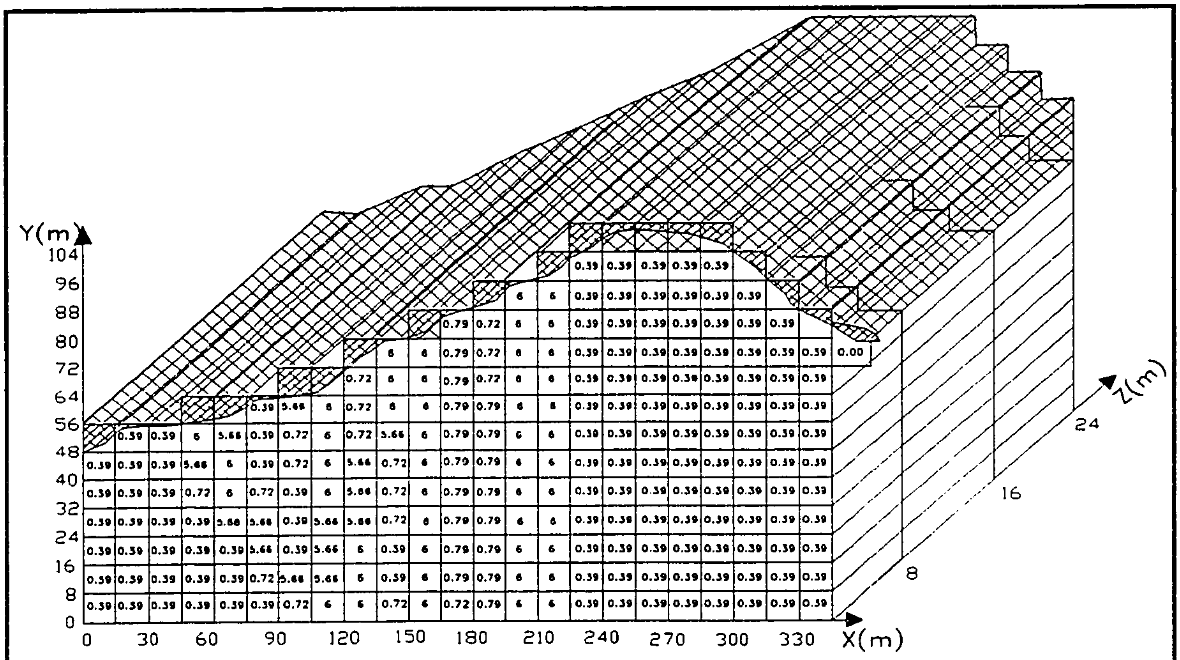


Figure 5.2 Blocks Grades for 3-D Section (g / Ton).

The information is to be used to carry out the following experiments: (i) to predict the mineral price for the economic block values calculation in the experimentation using the mineral price forecast model; (ii) to investigate the effect of variability of sample space on CS/MFNN and Lerchs-Grossmann algorithms; and (iii) to test for speed of calculation by the two algorithms.

The main tasks are to (i) forecast gold price using the mineral price model in chapter 3; (ii) estimate the grades, tonnages and economic block values of some selected blocks using CS method; (iii) optimize the pit limits using MFNN and Lerchs-Grossmann's algorithms to evaluate its economic potential; (iv) compare and analyze the results in (iii).

5.3 Mineral Price Experiment

The main objective in this experiment is to test for the effect of number of generated data on the mineral price forecast. 1995 data on the determinant factors is used to validate the model. The forecasting error and correlation between the actual price data and predicted prices are also investigated. Stage one of the experiment involves using the few data on the mineral price determinants in Table 5.1 within the socio-politico-economic cycle as input in MRM model to construct the multiple regression equation. In the second stage, the data on the determinant factors within the identified socio-politico-economic cycle is fed into multilayer feedforward neural networks (MFNN) to determine the average-annual monthly high and low gold price for each year during the life of the mine.

5.3.1 Description of Input Data

The input data for this experiment is the fifteen set of data of the identified socio-politico-economic cycle spanning 1980 to 1994 in Table 5.1. The data is on world (i) annual production of gold; (ii) annual consumption; (iii) average-annual gold price; (iv) average-monthly high gold price; and (v) average-monthly low gold price. The data are contained in the input file INP.DAT for the TSP program.

5.4 Sample Space Experiment

5.4.1 Variation of Sample Space

The information on Star Gold Project is used to carry out the following experiments to analyze and validate the CS/MFNN model. The experiments are (i) to test for the effect of variability of sample space on CS/MFNN and Lerchs-Grossmann algorithms; and (ii) to test for speed of calculation by the two algorithms. In the experimentation of the effect of sample space variability on CS/MFNN and Lerchs-Grossmann algorithms, six sets of experiments are conducted. The number of known blocks and/or distance between drill-hole spacing in the deposit are varied. The drill-hole spacing are 45m, 60m, 75m , 90m, 105m and 135m. The gold price calculated from the previous section is used in calculating the economic block values of all the pits for the six data sets.

Flow diagram of the experiment is shown in Figure 5.3. The first stage of the process is to use twelve data sets from the Star Gold Project data, six data set each for 2-D and 3-D respectively, in Table B8 to Table B19 at appendix B as input data in both algorithms to optimize the 2-D and 3-D sections. The multi-layer feedforward neural networks model (MFNN) in Figure 5.4 is used to classify the conditioned gridded-blocks into classes based on their block coordinates and target values in Figure 5.5.

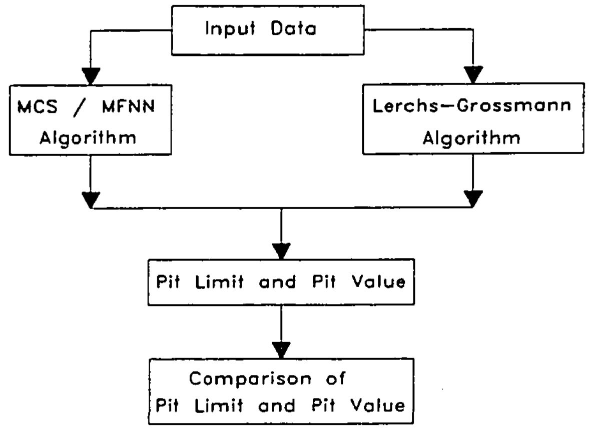

Input Data
MCS / MFNN Algorithm
Lerchs–Grossmann Algorithm
Pit Limit and Pit Value
Comparison of Pit Limit and Pit Value

Figure 5.3 Flow Diagram of Experiment

The optimum pit values and pit outlines returned by each of the two algorithms are analyzed to see how they compare with each other. In Figure 5.4, a_1 and a_2 are input layer neurons, N_1, N_2, N_3, N_4, N_5, N_6, N_7 and N_8 are hidden layer neurons and O is the output layer neuron. b_1 and b_2 are biases equivalent to -1 each. X_1 and X_2 are the x and y coordinates respectively and are inputs of the network. O_{pj} are the target values which are 1, 0 or -1 for positive, zero and negative economic block values respectively.

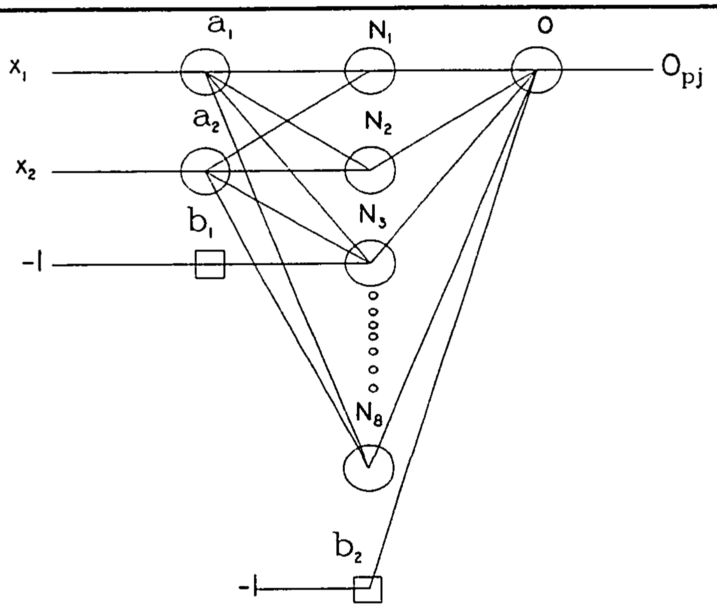
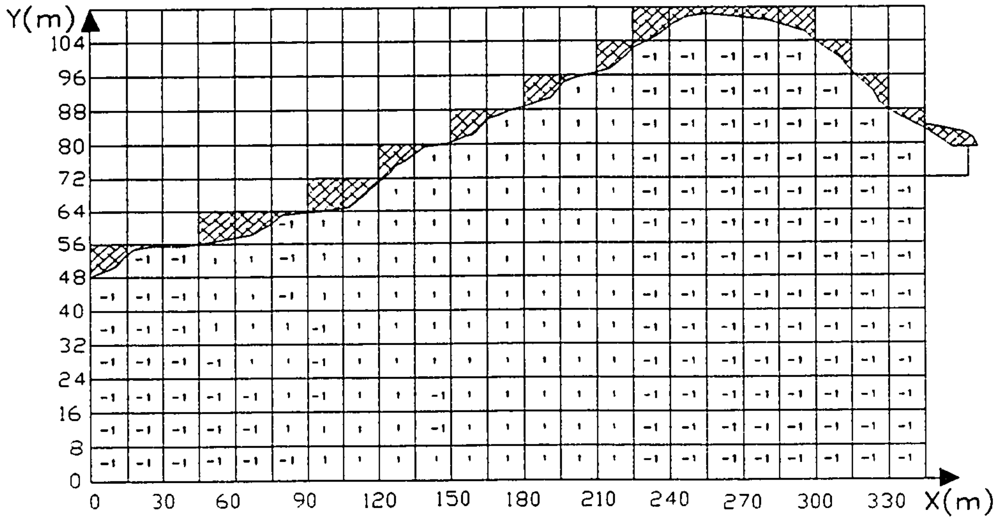


Figure 5.4 Neural Networks for Block Classification.



															-1	-1	-1	-1	-1			
													1	1	-1	-1	-1	-1	-1	-1		
											1	1	1	1	-1	-1	-1	-1	-1	-1	-1	
									1	1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	-1
								1	1	1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	-1
					-1	1	1	1	1	1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	-1
	-1	-1	1	1	-1	1	1	1	1	1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	1	1	-1	1	1	1	1	1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	1	1	1	-1	1	1	1	1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	1	1	-1	1	1	1	1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	1	-1	1	1	-1	1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	1	1	1	1	-1	1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	1	1	1	1	1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	-1

Figure 5.5 Target Values of Blocks.

5.4.2 Description of Input Data

The data in Figure 5.1 and Figure 5.2 consist of a set of gold analyses from the Star Project open pit mine. These analyses are estimates of the overall compositions of mining blocks, each obtained from a composite of the blast-hole samples collected within a block. They are used as known samples to find the values of other gridded blocks.

The input data for this experiment are (i) the center coordinates and grades of all the blocks with known sample grades contained in file CA.DT, for fitting variogram model to the data using CSMINE/VARIOC and VARWIN softwares; (ii) The parameters for the CS model are contained in files TURN2D1.DAT and TURN2D2.DAT for 2-D, and TURN1.DAT and TURN2.DAT for 3-D TBM model. The parameters of the input files for the 2-D and 3-D TBM models are listed in Table 5.2 and Table 5.3 respectively.

Table 5.2 Input Files for the 2-D CS

TURN2D1.DAT	
Simulation Date	060997
Number of Grid Points in the x, y and z Directions	23, 13, 1
Grid Spacing in the x, y and z Directions	15, 8, 0
Covariance Function	Spherical
TURN2D2.DAT	
Number of Lines to be used	16
Maximum Normalized Frequency	50
Normalized Frequency Increment	0.2
Number of Simulations to Perform	1000
Seed for the Uniform Random Number Generator	210467

File TURN1.DAT and TURN2D1.DAT contain physical information regarding the size of the grid for 2-D and 3-D simulation respectively. The information are (i) simulation date; (ii) the number of grid points in the x, y and z directions; (iii) the grid spacing in the x, y and z directions; and (iv) covariance function.

Table 5.3 Input File for the 3-D CS

TURN1.DAT	
Simulation Date	060997
Number of Grid Points in the x, y and z Directions	23, 13, 3
Grid Spacing in the x, y and z Directions	15, 8, 8
Covariance Function	Spherical
TURN2.DAT	
Number of Lines to be used	15
Maximum Normalized Frequency	100
Normalized Frequency Increment	0.2
Number of Simulations to Perform	1000
Seed for the Uniform Random Number Generator	210467

Files TURN2.DAT and TURN2D2.DAT contain parametric data related to the simulation itself for the 2-D and 3-D simulation respectively. The data are (i) the number of lines to be used; (ii) the maximum normalized frequency used in the standard spectral integration method; (iii) the normalized frequency increment; (iv) the number of simulations to perform; and (v) an arbitrary, odd integer, less than 2^{20}, to be the seed for the uniform random number generator in the function RAND. The center coordinates and grades of the set of gridded blocks selected as known samples are listed in Tables B8 to Table B19.

5.5 Summary

The details of the experimentation and validation have been described. The input data for all the programs and softwares have been prepared. The results and discussion of the experimentation and validation are discussed in chapter 6.

CHAPTER 6.0

RESULTS AND DISCUSSION

6.1 Mineral Price Experiment

Using the MFNN model illustrated in Figure 3.8, with the 15 data of world annual gold production and annual gold consumption in Table 5.1 as input, the monthly high and monthly low were predicted to be US$360 per ounce and US$325 per ounce respectively. The error and momentum term for complete training were 0.01 and 0.36 respectively. The MRM model yielded the regression coefficients tabulated in Table 6.1.

Table 6.1 MRM Results

Variable Name	Variable	Coefficient	Estimated Coefficient
Constant	β_1	β_1	54.809
Production	P	β_2	0. 042
Consumption	C	β_3	-0. 036
Monthly High	H	β_4	0.369
Monthly Low	L	β_5	0.490
R^2	99.1 %		

Equation (6.1) is the resulting equation using the regression coefficients in Table 6.1. According to equation (6. 1) the gold price is forecasted to be US$344.56 per ounce . This is the gold price used in the calculation of E.B.Vs for all the open pits.

$$\textbf{Gold Price} = 54.809 + 0.042\,P - 0.036\,C + 0.369\,H + 0.490\,L \qquad (6.1)$$

The negative estimated coefficient of world gold consumption depicts that it has negative effect on price and that if the price of gold falls, more quantity of gold will be bought. This confirms the basic economic principle of demand that the lower the gold price, the higher the consumption and vice versa. The positive estimated coefficient of world gold production means it has positive effect on price and that when the price of gold is high, the producers would want to produce more. This reaffirms the basic economic principle of supply that the higher the gold price, the higher the production and vice versa. R^2 is 99.1 %. This means that the MRM is able to explain 99.1 % of the data used. This underscores the degree of accuracy of the model and should encourage potential gold price forecasters to use.

When all the original 15 data of the independent variables are substituted in the multiple regression equation, it was found out that the error was negligible. The results are as shown in Table 6.2. The variation of the actual prices and the predicted ones by the model is shown in Figure 6.1. The actual prices and the predicted ones are almost super-imposed on each other.

Table 6.2 Actual and Predicted Gold Prices

Year	Actual Price (US $)/oz	Predicted Price (US $)/oz
1980	614.38	610.62
1981	459.22	465.07
1982	375.52	375.76
1983	423.52	429.21
1984	360.63	349.02
1985	317.35	318.81
1986	367.58	370.14
1987	446.66	442.38
1988	436.45	437.40
1989	381.27	385.28
1990	383.72	381.46
1991	362.34	370.19
1992	343.86	340.16
1993	360.06	361.99
1994	384.15	379.22

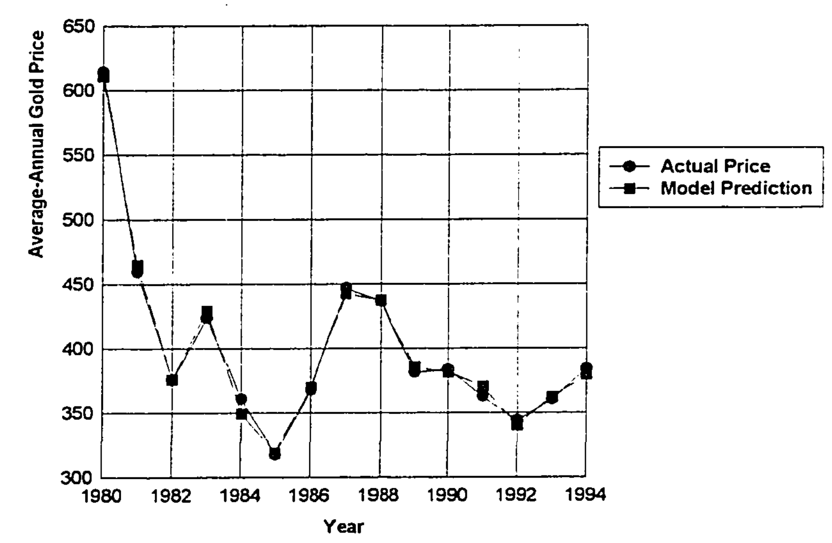


Figure 6.1 Variation of Actual and Model Predicted Prices

Also the mean, variance and standard deviation of the actual and the predicted prices are almost the same. Statistics of actual and predicted prices are shown in Table 6.3.

Table 6.3 Statistics of Actual and Predicted Prices

Statistics	Actual Prices (US $)/oz	Predicted Prices (US$)/oz
Mean	401.11	401.11
Variance	71.00	70.81
Standard Deviation	5040.69	5014.41

The mineral price model was used to forecast 1995 average-annual gold price to test its prediction accuracy and capability. The 1995 actual data on world annual gold production, consumption, average-monthly high price, average-monthly low price and average price are 2272.10 tonnes, 3008.00 tonnes, US$391.03 per ounce, US $376.64 per ounce and US $384.17 per ounce respectively. The networks training parameters of the MFNN model used in the prediction of average-annual monthly high and low gold prices are 0.01, 0.865 and 0.532 for the error, momentum and learning rate respectively. The model predictions of the 1995 average-annual gold price, average-annual monthly high gold price and average-annual monthly low gold price are US$376.21 per ounce, US$395.12 per ounce and US$384.62 per ounce respectively. When the MRM model is used to forecast the fifteen actual prices used in constructing the model, The prediction error in forecasting the 1995 average-annual gold price is 2.07 %.

6.2 Sample Space Experiment

6.2.1 Statistics and Variography

Statistics and semi-variograms for each of the experimental data set in Tables B8 to Table B19 was calculated with the aid of CSMINE/VARIOC and VARWIN softwares. Semi-variograms at 0°, 60°, 90°, 120° and 150° were calculated with each having a horizontal window of 25°. This made it possible to determine conclusively if there were anisotropy or not. Also, this insured that the samples in one direction were not used in calculating the variances in another direction. None of the data sets showed clear signs of anisotropy. Spherical model was fitted to all the data set. Both the experimental and theoretical/model semi-variograms for all the data sets are illustrated in Figures 6.2 to 6.13.

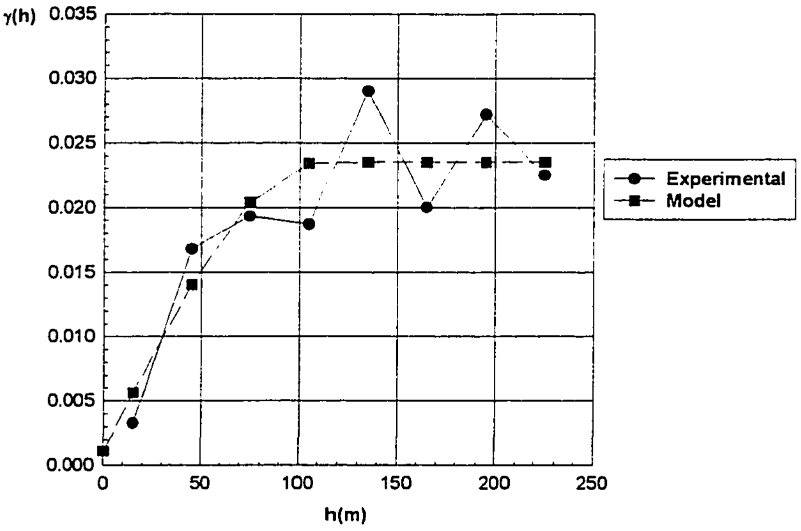


Figure 6.2 2-D Omnidirectional Semi-variogram for 45m Drill-hole Spacing

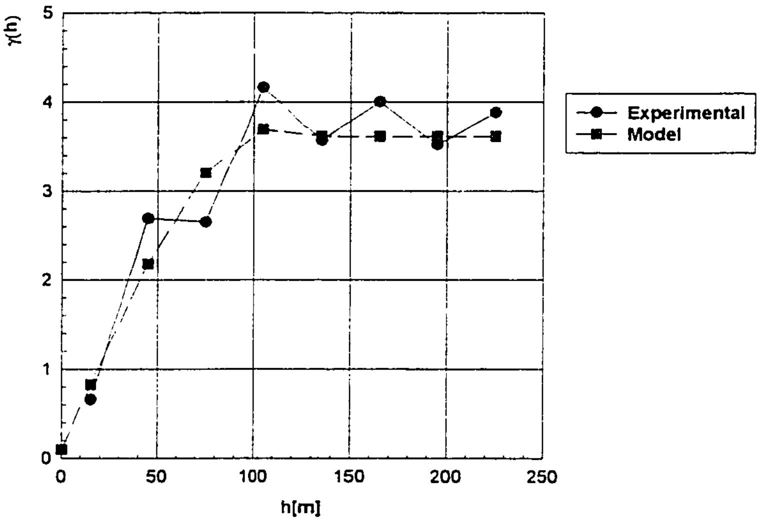


Figure 6.3 2-D Omnidirectional Semi-variogram for 60m Drill-hole Spacing

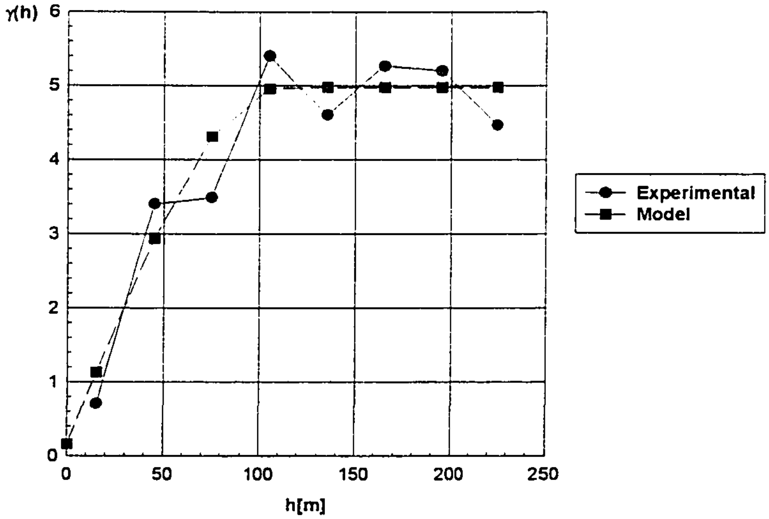


Figure 6.4 2-D Omnidirectional Semi-variogram for 75m Drill-hole Spacing

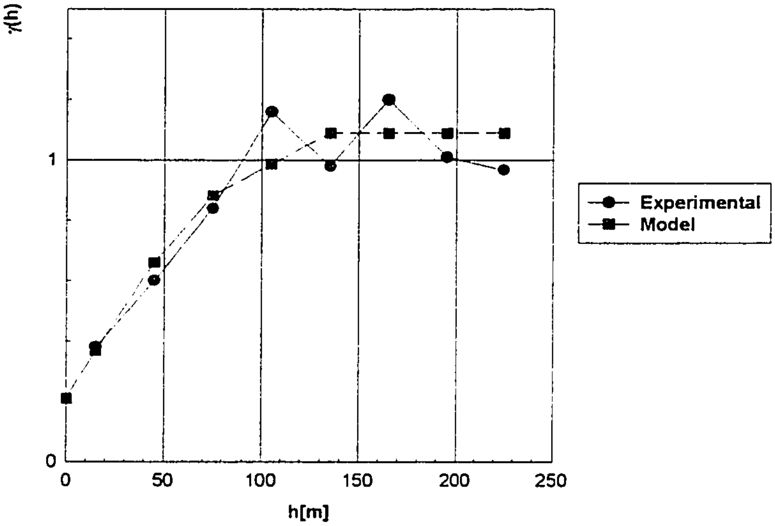


Figure 6.5 2-D Omnidirectional Semi-variogram for 90m Drill-hole Spacing

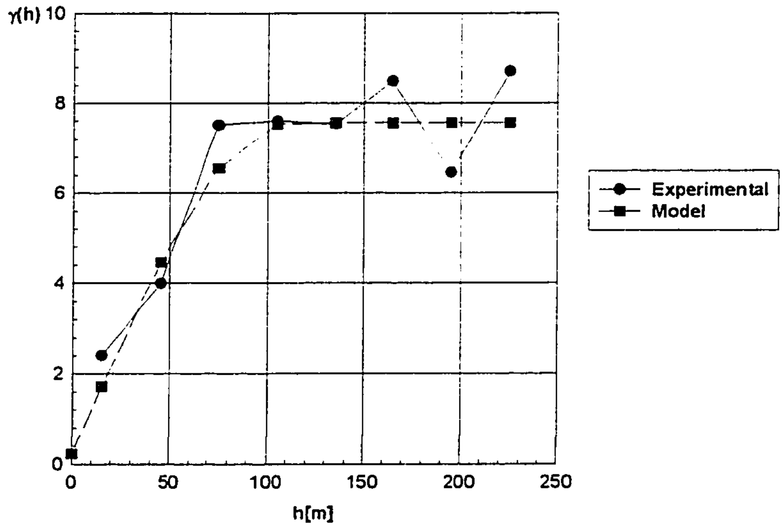


Figure 6.6 2-D Omnidirectional Semi-variogram for 105m Drill-hole Spacing

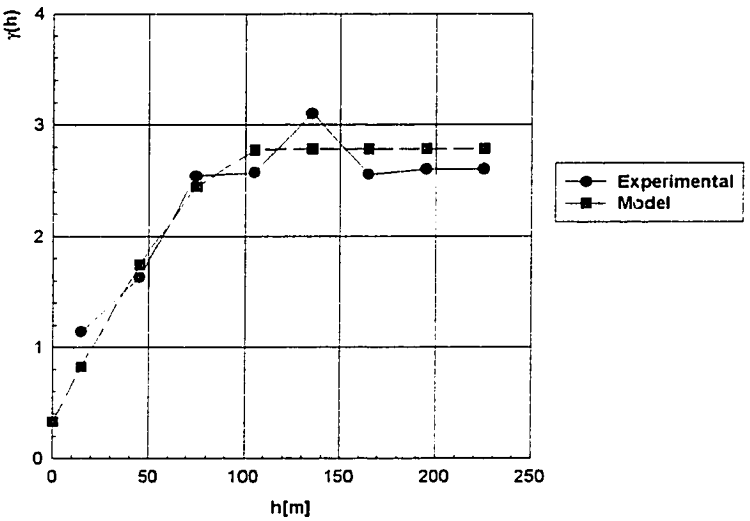


Figure 6.7 2-D Omnidirectional Semi-variogram for 135m Drill-holes Spacing

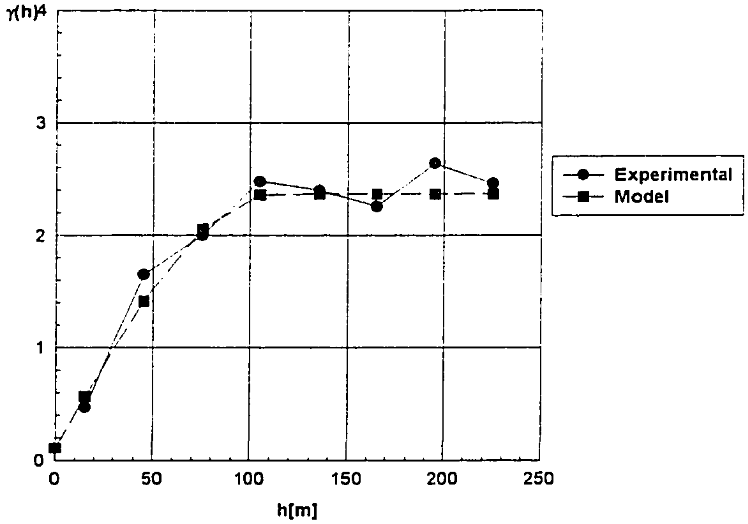


Figure 6.8 3-D Omnidirectional Semi-variogram for 45m Drill-hole Spacing

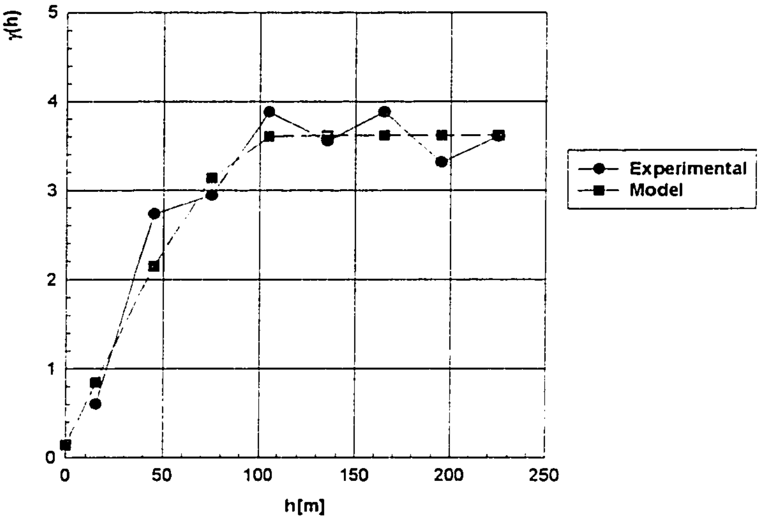


Figure 6.9 3-D Omnidirectional Semi-variogram for 60m Drill-hole Spacing

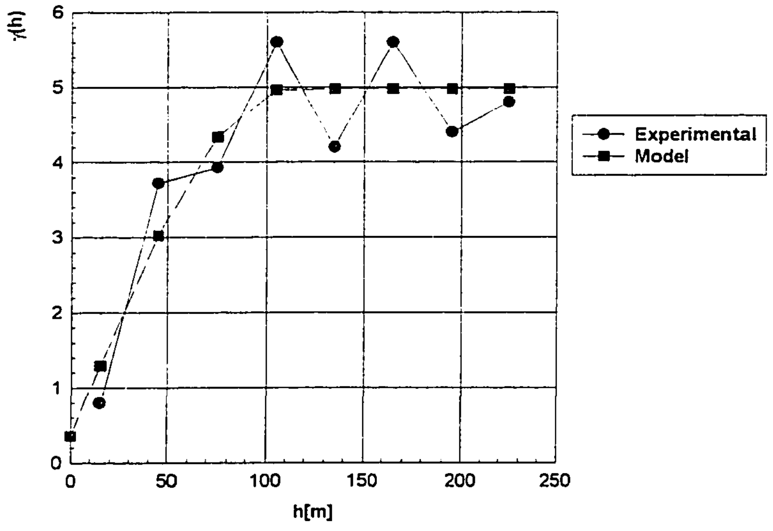


Figure 6.10 3-D Omnidirectional Semi-variogram for 75m Drill-hole Spacing

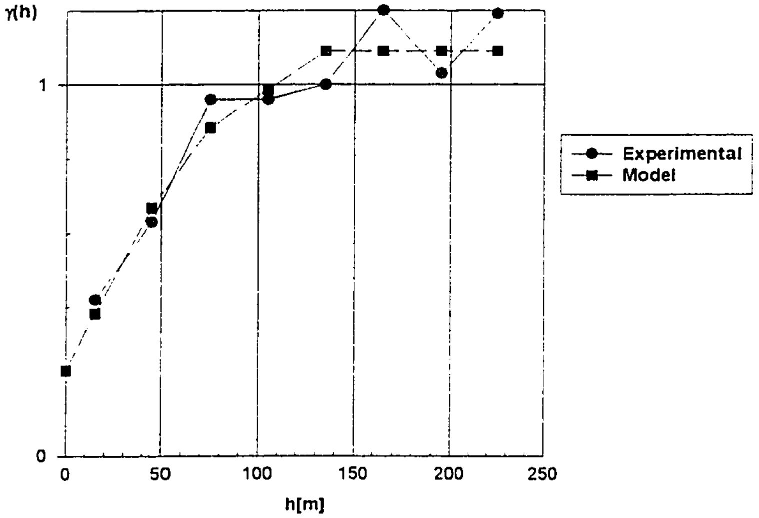


Figure 6.11 3-D Omnidirectional Semi-variogram for 90m Drill-hole Spacing

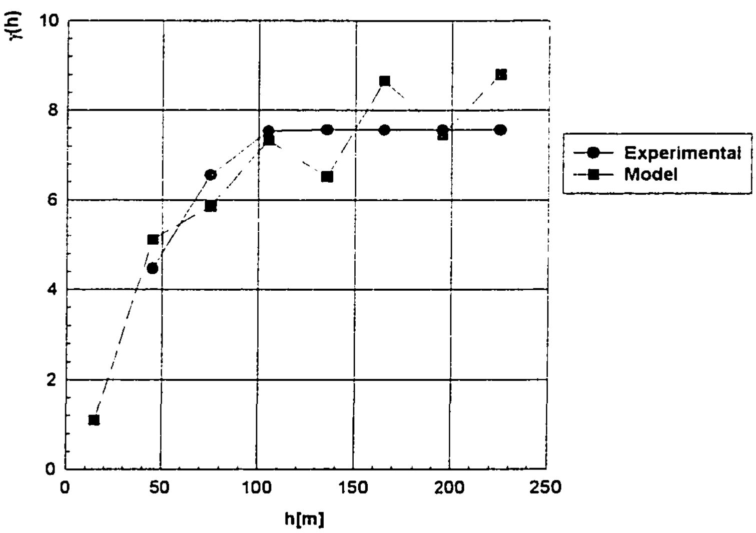


Figure 6.12 3-D Omnidirectional Semi-variogram for 105m Drill-hole Spacing

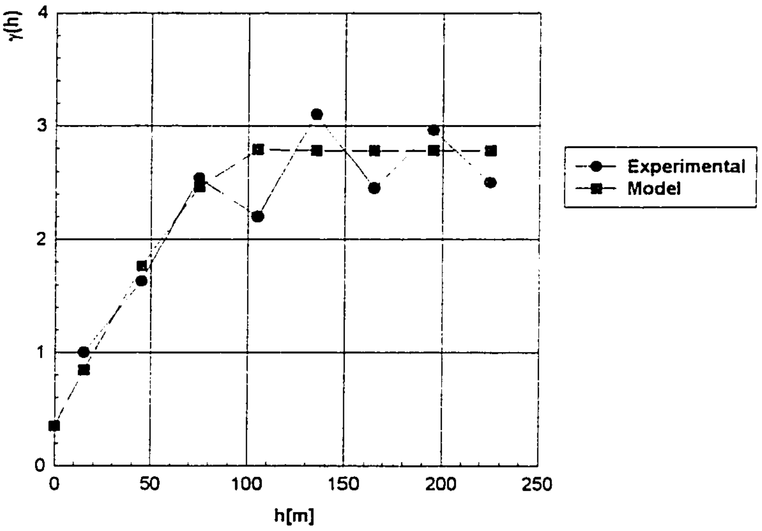


Figure 6.13 3-D Omnidirectional Semi-variogram for 135m Drill-holes Spacing

The CS yielded the block grades listed in Tables D2 to Tables D13. Employing the average error $\overline{E}$, variance of the errors σ_e^2 and estimator unbiasedness, the prediction accuracy of CS can be measured. $\overline{E}$ and σ_e^2 are given by equations (6.2) and (6.3) respectively (Knudsen, 1990).

$$\overline{E} = \frac{1}{n}\sum_{i=1}^{n}(X_i - X_i^{*}) \qquad \textbf{(6.2)}$$

$$\sigma_e^2 = \frac{\sum_{i=1}^{n}[(X_i - X_i^{*}) - \overline{E}]^2}{n-1} \qquad \textbf{(6.3)}$$

where X_i, X_i^{*} and n are actual block grades, estimated block grades and number of estimated block grades respectively. $\overline{E}$ and σ_e^2 values of all the estimations are calculated and tabulated in Tables B20 and B21. The variation of the drill-hole spacing with $\overline{E}$ and σ_e^2 in both 2-D and 3-D are depicted in Figure 6.14 and Figure 6.15 respectively. From Figure 6.14 and Figure 6.15, it is evident that the 45m, 60m, 75m and 90m drill-hole spacing have lower error and variance than 105m and 135 m drill-hole spacing. This is due to the reduction in the number of observed data with increase in drill-hole spacing. Subsequently, the experimental variograms constructed with 105m and 135m drill-hole spacing data do not give a true picture of the deposit. The field dimensions of the deposit used in the experimentation is 345 m × 24 m. Using drill-hole spacing from 45m to 90m gives pit limits equal to the actual one and optimum pit value a slightly less than the actual one. The ratio of the minimum drill-hole spacing (45m) to the respective length (345m) is 0.13. The ratio of the maximum drill-hole spacing (90m) to the respective length (345m) is 0.26. Based upon these results, it can be said that drill-hole spacing of between 0.13 and 0.26 of the field is enough to predict block grades, optimum pit limit and value with less error.

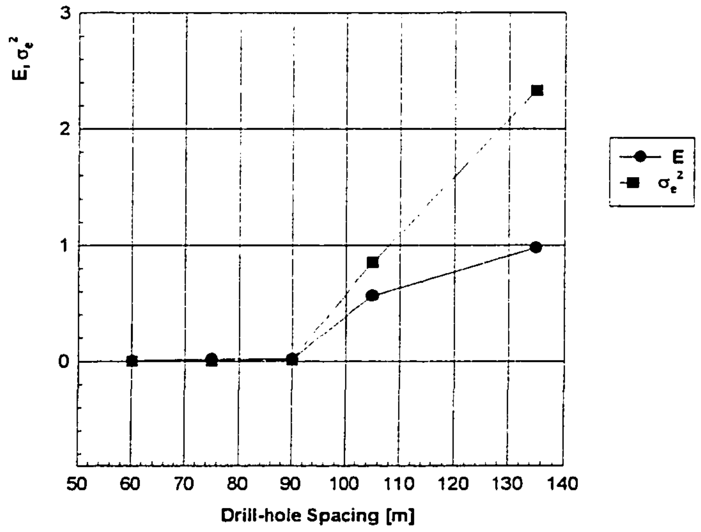


Figure 6.14 Variation of E and σ_e^2 with Drill-hole Spacing for 2-D

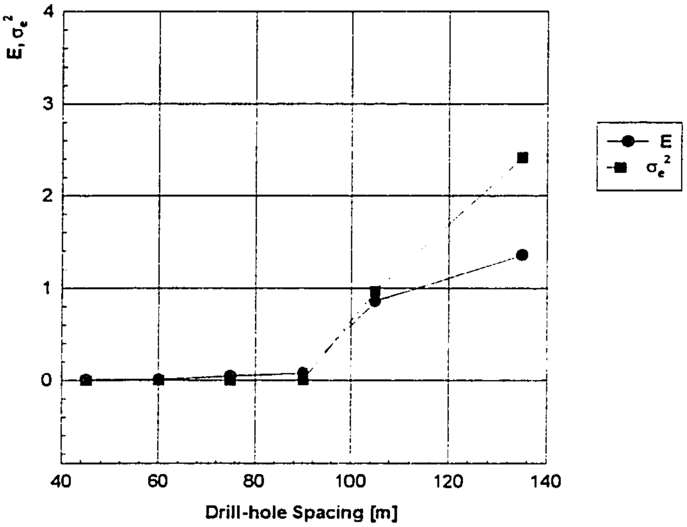


Figure 6.15 Variation of E and σ_e^2 with Drill-hole Spacing for 3-D

2-D Model Results

Using the predicted price, $346.15, the corresponding E.B.V of all the blocks were calculated and listed in Table D2 to Table D13. The actual data of the block model was used as input in the MFNN in Figure 6.14. Depicted in Table 6.4 are the final weights obtained by using the WinNN software package and applying error back-propagation training algorithm to the networks. The final weights are used in writing the equations of the decision lines for the separation of the gridded blocks in the pits. The error and momentum term for complete training were 0.01 and 0.685 respectively. The equations of the decision lines obtained from the MFNN in Figure 6.16 are as written in equations (6.4) to (6.11).

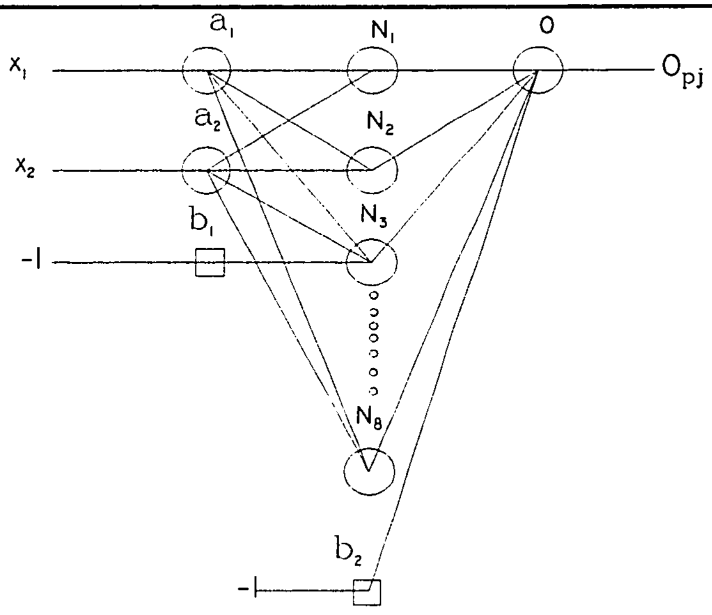


Figure 6.16 Neural Networks for Block Classification.

Table 6.4 Weights of input layer to hidden layer

Weight of node\node	X_1	X_2	b_1
N_1	-0.0020	-0.0025	-0.1863
N_2	0.0243	0.0168	2.6598
N_3	0.2907	- 0.2312	2.8331
N_4	0.0076	0.0028	0.8556
N_5	-0.0111	- 0.0072	-1.6263
N_6	0.0071	- 0.0052	0.9784
N_7	-0.0104	-0.0011	-2.4102
N_8	0.0582	-0.2610	2.7822

DL1: $g(X_1, X_2) = -0.0020\ X_1 - 0.0025X_2 + 0.1863$ (6.4)

DL2: $g(X_1, X_2) = 0.0243\ X_1 + 0.0168X_2 - 2.6598$ (6.5)

DL3: $g(X_1, X_2) = 0.2907\ X_1 - 0.2312X_2 - 2.8331$ (6.6)

DL4: $g(X_1, X_2) = 0.0076\ X_1 + 0.0028X_2 - 0.8556$ (6.7)

DL5: $g(X_1, X_2) = -0.0111\ X_1 - 0.0072X_2 + 1.6263$ (6.8)

DL6: $g(X_1, X_2) = 0.0071\ X_1 - 0.0052X_2 - 0.9784$ (6.9)

DL7: $g(X_1, X_2) = -0.0104\ X_1 - 0.0011X_2 + 2.4102$ (6.10)

DL8: $g(X_1, X_2) = 0.0582\ X_1 - 0.2610X_2 - 2.7822$ (6.11)

These decision lines as shown in Figure 6.17 divide all the gridded blocks in the block model into regions according to their target values being negative, positive or zero. Each region has gridded blocks with only negative, positive or zero economic block values.

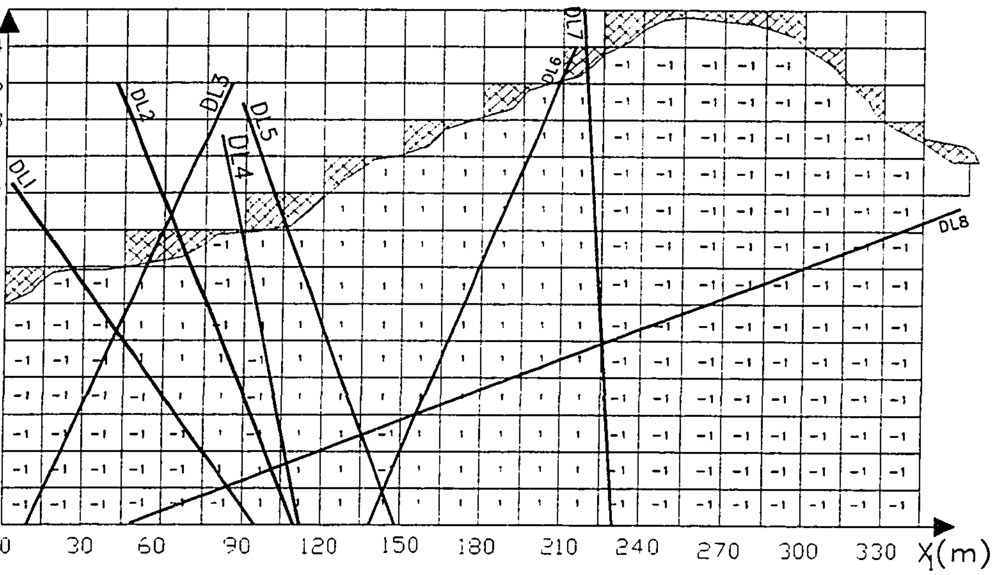


Figure 6.17 Decision Lines on the Blocks Target Values.

When the actual data of the block model was used as input in the 2-D CS/MFNN and 2-D Lerchs-Grossmann (LG) algorithms, each returned the same optimum limits and pit value of $9,053,840 shown in Table 6.5. Figure 6.18 and Figure 6.19 depict the imposed slope walls and optimum pit limits obtained from the 2-D CS/MFNN respectively. The pit limit as obtained from the 2-D Lerchs-Grossmann algorithm is illustrated in Figure 6.20.

Table 6.5 Optimum Pit Values for 2-D Experiment

Spacing	Actual (US$×$10^3$)	LG & CS/MFNN (US$×$10^3$)	Difference (US $×$10^3$)	% Error
45m	9053.84	9050.30	3.54	0.04
60m	9053.84	9047.90	5.94	0.07
75m	9053.84	9011.58	42.26	0.47
90m	9053.84	8963.63	90.21	0.99
105m	9053.84	7001.47	2052.37	22.67
135m	9053.84	4320.46	4733.38	52.28

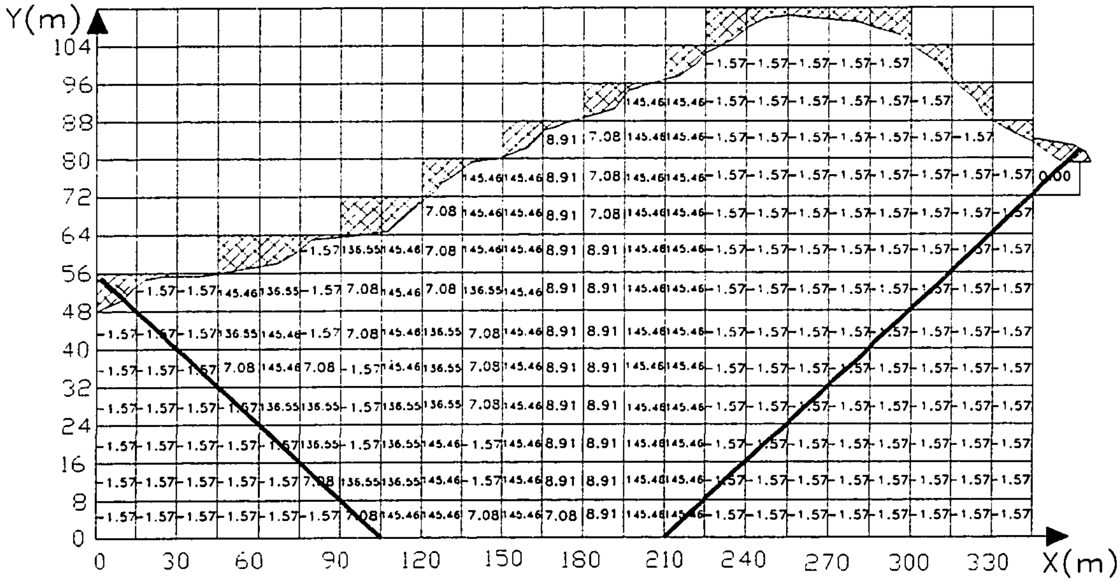


Figure 6.18 Imposed Slope Walls

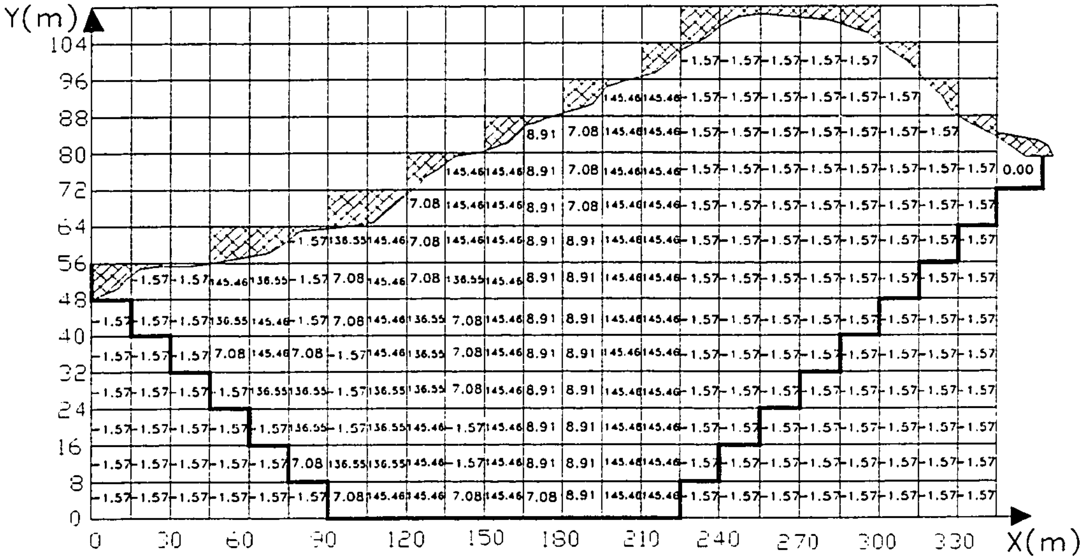


Figure 6.19 2-D Optimized Pit Outline Using CS/MFNN Algorithm.

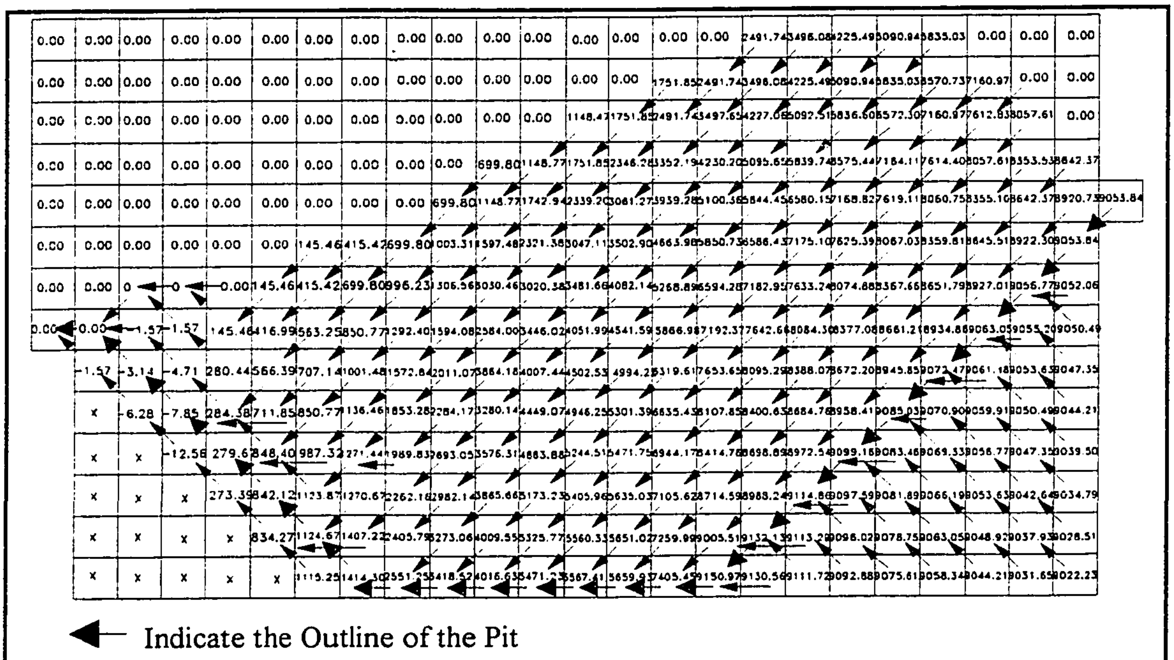


Figure 6.20 2-D Optimized Pit Outline Using Lerchs-Grossmann's Algorithm.

Experimenting with each of the six data sets of different drill-hole spacing as inputs in the algorithms, the drill-hole spacing of 45m, 60m, 75m and 90m yielded the same optimum limits but pit values of $9050.30, $9047.90, $9011.58, $8963.63, $7001.47, $4320.46 respectively from both 2-D CS/MFNN and 2-D Lerchs-Grossmann algorithms. The optimum pit values are tabulated in Table 6.5. The data sets of the drill-hole spacing of 105m and 135m yielded the same optimum limits but pit values of $7001.47 and $4320.46 respectively from both 2-D CS/MFNN and 2-D Lerchs-Grossmann algorithms. Thus, the pit values of drill-hole spacing from 45m to 90m give optimum pit limits equal to the actual pit limits and pit values of negligible errors compared to the actual optimum pit value. However, beyond 90m the optimum pit values are abysmally lower than the actual optimum pit value. The imposed slope walls shown in Figure 6.18 and the optimum pit layouts shown in Figure 6.19 give the optimum pit value and pit limits of the 2-D CS/MFNN algorithm for drill-hole spacing of 45m, 60m, 75m and 90m. The corresponding optimum pit limits for the 2-D Lerchs-Grossmann algorithm are illustrated in Figures 6.21 to 6.24.

Indicate the Outline of the Pit

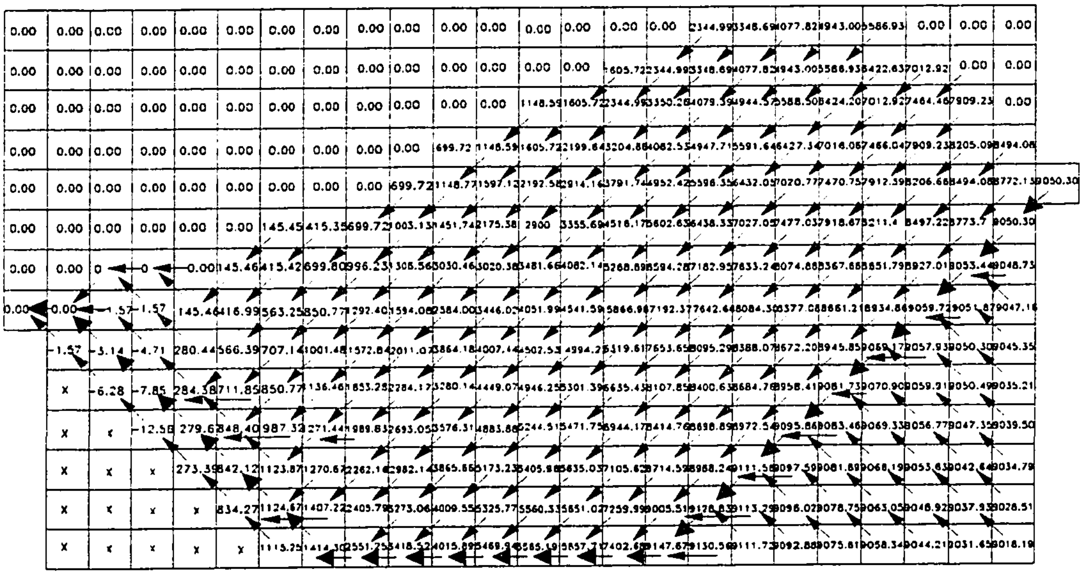
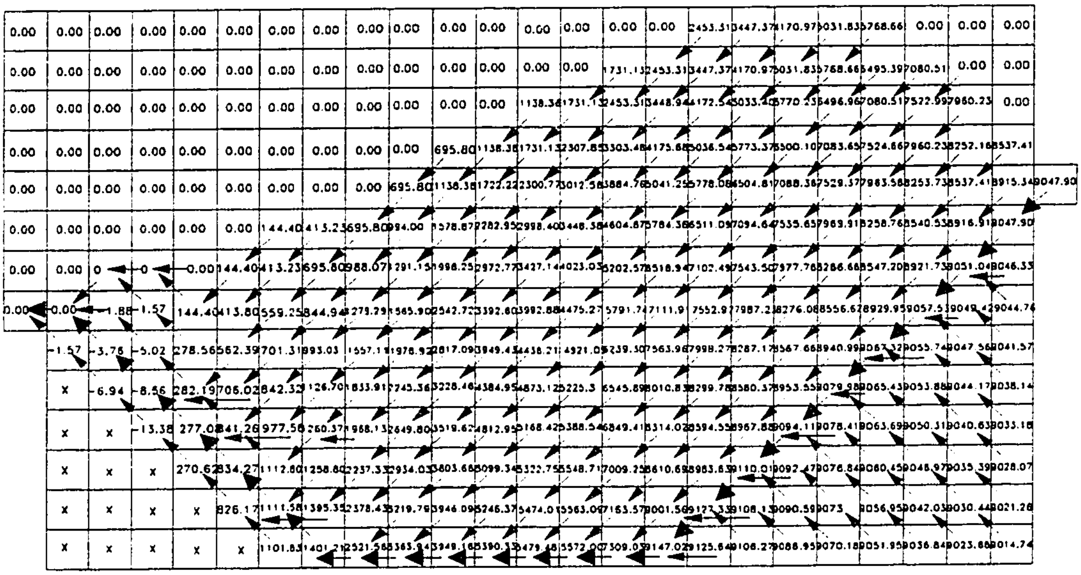


Figure 6.21 2-D Optimized Pit Outline Using Lerchs-Grossmann's Algorithm for Drill-hole Spacing of 45m



Indicate the Outline of the Pit

Figure 6.22 2-D Optimized Pit Outline Using Lerchs-Grossmann's Algorithm for Drill-hole Spacing of 60m

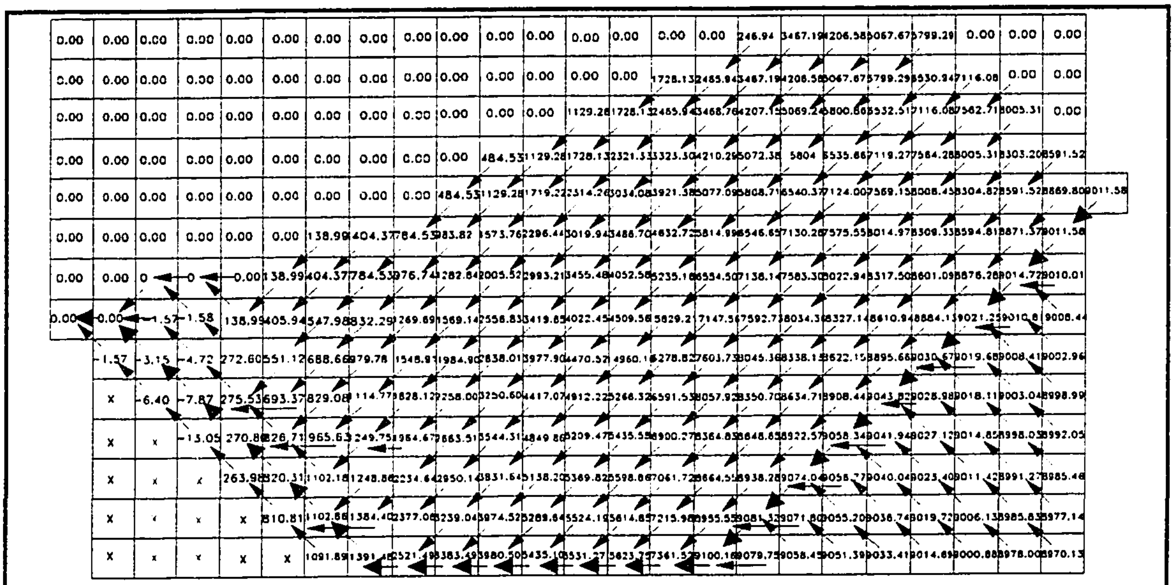
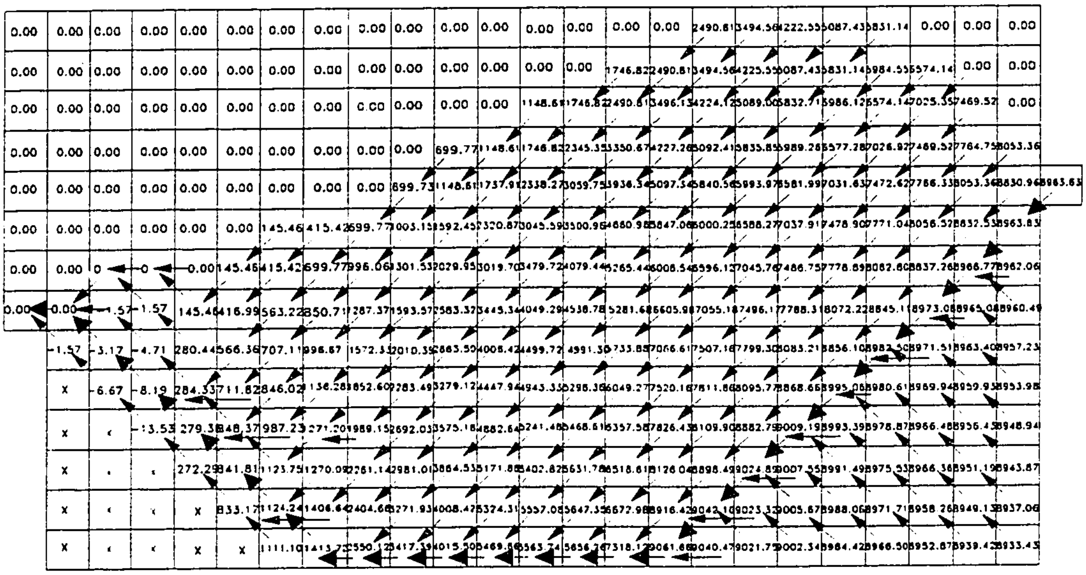


Figure 6.23 2-D Optimized Pit Outline Using Lerchs-Grossmann's Algorithm for Drill-hole Spacing of 75m



◀— Indicate the Outline of the Pit

Figure 6.24 2-D Optimized Pit Outline Using Lerchs-Grossmann's Algorithm for Drill-hole Spacing of 90m

The imposed slope walls which give the optimum pit value and pit limits of the 2-D CS/MFNN algorithm for drill-hole spacing of 105 m and 135 m are depicted in Figure 6.25. Figures 6.26 and 6.27 depict the corresponding optimum pit limit for the 2-D Lerchs-Grossmann algorithm. The variation of optimum pit value with drill-hole spacing of sample data for the 2-D block model is illustrated in Figure 6.28.

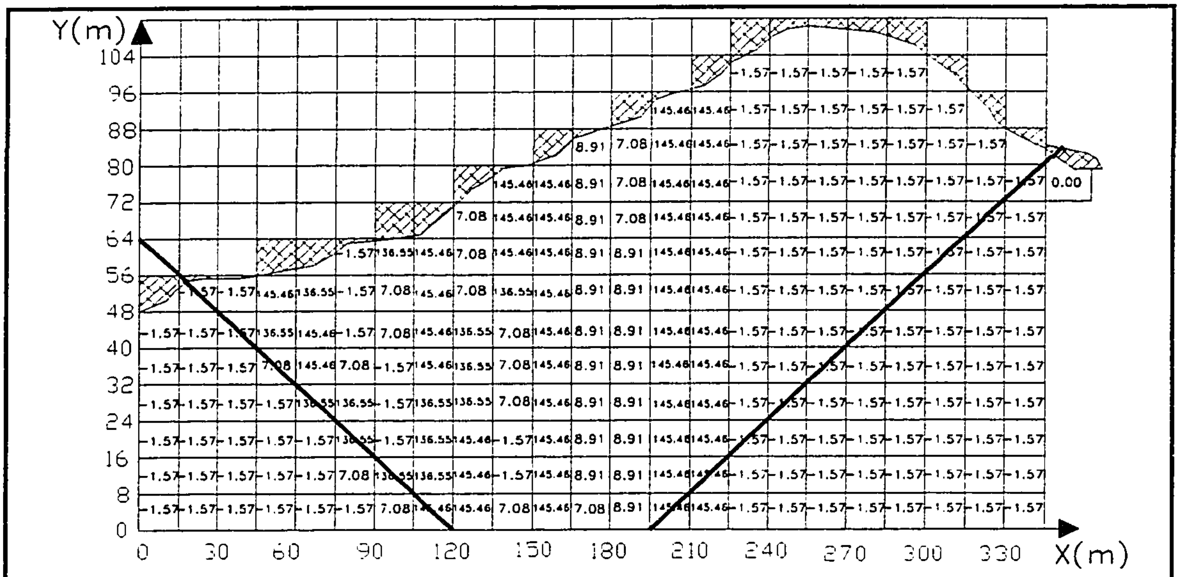


Figure 6.25 Imposed Slope Walls for Drill-hole Spacing of 105m and 135m.

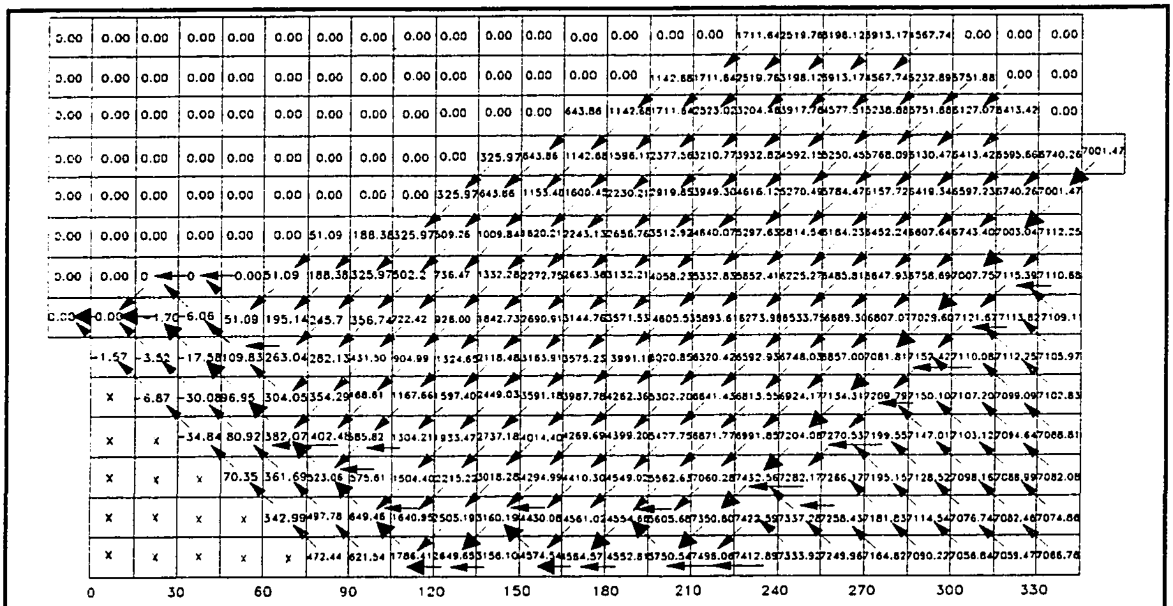
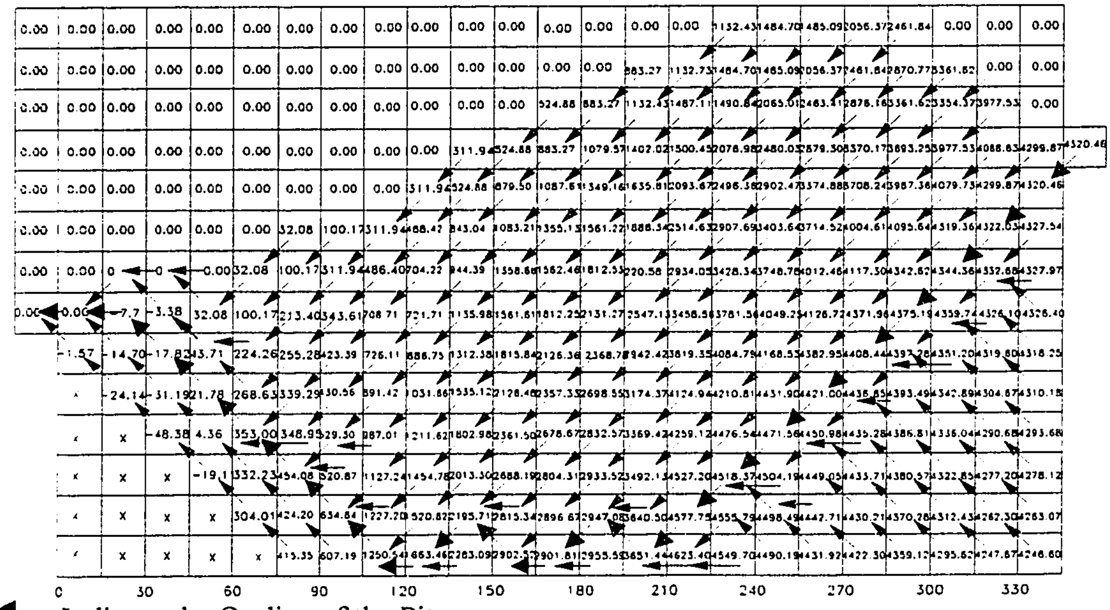


Figure 6.26 2-D Optimized Pit Outline Using Lerchs-Grossmann's Algorithm for Drill-hole Spacing of 105m



← Indicate the Outline of the Pit

Figure 6.27 2-D Optimized Pit Outline Using Lerchs-Grossmann's Algorithm for Drill-hole Spacing of 135m

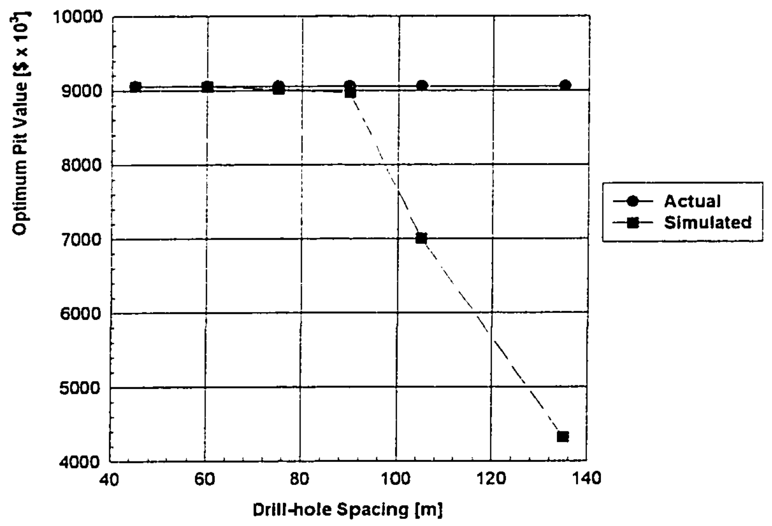


Figure 6.28 Variation of 2-D Optimum Pit Value with Drill-hole Spacing

3-D Model Results

In the 3-D model, using the actual data as inputs in the algorithms, the 3-D CS/MFNN and 3-D Lerchs-Grossmann algorithms each returned the same optimum limits and pit value of $27,161,520. The experimental results are as shown in Table 6.6.

Table 6.6 Optimum Pit Values for 3-D Experiment

Spacing	Actual (US$×10^3)	LG & CS/MFNN (US$×10^3)	Difference (US$×10^3)	% Error
45m	27161.52	27129.96	31.56	0.12
60m	27161.52	27091.29	70.23	0.26
75m	27161.52	26983.95	177.57	0.65
90m	27161.52	26886.2	275.32	1.01
105m	27161.52	20515.94	6645.58	24.47
135m	27161.52	12165.38	14996.14	55.21

Experimenting with each of the six data sets of different drill-hole spacing as inputs in the algorithms, the drill-hole spacing of 45m, 60m, 75m and 90m yielded the same optimum limits but pit values of $27129.96, $27091.29, $26983.95 and $26886.20 respectively from both 3-D CS/MFNN and 3-D Lerchs-Grossmann algorithms. In the case of drill-hole spacing of 105m and 135m , the optimum pit values were $20515.94 and $12165.38 respectively. Figure 6.29 illustrates the optimum pit limits for both 3-D CS/MFNN and 3-D Lerchs-Grossmann algorithms for the 45m, 60m, 75m and 90m drill-hole spacing. The optimum pit limits for drill-hole spacing of 105m and 135m for both 3-D CS/MFNN and 3-D Lerchs-Grossmann algorithms is shown in Figure 6.30 The variation of optimum pit value with drill-hole spacing of sample data for the 3-D block model is illustrated in Figure 6.31.

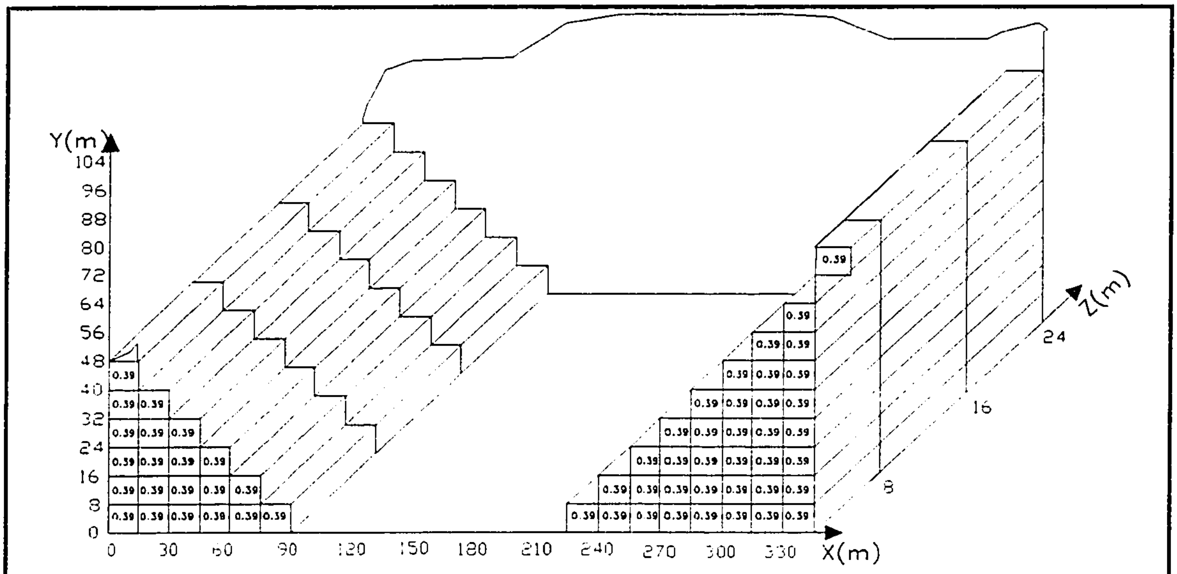


Figure 6.29 The optimum pit limits for both 3-D CS/MFNN and 3-D Lerchs-Grossmann algorithms for the 45m, 60m, 75m and 90m drill-hole spacing

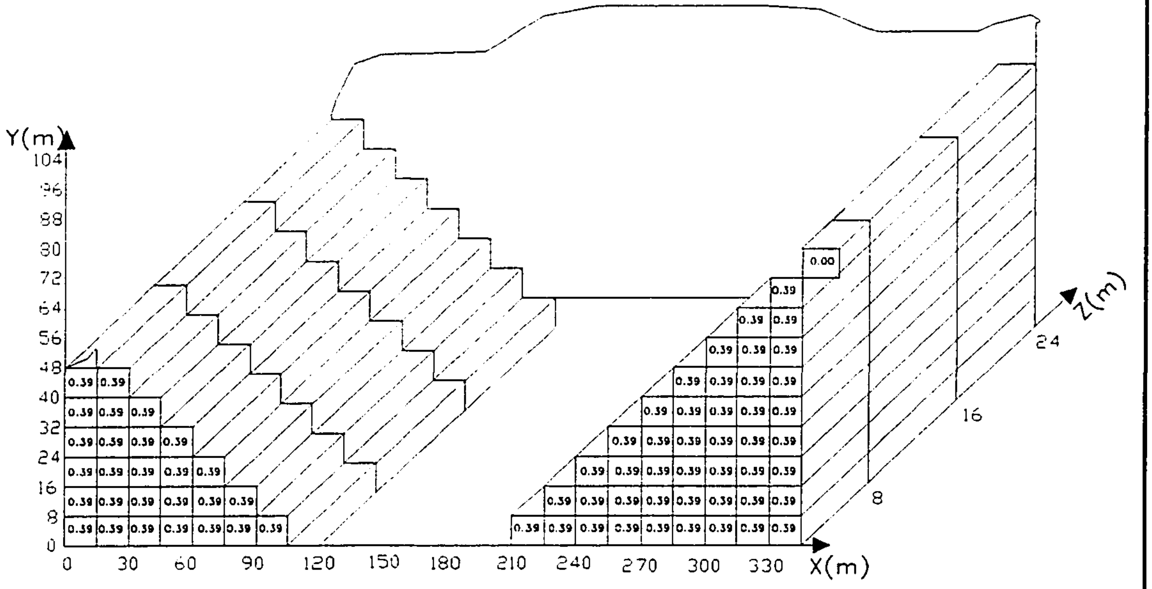


Figure 6.30 The optimum pit limits for both 3-D CS/MFNN and 3-D Lerchs-Grossmann algorithms for the 105m and 135m drill-hole spacing

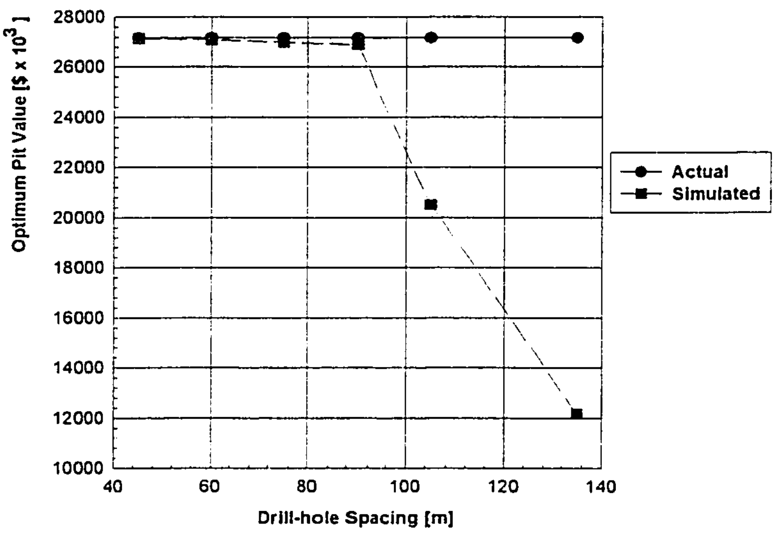


Figure 6.31 Variation of 3-D Optimum Pit Value with Drill-hole Spacing

To examine the behaviour of the pit limits when zero economic block values are present in the block model, zero target values (zero economic block values) are inserted in the pit at some few locations as shown in Figure 6.32. The MFNN model is used to delineate the blocks. Decision line equations of block model with zero target values are as written in equations (6.12) to (6.28).

DL1: $\mathbf{g(X_1, X_2) = 0.1457\ X_1 + 0.1266X_2 - 1.0633}$ **(6.12)**

DL2: $\mathbf{g(X_1, X_2) = -0.1414\ X_1 - 0.0909X_2 + 0.8909}$ **(6.13)**

DL3: $\mathbf{g(X_1, X_2) = -0.1412\ X_1 - 0.1429X_2 + 1.3840}$ **(6.14)**

DL4: $\mathbf{g(X_1, X_2) = 0.1548X_1 + 0.0417X_2 - 1.0060}$ **(6.15)**

DL5: $\mathbf{g(X_1, X_2) = -0.0869\ X_1 - 0.0385X_2 + 0.9554}$ **(6.16)**

DL6: $\mathbf{g(X_1, X_2) = 0.0952\ X_1 + 0.0286X_2 - 1.0762}$ **(6.17**

DL7: $\mathbf{g(X_1, X_2) = 0.1675\ X_1 - 0.0263X_2 - 0.9713}$ **(6.18)**

DL8: $\mathbf{g(X_1, X_2) = 0.6387\ X_1 - 0.3125X_2 - 1.9161}$ **(6.19)**

DL9: $\mathbf{g(X_1, X_2) = -0.0664\ X_1 - 0.0204X_2 + 0.9967}$ **(6.20)**

DL10: $\mathbf{g(X_1, X_2) = -0.1375\ X_1 - 0.0370X_2 + 2.1250}$ **(6.21)**

DL11: $\mathbf{g(X_1, X_2) = 0.0975\ X_1 - 0.0103X_2 - 1.1946}$ **(6.22)**

DL12: $\mathbf{g(X_1, X_2) =-0.0643\ X_1 - 0.0031X_2 + 0.9837}$ **(6.23)**

DL13: $\mathbf{g(X_1, X_2) = -0.1299\ X_1 + 0.0909X_2 + 0.9870}$ **(6.24)**

DL14: $\mathbf{g(X_1, X_2) = 0.2564\ X_1 - 0.3333X_2 - 0.8974}$ **(6.25)**

DL15: $\mathbf{g(X_1, X_2) = 0.0592\ X_1 - 0.2632X_2 - 0.9051}$ **(6.26)**

DL16: $\mathbf{g(X_1, X_2) = 0.2257\ X_1 - 0.6667X_2 - 1.8573}$ **(6.27)**

DL17: $\mathbf{g(X_1, X_2) =-0.0402\ X_1 - 0.7692X_2 + 2.3846}$ **(6.28)**

The optimum pit limits obtained using the Lerchs-Grossmann algorithm (Figure 6.33) are the same as in the previous cases. However, the optimum pit limits obtained using the CS/MFNN algorithm are different from the previous cases. As depicted in Figure 6.34 and Figure 6.35, the CS/MFNN algorithm leaves the zero blocks behind.

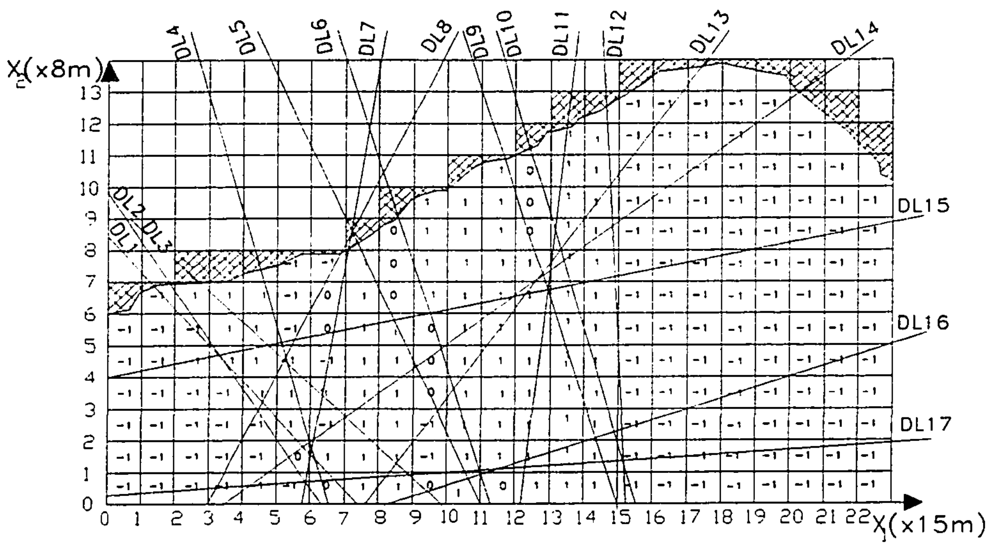


Figure 6.32 Decision Lines of Block Model with Zero Target Values.

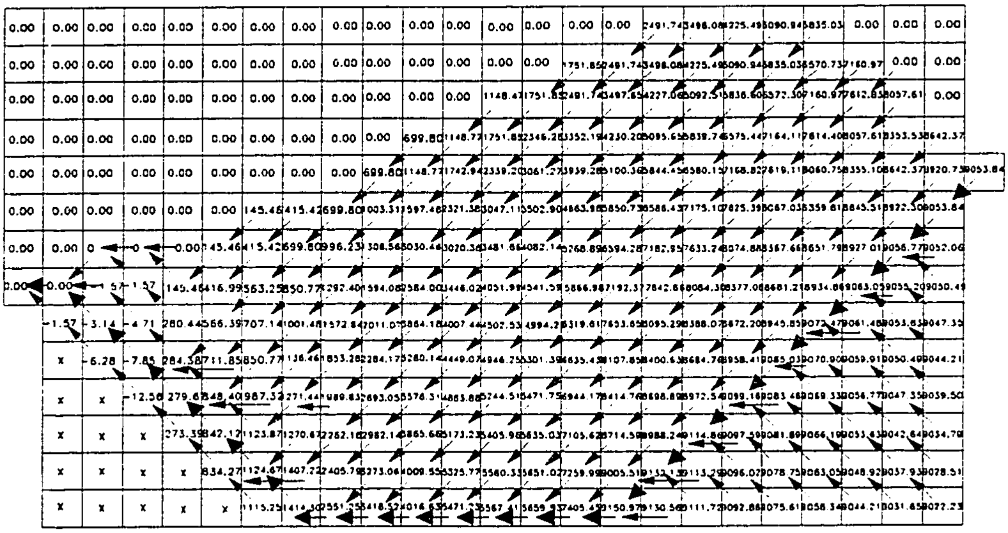
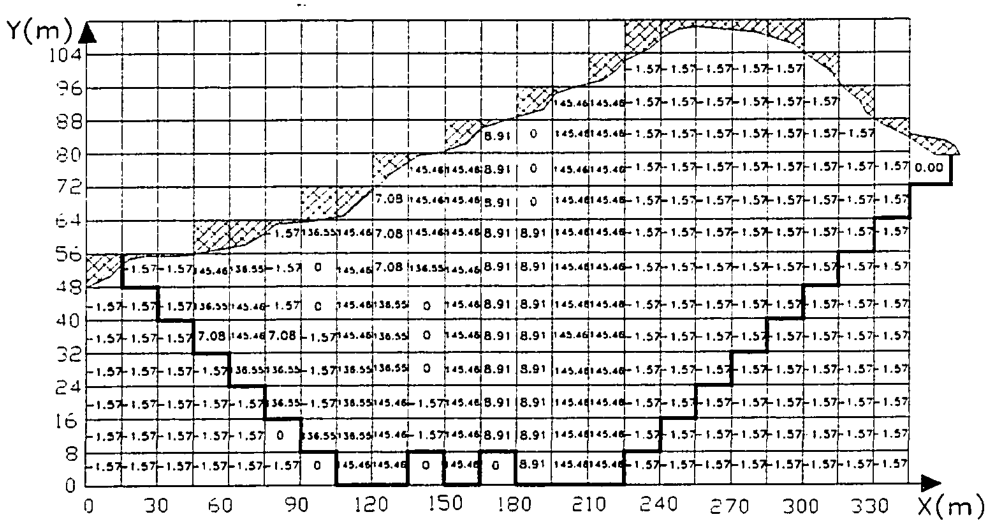


Figure 6.33 Optimized Pit Outline Using Lerchs-Grossmann algorithm



															-1.57	-1.57	-1.57	-1.57	-1.57				
													145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57			
											8.91	0	145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57		
									145.46	145.46	8.91	0	145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	0.00
								7.08	145.46	145.46	8.91	0	145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	
					-1.57	136.55	145.46	7.08	145.46	145.46	8.91	8.91	145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	
	-1.57	-1.57	145.46	136.55	-1.57	0	145.46	7.08	136.55	145.46	8.91	8.91	145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	
-1.57	-1.57	-1.57	136.55	145.46	-1.57	0	145.46	136.55	0	145.46	8.91	8.91	145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	
-1.57	-1.57	-1.57	7.08	145.46	7.08	-1.57	145.46	136.55	0	145.46	8.91	8.91	145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	
-1.57	-1.57	-1.57	-1.57	136.55	136.55	-1.57	136.55	136.55	0	145.46	8.91	8.91	145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	
-1.57	-1.57	-1.57	-1.57	-1.57	136.55	-1.57	136.55	145.46	-1.57	145.46	8.91	8.91	145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	
-1.57	-1.57	-1.57	-1.57	-1.57	0	136.55	136.55	145.46	-1.57	145.46	8.91	8.91	145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	
-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	0	145.46	145.46	0	145.46	0	8.91	145.46	145.46	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	-1.57	

Figure 6.34 Optimized Pit Outline Using CS/MFNN Algorithm for Zero E.B.Vs.

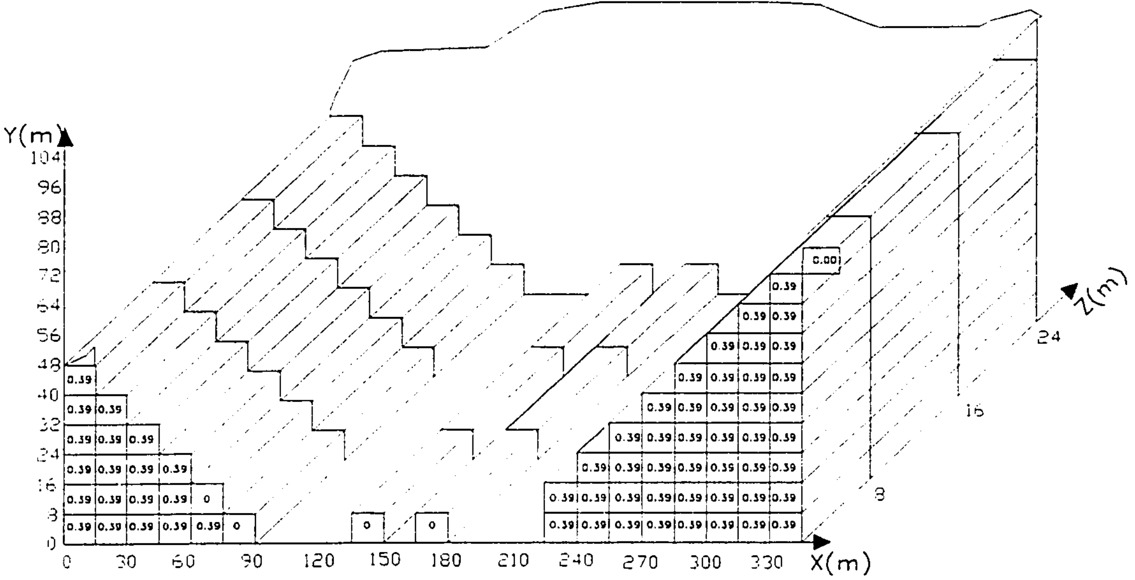


Figure 6.35 The optimum pit limits for 3-D CS/MFNN algorithm with zero E.B.Vs

6.5 Summary

The results from the MRM-MFNN model shows that data of a particular socio-politico-economic cycle can be used for price modelling. Understanding of the socio-politico-economic situation is of prime importance. The world annual gold production, annual gold consumption, average annual monthly low gold prices and average annual monthly high gold prices are critical determinants of mineral prices. With high uncertainties in metal prices in today's markets, investors in mineral projects will be better informed by the results of evaluation tools that treat the stochasticity of metal prices rigorously. The MRM-MFNN price model is set up to rigorously deal with metal price uncertainties and presents one of the most viable metal price forecasting methods in economic evaluation of mineral deposits.

From the results obtained from the CS model, it is able to predict the grades with less error if the drill-hole spacing is between 0.13 and 0.26 of the deposit dimensions. Based upon these results, it can be said that drill hole spacing of between 0.13 and 0.26 of the deposit dimensions is enough to predict block grades, optimum pit limit and value with less error. In the evaluation of the economic potential of a mineral deposit, the CS would be efficient in assigning grades to locations of unknown grades with less error. This would reduce the high cost of drilling so many drill-holes.

Comparing the Lerchs-Grossmann's and CS/MFNN algorithms, it can be said that both yield the same optimum pit value in the absence of grid blocks with zero economic block values. However, in the presence of grid blocks with zero economic block values. they may portray different pit outlines. Whereas the Lerchs-Grossmann's algorithm considers any gridded block irrespective of its economic block value in the design process, CS/MFNN algorithm does not consider gridded blocks with zero or negative economic block values unless they happen to lie above any of the gridded blocks in the positive regions delineated by the MFNN. CS/MFNN helps in selective mining of profitable blocks since it eliminates all the negative regions which are not economically viable for mining. The blocks with zero economic values are left behind. This will enable engineers to plan well to leave those blocks with zero EBVs and maintain proper production scheduling and planning. This will increase the profit margin of the company.

CHAPTER 7.0

CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER RESEARCH WORK

7.1 Summary

A new algorithm for open pit optimization using artificial neural networks on conditionally simulated blocks, CS/MFNN, has been developed and used to optimize open pit limits for 2-D and 3-D block models of an orebody. MRM-MFNN price model to forecast stochastic mineral prices to calculate the economic block values in open pit optimization has been developed.

This study achieved the objectives set out in Chapter 1 to develop, verify and validate (i) CS/MFNN algorithm for pit design and optimization; and (ii) mineral price model, MRM-MFNN, for economic block values calculation.

7.2 Results Achieved

The experimentation carried out using MRM-MFNN model to forecast mineral prices revealed that the mineral price model can predict average-annual gold price with negligible error. This is evidenced by 2.07 % error in the 1995 average-annual gold price prediction by the MRM-MFNN model. Also the mineral price model shows that a realistic forecasting of gold prices can be made with data, from an identified socio-politico-economic cycle, based on world annual gold production, annual gold consumption, average-annual monthly high gold price and average-annual monthly low gold price.

From the results obtained from the CS/MFNN model, it is able to estimate gridded-block grade with less error if the drill-hole spacing is between 0.13 and 0.26

of the deposit dimensions. With this same drill-hole spacing, the CS model is capable of estimating optimum pit value and limit with less error. CS/MFNN and Lerchs-Grossmann algorithms yield the same optimum pit value in the absence of grid blocks with zero economic block values.

7.3 Conclusions

From the analyses of the 2-D and 3-D CS/MFNN models, and the MRM-MFNN price model the following conclusions are drawn:

1. The 2-D and 3-D CS/MFNN and Lerchs-Grossmann algorithms return the same optimized pit value and limits.
2. CS/MFNN helps in selective mining of profitable blocks since it eliminates all the negative regions which are not economically viable for mining. The blocks with zero economic values are left behind. This will enable engineers to plan well to leave those blocks with zero E.B.Vs and maintain proper production scheduling and planning. This will increase the profit margin of the company.
3. The CS/MFNN algorithm can bring savings in drilling cost, due to geological and technological difficulties in obtaining many data to define a potential ore deposit.
4. The problem of overlapping zones and search patterns for optimum pit value in the floating cone algorithm is overcome by the CS/MFNN algorithm. The task of both forward and backward passes in the Lerchs-Grossmann's algorithm is obviated in CS/MFNN algorithm. CS/MFNN delineates the positive regions and incorporates the technical constraints to design the optimum pit.
5. CS/MFNN algorithm is applicable to any slope wall constraint in both 2-D and 3-D design. No smoothing is needed. Floating cone and Lerchs-Grossmann design slopes in multiples of blocks which need smoothing.
6. This algorithm combines a comprehensive orebody modelling and open pit optimization in random multivariable states.

7. The randomness associated with the distribution of the ore grades and reserves and metal prices are not taken into consideration by the existing algorithms. This randomness property can be incorporated in these algorithms by using the CS to calculate the block grades and/or tonnage and MRM-MFNN price model to calculate the economic block values. The omission of randomness property in any algorithm poses limitations on the pit limit definition, the economic potential of the mineral deposit, and the long-, intermediate-, and short-range mine plans. This might lead to poor investment decision and inaccurate mine design and optimization.
8. Comparing the Lerchs-Grossmann's and CS/MFNN algorithms, it can be said that both yield the same optimum pit value in the absence of grid blocks with zero economic block values. However, in the presence of grid blocks with zero economic block values they may portray different pit outlines. Whereas the Lerchs-Grossmann's algorithm considers any gridded block irrespective of its economic block value in the design process, CS/MFNN algorithm does not consider gridded blocks with zero or negative economic block values unless they happen to lie above any of the gridded blocks in the positive regions delineated by the MFNN.
9. Analysis of the results shows that the mineral price model predicts average-annual gold price with negligible error. A realistic forecasting of gold prices can be made with data from an identified socio-politico-economic cycle. World annual gold production, annual gold consumption, average-annual monthly high gold price and average-annual monthly low gold price are the most important factors which determine world average-annual gold prices. The main novelty of this methodology is the solution of the randomness property associated with mineral prices using multiple regression and artificial neural network to reduce the mineral price forecasting error. This is an important contribution to mineral venture evaluation, mine planning and design.

10. With high uncertainties in metal prices in today's markets, investors in mineral projects will be misled by the results of evaluation tools that do not treat the stochasticity of metal prices rigorously. The MRM-MFNN price model is set up to rigorously deal with metal price uncertainties in mineral project evaluation, and presents one of the most viable metal price forecasting methods in economic evaluation of mineral deposits.

Contributions of this study to evaluation and assessment of economic viability of open pit mining ventures and definition of open pit limits using open pit mining method can be summarized as follows:

1. CS/MFNN algorithm is used for the first time to evaluate and assess the economic potential of a mineral deposit using open pit mining method.
2. The MRM-MFNN price model can be used to forecast mineral prices in open pit mining venture evaluation.
3. For the first time, open pit optimization algorithm, CS/MFNN, that combines a comprehensive orebody modelling and open pit optimization in random multivariable states is developed.
4. Conditional simulation model can be used to calculate block grades in open pit mining.
5. Also the computing times precluding the time taken to create the revenue block model, the MFNN times in both CS and price models, for the same number of blocks the CS/MFNN algorithm is observed to be much quicker than Lerchs-Grossmann's algorithm.

7.4 Recommendations

Even though the models developed in this study have produced good and encouraging results, there are still some problems and limitations that need to be ironed out to improve the current results. The pit can be simulated using conditional simulation. The necessary information from mineral venture planners, evaluators and investors are not adequate for modelling the various processes involved in the evaluation and assessment of economic viability of open pit mining ventures. During grade estimation, only a database of few drill holes are used. The variogram used in the estimation is chosen from one of the existing theoretical models such as spherical, exponential and linear. Since the experimental variogram defined by the data from the field are not used, this may reduce the accuracy of the estimation.

Owing to the aforementioned problems, the most important recommendations for further research work are the following:

1. In this study, only gridded-blocks' grades are conditionally simulated but not the pit. However, the pit can also be conditionally simulated. In order to simulate the pit with conditional simulation, every simulation realization of the gridded-blocks' grades should be used in optimizing the pit, rather than the average values of the number of realizations as done in this study. Then a probability distribution should be constructed using the optimum pit values obtained for each realization. This would give the probability of optimum pit values for the evaluation of the economic potential of the ore deposit in question.
2. It is imperative to formulate mineral price model which can quantify the effects of various combinations of identifiable variables which are regularly encountered by mineral venture planners, evaluators and investors. This model can best be developed by thorough exchange of

ideas in groups including experienced mining engineers, evaluators, investment decision makers, stock brokers, financial institutions and commodity exchange markets. So good database should be developed to assemble all the necessary information from appropriate sources.

3. Further work is required to develop CS/MFNN user-friendly model for applications in the mining industry. For example, there is the need to interface the CS/MFNN and MRM-MFNN models into a single entity so that one input data file can serve both models.
4. *Since metal prices are highly volatile and stochastic, random field theory* and stochastic processes should be used in modelling.
5. Methods which will allow the data of semi-variogram to define themselves instead of imposing any of the existing models should be developed for the construction of semi-variogram in the CS model.

REFERENCES

Achireko P. K. and S. Frimpong, 1996, "Open Pit Optimization using Artificial Neural Networks on Conditionally Simulated Blocks"; Proc. 26th APCOM; Pittsburgh ; pp 144-137.

Beale R. and T. Jackson, 1990, Neural Computing; IOP Publishing Ltd, NewYork, NY.

Braticevic D.B., 1984, "Open Pit Optimization Method"; Proc. 18th APCOM; London; pp 133-137.

Bras, R. L. and I. Rodriguez-Iturbe, 1985, Randon Functions and Hydrology; Addison-Wesley, Reading, Massachussetts.

Carlson, T.R., Erickson, J.P., O'Brian, D.T.., et al., 1966, "Computer Techniques in Mine Planning";Mining Engineering; Vol. 18, No.5; pp.53-56.

Coleou, T. , 1989, "Technical Parameterization of reserves for Open Pit Design and Mine Planning"; 21st APCOM Proceedings; Littleton, Colorado; pp 485-494.

Danon, Y., 1996, WinNN 0.97, Neural Network Software Package ; Arad, Israel.

Denby, B. and D. Schofield, 1994,"Open Pit Design and Scheduling by use of Genetic Algorithms"; Transaction of the Institution of Mining & Metallurgy , Vol.102, Section A ; London; pp A21-A26.

Dowd, P.A. and Onur A.H., 1992, "Optimizing Open pit Design and Sequencing"; Optimal Open-pit Design"; 23rd APCOM Proceedings; Littleton, Colorado; pp 411-422.

Dowd, P.A. and Onur A.H., 1993, "Optimization of Open-pit Mine Design - Part 1: Optimal Open-pit Design"; Transaction of I MM, Vol.102, Section A; London; pp A95-A104.

Dowd, P.A. 1994, "The Optimal Design of Quarries"; Mineral Resource Evaluation II: Methods and Case Histories; Geological Society Special Publication No. 79, pp 141-155.

Fenton, G. A., 1995, "Theory of Random Fields"; Course Notes for AM6674; Technical University of Nova Scotia, Halifax Canada.

Francois-Bongarcon, D. M. and A. Marechal, 1976, " A New Method Open Pit Design: Parameterization of the Final Pit Contour"; 14th APCOM Proceedings; New York, New York; pp 573 -583.

Francois-Bongarcon, D. M. and D. Guibal, 1983a, " Parameterization of Optimal Design of an Open Pit "; AIME Transaction Vol.274; pp1801-1805.

Francois-Bongarcon, D. M. and D. Guibal, 1983b, " Algorithms for Parameterising Reserves Under Different Geometrical Constraints"; 17th APCOM Proceedings; New York, New York; pp 297-309.

Gauthier F.J. and Gray R.G., 1971, "Pit Design by Computer at Gaspe Copper Mines Ltd"; CIM Bulletin, Vol.64. pp 95-102.

Giannini L. and L. Cacceta, 1991, ; PITOPTIM: A New High Speed Network Flow Technique for Optimum Pit Design Facilitating Rapid Sensitivity Analysis; Proceeding of AIMM, Vol.296, No. 2; pp 57-62.

Gomm, J. B., Page, G. F. and D. Williams, 1993, "Introduction to Neural Networks"; Application of Neural Networks to Modelling and Control ; pp 1-7.

Gradshteyn, I. S. and I. M. Ryzhik, 1980, Table of Integrals, Series and Products; Academic, New York.

Lerchs, H and I. F. Grossmann, 1965, "Optimum Design of Open Pit Mines";C.I.M. Bulletin, Volume58; pp 47-54.

Grossmann, I. F. and H. Lerchs, 1965, "Optimum Design of Open-Pit Mines"; Transactions, C.I.M., Volume LXVIII; Montreal; pp17-24.

Grudnitski, G. and L. Osburn, 1993, "Forecasting S & P and Gold Futures Prices: An application of Neural Networks", Journal of Futures Markets.

Hartman, H. L., 1987, Introductory Mining Engineering; John Wiley and Sons Inc, New York; p 149.

Haykin, S., 1994, Neural Network: A Comprehensive Foundation, IEEE Press; pp. 179-181.

Hertz, J., Krogh, A. and R. G. Palmer, 1991, Introduction to the Theory of Neural Computation, Adison-Wesly Publishers, MA., U.S.A.

Hopfield, J. J., 1982, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities"; Proceeding of National Academy of Science U.S.A., 79, pp 2554-2558.

Hopfield, J.J., 1984, "Neurons with Graded Response have Collective Computational Properties Like Those of Two-State Neurons"; Proceeding of National Academy of Science U.S.A., pp 3088-3092.

Hustrulid, W. and M. Kuchta, 1995, Open Pit Mine Planning & Design, Volume I - Fundamentals; A.A.Balkema Publishers; Rotterdam, Netherlands; pp 413-442.

Huttagosol, P. and R.E. Cameron, 1992," Ultimate Pit Limit Design Using Transportation Algorithm"; 23rd APCOM Proceedings; Tucson, AZ; pp 443-460

Johnson, T.B. and D. G. Mickel, 1970, " Optimum Design of an Open Pit: An Application in Uranium"; 17th APCOM Proceedings, CIM Special Volume 12; Montreal, Canada; pp 331-338.

Johnson, T.B. and W. R. Sharp, 1971, "A Three-dimensional Dynamic Programming Method for Optimal Open Pit Design"; Rep. Invest. U.S. Bureau of Mines, No. 7553; p25.

Johnson, T.B., 1973, "A Comparative Study of Methods for Determining Ultimate Open Pit Mining Limits"; 11th APCOM Proceedings; Tucson, Arizona: University of Arizona; pp 129-138.

Johnson, T.B. and Barnes R. J., 1988, "Application of Maximal Flow Algorithm To Ultimate Pit Design" ; Engineering Design: Better Results Through Operations Research Methods; North Holland Publication Company; pp 518-531.

Journel, A. and C. H. Huijbregts, 1978, Mining Geostatistics, Academic Press, London.

Kennedy B. A.(Editor), 1990; Surface Mining, 2nd Edition, SME, Littleton; CO;pp 460-485.

Kim,Y.C., 1978, "Ultimate Pit Limit Design Methodologies Using Computer Models- The State of the Art"; Mining Engineering ; pp1454-1459.

Kitco Minerals & Metals Incorporated, Montreal, Canada.

Knudsen, H. P., Kim Y. C., and E. Mueller, 1975, "A Comparative Study of the Geostatistical ore Reserve Estimation Method Over the Conventional Methods";13th Application of Computers in the Mineral Industries Symposium; Techn,U. Clausthal, Fed. Rep. Germany.

Koborov, S., 1974, "Method for Determining Optimal Open Pit Limits; Rapport Report EP 14R-4, Ecole Polytechnique de Montreal.

Koenigsberg E., 1982, "The Optimum Contours of an Open Pit Mine: An Application of Dynamic Programming"; Proc. 17th APCOM; New York; pp 274- 287.

Kohonen, T., 1988, Self-Organization and Associative Memory; 2nd Edition; Springer-Verlag, Berlin and Heidelberg.

Kuchta, M. E., 1993, VarioC: Software for Variogram Analysis in 2-D and 3-D; Tekniska Hgskolani Lulea, S-951 87 Luleå, Sweden.

Labys, W. C., 1977, “Minerals Commodity Modelling- The State of the Art”; Deyoung J.H.(ed) Mineral Policies in Transition ; The Council of Economics of AIME (11); pp 80-106.

Lemieux M., 1968, “A Computerized Three Dimensional Optimum Open Pit Design System”; CIM Annual Meeting, Montreal, Canada.

Lemieux M., 1979, “Moving Cone Optimizing Algorithm”; Computer Methods for the 80s in the Mineral Industry; AIME, New York; pp 329-345.

Mantoglou, A., and J.L. Wilson, 1981, Simulation of Random Fields with the Turning Bands Method ; Report No. 264, Ralph M. Parsons Laboratory, Department of Civil Engineering , MIT, Cambridge.

Mantoglou, A., and J.L. Wilson, 1982, “The Turning Bands Method for Simulation of Random Fields Using Line Generation by a Spectral Method,” Water Resources Research, Volume 18, No. 5, pp 1379-1394.

Matern, B., 1960, “Spatial Variation”; Medd. Statens Skogsforskniginst. Swed., 49(5).

Matheron,G., 1973, “The Intrinsic Random Functions and their Applications,”Advanced Application of Probability., 5, pp 439-468.

Marino J. M. and J. P. Slama, 1973, “Ore Resrve Evaluation and Open Pit Planning”; Proc. 10th APCOM; SAIMMM, Johannesburg; South Africa; pp 139-144.

Meyer M., 1969, “Applying Linear Programming to the Design of Ultimate Pit Limits”; Managemant Science, Volume 16, No. 2; pp B121-35.

Nilson, D.,1982, “Open-Pit or Underground Mining”, SME Underground Mining Methods Handbook ,SME of AIME; pp 70-87.

Philips, D. A., 1972, "Optimum Design af an Open Pit Design"; Proc. 10th APCOM; SAIMMM, Johannesburg; South Africa; pp 145-157.

Pana, M. and R.K.Davey, 1965,"The Simulation Approach to Pit Design; Proc. 5th APCOM; Arizona; pp zz1-zz24.

Pana, M. and R.K.Davey, 1973," Pit Planning and Design; Mining Handbook; SME of AIME; pp 17-10 to 17-19.

Pannatier, Y., 1996, VARIOWIN: Software for Spatial Data Analysis in 2D; Springer-Verlag, New York, NY.

Pardo-Iguzquiza, E., Chica-Olmo M. and J. Delgado-Garcia, 1992, “SICON1D: A Fortran-77 Program for Conditional Simulation in One Dimension”; Computers and Geosciences, Volume 18, No. 6, pp 665-688.

Pindyck, R. S. and D. L. Rubinfeld, 1991, Econometric Models and Economic Forecasts; McGraw-Hill Inc., New York; pp 73-100.

Rice, S. O., 1954, "Mathematical Analysis of Random Noise"; Selected Papers on Noise and Stochastic Processes , Edited by N. Wax; Dova, Mineola, New York.

Robinson R.H. and Prenn N. B., 1973, "An Optimum Pit Design Model"; Proc. 10th APCOM; Johannesburg, RSA; pp 155-163.

Rumelhart, D. E., Hinton, G. E., and R. J. Williams, 1986, "Learning Internal Representations by Error Propagation"; Parallel Distributed Processing, Volume 1, MIT Press, Cambridge, MA; pp 318-362.

Sen, S., 1996, " Investigation of the Effects of Quantization Errors on Performance of Neural Networks"; Ph.D. Thesis; Technical University of Nova Scotia Halifax, Nova Scotia, Canada

Shenggui Z. and Starfield A. M., 1985, "Dynamic Programming with Colour Graphics Smoothing for Open Pit Design on a Personal Computer" ; International Journal of Mining Engineering, Vol.3; pp 27-34.

Shinozuka, M. and C. M. Jan, 1972, "Digital Simulation of Random Processes and its Applications,'' Journal of Sound Vibration, Volume 25, No. 1, pp 111-128.

Shoenberg, I. J., 1938, "Metric Spaces and Completely Monotone Functions,''Annals of Mathematics, Volume 39; pp 811-841.

Strauss, S. D., 1992, "Pricing and Trading in Metals and Minerals"; SME Mining Engineering Handbook; 2nd Edition Volume 1; pp 81-88.

Stuart N. J., (1992), "Pit Optimisation Using Solid Modelling and the Lerchs-Grossmann Algorithm"; International Journal of Surface Mining Reclamation Volume 6, Number 1; pp 19-30.

TSP statistical software Version 4.2B; 1994 TSP International; Palo Alto, California.

Thompson, A.F.B., Ababou, R. and L. W. Gelhar 1989, "Implementation of the Three-Dimensional Turning Bands Random Field Generator"; Water Resources Research; Volume 25, No. 10; pp 2227-2243.

Tolwinski, B. and R. Underwood, 1992, "An Algorithm to Estimate the Optimal Evolution Pit Mine"; 23rd APCOM Proceedings; Tucson, AZ; pp 399-409.

Wang, Q. and H. Sevim, 1992 ," Enhanced Production Planning in Open Pit Mining Through Intelligent Dynamic Search";23rd APCOM Proceedings; Tucson, AZ; pp 461-471.

Wasserman, P. D., 1992, Neural Computing: Theory and Practice; Van Nostrand Reinhold, New York; pp 53-54.

Wilke F. L., Muellar K. and E. A. Wright, 1984, "Ultimate Pit Limit and Production Scheduling Optimazationtions; 18th APCOM Proceedings; London: IMM; pp 29-38.

Williams, C. E., 1970, "Computerized Mine Planning"; AIME, Tucson, Arizona; pp 97-104.

Wright, E.A., 1987, "The Use of Dynamic Programming for Open Pit Mine Design: Some Pratical Implications; Mining Science and Technology, January; pp 97-104.

Wright, E.A., 1990, Open Pit Mine Design Models; Trans Tech Publications, Federal Republic of Germany.

Yegulalp, T.M. and J.A. Arias, 1992, "A Fast Algorithm to Solve the Ultimate Pit Limit Problem; 23rd APCOM Proceedings; Tucson, AZ ; pp391-397.

Yixian, Z. and Y.C. Kim, 1992, "A New Optimum Pit Limit Design Algorithm"; 23rd APCOM Proceedings; Tucson, AZ; pp 423-434

Zurada, J.M, 1992, Introduction to Artificial Neural Systems; West Publishing Company, St. Paul, MN.,U.S.A.

APPENDIX A

Mathematical Modelling of Lerchs-Grossmann's (LG) Algorithms

Mathematical Modelling of Lerchs-Grossmann's (LG) Algorithms

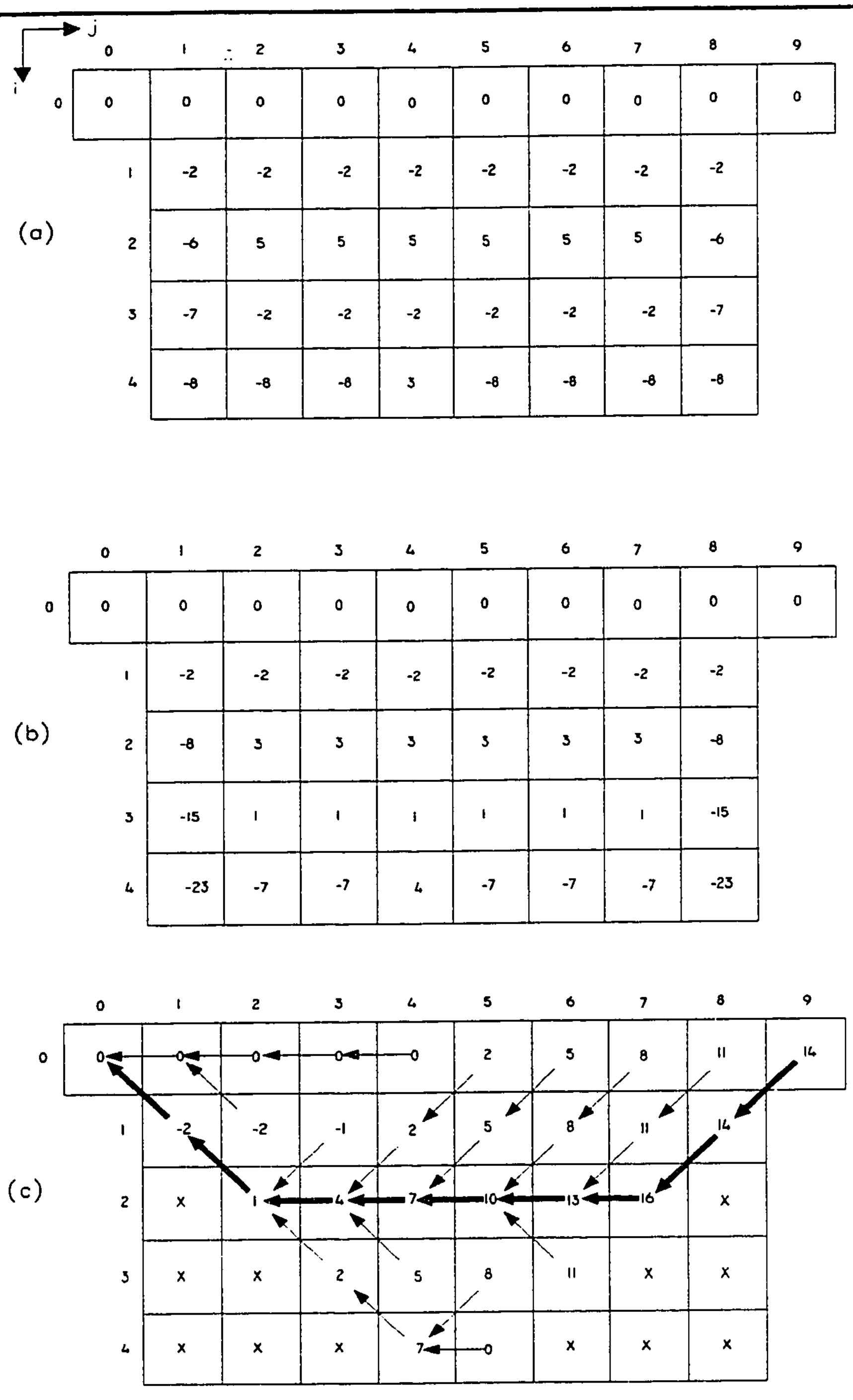
Lerchs and Grossmann (1965) have used the graph theory and dynamic programming to formulate an open pit model to define the optimum pit limit. The objective of Lerchs-Grossmann's algorithm is to design the contour of a pit (final pit limit) so as to maximize the difference between the total mine value of ore extracted and the total extraction cost of ore and waste - i.e., the total profit.

The Lerchs-Grossmann's algorithm is based on the following assumptions: (i) the type of material, its mine value and its extraction cost are given for each point; (ii) restrictions on the geometry of the pit are specified (surface boundaries and maximum allowable wall slopes); (iii) the objective is to maximize total profit- total mine value of material extracted minus total extraction cost. Lerchs and Grossmann used dynamic programming technique to design the 2-D pit (or a single vertical section of a mine), and a graph algorithm for the general 3-D pit.

Two-Dimensional Pit Modelling

The modelling of 2-D dynamic programming algorithm of Lerchs and Grossmann is clearly illustrated in Figure A.1. In the 2-D pit design, the units $\boldsymbol{u_i}$ and $\boldsymbol{u_j}$ of a rectangular grid system are selected such that

$$\frac{\boldsymbol{u_i}}{\boldsymbol{u_j}} = \tan\alpha \qquad \textbf{(A.1)}$$



(a)

i \ j	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1		-2	-2	-2	-2	-2	-2	-2	-2	
2		-6	5	5	5	5	5	5	-6	
3		-7	-2	-2	-2	-2	-2	-2	-7	
4		-8	-8	-8	3	-8	-8	-8	-8	

(b)

	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1		-2	-2	-2	-2	-2	-2	-2	-2	
2		-8	3	3	3	3	3	3	-8	
3		-15	1	1	1	1	1	1	-15	
4		-23	-7	-7	4	-7	-7	-7	-23	

(c)

	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	2	5	8	11	14
1		-2	-2	-1	2	5	8	11	14	
2		X	1	4	7	10	13	16	X	
3		X	X	2	5	8	11	X	X	
4		X	X	X	7	0	X	X	X	

Bold arrows for pit limit

Figure A.1. The Steps for Lerchs-Grossmann's 2D Algorithm for Pit Design.

For each unit rectangle (i, j), the block economic value m_{ij} (*EBV*) is determined as defined in equation (3.1). Figure A.1 (a) shows a vertical section of a block model with the block economic values, m_{ij}, written on each block. In addition, a row of 'air blocks' is superimposed on the section as the zero row. This row serves as the starting row and the collection row for the results. A new tableau (Figure A.1 (b)) is constructed with the quantities

$$M_{ij} = \sum_{k=1}^{i} m_{kj} \qquad \textbf{(A.2)}$$

M_{ij} represents the profit realized in extracting a single column with element (i, j) at its base. In the final tableau (Figure A.1 (c)), a temporary value, P_{ij} is computed, column by column, starting with column 1 on the top left corner.

$$P_{ij} = M_{ij} + \max_{k} \; (P_{i+k,\,j-1}), \text{ with } k = -1,\ 0,\ 1 \qquad \textbf{(A.3)}$$

$$P_{max} = \max_{k} \; P_{1k}, \text{ with } k = -1,\ 0,\ 1 \qquad \textbf{(A.4)}$$

The maximum is indicated by an arrow going from (i, j) to $(i + k, j - 1)$.

P_{ij} is the maximum possible contribution of columns 1 to j to any feasible pit that contains the element (i, j) on its contour. It follows that if the element (i, j) is part of the optimum contour, then this contour, to the left of element (i, j), can be traced by following the arrows starting from element (i, j). Any feasible contour must contain at least one element of the first row. If the maximum value of P in the first row is positive, then the optimum contour is obtained by following the arrows from and to the left of this element. If all elements of the first row are negative, then there exists no contour with positive profit. In the illustrative example, the pit value is *+14* in Figure A1 (c).

Three-Dimensional Pit Modelling

When the optimum contour of all the vertical sections are assembled, it invariably turns out that they do not fit together. This is because the wall slopes in a vertical section and at right angles with the sections that were optimized exceed the permissible angle α. The walls and the bottom of the pit are then "smoothed out". The resulting contour may be far from optimum, thus the solution yields a series of optima. Hence, the dynamic approach becomes impractical in three dimensions. Instead Lerchs and Grossmann applied a graph algorithm. The algorithm converts the three-dimensional grid of blocks in the orebody model into directed graph (Lerchs and Grossmann, 1965; Dowd and Onur, 1993).

The graph theory model for the 3-D pit design is derived as follows:

The entire pit is divided into a set of volume elements V_i. This division can be arbitrary but may also be obtained by taking for V_i the unit volumes defined by a three-dimensional grid. Associate to each volume element V_i a mass

$$m_i = v_i - c_i \tag{A.5}$$

where v_i and c_i are the mine value and the extraction cost of element V_i. Each element is represented by a vertex x_i of a graph. An arc (x_i, x_j) is drawn if V_j is adjacent to V_i, that is, V_i and V_j have at least one point in common, and if the mining volume V_i is not permissible unless volume V_j is also mined. Thus, a three-dimensional graph $G = (X, A)$ with a set of vertices X and a set of arcs A is obtained. Any feasible contour of the pit is represented by a closure of G, that is, a set of vertices Y such that if a vertex x_i belongs to Y and if the arc (x_i, x_j) exists in A then the vertex x_j must also belong to Y. If a mass m_i is associated to each vertex x_i, and if M_y is the total mass of a set of vertices Y , then the problem of optimum pit design comes to finding in a graph G a closure Y with maximum mass or, shortly, a maximum closure of G.

Definition of Some Important Terms and Concepts

A directed graph $\mathbf{G} = (\mathbf{X}, \mathbf{A})$ is defined by a set of elements $\mathbf{X}$ called vertices of $\mathbf{G}$, together with a set $\mathbf{A}$ of ordered pairs of elements $\mathbf{a}_i = (\mathbf{x}, \mathbf{y})$, called the arcs of $\mathbf{G}$. The graph $\mathbf{G}$ also defines a function Γ mapping $\mathbf{X}$ into $\mathbf{X}$ and such that $(\mathbf{x}, \mathbf{y}) \in \leftrightarrow \mathbf{y} \in \Gamma\mathbf{x}$
A path is a sequence of arcs $(\mathbf{a}_i, \mathbf{a}_i, \ldots \mathbf{a}_n)$ such that the terminal vertex of each arc corresponds to the initial vertex of the succeeding arc. A circuit is a path in which the initial vertex coincides with the terminal vertex. An edge, $\mathbf{e}_i = [\mathbf{x}, \mathbf{y}]$ of $\mathbf{G}$, is a set of two elements such that $(\mathbf{x}, \mathbf{y}) \in \mathbf{A}$ or $(\mathbf{y}, \mathbf{x}) \in \mathbf{A}$. A chain is a sequence $[\mathbf{e}_1, \mathbf{e}_2, \ldots \mathbf{e}_n]$ in which each edge has one vertex in common with the succeeding edge. A cycle is a chain in which the initial and final vertices coincide. A subgraph $\mathbf{G(Y)}$ of $\mathbf{G}$ is a graph $(\mathbf{Y}, \mathbf{A}_y)$ defined by a set of vertices of $\mathbf{Y} \subset \mathbf{X}$ and containing all the arcs that connect vertices of $\mathbf{Y}$ in $\mathbf{G}$. A partial graph $\mathbf{G(B)}$ of $\mathbf{G}$ is a graph $(\mathbf{X}, \mathbf{B})$ defined by a set of arcs $\mathbf{B} \subset \mathbf{A}$ and containing all the vertices of $\mathbf{G}$. A directed graph $\mathbf{G} = (\mathbf{X}, \mathbf{A})$ is a set of vertices $\mathbf{Y} \subset \mathbf{X}$ such that $\mathbf{x} \in \mathbf{Y} \rightarrow \Gamma\ \mathbf{x} \in \mathbf{Y}$. If $\mathbf{Y}$ is a closure of $\mathbf{G}$, then $\mathbf{G(Y)}$ is a closed subgraph of $\mathbf{G}$. By definition, the null set, $\mathbf{Y} = \phi$, is also a closure of $\mathbf{G}$. A tree is a connected and directed graph $\mathbf{T} = (\mathbf{X}, \mathbf{C})$ containing no cycles. A rooted tree is a tree with one distinguished vertex, the root. The arc obtained by suppressing an arc $\mathbf{a}_i$ in a rooted tree $\mathbf{T}$ has two components. The component $\mathbf{T}_i = (\mathbf{X}_i, \mathbf{A}_i)$ which does not contain the root of the tree is called a branch of $\mathbf{T}$. The root of the branch is the vertex of the branch that is adjacent to the arc $\mathbf{a}_i$. A branch is a tree itself, and branches of a branch are called twigs.

<u>Problem formulation</u>: Given a directed graph $\mathbf{G} = (\mathbf{X}, \mathbf{A})$ and for each vertex $\mathbf{x}_i$, find a closure $\mathbf{Y}$ of $\mathbf{G}$ with maximum mass. In other words, find a set of elements vertices $\mathbf{Y} \subset \mathbf{X}$ such that $\mathbf{x} \in \mathbf{Y} \rightarrow \Gamma\ \mathbf{x} \in \mathbf{Y}$ and $\mathbf{M}_\mathbf{Y} = \sum_{X_i \in Y} m_i$ is maximum. A closure with maximum mass is also called a maximum closure.

Algorithm: The graph **G** is first augmented with a dummy node $\mathbf{x_o}$ and dummy arcs ($\mathbf{x_o}$, $\mathbf{x_i}$). The algorithm starts with the construction of a tree $\mathbf{T^o}$ in **G**. $\mathbf{T^o}$ is then transformed into successive trees $\mathbf{T^1}$, $\mathbf{T^2}$, ... $\mathbf{T^n}$ following given rules, until no further transformation is possible. The maximum closure is then given by the vertices of a set of well identified branches of the final tree.

Each edge $\mathbf{e_k}$ (arc $\mathbf{a_k}$) of a tree **T** defines a branch, noted as $\mathbf{T_k} = (\mathbf{X_k}, \mathbf{A_k})$. The edge $\mathbf{e_k}$ (arc $\mathbf{a_k}$) is said to support the branch $\mathbf{T_k}$. The mass $\mathbf{M_k}$ of a branch $\mathbf{T_k}$ is the sum of all vertices of $\mathbf{T_k}$. This mass is associated with the edge $\mathbf{e_k}$ (arc $\mathbf{a_k}$) and the edge $\mathbf{e_k}$ (arc $\mathbf{a_k}$) supports a mass $\mathbf{M_k}$. In a tree T with root $\mathbf{x_o}$, an edge $\mathbf{e_k}$ (branch $\mathbf{T_k}$) is characterized by the orientation of the arc $\mathbf{a_k}$ with respect to $\mathbf{x_o}$; $\mathbf{e_k}$ is called a p-edge (plus-edge) if the arc $\mathbf{a_k}$ points toward the branch $\mathbf{T_k}$, that is, if the terminal vertex of $\mathbf{a_k}$ is part of the branch $\mathbf{T_k}$. $\mathbf{T_k}$ is then called a p-branch. If arc $\mathbf{a_k}$ points away from branch $\mathbf{T_k}$, then $\mathbf{e_k}$ is called an m-edge (minus-edge) and $\mathbf{T_k}$ an m-branch. Similarly, all twigs of a branch are divided into two classes: p-twigs and m-twigs. A p-edge (branch) is strong if it supports a mass that is strictly positive; an m-edge (branch) is strong if it supports a mass that is null or negative. Edges (branches) that are not strong are said to be weak. A vertex $\mathbf{x_i}$ is said to be strong if there exists at least one strong edge on the chain of **T** joining $\mathbf{x_i}$ to the root $\mathbf{x_o}$. Vertices that are not strong are said to be weak. Finally, a tree is normalized if the root $\mathbf{x_o}$ is common to all strong edges. Any tree **T** of a graph **G** can be normalized by replacing the arc ($\mathbf{x_k}$, $\mathbf{x_l}$) of a strong p-edge with a dummy arc ($\mathbf{x_o}$, $\mathbf{x_l}$), the arc ($\mathbf{x_q}$, $\mathbf{x_r}$) of a strong m-edge with a dummy arc ($\mathbf{x_o}$, $\mathbf{x_q}$) and repeating the process until all strong edges have $\mathbf{x_o}$ as one of their extremities. The tree in Figure A.3 is obtained by normalizing the tree in Figure A2. The graph **G** considered in the sequel will be an augmented graph obtained by adding to the original graph a dummy vertex $\mathbf{x_o}$ with negative mass and dummy arcs ($\mathbf{x_o}$, $\mathbf{x_i}$), joining $\mathbf{x_o}$ to every vertex $\mathbf{x_i}$. The introduction of dummy arcs ($\mathbf{x_o}$, $\mathbf{x_i}$), does not affect the problem because $\mathbf{x_o}$ cannot be part of any maximum closure of **G**. The vertex $\mathbf{x_o}$ will be the root of all trees considered.

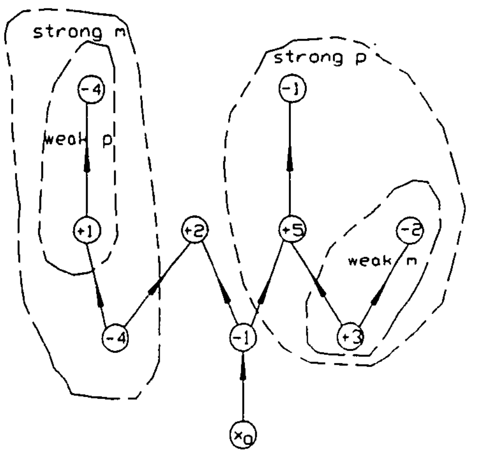


Figure A.2

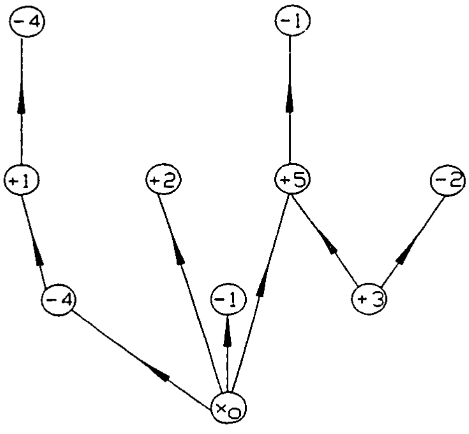


Figure A.3

Properties and Theorems of Normalized Tree :

Property 1: If a vertex belongs to the maximum closure **Z** of a normalized tree **T**, then all the vertices $\mathbf{X}_k$ of the branch $\mathbf{T}_k$ also belong to **Z**.

Property 2: The maximum closure of a normalized tree **T** is the set **Z** of its strong vertices.

Property 3: If, in the tree $\mathbf{T}^s$, $\mathbf{e}_d$ is an *m*-edge on the chain $[\mathbf{x}_m, . ., \mathbf{x}_o]$ then the mass M_d^s is strictly positive and larger than any mass supported by a p-edge that precedes $\mathbf{e}_d$ on the chain $[\mathbf{x}_m, . ., \mathbf{x}_o]$.

Theorem I : If, in a directed graph **G**, a normalized tree **T** can be constructed such that the set **Y** of strong vertices of **T** is a closure of **G**, then **Y** is a maximum closure of **G**.

Theorem II : A maximum closure of **G** is obtained in a finite number of steps.

Steps of the Algorithm: Construct a normalized tree $\mathbf{T}^0$ in **G** and enter the iterative process. Iteration **t+1** transforms a normalized tree $\mathbf{T}^t$ into a new normalized tree $\mathbf{T}^{t+1}$. Each tree $\mathbf{T}^t = (\mathbf{X}, \mathbf{A}^t)$ is characterized by its set of arcs $\mathbf{A}^t$ and its set of strong vertices $\mathbf{Y}^t$

. The process terminates when **Y** is a closure of **G**. Iteration **t+1** contains the following steps:

1.- If there exists an arc ($\mathbf{x_k}$, $\mathbf{x_l}$) in **G** such that $\mathbf{Y_k} \in \mathbf{Y^t}$, and $\mathbf{x_l} \in \mathbf{X\text{-}Y^t}$, then go to step 2. Otherwise go to step 4.

2.- Determine $\mathbf{x_m}$, the root of the strong branch containing $\mathbf{x_k}$. Construct the tree $\mathbf{T^s}$ by replacing the arc ($\mathbf{x_o}$, $\mathbf{x_m}$) of $\mathbf{T^t}$ with the arc ($\mathbf{x_k}$, $\mathbf{x_l}$). Go to step 3.

3.- Normalize $\mathbf{T^s}$. This yields $\mathbf{T^{t+1}}$. Go to step 1.

4.- Terminate. $\mathbf{Y^t}$ is a maximum closure of **G**. (See Figure A.4).

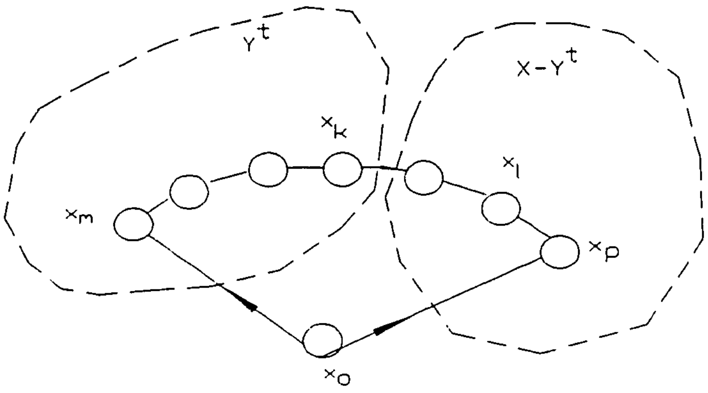


Figure A.4

Construction of $\mathbf{T^o}$: $\mathbf{T^o}$ can be obtained by constructing an arbitrary tree in **G** and then normalizing this tree as outlined earlier. A much simpler procedure, however, is to construct the graph (**X**, $\mathbf{A_D}$) where $\mathbf{A_D}$ is the set of all dummy arcs ($\mathbf{x_o}$, $\mathbf{x_l}$). This graph is a normalized tree.

<u>Transformations taking place in steps 2 and 3 of the algorithm:-</u>

(a) Construction of $\mathbf{T^s}$:

The tree $\mathbf{T^s}$ is obtained from $\mathbf{T^t}$ by replacing the arc ($\mathbf{x_o}$, $\mathbf{x_m}$) with the arc ($\mathbf{x_k}$, $\mathbf{x_l}$). The arc ($\mathbf{x_o}$, $\mathbf{x_m}$) supports in $\mathbf{T^t}$ a branch $\mathbf{T_m}^t$ with a mass $M_m^t > \mathbf{0}$. Let [$\mathbf{x_m}$, . ., $\mathbf{x_k}$, $\mathbf{n_l}$,. . ., $\mathbf{x_p}$, $\mathbf{x_o}$] be the chain of T^s linking x_m to x_o as shown in Figure A.4. Except for this chain, the status of an edge of T^s and the mass supported by the edge are unchanged by this transformation. On the chain $[x_m, \ldots, x_o]$ of T^s the transformation of masses are:

For an edge e_i on the chain $[x_m, \ldots, x_k]$

$$M_i^s = M_m^t - M_i^t \tag{A.6}$$

For an edge $[x_k, \ldots, x_l]$

$$M_k^s = M_m^t \tag{A.7}$$

For an edge e_j on the chain $[x_l, \ldots, x_p, x_o]$

$$M_j^s = M_m^t - M_j^t \tag{A.8}$$

In addition, all the edges e_i on the chain $[x_m, \ldots, x_k]$ have changed their status: a p-edge in T^t becomes an m-edge in T^s and vice versa. On the chains $[x_m, \ldots, x_k]$ and $[x_l, \ldots, x_p]$ in T^t, all p-edges support zero or negative masses and all m-edges support strictly positive masses as T^t is normalized. Hence, the following distribution of masses in T^s are obtained:

	m-edge	p-edge	
edge e_i on chain $[x_m, \ldots, x_k]$	$M_i^s \geq M_m^t$	$M_i^s < M_m^t$	**(A.9)**
edge $[x_k, x_l]$	$M_k^s = M_m^t$		**(A.10)**
edge e_j on $[x_l, \ldots, x_p, x_o]$	$M_j^s > M_m^t$	$M_j^s < M_m^t$	**(A.11)**

(b) Normalization of T^s: As T^t was normalized, all strong edges must be on the chain $[x_m, \ldots, x_o]$. The strong edges are removed one by one starting from the first strong edge encountered on the chain $[x_m, \ldots, x_o]$. This edge, say $e_a = [x_a, x_b]$, must be a p-edge according to property 3. $\mathbf{e_a}$ is replaced with a strong dummy edge $(\mathbf{x_o}, \mathbf{x_a})$. Thus, a p-twig is removed from the branch $\mathbf{T_p}^s$ and its mass from all the edges of the chain $[\mathbf{x_b}, \ldots, \mathbf{x_o}]$ must be subtracted. Property 3 remains valid on the chain $[\mathbf{x_b}, \ldots, \mathbf{x_o}]$. The next strong p-edge on the chain $[\mathbf{x_b}, \ldots, \mathbf{x_o}]$ is now searched for and the process is repeated until the last strong p-edge has been removed from the chain. The transformations (a) and (b) can be carried out simultaneously.

Illustrative Example of the Lerchs-Grossmann 3-D Algorithm

An optimum open pit is to be designed from the block model in Figure A.5 using Lerchs-Grossmann's graph 3-D algorithm.

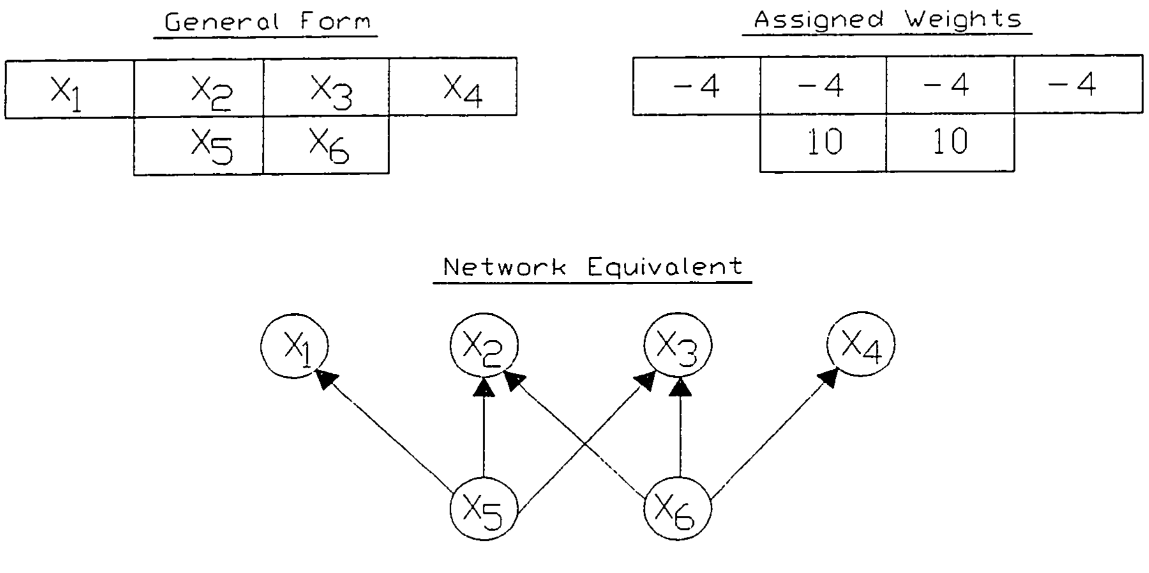


Figure A.5 Application of the Lerchs-Grossmann's 3-D Algorithm for Pit Design.

Step 1. Begin by adding a root node x_o and connecting arcs between the root and each of the other nodes. This is the formation of the initial tree $\mathbf{T}^0$ and all the arcs are plus as shown in Figure A.6.

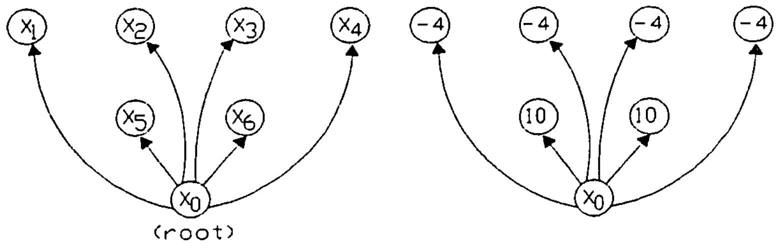


Figure A.6 Initial Tree with all Root Connection.

Step 2. The set (graph) of directed arcs is now split into two groups. Those connected to the root by strong-plus arcs are included in group $\mathbf{Y^0}$. The others are in group $\mathbf{X\text{-}Y^0}$. In this case nodes x_5 and x_6 are in group $\mathbf{Y^0}$. Their sum is 10 + 10 = 20.

Step 3. The possible connection between the two groups is examined, and following the sequencing constraints there are 6 directed arcs which can be drawn:

For node x_5 : (x_5, x_1), (x_5, x_2), (x_5, x_3)

For node x_6 : (x_6, x_2), (x_6, x_3), (x_6, x_4)

One of these is selected, say (x_5, x_1), and the directed arc (x_0, x_5) is removed and the directed arc (x_5, x_1) drawn. This is done in Figure A.7.

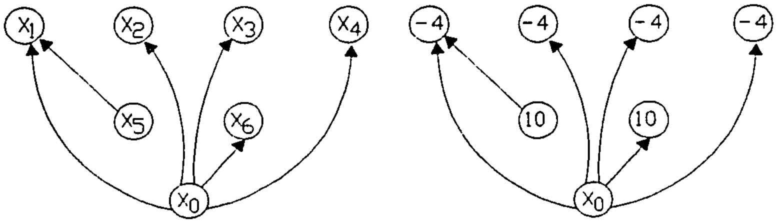


Figure A.7. Selection of Directed Arc (x_5, x_1).

Step 4. The normalizing process is now followed. Each arc is labelled with respect to 'plus' or 'minus' and 'strong' or 'weak'. The arc(connection) between x_o and x_1 is still strong-plus. Hence the members of the group are x_1, x_5 and x_6 shown in Figure A.8. The value of **Y** (closure) is -4 + 10 +10 = 16.

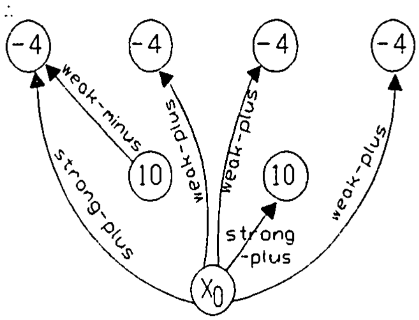


Figure A.8 Labelling of the Resulting Arcs in Step 4.

Step 5. Now return to step 3 to seek additional connections between the **Y** and **X-Y** (**X** without **Y**) groups. There are 5 feasible arcs: (x_5, x_2), (x_5, x_3) (x_6, x_2), (x_6, x_3), (x_6, x_4) The arc (x_5, x_2) will be added to the tree and arc (x_0, x_2) dropped. This is depicted in Figure A.9.

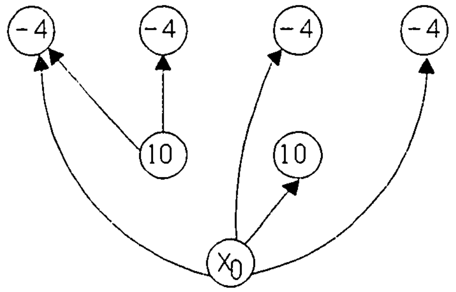


Figure A.9 Modification of Figure A.8 Based on Step 5.

Step 6. The new tree is now normalized as shown in Figure A.10. The nodes included in **Y** are x_1, x_2 , x_5, and x_6. The closure **Y** is - 4 + 10 +10 - 4 = 12.

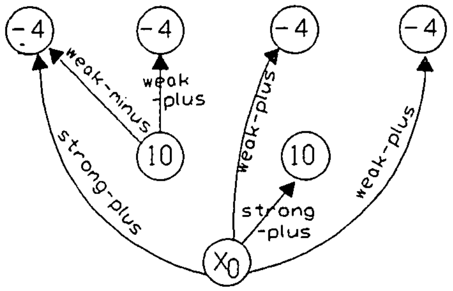


Figure A.10 Labelling of the Resulting Arcs in Step 6.

Step 7. Return to Step 5 and choose one of the 3 possible connections remaining: (x_5, x_3) (x_6, x_3), (x_6, x_4). Choosing (x_6, x_3) and dropping arc (x_0, x_3), the resulting normalized tree is as shown in Figure A.11. The arc (x_0, x_6) remains strong-plus and hence the nodes included within **Y** are x_1, x_2, x_3, x_5 and x_6. The overall closure is - 4 - 4 - 4 +10 + 10 = 8.

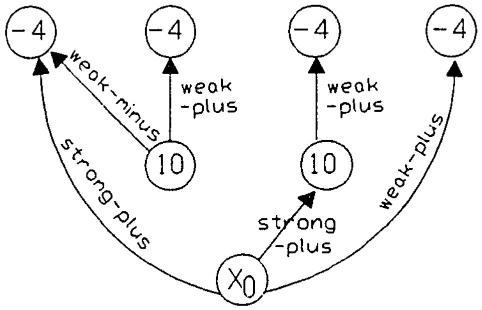


Figure A.11. Modification of Figure A.9 Based on Step 7.

Step 8. Return to Step 5 and take the one possible connections remaining: arc (x_6, x_4). Arc (x_0, x_4) is dropped and arc (x_6, x_4) added. The tree is now normalized as before with the result depicted in Figure A.12. All of the nodes are now attached directly to the root by chains having one strong edge.

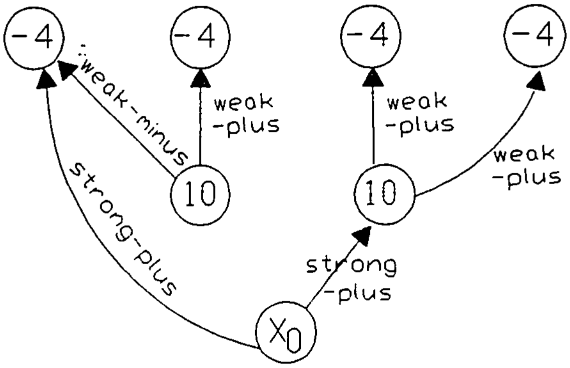


Figure A.12. Modification of Figure A.9 Based on Step 8.

Step 9. The maximum closure now is the cumulative sum of the nodes involved. In this case it is +4 and the included nodes are x_1, x_2, x_3, x_4, x_5 and x_6.

Adapted from Hustrulid W. and M. Kuchta (1995)

APPENDIX B

Computer Programs

1. PROGRAM BLUE.F
2. PROGRAM KALPHA.F
3. PROGRAM MBETA.F
4. PROGRAM LG23D.F
5. PROGRAM PMFNN.F

PROGRAM BLUE.F

```
C************************************************************
C PROGRAM TO SOLVE A LINEAR SYSTEM OF USING GAUSSIAN
ELIMINATION.
C VARIABLES USED ARE:
C       LIMIT : PARAMETER GIVING MAXIMUM DIMENSION OF MATRIX
C       LIMAUG: PARAMETER (LIMIT +1) FOR MAXIMUM # COLUMNS IN AUG
C       N     : NUMBER OF EQUATIONS AND UNKNOWNS
C      AUG    : AUGMENTED MATRIX FOR THE LINEAR SYSTEM
C       ETA   : SOLUTION VECTOR/BLUE COEFFICIENTS
C PP,PK,KQ,I,J,K : INDICES
C MALPHA,MBETA: THE AVERAGE GRADES AT POINTS ALPHA & BETA
RESPECTIVELY
C       KG    : KNOWN GRADE
C       kg.dt,kg.ut, cb.ut and ca.ut ARE AS EXPLAINED IN CHAPTER 4
C GRADE,G     : UNKNOWN GRADE BEING CALCULATED FOR
C       MULT  : MULTIPLIER USED TO ELIMINATE AN UNKNOWN
C       PIVOT : USED TO FIND NONZERO DIAGONAL ENTRY
C*************************************************************

      INTEGER LIMIT, LIMAUG,PP,PK,KQ
      PARAMETER (LIMIT = 100, LIMAUG =LIMIT +1,PK=30)
      DOUBLE PRECISION AUG(LIMIT, LIMAUG),
ETA(LIMIT),MULT,KG(1000),
   +  G,GRADE, MALPHA,MBETA
      INTEGER I,J,K,PIVOT
C READ COEFFICIENT MATRIX AND CONSTANT VECTOR
C

    OPEN(UNIT=1,FILE='kg.dt',STATUS='OLD')
    OPEN(UNIT=4,FILE='kg.ut',STATUS='OLD')
    OPEN(UNIT=2,FILE='cb.ut',STATUS='OLD')
    OPEN(UNIT=3,FILE='ca.ut',STATUS='OLD')

      N = 30

      READ(3,*)((AUG(I,J),J=1,N),I=1,N)
      READ(1,*)(KG(PP),PP=1,N)

      DO 400 KQ = 1,28

      READ(2,*)(AUG(I,N+1),I=1,N)

C GAUSSIAN ELIMINATION
```

```
      DO 70 I=1,N

C LOCATE NONZERO DIAGONAL ENTRY

      IF (AUG(I,I).EQ.0) THEN
      PIVOT = 0
      J=I+1
30    IF ((PIVOT.EQ.0) .AND. (J.LE.N)) THEN

      IF (AUG(J,I).NE.0) PIVOT=J
      J=J+1
      GO TO 30
      END IF
      IF (PIVOT.EQ.0) THEN
      STOP 'MATRIX IS SINGULAR'
      ELSE
C INTERCHANGE ROWS I AND PIVOT
      DO 40 J =1, N+1
      TEMP = AUG(I,J)
      AUG(I,J) = AUG(PIVOT,J)
      AUG(PIVOT,J) = TEMP
40    CONTINUE
      END IF
      END IF

C ELIMINATE ITH UNKNOWN FROM EQUATIONS I+1,..., N

      DO 60 J=I+1,N
      MULT = -AUG(J,I)/AUG(I,I)
      DO 50 K=I,N+1
       AUG(J,K)=AUG(J,K) + MULT*AUG(I,K)

50    CONTINUE
60    CONTINUE
70    CONTINUE

C FIND THE SOLUTIONS
C
      ETA(N)=AUG(N,N+1)/AUG(N,N)
      DO 90 J = N-1, 1, -1
      ETA(J)=AUG(J,N+1)
      DO 80 K = J+1,N
      ETA(J)=ETA(J)-AUG(J,K)*ETA(K)
80    CONTINUE
      ETA(J)=ETA(J)/AUG(J,J)
```

```
90    CONTINUE
C     WRITE(4,*)
C     WRITE(4,*),'SOLUTION VECTOR IS'
C     DO 110 I=1,N
C            WRITE(4,100)I,ETA(I)
C100  FORMAT(1X,'ETA(',I2,')=',F10.5)

C CALCULATION UNKNOWN BLOCK GRADE

    G=0
      DO 500 PP=1,N
    IF((PP.LE.3).OR.(PP.GE.13.AND.PP.LE.17).OR.
  + (PP.GE.18.AND.PP.LE.20).OR.(PP.GE.35.AND.PP.LE.37).OR.
  + (PP.GE.30.AND.PP.LE.34)
  +.OR.(PP.GE.47.AND.PP.LE.51).OR.
  + (PP.GE.64.AND.PP.LE.68).OR.(PP.GE.81.AND.PP.LE.85).OR.
  + (PP.GE.98.AND.PP.LE.102).OR.(PP.GE.115.AND.PP.LE.119).OR.
  + (PP.GE.129.AND.PP.LE.133).OR.(PP.GE.140.AND.PP.LE.144).OR.
  + (PP.GE.151.AND.PP.LE.155).OR.(PP.GE.160.AND.PP.LE.164).OR.
  + (PP.GE.168.AND.PP.LE.171).OR.(PP.GE.172.AND.PP.LE.175))
  +     MALPHA = 0.39

   IF((PP.GE.69.AND.PP.LE.71).OR.(PP.GE.86.AND.PP.LE.88))
  + MALPHA=3.195
   IF((PP.GE.52.AND.PP.LE.54).OR.(PP.GE.103.AND.PP.LE.105))
  + MALPHA=3.025


   IF((PP.GE.4.AND.PP.LE.6).OR.(PP.GE.21.AND.PP.LE.23).OR.
  +  (PP.GE.38.AND.PP.LE.40)) MALPHA=3.195
   IF(PP.GE.55.AND.PP.LE.57) MALPHA=5.66

   IF((PP.GE.72.AND.PP.LE.74).OR.(PP.GE.89.AND.PP.LE.91))
  + MALPHA=5.83
   IF((PP.GE.106.AND.PP.LE.108).OR.(PP.GE.120.AND.PP.LE.122))
  + MALPHA=3.19


   IF((PP.GE.7.AND.PP.LE.9).OR.
  +  (PP.GE.24.AND.PP.LE.26).OR.(PP.GE.41.AND.PP.LE.43))
  + MALPHA=3.395
   IF((PP.GE.58.AND.PP.LE.60).OR.
  +  (PP.GE.75.AND.PP.LE.77).OR.(PP.GE.92.AND.PP.LE.94))
  + MALPHA=3.225
   IF((PP.GE.109.AND.PP.LE.111).OR.
     + (PP.GE.123.AND.PP.LE.125)) MALPHA=0.755
      IF((PP.GE.134.AND.PP.LE.136).OR.(PP.EQ.156).OR.
     + (PP.GE.145.AND.PP.LE.147)) MALPHA=0.72


      IF((PP.GE.10.AND.PP.LE.12).OR.(PP.GE.27.AND.PP.LE.29).OR.
     + (PP.GE.44.AND.PP.LE.46).OR.(PP.GE.61.AND.PP.LE.63).OR.
     + (PP.GE.78.AND.PP.LE.80).OR.(PP.GE.95.AND.PP.LE.97).OR.
     + (PP.GE.112.AND.PP.LE.114).OR.(PP.GE.126.AND.PP.LE.128))
     +  MALPHA = 0.59
      IF((PP.GE.137.AND.PP.LE.139).OR.(PP.GE.148.AND.PP.LE.150)
     +.OR.(PP.GE.157.AND.PP.LE.159).OR.(PP.GE.165.AND.PP.LE.167))
     + MALPHA = 0.555


      G = G +(ETA(PP)*(KG(PP)-MALPHA))

 500    CONTINUE
        GRADE = G + MBETA
        WRITE(4,*) KQ,GRADE
400     CONTINUE
        CLOSE(1)
       CLOSE(2)
       CLOSE(3)
       CLOSE(4)
        END
```

PROGRAM KALPHA.F

```
C**********************************************************************
C
C THIS PROGRAM CALCULATES THE COVARIANCES OF THE SPHERICAL
C MODEL
C BETWEEN THE KNOWN SAMPLE POINTS FOR THE MATRIX  Kα
C FOR THE BLUE EQUATION PROGRAM.
C Z(I) : THE GRADE AT THE POINT X(I),Y(I)
C ca.dt: THE INPUT FILE OF KNOWN POINTS COORDINATES & GRADES
C ca.ut: THE OUTPUT FILE OF THE CALCULATED COVARIANCES FOR
C Kα  IN BLUE
C**********************************************************************
C
C
C
        DIMENSION X(1000),Y(1000),Z(1000),COV
        DOUBLE PRECISION COV

C      READ IN SAMPLES
      OPEN(UNIT=1,FILE='ca.dt',STATUS='OLD')
      OPEN(UNIT=2,FILE='ca.ut',STATUS='OLD')
      C1=4.106
      CO=0.084
      A=79.8

        NS=0
        I=1
30     READ(1,*,END=40) X(I),Y(I),Z(I)
        NS=NS+1
C       WRITE(2,*) X(I),Y(I),Z(I)
        I=I+1
        GO TO 30
40      N=56
      DO 125 I =1,N

      DO 125 J =1,N

      HX=X(I)-X(J)
      HY=Y(I)-Y(J)
C      HZ=Z(I)-Z(J)

        D=SQRT(HX*HX+HY*HY)
      IF (D.EQ.0)  COV=CO + C1
      IF (D.GE.A)  COV=0
      IF (D.GT.0.AND.D.LT.A) COV=C1*(1-1.5*D/A+0.5*(D**3)/(A**3))

      WRITE(2,*) COV
125     CONTINUE
      CLOSE(2)
      END
```

PROGRAM MBETA.F

```
C*******************************************************************
C THIS PROGRAM CALCULATES THE COVARIANCES OF THE SPHERICAL
C MODEL BETWEEN THE POINT WHOSE GRADE IS BEING SOUGHT AND
C THE KNOWN SAMPLE POINTS OF THE MATRIX Mβ OF THE BLUE
C EQUATION.
C
C ca.dt: THE INPUT FILE OF KNOWN POINTS COORDINATES & GRADES
C cb.dt: THE INPUT FILE OF UNKNOWN GRADES POINTS COORDINATES
C cb.ut: THE OUTPUT FILE OF THE CALCULATED COVARIANCES FOR
C KALPHA IN BLUE
C*******************************************************************

       DIMENSION X(1000),Y(1000),Z(1000),COV
       DOUBLE PRECISION COV

C      READ IN SAMPLES
C
      OPEN(UNIT=1,FILE='cb.dt',STATUS='OLD')
      OPEN(UNIT=2,FILE='cb.ut',STATUS='OLD')
      OPEN(UNIT=3,FILE='ca.dt',STATUS='OLD')
      C1 = SILL - Nuggent
      CO = Nuggent
      A  = RANGE

       NS=0
       I=1
30     READ(1,*,END=40) X(I),Y(I)
       NS=NS+1
C      WRITE(2,*)I,X(I),Y(I), Z(I)
       I=I+1
       GO TO 30
40     NP=0
       J=1
130    READ(3,*,END=41) X(J),Y(J),Z(I)
      NP=NP+1
      J=J+1
      GO TO 130
41     N=56
       P=175
      DO 125 J =1,P

      DO 125 I =1,N

      HX=X(I)-X(J)
      HY=Y(I)-Y(J)
C       HZ=Z(I)-Z(J)

        D=SQRT(HX*HX+HY*HY)
      IF (D.EQ.0)  COV=CO + C1
      IF (D.GE.A)  COV=0

      IF (D.GT.0.AND.D.LT.A) COV=C1*(1-1.5*D/A+0.5*(D**3)/(A**3))


      WRITE(2,*) COV
125     CONTINUE
      CLOSE(2)
      END
```

PITSEARCH PROGRAM LG23D.F

```
**********************************************************************
*
*Lerchs-Grossman method for determining open pit limits
*The deposit is divided into two- and three-dimensional array of rectangular blocks
*and EBV is assigned to each.  These values are stored in a two- and three-dimensional
*matrix value VAL(i, j, k) with dimensions:
*
*numx  -  number of rows
*numy  -  number of columns
*numz  -  number of levels
*
*The maximum values of these dimensions are set in a parameter statement to nx, ny
*and nz respectively.
*LOG(i, j, k) is a matrix with the same dimensions as VAL and which is used to indicate
*whether block (i, j, k) is inside (=1) or outside (=0) the pit
*IPLAN(i, j) has dimensions IK and JK respectively and defines contours of the optimum
*pit by storing the pit level at horizontal location (i, j)
*
*Other working matrices have dimensions:
*IROOT(lkm, 2), ITREE(nem), IPATH(ipkm, 3), ND(nem, 2), D(nem), NORM(knm):
*a sufficient value for each of them is numx * numy * numz /20
*        op4.dt------------input file containing Economic Block Values
*        op5.dt------------input file containing dimensions of the block model
*        op6.ut------------Output file
*
*
*Adapted from P.A. Dowd and revised to suit this study.
**********************************************************************
        Parameter (lkmax=2000,nemax=2000,ipkmax=2000,knmax=2000)
        Parameter (nxmax=50, nymax=50, nzmax=13)

        dimension val(nxmax,nymax,nzmax),log(nxmax,nymax,nzmax),
     +  d(nemax),iplan(nxmax,nymax), iroot(nemax,2),itree(nemax),
     +   nd(nemax,2),ipath(ipkmax,3),norm(knmax)

        data pi/3.14159263/
      OPEN(UNIT=4,FILE='op4.dt',STATUS='OLD')
      OPEN(UNIT=5,FILE='op5.dt',STATUS='OLD')
      OPEN(UNIT=6,FILE='op6.ut',STATUS='OLD')

900     format(40f9.1)
905     format(1x,40i2)
910     format(/20x,'Total value of blocks included in pit =',
     +          E15.5,//)
```

```
920     Format('Enter the number of blocks in the x, y, and z
   +     directions',  'respectively'/)
925  Format('Enter the x, y and z dimensions respectively of the',
   + 'blocks'/)

c       Read in data

        Write(6,920)
        read(5,*)numx,numy,numz
        write(6,925)
        read(5,*) ixdim,iydim,izdim
        do 5 k=1,numz
           do 5 i=1,numx
                 read(4,*)(val(i,j,k),j=1,numy)
5       continue
c       Initialise arrays and variables

        Do 10 i=1, numx
          do 10 j=1,numy
            iplan(i,j)=0
10      continue
        do 15 i=1,numx
        do 15 k=1,numz
                log(i,j,k)=0
                if (val(i,j,k).eq.0.00) log(i,j,k)=1
15      continue
        s=0.
        lk=0
        ne=0
        nem=0
        ipkm=0
        lkm=0
        knm=0

c       Begin with uppermost level of blocks and remove all positive valued
c       blocks. These blocks belong to optimal open pit: add thier values
c       to s, record their inclusion in the pit via log(i,j,l) and add them
c       to the contour array iplan(i,j)

        do 20 i=l, numx
          do 20 j=1,numy
                if (val(i,j,l).le.0) go to 20
                s=s+val(i,j,l)
                log(i,j,l)=1
                iplan(i,j)=1
```

```
20      continue

c       increment the level counter (k) by 1 and add the blocks on the Kth
c       level

        k=1
25     if (k.ge.numz) go to 400
        k=k+1
30      if (lk.ge.0) go to 35
32     ks=k
        go to 380
35     ltr=1

c       Connect blocks on Kth level to the root and establish trees
c       itree contains tree number
c       d     contains value of tree

40      nts=iroot(ltr,1)
        nds=itree(nts)
        if (d(nds).gt.0.) go to 55
41      if (ltr.lt.lk) then
           ltr=ltr+1
           go to 40
        endif

42     ltc=l
45      if(ltc.gt.lk) then
       ks=k
          go to 380
        endif
        nts=iroot(ltc,1)
        nds=itree(nts)
        if (d(nds).le.0.) go to 50
        lar=ltc
        lsw=4
        go to 340

50      ltc=ltc+1
        go to 45

55      lar=ltr
60     lsw=1
      go to 340
65     call coord(node,numx,numy,kl,j,i)
        if (kl.eq.1) go to 350
```

```
70    ny=(lg-1)*numx*numy+(n-1)*numx+m
       do 80 l=1,lk
         lir=1
         ntw=iroot(l,1)
         ndw=itree(ntw)
         if (d(ndw).gt.0.) go to 80
         ntk=ntw+iroot(1,2)-1
         do 75  lt=ntw,ntk
               na=itree(lt)
               na1=nd(na,1)
               na2=nd(na,2)
               if (ny.eq.na1) go to 95
               if (ny.eq.na2) go to 95
75          continue
80     continue
       go to 350

90    ny=(lg-1)*numx*numy+(n-1)*numx+m
       cpm=val(m,n,lg)
       log(m,n,lg)=2
       ne=ne+1
       nd(ne,1)=0
       nd(ne,2)=ny
       d(ne)=cpm
       itree(ne)=ne
       lk=lk+1
       iroot(lk,1)=ne
       iroot(lk,2)=1
       if (lkm.lt.Lk) lkm=lk
       if (nem.lt.ne) nem=ne
       lir=lk
95     nd(nds,1)=node
       nd(nds,2)=ny
       mbw=iroot(lir,1)
       mew=iroot(lir,2)+mbw-1
       mbs=iroot(lar,1)
       mes=mbs+iroot(lar,2)-1
       iroot(lir,2)=iroot(lir,2)+iroot(lar,2)
       iroot(lar,1)=0
       iroot(lar,2)=0
       if (mew+1-mbs) 100,140,120

100    ires=itree(mbs)
       n1=mew+1
       n2=mbs-1
```

```
          do 105 n=n1,n2
            nf=n2-n+n1
            itree(nf+1)=itree(nf)
 105      continue
          itree(mew+1)=ires
          do 110 l=1,lk
           if (iroot(l,1).eq.0) go to 110
     if(.not.(iroot(l,1).gt.mew.and.iroot(l,1).le.mbs))
    + go to 110
          iroot(l,1)=iroot(l,1)+1
 110      continue
          if (mbs.eq.mes) go to 140
          mbs=mbs+1
          mew=mew+1
          go to 100

 120      do 135 m=mbs,mes
            ires=itree(mbs)
            n1=mbs+1
            n2=mew
            do 125 n=n1,n2
                  itree(n-1)=itree(n)
 125        continue
            itree(mew)=ires
            mbw=mbw-1
            do 130 l=1,lk
                  if (iroot(l,1).eq.0) go to 130
     if (.not.(iroot(l,1).ge.mbs.and.iroot(l,1).le.mew))
    + go to 130
      iroot(l,1)=iroot(l,1)-1
 130        continue
 135      continue

 140      lcon=1
          go to 310
 145      continue
          ipa=ip
 150      n=ipath(ipa,1)
          if (n.eq.nds) go to 155
          d(n)=d(nds)-d(n)
          ipa=ipath(ipa,3)
          if (ipa.ne.o) go to 150
 155      lar=lir
          lsw=3
          go to 340
```

```
160   if (node.ne.ny) go to 350
      ipa=ip
165   nn=ipath(ipa,1)
      d(nn)=d(nn)+d(nds)
      ipa=ipath(ipa,3)
      if (ipa.ne.0) go to 165
170   kn=1
      norm(kn)=lir
175   do 180 kt=1,kn
        if (norm(kt).eq.0) go to 180
        lar=norm(kt)
        lsw=2
        go to 340
180   continue
      go to 30
185   continue
      do 190 ip=1,ipk
        if (ip.eq.1) go to 190
        md=ipath(ip,1)
        nod=iabs(ipath(ip,2))
      if (ipath(ip,2).lt.0.and.d(md).le.0.) go to 195
      if (ipath(ip,2).gt.0.and.d(md).gt.0.) go to 195
190   continue
      norm(kt)=0
      go to 175
195   nd(md,1)=0
      nd(md,2)=nod
      nod1=nod
      ipl=ip
200   iq=ipath(ip,3)
      mdl=ipath(iq,1)
      d(md1)=d(md1)-d(md)
      if (ipath(iq,3).eq.0) go to 205
      ip=iq
      go to 200

205   do 230 iq=ipl,ipk
        mc=ipath(iq,1)
        ndc=iabs(ipath(iq,2))
        naf=ipath(iq,3)
        if (ndc.eq.nod1) go to 215
        ip=naf
210     if (ip.eq.ipl) go to 215
        ip=ipath(ip,3)
        if (ip.lt.ipl) go to 230
```

```
          go to 210
 215      do 225 n=nit,nitk
               if (itree(n).ne.mc) go to 225
               if (n.eq.nitk) go to 225
               mem=itree(n)
               n1=n+1
               do 220 nz=n1,nitk
                 itree(nz-1)=itree(nz)
 220           continue
               itree(nitk)=mem
               go to 230
 225        continue
 230      continue
          do 235 n=nit,nitk
            if (itree(n).eq.md) go to 240
 235      continue
 240      iroot (lar,2)=n-nit
          lk=lk+1
          iroot(lk,1)=n
          iroot(lk,2)=nitk-n+1
          kn=kn+1
          norm(kn)=lk
          if (knm.lt.kn) knm=kn
          if (lkm.lt.lk) lkm=lk
          go to 175

 245      n=ipath(1,1)
          s=s+d(n)
          do 250 ip=1,ipk
            n=ipath(ip,1)
            nd(n,1)=0
            nd(n,2)=0
            d(n)=0.
            node=iabs(ipath(ip,2))
            call coord (node,numx,numy,kl,j,i)
            log(i,j,kl)=1
            if (iplan(i,j).lt.kl) iplan(i,j)=kl
 250      continue
          nel=ne
          n=0
 255      n=n+1
 260      if (n.eq.ne) go to 280
          if (nd(n,2).eq.0) go to 265
          go to 255
 265      n1=n
```

```
      n2=ne-1
      do 270 na=nl, n2
        nd(na,1)=nd(na+1,1)
        nd(na,2)=nd(na+1,2)
        d(na)=d(na+1)
270   continue
      ne=ne-1
      ml=nel
      do 275 m=1,ml
        if (itree(m).gt.n) itree(m)=itree(m)-1
275   continue
      go to 260
280   if (nd(ne,2).eq.0) ne=ne-1
      do 285 n=nit,nitk
        itree(n)=0
285   continue
      iroot(lar,1)=0
      iroot(lar,2)=0
      lcon=2
      go to 310
290   continue
      if (nitk.eq.nel) go to 300
      nl=nitk+1
      do 295 n=nl,nel
        itree(nit+n-nl)=itree(n)
295   continue
300   do 305 l=1, lk
        if (iroot(l,1).lt.nit) go to 305
        iroot(l,1)=iroot(l,1)-nc
305   continue
      go to 42

310   l=0
315   l=l+1
320   if (l.eq.lk) go to 335
      if (iroot(l,1).eq.0) go to 325
      go to 315
325   l1=l
      l2=lk-1
      do 330 la=l1,l2
        iroot(la,1)=iroot(la+1,1)
        iroot(la,2)=iroot(la+1,2)
330   continue
      if (lir.gt.l1) lir=lir-1
      lk=lk-1
```

```
335    if (iroot(lk,1).eq.0) lk=lk-1
       if (lk.eq.0) go to 32
       go to (145,290), lcon
340    nit=iroot(lar,1)
       nc=iroot(lar,2)
       nitk=nit+nc-1
       ipk=1
       nnd=itree(nit)
       ipath(1,1)=nnd
       ipath(1,2)=nd(nnd,2)
       ipath(1,3)=0
       ip=1
345    node=iabs(ipath(ip,2))
       nn=ipath(ip,1)
       go to (65,350,160,350),lsw
350    do 360 n=nit,nitk
         nnd=itree(n)
         if (nnd.eq.nn) go to 360
         if (node.ne.nd(nnd,1)) go to 355
         ipk=ipk+1
         ipath(ipk,2)=nd(nnd,2)
         ipath(ipk,1)=nnd
         ipath(ipk,3)=ip
         if (ipkm.lt.ipk) ipkm=ipk
         go to 360
355      if (node.ne.nd(nnd,2)) go to 360
         ipk=ipk+1
         ipath(ipk,2)=-nd(nnd,l)
         ipath(ipk,1)=nnd
         ipath(ipk,3)=ip
         if (ipkm.lt.ipk) ipkm=ipk
360     continue
       if (ip-ipk) 365,370,370
365    ip=ip+1
       go to 345
370    go to (41,185,450,245),lsw
380    im=0
       jm=0
       sm=0
       do 385 i=2,numx-1
         do 385 j=2, numy-1
               if (log(i,j,ks).gt.0) go to 385
               if (val(i,j,ks).le.0) go to 385
               if (val(i,j,ks).le.sm) go to 385
               sm=val(i,j,ks)
```

```
                 im=i
                 jm=j
                 km=ks
 385    continue
         if (sm.eq.0.) go to 25
         log(im,jm,km)=2
         ne=ne+1
         nd(ne,1)=0
         nd(ne,2)=(km-1)*numx*numy+(jm-1)*numx+im
         d(ne)=sm
         itree(ne)=ne
         lk=lk+1
         iroot(lk,1)=ne
         iroot(lk,2)=1
         if (nem.lt.ne) nem=ne
         if (lkm.lt.lk) lkm=lk
         nds=ne
         ltr=lk
         go to 55

c        print results

400     write(6,910) s
         kollu=0
         kol=0
         koll=0
         do 415 k=l, numz
           do 410 j=1,numy
                 do 405 i=1, numx
     if (log(i,j,k).eq.1) iplan(i,j)=k
     if (log(i,j,k).eq.1.and.val(i,j,k).lt.0) kol=kol+1
     if (log(i,j,k).eq.1.and.val(i,j,k).gt.0) koll=koll+1
     if (log(i,j,k).eq.1.and.val(i,j,k).ne.0) kollu=kollu+1
405             continue
410          continue
415      continue
         write(6,*)'total number of blocks in pit',kollu
         write(6,*)'number of positive blocks in pit',koll
         write(6,*)'number of negative blocks in pit',kol
         do 420 i=l, numx
           write(6,905) (iplan(i,j),j=1,numy)
420      continue
         stop
450     zz=1
         write(6,900) zz,nem,lkm,node
```

```
        close(4)
        close(5)
        close(6)
        stop
        end
        subroutine coord(n,numy,k,j,i)

c       subroutine to determine the array index co-ordinates of node (block)
c       n given that there are ik (x direction) x jk (y direction) nodes
c       on each horizontal level
c       Array index co-ordinates are returned as (i,j,k)

        kt=n/(numx*numy)
        k=kt+1
        if (n.eq.kt*numx*numy) k=k-1
     jt=(n-numx*numy*(k-1))/numx
        j=jt+1
        if ((n-numx*numy*(k-1)).eq.ik*jt) j=j-1
        i=n-numx*numy*(k-1)-numx*(j-1)
        return
        end
```

```
c               PROGRAM PITSEARCH FOR THE MFNN MODEL : PMFFN.F
c*********************************************************************
**
c       a, q =slope angle, b,r =intercepts, x,y=coordinates, z=E.B.V
c se, sw = east and west slope walls equatione respectively.
c       se = y - ax - b
c       sw = y -qx - r
c       pve = Optimum Pit Value when East Slope Wall moves and
c           West Slope Wall is Stationary
c
c     pve = Optimum Pit Value when West Slope Wall moves and
c         East Slope Wall is Stationary
c
c  op.dt: THE INPUT FILE OF BLOCK COORDINATES & EBVs
c  op.ut: THE OUTPUT FILE
c  g() : grade
c*********************************************************************
*
        Dimension se(300),sw(300),x(300),y(300),ebv(300)
c*********************************************************************
*
c Read in Gridded Blocks Coordinates & Economic Block Values
c*********************************************************************
*

        open (unit = 3, file = 'op.dt', status = 'old')
        open (unit = 4, file = 'op.ut', status = 'old')

c**********************************************************
      NS=0
      I=1
30     READ(3,*,END=40) x(i),y(i),ebv(i)
      NS=NS+1

      I=I+1
      GO TO 30



40      do 10 i = 1, 300
        sum = 0

        a=1
        q=1
      b=0.5
        r=0.5
        se(i) = y(i) - a*x(i) - b
      sw(i) = y(i) + q*x(i) - r

c  Shifting East Slope Wall while West is constant
c******************************************************************
        if ((se(i).gt.0).and.(sw(i).gt.0)) then
               sum = sum + ebv(i)

               pve = sum

               write(4, *) x(i),y(i),ebv(i), 'is included in pit'
        else
               write(4, *) x(i),y(i),ebv(i), 'is not included in pit'

               write(4, *) 'Pit Value = ',sum

        b = b + 0.5

        if (b.ge.100) write (4, *) 'End Calculation for pve'


        go to 140

        end if

  10   continue
  50   continue

c  Shifting West Slope Wall while East Slope Wall is constant
c******************************************************************

140    do 110 i = 1, 300
      sum = 0

      a=1
      q=1

      b=0.5
      r=0.5
      se(i) = y(i) - a*x(i) - b
      sw(i) = y(i) + q*x(i) - r

      if ((se(i).gt.0).and.(sw(i).gt.0)) then
```

```
          sum = sum + ebv(i)

          pvw = sum

          write(4, *) x(i),y(i),ebv(i), 'is included in pit'
    else
          write(4, *) x(i),y(i),ebv(i), 'is not included in pit'

          write(4, *) 'Pit Value = ',sum

          r = r + 0.5

    if (r.ge.100) write (4, *) 'End Calculation for pvw'

       go to 170
    end if

110   continue
150   continue

170   if (pve.ge.pvw) write (4, *) 'Optimum Pit Value =',pve
    if (pvw.ge.pve) write (4, *) 'Optimum Pit Value =',pvw


       close(3)
       close(4)

       end
```