Optimal design of drillhole locations using geostatistics

Zul Ichwan
University of Wollongong

UNIVERSITY OF WOLLONGONG
COPYRIGHT WARNING

You may print or download ONE copy of this document for the purpose of your own research or study. The University does not authorise you to copy, communicate or otherwise make available electronically to any other person any copyright material contained on this site. You are reminded of the following:

This work is copyright. Apart from any use permitted under the Copyright Act 1968, no part of this work may be reproduced by any process, nor may any other exclusive right be exercised, without the permission of the author.

Copyright owners are entitled to take legal action against persons who infringe their copyright. A reproduction of material that is protected by copyright may be a copyright infringement. A court may impose penalties and award damages in relation to offences and infringements relating to copyright material. Higher penalties may apply, and higher damages may be awarded, for offences and infringements involving the conversion of material into digital or electronic form.

Unless otherwise indicated, the views expressed in this thesis are those of the author and do not necessarily represent the views of the University of Wollongong.

Recommended Citation
OPTIMAL DESIGN OF DRILLHOLE LOCATIONS

USING GEOSTATISTICS

A thesis submitted in partial fulfilment of the requirements for the award of the degree of

Master of Engineering (Honours)

from

THE UNIVERSITY OF WOLLONGONG

by

ZUL ICHWAN
B.Eng. A.G.P Bandung, Indonesia

DEPARTMENT OF CIVIL AND MINING ENGINEERING
1990
"In The Name of Allah, The Beneficient and The Merciful"
DECLARATION

This research thesis has been purposefully and originally undertaken by the Author for a degree of Master of Engineering (Honours) at the University of Wollongong. This thesis has not been submitted for the same purpose to any other University.

ZUL ICHWAN
September, 1990
ACKNOWLEDGEMENTS

The research for a degree of Master Engineering has been completed under the marathon supervision of Dr. E.Y. Baafi together with the numerous discussions and invaluable suggestions which he has given throughout the whole study on Geostatistics Application in Mining and Exploration.

Special acknowledgements are due to my lovely wife, TIKTIK MULYATI, for her spiritual supports and her patient waiting for my completion of this study and during the process of compiling this thesis, and to my father and mother who imbued me with the religious spirit to succeed.

I wish to thank The Australian International Development Assistance Bureau (AIDAB) for their sponsorship in my study, and also the Head of Directorate of Mineral Resources, Indonesia, Ir. Salman Padmanagara, for releasing me to study at the University of Wollongong, NSW. Thanks are also given to Ir. Suryantoro, MSc, for encouraging me to study in Australia, and to Prof. Alan Hargraves who edited this thesis.
Optimal Sampling Design has been developed using FORTRAN-77L language. The developed program is an easy-to-use program with a friendly system. The input and output phase programs were made by way of multi-window displays in order to speed up the input or output data access.

The program developed is termed OPSAMP (Optimal Sampling Program). The basic methodology of the selection of optimal sampling uses the Branch and Bound method based on the estimation (kriging) variance method. The selection of the best set of additional samples is obtained if the minimum estimation (kriging) variance computation can be achieved.

These bases of applications are fruitful aids to either geologists or mining engineers in developing optimal sampling strategy in the area where there are drillholes existing from a mineral exploration.

A case study is also presented using drillhole data of a coal deposit located near Homer City, Pennsylvania. In this thesis, the alternative study was conducted for determination of an optimum number of potential additional drillholes required for the above purpose.

The result of the study provides significant improvement in optimal sampling development. Based on using the developed OPSAMP program, the combination of proposed additional drillholes was obtained as better than drilling programs of the others.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title Page</td>
<td>i</td>
<td></td>
</tr>
<tr>
<td>Declaration</td>
<td>ii</td>
<td></td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>iii</td>
<td></td>
</tr>
<tr>
<td>Abstract</td>
<td>iv</td>
<td></td>
</tr>
<tr>
<td>Table of Contents</td>
<td>vi</td>
<td></td>
</tr>
<tr>
<td>List of Tables</td>
<td>viii</td>
<td></td>
</tr>
<tr>
<td>List of Figures</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

Chapter 1. INTRODUCTION 1 - 1

1.1 General introduction 1 - 1
1.2 Design of Sampling Program 1 - 3
1.3 Estimation Error 1 - 5
1.4 Definition of Problem 1 - 8
1.5 Scope of work 1 - 8

Chapter 2. DESIGN OF OPTIMAL DRILL-SITES 2 - 1

2.1 The Mathematical Model proposed 2 - 1
2.2 The Branch and Bound Procedure 2 - 3
2.3 The Use of Branch and Bound Procedure in Sampling Selection 2 - 5
2.4 The Block Inverse Technique by Partition 2 - 8
2.5 The Procedure of Kriging Variance Modification 2 - 17
<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 3.</td>
<td><strong>OPSAMP COMPUTER PROGRAM</strong></td>
<td>3 - 1</td>
</tr>
<tr>
<td>3.1</td>
<td>An Overview of Programme <em>OPSAMP</em></td>
<td>3 - 2</td>
</tr>
<tr>
<td>3.2</td>
<td>The <em>OPSAMP</em> Modules</td>
<td>3 - 4</td>
</tr>
<tr>
<td>3.3</td>
<td>The User Interface System</td>
<td>3 - 7</td>
</tr>
<tr>
<td>3.4</td>
<td>The <em>PCHAND</em> and the <em>ANSI</em> Escape Sequences</td>
<td>3 - 8</td>
</tr>
<tr>
<td>3.5</td>
<td>The <em>PCHAND</em> System</td>
<td>3 - 14</td>
</tr>
<tr>
<td>3.6</td>
<td>The <em>OPSAMP</em> Menu System</td>
<td>3 - 34</td>
</tr>
<tr>
<td>3.7</td>
<td>The Program <em>TREE</em></td>
<td>3 - 39</td>
</tr>
<tr>
<td>3.8</td>
<td>Additional Options to <em>OPSAMP</em> Programme</td>
<td>3 - 42</td>
</tr>
<tr>
<td>3.9</td>
<td><em>OPSAMP</em> Library System And Utilities</td>
<td>3 - 43</td>
</tr>
<tr>
<td>Chapter 4.</td>
<td><strong>VALIDATION OF <em>OPSAMP</em> PROGRAM</strong></td>
<td>4 - 1</td>
</tr>
<tr>
<td>4.1</td>
<td>Procedure of <em>OPSAMP</em> Input Phase</td>
<td>4 - 1</td>
</tr>
<tr>
<td>4.2</td>
<td>Branch and Bound Logic And Response</td>
<td>4 - 18</td>
</tr>
<tr>
<td>4.3</td>
<td>Estimation (Kriging) Variance Calculation with the <em>TREE</em> Module</td>
<td>4 - 21</td>
</tr>
<tr>
<td>4.4</td>
<td>An Application of <em>OPSAMP</em> program</td>
<td>4 - 23</td>
</tr>
<tr>
<td>Chapter 5.</td>
<td><strong>SUMMARY AND CONCLUSIONS</strong></td>
<td>5 - 1</td>
</tr>
<tr>
<td>5.1</td>
<td>Summary</td>
<td>5 - 1</td>
</tr>
<tr>
<td>5.2</td>
<td>Conclusions</td>
<td>5 - 2</td>
</tr>
<tr>
<td>5.3</td>
<td>Recommendations for future work</td>
<td>5 - 3</td>
</tr>
<tr>
<td>TITLE</td>
<td>PAGE</td>
<td></td>
</tr>
<tr>
<td>------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>B - 1</td>
<td></td>
</tr>
<tr>
<td>APPENDICES</td>
<td>I - 1</td>
<td></td>
</tr>
<tr>
<td>I.  Listing of Program Validation</td>
<td>I - 1</td>
<td></td>
</tr>
<tr>
<td>II. Listing of <em>OPSAMP</em> Program Application</td>
<td>II- 2</td>
<td></td>
</tr>
<tr>
<td>III. Listing of <em>OPSAMP</em> Computer Program</td>
<td>III- 3</td>
<td></td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Example of Namelist File</td>
</tr>
<tr>
<td>3.2</td>
<td>Display Colour Setting</td>
</tr>
<tr>
<td>3.3</td>
<td>Cursor control</td>
</tr>
<tr>
<td>3.4</td>
<td>Display Mode</td>
</tr>
<tr>
<td>3.5</td>
<td>The Namelist File of Estimation variance <em>GLO.STR</em> for text structure menu</td>
</tr>
<tr>
<td>3.6</td>
<td>The Namelist File of Estimation variance <em>GLO.PAR</em> for Text Option Menu</td>
</tr>
<tr>
<td>3.7</td>
<td>Structure of Text File Model</td>
</tr>
<tr>
<td>3.8</td>
<td>Structure of Run Input File Model</td>
</tr>
<tr>
<td>3.9</td>
<td>The <em>OPSAMP.LIB</em> content</td>
</tr>
<tr>
<td>3.10</td>
<td>The <em>MENU.LIB</em> content</td>
</tr>
<tr>
<td>3.11</td>
<td>The <em>PCHAIN.LIB</em> content</td>
</tr>
<tr>
<td>4.1</td>
<td>Keyboard Convention Table</td>
</tr>
<tr>
<td>4.2</td>
<td>Block Area Definition for Program Validation</td>
</tr>
<tr>
<td>4.3</td>
<td>Example of Drillhole Information</td>
</tr>
<tr>
<td>4.4</td>
<td>The Output of <em>TREE</em> Search Procedure</td>
</tr>
<tr>
<td>4.5</td>
<td>Coordinates of the Block Centres</td>
</tr>
<tr>
<td>4.6</td>
<td>Summary of <em>OPSAMP</em> program for Estimation variance calculation of Sulfur</td>
</tr>
<tr>
<td>4.7</td>
<td>Summary of <em>SVAR</em> Program for Estimation variance calculation of Sulfur</td>
</tr>
<tr>
<td>TABLE</td>
<td>PAGE</td>
</tr>
<tr>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>4.8</td>
<td>Summary of <em>OPSAMP</em> Program for Kriging variance Calculation of Sulfur</td>
</tr>
<tr>
<td>4.9</td>
<td>Summary of Standard Kriging Program for Kriging variance Calculation of Sulfur</td>
</tr>
<tr>
<td>4.10</td>
<td>Statistical Parameters</td>
</tr>
<tr>
<td>4.11</td>
<td>Spherical Variogram Parameters</td>
</tr>
<tr>
<td>4.12</td>
<td>Proposed Additional Drillholes by Company and Cervantes</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Two Unbiased Estimators with Different Estimation variance</td>
<td>1 - 6</td>
</tr>
<tr>
<td>2.1 Flow Diagram of TREE Search Procedure</td>
<td>2 - 4</td>
</tr>
<tr>
<td>3.1 Flow Diagram of the OPSAMP Program</td>
<td>3 - 3</td>
</tr>
<tr>
<td>3.2 Definition of an Irregular Mining Area</td>
<td>3 - 6</td>
</tr>
<tr>
<td>3.3 Definition of a Regular Mining Area</td>
<td>3 - 7</td>
</tr>
<tr>
<td>3.4 Displaying an OPSAMP Logo</td>
<td>3 - 13</td>
</tr>
<tr>
<td>3.5 The PCHAND System</td>
<td>3 - 14</td>
</tr>
<tr>
<td>3.6 Displaying an OPSAMP Logo (from more sophisticated analysis but unchanged display from Fig. 3.4)</td>
<td>3 - 16</td>
</tr>
<tr>
<td>3.7 The Namelist File containing the OPSAMP Logo</td>
<td>3 - 16</td>
</tr>
<tr>
<td>3.8 The Main Menu Display</td>
<td>3 - 18</td>
</tr>
<tr>
<td>3.9 Setting the Namelist File of the OPSAMP Main Menu</td>
<td>3 - 19</td>
</tr>
<tr>
<td>3.10 Displaying the OPSAMP Sub-bar Menu</td>
<td>3 - 22</td>
</tr>
<tr>
<td>3.11 The First Input Phase of the Editing File</td>
<td>3 - 23</td>
</tr>
<tr>
<td>3.12 The Second Input Phase of the Editing File</td>
<td>3 - 23</td>
</tr>
<tr>
<td>3.13 The Third Input Phase of the Editing File</td>
<td>3 - 24</td>
</tr>
<tr>
<td>3.14 The FORMAT.STR</td>
<td>3 - 25</td>
</tr>
<tr>
<td>3.15 The Namelist File of BGROUND.DRW containing the ASCII characters</td>
<td>3 - 29</td>
</tr>
<tr>
<td>3.16 The Namelist File of BOX.DRW containing ASCII characters</td>
<td>3 - 30</td>
</tr>
<tr>
<td>3.17 The Namelist File of FLASH.FLS</td>
<td>3 - 33</td>
</tr>
<tr>
<td>FIGURE</td>
<td>PAGE</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>3.18</td>
<td>Flow Process of the <em>OPSAMP</em> Program 3 - 35</td>
</tr>
<tr>
<td>3.19</td>
<td>Input Phase Display of Program <em>SVAR</em> 3 - 37</td>
</tr>
<tr>
<td>4.1</td>
<td><em>OPSAMP</em> Logo 4 - 3</td>
</tr>
<tr>
<td>4.2</td>
<td><em>OPSAMP</em> Screen 4 - 4</td>
</tr>
<tr>
<td>4.3</td>
<td>Bar Menu for the Sampling Program 4 - 5</td>
</tr>
<tr>
<td>4.4</td>
<td>The First Input Phase of the Editing Data File 4 - 6</td>
</tr>
<tr>
<td>4.5</td>
<td>The Second Input Phase of the Editing Data File 4 - 6</td>
</tr>
<tr>
<td>4.7</td>
<td>The First Window of Input Phase of Scatter Program 4 - 8</td>
</tr>
<tr>
<td>4.8</td>
<td>The Second Window of Input Phase of Scatter Program 4 - 8</td>
</tr>
<tr>
<td>4.9</td>
<td>The Third Window of Input Phase of Scatter Program 4 - 9</td>
</tr>
<tr>
<td>4.10</td>
<td>The First Window of Input Phase of Declustering Program 4 - 10</td>
</tr>
<tr>
<td>4.11</td>
<td>The Second Window of Input Phase of Declustering Program 4 - 10</td>
</tr>
<tr>
<td>4.12</td>
<td>The Third Window of Input Phase of Declustering Program 4 - 11</td>
</tr>
<tr>
<td>4.13</td>
<td>The First Window of Input Phase of Estimation variance Program 4 - 11</td>
</tr>
<tr>
<td>4.14</td>
<td>The Second Window of Input Phase of Estimation variance Program 4 - 12</td>
</tr>
<tr>
<td>4.15</td>
<td>The Third Window of Input Phase of Estimation variance Program 4 - 12</td>
</tr>
<tr>
<td>4.16</td>
<td>The Fourth Window of Input Phase of Estimation variance Program 4 - 13</td>
</tr>
<tr>
<td>FIGURE</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>4.17</td>
<td>The Fifth Window of Input Phase of Estimation variance Program</td>
</tr>
<tr>
<td>4.18</td>
<td>The Last Window of Input Phase of Estimation variance Program</td>
</tr>
<tr>
<td>4.19</td>
<td>The First Window of Input Phase of Simple Kriging Program</td>
</tr>
<tr>
<td>4.20</td>
<td>The Second Window of Input Phase of Simple Kriging Program</td>
</tr>
<tr>
<td>4.21</td>
<td>The Third Window of Input Phase of Simple Kriging Program</td>
</tr>
<tr>
<td>4.22</td>
<td>The Fourth Window of Input Phase of Simple Kriging Program</td>
</tr>
<tr>
<td>4.23</td>
<td>The Fifth Window of Input Phase of Simple Kriging Program</td>
</tr>
<tr>
<td>4.24</td>
<td>The Last Window of Input Phase of Simple Kriging Program</td>
</tr>
<tr>
<td>4.25</td>
<td>The Window of the Run Option Menu</td>
</tr>
<tr>
<td>4.26</td>
<td>TREE Search of 4 New Samples</td>
</tr>
<tr>
<td>4.27</td>
<td>Block Assigned for OPSAMP Validation</td>
</tr>
<tr>
<td>4.28</td>
<td>Kriging Block Definition with Four Block Data</td>
</tr>
<tr>
<td>4.29</td>
<td>The Property Area and Locations of Existing and Additional Drillholes</td>
</tr>
<tr>
<td>4.30</td>
<td>Experimental and Fitted Variogram for Thickness</td>
</tr>
<tr>
<td>4.31</td>
<td>Experimental and Fitted Variogram for Sulfur</td>
</tr>
<tr>
<td>4.32</td>
<td>Comparison of Development Drilling Design with and without OPSAMP Program</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION
CHAPTER 1
INTRODUCTION

1.1 GENERAL INTRODUCTION

Prior to the development of a mine, detailed geological studies of the deposit are normally carried out by geologists or mining geologists. Mineral exploration involves the step by step proving of a deposit from orientation to production. The sequence of these steps is given in the following:

a. Orientation. This is undertaken where sampling (drilling) already exists in the preliminary survey of a mineral province. Few drillholes are normally located in the actual target area. Core and cutting samples are collected so as to provide the background information of the prospect area or mineralized zone. When the prospect area is extensively explored, the next step is to determine the existence of mineralization by reconnaissance.

b. Reconnaissance. This step is often performed for regional stratigraphic or lithologic information and interpretation. Some drillings for reconnaissance are always undertaken but drilling is not necessarily at regular spacing. The result of this interpretation is the basis on which to plan further reconnaissance for a target area.
c. Target Area Search. Most of the work in this step will involve the use of drillhole data for discovering the presence of mineralization. In this step, the geological information needed is for verification of the degree of mineralization as well as for qualitative description such as the mineralogical consist of ore and metallogenic type as well as details of faults, tectonics and other geological conditions. From this, exploration progresses to a prospect evaluation and the specification of the overall in situ resources by grade and tonnage.

d. Prospect Evaluation. The prospect area containing the mineralization is outlined, then the samples are drilled on a more or less regular grid within the area of the mineralized zone in order to define the size of ore body deposit as well as to determine its grade. On the basis of the results gained from these drillholes, in-fill drilling may be required to extend the accuracy of the survey of the mineralized zone. During this phase, two basic problems are normally faced by the geologist and mining engineer. The first problem is the global ore reserve estimation problem. The second problem is the confidence of the ore reserve assessment which should be based on the Australian Code for Reporting of Identified Mineral Resources and Ore Reserve.

e. Ore Reserve Calculation. In this step, the basis of accumulating the amount of ore reserve is calculated before deciding to mine. Drilling is performed in order to delineate the orebody or deposit. Some guidelines such as geotechnical and metallurgical investigations and mine development are taken into consideration. In this step, it will be shown that the methodology of geostatistics can solve this problem by means of structure analysis using variogram.
f. **Mining.** In the mine prospect area the ongoing drilling procedure is performed under control of the geologist. It is usual to drill on a small spaced grid which is required for the first production.

The above exploration steps are of considerable importance before the final contribution of an optimal drilling program for the life of the mine is conducted.

### 1.2 DESIGN OF SAMPLING PROGRAM

The principal aim of the sampling program is to collect information to define the distribution of population. Collections of samples which represent the typical distribution of population are broadly so-called "sampling". Samples are normally collected from the population which begins from a large to a small sampling domain. The possible information obtained relies obviously on the number of samples and their relative locations.

The procedure to group the samples is commonly called the sample survey design. Three procedures of sampling survey design are often used in mineral deposit evaluation. These procedures are shown in the following:

1) **Simple Random Sampling** - A collection of samples of some numbers $n$ is randomly drawn from the population. For example, 5 groups of samples with $n$ samples, then from each group, one sample is collected at random.
2) Stratified Random sampling - Samples are collected randomly from each separate stratum. This is useful for mineral deposits consisting of two or more different rock types. For example, let there be \( n \) strata of a deposit with proportion of total population \( L_i \). The mean sample is simply calculated as

\[
\overline{X_s} = \sum_{i=1}^{n} L_i \overline{X_i}
\]  

(1.1)

where \( \overline{X_i} \) is the mean of sample of the \( i \)th stratum.

3) Systematic Sampling - A sample of size \( n \) is selected. If the number of elements is \( N \), then \( n \) collected samples are divided by \( N \) into \( M \) segments \((M=n/N)\). For instance, if there are 50 major elements, a sample of 5 is selected and then one sample from each segment of 10 elements is collected. In this case, it will be taken in the same sequence to give (5, 15, 20, 35, etc) elements. This procedure is preferentially used for a typical massive deposit such as a massive gold deposit.

The above sampling procedures can be related to sampling pattern and spacing because the pattern of sampling and its spacing are based on the sampling procedure, the accessibility of area, the geologic condition, the requirements for geostatistical analysis, and the achievement. For instance, uniform grid spacing is used preferentially for a massive deposit so that the optimal sampling using geostatistic can be achieved.

For short term mine planning, grade control has two functions (Alexander, 1986). The first is to elucidate the boundaries of blocks which are to be mined such as ore-waste boundaries, and the second is to accurately predetermine the grade of the ore mined during the day. In order to achieve both
functions, blastholes must be sampled from the bench. In the case of open pit mining, commonly blasthole kriging is used for estimation. This is essentially as a basis of application for ore-waste control problems. To a large extent most of the grade control problems depend on these two functions because it is important to improve mined block grades.

In the case of blasthole sampling, the problem of identifying sampling sites involves a selection of a subset of locations from pre-determined locations. As the blasthole samples are principally used to improve the tonnage of the mean grade, the next blasting procedure should depend upon the information obtainable of ore from the previous drilling. This necessarily would involve the determination of potential drill sites in advance by the geologist. This thesis concentrates on the above sampling problems.

1.3 ESTIMATION ERROR

An error is included when the estimation of the block grade from adjacent samples is made. For any circumstances, this error is unknown but it can be predicted from the certain parameter, e.g., estimation variance. The estimation variance is defined as the variance of the error of estimation between two random functions, $Z(v) - Z(V)$, where $Z(v)$ is the estimated value within the block $v$, $Z(V)$ being the true value of the block $V$. This variance of the error or well-known Estimation Variance is given as:

$$\sigma_e^2 = \text{Var}[Z(V) - Z(v)]$$  \hspace{1cm} (1.2)
In terms of the geostatistic point of view, the estimated grade in the block is defined as

\[ Z(v) = \frac{1}{n} \sum_{i=1}^{n} Z(x_i) \]  \hspace{1cm} (1.3)

where:

- \( Z(v) \) = average value within the block \( v \),
- \( n \) = number of adjacent samples,
- \( Z(x_i) \) = value of samples at location \( x_i \).

As the unbiased estimator must be fulfilled, the expected error is defined as

\[ E[Z(V)-Z(v)]=0 \]  \hspace{1cm} (1.4)

The above estimator must be unbiased with a minimum estimation variance. In other words, the estimator must not be spread over the mean sample of the population as shown in Fig 1.1 where estimator 1 and estimator 2 are unbiased but estimator 2 has smaller variance than estimator 1.

**Fig 1.1. Two unbiased estimators with different estimation variance**
Using the semi-variogram, the estimation variance can be calculated from derivation of Equation (1.2), which becomes

\[ \sigma_e^2 = 2\gamma(V,v) - \gamma(V,V) - \gamma(v,v) \]  

(1.5)

where:
- \( \gamma(V,v) \) = average variogram between block and samples,
- \( \gamma(V,V) \) = average variogram between samples in the block,
- \( \gamma(v,v) \) = average variogram between samples themselves.

The above formulation suggests that the estimation variance computation depends on the sample locations and not the actual sample values from additional drillings, even though this development drilling strategy would physically disturb the mean value of the deposit.

Therefore, the use of estimation variance computation provides the determination of the impact of placing one or more additional samples in the estimate. For instance, it is possible to know how much the accuracy can be obtained by giving one or more additional samples within the confidence interval of 95% of the overall estimate. This is the advantage of the method, one can simulate the future additional drilling and select the best locations of the holes during the reserve estimation. Also, the above capability can be based on the available budget concern dealing with the cost-benefit of the marginal improvement of ore information over the marginal drilling cost of additional drillholes.
1.4 DEFINITION OF PROBLEM

Normally collecting samples creates a number of errors (e.g. sampling error). There are four factors affecting the estimation accuracy, whether it is global or local estimation. These are the underlying variogram, the relative size of the blocks being estimated, sample location and method of estimation within the area under study.

An evaluation of a deposit involves both collecting samples within the deposit and using an appropriate estimation technique. In most practical situations, the geologist would identify in advance the potential drillhole sites on the basis of some geological phenomena. There is usually a budget which must be met in drilling at all the pre-determined sites. With limited funds, the geologist will face the problem of selecting the best drillhole sites from these pre-determined sites.

1.5 SCOPE OF WORK

The purpose of the study is to use geostatistics in solving the optimum development sampling. The scope of the thesis is limited to both global and local estimation procedure to solve the optimal sampling problem on the basis of variance error calculation. The method of the selection of the new drillholes from a subset of the proposed drillholes candidates is determined by sequentially selecting the "best" set among all the candidates. This approach assumes that the variogram model is available for the deposit. The main task will involve developing a Branch and Bound technique for the purpose of searching for the best potential drillhole sites from pre-determined locations.
CHAPTER 2

DESIGN OF OPTIMAL
DRILL-SITES
As mentioned above, the evaluation of a deposit involves both collecting samples within the deposit and using an appropriate estimation technique, then the geologist identifies in advance the potential drillhole sites on the basis of some geological phenomena. In this thesis, a mathematical model is proposed to assist the geologist or mining engineer in selecting the optimal design of sampling program. This model utilises a \textit{BRANCH} and \textit{BOUND} procedure so as to select the best set of samples from the pre-determined samples. The procedure minimizes the estimation (\textit{kriging}) variance corresponding to the available drilling budget. A concept of global and local estimation on the basis of geostatistic method is used to achieve the optimal design of drilling strategy.

\section{THE MATHEMATICAL MODEL PROPOSED}

In general, an ore deposit is evaluated from the available drillhole samples. The $m$ holes are drilled at locations, $x_1, x_2, \ldots, x_m$. From these drillholes each sample value at each location, $z(x_1), z(x_2), \ldots, z(x_m)$ is determined. The evaluation of the deposit can be performed on the basis of this data using a geostatistic method of editing global and local estimation procedure. If the budget constraint requires that $p$ less potential drillhole candidates will not be drilled, the $(m-p)$ sites must be chosen in the estimation process. The problem then reduces to selecting $(m-p)$ drill sites from $m$ which minimise estimation
The mathematical model of this relationship is then formulated as follows:

The optimal model of drillhole selection (existing for \( j=1,...,n \), and additional samples \( s=1,2,...,m \)), then the mathematical model becomes:

\[
\text{minimize } \text{Var}(x_1, x_2,...,x_n, s_1, s_2,...,s_m)
\]

subject to \( \{s_1, s_2,...,s_{m-p}\} \subseteq \{s_1, s_2,...,s_m\} \)

The above mathematical model can also be termed the multiple objective function. From this model there is the best selection regarding which objective functions give the optimal solution. In this circumstance, the optimal sample can be efficient if no other best selection similarly minimizes the estimation (kriging) variance. In terms of efficiency, the conditional optimality for the best selection is governed as follows:

It is assumed that the initial selection of \( \{s_1, s_2,...,s_{m-p}\} \) from the additional drillholes can be optimal if and only if those potentials are of the best selection from the other selection \( \{s_1, s_2,...,s_{m-p_i}\} \) for \( i=1,2,...,m-1 \) and \( p_i < m \) such that the model would be:

\[
\text{Var}(x_1, x_2,...,x_n, s_1, s_2,...,s_{m-p_1}) \geq \text{Var}(x_1, x_2,...,x_n, s_1, s_2,...,s_{m-p_i}),
\]

and at least for one condition \( i_0 \),

\[
\text{Var}_{i_0}(x_1, x_2,...,x_n, s_1, s_2,...,s_{m-p_1}) \geq \text{Var}_{i_0}(x_1, x_2,...,x_n, s_1, s_2,...,s_{m-p_1}).
\]
Chapter 2 Design of optimal drill-sites

If the above selection is not optimal or so-called in-efficient due to the domination of the other combination which gives efficiently the best selection, the above initial model is shifted to the new model, e.g., by dropping \( p_2 \), as follows:

\[
\text{Vari}(x_1, x_2, ... x_n, s_1, s_2, ..., s_{m-p_2}) \geq \text{Vari}(x_1, x_2, ... x_n, s_1, s_2, ..., s_{m-p_1})
\]  

(2.3)

And at least for one condition \( i_0 \),

\[
\text{Vari}_{i_0}(x_1, x_2, ... x_n, s_1, s_2, ..., s_{m-p_1}) \geq \text{Vari}_{i_0}(x_1, x_2, ... x_n, s_1, s_2, ..., s_{m-p_1})
\]

Generally speaking, the efficiency can be achieved if the improvement of estimation (kriging) variance can not be tolerated anymore. Consequently, the model proposed using Branch and Bound procedure is seemingly to select the best selection from all efficient selection of the potential additional drill holes.

2.2 THE BRANCH AND BOUND PROCEDURE

Branching and Bounding is generally used in operational research to search the achievable solution of the problem which can shift the initial objective function. This is so-called the optimal modified continous point. Thus, the method used is the object sequential technique which starts from small set to finite set of samples.

The Branch and Bound procedure is also named the Decision Tree Search process (Fig 2.1). To illustrate the search procedure, assuming a set of points, \( x_1, x_2, ..., x_n \), corresponding to any variable concerned, e.g. drillholes, a search is made for the best combination (selection) for optimal purposes. The procedure
searches continuously in the direction of a set of points. The selections are then taken at a continuous stage of a set of points and the bounded points are conditional on the decision at the previous stage, the current efficient selection is held if the smallest criterion is achieved.

The mechanism of this procedure is to search any root of the branch of the Tree and to bound the efficient selection of any combination of the points as an optimal solution. The procedure will trace back to when the process arrives at the last search (the last point of the set). Consequently, the best set is the efficient selection of the point set satisfying the constraint.

Fig 2.1 Flow diagram of Tree Search procedure
2.3 THE USE OF BRANCH AND BOUND PROCEDURE IN SAMPLING SELECTION

In mine planning, an efficient analysis of the qualitative and quantitative parameters of the ore deposit requires the optimal number of samples. The preferable method used for optimal design of a sample program is a concept of regionalised variable on the basis of geostatistic method. The available samples are, however, quite often unreliable to represent the entire deposit due to many factors. One of the factors concerned is the need of more samples creating the higher cost of drilling which cannot economically be avoided in these circumstances. The problem of how many samples are required for the next drilling is to arrive at the optimal number. Using geostatistics together with the use of the Branch and Bound procedure should likely solve the problem of defining the optimum samples.

The Branch and Bound was introduced by the early work of Szidarovsky and Gershon (1984) for solving the problem of selection for the optimal sample. The procedure was originally adapted from the application of integer programming.

Branch and Bound is actually the systematic procedure developed for branching a set of any combination of variables (e.g., a set of drillhole data) and bounding the optimal solution to satisfy the given constraint. This is applied in selection of the best drillhole sites from a set of proposed additional sampling candidates on the basis of estimation (kriging) variance. As the method searches a feasible solution from a number of possible combinations (subsets), then the updating estimation variance resulting from the global or local estimation modifies the old solution (the previously calculated estimation (kriging) variance). This is
particularly useful to select any combination of potential additional samples in order to improve the reliability of the ore estimate.

The purpose of the use of the Branch and Bound procedure is two-fold. One is to quickly search the combination of the potential drillholes which give the best set of samples; the other is to improve the reliability of the estimate based on these additional drillholes.

The application of the Branch and Bound procedure has potential use after calculation of the kriging system. In other words, the number of samples selected including the proposed potential additional samples, have been in the kriging matrix system. Once this matrix has been calculated, the number of additional samples selected from the kriging process is retained, then the Branch and Bound procedure will keep this set for its initial process of searching the best set of samples.

To illustrate the use of the Branch and Bound search procedure in a selection of drillhole data, the mechanism of procedure is described as follows:
Assume that an area $D$ contains a set of existing sample points $D(x_1,x_2,x_3,x_4,\ldots,x_n)$ where $x_n$ are the locations of drillholes for $i=1,2,\ldots,n$. Then, $m$ additional drillholes, $D(x_{n+1},\ldots,x_m)$ where $x_m$ are the locations of additional samples for $j=1,\ldots,m$ are predetermined by the geologist in this area but the economically feasible number of drillholes is required by the manager because of limited budget. For instance, four additional holes are proposed. In this case, this is the initial set of additional drillholes for $j=1,\ldots,4$ in the Branch and Bound procedure (Fig 2.1). First, By applying the mathematical model in Equation (2.1), the estimation (kriging) variance of the entire set of drillholes
(existing and additional) is computed. Then, the procedure is directed to the Branch where the new subset of $m-1$ elements has been in root 1. In this stage, one hole has been dropped. Then, the computation of estimation \textit{(kriging)} variance using a mathematical model in Equation (2.2) is performed for a dropping procedure. If this estimation error is still not efficient, then it will go to the next step and drop a combination of 1, 2 or $p=2$ and do the same procedure. If the solution of optimality cannot give the best subset of sample, further regular dropping is performed one at a time. The procedure will search the possible dropping successively until the optimal solution is obtained.

When the search procedure has been performed for forward process, then the search procedure will automatically trace back to a searching for branching the subset of samples. In this case, an updating procedure, also one at a time, is performed. The backward process is finished when the search arrives at the first root of the branch. The next search, the second root of the branch, is performed for forward and backward process. The \textit{Branch and Bound} procedure will finish when all combinations have been searched. By the end of the search process, the optimal values with efficient selection are stored by computer and then the appropriate selection which meets the cost efficiency drilling program is justified.

The construction of the above procedure is described in the following steps:

Step 1. Set the initial set of points (drillholes) $S=\{s_1, s_2, \ldots, s_m\}$ and $M$ as the empty set or $M=0$. 
Step 2. Check $S$ having $m$ elements, the points starting from the current point have been searched, and at least one combination (selection) from $M$ is the best set of the selection.

If $S$ with $m$ elements is correct, add the selection to $M$. If there exists the best selection which produces the smallest estimation (kriging) variance, then go to Step 3. Otherwise go to Step 4.

Step 3. Perform backward process along the branches while updating the current new estimation (kriging) variance, and go to Step 2.

Step 4. Perform forward process along the branches which have not been searched, and update the new estimation (kriging) variance, then go to Step 5.

Step 5. If the current point of the subset $[s_1, s_2, \ldots, s_{m-1}] = S - s_m$, then the search process stops. Otherwise go to Step 2.

2.4 THE BLOCK INVERSE TECHNIQUE BY PARTITION

The block inverse technique by partition is described in the following.

Consider a matrix $P$ of $n \times n$ has the form:

$$P = \begin{pmatrix} AB \\ CD \end{pmatrix}$$

(2.4)

where $A$ and $D$ are matrices of order $i$ and $n-i$, respectively, and $B$ is row of $i \times (n-i)$ and $C$ is row of $(n-i) \times i$. The inverse of $A$ is known as $A^{-1}$. Using
the small inverse matrix $A^{-1}$, the matrix $P$ can be efficiently inverted by means of dividing the matrix into block form with partitions of the same order as equation (2.4), as:

$$P^{-1} = \begin{pmatrix} X & y \\ y^T & S \end{pmatrix}$$  \hspace{1cm} (2.5)$$

The relation between (2.4) and (2.5) forms the linear equation as shown below:

$$AX + By^T = I_i$$
$$Ay + BS = 0$$
$$CX + Dy^T = 0$$
$$Cy + DS = I_{n-i}$$  \hspace{1cm} (2.6)$$

where $I_i$ and $I_{n-i}$ are the unit matrix of $i$ and of $n-i$, respectively. From the equation (2.6), the small matrix inverses of $X, y, y^T$ and $S$ can be simply calculated. An application of this technique will be given in Section 2.5.

This inversion technique is useful to speed up the inversion process of kriging matrix if the kriging equation is to be solved a second time. In this thesis, this technique is utilized to speed up the computation of the modified kriging variance so as to avoid the repetition of the computation of the whole kriging process.

The use of the technique can be given as follows:

Consider the conventional kriging matrix, $G$, represents a dimension of matrix $(n+1) \times (n+1)$ with a diagonal of matrix $\gamma_{nn}=0$. This kriging matrix is rearranged by rows and columns. The procedures are given.
The conventional covariance matrix is normally written as

\[
G = \begin{pmatrix}
\gamma_1 & \cdots & \gamma_{1n} & 1 \\
          & \cdots & \cdots & \cdots \\
\gamma_{n1} & \cdots & \gamma_{nn} & 1 \\
1 & \cdots & 1 & 0
\end{pmatrix}
\]  

(2.7)

In order that the technique can be applicable, this covariance matrix must be rearranged in a form of matrix, \( G_{re} \), as shown:

\[
G_{re} = \begin{pmatrix}
0 & 1 & \cdots & 1 \\
1 & \gamma_{11} & \cdots & \gamma_{1n} \\
          & \cdots & \cdots & \cdots \\
1 & \gamma_{n1} & \cdots & \gamma_{nn}
\end{pmatrix}
\]  

(2.8)
The complete re-arrangement of the kriging system for the covariance of samples 

\((n+1) \times (n+1)\) is defined as

\[
\begin{pmatrix}
0 & 1 & \ldots & 1 \\
1 & \gamma_{11} & \ldots & \gamma_{1n} \\
\vdots & \vdots & \ddots & \vdots \\
1 & \gamma_{n1} & \ldots & \gamma_{nn}
\end{pmatrix}
\begin{pmatrix}
\mu \\
\lambda_1 \\
\vdots \\
\lambda_n
\end{pmatrix}
= 
\begin{pmatrix}
1 \\
\gamma_{v1} \\
\vdots \\
\gamma_{vn}
\end{pmatrix}
\]

(2.9)

The above matrix is, \(G_{re} \lambda = \gamma\), where \(G_{re}\) is a \((n+1) \times (n+1)\), \(\lambda\) and \(\gamma\) are \((n)\) dimensional kriging vectors and diagonal elements of \(G_{re}\) are zero. Assume that the first solution kriging system is \(\lambda_0\). If one additional sample point, \(x_{n+1}\), is updated, then the new updating kriging matrix would be
By simplifying the above kriging system, the kriging vectors $g_{\text{new}}$, $g_{\text{new}}^T$, $L_1$, $L_2$, $\gamma_1$, $\gamma_2$, and diagonal matrix, $D$ are derived to give:

$$
\begin{align*}
\text{g}_{\text{new}} &= \begin{pmatrix}
\gamma_{11} \\
\gamma_{12} \\
\vdots \\
\gamma_{nn}
\end{pmatrix} \\
\text{g}_{\text{new}}^T &= \begin{pmatrix}
1 & \gamma_{11} & \cdots & \gamma_{1n} \\
1 & \gamma_{21} & \cdots & \gamma_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
1 & \gamma_{nn} & \cdots & \gamma_{nn}
\end{pmatrix}
\end{align*}
$$

(2.10)
The simplified new kriging system would be:

\[
\begin{pmatrix}
G_{re} & g_{\text{new}} \\
g_{\text{new}}^T & D
\end{pmatrix}
\begin{pmatrix}
\lambda \\
\lambda_{\text{new}}
\end{pmatrix} =
\begin{pmatrix}
\gamma \\
\gamma_{\text{new}}
\end{pmatrix},
\]

(2.12)

The new kriging matrix in equation (2.10) then is partitioned into the block forms, and each block is assigned to unknown variables \(X, y, y^T, \text{ and } S\), respectively. As these unknown variables must be calculated to obtain the inverse of the new kriging matrix, the partitioned blocks containing these variables are multiplied by the new kriging matrix in order to obtain the equality of unit matrices of these unknown variables, as shown:

\[
K^{-1} = \begin{pmatrix}
X & y \\
y^T & S
\end{pmatrix},
\]

(2.13)

\[
\begin{pmatrix}
G_{re} & g_{\text{new}} \\
g_{\text{new}}^T & D
\end{pmatrix}
\begin{pmatrix}
X & y \\
y^T & S
\end{pmatrix} =
\begin{pmatrix}
I & 0 \\
0 & I
\end{pmatrix},
\]

The multiplication of the above element matrices will give the equality of unit matrices which implies each component matrix is calculated by unit matrices (\(I\) is the unit matrix), thus:
From the first equation above the following can be obtained:

\[ X = (G_{\text{re}})^{-1} - (G_{\text{re}})^{-1} g_{\text{new}} y^T, \]  

(2.15)

the second equation can be solved by computing

\[ y = - (G_{\text{re}})^{-1} g_{\text{new}} S, \]  

(2.16)

the third equation also calculates

\[ y^T = - g_{\text{new}}^T X / D, \]  

(2.17)

and the fourth equation can be simplified

\[ S = (D^{-1} - D^{-1} g^T y), \]  

(2.18)
by substituting variable $y$, in equation (2.18) using equation (2.16), the fourth equation can also be rewritten

$$S = (D - g_{\text{new}}^T (G_{\text{re}})^{-1} g_{\text{new}})$$

by substituting the variable, $X$, in equation (2.17) using equation (2.15), the above equation (2.17) would be

$$y^T = -S(G_{\text{re}})^{-1} g_{\text{new}}$$

hence, the following relations for each element matrix of the kriging inverse $K^{-1}$, are established as

$$S = (D - g_{\text{new}}^T (G_{\text{re}})^{-1} g_{\text{new}})$$

$$y = -S(G_{\text{re}})^{-1} g_{\text{new}}$$

$$y^T = -S g_{\text{new}}^T (G_{\text{re}})^{-1}$$

$$X = (G_{\text{re}})^{-1} - (G_{\text{re}}^{-1}) g_{\text{new}} y^T$$

For simplicity, in the first equation of (2.21) the variable $D=0$, because the diagonal kriging matrix is always zero, and also the constructed vector $y=y^T$, due to the lower and upper part of the diagonal matrix having the same covariance values.
The rewriting of the equations of (2.21) would give:

\[
S = - \frac{1}{g_{\text{new}}^T (G_{\text{re}})^{-1} g_{\text{new}}}
\]

\[
y = - G_{\text{re}}^{-1} g_{\text{new}} S
\]

\[
X = G_{\text{re}}^{-1} \frac{1}{S} y y^T,
\]

The linear equation of (2.12) before inverting forms the following:

\[
G_{\text{re}} \lambda + g_{\text{new}} \lambda_{\text{new}} = \gamma
\]

\[
g^T \lambda + D \lambda_{\text{new}} = \gamma_{\text{new}},
\]

hence, the new lambda coefficient, \( \lambda_{\text{new}} \), would be:

\[
\lambda_{\text{new}} = (\gamma_{\text{new}} - g_{\text{new}}^T \lambda)/D
\]

\[
\lambda = (G_{\text{re}})^{-1} (\gamma_1 - g_{\text{new}} \lambda_{\text{new}}),
\]

because \( S = D^{-1} \) and \( \lambda_0 = (G_{\text{re}})^{-1} \gamma_{\text{new}} \) then the equations of (2.24) and (2.25) would be

\[
\lambda_{\text{new}} = (\gamma_{\text{new}} - g_{\text{new}}^T \lambda_0) S
\]

\[
\lambda = \lambda_0 - G_{\text{re}}^{-1} g_{\text{new}} \lambda_{\text{new}},
\]
where $\lambda_0$ is the old coefficient of the kriging system. And finally, the new updating kriging variance is solved as

$$V_{i_1...i_{n+1}}^{2} = V_{i_1...i_n} + \frac{\lambda_{\text{new}}^2}{s}$$

or

$$V_{\text{new}} = V_{\text{old}} + (\lambda_{\text{new}})^2 / S$$

The above new kriging matrix variance is the modification of the old kriging matrix, $V_{i_1...i_n} = V_{\text{old}}$.

2.5. THE PROCEDURE OF KRYING VARIANCE MODIFICATION

**Updating of a sample - Procedure 1**

As derived above, the solution (2.28) is that one new sample (drillhole) is updated within the existing drillholes. Then the whole estimation procedure can be processed by the following simple steps:

Step 1. Calculate the vector matrix $L = (G_{\text{re}})^{-1} g_{\text{new}}$

Step 2. Calculate the small matrix block, $X, y = y^T$ and $S$
Step 3. Compute the weights $\lambda$ and $\lambda_{new}$

Step 4. Modify the old estimation variance by updating the new estimation variance sample.

$$V_{new} = V_{old} + (\lambda_{new})^2 / S$$

**Dropping of a sample - Procedure 2**

The dropping of one sample from an existing sample set is done by assuming that this sample has been updated with one point, $x_{n+1}$ (this procedure recalculates the old estimation variance), then by following the steps given below:

Step 1. Calculate the matrix vector $m = -1 / S \cdot (y)$

Step 2. Calculate the matrix $G_{re}^{-1} = X - 1 / S \cdot (y y^T)$

Step 3. Calculate the weight $\lambda_o = \lambda + m \cdot \lambda_{new}$

Step 4. Calculate the old estimation variance

$$V_{old} = V_{new} - (\lambda_{new})^2 / S$$
CHAPTER 3

\textit{OP}SAMP COMPUTER PROGRAM
CHAPTER 3

**OPSAMP COMPUTER PROGRAM**

An ore reserve is traditionally estimated from available samples such as core samples, chip samples and churn samples. One of the factors which influence the reliability of such an estimate is the number of available samples. In practice, most sampling designs involve identifying the locations of potential sampling sites which meet the budget. The reliability of the estimate is usually a secondary factor.

The reliability of an estimate of ore reserve does not depend on the assay values, but rather on (i) the underlying variogram, (ii) the block size, and (iii) the relative locations of samples with respect to the block. This suggests that it is possible to assess the efficiency of a sampling program prior to the actual drilling provided that the underlying variogram can be estimated. A computer program **OPSAMP** was developed in this thesis to estimate the reliability of either a local or global estimate using geostatistical concepts of estimation (kriging) variance. Using **OPSAMP**, it is possible to evaluate various sampling strategies prior to the actual drilling.
3.1 AN OVERVIEW OF PROGRAMME OPSAMP

The computer program OPSAMP discussed in this thesis consists of several modules, each subprogram performing a specific task. The program first reads the run parameters and drillhole information. It then generates an estimation covariance matrix and finally computes either estimation variance (global estimation) or kriging variance (local estimation).

In the case of global estimation, the computer program uses the entire (existing and potential) suites of samples. The impacts of the additional samples are then evaluated by calling the module TREE of the OPSAMP system. The TREE subprogram serves two functions. First, the module identifies the 'best' subset of potential samples which meet a specified reliability. If the geologist proposes ten potential drill-sites and only eight of the ten sites are required for one reason or the other, TREE module can be used to identify the best eight sites from the predetermined ten sites. The second function of the subprogram is to reduce the repetitive computation of estimation variance while searching for the 'best' subsets of the minimum estimation variance. The search procedure starts by dropping a sample one at time until the required reliability is reached. For an example, if there are $n$ additional samples, the program first determines the reliability for the entire $n$ potential samples. The system then determines the next 'best' $(n-1)$ subset of additional samples. The process is repeated for $(n-2), (n-3), \ldots, 2$ subsets of the additional sample. Figure 3.1 shows a simplified flowchart of the OPSAMP system.
Fig 3.1 Flow diagram of the *OPSAMP* program
3.2 OPSAMP MODULES

The OPSAMP system consists of the MAIN program and several subroutines, the listings of which are provided in Appendix III. OPSAMP requires three data types. These are the run parameters, the drillhole data file, and the area definition data file. The structures of these files are described:

1) Run Parameters - The run parameters include project definition, the names of ASCII data files to be used, variogram parameters and program constants. The run parameters are stored in a NAMELIST file which may be accessed by using FORTRAN intrinsic NAMELIST statement. The command is used to read non-formatted input statement in a list of format specifiers. Table 3.1 shows a typical NAMELIST file. The first record is the NAMELIST program name prefixed with the character '&'. All the NAMELIST variables must conform to FORTRAN naming rules. The '&END' statement signifies the end of record. The data in Table 3.1 are created using a text editor and saved in a file, e.g., SVAR.LST to be accessed by the OPSAMP system.

Table 3.1 Example of NAMELIST file

<table>
<thead>
<tr>
<th>&amp;SVR</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE='COAL PROJECT'</td>
</tr>
<tr>
<td>FNAME='COAL.DAT'</td>
</tr>
<tr>
<td>BLOC_TYPE='RECTAN'</td>
</tr>
<tr>
<td>POLY_FILE='AREA.DAT'</td>
</tr>
<tr>
<td>LIST='YES'</td>
</tr>
<tr>
<td>SILL=0.36</td>
</tr>
<tr>
<td>Co=0.036</td>
</tr>
<tr>
<td>C=0.33</td>
</tr>
<tr>
<td>&amp;END</td>
</tr>
</tbody>
</table>

All the variables assigned in a *NAMELIST* file must be declared in the application program. The variables in Table 3.1 are declared in *OPSAMP* program as below:

```
NAMELIST/SVR/TITLE,FNAME,BLOCK_TYPE,POLY_FIL,LIST, SILL,CO,C
CHARACTER TITLE*20,FNAME*20,BLOCK_TYPE*6,POLY_FIL*12,LIST*3
OPEN(UNIT=IPT,FILE='SVAR.LST',STATUS='UNKNOWN',ACCESS='SEQUENTIAL')
READ(IPT,SVAR)
```

A *NAMELIST* command followed by a *NAMELIST* name 'SVR' is used to assign values to variables `TITLE, FNAME, BLOCK_TYPE, POLY_FILE, LIST`, `CO` and `C`. The number of these run parameters declared must be the same as those which are in the *NAMELIST* file. The second line is *CHARACTER* declaration statement, while the last two statements are *OPEN* and *READ* commands, respectively. The file status in the *OPEN* statement is unknown status, since the file created is a temporary one. The *READ* command reads all the assigned values in the file *SVAR.LST* specified by a namelist name 'SVR'.

2) Drill hole data - Both the existing and potential drillhole information are stored in a formatted file whose name is assigned in the run parameter file (*COAL.DAT* as example). The structure of a drillhole data file is shown below:

```
COAL PROJECT
(a10,7f10.2)
4
1=Ash  2=Volatile  3=Sulfur  4=Thickness
Hole1  234000  123000  100  0.2  0.5  1.2  0.0
Hole2  235000  120000  150  0.3  0.4  1.3  0.1
Hole3  236000  122000  125  0.2  0.2  1.5  0.2
```
The first record of a drillhole file is the identification of the project followed by the format to be used to read the drillhole information. The third record defines the number of parameters (assay values) per hole with their name given in the fourth record. Subsequent records (record 5 and the rest) are respective drillhole identification, northern coordinate, eastern coordinate, elevation and parameters (assay values).

The program OPSAMP assumes that potential drillholes have zero assay values whereas the existing drillholes have non-zero assay values.

3) Area Definition File. The program OPSAMP also requires a file containing the coordinates of points (vertices) defining the area to be estimated (Fig 3.2). Two options are provided in OPSAMP for selecting the form of the area. If the variable NTYPE=1, a file containing the coordinates of the nodes of an irregular area is required. The name of such a file must be defined in the run parameter file. If NTYPE=2, the program generates regular blocks whose size depends upon input values of the variables WIDX,WIDY, NDX,and NDY (Fig 3.3).

Fig.3.2 Definition of an irregular mining area
3.3 THE USER INTERFACE SYSTEM

The common data input modes used in the mining industry include the use of an editor to create a text file which can then be read by the application program. In some applications numerous questions may be answered by the user during the data input phase. In this case, it is impossible to modify the input data after it has been read by the application programs, unless the user exits from the program and starts all over again. Both the above two data input modes are tedious, time consuming and inflexible in some applications. Unfortunately, program OPSAMP requires much data. In order to ensure some form of flexibility with program OPSAMP during the data input phase, a screen handling system (PCHAND) was developed. The system makes use of the escape characters of ANSI sequence codes with standard FORTRAN statements.

The PCHAND system performs a multi-viewing access of input data entries which can be validated by the user. The viewing process itself can transfer default values from files, if available, and also the data entered by the user. Such an approach can speed data input phase and also provides the flexibility to evaluate various drilling scenarios.
3.5 THE PCHAND AND THE ANSI ESCAPE SEQUENCES

The personal computer MS-DOS environment has several unique systems, one of which is the ANSI Escape Sequence. The ANSI is a set standard character based on the American Numerical Standard Institute. The ANSI Escape Sequence is a series of characters which begin with an escape character (ESC) and has a capability of defining functions to Disk Operating System (DOS) which in turn can affect cursor movement, change screen modes, reassign keys to display colour setting and to erase functions.

All ANSI codes are recognised through a console devised which is configured by a system called ANSI.SYS. This ANSI.SYS is a DOS service which allows the user to create a simple keyboard program by redefining function keys. In other words, ANSI.SYS is a program developed to recognise the redefined keys which is driven onto screen or console device. The following examples illustrate the basic function of the ANSI Escape Sequence.

1) Cursor control

a. Move Down

Syntax \[ ESC/nA \]

Remark To move the cursor down specified by \( n \) lines

Example \[ ESC/3A \] moves the cursor 3 lines down without changing the current column
b. Move Up

Syntax  \textit{ESC}[n B

Remark  To move the cursor up by \textit{n} lines

Example  \textit{ESC}[2;3 B moves the cursor 2 lines up on column 3

c. Move Right

Syntax  \textit{ESC}[n C

Remark  To move the cursor to the right by \textit{n} lines

Example  \textit{ESC}[2;3 C moves the cursor by 3 columns to the right on the current row 2

s. Move Left

Syntax  \textit{ESC}[n D

Remark  To move the cursor to the left by \textit{n} lines

Example  \textit{ESC}[2;3 D moves the cursor by 3 columns to the left on row 2

e. Locate position

Syntax  \textit{ESC}[r;c H

Remark  To display the cursor onto monitor at row, \textit{r}, and column, \textit{c}.

Example   \textit{ESC}[2;3 H displays the cursor at a position defined by row 2 and column 3
f. Horizontal and vertical position

Syntax \[ \text{ESC}\[r;cf \]

Remark To locate the cursor onto monitor at row, \( r \), and column, \( c \)

Example \[ \text{ESC}\[2;5f \text{ to ESC}\[2;7f \] moves the cursor from column 5 to column 7 on the same row 2.

\[ \text{ESC}\[2;5f \text{ to ESC}\[4;5f \] moves the cursor from row 2 to row 4 down on the same column 5

2) Screen Mode

Syntax \[ \text{ESC} [=nh \]

Remark To set screen mode display specified to \( n \) by \( n \) pixel size

Example \[ \text{ESC} [=0h \text{ sets monitor to 40 by 25 with mono text mode}

\[ \text{ESC} [=1h \text{ sets monitor to 40 by 25 with colour text mode}

3) Keyboard re-assigment

Syntax \[ \text{ESC} [ nl;n2p \]

Remark To reassign ASCII code \( nl \) to ASCII code \( n2 \)

Example \[ \text{ESC} [ 81;65p \text{ reassign the Q to the A (when the key A is intercepted, this intercepted key is mapped to the key Q)}

The summary of the above functions used is shown in Table 3.2 through Table 3.4.
### Table 3.2 Display colour setting

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Attribute</th>
<th>Background Color</th>
<th>Foreground Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>Esc[a;b;fm</td>
<td>0 = off</td>
<td>30 = black</td>
<td>40 = black</td>
</tr>
<tr>
<td></td>
<td>1 = high</td>
<td>31 = red</td>
<td>41 = red</td>
</tr>
<tr>
<td></td>
<td>2 = normal</td>
<td>32 = green</td>
<td>42 = green</td>
</tr>
<tr>
<td></td>
<td></td>
<td>33 = yellow</td>
<td>43 = yellow</td>
</tr>
<tr>
<td></td>
<td>4 = underline*</td>
<td>34 = blue</td>
<td>44 = blue</td>
</tr>
<tr>
<td></td>
<td>5 = blink</td>
<td>35 = mangenta</td>
<td>45 = mangenta</td>
</tr>
<tr>
<td></td>
<td></td>
<td>36 = cyan</td>
<td>46 = cyan</td>
</tr>
<tr>
<td></td>
<td>7 = reverse video</td>
<td>37 = white</td>
<td>47 = white</td>
</tr>
<tr>
<td></td>
<td>8 = concealed**</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* monochrome
** foreground set to background

### Table 3.3 Cursor control

<table>
<thead>
<tr>
<th>Cursor Function</th>
<th>Syntax</th>
<th>Default Parameters</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move Down</td>
<td>Esc[nA</td>
<td>n=1</td>
<td>Esc[3A (3 rows)</td>
</tr>
<tr>
<td>Move Up</td>
<td>Esc[nB</td>
<td>n=1</td>
<td>Esc[3B (3 rows)</td>
</tr>
<tr>
<td>Move Right</td>
<td>Esc[nC</td>
<td>n=1</td>
<td>Esc[2C (2 columns)</td>
</tr>
<tr>
<td>Move Left</td>
<td>Esc[nD</td>
<td>n=1</td>
<td>Esc[2D (2 columns)</td>
</tr>
<tr>
<td>Locate position</td>
<td>Esc[c;r H</td>
<td>c=1;r=1</td>
<td>Esc[2;7 H</td>
</tr>
<tr>
<td>Locate position</td>
<td>Esc[c;r f</td>
<td>c=1;r=1</td>
<td>Esc[2;7 f</td>
</tr>
<tr>
<td>Save position</td>
<td>Esc[S</td>
<td>---</td>
<td>Esc[S</td>
</tr>
<tr>
<td>Restore position</td>
<td>Esc[U</td>
<td>---</td>
<td>Esc[U</td>
</tr>
</tbody>
</table>
Table 3.4 Display mode

<table>
<thead>
<tr>
<th>Mode Function</th>
<th>Syntax</th>
<th>Example</th>
<th>Set</th>
<th>Parameter definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set mode</td>
<td>Esc[=nh</td>
<td>Esc[2h</td>
<td>n=0</td>
<td>40 by 25 mono text</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>n=1</td>
<td>40 by 25 color text</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>n=3</td>
<td>80 by 25 color text</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>n=4</td>
<td>320 by 200 mono graphic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The above ANSI escape sequences may be utilised to display and to accept data in two ways. ANSI codes or syntaxes may form part of the FORTRAN statements. These syntaxes can be written either at the beginning of MAIN program or in a subprogram. The following statements can be used to display OPSAMP logo shown in Figure 3.4.

PROGRAM LOGO
CHARACTER*1 A
OPEN(5, FILE='CON', ACCESS='TRANSPARENT')
PRINT,'ESC[9;21f','
PRINT,'ESC[10;21f','
PRINT,'ESC[11,21f','
PRINT,'ESC[12,21f','
PRINT,'ESC[13,21f','
PRINT,'ESC[14,21f','
STOP
END

Press any key to continue'
The first statement is the optional symbolic program name. The next statement is the character declaration of variable A. The OPEN statement on the third line is required to access console device defined by unit 5. In the fourth line, PRINT command is used to display part of the logo. The OPSAMP logo is obtained using ASCII characters which are obtained by pressing the Alt-key followed by the combinations of 220, 221 and 223. The top part of the logo is printed using ANSI syntax, 'Esc[9;2If' which locate the cursor at column 21 on row 9 prior to printing. The next PRINT commands are used to display a complete OPSAMP logo by keeping the same column and updating the row position to one line at a time. The STOP and END statements are the termination of the instruction and the end of the menu display program, respectively.

In order to ensure some form of flexibility with the use of ANSI escape sequences, the generalised FORTRAN routine PCHAND is developed. The routine was designed in such a way that it can be adapted to any application program with limited modifications. A description of the PCHAND system was given above and further details follow.
3.6 THE **PCHAND** SYSTEM

**PCHAND** is a personal computer based screen handling system specially developed to speed the data entry state of the **OPSAMP** system. **PCHAND** is a **FORTRAN** program generated to drive the user interface menus. The system consists of five modules which are used to update, to edit, to view, to delete or to save input data. These sub-modules are **CPOS, CTEXT, KEYSTROKE, GETDAT** and **WRTDAT** as shown in Figure 3.5.

![Fig 3.5 PCHAND system](image)

**CPOS and CTEXT Subprograms**

The **OPSAMP** logo shown in Fig 3.4 may also be displayed using **CPOS** and **CTEXT** modules of the **PCHAND** subsystem.

The **CPOS** is a subprogram used to allocate cursor position at row **JROW** and column **JCOLUM**.

The syntax is

\[ CPOS(JROW,JCOLUMN) \]

where \( JROW \) = row number on the console, and \( JCOLUMN \) = column number on the console.
An example is

```
PROGRAM CSRPOS
      ........
      ........
      CALL CPOS(2,3)
      STOP
      END
```

The *CTEXT* module supports the system by displaying information. This can be called by *PCHAND* using the syntax.

The syntax is

\[
CTEXT(I,J,CB,CF,LENCHAR)
\]

where
- \(I\) = cursor row position text
- \(J\) = cursor column position text
- \(CB\) = setting background colour
- \(CF\) = setting foreground colour
- \(LENCHAR\) = the length characters to be displayed

The characters displayed in Fig 3.6 (which are the same as in Fig 3.4 derived from a less sophisticated approach) are assigned to array *CHARI* the elements of which are stored in a namelist file shown in Fig 3.7. In this example, the string of characters will be displayed from cursor position defined by row and column variables \(JROW(1)=9\) and \(JCOL(1)=21\), respectively.
Fig 3.6. Displaying an *OPSAMP* logo
(from more sophisticated analysis but unchanged display from Fig 3.4)

&LGO
JROW(1)=9
JCOL(1)=21
NDIGT(1)=45
ITER=5
ISTP=1
CHAR1(1)=
CHAR1(2)=
CHAR1(3)=
CHAR1(4)=
CHAR1(5)=
CHAR1(6)=
CTP(1)=C
CTP(2)=C
CTP(3)=C
CTP(4)=C
CTP(5)=C
&END

Fig 3.7. The namelist file containing the *OPSAMP* logo

The main program first reads the contents of the namelist file *LOGO.LGO* (Fig 3.7) and then invokes *CPOS* module to locate the cursor at respective locations. The *CPOS* program listing is:
SUBROUTINE CPOS(JROW,JCOLUMN)
CHARACTER*10 POS
IF(JROW.LE.9) THEN
WRITE(POS,'(A2,11,A1,I2,A1)') 'Esc[1I;J,f'
PRINT, POS(1:7)
ELSE
   WRITE(POS,'(A2,I2,A1,I2,A1)') 'Esc[1I;J,f'
PRINT, POS(1:8)
ENDIF
RETURN
END

The variable POS is set length (10) of the characters required in the ANSI syntaxes, 'Esc[1I;J,f'. The third line redefines the ANSI syntaxes, 'Esc[1I;J,f' (depending on row I and column J) and assigns this syntax to POS using the format (A4,I1,A1,I1,A1)'. The variable POS is used to move the cursor to the respective locations depending on the values of the arguments in the calling statement.

The CTEXT listing program is:

SUBROUTINE CTEXT(I,J,CB,CF,LENCHAR)
DIMENSION CHAR1(50)
COMMON/PAR/CHAR1
CHARACTER*5 CB,CF
CHARACTER POS*8,CHAR1*72
WRITE(POS,'(A4,I1,A1,I1,A1)') 'Esc[1I;J,f'
PRINT, CB,CF,POS(1:8),CHAR1(1,LENCHAR),'Esc[0m'
RETURN
END
The arguments \( I \) and \( J \) are the cursor's row and column positions. The colour set modes are given by \( CB \) (background colour) and \( CF \) (foreground colour). The last argument of \( CTEXT \), \( LENCHAR \), is the character length of the text to be displayed. The maximum number of variable displays is 50 and is assigned to variable \( CHAR1 \) through \( COMMON \) statement. The next line assigns the ANSI syntaxes to variable \( POS \) as described above. The \( PRINT \) statement displays characters in variable \( CHAR1 \) with the appropriate background and foreground colour of the display. In order to give the complete display of the logo, the ASCII characters \( CHAR1(1),...,CHAR(5) \) are printed one at a time.

The next display of the \( OPSAMP \) system is the main menu shown in Fig 3.8

![Fig. 3.8. The main menu display](image-url)
Namelist file *OPSAMP.MNU* (Fig 3.9) contains the data used to display the menu. After reading, the file system displays the text contained in array *CHARI* at the position of the screen defined by *JROW(1)* and *JCOL(1)*. Subprogram *CPOS* is used to locate the cursor initially at *JROW(1)=11* and *JCOL(1)=22* and *CTEXT* is used to display the menu options.

```
&MNU
HLINE='Opsamp main Menu'
ITER=3
ISTP=1
NDIGT(1)=15
JROW(1)=11
JCOL(1)=23
CTP(1)='C'
CTP(2)='C'
CTP(3)='C'
CHAR1(1)=' [1] Optimal sampling design'
CHAR1(2)=' [2] Global and local estimation'
CHAR1(3)=' [3] Quit'
DINF(1) = 'Optimal sampling system'
DINF(2) = 'Estimation (kriging) variance computation'
DINF(3) = 'Terminate OPSAMP system'
&END
```

Fig 3.9 Setting the namelist file of the *OPSAMP* main menu

The module *CTEXT* is called three times (*ITER=3*) in order to complete a display of the main *OPSAMP* program options. The menu in Fig 3.8 appears with a red cursor highlighted on the first option of the menu at *JROW(1)=11* and *JCOL(1)=23*. The user can then move the cursor key to select the appropriate task. To activate the cursor movement, the program invokes the subroutine *KEYSTROKE* to activate the cursor movement.
The syntax is

\[ KEYSTROKE(I, J, NCOD, IP) \]

where

- \( I \) = row number on the cursor position,
- \( J \) = column number on the console,
- \( NCOD \) = code number used to specify the keyboard function,
  - 1 = up
  - 2 = down
  - 3 = right
  - 4 = left
  - 5 = save
  - 6 = quit, and
- \( IP \) = identifier of character position

The \textit{KEYSTROKE} listing program is:

```c
C----------Read any symbol on keyboard and convert to ASCII code
C         READ(40,'(A)') KEY
C         STRNG=ICHAR(KEY)
C          .......
C----------If a key code satisfied with one of the conditions below
C
C          IF(STRNG.EQ.72.) THEN
C          ........Move cursor UP
C          NCOD=1
C          ELSEIF(STRNG.EQ.80.) THEN
C          ........Move cursor DOWN
C          NCOD=2
C          ELSEIF(STRNG.EQ.77.) THEN
C          ........Move cursor to RIGHT
C          NCOD=3
C          ELSEIF(STRNG.EQ.75.) THEN
C          ........Move cursor to LEFT
C          NCOD=4
C          ELSEIF(STRNG.EQ.84.) THEN
```
The first line of the program is to read any key selected by the user in character format type. Once this key has been input, the second line of the program redefines the ASCII symbol of the keystroke using FORTRAN intrinsic ICHAR to be assigned to variable STRNG. The next lines in the above program are the FORTRAN statement which is used to pass the STRNG to one of the conditions given, e.g., STRNG=72 the cursor up, then encode this to NCOD=1. The program therefore encodes the keyboard character selected by the user which is used to call the module CPOS to move the cursor. A red flash is shown to identify the current position of the cursor in that menu. The procedure of flashing this text menu is described later.

From the main menu of OPSAMP, the program allows the user to select an option (Fig 3.10). If option 1 is selected, the sub-bar menu appears on the side of the screen which offers a number of tasks to be performed. These include EDIT, SAVE, VIEW, MSCAT, MDCLUS, MSVAR, MSKRIG, RUN, MENU, HELP, and EXIT.
Fig 3.10 Displaying the $OPSAMP$ sub-bar menu

If the user selects the option 1 from the submenu, the system requests the filename of the drillhole data, e.g., $COAL.DAT$ (Fig 3.11). This is followed by the next submenu shown in Fig 3.12. If option 1 is selected from Figure 3.12, the drillhole data with the sulfur values are displayed in the next window shown in Fig 3.13.
Fig 3.11 The first input phase of the editing file

Fig 3.12 The second input phase of the editing file
Fig 3.13 The third input phase of the editing file

The modules CPOS and CTEXT are constantly used to display all those menus depicted in Fig 3.10 through 3.13. Fig 3.13 shows the cursor flashing in the first row of column of hole identification. Displaying the data requires a namelist file which contains a template of five columns. This file is termed WINDW2.WDW in the PCHAND subsystem. To draw the template requires the CTEXT module again. Once the template is displayed, the module WRTDAT is called from the main program to echo ten sets of drillhole data. A green-red cursor is displayed on the left top of the first column template to highlight the data.

Fig 3.14 is a namelist file (FORMAT.STR) which contains the variables used to display the editing window.
&FRM
ITER=10
STP=1
JROW(1)=11
JCOL(1)=18  NDIGT(1)=7  CTP(1)=C'
JCOL(2)=26  NDIGT(2)=7  CTP(2)=N'
JCOL(3)=35  NDIGT(3)=7  CTP(3)=N'
JCOL(4)=44  NDIGT(4)=7  CTP(4)=N'
JCOL(5)=53  NDIGT(5)=7  CTP(5)=N'
&END

Fig 3.14 The \textit{FORMAT.STR} file

The module \texttt{WRTDAT} actually performs a data display which is specially devised using the \textit{ANSI} syntax, \texttt{'ESC['}, \texttt{\textit{I}},',\texttt{;'},\texttt{\textit{J}},',\texttt{;'},\texttt{f}'. The program listing is given below.

The syntax is

\texttt{WRTDAT(I,J,CB,CF,DLEN,TVAL)}

where $I=$ Row number on the console,

$J=$ Column number on the console,

$CB=$ Setting background colour,

$CF=$ Setting foreground colour,

$DLEN=$ Length of character, and

$TVAL=$ Data used to display.
The WRTDAT listing program is

```fortran
SUBROUTINE WRTDAT(I,J,CLRB,CLRF,DLEN,TVAL)
COMMON/PAR/FLSH
CHARACTER PST*73,FRM*10,FLSH*25

C------Set a flashed box with a length (DLEN)
C
FRM='(F10.2)'
PST(1:DLEN)=FLSH

C------Set data in a formatted form
C 'FRM' refers to decimal digit set
C 'PST' is a variable contained the character flashing for highlighting
C data

WRITE(PST,FRM) TVAL
WRITE(POS,'(A2,Il,Al,Il,Al)','ESC[',I,';',J,';','f
PRINT,CLRB,CLRF,POS(1:7),PST(L1:L2),'ESC[0m'
RETURN
END
```

The printing format FRM='(F10.2)' and the ASCII character to flash is assigned to FLSH variable. FLSH is transferred to PST(1:DLEN) where DLEN is the number of digit used. Any value contained in TVAL is assigned to PST while setting the position of the cursor in the ANSI syntax. For an example, let TVAL=1560000, provided in the second column of the template in Fig 3.13. In order to display this value using the WRTDAT, row and column has been specified by JROW(1)=11 and JCOL(2)=25 and the variable type is an array, e.g., CTP(2)=’N’ where N is numeric type. The next step is to assign TVAL to PST in a format F10.2 and specify the cursor position by the syntax, ‘Esc[11;25f’. Printing the POS(1:7) followed by PST(8:15) with background colour, CLRIB, and fore-ground colour, CLRIB, display 1560000.
The position of the cursor must first be determined by CPOS. To replace the value with 1550000, the position of each digit in PST(1:JR) must be determined by identifier JR starting from 1 to 7 in this case. While typing the new value, e.g., 1550000 on this old value (the value to be replaced), the identifier, JR, keeps on changing to every JR+1. If the new value is 7 characters in length, the identifier stops at JR=7. During the typing process, each value entered is stored in PST(1:JR), e.g., PST(1:7)=1550000. To pick this value, the return key is pressed to invoke the routine GETVAL. From the routine GETVAL, the value contained in PST(1:7) is defined using the intrinsic command INDEX to define an integer or a real value. This indexed variable is then assigned to KPOS. If KPOS=0, the format, FRM='(I)' is set for the integer value, and KPOS>0, the format, FRM='(F )' is set for the real value. In this example, KPOS=0 implies the value entered is an integer. To pick this value from the screen, the character length of the value is specified by either the format FRM(3:3)='(I1)' if the length, JR<10 or FRM(3:4) for JR>=10. The new value is then picked by reading PST(1:7) in the format FRM(1:3)='(II)' in this case, TVAL is used to pick the value which is set in the format FRM(1:3) and then assigned to GETVAL.

The syntax is

\[ \text{Function GETVAL( )} \]

The listing program is

```
COMMON JR
C--------Index a decimal point if one exists and assign to 'KPOS'
C
KPOS=INDEX(PST(1:JR), '.')
```
C       Initial set of variable to zero
C
GETVAL=0.0
IF(KPOS.EQ.0) THEN
  FRM='(I )'
C
C-------Get integer values from 'IVAL' and assign to 'GETVAL'
C
IF(KL.LT.10) WRITE(FRM(3:3), '(II)') JR
IF(KL.GE.10) WRITE(FRM(3:4),'(I2)') JR
READ PST(1:KL), FRM) IVAL
GETVAL=IVAL
ELSE
  FRM='(F )'
C
C-------Get non-integer values to 'IVAL' and assign to 'GETVAL'
C
IF(KL.LT.11) WRITE(FRM(3:5), '(II,"."11)') KPOS-1,KL-KPOS
IF(KL.GE.11) WRITE(FRM3:6),'(I2,"."I1)') KPOS-1,KL-KPOS
READ PST(1:KL), FRM) DVAL
GETVAL=DVAL
ENDIF

SUPPORTING ROUTINES

In addition to the above routines, some modules such as BGROUND, BOX, FRAME, HLIGHT and SCROLL are frequently used to enhance the performance of menu displays. The description of these modules is given below:

1) **BGROUND module.** This module is made to give the background colour of the system. The basic data ASCII characters are stored in the namelist file, BGROUND.DRW (Fig 3.15). These ASCII characters are obtained by pressing the Alt-key followed by the ASCII values of 221=\[\], 23=\^, and 24=\.|
The namelist file of $BGROUND.DRW$ containing the ASCII characters

To display these ASCII characters, the blue background colour, $CB='Esc[34m'$ and the yellow foreground colour, $CF='Esc43m'$ initially are set by $CTEXT$ module. An iteration of 23 is required to display fully background $OPSAMP$ screen. This means that the module $CTEXT$ is called 23 times so as to write those ASCII characters. The syntax and the list of program $BGROUND$ is given below:

The syntax is

$$BGROUND(LOOP)$$

where

$$LOOP=$$ number of iteration

The $BGROUND$ module listing program is

```plaintext
SUBROUTINE BGROUND(LOOP)
DIMENSION CHAR(20)
COMMON/PAR/PST,HLINE
COMMON/FLASH/PSH
NAMELIST/BG/CHAR
CHARACTER*80 PST,HLINE,CHAR
OPEN(55,FILE='BGROUND.DRW',STATUS='OLD',ACCESS='SEQUENTIAL)
READ(55,BG)
DO 10 I=1,LOOP
```

Fig 3.15 The namelist file of $BGROUND.DRW$ containing the ASCII characters
Chapter 3. OPSAMP computer program 3-30

IF(I,EQ.1) PST=CHAR(1)
IF(I.GT.1.AND.I.LT.23) PST=CHAR(2)
IF(EQ.23) PST=CHAR(3)
L=LEN(CHARNB(PST))
CALL CTEXT(I,1,'ESC[34m','ESC[43m,,1,L)
10 CONTINUE
RETURN
END

(2) BOX module. The box module just performs a box drawing around the menu (Fig. 3.8). This module requires the namelist file of BOX.DRW. The procedure of drawing a box is the same as mentioned in the BGROUND module.

\begin{center}
\begin{tabular}{l}
\texttt{&BG} \\
\texttt{CHAR(1)=} \\
\texttt{CHAR(2)=} \\
\texttt{CHAR(3)=} \\
\texttt{&END}
\end{tabular}
\end{center}

\textbf{Fig. 3.16 The namelist file of BOX.DRW containing ASCII characters.}

The syntax and the listing program is:

The syntax is

\texttt{BOX}

The BOX listing program is

\begin{verbatim}
SUBROUTINE BOX
COMMON/PAR/PST,HLINE
COMMON/FLASH/PSH
DIMENSION CHAR(5)
NAMELIST/BX/CHAR,KCOL,KROW,ITER,CB,CF
CHARACTER*80 PST,HLINE,CHAR
\end{verbatim}
Chapter 3. OPSAMP computer program 3-31

OPEN(56,FILE='BOX.DRW',STATUS='OLD',ACCESS='SEQUENTIAL')
READ(56,BX)
DO 10 I=1,ITER
  IF(I.EQ.1) PST=CHAR(1)
  IF(I.GT.1.AND.I.LT.ITER) PST=CHAR(2)
  IF(I.EQ.ITER) PST=CHAR(3)
  K=LEN(CHARNB(PST))
  CALL CTEXT(KROW+1,KCOL,CB,CF,1,K)
10 CONTINUE
RETURN
END

(3) FRAME module. This module is required to draw a box around the menu displayed as shown in Fig 3.12. While drawing a box, the title of menu is displayed on top of the box. The listing of the program can be seen in Appendix III.

The syntax is

\texttt{FRAME(JRW,JCL,CB,CF,LC,NDIGT,IB,ITER,ISTP)}

where

\texttt{JRW} = Row position,
\texttt{JCL} = Column position,
\texttt{CB} = Background colour,
\texttt{CF} = Foreground colour,
\texttt{LC} = Length of character,
\texttt{NDIGT} = Number of digit,
\texttt{IB} = Identifier of the position of the variable array,
\texttt{ITER} = Number of iteration, and
\texttt{ISTP} = Number of spacing.
To draw the displayed menu options onto the screen, the maximum length of the character text menu is required. For instance, before displaying the OPSAMP program menu, the program invokes the FRAME module in order to draw the box. The variables such as title of the header, HLINE, maximum line, MAXLIN, and the length of the header must be known. The HLINE can be accessed from the namelist file whereas the MAXLIN can be obtained by searching the number of digits used which is assigned to the variable, NDIGT(K), K=1, ITER. The length of the title of the headline menu contained in HLINE is measured using FORTRAN statement, LEN(CHARNB( )). Then MAXLIN assigned to LIN is subtracted by LHEAD and divided by two in order to find the mid position of the first character of the title, e.g., MID=(LIN-LHEAD)/2 if LHEAD<LIN. The values of HLIN, PSH, CUL, CUR, CDL, CDR, and VLIN (in data statement from this module) are then assigned to PST(1:LIN). The final process of this module is to call the CTEXT to print the line with the given background and foreground colour.

(4) HLIGHT module. This module performs the open access of the file, FLASH.FLS, containing the ASCII character, (Fig 3.17) These ASCII characters are obtained by pressing the Alt-key followed with the ASCII value of 221. Using this ASCII characters, each menu displayed by cursor movement can be highlighted as shown in Fig 3.8. The ASCII character CHAR1(1) is assigned to the variable PSH. Once this is done, the menu can be flashed. The procedure of flashing the menu at the current position of the cursor is explained in the SCROLL module.
Fig 3.17. The namelist file of \textit{FLASH.FLS}

The syntax and the listing of program are shown as follows:

The syntax is

\textit{HLIGHT}

The \textit{HLIGHT} listing program is

\begin{verbatim}
SUBROUTINE HLIGHT
COMMON/PAR/PST,HLINE
COMMON/FLASH/PSH
NAMELIST/FLS/CHAR
DIMENSION CHAR(5)
CHARACTER*80 PST,PSH,HLINE,CHAR
OPEN(57,FILE='FLASH.FLS',STATUS='OLD',ACCESS='SEQUENTIAL')
READ(57,FLS)
PST=CHAR(1)
PSH=CHAR(2)
RETURN
END
\end{verbatim}

(5) \textit{SCROLL Module}. The function of this module is to provide the menu displayed with the colour flashing cursor as well as to control the cursor movement. The module actually calls the \textit{HLIGHT} module, transferring the \textit{ASCII} character from the variable, \textit{PSH}, to \textit{PST}, rewriting the text menu option
and controlling the cursor movement. In this module, the \textit{CEXT} is functioned to display the \textit{ASCII} characters and to rewrite the text menu option. From this module, the subprogram \textit{HLIGHT} is first invoked in order to obtain the \textit{FLASH} character in \textit{ASCII} code. After this, the \textit{CTEXT} is functionally used to display the flash characters and to rewrite the text menu. By setting the red background colour, \textit{CB='Esc[41m'} and the white foreground colour, \textit{CF='Esc[37m'}, \textit{CTEXT} module displays all the variables with the appropriate colour.

The syntax is

\texttt{SCROLL(JRW,JCL)}

The listing of module \textit{SCROLL} is provided in Appendix III.

3.6 THE \textit{OPSAMP} MENU SYSTEM

\textit{OPSAMP} menus are created and driven by the user interface modules, \textit{PCHAND}, described in the previous chapter. The flow process of \textit{OPSAMP} menus can be seen in Fig. 3.18.
Fig. 3.18. Flow Process of the OPSAMP Program
The advantages of such a menu system are:

1. **Immediate execution of the command selected**

   The **OPSAMP** computer program using **FORTRAN-77L** instantly translates each command selected by the user and opens the menus quickly in the form of a window process.

2. **Immediate execution of FORTRAN program application**

   The user can select the run application program from the menu and execute instantly.

3. **Direct dialog through the menu system**

   The user can change the menu option dialog with the option desired directly or can use the inherent option menu if necessary.

4. **Multiple editing MENU**

   Program **OPSAMP** provides the multiple editing windows to allocate variable tasks.

The **OPSAMP** menu system basically is set up in two different file models. These models are **TEXT FILE MODEL** and **RUN-INPUT FILE MODEL** which are shown in Table 3.7 and 3.8 respectively. The files such as **LOGO** file, **OPSAMP** main menu file, and sub-bar menu file are set in **TEXT MODEL** whereas the files of the program menus such as **Mstat**, **Mdclus**, **Mscat**, **Msvar**, and **Mskrig** (Fig 3.11) require these two file models. Fig 3.19 shows that the texts displayed on the left-hand side are the numerous questions given to the user which are neatly stored in the file of **TEXT MODEL**. These files have extension **.STR** files, e.g., **GLO.STR** (Table 3.5). At the right side of each text (Fig 3.19) are the input default data displayed which are stored in the file of
RUN-INPUT FILE MODEL with extensions .PAR file, e.g., GLO.PAR. (Table 3.6).

Fig 3.19. Input phase display of program SVAR

Table 3.5

The namelist file of estimation variance
GLO.STR for text structure menu

&STR4
HLINE='Variogram Model Selection'
JROW(1)=12
JCOL(1)=17
ISTP=2
ITER=2
NDIGT(1)=7 CTP(1)='C'
NDIGT(2)=2 CTP(2)='C'
CHAR1(1)='VARIOGRAM MODEL [SPH/GAUSS/EXP] :
CHAR1(2)='NUMBER OF MODELS (MAX=5)
DINF(1)='Select a model of variogram desired'
DINF(2)='Enter the number of variogram model required'
&END
Table 3.6

The namelist file of estimation variance
**GLO.PAR** for text option menu

```
&PAR3
JROW(1)=12
JCOL(1)=51
ISTP=2
ITER=2
NDIGT(1)=4  CTP(1)=‘C’
NDIGT(2)=6  CTP(2)=‘N’
CHAR2(1)=‘SPH’
VAL(2)=1.
&END
```

The main difference between the two files (Tables 3.5 and 3.6) is that of the information storing system. In Table 3.5, **CHAR1(n)**, where \( n=1,2, \ldots \), are used to store texts of numerous questions with each text specified by a character type \( CTP(1)=‘C’ \) and followed by an explanation assigned to the variable **DINF** whereas Table 3.6, **CHAR2(n)** are used to store the run input data of text type and **VAL(k)**, \( k=1,2, \ldots \), are for the run input data of numeric type. The example of both these types can be seen in Fig 3.19.

Table 3.7

Structure of text file model

<table>
<thead>
<tr>
<th>Variables</th>
<th>Type</th>
<th>Field</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CHAR1</strong></td>
<td>C*</td>
<td>1-40</td>
<td>Text menu</td>
</tr>
<tr>
<td><strong>DINF</strong></td>
<td>C*</td>
<td>1-70</td>
<td>Menu information</td>
</tr>
</tbody>
</table>

* C=character type
Chapter 3: OPSAMP computer program 3-39

Table. 3.8
Structure of Run Input File Model

<table>
<thead>
<tr>
<th>Variables</th>
<th>Type</th>
<th>Field</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR2</td>
<td>C</td>
<td>1-30</td>
<td>Text option</td>
</tr>
<tr>
<td>VAL</td>
<td>N*</td>
<td>1-1</td>
<td>Value assigned</td>
</tr>
</tbody>
</table>

* numeric type

3.7 THE PROGRAM TREE

The program TREE is constructed in FORTRAN language for the search tree process. This determines the best subset of potential drillholes using the Branch and Bound technique described in Chapter 2.2.

The construction of the search TREE procedure has been described in Chapter 2.3. The algorithm of this procedure relating to the TREE diagram shown in Fig 2.1 is described in the following steps:

Step 1.
Assume four additional potential samples, IADD, within the area of existing samples are defined by a number of samples, NSAMP = [1,...,4]. Thus the selection of the best sample set, in this case, involves four new additional drillhole candidates.

Step 2.
Define the initial computation of estimation or kriging variance from conventional computation of either SVAR or SKRIG.
Step 3.
Set the variables \( NT=0, NDROP=0, TVAR_1=\infty, TVAR=\infty \)

Step 4.
Test the variables which are satisfied in the conditions below:

(a) \( NT=IADD \). The current number of potential holes is equal to \( IADD \) holes.
(b) \( NDROP=IADD-1 \). All dropping of every branch has been searched.
(c) \( TVAR_1<TVAR \). If the estimation (kriging) variance, \( TVAR_1 \) is less than \( TVAR \) save it as the current minimum variance of the optimum sample and transfer to \( TVAR \).

The condition (a) is satisfied if all holes in every branch of the root have been searched. The holes are added one at a time using the backward procedure. This process then goes to step 7. If none of the above conditions is satisfied, then go to step 6. But in the case of one of the conditions being satisfied, no \( TVAR \) is changed.

Step 5.
Perform a backward (updating) procedure from the same root of the branches. If no updating estimation variance is to be made in this stage, then go to step 2.

Step 6.
Perform a forward (dropping) procedure for the next root of the branches. Modify the old estimation (kriging) variance with the new one, then go to step 5.

Step 7.
If the \( NDROP=1 \) or \( NT=IADD \), then program terminates. Otherwise go to step 2.
During step 5 or 6, the process of the inversion technique by block partition is performed by calling the subprogram either ZVAR or KVAR so as to compute the estimation (kriging) variance for the new updating hole from deletion or addition of sample process. In the subprogram ZVAR, two procedures of estimation (kriging) variance computation are performed – deleting procedure and updating procedure.

3.8 ADDITIONAL OPTIONS TO OPSAMP PROGRAM

There are three additional programs BSTAT, DCLUS, and SCAT provided in the OPSAMP. These involve a service of analysing the distribution of data before the estimation is performed by either global or local procedure. The listing of these programs is provided in Appendix III.

1. Programme SCAT - SCAT is optionally recalled from the OPSAMP to display scaled drillhole data in two-dimension. The function is physically to look at the distribution of data over the whole area. From this observation, in some areas under study a cluster case can be found.

2. Programme STAT - This performs the basic calculation of the statistical parameter of data. This is useful to look at the whole suite of drillhole data by means of calculating the overall mean value and of plotting histogram over the area under study.

3. Programme DECLUSTER - This program is a technique to split the dependence model (one population) which is required in the estimation process. This program computes the declustered mean in a grid of equal size cells
where each datum receives a weight \( \frac{1}{n} \). Often drillhole sites are irregular but not randomly located over the area because, for instance, in areas of preferential access, more drillholes are located than in other areas. This irregularity is normally detected as a cluster of drillholes. As result of this, the spatial average of either global or local procedure can not be correctly estimated.

### 3.9 OPSAMP LIBRARY SYSTEM AND UTILITIES

Three library systems of integrated programs were created to speed the debugging phase of the system. The *OPSAMP.LIB* file contains four application programs *SCAT, DCLUS, BSTAT, SVAR, and SKRIG* and one supporting routine *TREE* which is used in both *SVAR* and *SKRIG*. Table 3.9 describes the respective functions of the programs. The purpose of this is to make a stand-alone program that can simply be linked with the main program.
Table 3.9

The *OPSAMP*.*LIB* content

<table>
<thead>
<tr>
<th>Name file</th>
<th>Description</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>DCLUS.FOR</em></td>
<td>Decluster program</td>
<td>Declustering Data</td>
</tr>
<tr>
<td><em>BSTAT.FOR</em></td>
<td>Elementary statistic program</td>
<td>Calculating basic statistic including mean, variance, standard deviation</td>
</tr>
<tr>
<td><em>SCAT.FOR</em></td>
<td>Scatter diagram program</td>
<td>Plotting scatter data in x-y axes</td>
</tr>
<tr>
<td><em>SVAR.FOR</em></td>
<td>Estimation variance program</td>
<td>Calculating estimation variance of global ore reserve</td>
</tr>
<tr>
<td><em>SKRIG.FOR</em></td>
<td>Program simple kriging</td>
<td>Calculating kriging of local reserve estimation</td>
</tr>
<tr>
<td><em>TREE.FOR</em></td>
<td>Branch and Bound program</td>
<td>Searching optimal drillhole samples</td>
</tr>
</tbody>
</table>

The second stand-alone program developed is *MENU.LIB*. This library contains routine programs which are used to execute the commands selected by the user during the run phase. The function of eleven routines which form the *MENU.LIB* are presented in Table 3.10.
## Table 3.10
The `MENU.LIB` content

<table>
<thead>
<tr>
<th>Name file</th>
<th>Description</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>Main menu program</td>
<td>To open main menu program</td>
</tr>
<tr>
<td>MENU</td>
<td>Sub-menu program containing the tasks</td>
<td>To display the tasks</td>
</tr>
<tr>
<td>EDIT</td>
<td>Routine <code>OPEN</code> program</td>
<td>To edit file-name of drillhole data</td>
</tr>
<tr>
<td>SAVE</td>
<td>Routine <code>SAVE</code> program</td>
<td>To save the modified data input</td>
</tr>
<tr>
<td>SCAT</td>
<td>Routine <code>SCAT</code> menu</td>
<td>To a menu of <code>SCAT</code> program</td>
</tr>
<tr>
<td>MDCLS</td>
<td>Routine <code>MDCLS</code> menu</td>
<td>To open a menu of decluster program</td>
</tr>
<tr>
<td>MSTAT</td>
<td>Routine <code>MSTAT</code> menu</td>
<td>To open a menu of <code>STAT</code> program</td>
</tr>
<tr>
<td>MSVAR</td>
<td>Routine <code>MSVAR</code> menu</td>
<td>To open a menu of <code>SVAR</code> program</td>
</tr>
<tr>
<td>MSKRIG</td>
<td>Routine <code>MSKRIG</code> menu</td>
<td>To open a menu of <code>SKRIG</code> program</td>
</tr>
<tr>
<td>RUN</td>
<td>Routine run option</td>
<td>To open a menu of run program</td>
</tr>
<tr>
<td>HELP</td>
<td>Routine <code>HELP</code> menu</td>
<td>To open <code>HELP</code> menu</td>
</tr>
</tbody>
</table>

The third of the library system is `PCHAND.LIB`. This file is a permanent routine program, which contains generalised routine programs `KEYSTROKE, CPOS, CTEXT, GETVAL,` and `WRTDAT`, which are used to edit, input, and write data or text information when displaying the window of dialog menus. Table 3-11 presents the functions of these programs.
Table 3.11
The PCHANLD.LIB content

<table>
<thead>
<tr>
<th>Name file</th>
<th>Description</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEYSTROK.FOR</td>
<td>Keyboard redefinition</td>
<td>to identify symbols (key redefinition)</td>
</tr>
<tr>
<td></td>
<td>key program</td>
<td></td>
</tr>
<tr>
<td>CPOS.FOR</td>
<td>Routine program</td>
<td>to locate the cursor by defining row and column of screen</td>
</tr>
<tr>
<td>CTEXT.FOR</td>
<td>Routine program</td>
<td>to display and writing the text</td>
</tr>
<tr>
<td>GETVAL ()</td>
<td>Function routine program</td>
<td>to read and collect data from the screen</td>
</tr>
<tr>
<td>WRTDAT.FOR</td>
<td>Routine program</td>
<td>to write data onto screen</td>
</tr>
</tbody>
</table>

The OPSAMP program utilizes an external library program PLOT88.LIB for plotting data. This PLOT88 has numerous plotting options which can be selected from the PLOT88 library.
CHAPTER 4

VALIDATION OF OPSAMP MODULE
CHAPTER 4

VALIDATION OF OPSAMP PROGRAM

The results obtained from any computer model are only as good as the assumed data and the model itself. In practice, an absolute measure of validity does not exist. In this Chapter, attempts are made to 'validate' the development drilling model. The basis of validation is presented under the procedure of program input data phase, Branch and Bound logic and response, and evaluation of the estimation (kriging) variance computation with the TREE module.

4.1 PROCEDURE OF OPSAMP INPUT PHASE

The system OPSAMP requires the following hardware:

(1) An IBM Personal Computer System which runs MS-DOS version, at least, 2.1
(2) Two floppy disks required or preferably one hard disk.
(3) At least 640 K installed memory.
(4) An American Numeric Standard Institute system file ANSI.SYS which is configured in CONFIG.SYS.
(5) Coprocessor 8087/IBM-XT or 28087/IBM-AT.

Also required is the standard keyboard convention; the KEYNAME convention is shown in Table 4.1.
Table 4.1  
Keyboard convention table

<table>
<thead>
<tr>
<th>Type of convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Esc]</td>
<td>The key used to proceed the next part of the window process.</td>
</tr>
<tr>
<td>[RETURN]</td>
<td>The enter key to input any desired option.</td>
</tr>
<tr>
<td>[↑]</td>
<td>The arrow key to move cursor direction up.</td>
</tr>
<tr>
<td>[↓]</td>
<td>The arrow key to move cursor direction down.</td>
</tr>
<tr>
<td>[→]</td>
<td>The arrow key to move cursor direction key forward to the next box dialog.</td>
</tr>
<tr>
<td>[←]</td>
<td>The arrow key to move cursor direction backward to the previous box dialog.</td>
</tr>
</tbody>
</table>

To start the program, the computer must be in an *IBM-DOS* system, working from the hard disk (*C*: ) or the floppy disk (*A*: ). The program can be run in either floppy disk or hard disk by typing the name of the program on the dot prompt 'A:' or 'C:' (hard disk). Upon starting the program the *OPSAMP* logo appears as shown in Fig 4.1.
Fig 4.1 OPSAMP logo
From the *OPSAMP* logo, the following menus are displayed below to accept data.

**a) *OPSAMP* screen.**

Next, the *MAIN* menu of the program will be displayed. From this menu, three options are offered to the user. These are sampling analysis, grade analysis and quit.

---

**Fig 4.2. *OPSAMP* screen**

**b) Sub-menu display**

Eleven tasks can be performed if option 1 is selected. These are displayed as the bar menu which appears on the right side of the screen (Fig 4.3).
The eleven tasks are:

**Task 1**: *Edit* is to open a data filename. This command allows the user to display data as well as to modify drillhole data. The subsequent windows displayed in the editing mode are shown in Fig 4.4 through 4.6.
**Fig 4.4. The first input phase of the editing data file**

**Fig 4.5. The second input phase of the editing data file**
Fig 4.6. Example of the editing drillhole data file

**Task 2**: *Save* is to save any modified data or text input from either window or template. For instance, once the data file as shown in Fig 4.6 has been modified from the window, then simply select the 'Save' command to update to opened file.

**Task 3**: *View* is to display by viewing a current plotting graph.

**Task 4**: *Mscat* is to open a scatter plot program menu as well as to display a two dimensional graphic plot. This program requires the file of data whether from the file of drillhole data or the file of two variable plots.

Fig 4.7 and 4.8 are displayed when performing task 4.
Fig 4.7. The first window of input phase of scatter program

Fig 4.8. The second window of input phase of scatter program
Fig. 4.9. The third window of input phase of scatter program
Task 5: *Mdclus* is to open declustering program menu as well as to decluster drillhole data having clustered holes. The respective windows are shown in Fig 4.9 through 4.12.

Fig 4.10. The first window of input phase of declustering program

Fig 4.11. The second window of input phase of declustering program
Fig 4.12. The third window of input phase of declustering program

Task 6: *Msvar* option opens estimation variance program menu, shown in Fig 4.12 through 4.17.

Fig 4.13. The first window of input phase of estimation variance program
Chapter 4 Validation of OPSAMP module  4- 12

Fig 4.14. The second window of input phase of estimation variance program

Fig 4.15. The third window of input phase of estimation variance program
Fig 4.16. The fourth window of input phase of estimation variance program

Fig 4.17. The fifth window of input phase of estimation variance program
Fig 4.18. The last window of input phase of estimation variance program

Task 7: Mskrig option opens simple kriging program menu shown in Fig 4.18 through 4.23.

Fig 4.19. The first window of input phase of simple kriging program
Chapter 4 Validation of OPSAMP module

Fig 4.20. The second window of input phase of simple kriging program

Fig 4.21. The third window of input phase of simple kriging program
Chapter 4 Validation of OPSAMP module

Fig 4.22. The fourth window of input phase of simple kriging program

Fig 4.23. The fifth window of input phase of simple kriging program
Fig 4.24. The fifth window of input phase of simple kriging program

Task 8: Run option executes the loaded programme in memory. It is advised to select the given options after input data have been saved in the appropriate namelist files. Fig 4.24 shows a typical task and window display.
Fig 4.25. The window of the run option menu

Task 9: *Help* option guides the user on the use of the system.

Task 10: *Menu* option returns a main menu *OPSAMP* program to enable the user to select another run process.

Task 11: *Exit* option returns to the *DOS* level.

### 4.2 BRANCH AND BOUND LOGIC AND RESPONSE

The *TREE* search procedure (Fig 4.26) systematically determines the best set of samples which gives the minimum estimation (kriging) variance. In order to ensure that the Branch and Bound technique works logically, eight hypothetical samples (four existing holes and four potential drillholes) were used to estimate a 4500 m x 4500 m block *ABCD* (Fig 4.27). The coordinates of the block corners are
and holes identified by prefix $HOLE$ are the existing samples with non-zero assay values.

Fig 4.26. $TREE$ search of 4 New Samples

Fig 4.26 depicts how the Branch and Bound algorithm works with the four samples. The results presented in Table 4.4 confirm the procedure shown in Fig 4.26.
Chapter 4 Validation of OPSAMP module 4-20

Fig 4.27. Block assigned for OPSAMP validation

Table 4.2
Block Area Definition for Program Validation

<table>
<thead>
<tr>
<th>Northing (m)</th>
<th>Easting (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>400390</td>
<td>1562890</td>
</tr>
<tr>
<td>400390</td>
<td>1567390</td>
</tr>
<tr>
<td>404890</td>
<td>1567390</td>
</tr>
<tr>
<td>404890</td>
<td>1562890</td>
</tr>
</tbody>
</table>

Table 4.3
Example of drillhole information

<table>
<thead>
<tr>
<th>Hole-Id</th>
<th>Easting(m)</th>
<th>Northing(m)</th>
<th>Assay(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>New1</td>
<td>1561000</td>
<td>403000</td>
<td>0</td>
</tr>
<tr>
<td>New2</td>
<td>1565000</td>
<td>402500</td>
<td>0</td>
</tr>
<tr>
<td>New3</td>
<td>1566000</td>
<td>406000</td>
<td>0</td>
</tr>
<tr>
<td>New4</td>
<td>1564000</td>
<td>404700</td>
<td>0</td>
</tr>
<tr>
<td>Hole1</td>
<td>1568000</td>
<td>405830</td>
<td>2.10</td>
</tr>
<tr>
<td>Hole2</td>
<td>1570000</td>
<td>404700</td>
<td>2.00</td>
</tr>
<tr>
<td>Hole3</td>
<td>1560000</td>
<td>401000</td>
<td>2.05</td>
</tr>
<tr>
<td>Hole4</td>
<td>1563000</td>
<td>408900</td>
<td>1.99</td>
</tr>
</tbody>
</table>
Chapter 4 Validation of OPSAMP module

### Table 4.4

<table>
<thead>
<tr>
<th>Deleted additional samples</th>
<th>Remaining additional samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>New1</td>
<td>New2, New3, New4</td>
</tr>
<tr>
<td>New1, New2</td>
<td>New3, New4</td>
</tr>
<tr>
<td>New1, New2, New3</td>
<td>New4</td>
</tr>
<tr>
<td>New1, New2</td>
<td>New3, New4</td>
</tr>
<tr>
<td>New1</td>
<td>New2, New3, New4</td>
</tr>
<tr>
<td>New2</td>
<td>New1, New3, New4</td>
</tr>
<tr>
<td>New2, New3</td>
<td>New1, New4</td>
</tr>
<tr>
<td>New2, New3, New4</td>
<td>New1</td>
</tr>
<tr>
<td>New2, New3</td>
<td>New1, New4</td>
</tr>
<tr>
<td>New2</td>
<td>New1, New3, New4</td>
</tr>
<tr>
<td>New3</td>
<td>New1, New2, New4</td>
</tr>
<tr>
<td>New3, New4</td>
<td>New1, New2</td>
</tr>
<tr>
<td>New3</td>
<td>New1, New2, New4</td>
</tr>
<tr>
<td>New4</td>
<td>New1</td>
</tr>
</tbody>
</table>

### 4.3 ESTIMATION (KRIGING) VARIANCE CALCULATION WITH THE TREE MODULE

After drillholes have been dropped from the available data by TREE module, the estimation (kriging) variances with the remaining samples are calculated using equations (1.5) and (2.28). To ensure that the variance computation is correct, the results obtained from OPSAMP system were compared with those obtained by a standard estimation (kriging) variance program SVAR. A hypothetical 4500 m x 4500 m block and drillhole samples shown in Fig 4.28 and Table 4.5 were used in the exercise. The spherical variogram parameters $C_0=10\, (\%)^2$, $C=95\, (\%)^2$ and $a=8000\,\text{m}$ were assigned for this exercise.
Table 4.6 through 4.9 summarize the comparative results. The complete validation results of *OPSAMP* are given in Appendix I.

![Kriging block definition with four block data centres](image)

**Fig 4.28.** Kriging block definition with four block data centres

<table>
<thead>
<tr>
<th>Easting (m)</th>
<th>Northing (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1573115</td>
<td>401515</td>
</tr>
<tr>
<td>1573115</td>
<td>402640</td>
</tr>
<tr>
<td>1575240</td>
<td>401515</td>
</tr>
<tr>
<td>1575240</td>
<td>402640</td>
</tr>
</tbody>
</table>
### Table 4.6
Summary of *OPSAMP* program for estimation variance calculation of sulfur

<table>
<thead>
<tr>
<th>Number of additional holes used</th>
<th>Estimation variance (%)$^2$</th>
<th>Additional hole identification</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.2254</td>
<td>New2, New3, New4</td>
</tr>
<tr>
<td>2</td>
<td>0.3129</td>
<td>New3, New4</td>
</tr>
<tr>
<td>1</td>
<td>0.3326</td>
<td>New4</td>
</tr>
</tbody>
</table>

### Table 4.7
Summary of *SVAR* program for estimation variance calculation of sulfur

<table>
<thead>
<tr>
<th>Number of additional holes used</th>
<th>Estimation variance (%)$^2$</th>
<th>Additional hole identification</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.2254</td>
<td>New2, New3, New4</td>
</tr>
<tr>
<td>2</td>
<td>0.3129</td>
<td>New3, New4</td>
</tr>
<tr>
<td>1</td>
<td>0.3326</td>
<td>New4</td>
</tr>
</tbody>
</table>

### Table 4.8
Summary of *OPSAMP* program for kriging variance calculation of sulfur

<table>
<thead>
<tr>
<th>Block Centre Coordinates, m</th>
<th>Kriging variance</th>
<th>Hole used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Easting    Northing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1574115.  401515.</td>
<td>1.4581</td>
<td>New3, New4, New1</td>
</tr>
<tr>
<td>1574115.  401515.</td>
<td>1.4584</td>
<td>New4, New1</td>
</tr>
<tr>
<td>1574115.  401515.</td>
<td>1.4587</td>
<td>New1</td>
</tr>
</tbody>
</table>
Table 4.9
Summary of standard KRIGING program for kriging variance calculation of sulfur

<table>
<thead>
<tr>
<th>Block Centre Coordinates, m</th>
<th>Kriging variance ($%$)$^2$</th>
<th>Hole identification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Easting</td>
<td>Northing</td>
<td></td>
</tr>
<tr>
<td>1574115.</td>
<td>401515.</td>
<td>1.4581</td>
</tr>
<tr>
<td>1574115.</td>
<td>401515.</td>
<td>1.4584</td>
</tr>
<tr>
<td>1574115.</td>
<td>401515.</td>
<td>1.4587</td>
</tr>
</tbody>
</table>

4.4 AN APPLICATION OF OPSAMP PROGRAM

This section compares estimate variance results reported by Kim (1981) with those obtained using the OPSAMP system. The main difference in the two approaches is that OPSAMP "optimises" the combination of the best available set of samples by computer whereas Kim's approach uses the manual selection of the best set. Fig 4.29 shows the property area and the locations of existing and additional holes.
The spatial variabilities of both thickness and sulfur are shown in Fig 4.30 and Fig 4.31 and the statistics and spherical variogram parameters used in this study are given in Table 4.10 and 4.11. The management proposed to drill an additional 26 holes at a cost of $10,000/drillhole. Table 4.12 summarizes the locations of the additional holes suggested by two independent groups, Company and Cervantes. Using estimation variance, Kim (1981) found that the reliability of the global coal reserve can be achieved by 15 holes instead of 26 holes.

Using OPSAMP for both the proposed Company and Cervantes drillhole data gives the best result as shown in Fig 4.32. The results show that the combination of the best set of sample using the Company's data and OPSAMP is not much different to that originally selected by Cervantes. Using both Cervantes drillholes and OPSAMP also gave better results than those determined by Cervantes.

Fig.4.29. The property area and locations of existing and additional drillholes
Table 4.10

Statistical parameters

<table>
<thead>
<tr>
<th>Statistic Parameters</th>
<th>Thickness</th>
<th>Sulfur</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>51.02 in</td>
<td>2.7 %</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>11.56 in</td>
<td>1.17 %</td>
</tr>
<tr>
<td>Number of samples</td>
<td>123</td>
<td>112</td>
</tr>
</tbody>
</table>

Figure 4.30. Experimental and Fitted Variogram for Thickness
Fig 4.31. Experimental and Fitted Variogram for Sulfur

Table 4.11

Spherical Variogram Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Thickness</th>
<th>Sulfur</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nugget (Co)</td>
<td>10.0 in²</td>
<td>0.20 (%/²)</td>
</tr>
<tr>
<td>C</td>
<td>95.0 in²</td>
<td>1.16 (%/²)</td>
</tr>
<tr>
<td>Range (a)</td>
<td>8000 ft</td>
<td>6600 ft</td>
</tr>
<tr>
<td>No.</td>
<td>Company</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td></td>
<td>Easting</td>
<td>Northing</td>
</tr>
<tr>
<td>1</td>
<td>1584281</td>
<td>429197</td>
</tr>
<tr>
<td>2</td>
<td>1587572</td>
<td>427849</td>
</tr>
<tr>
<td>3</td>
<td>1584438</td>
<td>422118</td>
</tr>
<tr>
<td>4</td>
<td>1581545</td>
<td>423574</td>
</tr>
<tr>
<td>5</td>
<td>1578221</td>
<td>426401</td>
</tr>
<tr>
<td>6</td>
<td>1578533</td>
<td>417766</td>
</tr>
<tr>
<td>7</td>
<td>1580533</td>
<td>419591</td>
</tr>
<tr>
<td>8</td>
<td>1582049</td>
<td>421605</td>
</tr>
<tr>
<td>9</td>
<td>1588522</td>
<td>419268</td>
</tr>
<tr>
<td>10</td>
<td>1588458</td>
<td>421267</td>
</tr>
<tr>
<td>11</td>
<td>1585297</td>
<td>423958</td>
</tr>
<tr>
<td>12</td>
<td>1582210</td>
<td>425150</td>
</tr>
<tr>
<td>13</td>
<td>1580866</td>
<td>427120</td>
</tr>
<tr>
<td>14</td>
<td>1575934</td>
<td>424030</td>
</tr>
<tr>
<td>15</td>
<td>1575934</td>
<td>430042</td>
</tr>
<tr>
<td>16</td>
<td>1577730</td>
<td>429591</td>
</tr>
<tr>
<td>17</td>
<td>1576717</td>
<td>431969</td>
</tr>
<tr>
<td>18</td>
<td>1579445</td>
<td>431670</td>
</tr>
<tr>
<td>19</td>
<td>1582403</td>
<td>430895</td>
</tr>
<tr>
<td>20</td>
<td>1584046</td>
<td>434373</td>
</tr>
<tr>
<td>21</td>
<td>1585465</td>
<td>438045</td>
</tr>
<tr>
<td>22</td>
<td>1587471</td>
<td>435069</td>
</tr>
<tr>
<td>23</td>
<td>1585067</td>
<td>436116</td>
</tr>
<tr>
<td>24</td>
<td>1578117</td>
<td>438322</td>
</tr>
<tr>
<td>25</td>
<td>1586284</td>
<td>430862</td>
</tr>
<tr>
<td>26</td>
<td>1588977</td>
<td>431201</td>
</tr>
</tbody>
</table>
Fig 4.32 Comparison of development drilling design with and without \textit{OPSAMP} program
CHAPTER 5

SUMMARY AND CONCLUSION
5.1. SUMMARY

Program _OPSAMP_ has been developed using _FORTRAN-77L_ compiler. The basis of application of the program is to aid the geologist and mining engineer in developing the optimal sampling strategy for ore and non-ore reserve estimation.

The procedure of optimal sampling design using the _OPSAMP_ program principally does not depend upon the value of individual sample grade; instead the use of the underlying variogram model of the existing samples allows arrival at a geostatistical solution of the impact of additional samples. The system developed takes advantage of the geostatistical method in assessing the best selection of new drill-sites. These best new drill-sites chosen by _OPSAMP_ must be part of the possible new drill-sites which are commonly predetermined by the geologist. In other words, the proposed new drill-site must be determined by a geologist prior to using the _OPSAMP_ program.

The program offers either global or local estimation procedure for analysing optimal sampling design. In the global estimation procedure, all existing samples are used to contribute to the impact of any new additional sample while in the local procedure, the contribution of the selected samples around the block is used for optimizing sampling development. In addition to this, the procedure of the selection of every combination of the sample set utilizes the subroutine _TREE_. This subroutine
performs the search procedure, then from each combination of the sample set the estimation (kriging) variance is computed. Every selection of samples creates the systematic error; this error is statistically called dispersion of error distribution. Using the geostatistical method it can be minimized by proper selection of the best set of samples using estimation (kriging) computation. So, the minimum variance estimation provides the best selection of samples from those available.

The program OPSAMP uses the ANSI codes to drive multi-window input-output menus for displaying data or text informations. The program has been purposely developed for easy use. The program can handle a maximum of 150 additional data (drillhole data) and it can be extended to a limit of 250 data. Furthermore, the program can be run in, at least PC-Turbo/IBM-XT with the aid of co-processor of XT computer type.

5.2. CONCLUSION

The purpose of developing the OPSAMP program is to assist the geologist to avoid the misjudgement of locating the sites of holes or at least to minimize the error produced by ill-placed drill sites as well as to allow a quick determination of the impact of an additional drillhole site for ore reserve estimation. Throughout the results obtained from the case example given in Chapter 4, it was found that the selection of the optimal sampling program by the computer is one of the alternative ways which can practically reduce the error made by manual selection. The improvement of the optimal sampling program can arbitrarily be made prior to actual drilling procedure; thus the more the improvement is achieved the better the result is obtained with the minimum sampling error.
The developed *OPSAMP* program is fruitfully used to select the additional samples pre-determined by the geologist in conjunction with the available marginal cost of the drilling program. Even though the impact of arbitrary samples on the evaluation of ore reserve may affect the mean assay value of samples, the reduction of this mean might not reduce the quality of ore estimate too much because the improvement of sampling development can basically provide the reliability of ore estimate.

5.3 RECOMMENDATION FOR FUTURE WORK

The program *OPSAMP* developed can only be suitable in assessing the optimal sampling design with no drift assumed. In other words the presence of drift can be assumed to be close to zero value. Under drift assumption, the *OPSAMP* program is not well-suited. But the alternative way to cope with this problem, an application of universal kriging variance method is more fruitful for such a case.

In order to promote the extensive use of the program in a practical way, the program should be developed for two type functions of estimation procedure (with drift and no drift). In addition to these, a declustering technique should be incorporated for reducing the influence of more weights from the rich zone to the poor zone which may affect the uncertainty of the estimation procedure.

The program developed can suitably be run in at least Turbo/IBM-XT because the computer speed is required for rapid processing of data. Even though the technique of fixed selection by rejecting unnecessary samples is set up in the program for cutting down the search process of *TREE*, in searching a set of finite possible combination of samples, the computer speed seems to be the major priority to assist the project to be done quickly.
BIBLIOGRAPHY


Barnes, R.J. and Johnson, T.B., 1984, Positive Kriging, in G. Verly, M. David, A.G. Journel, and A. Marechal (Eds) Geostatistics for Natural Resources Characterization, NATO ASI Series, pp. 231-244.


*Kim, Y. C., 1981, Use of Geostatistics to improve Development Drilling Activity - Case Study I, Applications of Geostatistics to Coal Resources Characterization - Case Study Examples, Department of Mining and Geological Engineering, The University of Arizona, Tuscon, June, pp. 11-25.


*Papers referred to in this thesis are indicated by an asterisk (*)
APPENDIX I

OUTPUT OF PROGRAM VALIDATION
VALIDATION OF OPSAMP PROGRAM
FOR GLOBAL ESTIMATION PROCEDURE

INPUT OF OPSAMP PROGRAM:

FILES OF DATA

DRILLHOLE DATA FILE NAME : DRILL.DAT
BLOCK DATA FILE NAME : BLK.DAT
NAMELIST FILE NAME : SVAR.LST

OUTPUT OF OPSAMP PROGRAM:

OUTPUT FILE NAME : SVAR.OUT
SUMMARY OUTPUT FILE NAME : REPORT.FIL
DATA FILES OF OPSAMP PROGRAM

DRILL.DAT

COAL DATA
(A10,4F12.2)
1
[SULFUR]
NEW1  403000.  1561000.  0.  0.0
NEW2  402500.  1565000.  0.  0.0
NEW3  406000.  1566000.  0.  0.0
NEW4  404700.  1564000.  0.  0.0
HOLE1 405830.  1568000.  0.  2.1
HOLE2 404700.  1570000.  0.  2.0
HOLE3 401000.  1560000.  0.  2.05
HOLE4 408900.  1563000.  0.  1.99

BLOCK.DAT

400369.,  1562890.,  0.0
404890.,  1562890.,  0.0
404890.,  1567390.,  0.0
4003690.,  1567390.,  0.0
SVAR.LST

\&SVR
TITLE = 'COAL PROJECT'
FILENAME = 'DRILL.DAT'
LIST = 'YES'
NOPT = 10.0
BLOCK_TYPE='POLY' POLY_FIL='BLK.DAT'
TYPE_OF_VG = 'SPH'
NESTED_GH = 1
SILL = 1.28
C(1) = 0.82
CO = 0.46
RANG(1) = 12000
XMIN = 1560000
YMIN = 400000
ZMAX = 1000
WIDX = 2500
WIDY = 2500
WIDZ = 2500
NDX = 4
NDY = 4
NDZ = 1
ALPHA = 0
BETA = 0
THETA = 0
SEMI_MAJOR = 1
MINOR = 0
\&END
OUTPUT OF OPSAMP PROGRAM

RUN PARAMETERS:

ANGLES OF ROTATION

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ALPHA</td>
<td>0.0</td>
</tr>
<tr>
<td>BETA</td>
<td>0.0</td>
</tr>
<tr>
<td>THETA</td>
<td>0.0</td>
</tr>
</tbody>
</table>

ANISOTROPY FACTORS

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SEMI-MAJOR</td>
<td>1.0</td>
</tr>
<tr>
<td>MINOR</td>
<td>0.0</td>
</tr>
</tbody>
</table>

VARIOGRAM PARAMETERS

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SILL</td>
<td>1.28</td>
</tr>
<tr>
<td>NUGGET EFFECT</td>
<td>0.46</td>
</tr>
<tr>
<td>MODEL</td>
<td>1</td>
</tr>
<tr>
<td>C-VALUE</td>
<td>0.82</td>
</tr>
<tr>
<td>RANGE</td>
<td>12000.00</td>
</tr>
</tbody>
</table>

DISCRETIZATION PARAMETERS

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NDX= NUMBER OF POINTS EAST-WEST</td>
<td>4</td>
</tr>
<tr>
<td>NDY= NUMBER OF POINTS NORTH-SOUTH</td>
<td>4</td>
</tr>
<tr>
<td>NDZ= NUMBER OF POINTS VERTICALLY</td>
<td>1</td>
</tr>
</tbody>
</table>
Appendix I - 6

DRILLHOLE ANALYSIS

NUMBER OF HOLES = 8
BLOCK VARIANCE = 0.11177E+03
VARIANCE OF A POINT WITHIN THE BLOCK = 0.70199E+00
COVARIANCE BETWEEN SAMPLES = 0.83592E+00
COVARIANCE SAMPLES AND BLOCK = 0.87318E+00
ESTIMATION VARIANCE = 0.20846E+00

DRILLHOLE LOCATIONS

<table>
<thead>
<tr>
<th>HOLES</th>
<th>EAST</th>
<th>NORTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW1</td>
<td>0.156E+07</td>
<td>0.403E+06</td>
</tr>
<tr>
<td>NEW2</td>
<td>0.156E+07</td>
<td>0.402E+06</td>
</tr>
<tr>
<td>NEW3</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
<tr>
<td>NEW4</td>
<td>0.156E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>HOLE1</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
<tr>
<td>HOLE2</td>
<td>0.157E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>HOLE3</td>
<td>0.156E+07</td>
<td>0.401E+06</td>
</tr>
<tr>
<td>HOLE4</td>
<td>0.156E+07</td>
<td>0.409E+06</td>
</tr>
</tbody>
</table>

NUMBER OF HOLES = 7
NUMBER OF NEW HOLES = 3
BLOCK VARIANCE = 0.11177E+03
VARIANCE OF A POINT WITHIN THE BLOCK = 0.70199E+00
COVARIANCE BETWEEN SAMPLES = 0.81639E+00
COVARIANCE SAMPLES AND BLOCK = 0.87189E+00
ESTIMATION VARIANCE = 0.22540E+00

DRILLHOLE LOCATIONS

NEW HOLES IN SAMPLING

<table>
<thead>
<tr>
<th>NEW HOLES IN SAMPLING</th>
<th>EAST</th>
<th>NORTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW2</td>
<td>0.156E+07</td>
<td>0.402E+06</td>
</tr>
<tr>
<td>NEW3</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
<tr>
<td>NEW4</td>
<td>0.156E+07</td>
<td>0.405E+06</td>
</tr>
</tbody>
</table>
NUMBER OF HOLES = 6
NUMBER OF NEW HOLES = 2

BLOCK VARIANCE = 0.11177E+03
VARIANCE OF A POINT WITHIN THE BLOCK = 0.70199E+00
COVARIANCE BETWEEN SAMPLES = 0.80884E+00
COVARIANCE SAMPLES AND BLOCK = 0.91189E+00
ESTIMATION VARIANCE = 0.31296E+00

**DRILLHOLE LOCATIONS**

<table>
<thead>
<tr>
<th>NEW HOLES IN SAMPLING</th>
<th>EAST</th>
<th>NORTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW3</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
<tr>
<td>NEW4</td>
<td>0.156E+07</td>
<td>0.405E+06</td>
</tr>
</tbody>
</table>

NUMBER OF HOLES = 5
NUMBER OF NEW HOLES = 1

BLOCK VARIANCE = 0.11177E+03
VARIANCE OF A POINT WITHIN THE BLOCK = 0.70199E+00
COVARIANCE BETWEEN SAMPLES = 0.82420E+00
COVARIANCE SAMPLES AND BLOCK = 0.92944E+00
ESTIMATION VARIANCE = 0.33269E+00

**DRILLHOLE LOCATIONS**

<table>
<thead>
<tr>
<th>NEW HOLES IN SAMPLING</th>
<th>EAST</th>
<th>NORTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW4</td>
<td>0.156E+07</td>
<td>0.405E+06</td>
</tr>
</tbody>
</table>
EXECUTION OF ESTIMATION VARIANCE PROGRAM

RUN PARAMETERS:

<table>
<thead>
<tr>
<th>ANGLES OF ROTATION</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ALPHA</td>
<td>0.0</td>
</tr>
<tr>
<td>BETA</td>
<td>0.0</td>
</tr>
<tr>
<td>THETA</td>
<td>0.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ANISOTROPY FACTORS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SEMI-MAJOR</td>
<td>1.0</td>
</tr>
<tr>
<td>MINOR</td>
<td>0.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VARIOGRAM PARAMETERS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SILL</td>
<td>1.28</td>
</tr>
<tr>
<td>NUGGET EFFECT</td>
<td>0.46</td>
</tr>
<tr>
<td>MODEL</td>
<td>1</td>
</tr>
<tr>
<td>C-VALUE</td>
<td>0.82</td>
</tr>
<tr>
<td>RANGE</td>
<td>12000.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DISCRETIZATION PARAMETERS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NDX= NUMBER OF POINTS EAST-WEST</td>
<td>4</td>
</tr>
<tr>
<td>NDY= NUMBER OF POINTS NORTH-SOUTH</td>
<td>4</td>
</tr>
<tr>
<td>NDZ= NUMBER OF POINTS VERTICALLY</td>
<td>1</td>
</tr>
</tbody>
</table>
DEFINITION OF BLOCK-NODES

(1562890., 400390., 0.0)
(1562890., 404890., 0.0)
(1567390., 404890., 0.0)
(1567390., 400390., 0.0)

AREA OF BLOCK = 0.20256670E+08

NUMBER OF HOLES = 7
COVARIANCE SAMPLES AND BLOCK = 0.87189E+00
COVARIANCE BETWEEN SAMPLES = 0.81639E+00
VARIANCE OF A POINT WITHIN THE BLOCK = 0.70199E+00
ESTIMATION VARIANCE = 0.22540E+00

DRILLHOLE LOCATIONS

<table>
<thead>
<tr>
<th>HOLES</th>
<th>EAST</th>
<th>NORTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW2</td>
<td>0.156E+07</td>
<td>0.402E+06</td>
</tr>
<tr>
<td>NEW3</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
<tr>
<td>NEW4</td>
<td>0.156E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>HOLE1</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
<tr>
<td>HOLE2</td>
<td>0.157E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>HOLE3</td>
<td>0.156E+07</td>
<td>0.401E+06</td>
</tr>
<tr>
<td>HOLE4</td>
<td>0.156E+07</td>
<td>0.409E+06</td>
</tr>
</tbody>
</table>
DEFINITION OF BLOCK-NODES

(1562890., 400390., 0.0)
(1562890., 404890., 0.0)
(1567390., 404890., 0.0)
(1567390., 400390., 0.0)

AREA OF BLOCK = 0.20256670E+08

NUMBER OF HOLES = 6
COVARIANCE SAMPLES AND BLOCK = 0.91189E+00
COVARIANCE BETWEEN SAMPLES = 0.80884E+00
VARIANCE OF A POINT WITHIN THE BLOCK = 0.70199E+00
ESTIMATION VARIANCE = 0.31296E+00

DRILLHOLE LOCATIONS

<table>
<thead>
<tr>
<th>HOLES</th>
<th>EAST</th>
<th>NORTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW3</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
<tr>
<td>NEW4</td>
<td>0.156E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>HOLE1</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
<tr>
<td>HOLE2</td>
<td>0.157E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>HOLE3</td>
<td>0.156E+07</td>
<td>0.401E+06</td>
</tr>
<tr>
<td>HOLE4</td>
<td>0.156E+07</td>
<td>0.409E+06</td>
</tr>
</tbody>
</table>
DEFINITION OF BLOCK-NODES

(1562890., 400390., 0.0)
(1562890., 404890., 0.0)
(1567390., 404890., 0.0)
(1567390., 400390., 0.0)

AREA OF BLOCK = 0.20256670E+08

NUMBER OF HOLES = 5
COVARIANCE SAMPLES AND BLOCK = 0.92944E+00
COVARIANCE BETWEEN SAMPLES = 0.82420E+00
VARIANCE OF A POINT WITHIN THE BLOCK = 0.70199E+00
ESTIMATION VARIANCE = 0.33269E+00

DRILLHOLE LOCATIONS

<table>
<thead>
<tr>
<th>HOLES</th>
<th>EAST</th>
<th>NORTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW4</td>
<td>0.156E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>HOLE1</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
<tr>
<td>HOLE2</td>
<td>0.157E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>HOLE3</td>
<td>0.156E+07</td>
<td>0.401E+06</td>
</tr>
<tr>
<td>HOLE4</td>
<td>0.156E+07</td>
<td>0.409E+06</td>
</tr>
</tbody>
</table>
VALIDATION OF OPSAMP PROGRAM FOR LOCAL ESTIMATION PROCEDURE

INPUT OF OPSAMP PROGRAM:

FILES OF DATA

DRILLHOLE DATA FILE NAME : DRILL.DAT
BLOCK DATA FILE NAME : NODE.DAT
NAMELIST FILE NAME : SKRIG.LST

OUTPUT OF OPSAMP PROGRAM:

OUTPUT FILE : SKRIG.OUT
REPORT FILE : REPORT.FIL
DATA FILES OF OPSAMP PROGRAM

NODE.DAT

<table>
<thead>
<tr>
<th>Node</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>401515</td>
<td>1574115</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>402640</td>
<td>1574115</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>401515</td>
<td>1575240</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>402640</td>
<td>1575240</td>
<td>0.0</td>
<td></td>
</tr>
</tbody>
</table>

SKRIG.LST

&SKRG
TITLE = 'COAL PROJECT'
FILENAME = 'DRILL.DAT'
GEN_BLOCKS = 'NO'
LIST = 'YES'
SEARCH_TYPE = 'USUAL'
NOPT = 10.0
BLOCK_FILE = 'NODE.DAT'
TYPE_OF_VG = 'SPH'
NESTED_GH = 1
SILL = 1.28
C(1) = 0.82
CO = 0.46
RANG(1) = 12000
XMIN = 1560000
YMIN = 400000
ZMAX = 1000
WIDX = 2500
WIDY = 2500
WIDZ = 2500
NDX = 4
NDY = 4
NDZ = 1
RMAX = 200000
ZLIMT = 40
MIN_HOLES = 2
MAX_HOLES = 20
ALPHA = 0
BETA = 0
THETA = 0
SEMI MAJOR = 1
MINOR = 0
&END
OUTPUT OF OPSAMP PROGRAM

RUN PARAMETERS:

ANGLES OF ROTATION

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALPHA</td>
<td>0.0</td>
</tr>
<tr>
<td>BETA</td>
<td>0.0</td>
</tr>
<tr>
<td>THETA</td>
<td>0.0</td>
</tr>
</tbody>
</table>

ANISOTROPY FACTORS

<table>
<thead>
<tr>
<th>Factor</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEMI-MAJOR</td>
<td>1.0</td>
</tr>
<tr>
<td>MINOR</td>
<td>0.0</td>
</tr>
</tbody>
</table>

VARIOGRAM PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SILL</td>
<td>1.28</td>
</tr>
<tr>
<td>NUGGET EFFECT</td>
<td>0.46</td>
</tr>
<tr>
<td>MODEL</td>
<td>1</td>
</tr>
<tr>
<td>C-VALUE</td>
<td>0.82</td>
</tr>
<tr>
<td>RANGE</td>
<td>12000.00</td>
</tr>
</tbody>
</table>

DISCRETIZATION PARAMETERS

- NDX: NUMBER OF POINTS EAST-WEST
- NDY: NUMBER OF POINTS NORTH-SOUTH
- NDZ: NUMBER OF POINTS VERTICALLY

THE BLOCK CENTRE DATA

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>401515</td>
<td>1574115</td>
<td>0.0</td>
</tr>
<tr>
<td>402640</td>
<td>1574115</td>
<td>0.0</td>
</tr>
<tr>
<td>401515</td>
<td>1575240</td>
<td>0.0</td>
</tr>
<tr>
<td>402640</td>
<td>1575240</td>
<td>0.0</td>
</tr>
</tbody>
</table>
### DRILLHOLE ANALYSIS

**Kriged Block**

(1574115., 401515., 0.0)

<table>
<thead>
<tr>
<th>Number of Holes</th>
<th>= 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block Variance</td>
<td>= 0.10500E+01</td>
</tr>
<tr>
<td>Kriging Variance</td>
<td>= 0.14520E+01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Holes</th>
<th>East</th>
<th>North</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW2</td>
<td>0.156E+07</td>
<td>0.402E+06</td>
</tr>
<tr>
<td>NEW3</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
<tr>
<td>NEW4</td>
<td>0.156E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>NEW1</td>
<td>0.156E+07</td>
<td>0.403E+06</td>
</tr>
<tr>
<td>HOLE4</td>
<td>0.156E+07</td>
<td>0.409E+06</td>
</tr>
<tr>
<td>HOLE2</td>
<td>0.157E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>HOLE3</td>
<td>0.156E+07</td>
<td>0.401E+06</td>
</tr>
<tr>
<td>HOLE1</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Number of Holes</th>
<th>= 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of New Holes</td>
<td>= 3</td>
</tr>
<tr>
<td>Block Variance</td>
<td>= 0.10500E+01</td>
</tr>
<tr>
<td>Kriging Variance</td>
<td>= 0.14581E+01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>New Holes in Sampling</th>
<th>East</th>
<th>North</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW3</td>
<td>0.157E+07</td>
<td>0.406E+06</td>
</tr>
<tr>
<td>NEW4</td>
<td>0.156E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>NEW1</td>
<td>0.156E+07</td>
<td>0.403E+06</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Number of Holes</th>
<th>= 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of New Holes</td>
<td>= 2</td>
</tr>
<tr>
<td>Block Variance</td>
<td>= 0.10500E+01</td>
</tr>
<tr>
<td>Kriging Variance</td>
<td>= 0.14584E+01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>New Holes in Sampling</th>
<th>East</th>
<th>North</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW4</td>
<td>0.156E+07</td>
<td>0.405E+06</td>
</tr>
<tr>
<td>NEW1</td>
<td>0.156E+07</td>
<td>0.403E+06</td>
</tr>
</tbody>
</table>
NUMBER OF HOLES = 5
NUMBER OF NEW HOLES = 1
BLOCK VARIANCE = 0.10500E+01
KRIGING VARIANCE = 0.14587E+01

NEW HOLES EAST NORTH IN SAMPLING

NEW1 0.156E+07 0.403E+06

OPTIMAL SOLUTION OF DRILLHOLE ANALYSIS

HOLES KRGING HOLES-ID

4 1.45202 NEW1 NEW2 NEW3 NEW4
3 1.45813 NEW3 NEW4 NEW1
2 1.45837 NEW4 NEW1
1 1.45866 NEW1
EXECUTION OF KRIGING PROGRAM

RUN PARAMETERS:

ANGLES OF ROTATION

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALPHA</td>
<td>0.0</td>
</tr>
<tr>
<td>BETA</td>
<td>0.0</td>
</tr>
<tr>
<td>THETA</td>
<td>0.0</td>
</tr>
</tbody>
</table>

ANISOTROPY FACTORS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEMI-MAJOR</td>
<td>1.0</td>
</tr>
<tr>
<td>MINOR</td>
<td>0.0</td>
</tr>
</tbody>
</table>

VARIOGRAM PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SILL</td>
<td>1.28</td>
</tr>
<tr>
<td>NUGGET EFFECT</td>
<td>0.46</td>
</tr>
<tr>
<td>MODEL</td>
<td>1</td>
</tr>
<tr>
<td>C-VALUE</td>
<td>0.82</td>
</tr>
<tr>
<td>RANGE</td>
<td>12000.00</td>
</tr>
</tbody>
</table>

DISCRETIZATION PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NDX= NUMBER OF POINTS EAST-WEST</td>
<td>4</td>
</tr>
<tr>
<td>NDY= NUMBER OF POINTS NORTH-SOUTH</td>
<td>4</td>
</tr>
<tr>
<td>NDZ= NUMBER OF POINTS VERTICALLY</td>
<td>1</td>
</tr>
</tbody>
</table>
THE BLOCK CENTRE DATA

(0.15741E+07, 0.40151E+06, 0.0)

NUMBER OF SAMPLES = 7
BLOCK VARIANCE = 1.0500
KRIGING VARIANCE = 0.14581E+01

DRILLHOLE LOCATIONS

<table>
<thead>
<tr>
<th>HOLES</th>
<th>EAST</th>
<th>NORTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOLE2</td>
<td>0.15700E+07</td>
<td>0.40470E+06</td>
</tr>
<tr>
<td>HOLE1</td>
<td>0.15680E+07</td>
<td>0.40583E+06</td>
</tr>
<tr>
<td>NEW3</td>
<td>0.15660E+07</td>
<td>0.40600E+06</td>
</tr>
<tr>
<td>NEW4</td>
<td>0.15640E+07</td>
<td>0.40470E+06</td>
</tr>
<tr>
<td>NEW1</td>
<td>0.15610E+07</td>
<td>0.40300E+06</td>
</tr>
<tr>
<td>HOLE4</td>
<td>0.15630E+07</td>
<td>0.40890E+06</td>
</tr>
<tr>
<td>HOLE3</td>
<td>0.15600E+07</td>
<td>0.40100E+06</td>
</tr>
</tbody>
</table>

THE BLOCK CENTRE DATA

(0.15741E+07, 0.40151E+06, 0.0)

NUMBER OF SAMPLES = 6
BLOCK VARIANCE = 1.0500
KRIGING VARIANCE = 0.14584E+01

DRILLHOLE LOCATIONS

<table>
<thead>
<tr>
<th>HOLES</th>
<th>EAST</th>
<th>NORTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOLE2</td>
<td>0.15700E+07</td>
<td>0.40470E+06</td>
</tr>
<tr>
<td>HOLE1</td>
<td>0.15680E+07</td>
<td>0.40583E+06</td>
</tr>
<tr>
<td>NEW4</td>
<td>0.15640E+07</td>
<td>0.40470E+06</td>
</tr>
<tr>
<td>NEW1</td>
<td>0.15610E+07</td>
<td>0.40300E+06</td>
</tr>
<tr>
<td>HOLE4</td>
<td>0.15630E+07</td>
<td>0.40890E+06</td>
</tr>
<tr>
<td>HOLE3</td>
<td>0.15600E+07</td>
<td>0.40100E+06</td>
</tr>
</tbody>
</table>
THE BLOCK CENTRE DATA

(0.15741E+07, 0.40151E+06, 0.0)

NUMBER OF SAMPLES = 5
BLOCK VARIANCE = 1.0500
KRIGING VARIANCE = 0.14587E+01

DRILLHOLE LOCATIONS

<table>
<thead>
<tr>
<th>HOLES</th>
<th>EAST</th>
<th>NORTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOLE2</td>
<td>0.15700E+07</td>
<td>0.40470E+06</td>
</tr>
<tr>
<td>HOLE1</td>
<td>0.15680E+07</td>
<td>0.40583E+06</td>
</tr>
<tr>
<td>NEW1</td>
<td>0.15610E+07</td>
<td>0.40300E+06</td>
</tr>
<tr>
<td>HOLE4</td>
<td>0.15630E+07</td>
<td>0.40890E+06</td>
</tr>
<tr>
<td>HOLE3</td>
<td>0.15600E+07</td>
<td>0.40100E+06</td>
</tr>
</tbody>
</table>
APPLICATION

OUTPUT OF OPSAMP

APPENDIX II
APPENDIX II

OUTPUT OF OPSAMP PROGRAM
APPLICATION

INPUT OF OPSAMP PROGRAM
-----------------------------

DATA FILES :

DRILLHOLE DATA FILE NAME : COAL.DAT
BLOCK DATA FILE NAME : POLY.DAT
NAMELIST FILE NAME : SVAR.LST

OUTPUT OF OPSAMP PROGRAM
----------------------------

OUTPUT FILES :

OUTPUT FILE NAME : SVAR.OUT
SUMMARY OUTPUT FILE NAME : REPORT.FIL
&SVR
TITLE = 'COAL PROJECT'
FILENAME = 'COAL.DAT'
LIST = 'YES'
NOPT = 10.0
BLOCK_TYPE = 'POLY'
POLY_FIL = 'POLY.DAT'
TYPE_OF_VG = 'SPH'
NESTED_GH = 1
SILL = 105
C(1) = 95.
CO = 8000
RANG(1) = 12000
XMIN = 1560000
YMIN = 4000000
ZMAX = 1000
WIDX = 4000
WIDY = 4000
WIDZ = 1
NDX = 17
NDY = 18
NDZ = 1
ALPHA = 0
BETA = 0
THETA = 0
SEMI_MAJOR = 1
MINOR = 0
&END
## OUTPUT OF OPSAMP PROGRAM
USING COMPANY DRILLHOLE DATA

### OPTIMAL SOLUTION OF DRILLHOLE ANALYSIS

<table>
<thead>
<tr>
<th>NO. HOLES</th>
<th>ESTIMATION VARIANCE</th>
<th>POTENTIAL HOLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.945148</td>
<td>NEW1 NEW2 NEW3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW4 NEW5</td>
</tr>
<tr>
<td>10</td>
<td>0.830749</td>
<td>NEW1 NEW2 NEW3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW4 NEW5 NEW6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW7 NEW8 NEW25</td>
</tr>
<tr>
<td>15</td>
<td>0.762585</td>
<td>NEW1 NEW2 NEW3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW4 NEW5 NEW6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW7 NEW8 NEW9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW20 NEW21 NEW22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW23 NEW24 NEW25</td>
</tr>
<tr>
<td>20</td>
<td>0.673815</td>
<td>NEW6 NEW7 NEW8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW9 NEW10 NEW11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW12 NEW13 NEW14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW15 NEW16 NEW17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW18 NEW19 NEW20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW21 NEW22 NEW23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW24 NEW25</td>
</tr>
<tr>
<td>25</td>
<td>0.660260</td>
<td>NEW1 NEW2 NEW3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW4 NEW5 NEW6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW7 NEW8 NEW9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW10 NEW11 NEW12</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW13 NEW14 NEW15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW16 NEW17 NEW18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW19 NEW20 NEW21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW22 NEW23 NEW24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NEW25</td>
</tr>
</tbody>
</table>
## OUTPUT OF OPSAMP PROGRAM USING CERVANTES DRILLHOLE DATA

### OPTIMAL SOLUTION OF DRILLHOLE ANALYSIS

<table>
<thead>
<tr>
<th>NO. HOLES</th>
<th>ESTIMATION VARIANCE</th>
<th>POTENTIAL HOLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.935267</td>
<td>NEW20 NEW21 NEW22 NEW23 NEW25</td>
</tr>
<tr>
<td>10</td>
<td>0.830118</td>
<td>NEW1 NEW2 NEW3 NEW4 NEW5 NEW6 NEW7 NEW8 NEW9 NEW10</td>
</tr>
<tr>
<td>15</td>
<td>0.732466</td>
<td>NEW1 NEW2 NEW3 NEW4 NEW5 NEW6 NEW7 NEW8 NEW9 NEW10 NEW11 NEW12 NEW13 NEW14 NEW15</td>
</tr>
<tr>
<td>20</td>
<td>0.651444</td>
<td>NEW1 NEW2 NEW3 NEW4 NEW5 NEW6 NEW7 NEW8 NEW9 NEW10 NEW11 NEW12 NEW13 NEW14 NEW15 NEW16 NEW17 NEW18 NEW19 NEW20 NEW21 NEW22 NEW23 NEW24 NEW25</td>
</tr>
<tr>
<td>25</td>
<td>0.608607</td>
<td>NEW1 NEW2 NEW3 NEW4 NEW5 NEW6 NEW7 NEW8 NEW9 NEW10 NEW11 NEW12 NEW13 NEW14 NEW15 NEW16 NEW17 NEW18 NEW19 NEW20 NEW21 NEW22 NEW23 NEW24 NEW25</td>
</tr>
</tbody>
</table>
Program

Listing of OPSAMP

Appendix III
APPENDIX III

LISTING OF OPSAMP PROGRAM

Program Menu: A Menu Program System
Author: Zul Ichwan
Department of Civil and Mining Engineering
University of Wollongong
Wollongong, Australia
Program language: FORTRAN-77L
Version: August, 1990

General Information:
Program Menu provides the following capabilities:
1. Data editing
2. Menu editing (Statistic, Decluster, Scatter, Local and Global estimation programs)
3. Plotting (On Screen, Dot Printer, Plotter)
4. Guidance Information

Program Limitation:
1. 150 drillhole data
2. Co-processor required

COMMON/VARI/CHAR1(50),CHAR2(50),DINF2(50),CTP(50),VAL(50)
+C(5),RANG(5)
COMMON/PAR/PST,HLINE
COMMON/FLASH/PSH
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/SWT/NCOD,ISW,IVAR
COMMON/SEL/OPT,KEYS
COMMON/CALL/NTYPE,CHK
CHARACTER*80 PST,PSH,HLINE
CHARACTER*5 CF1,CB1,CF2,CB2
C
C CLEAR A SCREEN
C
5 PRINT,' [0m [2J'
C
C OPEN A TRANSPARENT CONSOLE
C
OPEN(40,FILE='CON',ACCESS='TRANSPARENT')
C
C OPEN TEMPLATE FILES
C
OPEN(45,FILE='WINDW1.WDW',ACCESS='SEQUENTIAL',STATUS='OLD')
OPTION BREAK
CHK=0.
PRINT,' [=3h'
PRINT,' [? 7h'
CALL BGROUND(23)
CALL BOX
CALL LOGO
CALL MAIN
10 CALL MENU
IF(NCOD.EQ.8) GOTO 20
GOTO 10
20 PRINT,' [0m [2J'
PRINT,' [41m', '[34m', '[12:25f', 'Get out from OPSAMP program'
PRINT,' [0m'
STOP
END
SUBROUTINE LOGO

Subroutine to display a logo "OPS AMP"

COMMON/V ARI/CHAR1(50),CHAR2(50),DINF2(50),CTP(50),VAL(50) 
+C(5)
COMMON/LENGTH/LTG
COMMON/SWT/NCOD,ISW,IVAR
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/PAR/PST,HLINE
COMMON/FORMT/FRM,LCHAR
COMMON/SEL/OPT,KEYS
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
NAMELIST/LGO/CHAR1,JROW,JCOL,NDIGT,ISTP,ITER,CTP
CHARACTER PST*80,DINF2*80,CTP*1,CHAR1*80,CHAR2*80
1 ,HLINE*80,FRM*10
CHARACTER*5 CB1,CF1,CB2,CF2

Call background color to paint the OPS AMP screen

IF(NCOD.EQ.10) THEN
CALL BGROUND(23)
CALL BOX
ENDIF
CB1=' [47m'
CF1=' [33m'
CB2=' [44m'
CF2=' [30m'

Open OPSAMP logo

OPEN(UNIT=43,FILE='LOGO.LGO,,STATUS='OLD',ACCESS='SEQUENTIAL')
READ(43,LGO)
REWIND 43
CB1=' [47m'
CF1=' [34m'
CB2=' [44m'
CF2=' [30m'

CALL LENCHAR(LCHAR, CHARI, ITER)
JRW=JROW(1)
JCL=JCOL(1)
CALL FRAME(JRW-1,JCL-5,CB1,CF1,LCHAR,NDIGT,1,ITER,ISTP)
JRW=JROW(1)
JCL=JCOL(1)
DO 10 I=1,ITER
CALL DSPLAY(JRW,JCL,CB1,CF1,IB,1)
IB=IB+1
JRW = JRW + ISTP

10 CONTINUE

CB1 = ' [47m'
CF1 = ' [34m'

NDAT = IER
PST = 'Press any Key to continue !'
CALL CTEXT(24,25,CB2,CF2,1,LEN(CHARNB(PST)))
READ(40,'(A1)') KEY
RETURN
END
C-----------------------------|
SUBROUTINE BOX
C-----------------------------|
C
C Subroutine to draw a box around the main menu  |
C-----------------------------|
COMMON/PAR/PST,HLINE
COMMON/FLASH/PSH
DIMENSION CHAR(5)
CHARACTER*5 CF,CB
CHARACTER*80 PST,PSH,HLINE,CHAR
NAMELIST/BX/CHAR,KCOL,KROW,ITER,CB,CF
C-----------------------------|
C Open a box template in the 'BOX.DRW' file  |
C-----------------------------|
OPEN(56,FILE='BOX.DRW',STATUS='OLD',ACCESS='SEQUENTIAL +L')
READ(56,BX)
DO 10 I=1,ITER
IF(I.EQ.1) PST=CHAR(1)
IF(I.GT.1.AND.I.LT.ITER) PST=CHAR(2)
IF(I.EQ.ITER) PST=CHAR(3)
K=LEN(CHARB(PST))
CALL CTEXT(KROW+I,KCOL,CB,CF,1,k)
10 CONTINUE
REWIND(56)
RETURN
END
SUBROUTINE HLIGHT

COMMON/PAR/PST,HLINE
COMMON/FLASH/PSH
NAMELIST/FLS/CHAR
DIMENSION CHAR(5)
CHARACTER*80 PST,PSH,HLINE,CHAR

OPEN(57,FILE='FLASH.ELS',STATUS='OLD',ACCESS='SEQUENTIAL')
READ(57,FLS)
PST=CHAR(1)
PSH=CHAR(2)
REWIND(57)
RETURN
END
SUBROUTINE BGROUND(LOOP)

This subroutine provides the background color of the OPSAMP screen.

DIMENSION CHAR(20)
COMMON/PAR/PST,HLINE
COMMON/FLASH/PSH
NAMELIST/BG/CHAR
   CHARACTER*80 PST,PSH,HLINE,CHAR
OPEN(55,FILE='BGROUND.DRW',STATUS='OLD',ACCESS='SEQUENTIAL')
READ(55,BG)
DO 10 I=1,LOOP
   IF(I.EQ.1) PST=CHAR(1)
   IF(I.GT.1.AND.I.LT.23) PST=CHAR(2)
   IF(I.EQ.23) PST=CHAR(3)
   L=LEN(CHAR(PST))
   CALL CTEXT(I,1,'[34m1','[43m',1,L)
10 CONTINUE
REWIND(55)
RETURN
END
SUBROUTINE MENU

COMMON/CHARI(50),CHAR2(50),DINF(50),CTP(50),VAL(50)
1,C(5),RANG(5)
COMMON/DATA/HOLID(150),Y(150),X(150),Z(150),G(150),WT(150),N
1HOL
COMMON/SEL/OPT,KEYS
COMMON/DIV/IDIV,IDIF
COMMON/LENGTH/LTG
COMMON/NSV/NSAV,CHAR3(25),GVAL(50)
COMMON/PAR/PST,HLINE
COMMON/FORMT/FRM,LCHAR
COMMON/SWT/NCOD,ISW,IVAR
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
NAMELIST/SMN/HLINE,JROW,JCOL,CHAR1,ISTP,ITER,DINF,NDI
+GT,CTP
CHARACTER*5 CB1,CF1,CB2,CF2
CHARACTER*80 CHAR1*80,CHAR2*80,CTP*1,DINF*80,HOLID*10,HLI
+NE*80,
1 PST*80,CHAR3*80,FRM*10
CB1=' [47m'
CF1=' [34m'
CF2=' [43m'
CB2=' [30m'
IDIV=1

C Set ISW=1 For calling menu of sampling program
C ISW=2 For calling menu of ore analysis program
C ISW=3 For getting out of the programme

15 DO 18 I=1,50
CHAR1(I)=" 
CHAR2(I)=" 
CTP(I) ="
DINF(I)="
VAL(I) =0.
18 CONTINUE
KEYS=OPT
ISW=1

OPEN(22,FILE='SUB1.SMN',ACCESS='SEQUENTIAL',STATUS='OL
1D')
READ(22,SMN)
REWIND 22
LCHAR=0
CALL FRAME(JROW(1),JCOL(1)-1,CB2,'35m',LCHAR,NDIGT,1,
1ITER-1,ISTP)
ELSEIF(OPT.EQ.2) THEN
  KEYS=OPT
  OPEN(22,FILE='SUB1.SMN',ACCESS='SEQUENTIAL',STATUS='OLD')
  READ(22,SMN)
  REWIND 22
  LCHAR=0

  CALL FRAME(JROW(1),JCOL(1)-1,CB2,'35m',LCHAR,NDIGT,1,ITER-1,ISTP)
  ISW=1

  CONTINUE
  REWIND 22
  LCHAR=0
  CALL FRAME(JROW(1),JCOL(1)-1,CB2,'35m',LCHAR,NDIGT,1,ITER-1,ISTP)
  ELSEIF(OPT.EQ.3) THEN
    NCOD=8
    GOTO 30
  ENDIF

  JRW=JROW(1)
  JCL=JCOL(1)
  LMT=JRW
  LMD=JROW(1)+ITER
  IB=1
  LCHAR=NDIGT(1)
  DO 20 I=1,LMD-LMT
    CALL DSPLAY(JRW,JCL,CB1,CF1,IB,1)
    IB=IB+1
    JRW=JRW+ISTP
  20 CONTINUE
  NDAT=ITER

  CALL SCROLL(JROW(1),JCOL(1))
  CALL SUBMENU

  IF(NCOD.EQ.9.OR.NCOD.EQ.8.OR.NCOD.EQ.6) THEN
    OPT=KEYS
  ENDIF

  RETURN
END
Appendix III - 10

SUBROUTINE DSPLAY(JRW, JCL, CB1, CF1, I1, J2)

C-------------------------------------------------------------------------
C-----------------------------------------------------------------------
C Subroutine to display a window program menu
C-----------------------------------------------------------------------

COMMON/VARI/CHAR1(50), CHAR2(50), DINF(50), CTP(50), VAL(50),
+C(5), RANG(5)
COMMON/DATA/HOLID(150), Y(150), X(150), Z(150), G(150), WT(150),
1 NHOL
COMMON/PAR/PST,HLINE
COMMON/FLASH/PSH
COMMON/SWT/NCOD, ISW, IVAR
COMMON/LOC/JROW(10), JCOL(10), NDIGT(50), ISTP, ITER, NDAT
COMMON/SEL/OPT, KEYS
COMMON/FORMT/FRM, LCHAR
CHARACTER*5 CB1, CF1
CHARACTER CHAR1*80, CHAR2*80, HOLID*10, CTP*1, PST*80,
+ PSH*80
CHARACTER*80, DINF*80, HLINE*80, FRM*10
CALL HLIGHT
IF(ISW.EQ.1) THEN
C---------------------------------------------------------------------1
C If ISW=1, Text display only
C-------------------------------------- 1
PST(1:LCHAR)=CHAR1(I1)
CALL CTEXT(JRW, JCL, CB1, CF1, I1, NDIGT(I1))
ENDIF

C If ISW=3, Drillhole data display
ELSIF(ISW.EQ.3) THEN
CALL HLIGHT
PST=HOLID(I1)
CALL CTEXT(JRW, JCOL(1), CB1, CF1, I1, NDIGT(I1))
ENDIF

C If ISW=2, Both text and data display
ELSIF(ISW.EQ.2) THEN
IF(CTP(I1).EQ.'C') THEN
PST=CHAR2(I1)
CALL CTEXT(JRW, JCOL(2), CB1, CF1, I1, NDIGT(I1))
ELSIF(CTP(I1).EQ.'N') THEN
PST=PST(1:NDIGT(I1))
CALL WRTDAT(JRW, JCOL(2), CB1, CF1, NDIGT(I1), VAL(I1))
ENDIF

CALL HLIGHT
PST=HOLID(I1)
CALL CTEXT(JRW, JCOL(1), CB1, CF1, I1, NDIGT(I1))
CALL HLIGHT
PST=PST(1:NDIGT(2))
CALL WRTDAT(JRW, JCOL(2), CB1, CF1, NDIGT(2), X(I1))
CALL HLIGHT
PST=PST(1:NDIGT(3))
CALL WRTDAT(JRW, JCOL(3), CB1, CF1, NDIGT(3), Y(I1))
CALL HLIGHT
PST=PST(1:NDIGT(4))
CALL WRTDAT(JRW,JCOL(4),CB1,CF1,NDIGT(4),Z(I1))
CALL HLIGHT
PST=PST(1:NDIGT(5))
CALL WRTDAT(JRW,JCOL(5),CB1,CF1,NDIGT(5),G(I1))

120 CONTINUE
C----------------------------------------------------------------------|
C  If ISW=4, Highlighted Texts or Data on the given column          |
C----------------------------------------------------------------------|
ELSEIF(ISW.EQ.4) THEN
CALL HLIGHT
IF(I2.EQ.1) THEN
PST=HOLID(I1)
CALL CTEXT(JRW,JCL,CB1,CF1,1,NDIGT(I2))
GOTO 30
ELSEIF(I2.EQ.2) THEN
VALUE=X(I1)
ELSEIF(I2.EQ.3) THEN
VALUE=Y(I1)
ELSEIF(I2.EQ.4) THEN
VALUE=Z(I1)
ELSEIF(I2.EQ.5) THEN
VALUE=G(I1)
ENDIF
PST=PST(1:NDIGT(I2))
CALL WRTDAT(JRW,JCL,CB1,CF1,NDIGT(I2),VALUE)
ENDIF
30 RETURN
END
SUBROUTINE SCROLL(JRW,JCL)

SUBROUTINE to scroll the cursor up and down

COMMON/VARI/CHAR1(50),CHAR2(50),DINF(50),CTP(50),VAL(50)
1,C(5),RANG(5)
COMMON/PAR/PST,HLINE
COMMON/DATA/HOLID(150),Y(150),X(150),Z(150),G(150),WT(150),
1NHOL
COMMON/DIV/IDIV,IDIF
COMMON/FLASH/PSH
COMMON ITP,ERR
COMMON/SWT/NCOD,ISW,IVAR
COMMON/LENGTH/LTG
COMMON/SEL/OPT,KEYS
COMMON/FORMT/FRM,LCFLAR
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
CHARACTER*5 CB1,CF1,CB2,CF2
CHARACTER CHAR1*80,CHAR2*80,DINF*80,CTP*1,PST*80,
+PSH*80,HOLID*10,HLINE*80,FRM*10
I2=1
NC=0

An upper row limit and a lower row limit setting

LMT=JRW
LMD=(ITER-1)*ISTP+LMT
JR=JRW
JC=JCL
I1=1
I2=1
KW=0
ILG=1

Mark an ASCII code used to NCOD

IF NCOD =1, Cursor up
NCOD =2, Cursor down
NCOD =3, Move right
NCOD =4, Move left
NCOD =5, Return Key
NCOD =6, Escape
NCOD =7, DELETE
NCOD =10, Clear SCREEN
30 IF(ISW.GE.3) GOTO 32
PST=PSH
CALL CTEXT(24,19,' [45m',' [30m',1,50)

KW =0 a switch used to pass all current variables

IF(KW.EQ.0) THEN
CALL DSPLAY(JR,JC,CB2,CF2,I1,I2)
C    ISW = 2  Data display onto screen
C---------------------------------------------------------------
IF(ISW.GT.2) GOTO 33
PST=DINF(I1)
   CALL CTEXT(24,20,' [45m',I,LEN(CHARNB(DINF(I1))))
ENDIF
C---------------------------------------------------------------
C Call key symbol function
C---------------------------------------------------------------
33 CALL KEYPASS(JR,JC,NCOD,I1)
C---------------------------------------------------------------
C Move cursor UP
C---------------------------------------------------------------
IF(NCOD.EQ.1) THEN
   JR1=JR
   JC1=JC
   JB1=I1
   JB2=I2
   JR=JR-ISTP
   I1=I1-1
   KW=0
   IF(JR.LT.LMT) THEN
      PRINT,CHAR(7)
      JR=LMT
      I1=I1+1
      JR1=JRW
      JB1=I1
      KW=1
   ENDIF
C-------------------------
C Cursor DOWN
C-------------------------
ELSEIF(NCOD.EQ.2) THEN
   JR1=JR
   JC1=JC
   JB1=I1
   JB2=I2
   JR=JR+ISTP
   I1=I1+1
   KW=0
   IF(JR.GT.LMD) THEN
      PRINT,CHAR(7)
      JR=LMD
      I1=I1-1
      JR1=JRW
      JB1=I1
      KW=1
   ENDIF
C-------------------------------
C Cursor forward
C-------------------------------
ELSEIF(NCOD.EQ.3) THEN
   JC1=JC
   JR1=JR
   JB1=I1
If IDIV=1 one column provided

Else if IDIV>1 More one column provided

IF(IDIV.EQ.1) THEN
  JC=JCOL(1)
  ILG=1
ELSEIF(IDIV.GT.1) THEN
  move the cursor to the given column JCOL

I2=I2+1
JC=JCOL(I2)
IF(IDIV.EQ.0) ILG=1
IF(IDIV.EQ.1) ILG=2
ENDIF
KW=0
IF(I2.GE.IDIV) THEN
  I2=IDIV
  JC=JCOL(I2)
  PRINT,CHAR(7)
  NC=NC+1
ENDIF
IF(NC.LE.1) I1=I1+ITER*(ILG-1)

Cursor backward

ELSEIF(NCOD.EQ.4) THEN
  NC=0
  JR1=JR
  JC1=JC
  JB1=I1
  JB2=I2
  KW=0
  IF(IDIV.EQ.1) THEN
    JC=JCOL(1)
    ILG=1
  ELSEIF(IDIV.GT.1) THEN
    I2=I2-1
    JC=JCOL(I2)
    IF(IDIV.EQ.0) ILG=1
    IF(IDIV.EQ.1) ILG=2
    ENDIF
    IF(I2.LT.1) THEN
      I2=1
      JC=JCOL(I2)
      PRINT,CHAR(7)
      I1=1
    ELSE
      I1=I1-ITER*(ILG-1)
    ENDIF
ENDIF
ELSEIF(NCOD.EQ.5) THEN
  OPT=11
  IF(CTP(I1),EQ,'C'.AND.ISW.NE.1.AND.ISW.EQ.2) THEN
    CHAR2(I1)=PST(1:ITP)
    GOTO 30
  ELSEIF(CTP(I1),EQ,'N'.AND.ISW.LE.2) THEN
    VAL(I1)=GETVAL()
    GOTO 30
  ELSEIF(ISW.GE.3.AND.I2.EQ.1) THEN
    HOLID(I1)=PST(1:ITP)
    GOTO 30
  ELSEIF(ISW.GE.3.AND.I2.EQ.2) THEN
    X(I1)=GETVAL()
    GOTO 30
  ELSEIF(ISW.GE.3.AND.I2.EQ.3) THEN
    Y(I1)=GETVAL()
    GOTO 30
  ELSEIF(ISW.GE.3.AND.I2.EQ.4) THEN
    Z(I1)=GETVAL()
    GOTO 30
  ELSEIF(ISW.GE.3.AND.I2.EQ.5) THEN
    G(I1)=GETVAL()
    GOTO 30
  ENDIF
  CALL SCREEN(JR,JC,CB1,CF1,I1,1,1,1,1,KW)
  GOTO 40
ELSEIF(NCOD.EQ.6) THEN
  OPT=4
  NCOD=9
  GOTO 40
ENDIF

CALL SCREEN(JR,JC,CB1,CF1,I1,1,1,1,1,KW)
GOTO 40
ELSEIF(NCOD.EQ.7) THEN
  OPT=5
  NCOD=9
  GOTO 40
ENDIF

IF(KW.EQ.1.AND.ISW.EQ.4) THEN
  Scroll data onto the screen every ten at a time
  controlled by NACC=-1 to move UP
  and NACC=1 to move DOWN
ENDIF

IF(NCOD.EQ.1.AND.I1.GT.1) NACC=-1
IF(NCOD.EQ.1.AND.I1.EQ.1) GOTO 35
IF(NCOD.EQ.2.AND.I1.LT.NHOL) NACC=1
IF(NCOD.EQ.2.AND.I1.GT.NHOL) GOTO 35
IR=JROW(1)
JL=JCOL(1)
IN=I1
ISW=3
DO 44 LK=1,10
  IN=IN+NACC
  CALL DISPLAY(IR,JL,CB1,CF1,IN,1)
IR=IR+1
CONTINUE
ISW=4
JC=JCOL(1)
KW=0
I1=IN
JB1=I1
ENDIF

35 ITR1=1
IF(ISW.EQ.3.AND.KW.EQ.1) THEN
ISW=3
CALL SCREEN(JR1,JC1,CB1,CF1,JB1,JB2,ISTP,ITR1,1,KW)
ISW=4
I1=I1+ITR1
GOTO 30
ENDIF
IF (NCOD.GT.4) GOTO 40
CALL SCREEN(JR1,JC1,CB1,CF1,JB1,JB2,ISTP,ITR1,1,KW)
GOTO 30
40 RETURN
END
SUBROUTINE SCREEN(JR1,JC1,CB1,CF1,JB1,ILAG,I1,ITR1,ITR2,1,KW)

Subroutine to screen a selected menu

CHARACTER*5 CB1,CF1
COMMON/SWT/NCOD,ISW,IVAR
COMMON/VARI/CHAR1(50),CHAR2(50),DINF(50),CTP(50),VAL(50),IC(5),RANG(5)
COMMON/DATA/HOLID(150),Y(150),X(150),Z(150),G(150),WT(150),1NHOL
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
CHARACTER HOLID*10
JR2=JR1
JB2=JB1
JB3=JB1
JC2=JC1
ILG=ILAG
DO 10 I=1,ITR1
DO 11 J=1,ITR2
CALL DSPLAY(JR2,JC2,CB1,CF1,JB2,ILG)
ILG=ILG+1
JC2=JC2+ITR1
JB2=JB2+ITR1
11 CONTINUE
ILG=ILAG
JB3=JB3+1
JC2=JC1
JR2=JR2+I1
JB2=JB3
10 CONTINUE
KW=0
RETURN
END
C-- SUBROUTINE WINDOW1
----------------------------------------

Subroutine to open a template file
----------------------------------------

DIMENSION CHAR1(16),JROW(50),JCOL(50)
COMMON/PAR/PST,HLINE
NAMELIST/TEMPL1/JROW,JCOL,ITER,CHAR1,ISTP
CHARACTER*80 CHAR1,PST,HLINE
CHARACTER*5 CLRB,CLRF
READ(45,TEMPL1)
REWIND 45
CLRB=' [44m'
CLRF=' [37m'
KLOC=JROW(1)
DO I=1,ITER
PST=CHAR1(I)
LENCHR=LEN(CHARNB(PST))
CALL CTEXT(KLOC,JCOL(I),CLRB,CLRF,1,LENCHR)
KLOC=KLOC+ISTP
CONTINUE
RETURN
END
SUBROUTINE SUBMENU

Subroutine to display routine tasks of OPSAMP program

COMMON/V ARI/CHAR1(50),CHAR2(50),DINF(50),CTP(50),VAL(50)
COMMON/RANG(5)
COMMON/PAR/PST,HLINE
COMMON/SWT/NCOD,ISW,IVAR
COMMON/CALL/NTYPE,CHK
COMMON/SEL/OPT,KEYS
COMMON/NSV/NSAV,CHAR3(25),GVAL(50)
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/D AT A/HOLID(150),Y(150),X(150),Z(150),G(150),WT(150),
+NHOL
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
CHARACTER HLINE*80,CHAR3*80,HOLID*10
CHARACTER*5 CB1,CF1,CB2,CF2
CHARACTER PST*80,DINF*80,CHAR1*80,CHAR2*80,CTP*1
NTYPE=0
IVAR=0

Call routine tasks

PST='[Esc =Next menu'
CALL CTEXT(24,2,' [42m',' [31m',l,LEN(CHARNB(PST)))
IF(CHAR1(OPT).EQ.'Edit') THEN
   ISW=3
   CALL EDIT(1)
ELSEIF(CHAR1(OPT).EQ.'View') THEN
   CALL MVIEW
ELSEIF(CHAR1(OPT).EQ.'Mscat') THEN
   CALL MSCAT
ELSEIF(CHAR1(OPT).EQ.'Mdclus') THEN
   IVAR=1
   ISW=3
   CALL MDCLUS
ELSEIF(CHAR1(OPT).EQ.'Mstat') THEN
   IVAR=2
   ISW=3
   CALL MSTAT
ELSEIF(CHAR1(OPT).EQ.'Msvar') THEN
   IVAR=3
   ISW=3
   CALL MGLOBAL
ELSEIF(CHAR1(OPT).EQ.'Mskrig') THEN
   IVAR=4
   CALL MLOCAL
ELSEIF(CHAR1(OPT).EQ.'Run') THEN
   CALL RUN
ELSEIF(CHAR1(OPT).EQ.'Save') THEN
   CALL SAVE
ELSEIF(CHAR1(OPT).EQ.'Help') THEN
   CALL HELP
C Clear all texts from the monitor
C
NCOD=10
ELSEIF(CHAR1(OPT).EQ.'Menu') THEN
NCOD=10
CALL MAIN
ELSEIF(CHAR1(OPT).EQ.'Exit') THEN
CALL EXIT
ENDIF
RETURN
END
SUBROUTINE MVIEW

C-------------------------------------------------|      
C Subroutine to display estimation variance vs a number of potential holes |      
C-------------------------------------------------|      
C
COMMON/VARI/CHAR1(50),CHAR2(50),DINF(50),CTP(50),VAL(50)  
+C(5),RANG(5)  
COMMON/LENGTH/LTG  
COMMON/CALL/NTYPE,CHK  
COMMON/DATA/HOLID(150),Y(150),X(150),Z(150),G(150),WT(150),  
+N HOL  
COMMON/SWT/NCOD,ISW,IVAR  
COMMON/NSV/NSAV,CHAR3(25),GVAL(50)  
COMMON/COLOR/CB1,CF1,CB2,CF2  
COMMON/PAR/PST,HLINE  
COMMON/SEL/OPT,KEYS  
COMMON/FORMT/FRM,LCHAR  
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT  
DIMENSION X1(150),Y1(150),G1(150)  
NAMELIST/STR1/NWIND  
NAMELIST/STR2/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,ITER,DINF  
NAMELIST/STR3/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,ITER,DINF  
NAMELIST/PAR1/JROW,JCOL,CHAR2,CTP,ISTP,NDIGT,ITER  
CHARACTER PST*80,DINF*80,HOLID*10,CTP*1,CHAR1*80,  
+CHAR2*80  
CHARACTER*5 CB1,CF1,CB2,CF2,FRMT*25  
CHARACTER  HLINE*80,CHAR3*80,FRM*10,NAMX*25,  
+NAMY*25  
CALL BGROUND(22)  
PST='Programme X-Y plot Menu'  
CB1=' [45m'  
CF1=' [34m'  
CB2=' [43m'  
CF2=' [30m'  
CALL CTEXT(2,23,CB1,CF1,1,LEN(CHARNB(PST)))  
LK=0  
NTYPE=2  
FRM='(F7.2)'  
NSAV=2  
L=1  
C-------------------------------------------------|      
C Read files containing structure and run parameter program |      
C-------------------------------------------------|      
OPEN(UNIT=43,FILE='MVIEW.STR',ACCESS='SEQUENTIAL',  
+STATUS='OLD')  
OPEN(UNIT=55,FILE='MVIEW.PAR',ACCESS='SEQUENTIAL',  
STATUS='OLD')  
READ(43,STR1)  
5 IF (L.GT.NWIND) GOTO 20  
NCOD=8
IF(L.EQ.1) THEN
  ISW=1
  READ(43,STR2)
  CALL LENCHAR(LCHAR,CHAR1,ITER)
ENDIF

 CALL FRAME(JROW(1)-1,JCOL(1)-1,' [42m',' 30m',LCHAR, +NDIGT,ITER,ISTP)
  CALL SCREEN(JROW(1),JCOL(1),' [42m',CF2,1,1,ISTP,ITER +1,KW)
  READ(55,PAR1)
  ISW=2
  NCOD=0
  IT1=1
  CALL SCREEN(JROW(1),JCOL(1),' [42m',CF2,1,1,ISTP,ITER +1,KW)
ELSEIF(L.EQ.2) THEN
CALL HEADER(CHAR2(1),N,1)
  ISW=1
  READ(43,STR3)
  CALL LENCHAR(LCHAR,CHAR1,ITER)
  CALL FRAME(JROW(1)-1,JCOL(1)-5,' [44m',' [32m',LCHAR, +NDIGT,ITER,ISTP)
  CALL SCREEN(JROW(1),JCOL(1),' [44m',CF2,1,1,ISTP,ITER,1,KW)
  IT1=1
  NCOD=0
ENDIF

OPEN(29,FILE=CHAR2(l),STATUS='OLD',ACCESS='SEQUENTIAL +')
  K=0
  READ(29,*) FRMT

  K=K+1
  READ(29,FRMT,END=25) X1(K),Y1(K)
  G1(K)=1.
  GOTO 15
25 CONTINUE
  NDAT=K-1
ENDIF
L=L+1
GOTO 5

20 REWIND 43
REWIND 55
NAMX='Number of holes'
NAMY='Estimation variance [in2]' 

C-----------------------------------------------
C  Set output device  |
C-----------------------------------------------
IF(OPT.EQ.1) THEN
  IOPORT=97
  MODEL=96
ELSEIF(OPT.EQ.2) THEN
  IOPORT=0
MODEL=10
ENDIF
CALL PLT2D(X1,Y1,G1,NAMX,NAMY,NDAT,3,IOPORT,MODEL)
CALL BGROUND(23)
RETURN
END
SUBROUTINE SAVE

COMMON/VARI/CHAR1(50),CHAR2(50),DINF(50),CTP(50),+VAL(50),C(5),RANG(5)
COMMON/LENGTH/LTG
COMMON/SWT/NCOD,ISW,IVAR
COMMON/PAR/PST,HLINE
COMMON/NSV/NSAV,CHAR3(25),GVAL(50)
COMMON/SEL/OPT,KEYS
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
CHARACTER PST*80,DINF*80,CTP*1,CHAR1*80,CHAR2*80
HLINE*80,CHAR3*80

C-------------------------------------------------------------

C Create a namelist file for kriging parameters I
C-------------------------------------------------------------
GOTO (5,15,33,34) NSAV
5 OPEN(43,FILE='DCLUS.LST',STATUS='UNKNOWN',ACCESS= +'SEQUENTIAL')
WRITE(43,50) '&DCLS'
WRITE(43,51) 'TITLED, ...,CHAR3(1), ...'
WRITE(43,51) 'FNAME=', ...,CHAR3(2), ...'
WRITE(43,52) 'BLKDIM=',GVAL(3)
WRITE(43,51) 'OPTD=', ...,CHAR3(4), ...'
WRITE(43,52) 'CMIN=',GVAL(5)
WRITE(43,52) 'CMAX=',GVAL(6)
WRITE(43,52) 'NCELL=',GVAL(7)
WRITE(43,52) 'NOFF=',GVAL(8)
WRITE(43,52) 'ANISY=',GVAL(9)
WRITE(43,52) 'ANISZ=',GVAL(10)
WRITE(43,50) '&END'
GOTO 44

C-------------------------------------------------------------

C Create namelist file for BSTAT I
C-------------------------------------------------------------
5 OPEN (43 ,FILE=' STAT.LST' ,S TATUS='UNKN OWN', ACCES S= +'SEQUENTIAL')
WRITE(43,50) '&OPTIONS'
WRITE(43,51) "TITLE=","",CHAR3(1),"
WRITE(43,51) "FILENAME=","",CHAR3(2),"
WRITE(43,51) "LIST=","",CHAR3(3),"
WRITE(43,51) "LOGS=","",CHAR3(4),"
WRITE(43,51) "PROB_PLOT=","",CHAR3(5),"
WRITE(43,51) "AUTO_SCALE=","",CHAR3(6),"
WRITE(43,52) "NCELL=",VAL(7)
WRITE(43,52) "XLOW=",VAL(8)
WRITE(43,52) "CLASS_WIDTH=",VAL(9)
REWIND 43
GOTO 44

C-------------------------------------------------------------

C Write a namelist file for SVAR program I
C-------------------------------------------------------------
33 OPEN(43,FILE='SVAR.LST',STATUS='UNKNOWN',ACCESS= +'SEQUENTIAL')
WRITE(43,50) '&SVR'
WRITE(43,51) "TITLE='",CHAR3(1),"'
WRITE(43,51) "FILENAME='",CHAR3(2),"'
WRITE(43,51) "LIST='",CHAR3(3),"'
WRITE(43,51) "BLOCK_TYPE='",CHAR3(4),"'
WRITE(43,51) "POLY_FIL='",CHAR3(5),"'
WRITE(43,52) 'NOPT=',GVAL(6)
LW=6
GOTO 35
C---------------------------------------------
C Write a namelist file for SKRIG program
C---------------------------------------------
34 OPEN(43,FILE='SKRIG.LST',STATUS='UNKNOWN',ACCESS='SEQUENTIAL')
WRITE(43,50) '&SKRG'
WRITE(43,51) "TITLE='",CHAR3(1),"'
WRITE(43,51) "FILENAME='",CHAR3(2),"'
WRITE(43,51) "GEN_BLOCKS='",CHAR3(3),"'
WRITE(43,51) "LIST='",CHAR3(4),"'
WRITE(43,51) "SEARCH_TYPE='",CHAR3(5),"'
WRITE(43,52) 'NOPT=',GVAL(6)
IF(CHAR3(3).EQ.,NO'.OR.CHAR3(3).EQ.'no') THEN
WRITE(43,51) "BLOCK_FILE='",CHAR3(7),"'
ELSE
C---------------------------------------------
C Definition of area to be kriged
C---------------------------------------------
WRITE(43,52) 'NBC=',GVAL(7)
WRITE(43,52) 'NEC=',GVAL(8)
WRITE(43,52) 'NBR=',GVAL(9)
WRITE(43,52) 'NER=',GVAL(10)
WRITE(43,52) 'NLEV=',GVAL(11)
LW=11
ENDIF
35 I=0
18 LW=LW+1
I=I+1
GOTO (19,20,21) I
19 WRITE(43,51) "TYPE_OF_VG="",CHAR3(LW),"
GOTO 18
20 WRITE(43,52) "NESTED_GH=",GVAL(LW)
NP=GVAL(LW)
GOTO 18
21 I1=0
DO 10 I=1,1,NP+1
IF(LEQ.I) THEN
WRITE(43,53) "SILL=",GVAL(LW),"CO=",GVAL(LW+NP+1)
ELSE
I1=I1+1
WRITE(43,53) "C("",I1,"")=",GVAL(LW),"RANG("",I1,"")=",
+GVAL(LW+NP+1)
ENDIF
LW=LW+1
10 CONTINUE
LW=LW+NP+1
DO 11 I=1,3
    IF(I.EQ.1) THEN
        WRITE(43,52) 'XMIN=',GVAL(LW),'WIDX=',GVAL(LW+3),'NDX=',
        1GVAL(LW+6)
    ELSEIF(I.EQ.2) THEN
        WRITE(43,52) 'YMIN=',GVAL(LW),'WIDY=',GVAL(LW+3),'NDY=',
        1GVAL(LW+6)
    ELSEIF(I.EQ.3) THEN
        WRITE(43,52) 'ZMAX=',GVAL(LW),'WIDZ=',GVAL(LW+3),'NDZ=',
        1GVAL(LW+6)
    ENDIF
    LW=LW+1
11 CONTINUE
LW=LW+6
DO 12 L=1,9
    IF(NSAV.EQ.3.AND.L.LE.4) GOTO 12
    IF(L.EQ.1) THEN
        WRITE(43,52) 'RMAX=',GVAL(LW)
    ELSEIF(L.EQ.2) THEN
        WRITE(43,52) 'ZLIMIT=',GVAL(LW)
    ELSEIF(L.EQ.3) THEN
        WRITE(43,52) 'MIN_HOLES=',GVAL(LW)
    ELSEIF(L.EQ.4) THEN
        WRITE(43,52) 'MAX_HOLES=',GVAL(LW)
    ELSEIF(L.EQ.5) THEN
        WRITE(43,52) 'ALPHA=',GVAL(LW)
    ELSEIF(L.EQ.6) THEN
        WRITE(43,52) 'BETA=',GVAL(LW)
    ELSEIF(L.EQ.7) THEN
        WRITE(43,52) 'THETA=',GVAL(LW)
    ELSEIF(L.EQ.8) THEN
        WRITE(43,52) 'SEMI_MAJOR=',GVAL(LW)
    ELSEIF(L.EQ.9) THEN
        WRITE(43,52) 'MINOR=',GVAL(LW)
    ENDIF
    LW=LW+1
12 CONTINUE
OPT=4
NCOD=9
44 CONTINUE
WRITE(43,51) '&END'
RETURN
50 FORMAT(A10)
51 FORMAT(A12,A1,A20,A1)
52 FORMAT(A12,F10.2,1X,A12,F10.2,1X,A12,F10.2)
53 FORMAT(A12,I2,A6,F10.2,A12,I2,A6,F10.2,2X,I2)
END
SUBROUTINE EDIT(KCOD)

Subroutine to edit data onto screen

COMMON/VAR1YCHAR1(50),CHAR2(50),DINF(50),CTP(50),
+VAL(50),C(5),RANG(5)
COMMON/DATA/HOLID(150),Y(150),X(150),Z(150),G(150),WT(150)
+,NHOL
COMMON/SWT/NCOD,ISW,IVAR
COMMON/COLOR/CB1,CF1,CF2
COMMON/PAR/PST,HLINE
COMMON/FLASH/PSH
COMMON/DIV/IDIF,IDIF
COMMON/SEL/OPT,KEYS
COMMON/FORMT/FRM,LCHAR
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NAT
NAMELIST/FM/HLINE,JROW,JCOL,ITER,CTP,NDIGT,ISTP
CHARACTER/PST*80,DINF*80,CTP*1,CHAR1*80,CHAR2*80
CHARACTER*5 CB1,CF1,CF2
CHARACTER/NFIL*12,PSH*80,HOLID*10,HLINE*80,FRM*10
IDIV=5
IDIF=0

CB1='[41m'
CF1='[37m'
CB2='[43m'
CF2='[33m'
DO 1 I=1,100
HOLID(I)=" ",X(I)=0.
Y(I)=0.
Z(I)=0.
1 G(I)=0.
IF(KCOD.EQ.0) THEN
PST='Enter data filename to run :
ELSE
PST='Enter file name of data to be EDITED :
ENDIF
CALL CTEXT(23,12,'[43m','[34m',LEN(CHARNB(PST)))
READ(*,*) NFIL
CALL HEADER(NFIL,N,KCOD)
IF(KCOD.EQ.0) RETURN

CALL WINDOW1
PST=HLINE
CALL CTEXT(9,26,'[47m','[44m',1,LEN(CHARNB(PST)))
OPEN(UNIT=43,FILE='FORM.STR',STATUS='OLD',ACCESS=?
+‘SEQUENTIAL’)
READ(43,FM)
REWIND 43
CALL HLIGHT
2 ISW=3
I=0
JB=1
JRW=JROW(1)
JCL=JCOL(1)
7 JR=JRW
JC=JCL
   FRM='(F10.2)'
8 IF(I.EQ.10) GOTO 10
   CALL DISPLAY(JR,JC,CB1,CF1,JB,1)
   I=I+1
   JR=JR+1
   JB=JB+1
   GOTO 8
10 ISW=4
   NDAT=N
   CALL SCROLL(JROW(1),JCOL(1))
   RETURN
END
SUBROUTINE RUN

COMMON/VARI/CHAR1(50),CHAR2(50),DINF(50),CTP(50),VAL(50), +C(5),RANG(5)
COMMON/DATA/HOLID(150),Y(150),X(150),Z(150),G(150),WT(150), +NHOL
COMMON/CALL/NTYPE,NW,KCOD,IADD,NOPT
COMMON/SWT/NCOD,ISW,IVAR
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/PAR/PST,HLINE
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
COMMON/SEL/OPT,KEYS
COMMON/FORMT/FRM,LCHAR
NAMELIST/RMN/ITER,JROW,JCOL,CTP,CHAR1,NDIGT,ISTP
CHARACTER PST*80,DINF*80,HOLID*10,CTP*1
CHARACTER CHAR1*80,CHAR2*80,HLINE*80,FRM*10
CHARACTER*5 CB1,CF1,CB2,CF2
NCOD=9
ISW=1
IB=1
IF(IVAR.EQ.0) THEN
CALL BOX
OPEN(UNIT=58,FILE='RUN.MNU',STATUS='OLD',ACCESS='SEQUENTIAL')
READ(58,RMN)
REWIND(58)
CALL LENCHAR(LCHAR,CHARI,ISTP)
JRW=JROW(1)
JCL=JCOL(1)
ISW=1
PST=' Run Options '
CALL CTEXT(7,32,' [41m',' [35m',1,14)
DO 5 I=1,ITER
CALL DSPLAY(JRW,JCL,' [36m',' [42m',IB,1)
JRW=JRW+ISTP
IB=IB+1
5 CONTINUE
CALL SCROLL(JROW(1),JCOL(1))
C If IVAR=1, execute BASIC STATISTIC program |
C IVAR=2, execute SCATTER program |
C IVAR=3, execute SVAR program |
C IVAR=4, execute SKRIG program |
C IF(OPT.EQ.1) IVAR=1
IF(OPT.EQ.2) IVAR=2
IF(OPT.EQ.3) IVAR=3
IF(OPT.EQ.4) IVAR=4
ENDIF
IF(IVAR.EQ.1) THEN
CALL BSTAT
ELSEIF(IVAR.EQ.2) THEN
CALL DCLUS
NTYPE=1
CALL OPSAMP
IF(IVAR.EQ.3) THEN
  IF(IVAR.EQ.4) THEN
    NTYPE=2
    CALL OPSAMP
  ENDIF
C IF(CHK.EQ.1.) CALL OPSAMP
C IF(CHK.EQ.0.) CALL ERRM(4)
RETURN
END
SUBROUTINE HELP

This program displays the HELP information.

COMMON/VARI/CHAR1(50),CHAR2(50),DINF(50),CTP(5),C(5),RANG(5)
COMMON/LENGTH/LTG
COMMON/SWT/NCOD,ISW,IVAR
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/PAR/PST,HLINE
COMMON/FORMT/FRM,LCHAR
COMMON/SEL/OPT,KEYS
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
NAMELIST/HLP/CHAR1,JROW,JCOL,NDIGT,ISTP,ITER,CTP,
+HLINE
CHARACTER PST*80,DINF*80,CTP*1,CHAR1*80,CHAR2*80
1,HLINE*80,FRM*10
CHARACTER*5 CB1,CF1,CB2,CF2
IF(NCOD.EQ.10) THEN
CALL BGROUND(23)
CALL BOX
ENDIF
  CB1=' [47m*
  CF1=' [33m'
  CB2=' [44m'
  CF2=' [30m'
C---------------------------|
C Open HELP file |
C---------------------------|
OPEN(UNIT=43,FILE='HELP.HLP',STATUS='OLD',ACCESS='SEQUENTIAL')
READ(43,hlp)
REWIND 43
IB=1
ISW=1
CALL LENCHAR(LCHAR,CHAR1,ITER)
JRW=JROW(1)
JCL=JCOL(1)
CALL FRAME(JRW-1,JCL-5,CB1,CF1,LCHAR,NDIGT,1,ITER,ISTP)
JRW=JROW(1)
JCL=JCOL(1)
DO 10 I=1,ITER
CALL DSPLAY(JRW,JCL,CB1,CF1,IB,1)
IB=IB+1
JRW=JRW+ISTP
10 CONTINUE
PST='Press enter Key to exit!'
CALL CTEXT(24,25,CB2,CF2,1,LEN(CHARNB(PST)))
READ(40,'(A1)') KEY
NCOD=9
RETURN
END
SUBROUTINE MAIN

Subroutine to display the main menu OPS AMP program

COMMON/VARI/CHAR1(50),CHAR2(50),DINF(50),CTP(50),VAL(50)
+C(5) ,RANG(5)
COMMON/DIV/IDIV,IDIF
COMMON/LENGTH/LTG
COMMON/SWT/NCOD,ISW,IVAR
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/PAR/PST,HLINE
COMMON/CALL/NTYPE,NW,KCOD,IADD,NOPT
COMMON/FORMT/FRM,LCHAR
COMMON/CALL/OPT,KEYS
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
NAMELIST/MNU/HLINE,CHAR1,DINF,JROW,JCOL,NDIGT,ISTP,
+ITER,CTP
CHARACTER PST*80,DINF*80,HOLID*10,CH*80,CTP*1,
+CHAR1*80,CHAR2*80,HLINE*80,FRM*10
IDIV=1
IF(NCOD.EQ.10) THEN
CALL BGROUND(23)
CALL BOX
ENDIF
CB1=' [47m'
CF1=' [34m'
CB2=' [43m*
CF2=' [30m'
Open OPSAMP menu I
OPEN (UNIT=43,FILE='OPS AMP.MNU',STATUS='OLD',ACCESS=
+'SEQUENTIAL')
READ(43,MNU)
REWIND 43
CB1= ' [47m'
CF1=' [34m'
CB2=' [43m'
CF2=' [30m'
------------------------------------------------------------
Measure the character length of text displayed I
------------------------------------------------------------
CALL LENCHAR(LCHAR,CHAR1,ITER)
JR=JROW(1)
JCL=JCOL(1)
CALL FRAME(JR-1,JCL-5,CB1,CF1,LCHAR,NDIGT,1,ITER,ISTP)
JR=JROW(1)
JCL=JCOL(1)
C Display the complete main menu by calling DSPLAY subroutine

DO 10 I=1,ITER
    CALL DSPLAY(JRW,JCL,CB1,CF1,IB,1)
    IB=IB+1
    JRW=JRW+ISTP
10 CONTINUE
    CB1=' [47m'
    CF1=' [34m'
    NDAT=ITER

C Call SCROLL subroutine to drive the cursor movement

CALL SCROLL(JROW(1),JCOL(1))
NW=OPT
RETURN
END
SUBROUTINE LENCHAR(LCHAR,CHAR1,ITER)

DIMENSION CHAR1(50)
CHARACTER CHAR1*80
LCHAR=0

C Measure the character length of the text

DO 10 I=1,ITER
    IF(LCHAR.LT.LEN(CHARNB(CHAR1(I)))) THEN
        LCHAR=LEN(CHARNB(CHAR1(I)))
    ENDIF
10 CONTINUE
RETURN
END
SUBROUTINE MSCAT

------------------------------------------------------------------------
Subroutine to open the file of scatter program menu
-------------------------------------------------------------------------

COMMON/VARI/CHAR1(50),CHAR2(50),DINF(50),CTP(50),
+VAL(50),C(5),RANG(5)
COMMON/LENGTH/LTG
COMMON/CALL/NTYPE,NW,KCOD,IADD,NOPT
COMMON/DATA/HOLID(150),Y(150),X(150),Z(150),G(150),
+WT(150),NHOL
COMMON/SWT/NCOD,ISW,IVAR
COMMON/NSV,CHAR3(25),GVAL(50)
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/PAR/PST,HLINE
COMMON/SEL/OPT,KEYS
COMMON/FORMAT/FRM,CHAR
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
NAMELIST/STR1/NWIND
NAMELIST/STR2/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR3/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/PAR1/JROW,JCOL,CHAR2,CTP,ISTP,NDIGT,ITER
CHARACTER PST*80,DINF*80,HOLID*10,CTP*1,CHAR1*80,
+CHAR2*80
CHARACTER*5 CB1,CF1,CB2,CF2
CHARACTER HLINE*80,CHAR3*80,FRM*10,NAMX*15,NAMY*15
CALL BGROUND(22)
PST='Program Scatter Plot Menu'
CB1=' [45m'
CF1=' [34m'
CB2=' [43m'
CF2=' [30m'
CALL CTEXT(2,23,CB1,CF1,1,LEN(CHARNB(PST)))
LK=0
LM=0
NTYPE=2
FRM='(F7.2)'
NSAV=2
L=1

C---------------------------
C  Read "structure" and "default run parameters" program
C---------------------------

OPEN(UNIT=43,FILE='SCAT.STR',ACCESS='SEQUENTIAL',
+STATUS='OLD')
OPEN(UNIT=55,FILE='SCAT.PAR',ACCESS='SEQUENTIAL',
+STATUS='OLD')
READ(43,STR1)
5 IF (L.GT.NWIND) GOTO 20
NCOD=8
IF(L.EQ.1) THEN
ISW=1
CALL LENCHAR(LCHAR,CHAR1,ITER)

C Frame a catalog by calling subroutine FRAME

CALL FRAME(JROW(1)-1,JCOL(1)-1, [42m', [30m',LCHAR, +NDIGT,ITER,ISTP]
CALL SCREEN(JROW(1),JCOL(1), [42m',CF2,1,1,ISTP,ITER,1,KW)
READ(55,PAR1)
ISW=2
NCOD=0
IT1=1
CALL SCREEN(JROW(1),JCOL(1), [42m',CF2,1,1,ISTP,ITER
+,IT1,KW)
ELSEIF(L.EQ.2) THEN
CALL HEADER(CHAR2(1),N,1)
ISW=1
READ(43,STR3)
CALL LENCHAR(LCHAR,CHAR1,ITER)
CALL FRAME(JROW(1)-1,JCOL(1)-5, [44m', [32m',LCHAR,NDIGT, +1,ITER,ISTP]
CALL SCREEN(JROW(1),JCOL(1), [44m',CF2,1,1,ISTP,ITER,1,KW)
IT1=1
NCOD=0
ENDIF
CALL SCROLL(JROW(1),JCOL(1))
LP=0
DO 15 M=1,IT1
DO 15 N=1,ITER
LP=LP+1
LK=LK+1
IF(CTP(LP).EQ.,C) THEN
CHAR3(LK)=CHAR2(LP)
ELSEIF(CTP(LP).EQ.'N') THEN
GVAL(LK)=VAL(LP)
ENDIF
15 CONTINUE
L=L+1
GOTO 5

20 REWIND 43
REWIND 55
NAMX='X-AXIS'
NAMY='Y-AXIS'

C Set output device

IF(OPT.EQ.1) THEN
IOPORT=97
MODEL=97
ELSEIF(OPT.EQ.2) THEN
IOPORT=0
MODEL=12
ENDIF
CALL PLT2D(X,Y,G,NAMX,NAMY,NHOL,1,IOPORT,MODEL)
CALL BGROUND(22)
SUBROUTINE MDCLUS

Subroutine to open the file of decluster program menu

COMMON/VARI,CHAR1(50),CHAR2(50),DINF(50),CTP(50),
  +VAL(50),C(5),RANG(5)
COMMON/LENGTH/LTG
COMMON/CALL/NTYPE,CHK
COMMON/DATA/HOLID(150),Y(150),X(150),Z(150),G(150),WT(150),
  +NHOL
COMMON/SWT/NCOD,ISW,IVAR
COMMON/NSV/NSAV,CHAR3(25),GVAL(50)
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/PAR/PST,HLINE
COMMON/SEL/OPT,KEYS
COMMON/FORMALT/LCHAR
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
NAMELIST/STR1/NWIND
NAMELIST/STR2/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
  +ITER,DINF
NAMELIST/STR3/HLINE,JROW,JCOL,CHARI,NDIGT,CTP,ISTP,
  +ITER,DINF
NAMELIST/STR4/HLINE,JROW,JCOL,CHAR2,CTP,ISTP,NDIGT,ITER
CHARACTER PST*80,DINF*80,HOLID*10,CTP*1,CHAR1*80,
  +CHAR2*80
CHARACTER*5 CB1,CF1,CB2,CF2
CALL BGROUND(22)
CB1=' [45m'
CF1=' [34m'
CB2=' [43m'
CF2=' [30m'
PST='Program Decluster Menu'
CALL CTEXT(2,25,CB1,CF1,1,LEN(CHARNB(PST)))
LK=0
LM=0
NTYPE=2
FRM='(F7.2)'
NSAV=1
L=1

OPEN(UNIT=43,FILE='DCLUS.STR',ACCESS='SEQUENTIAL',
  +STATUS='OLD')
OPEN(UNIT=55,FILE='DCLUS.PAR',ACCESS='SEQUENTIAL',
  +STATUS='OLD')
READ(43,STR1)
5 IF (L.GT.NWIND) GOTO 20
NCOD=8
IF(L.EQ.1) THEN
ISW=1
READ(43,STR2)
CALL LENCHAR(LCHAR,CHAR1,ITER)
ENDIF
Elseif (L.EQ.1) THEN
ISW=1
READ(43,STR3)
CALL LENCHAR(LCHAR,CHAR1,ITER)
CALL FRAME(JROW(1)-1,JCOL(1)-1, [42m', [30m', LCHAR,
+NDIGT,1,ITER,ISTP)
CALL SCREEN(JROW(1),JCOL(1), [42m', CF2,1,1,ISTP,ITER,1,KW)
READ(55,PAR1)
ISW=2
NCOD=0
IT1=1
CALL SCREEN(JROW(1),JCOL(1), [42m', CF2,1,1,ISTP,ITER,
+IT1,KW)
Elseif (L.EQ.2) THEN
ISW=1
READ(43,STR4)
CALL LENCHAR(LCHAR,CHAR1,ITER)
CALL FRAME(JROW(1)-1,JCOL(1)-1, [46m', [32m', LCHAR,
+NDIGT,1,ITER,ISTP)
CALL SCREEN(JROW(1),JCOL(1), [46m', CF2,1,1,ISTP,ITER,1,KW)
READ(55,PAR2)
ISW=2
IT1=1
CALL SCREEN(JROW(1),JCOL(1), [46m', CF2,1,1,ISTP,ITER,
+IT1,KW)
Elseif (L.EQ.3) THEN
GPAR=VAL(1)
ISW=1
READ(43,STR5)
CALL LENCHAR(LCHAR,CHAR1,ITER)
CALL FRAME(JROW(1)-1,JCOL(1)-1, [42m', [35m', LCHAR,
+NDIGT,1,ITER,ISTP)
IF(GPAR.EQ.1.) ITER=ITER-1
CALL SCREEN(JROW(1),JCOL(1), [42m', CF2,1,1,ISTP,ITER,1,KW)
READ(55,PAR3)
FRM='(F12.2)'
ISW=2
IT1=1
IF(GPAR.EQ.1.) ITER=ITER-1
CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,ITER,IT1,KW)
ENDIF
CALL SCROLL(JROW(1),JCOL(1))
LP=0
DO 15 M=1,IT1
DO 15 N=1,ITER
LP=LP+1
LK=LK+1
IF(CTP(LP).EQ.'C') THEN
CHAR3(LK)=CHAR2(LP)
ELSEIF(CTP(LP).EQ.'N') THEN
GVAL(LK)=VAL(LP)
ENDIF
15 CONTINUE
   L=L+1
   GOTO 5
20 REWIND 43
    REWIND 55
    RETURN
END
SUBROUTINE MSTAT

Subroutine to open the file of statistical program menu

COMMON/VARI/CHAR1(50),CHAR2(50),DINF(50),CTP(50),VAL(50),+C(5),RANG(5)
COMMON/LENGTH/LTG
COMMON/CALL/NTYPE,CH
COMMON/DATA/HOLID(150),Y(150),X(150),Z(150),G(150),WT(150),+NHOL
COMMON/SWT/NCOD,ISW,IVAR
COMMON/NSV/NSAV,CHAR3(25),GVAL(50)
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/PAR/PST,HLINE
COMMON/SEL/OPT,KEYS
COMMON/FORMT/FRM,LCHAR
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
NAMELIST/STR1/NWIND
   NAMELIST/STR2/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,+ITER,DINF
   NAMELIST/STR3/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,+ITER,DINF
   NAMELIST/STR4/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,+ITER,DINF
   NAMELIST/PAR1/JROW,JCOL,CHAR2,CTP,ISTP,NDIGT,ITER
   NAMELIST/PAR2/JROW,JCOL,CHAR2,CTP,ISTP,NDIGT,ITER
   NAMELIST/PAR3/JROW,JCOL,VAL,CTP,ISTP,NDIGT,ITER
CHARACTER PST*80,DINF*80,HOLID*10,CTP*1,CHAR1*80,+CHAR2*80
CHARACTER*5 CB1,CF1,CB2,CF2
CHARACTER HLINE*80,CHAR3*80,FRM*10
CALL BGROUND(22)
   CB1='[45m'
   CF1='[34m'
   CB2='[43m'
   CF2='[30m'
PST='Program Basic Statistic Menu'
   CALL CTEXT(2,23,CB1,CF1,1,LEN(CHARNB(PST)))
   LK=0
   LM=0
   FRM='(F7.2)'
   NSAV=2
   L=1

C-----------------------------------------------|
C   Read structure and parameter file           |
C-----------------------------------------------|
OPEN(UNIT=43,FILE='STAT.STR',ACCESS='SEQUENTIAL',+STATUS='OLD')
OPEN(UNIT=55,FILE='STAT.PAR',ACCESS='SEQUENTIAL',+STATUS='OLD')
READ(43,STR1)
5 IF (L.GT.NWIND) GOTO 20
NCOD=8
IF(L.EQ.1) THEN
   ISW=1
   READ(43,STR2)
   CALL LENCHAR(LCHAR,CHAR1,ITER)
C--------------------------------------
C                   window frame setting
C--------------------------------------
   CALL FRAME(JROW(1)-1,JCOL(1)-1,' [42m', [30m',LCHAR,
      +NDIGT,ITER,ISTP)
   CALL SCREEN(JROW(1),JCOL(1),' [42m',CF2,1,1,ISTP,ITER,1,KW)
   READ(55,PAR1)
   ISW=2
   NCOD=0
   IT1=1
   CALL SCREEN(JROW(1),JCOL(1),' [42m',CF2,1,1,ISTP,ITER,IT1,
      +KW)
ELSEIF(L.EQ.2) THEN
   ISW=1
   READ(43,STR3)
   CALL LENCHAR(LCHAR,CHAR1,ITER)
   CALL FRAME(JROW(1)-1,JCOL(1)-1,' [46m', [32m',LCHAR,
      +NDIGT,1,ITER,ISTP)
   CALL SCREEN(JROW(1),JCOL(1),' [46m',CF2,1,1,ISTP,ITER,1,KW)
   READ(55,PAR2)
   ISW=2
   IT1=1
   CALL SCREEN(JROW(1),JCOL(1),' [46m',CF2,1,1,ISTP,ITER,IT1,
      +KW)
ELSEIF(L.EQ.3) THEN
   ISW=1
   FRM='(F7.2)'
   READ(43,STR4)
   CALL LENCHAR(LCHAR,CHAR1,ITER)
   CALL FRAME(JROW(1)-1,JCOL(1)-1,' [42m', [35m',LCHAR,
      +NDIGT,ITER,ISTP)
   CALL SCREEN(JROW(1),JCOL(1),' [42m',CF2,1,1,ISTP,ITER,1,KW)
   READ(55,PAR3)
   ISW=2
   IT1=1
   CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,ITER,IT1,KW)
ENDIF
CALL SCROLL(JROW(1),JCOL(1))
LP=0
DO 15 M=1,IT1
   DO 15 N=1,ITER
      LP=LP+1
      LK=LK+1
      IF(CTP(LP).EQ.'C') THEN
         CHAR3(LK)=CHAR2(LP)
      ELSEIF(CTP(LP).EQ.'N') THEN
         GVAL(LK)=VAL(LP)
      ENDIF
   15 CONTINUE
L=L+1
GOTO 5
20  REWIND 43
    REWIND 55
    RETURN
    END
SUBROUTINE MLOCAL

Subroutine to open the file of kriging program menu

COMMON/VARVCHAR1(50),CHAR2(50),DINF(50),CTP(50),VAL(50),
+C(5),RANG(5)
COMMON/LENGTH/LTG
COMMON/CALL/NTYPE,CHK
COMMON/DATA/HOLID(150),Y(150),X(150),Z(150),G(150),WT(150),
+NHOL
COMMON/SWT/NCOD,ISW,IVAR
COMMON/NSV/NSAV,CHAR3(25),GVAL(50)
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/PAR/PST,HLINE
COMMON/DIV/IDIV,IDIF
COMMON/SEL/OPT,KEYS
COMMON/FORMT/FRM,LCHAR
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
NAMELIST/STR1/NWIND
NAMELIST/STR2/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR3/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR4/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR5/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR6/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR7/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR8/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/PAR1/JROW,JCOL,CHAR2,CTP,ISTP,NDIGT,ITER
NAMELIST/PA R2/JROW,JCOL,CHAR2,VAL,CTP,ISTP,NDIGT,ITER
NAMELIST/PA R3/JROW,JCOL,VAL,CTP,ISTP,NDIGT,ITER
NAMELIST/PA R4/JROW,JCOL,CHAR2,VAL,CTP,ISTP,NDIGT,ITER
NAMELIST/PA R5/JROW,JCOL,VAL,CTP,ISTP,NDIGT,ITER
NAMELIST/PA R6/JROW,JCOL,VAL,CTP,ISTP,NDIGT,ITER
NAMELIST/PA R7/JROW,JCOL,VAL,CTP,ISTP,NDIGT,ITER
CHARACTER PST*80,DINF*80,HOLID*10,CTP*1,CHAR1*80,
+CHAR2*80
CHARACTER*5 CB1,CF1, CB2,CF2
CHARACTER HLINE*80,CHAR3*80,FRM*10,TEXT*4
CALL BGROUND(22)
CB1='[45m'
CF1='[34m'
CB2='[43m'
CF2='[30m'
PST='Program Simple Kriging Menu'
CALL CTEXT(2,23,CB1,CF1,1,LEN(CHARNB(PST)))
IDIV=1
IDIF=1
LK=0
LM=0
LD=0
NTYPE=2
NSAV=4
L=1
C------------------------------------------------------------------
C     Read structure and parameter files   
C------------------------------------------------------------------
     OPEN(UNIT=43,FILE='LOC.STR',ACCESS='SEQUENTIAL',
          +STATUS='OLD')
     OPEN(UNIT=55,FILE='LOC.PAR',ACCESS='SEQUENTIAL',
          +STATUS='OLD')
     READ(43,STR1)
5   IF (L.GT.NWIND) GOTO 20
    NCOD=8
    IF(L.EQ.1) THEN
      ISW=1
      READ(43,STR2)
      CALL LENCHAR(LCHAR,CHAR1,ITER)
      CALL FRAME(JROW(l)-1,JCOL(l)-1,' [42m',' [30m',LCHAR,NDIGT,
                   +ITER,ISTP)
      CALL SCREEN(JROW(l),JCOL(l),' [42m',CF2,1,1,ISTP,ITER,1,KW)
      READ(55,PAR1)
      ISW=2
      NCOD=0
      IT1=1
      CALL SCREEN(JROW(l),JCOL(l),' [46m',CF2,1,1,ISTP,ITER,IT1,
                   +KW)
    ELSEIF(L.EQ.2) THEN
      TEXT=CHAR2(3)
      ISW=1
      READ(43,STR3)
      IF(TEXT.EQ.'YES'.OR.TEXT.EQ.'Yes') THEN
        ITER=ITER-1
      ENDIF
      CALL LENCHAR(LCHAR,CHAR1,ITER)
      CALL FRAME(JROW(1)-1,JCOL(1)-1,' [46m',' [32m',LCHAR,NDIGT,
                   +ITER,ISTP)
      CALL SCREEN(JROW(1),JCOL(1),' [46m',CF2,1,1,ISTP,ITER,1,KW)
      READ(55,PAR2)
      IF(TEXT.EQ.'YES'.OR.TEXT.EQ.'Yes') THEN
        ITER=ITER-1
      ENDIF
      ISW=2
      IT1=1
      FRM='(F7.2)'
      CALL SCREEN(JROW(1),JCOL(1),' [46m',CF2,1,1,ISTP,ITER,IT1,
                   +KW)
    ELSEIF(L.EQ.3) THEN
LD=1
  IF(TEXT.EQ.'YES'.OR.TEXT.EQ.'yes'.OR.text.EQ.'Yes') THEN
    ISW=1
    READ(43,STR4)
    CALL LENCHAR(LCHAR,CHAR1,ITER)
    CALL FRAME(JROW(1)-1,JCOL(1)-1,' [42m',' [35m',LCHAR,
    +NDIGT,1,ITER,ISTP)
    CALL SCREEN(JROW(1),JCOL(1),1,1,ISTP,ITER,1,KW)
    READ(55,PAR3)
    ISW=2
    IT1=1
    CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,ITER,IT1,KW)
    LD=0
  ENDIF
  ELSEIF(L.EQ.4) THEN
    ISW=1
    READ(43,STR5)
    CALL LENCHAR(LCHAR,CHAR1,ITER)
    CALL FRAME(JROW(1)-1,JCOL(1)-1,' [43m',' [35m',LCHAR,NDIGT,
    1,ITER,ISTP)
    CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,ITER,IT1,KW)
    READ(55,PAR4)
    ISW=2
    IT1=1
    CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,ITER,IT1,KW)
  ELSEIF(L.EQ.5) THEN
    IDIV=2
    MODEL=VAL(2)+1
    ISW=1
    READ(43,STR6)
    ITER=MODEL
    CALL LENCHAR(LCHAR,CHAR1,ITER)
    CALL FRAME(JROW(1)-1,JCOL(1)-1,' [41m',' [35m',LCHAR,
    +ITER,ISTP)
    CALL SCREEN(JROW(1),JCOL(1),1,1,ISTP,ITER,1,KW)
    READ(55,PAR5)
    ISW=2
    IT1=2
    ITER=MODEL
    CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,ITER,IT1,KW)
  ELSEIF(L.EQ.6) THEN
    FRM='(F10.2)' 
    IDIV=3
    ISW=1
    READ(43,STR7)
    CALL LENCHAR(LCHAR,CHAR1,ITER)
    CALL FRAME(JROW(1)-1,JCOL(1)-1,' [45m',' [30m',LCHAR,
    +NDIGT,1
    1,ITER,ISTP)
    CALL SCREEN(JROW(1),JCOL(1),1,1,ISTP,ITER,1,KW)
    READ(55,PAR6)
    ISW=2
    IT1=3
    CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,ITER,IT1,KW)
  ELSEIF(L.EQ.7) THEN

ISW=1
  FRM='(F10.2)'
READ(43,STR8)
CALL LENCHAR(LCHAR,CHAR1,ITER)
  CALL FRAME(JROW(1)-1,JCOL(1)-1,' [47m', [30m',LCHAR,
+NDIGT,1
1,ITER,ISTP)
  CALL SCREEN(JROW(1),JCOL(1),' [47m',CF2,1,1,ISTP,ITER,1,KW)
READ(55,PAR7)
ISW=2
IT1=1
IF(L.EQ.8) THEN
  ISW=1
  READ(43,STR8)
  CALL LENCHAR(LCHAR,CHAR1,ITER)
  CALL FRAME(JROW(1)-1,JCOL(1)-1,' [42m', [33m',LCHAR,
+NDIGT,1,
1ITER,ISTP)
  CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,ITER,IT1,KW)
ELSEIF(L.EQ.8) THEN
  ISW=1
  READ(43,STR8)
  CALL LENCHAR(LCHAR,CHAR1,ITER)
  CALL FRAME(JROW(1)-1,JCOL(1)-1,' [42m', [33m',LCHAR,
+NDIGT,1,
1ITER,ISTP)
  CALL SCREEN(JROW(1),JCOL(1),' [42m',CF2,1,1,ISTP,ITER,1,KW)
ENDIF
IF(LD.EQ.1) THEN
  LD=0
  L=L+1
  GOTO 5
ELSE
  CALL SCROLL(JROW(1),JCOL(1))
ENDIF
LP=0
DO 15 M=1,IT1
  DO 15 N=1,ITER
    LP=LP+1
    LK=LK+1
    IF(CTP(LP).EQ.'C') THEN
      CHAR3(LK)=CHAR2(LP)
    ELSEIF(CTP(LP).EQ.'N') THEN
      GVAL(LK)=VAL(LP)
    ENDIF
  CONTINUE
L=L+1
IDIV=1
GOTO 5
20 REWIND 43
REWIND 55
RETURN
END
SUBROUTINE MGLOBAL

COMMON/V ARI/CHAR1(50),CHAR2(50),DINF(50),CTP(50),
+VAL(50),C(5),RANG(5)
COMMON/LENGTH/LTG
COMMON/CALL/NTYPE,CHK
COMMON/DATA/HOLID(150),Y(150),X(150),Z(150),G(150),
+WT(150),NHOL
COMMON/SWT/NCOD,ISW,IVAR
COMMON/NSV/NSAV,CHAR3(25),GVAL(50)
COMMON/COLOR/CB1,CF1,CB2,CF2
COMMON/PAR/PST,HLINE
COMMON/SEL/OPT,KEYS
COMMON/DIV/IDIV,IDIF
COMMON/FORMT/OPT,FRM,LCHAR
COMMON/LOC/JROW(10),JCOL(10),NDIGT(50),ISTP,ITER,NDAT
NAMELIST/STR1/NWIND
NAMELIST/STR2/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR3/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR4/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR5/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR6/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR7/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/STR8/HLINE,JROW,JCOL,CHAR1,NDIGT,CTP,ISTP,
+ITER,DINF
NAMELIST/PAR1/JROW,JCOL,CHAR2,CTP,ISTP,NDIGT,ITER
NAMELIST/PAR2/JROW,JCOL,CHAR2,VAL,CTP,ISTP,NDIGT,ITER
NAMELIST/PAR3/JROW,JCOL,CHAR2,VAL,CTP,ISTP,NDIGT,ITER
NAMELIST/PAR4/JROW,JCOL,VAL,CTP,ISTP,NDIGT,ITER
NAMELIST/PAR5/JROW,JCOL,VAL,CTP,ISTP,NDIGT,ITER
NAMELIST/PAR6/JROW,JCOL,VAL,CTP,ISTP,NDIGT,ITER
CHARACTER PST*80,DINF*80,HOLID*10,CTP*1,CHAR1*80
+,CHAR2*80
CHARACTER*5 CB1,CF1,CB2,CF2
CHARACTER  HLINE*80,CHAR3*80,FRM*10
CALL BGROUND(22)
CB1=' [45m'
CF1=' [34m'
CB2=' [43m'
CF2=' [30m'
PST='Program Estimation Variance Menu'
CALL CTEXT(2,20,CB1,CF1,1,LEN(CHARNB(PST)))
IDIV=1
IDIF=1
LK=0
LM=0
NTYPE=1
NSAV=3
L=1
C------------------------------C
C Read structure and parameter files
C------------------------------C
      OPEN(UNIT=43,FILE='GLO.STR',ACCESS='SEQUENTIAL',
+STATUS='OLD')
      OPEN(UNIT=55,FILE='GLO.PAR',ACCESS='SEQUENTIAL',
+STATUS='OLD')
      READ(43,STR1)
      5 IF (L.GT.NWIND) GOTO 20
      NCOD=8
      IF(L.EQ.1) THEN
        ISW=1
        READ(43,STR2)
        CALL LENCHAR(LCHAR,CHAR1,ITER)
        C ---------------------------
        C window frame setting
        C------------------------------------
        CALL FRAME(JROW(1)-1,JCOL(1)-1,' [42m',' [30m',LCHAR,NDIGT
+1,ITER,ISTP)
        CALL SCREEN(JROW(1),JCOL(1),[' [42m',CF2,1,1,ISTP,ITER,1,KW)
        READ(55,PAR1)
        ISW=2
        NCOD=0
        IT1=1
        CALL SCREEN(JROW(1),JCOL(1),[' [42m',CF2,1,1,ISTP,ITER,IT1,
+KW)
      ELSEIF(L.EQ.2) THEN
        ISW=1
        READ(43,STR3)
        CALL LENCHAR(LCHAR,CHAR1,ITER)
        CALL FRAME(JROW(1)-1,JCOL(1)-1,' [46m',' [32m',LCHAR,NDIGT
+1,ITER,ISTP)
        CALL SCREEN(JROW(1),JCOL(1),[' [46m',CF2,1,1,ISTP,ITER,1,KW)
        READ(55,PAR2)
        ISW=2
        IT1=1
        FRM='(F5.2)'
        CALL SCREEN(JROW(1),JCOL(1),[' [46m',CF2,1,1,ISTP,ITER,IT1,
+KW)
      ELSEIF(L.EQ.3) THEN
        ISW=1
        READ(43,STR4)
        CALL LENCHAR(LCHAR,CHAR1,ITER)
        CALL FRAME(JROW(1)-1,JCOL(1)-1,' [41m',' [35m',LCHAR,
+NDIGT,1,
+ITER,ISTP)
        CALL SCREEN(JROW(1),JCOL(1),[' [41m',CF2,1,1,ISTP,ITER,1,KW)
        READ(55,PAR3)
        ISW=2
        IT1=1
        CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,ITER,IT1,KW)
      ELSEIF(L.EQ.4) THEN
        IDIV=2
MODEL=VAL(2)+1
ISW=1
READ(43,STR5)
ITER=MODEL
CALL LENCHAR(LCHAR,CHAR1,ITER)
   CALL FRAME(JROW(1)-1,JCOL(1)-1,' [45m', [30m',LCHAR,
   +NDIGT,,MODEL,ISTP)
   CALL SCREEN(JROW(1),JCOL(1),' [45m',CF2,1,1,ISTP,MODEL,1,
   +KW)
READ(55,PAR4)
ISW=2
IT1=2
ITER=MODEL
   CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,MODEL,IT1,
   +KW)
ELSEIF(L.EQ.5) THEN
   IDIV=6
   ISW=1
   FRM='(F10.2)'
   READ(43,STR6)
   CALL LENCHAR(LCHAR,CHAR1,ITER)
   CALL FRAME(JROW(1)-1,JCOL(1)-1,' [47m', [30m',LCHAR,
   +NDIGT,1
1,ITER,ISTP)
   CALL SCREEN(JROW(1),JCOL(1),' [47m',CF2,1,1,ISTP,ITER,1,KW)
READ(55,PAR5)
ISW=2
IT1=3
   FRM='(F10.2)'
   CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,ITER,IT1,KW)
ELSEIF(L.EQ.6) THEN
   ISW=1
   READ(43,STR7)
   CALL LENCHAR(LCHAR,CHAR1,ITER)
   CALL FRAME(JROW(1)-1,JCOL(1)-1,' [42m', [33m',LCHAR,
   +NDIGT,1
1,ITER,ISTP)
   CALL SCREEN(JROW(1),JCOL(1),' [42m',CF2,1,1,ISTP,ITER,1,KW)
ISW=2
READ(55,PAR6)
   FRM='(F10.2)'
   IT1=1
   CALL SCREEN(JROW(1),JCOL(1),CB1,CF1,1,1,ISTP,ITER,IT1,KW)
ENDIF
CALL SCROLL(JROW(1),JCOL(1))
LP=0
DO 15 M=1,IT1
DO 15 N=1,ITER
   LP=LP+1
   LK=LK+1
   IF(CTP(LP).EQ.'C') THEN
      CHAR3(LK)=CHAR2(LP)
   ELSEIF(CTP(LP).EQ.'N') THEN
      GVAL(LK)=VAL(LP)
   ENDIF
15 CONTINUE
15 CONTINUE
L=L+1
IDIV=1
GOTO 5
20 REWIND 43
REWIND 55
RETURN
END
C--------------------------------|  
    SUBROUTINE EXIT  
C--------------------------------|
C
C---------------------------------|
C    Subroutine to terminate the OPSAMP system  
C---------------------------------|
C
COMMON/SWT/NCOD,ISW,IVAR  
NCOD=8  
RETURN  
END
Appendix III - 53

SUBROUTINE HEADER(FNAME,N,KCOD)

Subroutine to read the title of project and the variable names from a drillhole data file

LOGICAL ERROR, FIXIT
PARAMETER (MAXVAL=10)
COMMON/FLASH/ PSH
COMMON/PAR/ PST,HLINE
COMMON/OI/ IPT,IOUT
COMMON/SWT/ NCOD,ISW,IVAR
COMMON/FOM/ TFORM
COMMON/DATA/ HOLID(150),Y(150),X(150),Z(150),G(150),
+WT(150),NHOL
COMMON/COLOR/ CB1,CF1,CB2,CF2
COMMON/FORMAT/ FRM,LCHAR
NAMELIST/PRM1/JROW,JCOL,HLINE
NAMELIST/PRM2/JROW,JCOL,ITER,ISTP1
NAMELIST/PRM3/JROW,JCOL,ITER,ISTP2,CHAR1,NDIGT
DIMENSION JROW(5),JCOL(5),CHAR1(20),NDIGT(20)
IPT=21
CHARACTER*5 FRMT,CB1,CF1,CB2,CF2
CHARACTER*80 PST,PSH,CHAR1,HLINE
CHARACTER*1 BEEP,CR
CHARACTER*15 VARNAM(MAXVAL)
CHARACTER TFORM*50,HOLID*10,TITLE*12VARNAMS*75,CH*75
BEEP=CHAR(7)

C Open the template for displaying drillhole data

OPEN(UNIT=46,FILE='DATA. STR',STATUS='OLD')
READ(46,PRM1)
READ(46,PRM2)
READ(46,PRM3)
REWIND 46

C Open drillhole data file name

OPEN(IPT,FILE=FNAME,ACCESS='SEQUENTIAL',STATUS='UNKNOWN')
ERROR=.FALSE.
READ(IPT,100) tide,TFORM,MVAL,VARNAMS
100 FORMAT(A75/,A50/J2/,A75/)

CB1=' [34m'
CB2=' [42m'
IB=1
LCHAR=NDIGT(1)
CALL FRAME(JROW(2),JCOL(2),CB1,CB2,LCHAR,NDIGT,IB,ITER
+,ISTP1)

C return to the menu if incorrect a format data is read

C

C

C
IF(TFORM.EQ.' ') RETURN
IF(MVAL.LE.0 .OR. MVAL.GT.MAXVAL) RETURN
DO 40 I=1,MVAL
WRITE(VARNAM(I),200) I
CALL SET(VARNAM(I),VARNAM(I))
40 CONTINUE
IS=INDEX(VARNAMS,',')+1
IE=INDEX(VARNAMS,'')
IF(IS.LT.1.OR.IE.LT.1.OR.IE.LE.IS) GOTO 80
VARNAMS(IE:IE)=',';
JRW=JROW(3)
JCL=JCOL(3)
DO 60 I=1,MVAL
IC=IS-1+INDEX(VARNAMS(IS:IE),',')
IF(IC.EQ.0) RETURN
IF(IC-IS.LT.1) GOTO 50
VARNAM(I)=VARNAMS(IS:IC-1)
CH(1:4)=CHAR1(I)
CH(5:18)=VARNAM(I)
PST=CH(1:LEN(CHARNB(CH)))
CALL CTEXT(JRW,JCL,CB1,CB2,1,NDIGT(I))
JRW=JRW+ISTP2
50 IS=IC+1
IF(IC.GE.IE) GOTO 65
60 CONTINUE
65 PST='Enter Option [1,2,..
CALL CTEXT(JRW,JCL,' [42mV [34m', 1,23)
READ(*,*) IVARI
IF(KCOD.EQ.1) THEN
HLINE='Displaying variable of '
L1=LEN(CHARNB(HLINE))+2
L2=LEN(CHARNB(VARNAM(IVARI)))
HLINE(L1:L1+L2)=VARNAM(IVARI)
ENDIF
CALL REDAT(MVAL,IVARI,N,9999.)
NCOD=9
REWIND IPT
80 RETURN
200 FORMAT('VAR',IS)
END
SUBROUTINE SET(STR1,STR2)

The character texts setting

CHARACTER*15 STR1,STR2
CHARACTER*256 TEMP
IL1=LEN(STR1)
IL2=LEN(STR2)
I2=0
TEMP=''
IF(IL1.LT.1.OR.IL2.LT.1) GOTO 30
DO 10 I=1,IL1
  IF(STR1(I:I).EQ.' ') GOTO 10
  IF(I2+1.GT.IL2) GOTO 20
  I2=I2+1
  TEMP(I2:I2)=STR1(I:I)
10 CONTINUE
20 STR2=TEMP(1:IL2)
30 RETURN
END
SUBROUTINE REDAT(MVAL, ITYPE, N, RMIS)

COMMON/FOM/TFORM
COMMON/PAR/PST,HLINE
COMMON/IO/IPT, IOUT
COMMON/FORMAT/FRM, LCHAR
COMMON/DATA/HOLID(150), Y(150), X(150), Z(150), G(150), WT(150)
   +, NHOL
COMMON/LOC/JROW(10), JCOL(10), NDIGT(50), ISTP, ITER, NDAT
CHARACTER*10 FRM
CHARACTER*75 PST
CHARACTER TFORM*50, HOLID*10, HLINE*80
DIMENSION GVAR(10)

M = 1
5 READ(IPT, TFORM, END=40) HOLID(M), Y(M), X(M), Z(M), (GVAR(K),
   1K = 1, MVAL)
IF(GVAR(ITYPE).EQ.RMIS) GOTO 40
G(I) = GVAR(ITYPE)
M = M + 1
NDIGT(1) = 10
GOTO 5
40 N = M
   NHOL = N - 1
44 FORMAT(5X, 2F6.3, I2)
RETURN
END
SUBROUTINE FRAME(JRW,JCL,CB,CF,LC,NDIGT,IB,ITER,ISTEP)

DIMENSION CHARI(50),NDIGT(50)
COMMON/PAR/PST,HLINE
CHARACTER CHARI*80,HLIN*80,HLINE*80
CHARACTER PST*80,PSH*80,CB*5,CF*5
CHARACTER* 1 CUL,CUR,CDL,CDR,VLIN
DATA CUL,CUR,CDL,CDR,VLIN/* These symbols are not appeared here, see the program on the diskette provided */
DATA HLIN/*These symbols are not appeared here, see the program on the diskette provided*/
DATA PSH/* _________________________________________________
_________________________

LIN=LC+2
LHEAD=LEN(CHARNB(HLINE))
JR=JRW
JC=JCL
IL=IB
MAXLIN=0
DO 10 I=1,ITER
IF (MAXLIN.LT.NDIGT(IL)) THEN
MAXLIN=NDIGT(IL)
ENDIF
IL=IL+1
10 CONTINUE
15 CONTINUE
LIN=LIN+MAXLIN
MID=INT((LIN-LHEAD)/2)
PST(1:1)=CUL
PST(2:MID)=HLIN(1:MID)
PST(MID+1:MID+LHEAD)=HLINE
PST(MID+LHEAD+1:LIN-1)=HLIN(1:MID)
PST(LIN:LIN)=CUR
CALL CTEXT(JR-1,JC,CB,CF,1,LIN)
PST(1:1)=VLIN
PST(LIN:LIN)=VLIN
DO 18 I=1,LIN-1
PST(2:I)=" 
18 CONTINUE
DO 20 I=2,ITER*ISTEP+2
CALL CTEXT(JR,JC,CB,CF,1,LIN)
JR=JR+IB
20 CONTINUE
PST(1:1)=CDL
PST(2:LIN)=HLIN(1:LIN)
PST(LIN:LIN)=CDR
CALL CTEXT(JR,JC,CB,CF,1,LIN)
RETURN
END
SUBROUTINE KEYSTROKE(IRW, ICL, NCOD, IP)

Subroutine to redefine the keyboard functions

Symbol Description
----- ------------
IRW  Row number on the cursor position
ICL  Column number on the cursor position
NCOD Code number used for keyboard redefinition
IP   Identifier of character position

COMMON/PAR /PST,HLINE
COMMON ITP, ERR
COMMON/LOC/IROW(10), ICOL(10), NDIGT(50), ISTP, ITER, NDAT
COMMON/CHA/CH, IST
COMMON/COLOR/CB1, CF1, CB2, CF2
COMMON/FORMT/FRM, LCHAR
CHARACTER*1 KEY
CHARACTER*80 PST, HLINE
CHARACTER CH*80, FRM*10

IL=0
NW=0
NT=0
ILAG=1
ICOL=ICL
IROW=IRW
ITP=1
CALL CPOS(IROW, ICOL)
25 READ(40, '(A1)') KEY

Excepted characters for the key redefinition

IF (KEY.EQ.'P') GOTO 26
IF (KEY.EQ.'S') GOTO 26
IF (KEY.EQ.'M') GOTO 26
IF (KEY.EQ.'H') GOTO 26
IF (KEY.EQ.'K') GOTO 26
IF (KEY.EQ.'Q') GOTO 26
IF (KEY.EQ.'T') GOTO 26
GOTO 27
26 IL=1
GOTO 50
27 CONTINUE
Appendix III - 60

C Convert a character value to an integer equivalent representation

C

STRING=ICHAR(KEY)
IF(STRING.EQ.0.) THEN
READ(40,'(A1)') KEY
STRING=ICHAR(KEY)
ENDIF

C move cursor to the right
C

IF (STRING.EQ.77.) THEN
NCOD=3
GOTO 40
C

C Move cursor to the left
C

ELSEIF (STRING.EQ.75.) THEN
NCOD=4
GOTO 40
C

C Move cursor down
C

ELSEIF (STRING.EQ.80.) THEN
NCOD=2
GOTO 40
C

C Move cursor up
C

ELSEIF (STRING.EQ.72.) THEN
NCOD=1
GOTO 40
C

C Escape function used to quit the task
C

ELSEIF(STRING.EQ.27.) THEN
NCOD=6
GOTO 40
C

C Delete character at cursor position
C

ELSEIF(STRING.EQ.83..OR. STRING.EQ.8.) THEN
NCOD=7
KEY=''
IF(NT.EQ.0) THEN
NW=1
NT=1
GOTO 50
ENDIF
NW=1
IF(ITP.LT.1) THEN
ITP=ITP+1
ICOL=ICOL+1
ELSE
ITP=ITP-1
ICOL=ICOL-1
ENDIF
GOTO 50

C-----------------------------------
C Carriage-return key
C-----------------------------------
ELSEIF(STRNG.EQ.13.) THEN
NCOD=5
ITP=ITP-1
GOTO 40
ENDIF

50 PST(ITP:ITP)=KEY
CALL CPOS(IROW,ICOL)
CALL CTEXT(IROW,ICOL,CB1,CF1,ITP,ITP)
IF(ITP.LE.1.AND.NW.EQ.1) THEN
NW=0
ITP=1
ICOL=ICL
ILAG=1
IL=1
PRINT,CHAR(7)
GOTO 25
ELSEIF(ITP.EQ.NDIGT(IP).AND.NT.EQ.0) THEN
PRINT,CHAR(7)
IL=0
ILAG=0
GOTO 37
ENDIF

IF(NW.EQ.1) THEN
NW=0
GOTO 25
ENDIF
ILAG=1
IL=1

37 ITP=ITP+ILAG
ICOL=ICOL+IL
NT=0
GOTO 25

40 RETURN
END
FUNCTION GETVAL()

COMMON/PAR/ PST,HLINE
COMMON ITP, ERR
CHARACTER*80 PST
CHARACTER*9 FMR
CHARACTER*9 FMA

KPOS=INDEX(PST(1:ITP),'.')
GETVAL=0.
KL=ITP
IF(KPOS.EQ.0) THEN

FMR='(I )'
IF(KL.LT.10) WRITE(FMR(4:4),'(I1)') KL
IF(KL.GE.10) WRITE(FMR(3:4),'(I2)') KL
READ(PST(1:KL),FMR,ERR=900) IVAL
GETVAL=IVAL
ELSE

FMA='(F )'
IF(KL.LT.10) WRITE(FMA(3:5),'(I1,".",I1)') KL,KL-KPOS
IF(KL.GE.10) WRITE(FMA(3:6),'(I2,".",I1)') KL,KL-KPOS
READ(PST(1:KL),FMA,ERR=900) DVAL
GETVAL=DVAL
ENDIF
RETURN

900 PRINT, char(7)
ERR=1
RETURN
END
C-----------------------------------------------------------------------------------------------|

SUBROUTINE WRTDAT(IROW,ICOL,CB1,CF1,KL,TVAL)
C-----------------------------------------------------------------------------------------------|
C
C-----------------------------------------------------------------------------------------------|

Subroutine to write data onto screen
C-----------------------------------------------------------------------------------------------|

COMMON/PAR/PST,HLINE
COMMON/FORMT/FRM,LCHAR
CHARACTER*5 CB1,CF1
CHARACTER*10 FRM
CHARACTER*80 PST,HLINE
PST=PST(1:KL)
WRITE(PST,FRM) TVAL
IF(I.LE.9.AND.J.LE.9) THEN
    WRITE(POS,'(A2,I1,A1,I1,A1)') '[I;',':',J,']
    PRINT,CLRB,CLRF,POS(L1 :L2),' [b'
ELSEIF(I.LE.9.AND.J.GT.9) THEN
    WRITE(POS,'(A2,I2,A1,12,A1)') '[M/J
    PRINT,CLRB,CLRF,POS(L1 :7),PST(L1 :L2),' [b'
ELSEIF(I.GT.9.AND.J.GT.9) THEN
    WRITE(POS,'(A2,I2,A1,11,A1)') '[,,I,,;,,J,,f
    PRINT,CLRB,CLRF,POS(L1 :8),PST(L1 :L2),' [b'
ELSEIF(I.GT.9.AND J.LE.9) THEN
    WRITE(POS,'(A2,I2,A1,11,A1)') '[,,I,,;,,J,,f
    PRINT,CLRB,CLRF,POS(L1 :7),PST(L1 :L2),' [b'
ENDIF
RETURN
END
SUBROUTINE CPOS(JROW,JCOLUMN)

C Subroutine to locate the cursor on the screen

COMMON/PAR/PST,HLINE
CHARACTER*5 FRMT
CHARACTER*80 PST,HLINE
CHARACTER*80 POS

IF(JROW.LE.9) THEN
  WRITE(POS,'(A2,I1,A1,I2,A1)') '['JROW,':'JCOLUMN,']'
  PRINT ,POS(1:7)
ELSE
  WRITE(POS,'(A2,I2,A1,I2,A1)') '['JROW,':'JCOLUMN,']'
  PRINT ,POS(1:8)
ENDIF
RETURN
END
SUBROUTINE CTEXT(I,J,CLRB,CLRF,L1,L2)
COMMON/PAR/PST,HLINE
CHARACTER*80 POS,PST,HLINE
CHARACTER*5 CLRB,CLRF
IF(I.LE.9.AND.J.LE.9) THEN
   WRITE(POS,'(A2,Il,AUl,Al)') ' ',I,'','';J,'f'
   PRINT,CLRB,CLRF,POS(1:6),PST(L1:L2),'[0m'
ELSEIF(I.LE.9.AND.J.GT.9) THEN
   WRITE(POS,'(A2,I1,A1,12,A1)') ' ',J,'','';I,'f'
   PRINT,CLRB,CLRF,POS(1:7),PST(L1:L2),'[0m'
ELSEIF(I.GT.9.AND.J.GT.9) THEN
   WRITE(POS,'(A2,I2,A1,11,A1)') ' ',J,'','';I,'f'
   PRINT,CLRB,CLRF,POS(1:8),PST(L1:L2),'[0m'
ELSEIF(I.GT.9.AND.J.LT.9) THEN
   WRITE(POS,'(A2,I2,A1,11,A1)') ' ',I,'','';J,'f'
   PRINT,CLRB,CLRF,POS(1:7),PST(L1:L2),'[0m'
ENDIF
RETURN
SUBROUTINE TREE(NDAT)

This subroutine searches the samples using the Branch and Bound Method

Symbol | Description
---|---
KSET | Dummy variable
A1[20,41] | Inversed kriging matrix
C[20,41] | Kriging covariance matrix
A[20000] | Global covariance matrix
X[20,41] | Buffer variable
JSET[150] | Variable containing dropped sample
LSET[150] | Variable containing undropped sample
NSAMP[150] | Number of samples
IADD | Number of additional samples

DIMENSION KSET(150)
COMMON/MAT/ Y(20),X(20,41)
COMMON/KBUF/A1(20,41),C(20,41)
COMMON/ESBUF/A(20000),AA(20000)
COMMON/UPDRP/JSET(150),LSET(150),LST(150),KW
COMMON/UPDEL/NDROP,NLEFT,TV(150),S,NDUM
COMMON/POS / IPOS(150),NDP,NSAM(150)
COMMON/CALL/NTYPE,NW,KCOD,IADD,NOPT
COMMON/WEIGH/RLAMB(151),RX(151),LEFHL(30,30),RL,ND
COMMON/KRG/IHOL(151),GA(151),R(151),DIST(151)
COMMON/DATA/HOLID(150),YN(150),XE(150),ZL(150),GR(150),+WT(150),NH
COMMON/KPAR/VKRG,BLKVAR,TVAR(150)
COMMON/ESTVAR/PTBLOC,COBLK,CVBLK,COSAM,CSM,+
SAMBLK
COMMON/BLOCK/DX(150),DY(150),DYZ(150),YN(150),+
XEA(150),VOL,NVOL,NDX,NDY,NDZ,NPOINTS,M
COMMON/SWICTH/MSW
CHARACTER HOLED*10
IF(IADD.LE.1) RETURN
KM=0
KW=0
NDUM=0
NOPT=20
IF(KCOD.EQ.2) THEN
NDA=NDAT-1
ELSEIF(KCOD.EQ.1) THEN
NDA=NDAT
ENDIF
DO 5 I=1,NDAT
NSAM(I)=I
5 CONTINUE
C-----------------------------------------------|
C Fill the partitioned block matrix
C-----------------------------------------------|
CALL REFILL(NDAT,KCOD,KW)
7 FORMAT(2X,12(1X,E10.3))
C------------------------------------
C Set variables to zero
C-----------------------------------
8 FORMAT(A75)
20 CONTINUE
DO 15 I=1,NDAT
KSET(I)=0
LSET(I)=0
JSET(I)=0
15 CONTINUE
K=0
NT=0
IL=0
NDROP=0
JS=0
NLEFT=NDAT
CALL SUMMA(0,KCOD,NW,IADD,0)
C-------------------------------------------------------------------
C Check n dropping samples satisfying the constraints
C-------------------------------------------------------------------
25 IF (NDROP.EQ.IADD-1) THEN
NW=1
GOTO 30
ELSEIF (NT.EQ.IADD.OR.JS.EQ.IADD) THEN
30 K=0
C---------------------------------1
C Add a sampled
C---------------------------------1
NT=JS
NDROP=NDROP-1
IF(NT.EQ.IADD) THEN
NDROP=0
JS=0
IL=IL+1
NT=IL
CALL REFILL(NDAT,KCOD,KW)
GOTO 25
ENDIF
GOTO 55
ENDIF
K=1
C---------------------------------1
C Drop a sample
C---------------------------------1
NT=NT+1
C JS=NT
The setting combination - NOPT in order to speed up the search process

```
C---------------------------------------------------------------
C IF(NDROP.ge.NOPT) THEN
NW=1
GOTO 25
ENDIF
JS=NT
NDROP=NDROP+1
KSET(NDROP)=NT
55 N=1
IN=1
NK=1
31 IF(NK.GT.NDA) GOTO 35
IF(KSET(IN).NE.NSAM(NK)) THEN
IF(KCOD.EQ.1) LSET(N)=NSAM(NK)
ELSE
IF(IN.EQ.NDROP) GOTO 32
ENDIF
ENDIF
32 NK=NK+1
GOTO 31
35 N=N+1
62 IF(NDROP.EQ.0) THEN
C---------------------------------------------------------------
C Refill partitioned block matrix X(I,J)
CALL REFILL(NDAT,KCOD,KW)
GOTO 25
ENDIF
57 DO 70 I=1,NDROP
J SET(I)=KSET(I)
CONTINUE
C---------------------------------------------------------------
C If KCOD=1, run ZVAR for the updating of the ESVAR program
C KCOD=2, run KVAR for updating of the SKRIG program
C---------------------------------------------------------------
C IF(KCOD.EQ.1) THEN
CALL ZVAR(JM,K)
ELSE
C Rearrange the position of remaining samples by adding NLEFT+1
C---------------------------------------------------------------
DO 23 I=2,NLEFT
LSET(I)=LSET(I)+1
CALL KVAR(JM,K)
ENDIF
JS=KSET(NDROP)
24 IF(K.NE.0) THEN
IF(NDROP.EQ.1.AND.JS.EQ.IADD) GOTO 33
```
ENDIF
GOTO 25
33 IF(KM.EQ.0) THEN
   KM=1
C--- Call SUMMA to print the results of sampling analysis
C---
   CALL SUMMA(NTYPE,1,1,IADD,NOPT)
ENDIF
77 FORMAT(25X,'VALIDATION OF PROGRAMME',//,
   15X,'-----------------------------------------------'//,
   16X,'DROPPED HOLES UNDROPPED HOLES',//,
   15X,'-----------------------------------------------'//)
78 FORMAT(5X,'------------------------------------------------------')
   VKRG=0.0
RETURN
END

SUBROUTINE REFILL
---------------------------------------
SUBROUTINE TO REFILL THE MATRIX A(IJ)
---------------------------------------
SUBROUTINE REFILL(NDAT,KCOD,KW)
COMMON/WEIGHT/RLAMB(151),RX(151),LEFHL(30,30),RL,NDA
COMMON/DATA/HOLID(150),YN(150),XE(150),ZL(150),GR(150),
   WT(150),NHOL
COMMON/KBUF/A1(20,41),C(20,41)
COMMON/MAT/Y(20),X(20,41)
COMMON/KPAR/VKRG,BLVAR,TVAR(150)
COMMON/KRG/HOL(151),GA(151),R(151),DIST(151)
COMMON/ESTVAR/PTBLOC,COBLK,CVBLK,COSAM,CSM,SAMBLK
CHARACTER HOLID*10
IF(KCOD.EQ.2) THEN
   DO 10 I=1,NDAT
      RX(I)=R(I)
      WRITE(33,20) (C(I,K),K=1,NDAT),R(I)
   DO 10 J=1,NDAT
      X(I,J)=A1(I,J)
   10 CONTINUE
ELSE
   CVBLK=COBLK
   CSM=COSAM*NHOL*NHOL
ENDIF
KW=1
20 FORMAT(1X,10F10.7)
RETURN
END