UNIVERSITY OF ALBERTA

DRAGLINE PIT DESIGN AND SENSITIVITY ANALYSIS

by

ROBERT WILLIAM HAMILTON ©

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF

MASTER OF SCIENCE

IN

MINING ENGINEERING

DEPARTMENT OF

MINING, METALLURGICAL AND PETROLEUM ENGINEERING

EDMONTON, ALBERTA

FALL, 1990

ISBN 0-315-65105-9

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR:	ROBERT WILLIAM HAMILTON
TITLE OF THESIS:	DRAGLINE PIT DESIGN AND SENSITIVITY ANALYSIS
DEGREE:	MASTER OF SCIENCE
YEAR THIS DEGREE GRANTED:	FALL, 1990

(SIGNED) [illegible]

PERMANENT ADDRESS:

Box 1533

Tumbler Ridge, B.C.

V0C 2W0

Date Oct. 10, 1990

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled **DRAGLINE PIT DESIGN AND SENSITIVITY ANALYSIS** submitted by Robert William Hamilton in partial fulfilment of the requirements for the degree of Master of Science in Mining Engineering.

[illegible]
Prof. W. Griffin
Supervisor

[illegible]
Dr. K. Barron

[illegible]
Dr. J.M. Whiting

[illegible]
Prof. A. Peterson

Date Oct. 5, 1990

Abstract

This thesis describes a dragline pit design computer program developed to assist the Obed Mountain Coal Company's mine planning. This program draws range diagrams for most simple cases of one and two seam operations. One model, REH2, incorporates many site-specific operating procedures for the case when two seams are uncovered, and rehandle is introduced in the form of an extended bench. This may restrict the model's applicability at other mines, but a custom-made program is necessary to correctly model the dragline operation. The program, DRAG, was written in Fortran 77 and uses a plotting library, called Plot88, to produce graphical output. Each part of the program is run using menus. The user is protected from some poor assumptions because the program checks for invalid responses. Program input comes from several data files, so users can customize the applications to suit their needs by simply changing the data values.

A Taylor series approximation technique is applied in the form of sensitivity analysis. This technique offers additional information when compared to other sensitivity analysis techniques, because it allows the us to determine how much influence individual variables have on a complex function.

Acknowledgements

I would like to extend my gratitude to those people who assisted me over the course of this study. I am grateful to Professor Wayne Griffin for providing the supervision of the research for this thesis. His technical assistance and continued encouragement have been deeply appreciated.

A special thanks also to Dr. Ken Barron for his supervision during the absence of Professor Griffin. His many suggestions proved to be invaluable towards the successful completion of my thesis.

Appreciation is extended to Chuck Williams of Obed Mountain Coal Co. Ltd. for supporting this project's ideas and discussing the many details in the development of the dragline program.

I also would like to thank Mr. Doug Booth for answering all of my distress calls when the computers had me cornered. The fellow students in room 280 also made life more fun when progress was minimal.

Finally, I would like to thank my wife, Cathy, for her continued support and patience during the many months of this study. She made the difficult times bearable and the completion of this thesis possible.

Table of Contents

Page

Page

List of Tables

Page

List of Figures

Page

Page

Nomenclature

θ	angle of repose of overburden and interburden (degrees)
β	overburden highwall angle (degrees)
α	interburden highwall angle (degrees)
γ	cut angle in spoil (degrees)
μ	population mean
σ	standard deviation
X_i	random variable
VOB	bank overburden volume (bcm)
VOBSP	loose overburden volume (lcm)
VIB	bank interburden volume (bcm)
VIBSP	loose interburden volume (lcm)
VCHOP	bank chopcut volume (bcm)
VSPOIL	total spoil volume (lcm)
OROB	dragline operating radius from overburden bench (m)
ORIB	dragline operating radius from interburden bench (m)
H	dragline stacking height (m)
ET	width of extended bench (m)
RVOL	rehandle volume (lcm)
EVOL	extended bench volume (lcm)
VOLMAX	maximum overburden volume capable of being spoiled by dragline (lcm)

1. Introduction

1.1 Dragline Planning

Strip mining is a mining method where the overlying waste is moved a short distance to one side to allow excavating equipment to begin removal of the coal. Due to their high production rates and versatility, draglines are generally the primary excavating unit used in a strip mining application. Since the late sixties, draglines have been increasing in size to provide more overburden removal capability and to lower the unit mining cost. With the increase in size, draglines often become the highest capital investment item at a strip mine and also make up a large portion of the operating cost. Therefore the operating procedures of the dragline deserve considerable investigation. The overall success of a strip mining operation depends in a large part upon effective utilization of the dragline to remove overburden.

Dragline pit design is of considerable importance at a strip mine since coal production is directly affected by the selection of stripping methods and pit geometries. Dragline production paces all other operations in a strip mine, regardless of whether they occur before or after the dragline has excavated its cut.

The planning of dragline operations is very well suited to computer application due to the repetition of geometric calculations. Dragline mining operations often use computer programs to automate the preparation of range diagrams and volumetric calculations.

Every single or multiple seam operation is a unique combination of natural conditions, type of overburden materials and equipment available to do the job. The operating methods that work well at one mine may not necessarily work at another location. Complex operational sequencing is often required to optimize machine performance when box cuts, multiple seams, tandem draglines, pitching seams and/or very deep digging conditions are involved.

1.2 Problems Faced by the Obed Mountain Coal Co. Ltd.

The Obed Mountain Coal Co. Ltd. currently mines two main horizontal coal seams with a single dragline. The dragline is equipped with a 57 cubic metre bucket and an 80 metre boom inclined at 33 degrees to the horizontal. The mine began production in 1984 and it now produces approximately one and one-half million tonnes of clean coal per year. Most of the dragline pit design to date has been done manually by on site engineers. These engineers have difficulty in analyzing alternate pit designs due to the amount of time required to produce the requisite range diagrams. Deeper overburden and higher coal production requirements have made it important for them to look at automating their range diagram generation through the use of computers. As with many strip mines, unique operating conditions are present at Obed and therefore a customized program would improve the application to their operation.

1.3 Objective of this Study

The objective of this study was to write a selection of interactive programs to produce dragline range diagrams in accordance with Obed's current equipment and operating methods for both one and two seams. Various methods of looking at the sensitivity of input parameters upon calculations were also considered so that appropriate methods of sensitivity analysis could be recommended.

1.4 Analysis of the Problem

Although there are many programs available to generate range diagrams for strip mines, a considerable number of site specific assumptions must be made when writing a program of this nature; therefore it is difficult to find a model with all of the desired assumptions. For example, there are an infinite number of ways to rehandle spoil when a dragline has inadequate reach or height. This problem is magnified when multiple seams are being considered. The resulting programs of this study represent the existing operating methods at Obed and they were set up so that an inexperienced computer user could effectively use them on a personal computer as a planning tool.

Although many powerful tools are available to mine planners, often little is done to test models for sensitivity. Input values to models are often assumed to have no variability and this is also reflected in the calculated result. When the variability of the pit design parameters is defined in an analysis, the true variance of a calculated result can be approximated. This can be very useful because if an approximated variance is too large to be considered acceptable, it may be determined how much each pit design variable is contributing to the approximated variance and then measures can be undertaken to better define one or more of the variables.

1.5 Thesis Layout

Chapter 2 gives a general description of some dragline methods which are being used in strip mines. Problems associated with these methods will also be discussed.

Chapter 3 provides a general overview of how the dragline models are set up in their respective subroutines within the main program DRAG. Computer implementation of the models will be described in detail. A brief discussion will explain some assumptions that are common to all of the models.

Chapters 4 and 5 discuss the models that have been developed for one and two seam mining methods. Specific assumptions and formulas will also be explained.

Chapter 6 looks at some of the different probabilistic analysis techniques that have been documented in the literature. The applicability of these techniques to dragline mining will be considered as well as methods that have already been applied.

The sensitivity analysis library, VARSIM, is applied in chapter 7 to the selection of an adequate dragline operating radius for a single seam mine.

Using the final models of range diagrams and sensitivity analysis for each application, recommendations for further application will be discussed in chapter 8. Conclusions of the study will also be presented in this chapter.

Appendix A contains a complete listing of the source code for the range diagram program DRAG while its sample data files are documented in appendix B. A complete listing of the source code for VARSIM is located in appendix C. Appendix D consists of descriptions of all the possible pit configurations which may be generated by the two seam extended bench model used in subroutine REH2. Appendix E contains a listing of the function REC used by the VARSIM library in chapter 7.

Tables, figures and equations are numbered consecutively using the chapter numbers. In the equations, a multiplication sign is expressed by a dot rather than a star. This is only done to make the equations more easy to read and should not be assumed to be the correct syntax for a programming language. References are numbered in alphabetic order with only the numbers enclosed in square brackets being used within the text and appendices of the thesis.

2. Dragline Mining Methods

2.1 Simple Sidecasting

The simplest and most common stripping method is sidecasting. Sidecasting involves having the dragline sit on the overburden, dig down to the coal and then spoil into a pile on the side.[8] A plan and cross section view of the sidecasting method is shown in Figure 2.1. Simple sidecasting does not have any rehandle because the dragline has adequate reach and stacking capability to spoil in the available spoil area. For single seam operations, the dragline capability is almost always constrained by dumping radius rather than dumping height. Most other stripping methods are modified versions of the simple sidecasting method.

The material in this figure has been removed
because of the unavailability of copyright permission

Figure 2.1 Cross-section of the sidecasting method (after [8])

At the start of a stripping operation, the dragline first makes a box cut opening as shown in Figure 2.2. The box cut is generally made along the coal seam strike line.[29] Some operators have tried opening the box cut perpendicular to the strike line of the coal [11], but most have found this alternative economically unattractive. Only a single box cut should be necessary as later cuts will always have an opening in which to spoil. A case where more box cuts are required is sometimes encountered when the ground is undulating and the coal outcrops in different areas.

The material in this figure has been removed
because of the unavailability of copyright permission

Figure 2.2 Box cut (after [8])

Once the first cut has been established, the dragline is positioned at the inside limit of the second cut in order to make the initial key cut. From this position, the dragline can establish a competent new highwall slope. The key cut also provides another free face for digging the remaining material from the cut. The key cut is usually one to two buckets wide at the bottom. Figure 2.3 shows the positions of the dragline during key cut removal.

Figure 2.3 Key cut diagram (after [14])

After the key cut is made, the dragline moves closer to the edge of the highwall to remove the rest of the material. The remaining material is removed from the cut segment from a number of intermediate positions as shown in Figure 2.4. These additional positions allow the operator to utilize maximum reach for the spoiling of the material. After all the material is spoiled into the previous cut, the dragline walks to the next key cut position and repeats the cycle.[8]

Figure 2.4 General dragline digging positions (after [8])

2.2 Advanced Bench

The advanced bench (or chopcut) method is often used when surface material is unsuitable for dragline support. This method involves chopping material above the dragline operating bench either directly behind the dragline, or from the next pit over and then spoiling into the empty spoil pit. Chopping allows the dragline operating bench to be prepared at an elevation lower than the original ground surface. As a relatively level operating bench is required by the dragline, advanced benching also reduces the amount of fill required when hilly terrain is encountered and helps prepare drill pads for safe and efficient drilling. A well-graded flat bench reduces the potential for high local loading of the bottom of the dragline tub and minimizes distortion of the machine structure.[40]

The dragline's effective reach can be increased by operating at a lower bench elevation due to the effect of the highwall angle. This may reduce or eliminate rehandle in stripping operations where dragline reach is not quite sufficient to spoil the overburden.[14] Figure 2.5 shows the advanced bench and the lower dragline operating elevation.

The material in this figure has been removed
because of the unavailability of copyright permission

Figure 2.5 Advanced bench (after [14])

The chopcut process is relatively inefficient when compared to normal digging and this will be further explained in section 2.6.3. Heavier wear on the dragline and poorer digging production make it necessary to determine an optimum bench elevation which takes advantage of pad preparation and reach extension while still considering the disadvantages. Local conditions may also allow other methcds or equipment to remove the chopcut material more cost effectively.

2.3 Extended Bench

The extended bench method can be utilized to increase the dragline operating radius when the overburden volume increases to the point where the dragline reach is insufficient. Material from the key cut and sometimes the chopcut is used to extend the working bench out from the highwall crest and into the previous pit. The extended bench width is equal to the required operating radius less the actual dragline operating radius.[13] Figure 2.6 shows an extended bench and how it not only extends the reach of the dragline, but also introduces rehandle material at the same time. The extended bench material is levelled by a dozer to form a level dragline pad. As the overburden depth increases, the amount of rehandle material will also increase in extended bench operations. An added benefit of using the extended bench method is that blasting is not inhibited by the need to preserve the old highwall.[42]

Figure 2.6 Extended bench (after [24])

Reduced rehandling may also be possible if compaction of the extended bench material makes it possible to over-steepen the spoil pile angle during the rehandle operation.[10] The degree of over-steepening that can be used is primarily dependent upon the spoil characteristics, the spoil handling method and the extended bench height. Instances of over-steepening up to 80 degrees from the horizontal have been observed in northern Appalachia.[29] The spoil over-steepening concept is illustrated in Figure 2.7.

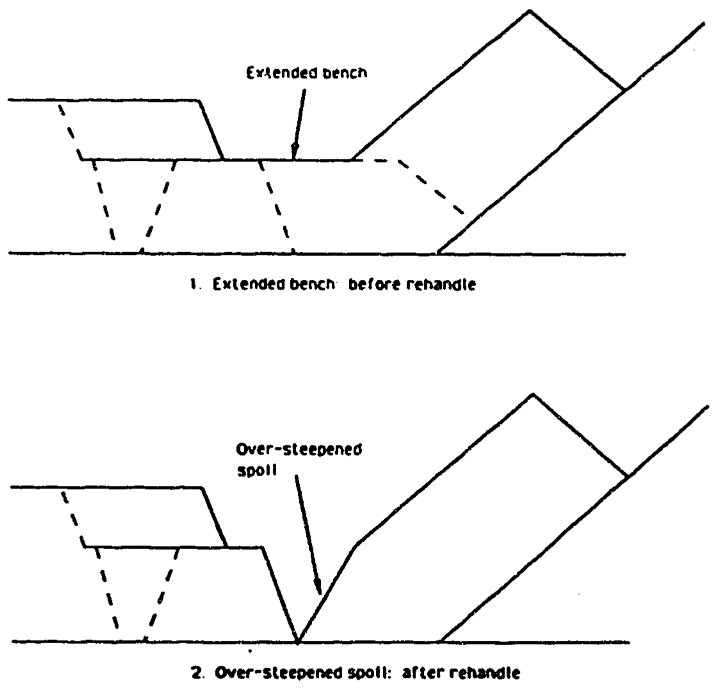

Extended bench
1. Extended bench before rehandle
Over-steepened spoil
2. Over-steepened spoil: after rehandle

Figure 2.7 Spoil over-steepening (after [29])

A combination of the extended bench method and the advanced bench method can be used so that as the depth of the chopcut bench increases, the amount of rehandle decreases. This combination works well in deep overburden and mine planners try to find the bench elevation which works most effectively at their location. If a deep bench is selected, the dragline dumping height may also be an important operating constraint. An example of sequencing the extended and advanced bench method is given below.

(a)

(d)

(b)

(e)

(c)

(f)

Figure 2.8 Sequencing for an extended bench with an advanced bench (after [22])

Another method of extending the dragline reach is to use a split bench method where two passes are used to remove the overburden. In this method, the elevation of the first pass is either the surface topography or a shallow bench while the second pass occurs at a bench elevation which has been excavated to on the first pass. This method is preferred less than the extended bench method because it only uncovers coal on the second pass of the dragline.

2.4 Spoil Side Pull Back

The pull back method is used when the dragline operating radius is inadequate. This method can use either a single dragline or multiple draglines. A dragline first spoils the overburden using either the advanced bench or sidecasting method. Next, a pad is prepared on the spoil side by dozers so that a dragline can pull back the excess spoil away from the highwall and spoil behind the dragline as in Figure 2.9.

Figure 2.9 Spoil side pull back method (after [14])

Dragline bench stability is always a major concern in a dragline operation, but it becomes more of a concern when the dragline is operating on the spoil side because the bench is no longer virgin ground and material properties are very unpredictable. The extended bench method would be better at a location where spoil piles have poor stability. A detailed stability analysis should be undertaken prior to attempting the pull back type of operation. This method has been used effectively in some operations [12], but it is not being used at the Obed mine so it will not be addressed further.

2.5 Multiple Seam Methods

The multiple seam methods considered will be restricted to two different two seam applications. The first two seam method uses direct sidecasting from both the overburden and interburden levels with an option to use an advanced bench if desired. It is assumed that there is no rehandle in this method. This is an idealized method because there is always some rehandle involved in multiple seam dragline mining, however, this method is a good place to begin planning when a feel for possible operating constraints is desired. It is always much easier to consider alternatives when no rehandle is required because of the large number of ways that rehandle material can be moved.

The second method uses sidecasting of the overburden by a dragline operating on the overburden. Once again, a chopcut bench may be cut. The dragline completes the overburden cut, ramps down to the interburden elevation, and then begins spoiling from the interburden. A safety bench is usually used at the toe of the overburden highwall to provide a safer highwall as well as more operating room for the dragline. When the key cut is being cut on the interburden bench, the safety bench can provide adequate room so that the dragline house will not hit the overburden highwall during rotation of the boom to cast the key cut spoil.[29] Dozers can be used to make the interburden key cut if operating room is restricted. Dozers simply push the interburden material down into the pit. An extended bench is used on the interburden bench if it is needed. When operating on the interburden bench, the dragline dumping height becomes an important operating constraint. A way that is used to reduce the spoil height is to spoil from two or more positions on the extended bench so that two or more spoil peaks are formed. This may require cutting into the spoil piles, spoiling, and then walking the dragline away from the spoil pile and spoiling again. The first position near the spoil may be covered with spoil after all the spoiling is accomplished. A method similar to this [50] is utilized at the Obed mine and will be further described in chapter 5.

2.6 Problems Associated with Dragline Mining

In addition to selecting a dragline operating method, there are many natural and operational conditions which must be addressed when planning the operation of a dragline. Some of these are discussed in the following sections.

2.6.1 Sequencing

A serious problem in dragline mining is the scheduling of drilling, blasting, stripping and coal removal. All mining functions must revolve around the scheduling of the stripping equipment. Scheduling of the different operations is generally easier if a wider pit is used because it takes longer to mine the larger sized pit.

Topsoil removal is generally the first operation on a strip. The dragline will often pile the topsoil into a stockpile for later removal by truck and shovel. Some strip mines remove the topsoil with scrapers and haul it directly to areas being reclaimed.

Drilling and blasting is required when the overburden material is too difficult for the dragline to dig. Drilling should be kept well in advance of the dragline so that should a drill breakdown occur, adequate time would be available for repairs without running out of blasted material. Very few mines also have to blast the coal seams so that coal loading equipment may load the haul trucks.

The coal removal cycle should be planned so that the coal at the end of the strip is removed as soon as possible after the dragline has finished excavating the overburden in the strip. This may require leaving exposed coal farther down the pit for later removal, but lost dragline production is not usually tolerated so it becomes a top priority not to interfere with its operation. For a two seam operation, it is necessary for the dragline to ramp down to the interburden level. Sump cleanup is also necessary to ensure continued pit drainage. With a dragline ramp, a drainage sump and possibly an end of pit haul road entrance, significant rehandle is often required to maintain each in functional form. Spoil area is very limited at the end of a pit with these constraints, so detailed planning should be carried out well in advance. Site specific materials handling usually results at the end of the pit. In order to minimize the lost dragline operating time at the end of the pit, preventative maintenance is often scheduled for the dragline to allow the coal loading equipment to catch up.

When conditions permit, the regular spacing of pit entrances from main roads will provide for pit hauls with one way traffic. As overburden gets deeper, it becomes more difficult to keep coal haulage inclines open. As each strip passes a ramp entry, the spoil that would have gone into the area where the ramp is located must be dumped to either side of the ramp. This may require a larger extended bench and the effects of this may extend for a considerable distance on each side of the ramp.[7]

Power cable moves should be planned in advance of actual dragline moves. A simple way of avoiding dragline down time is to use traffic bridges to allow vehicles to cross the dragline cable without causing undue damage.

2.6.2 Width and length of dragline pits

A choice of pit width is only available when a dragline is stripping well below its maximum capability. The choice of pit width must take into account the effect on subsequent strips as the spoil area available for the next strip is determined by the current strip dimensions. The pit width is primarily dictated by the minimum amount of room needed for operation of coal loading and haulage equipment[29] and the need to cast key cut material over the old highwall.[42] The allowable rehandle is also a constraint and while rehandle is usually avoided if possible, the cutoff amount allowed to be rehandled is an engineering decision unique to each mine and the method being used. A wider pit generally reduces the total amount of dragline walking and produces a safer working environment by allowing better maneuverability of coal loading equipment within the pit as well as less operation close to the highwall.[9,42] If deadheading is used in the operating method, a wider pit will result in less deadheading over the life of the mine.

A deep pit may benefit from extra pit width when groundwater and spoil instability pose potential operating problems. As overburden becomes deeper, maintenance of the open pit becomes more expensive so it may become more desirable to use shorter pit lengths. This is especially true when frequent slope failures occur. When pit lengths are short, a wide pit may be needed to provide the required inventory of exposed coal.[14]

Up to a certain point, a narrow pit will reduce dragline cycle times due to smaller swing angles. However, if a pit is too narrow, productivity is not gained because the hoisting of the loaded bucket becomes the limiting component of the cycle time.

With wide pits, spoil piles are high and widely spaced whereas with narrow pits, they are lower and more closely spaced. High spoil piles result in higher grading costs. In uniform overburden depths, the spoil crests will be the same distance apart as the pit width.[42] The large height of the spoil peaks means that not only must more material be dozed to restore a flat topography, but the material must be dozed farther.

Curved pits result when the overburden depth to be stripped increases or decreases nonuniformly and the dragline operator tries to make an equal sized spoil pile from unequal volumes of overburden along the cut. Curved pits may also result when an area of low strip ratio coal is available and operators decide to excavate it. The inside curve has less spoil area while the outside curve has more spoil area. In this context, an inside curve is considered as a curve where the dragline is spoiling onto the side of the pit which creates an angle less than 180 degrees. The outside curve is a good location for an entrance into the pit.[8] One way to gradually straighten a curved pit is to widen the pit at the outside curves and narrow it at inside curves.[29]

2.6.3 Operating efficiencies

Dragline digging efficiency is very dependent upon how well the material has been fragmented by blasting as well as operator competency. Digging efficiency is reduced and machine abuse is increased when the dragline digs above itself (chopdown). It is difficult to control the bucket penetration into the material face.[40] The chopdown cycle time is forty to fifty percent longer than normal cut excavation.[26] On the other hand, rehandle material is dug more easily than the prime overburden because it has swelled due to prior handling as well as being fragmented by the blast.[42] An operating efficiency of eighty-five percent can be used for excavating virgin material while ninety-five percent can be used for rehandle material. This is because the operator can not obtain a full bucket on every pass and it is more difficult to obtain a high bucket fill factor in virgin material such as chopcut material.[26]

One way to improve operator efficiency is to obtain more information on actual machine performance by using a recording device such as the General Electric Data Logger. This device is designed to collect continuous information for several basic quantities such as walk time, swing time, load time, etc..[26] An operator console is provided for inserting information on operational delays. Feedback displays show current or accumulated performance figures. All functions can be monitored and permanently recorded. McDonnell Douglas Electronics has even developed a sophisticated training unit to upgrade dragline operator training. This includes a closed circuit television system to provide simulation training similar to that applied in the aircraft industry.[28]

Cleaning the top of the coal seam with the dragline bucket is sometimes done very well, but doing so reduces dragline production. Auxiliary equipment such as graders and dozers should be used to clean the top of the seam because they can do it more economically.[8]

2.6.4 Digout length

The digout is a block of overburden that is excavated from a given dragline position. Dragline blocks (or digouts) are normally twenty to thirty meters long in the strip direction.[27] Making the digout longer decreases nonproductive walking time per pit. This should improve dragline production per pit, however, sometimes an increase of the distance over which the loaded bucket has to be dragged prior to hoisting will decrease production. A wider pit will also reduce the number of digouts over a given time interval. Even though advantages and disadvantages are known for the dimensions of digouts and pits, it is difficult to arrive at good decisions by making simultaneous tradeoffs of the advantages and disadvantages of each decision variable.[29]

2.6.5 Coal losses

Coal losses in a dragline pit may be due to truck spillage, poor recovery at the top and bottom of the coal seam or difficult access to load coal safely or economically. At many mines in the western U.S., a coal wedge (also called a coal fender or a coal rib) is left at the toe of the spoil. It is left either to support the spoil toe or because loading is difficult near the spoil.[29,46] A coal wedge is caused when spoil is allowed to ride up the coal edge. This can be the result of poor operating practice or simply insufficient dragline reach. Wider pits mean fewer wedges and fewer wedges mean better resource recovery.[8,14]

2.6.6 Slope stability and drainage

Effective spoil pile stability generally results from control of surface water.[27]

There is always some runoff water that escapes the collection system and finds its way into the pit. Also, seepage from the highwall and rainfall in the spoil area collect in lows on the pit bottom.[48] Inadequate pit drainage can reduce the overall operating efficiency of equipment in the pit. If much rainfall occurs at a location, additional width should be allowed for drainage ditches within the pit.[29] It may be required to clean out mud and water from the previous pit floor before spoiling so that the base of the new spoil pile can be constructed onto a sound foundation to minimize spoil failure possibilities.[7]

Drainage sumps are often located at the downdip end of the pit so that gravity can be used to move the water. Drainage pumps are usually located in the sumps so that the water can be pumped beyond the end spoil pile if necessary.

Highwall and spoil stability problems exist in the glaciated areas of Illinois and North Dakota due to the unconsolidated nature of the cverburden material and a high water table.[29] If seepage into an existing pit is excessive, the height of the undisturbed groundwater level above the pit bottom should be determined. Then the normal draw down curve, as shown in Figure 2.10, should be lowered by implementing a new dewatering program. Overburden strips are partially drained by natural outflow at the highwall face. Pumps may be used to produce draw down zones around the pumped areas. Since coal seams usually have higher permeability than overburden, pumps are usually located at or close to the bottom of the seam as shown in Figure 2.10.

The extended bench method provides better stabilization of the spoil slope due to compaction by the machine working on the bench. Slides and sloughing at the base of the highwall are also prevented. With a continuous floor across the bench, the dragline is protected from the danger of caving from the upper portions of the highwall.[48]

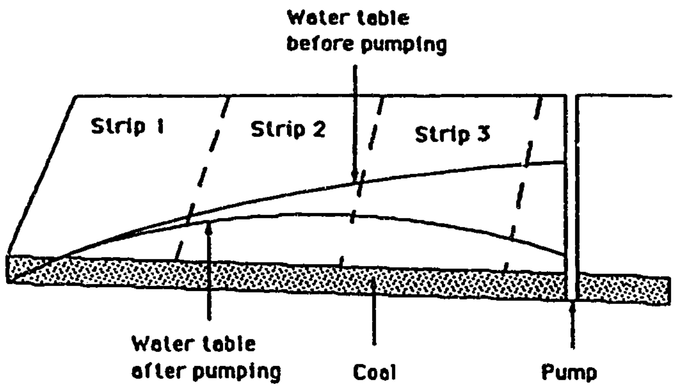


Figure 2.10 Draw down zone around pumped area (after [25])

When operating the dragline with an exposed highwall, stability is dependent upon good blast design for highwall competency. If poor highwall stability exists, increased positioning factors should be used to keep the dragline from exerting pressure close to the bench crest.

2.6.8 Reclamation

As discussed in section 2.6.2, it is better to use narrower pit widths for purposes of reducing spoil grading costs. If the dragline has some excess spoiling capability, this can be somewhat alleviated by varying the swing angle when spoiling. Varying the swing angle can also be used to obtain selectivity in spoil placement.[21] In order to bury interburden material in spoil piles, an extended bench may be used even though it may not be required for production purposes.

Chopdown should be the last activity in a given digout length so that chopcut material is placed on top of spoil piles. Placement of chopcut materials on the pit floor is undesirable from spoil stability and reclamation standpoints.[29]

If an undesirable layer is present in the dragline digout, it may be beneficial to handle material in horizontal layers. The production for this mode of operation may be significantly less, but it will allow selective burial of acidic or toxic material. If undesirable material is spoiled near the surface of spoil piles, it may contaminate surface water or complicate revegetation attempts.[3,4,5,6,7]

2.6.8 Summary

It is relatively easy to select a satisfactory stripping method, but it is difficult to select the method and geometry that will yield the most economic results. This is due to the many decision variables that must be used and the many combinations that can be utilized. It becomes difficult to quantify how much value is placed upon different advantages versus the disadvantages that are sacrificed to achieve the best pit layout. The values would be unique for every mine location due to different equipment and site characteristics.

3. Computer Application of Dragline Methods

3.1 Review of Some Available Programs

In the eighties, several universities and private companies were funded by different branches of the U.S. federal government to create programs for designing strip mine pits. Some of these developers include McDonnell-Douglas Electronics Co., Pennsylvania State University, Montana State University, Fluor Mining and Metals Co., and Mathtec Inc.. White and Jones [49] reviewed the programs from these developers by modifying them to run on the same IBM 370. All of these programs are available to the public because of government funding, but their use has not been widespread in the mining industry. There are several reasons for this: (1) many coal companies are not aware that these programs are available (2) all of the programs were written to simulate only a single seam operation and (3) the programs were not written to run on a personal computer. These programs were written in the Fortran programming language for use on different mainframe computers and range in scope from simple volume calculators to dragline simulators capable of simulating every bucket of overburden which is dug. The very detailed programs will probably require much data gathering and hardware capability. They will also be more time consuming when analyzing many different pit configurations. For general mine planning, a simple range diagram program that keeps track of all the volumes is often the best solution. For more detailed work, the simpler programs may be used in combination with other models.

Hrebar [18] and Hrebar and Cook [19] state that the Colorado School of Mines promotes the concept of creating individual microcomputer programs to deal with various aspects of the mine design problem. Two graduate students from the Colorado School of Mines, Dagdelen [14] and Cook [13], wrote Fortran programs that can be used to calculate dragline pit dimensions and corresponding average production indices. Dagdelen's program considers only a single seam, does not allow the cut width to be defined differently from the spoil pit width, and is not interactive. Cook's program is a continuation of the work initiated by Dagdelen. It is interactive, includes graphical output through the use of an external plotting package, and can handle one or two dipping seams. Both Dagdelen and Cook attempt to modify the two dimensional operating radius equation so that it will better represent the three dimensional spoiling problem. Both students made references to a paper by Chatterjee et al. [9] which states that the excavated material should be dumped in such a position that the smallest swing angle can be recorded without creating rehandle. Therefore since the overburden must be cast along an arc to avoid losing spoil volume, the operating radius must be larger than the two dimensional dragline operating radius. For example, Dagdelen [14] and Dagdelen and Hrebar [20] suggest that the single seam operating radius equation should be restated as

$$DORM = DOR + \frac{X^2}{4 \cdot DOR} \tag{3.1}$$

where: DORM = modified dragline operating radius (m)
DOR = two dimensional dragline operating radius (m)
X = set-up or cut length (m)

This problem is usually accounted for in practice by adding some length (e.g. 3 meters) to the required operating radius. Adding a fixed length is more than adequate when one considers the assumptions that have been made in Dagdelen's equations (e.g. continuous pit, no haul road through the spoil, no curves in the pit, flat topography)[14].

Operations research techniques have been used by several authors [15,32,33,38] to determine the best pit configuration. These work well in theory, however the assignment of objective values and operating constraints is a very subjective matter. Programs that claim to optimize the pit configuration are also usually limited to a single seam operation. If a multiple seam case was attempted, the many rehandling sequences would make it much more difficult to obtain a realistic "best" solution.

Stuart and Cobb [47] discuss the application of a rule-based expert system to aid in the planning of draglines. In this expert system, the Prolog programming language is used to apply heuristic rules such that only feasible solutions are accepted. While this system can provide immediate feedback of many feasible solutions, a danger exists that the user may not have an adequate understanding of the limitations and processes which produced it. An expert system may actually limit the creativity of an engineer when designing a new type of pit because the rule base is not flexible. On the other hand, if the dragline is severely constrained in its operation, an expert system may determine a not so obvious, but effective solution.

If severe constraints are being imposed upon a dragline or more accurate planning is desired, it may be necessary for the company to purchase a program which is capable of three dimensional simulation. Fluor Minerals & Metals, Inc. and Bonner & Moore Associates have developed a group of computer models [34,41] that can be used to simulate single or multi-seam three dimensional dragline mining situations. While their programs can accommodate the effects of haul roads and other varying pit shapes and layouts, they also require considerably large amounts of computer storage capacity and execution time. The operating personnel will have to use a more complex model and therefore will require a competent working knowledge of both draglines and computers in order to achieve the best results.

3.2 Computer Program DRAG

Program DRAG was written so that Obed engineers would have an interactive computer tool which would automate the preparation of range diagrams and volumetric calculations. By allowing interactive definition of many variables, the program is very flexible and allows the user to investigate many pit configurations. The planning of dragline operations is very well suited to the application of computers due to the repetition of geometric calculations.

The program DRAG allows the user to select from four different dragline models. Two of these models are used to generate pit configurations of a single seam dragline cut while the other two are for a two seam case. These models assume that the pit is continuous and straight. Irregularities such as haul roads through the spoil, uneven surface topography, end of pit conditions and interaction with other equipment are not taken into account. Even though the program DRAG has these inherent weaknesses, the ease of variable manipulation allows the user to have a good feel for changes being made. The graphical approach in designing a strip pit is easily understood and remains the most familiar way in which the problem has been historically solved.

3.2.1 Hardware and Software Requirements

The program DRAG in its present form takes just under 208 kilobytes of core storage for the executable code. Most personal computers can easily accommodate this storage requirement. The language chosen to implement the design models on the personal computer is Fortran 77 because of its acceptance and use by many practising engineers. One disadvantage of using Fortran 77 is that it does not have any built in graphics commands or commands for manipulating the cursor on the text screen.

In order to draw range diagrams, the PLOT88 graphics package [36] is used with the Microsoft Fortran compiler [31]. The PLOT88 routines are restricted to a single subroutine for each of the above models so that the plotting can be changed with relative ease if portability is a problem.

Several routines from the Fortran Utilities software package [45] are used for screen cursor manipulation. This is very useful in setting up data cells on the screen so that they can be accessed simply by pressing cursor keys.

The printer is configured according to the parameters given in PRINT.DAT. A sample PRINT.DAT file is given as follows,

```
0        IOPORT VALUE
42       MODEL VALUE
0        0=80 COL PRINTER, 1=132 COL PRINTER
```

Figure 3.1 Sample PRINT.DAT file

If PRINT.DAT does not exist, the default printer is an Epson LQ500 printer. The plotter default is a Hewlett Packard 7475A plotter and this can not be changed without recompiling the program source code. However, the configuration values for a different plotter can be used in PRINT.DAT so that it is defined as a printer in the output selection menu. The configuration values for many different hardware devices are listed in the PLOT88 reference manual [36]. Both CGA and EGA screens are supported by DRAG and listed in the output selection menu. If a wide carriage printer is available, the third value in PRINT.DAT can be set equal to 1. This will move the variable information higher on the paper and allow more room for the range diagram to be drawn. This is especially useful when a scale of 1:500 is desired.

The program DRAG contains a series of subroutines and functions so that it could be broken down into small units which can be worked with and understood with less effort. Common statements are not used in DRAG, so all parameters are passed between routines in a list of arguments.

3.2.2 Program Logic and Flow Diagram

Menus are used throughout the program and each of them has checks for user input errors. The main menu is displayed in Figure 3.2.

```
        ** CHOOSE  A DRAGLINE   METHOD   **

   1       SIDECASTING          - SINGLE  SEAM
   2       EXTENDED   BENCH     - SINGLE  SEAM
   3       SIDECASTING          - TWO  SEAMS
   4       EXTENDED   BENCH     - TWO  SEAMS

               ESC TO QUIT
```

Figure 3.2 Main menu in DRAG for individual method selection

The program DRAG uses the subroutines SEAM1, REH1, SEAM2, and REH2 to represent the models 1, 2, 3, and 4, respectively. Once a dragline mining model has been selected from the main menu, the appropriate subroutine continues to follow the flow diagram in Figure 3.3.

All of the models use the sidecasting method of waste removal, regardless of whether the dragline is operating on the interburden or overburden. The extended bench methods which are currently in use at Obed are modified versions of sidecasting. Models 1 and 3 will be used to look at no rehandle cases where the required operating radius and stacking height are determined. Models 2 and 4 will include the operating radius and stacking height as input variables and use an extended bench where necessary. All rehandle volumes and extended bench widths will be determined. A chopcut is allowed in each of the above models.

3.2.3 DRAG Input

Each of the above four subroutines reads the input variable values and names from a data file and then uses them to create an input screen. For example, subroutine SEAM1 reads SEAM1.DAT to define its variables. If this default data file is not available, the user is prompted for another file. Sample data files are located in appendix B. By having these data files, the user can change the variable names and initial values with a file editor.

Menus are used throughout the program. All screens also check for data input errors from the user. The user can use the cursor keys to move a highlighted cell around the screen until the desired data value is highlighted. The value can be changed by simply typing in the new value and then storing the value by either pressing the carriage return

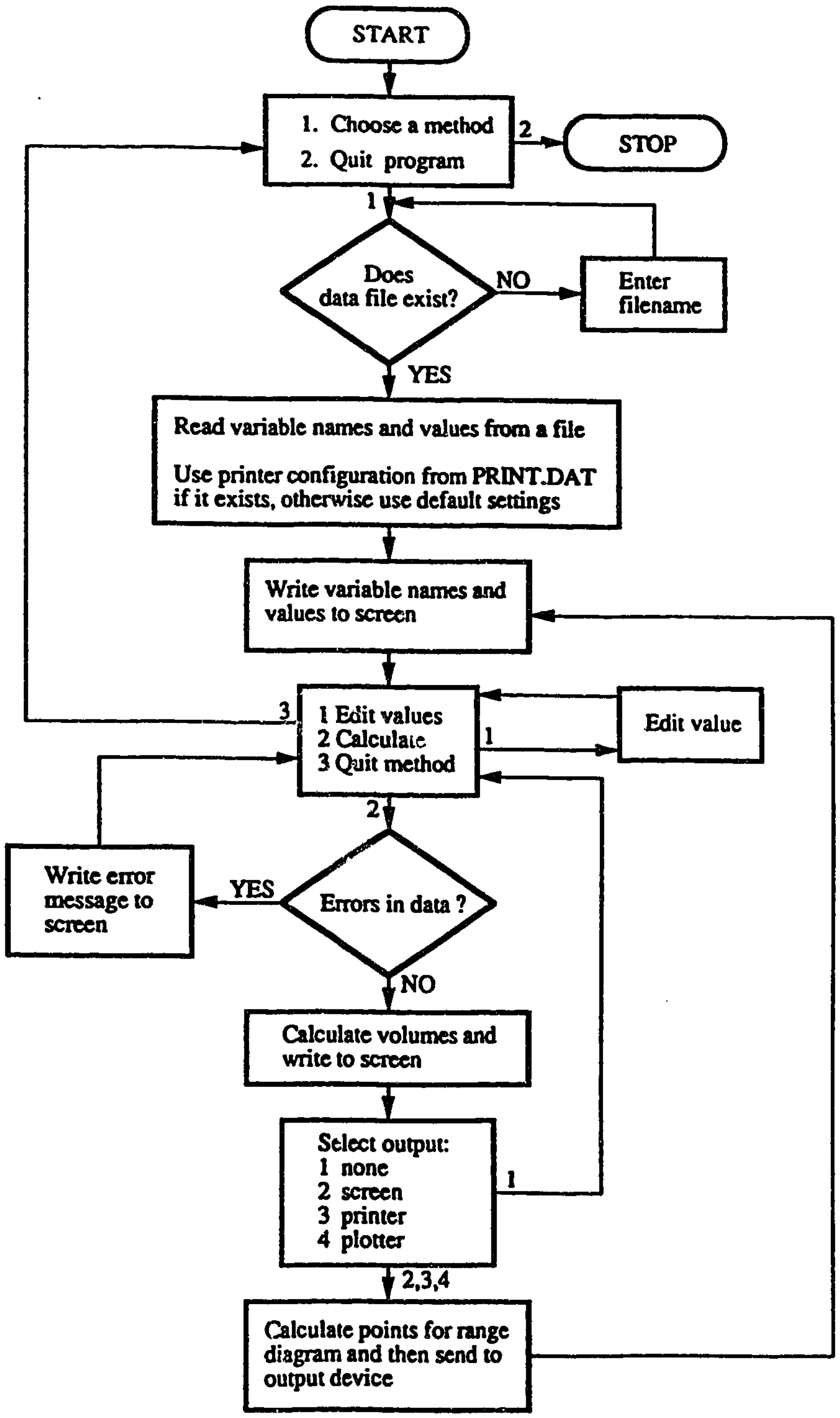

START
1. Choose a method
2. Quit program
2
STOP
1
Does data file exist?
NO
Enter filename
YES
Read variable names and values from a file
Use printer configuration from PRINT.DAT if it exists, otherwise use default settings
Write variable names and values to screen
3
1 Edit values
2 Calculate
3 Quit method
1
Edit value
2
Write error message to screen
YES
Errors in data ?
NO
Calculate volumes and write to screen
Select output:
1 none
2 screen
3 printer
4 plotter
1
2,3,4
Calculate points for range diagram and then send to output device

Figure 3.3 General flow diagram for DRAG

or a cursor key. While typing a new value, the previous value is written beside the highlighted cell as a reference. Should the user decide to use the previous value while entering the new value, the F1 function key may be pressed to restore the previous value. The user can only enter the keys for the numbers 0 to 9 and a single decimal. The remaining keys can be pressed, but they will not be acknowledged on the screen or in the storage location for the variable.

Besides changing the variable values, the user has two other options: (1) Calculate the output values by pressing the space bar or (2) Quit the current method and return to the main menu by pressing the ESC key. If unreasonable data, as described in chapter 4, is present when the space bar has been pressed, an error message is displayed on the screen. The user is then returned to the input screen after pressing any key.

3.2.4 DRAG Output

The output screens always display the volumes of material which are handled by the dragline. If an extended bench is allowed by the current method, the width of the extended bench is also displayed. The user has six different alternatives when the output screen is displayed. The choices are shown in Figure 3.4.

0	NONE
1	CGA SCREEN
2	EGA SCREEN
3	PRINTER (1:500)
4	PRINTER (1:1000)
5	PLOTTER (1:1000)

Figure 3.4 Output devices which may be selected

Only by pressing the keys representing the numbers 0 to 5 will the program be allowed to continue. If 0 is selected, the data edit menu is returned to the display. If 1 or 2 is entered, the range diagram is displayed on a CGA or EGA screen respectively. The range diagram is scaled to fit the selected screen so the drawing scale will vary, but the horizontal and vertical scales will remain equal to each other. The range diagrams that are drawn to the screen are not interactive and are displayed until a key is pressed. If 3 or 4 are selected, the range diagram will be sent to the specified printer at a scale of 1:500 or 1:1000 respectively. If 5 is chosen, the range diagram will be sent to a Hewlett Packard 7475A plotter at a scale of 1:1000. Sample output screens are given in Chapters 4 and 5. The user is returned to the input screen when the output is completed.

The two seam extended bench model (REH2) contains an intermediate output menu which allows the user to generate a range diagram for either the overburden material or the combined overburden and interburden material. This is the only significant deviation from the general flow diagram.

3.2.5 Common Assumptions of Different Models Selected in DRAG

A range diagram is essentially a dragline pit cross-section which is perpendicular to the direction of dragline advance. Areas on the range diagram are multiplied by one metre of pit length so that they represent unit volumes. When volumes are discussed in this thesis, unit volumes are being implied. The different volumes are calculated by equalizing the bank and spoil side areas while accounting for the swell of the bank material. The swell factor in the input data is expressed as a decimal fraction. Therefore a swell factor of 1.20 will result in the spoil volume being 20 percent larger than the bank volume. The angle of repose is assumed to be the same for both the interburden and overburden material.

The four models all assume that the coal face is vertical and, except for REH2, that the spoil does not ride up the coal seam. More restrictions exist when rehandling occurs and therefore more site specific assumptions are required. This is especially true when two seams are considered in REH2. Details of particular calculations and resolution of constraints are given under analysis details in the following chapters 4 and 5. Several error checks are similar for each of the different models and will be noted in chapter 4 once some common input variables are also defined.

4. Single Seam Analysis

The two subroutines, SEAM1 and REH1, deal with uncovering a single horizontal coal seam with a dragline and are called from the program DRAG. SEAM1 calculates a required dragline operating radius and stacking height for a given pit geometry such that no rehandle is required. This type of program is useful when sizing draglines. It also provides the user with a simple range diagram which can be worked with to try new ideas and operating methods. REH1 is very similar to SEAM1 except that the operating radius of the dragline is an input variable. If the dragline operating radius is exceeded when spoiling, an extended bench is used to accommodate the dragline.

4.1 SEAM1

This section will discuss some of the error checks that are common to all of the models as well as the calculations used in SEAM1. The variables used in this analysis are listed in Table 4.1 and displayed on a pit diagram in Figure 4.1. As explained in chapter 3, the names and initial values of these variables are read from SEAM1.DAT and then written to the screen as in Figure 4.2. The common error checks are described as follows: (1) All angles are required to be greater than 0 degrees and less than 90 degrees. (2) The dragline positioning on a bench is calculated by multiplying the positioning factor (ie. PO) with the dragline tub diameter (D). This distance is measured from the centre of the tub to the bench crest. The dragline positioning factors are required to be at least 0.5 so that the dragline tub does not hang over the bench crest. (3) When the dragline is operating on the overburden bench, it must be positioned within the chopcut and main cut. If no chopcut is being used in the current pit geometry, the dragline is still restricted by this check. The user can overcome this by using a specified chopcut width with a thickness of zero.

Table 4.1 Input Variables used in SEAM1

VARIABLE NAME	UNITS	VARIABLE
cut width	m	WC
spoil pit width	m	WP
chopcut width	m	WCH
overburden thickness	m	TOB
chopcut thickness	m	TCH
coal seam thickness	m	TTC
spoil pile angle	degrees	THETA (θ)
overburden highwall angle	degrees	BETA (β)
overburden swell factor	(decimal)	SWOB
overburden positioning factor	(decimal)	PO
tub diameter	m	D

If none of the described data errors occur after the user has pressed the space bar, the cut and spoil volumes are calculated. The bank volumes of the chopcut (VCHOP) and the overburden (VOB) are calculated by multiplying their respective thicknesses by their widths as follows,

$$VCHOP = WCH \cdot TCH \tag{4.1}$$

$$VOB = TOB \cdot WC \tag{4.2}$$

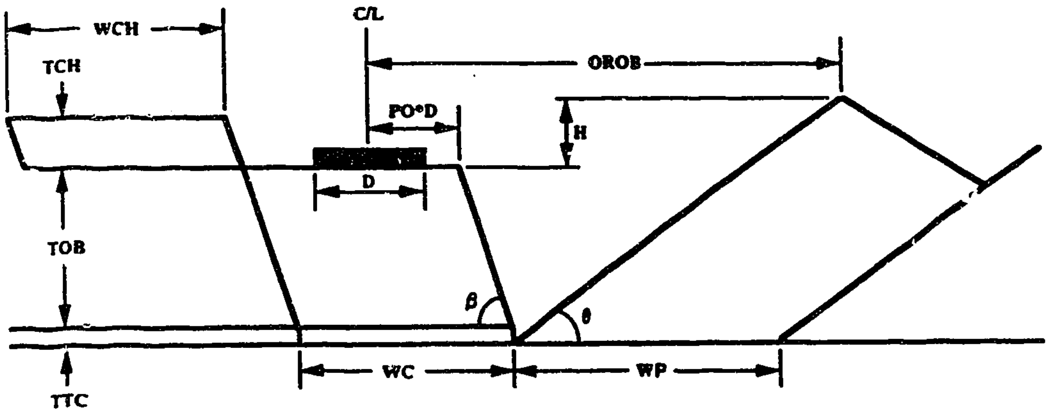
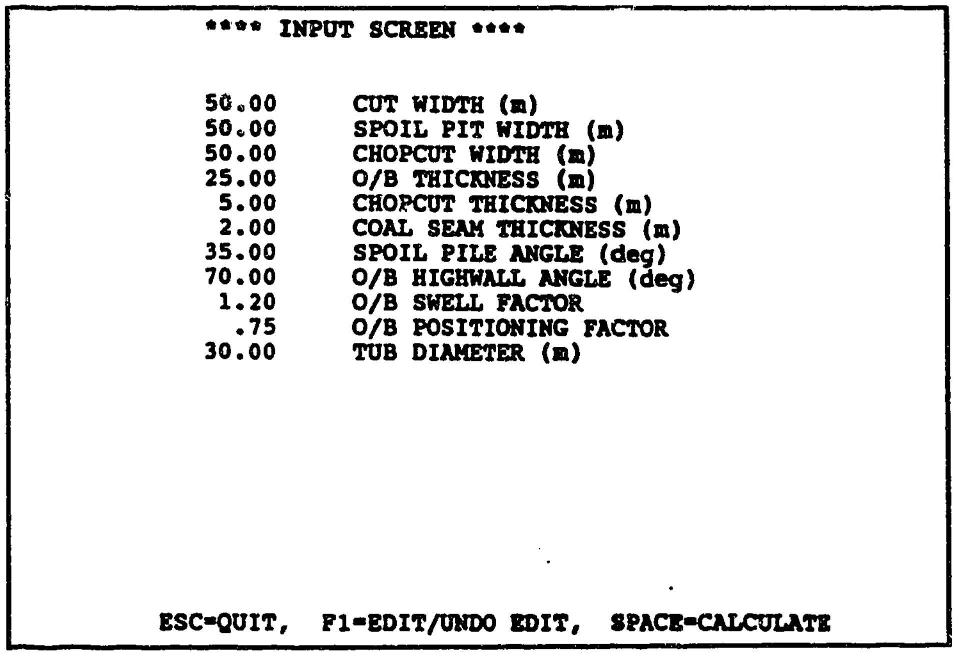


Figure 4.1 SEAM1 pit variables



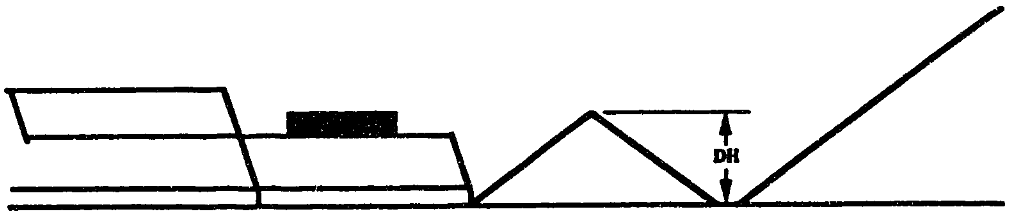
```
                **** INPUT SCREEN ****


            50.00      CUT WIDTH (m)
            50.00      SPOIL PIT WIDTH (m)
            50.00      CHOPCUT WIDTH (m)
            25.00      O/B THICKNESS (m)
             5.00      CHOPCUT THICKNESS (m)
             2.00      COAL SEAM THICKNESS (m)
            35.00      SPOIL PILE ANGLE (deg)
            70.00      O/B HIGHWALL ANGLE (deg)
             1.20      O/B SWELL FACTOR
              .75      O/B POSITIONING FACTOR
            30.00      TUB DIAMETER (m)









        ESC=QUIT,  F1=EDIT/UNDO EDIT,  SPACE=CALCULATE
```

Figure 4.2 Input screen for SEAM1

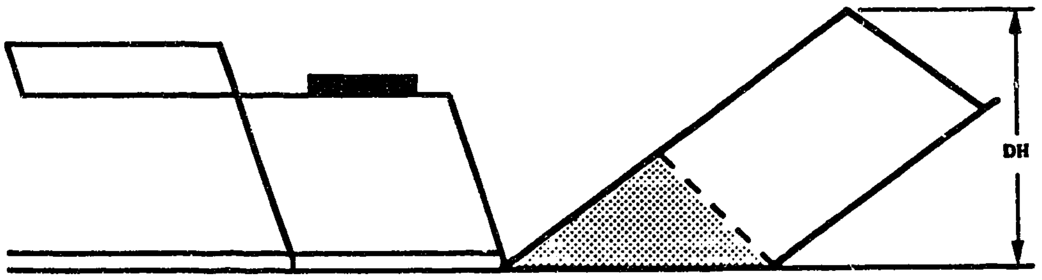
The volume of the spoil (VSPOIL) is simply the chopcut volume added to the overburden (or main cut) volume and then multiplied by the swell factor of the overburden (SWOB).

$$VSPOIL = (VCHOP + VOB)\cdot SWOB \tag{4.3}$$

The next step in SEAM1 is to calculate the required dragline operating radius and stacking height. Figure 4.3 shows the two possible pit geometries that are used by SEAM1.



(4.3a) spoil volume not large enough to cover spoil pit



(4.3b) spoil volume rides up previous spoil pile

Figure 4.3 Possible pit geometries for SEAM1

The shaded area in Figure 4.3b represents a test volume (VTEST) for deciding which pit geometry to use. For example, if VSPOIL is greater than VTEST, the geometry in Figure 4.3b is used. Otherwise the geometry in Figure 4.3a is used.

$$VTEST = \frac{WP^2 \cdot TAN(\theta)}{4} \tag{4.4}$$

An internal variable DH is calculated according to which geometrical configuration is required. DH is the height of the spoil pile above the bottom of the coal seam. For the geometry in Figure 4.3a, DH is calculated as

$$DH = \sqrt{VSPOIL \cdot \tan(\theta)} \tag{4.5}$$

while DH is calculated for the geometry in Figure 4.3b as

$$DH = \frac{VSPOIL + VTEST}{WP} \tag{4.6}$$

The required overburden operating radius (OROB) is calculated as

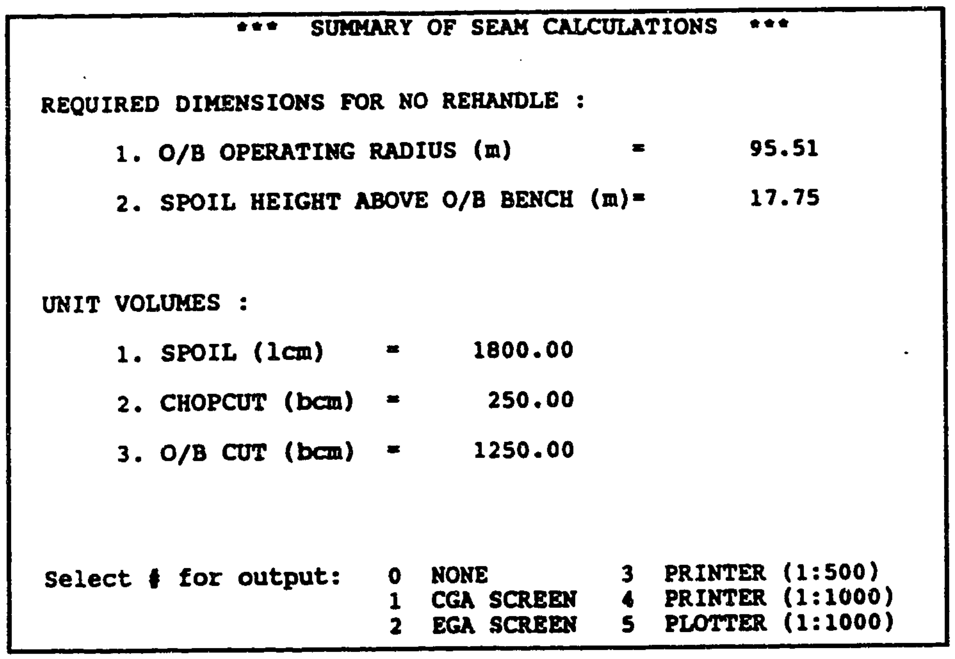
$$OROB = PO \cdot D + \frac{TOB}{\tan(\beta)} + \frac{DH}{\tan(\theta)} \tag{4.7}$$

while the required stacking height (H) is then calculated as

$$H = DH - TOB - TTC \tag{4.8}$$

A sample output screen is shown in Figure 4.4.

The next step for the user is to choose which output option is desired. If 0 is pressed, the user is returned to the edit screen and is allowed to change the variable values again. All of the other output options require that the coordinates of a range diagram are calculated in order that they can be sent to an output device.



```
                  ***  SUMMARY OF SEAM CALCULATIONS  ***

REQUIRED DIMENSIONS FOR NO REHANDLE :

     1. O/B OPERATING RADIUS (m)          =        95.51

     2. SPOIL HEIGHT ABOVE O/B BENCH (m)=         17.75


UNIT VOLUMES :

     1. SPOIL (lcm)      =     1800.00

     2. CHOPCUT (bcm)    =      250.00

     3. O/B CUT (bcm)    =     1250.00



Select # for output:   0  NONE          3  PRINTER (1:500)
                       1  CGA SCREEN    4  PRINTER (1:1000)
                       2  EGA SCREEN    5  PLOTTER (1:1000)
```

Figure 4.4 SEAM1 output screen

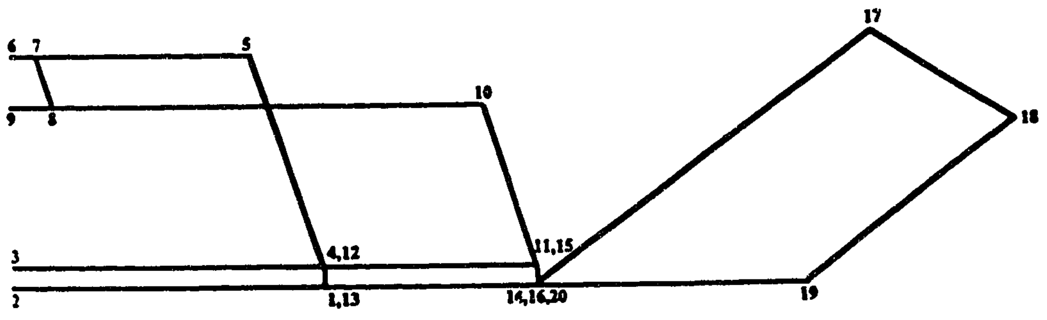
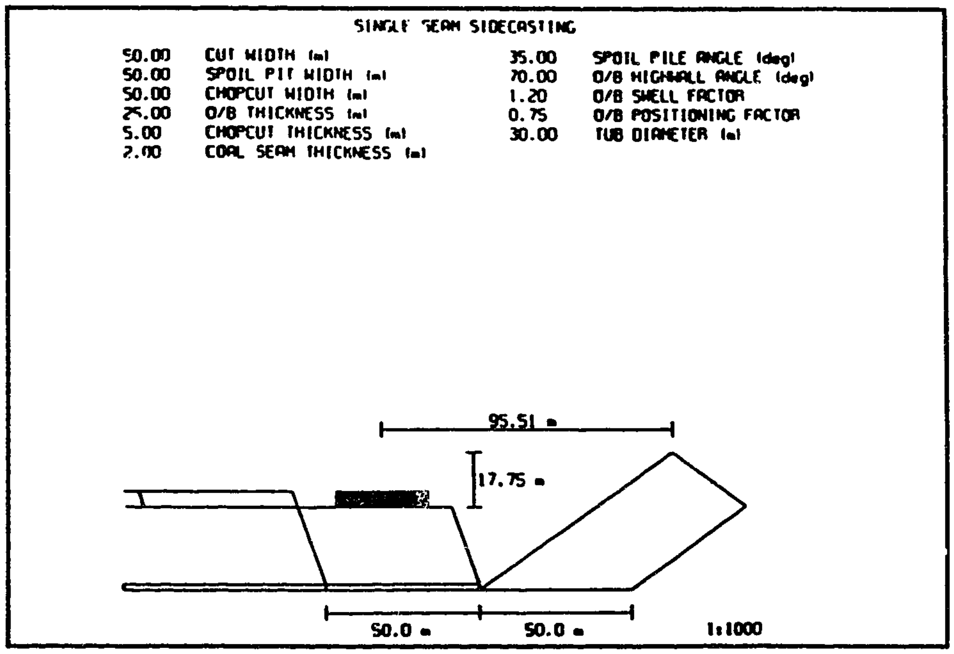


Figure 4.5 Point locations used for SEAM1 graphical output

The coordinates, calculated with respect to the points in Figure 4.5, are then passed into subroutine PLOT1 and drawn using the PLOT88 software package [36]. The input data in Figure 4.2 has been used to produce the graphical output shown in Figure 4.6. After completion of the graphics output, the user is returned to the edit screen. The user then may either try another pit configuration or return to the main model selection menu.



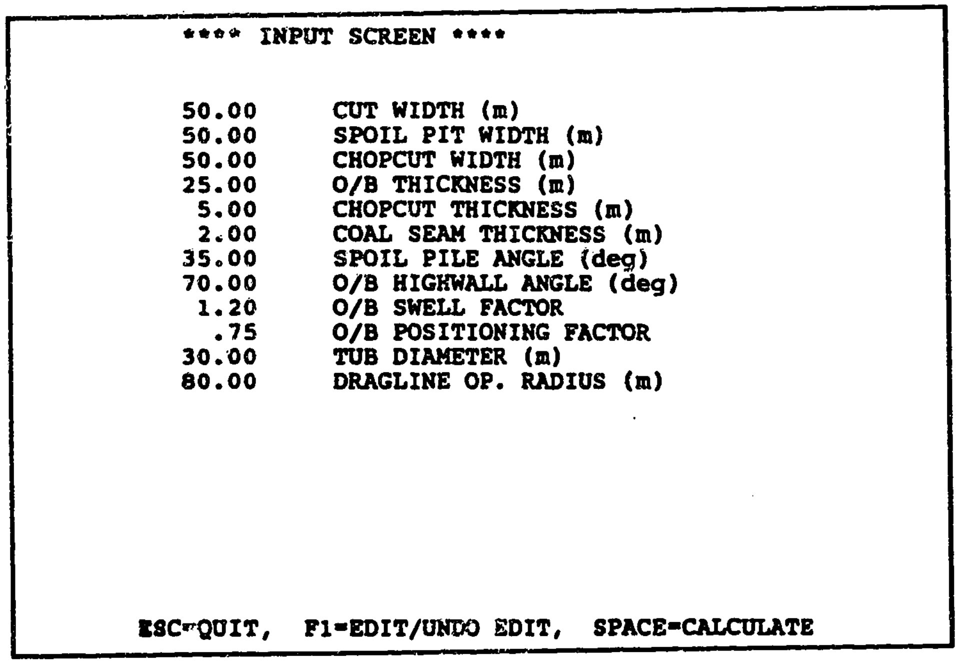
```
                         SINGLE SEAM SIDECASTING
50.00   CUT WIDTH (m)                     35.00   SPOIL PILE ANGLE (deg)
50.00   SPOIL PIT WIDTH (m)               70.00   O/B HIGHWALL ANGLE (deg)
50.00   CHOPCUT WIDTH (m)                 1.20    O/B SWELL FACTOR
25.00   O/B THICKNESS (m)                 0.75    O/B POSITIONING FACTOR
5.00    CHOPCUT THICKNESS (m)             30.00   TUB DIAMETER (m)
2.00    COAL SEAM THICKNESS (m)
```

Figure 4.6 SEAM1 graphical output

4.2 REH1

REH1 assumes that only the dragline reach will restrict the extended bench procedure for a single seam. Therefore the stacking height is calculated within the program, but it is not checked for operating feasibility. REH1 also assumes that the slope of the extended bench highwall is the same as the spoil pile angle (θ).

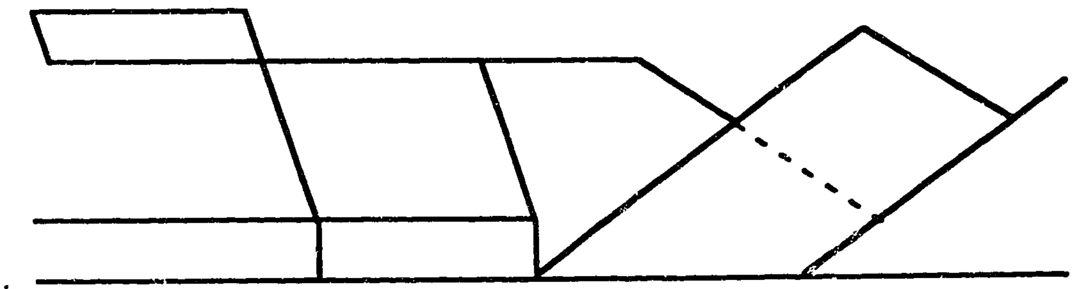
All of the input variables listed in Table 4.1 are used by REH1. The dragline operating radius (DOR) in metres is the only additional variable. These variables can be seen in Figure 4.1. The names and initial values are read from the default file called REH1.DAT. The input procedures and input data value checks are the same as those described for SEAM1. A sample input screen is shown in Figure 4.7.



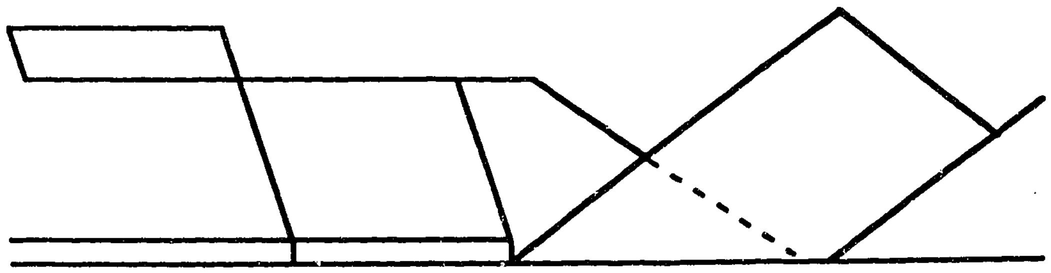
```
                **** INPUT SCREEN ****

                50.00     CUT WIDTH (m)
                50.00     SPOIL PIT WIDTH (m)
                50.00     CHOPCUT WIDTH (m)
                25.00     O/B THICKNESS (m)
                 5.00     CHOPCUT THICKNESS (m)
                 2.00     COAL SEAM THICKNESS (m)
                35.00     SPOIL PILE ANGLE (deg)
                70.00     O/B HIGHWALL ANGLE (deg)
                 1.20     O/B SWELL FACTOR
                  .75     O/B POSITIONING FACTOR
                30.00     TUB DIAMETER (m)
                80.00     DRAGLINE OP. RADIUS (m)

          ESC=QUIT,  F1=EDIT/UNDO EDIT,  SPACE=CALCULATE
```

Figure 4.7 REH1 input screen

The first step in REH1 calculations is to calculate the required dragline operating radius (OROB) and stacking height (H) as was done in SEAM1. If OROB is less than the actual dragline operating radius, an extended bench is not required and the output is identical to that of SEAM1. As in SEAM1, there are two different pit geometries that are possible when there is no extended bench. If an extended bench is required, there are two more pit geometries which may be possible. The difference between these two pit configurations is that one of them has an extended bench highwall which rides up the spoil pile slope while the other does not. These two pit geometries are shown in Figure 4.8. If the extended bench is larger than the span across the spoil pit, the user is provided with two options. The first option is to restrict the extended bench width so that it just reaches the final spoil pile and then calculate the required dragline operating radius. This may be useful if the user wants to see how inadequate the dragline's operating radius is without cutting into the final spoil. The second option returns the user to the edit screen.



(a) Extended bench rides up spoil slope



(b) Extended bench does not ride up spoil slope

Figure 4.8 REH1 pit geometries using an extended bench

If an extended bench is required, the width of the extended bench (ET) is calculated as the difference between the required dragline operating radius (OROB) and the actual dragline operating radius (DOR).

$$ET = OROB - DOR \tag{4.9}$$

Several internal variables used in REH1 are displayed in Figure 4.9. These variables include E1, E2 and EB.

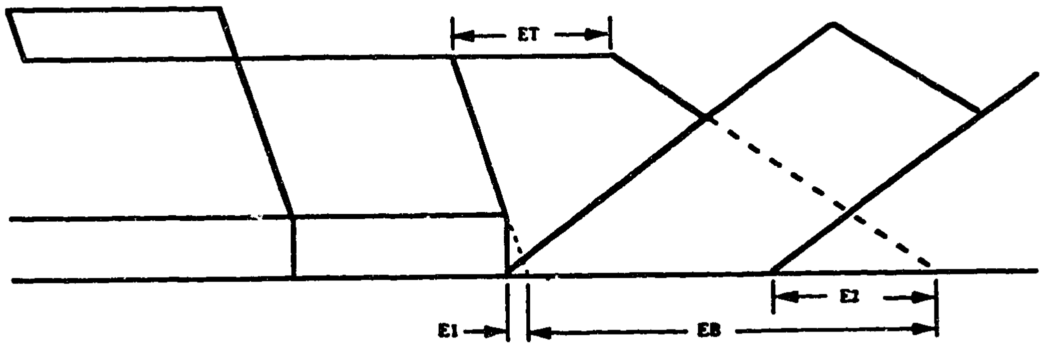


Figure 4.9 Internal variables used in REH1

E1 and EB are defined as follows,

$$E1 = \frac{TTC}{\tan(\beta)} \tag{4.10}$$

$$EB = ET + (TOB + TTC)\cdot\left(\frac{1}{\tan(\theta)} - \frac{1}{\tan(\beta)}\right) \tag{4.11}$$

If the sum of E1 and EB is less than the width of the spoil pit (WP), the value of E2 is zero and the extended bench highwall does not ride up the spoil slope. Otherwise E2 is calculated as follows,

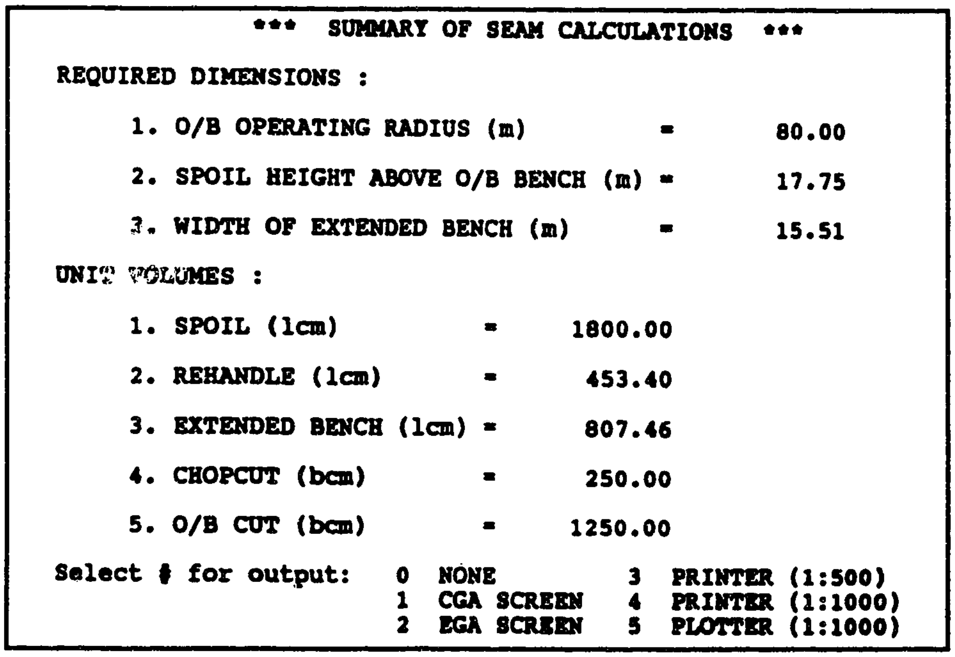
$$E2 = E1 + EB - WP \tag{4.12}$$

The bank and spoil volumes are calculated in the same manner as for SEAM1. The rehandle (RVOL) and extended bench (EVOL) volumes are then calculated as shown below,

$$RVOL = \frac{(ET + EB)(TOB + TTC) + E1 \cdot TTC}{2} - \frac{\tan(\theta) \cdot (EB + E1)^2}{4} \tag{4.13}$$

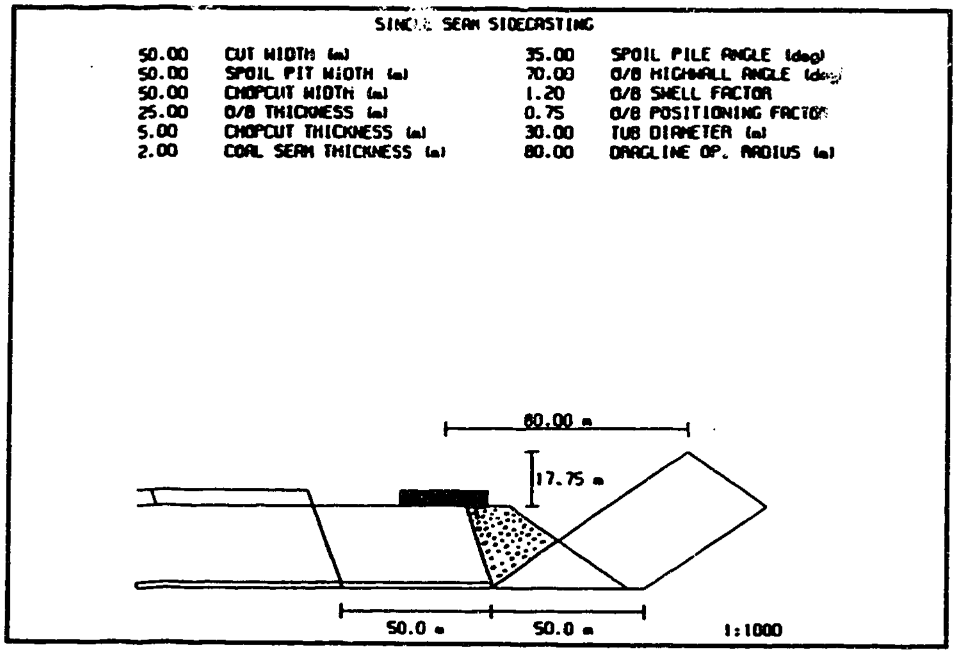
$$EVOL = \frac{(ET + EB)(TOB + TTC) + E1 \cdot TTC}{2} - \frac{\tan(\theta) \cdot (E2)^2}{4} \tag{4.14}$$

A sample REH1 output screen is shown in Figure 4.10 with the corresponding graphical output shown in Figure 4.11.



```
            ***  SUMMARY OF SEAM CALCULATIONS  ***
REQUIRED DIMENSIONS :
     1. O/B OPERATING RADIUS (m)            =      80.00
     2. SPOIL HEIGHT ABOVE O/B BENCH (m) =      17.75
     3. WIDTH OF EXTENDED BENCH (m)         =      15.51
UNIT VOLUMES :
     1. SPOIL (lcm)           =     1800.00
     2. REHANDLE (lcm)        =      453.40
     3. EXTENDED BENCH (lcm) =      807.46
     4. CHOPCUT (bcm)         =      250.00
     5. O/B CUT (bcm)         =     1250.00
Select # for output:    0  NONE          3  PRINTER (1:500)
                        1  CGA SCREEN    4  PRINTER (1:1000)
                        2  EGA SCREEN    5  PLOTTER (1:1000)
```

Figure 4.10 REH1 output screen



```
                         SINGLE SEAM SIDECASTING

50.00   CUT WIDTH (m)                  35.00   SPOIL PILE ANGLE (deg)
50.00   SPOIL PIT WIDTH (m)            70.00   O/B HIGHWALL ANGLE (deg)
50.00   CHOPCUT WIDTH (m)              1.20    O/B SWELL FACTOR
25.00   O/B THICKNESS (m)              0.75    O/B POSITIONING FACTOR
5.00    CHOPCUT THICKNESS (m)          30.00   TUB DIAMETER (m)
2.00    COAL SEAM THICKNESS (m)        80.00   DRAGLINE OP. RADIUS (m)
```

Figure 4.11 REH1 graphical output

The rehandle volume is represented in Figure 4.11 by the shaded area. The extended bench volume includes all of the rehandle volume as well as the final spoil pile volume located below the extended bench slope. The coordinates sent by REH1 to subroutine PLOTR1 are the same as the coordinates sent by SEAM1 to subroutine PLOT1. This is because the final spoil configuration is the same in both cases as the spoil slope rises from the base of the coal seam. REH1 also passes the values of E1, E2, EB and ET to the subroutine PLOTR1 so that the extended bench coordinates can be calculated and drawn.

4.3 Summary

The two single seam methods described in subroutines SEAM1 and REH1 are useful for preliminary analysis of multiple seam operations as well as single seam operations.

There are few site specific assumptions made in either and the user can obtain a rough feel for how much difficulty a dragline will encounter for a given overburden depth.

5. Two Seam Analysis

Subroutines SEAM2 and REH2 are called from the program DRAG for calculations with respect to uncovering two horizontal coal seams with a dragline. SEAM2 calculates the required operating radii from both the overburden and interburden benches such that no rehandle is required and the required stacking height when the dragline is operating on the interburden bench. In REH2, the overburden and interburden spoiling practices are restricted by the dragline dimensions. When the overburden is being spoiled at the dragline's maximum operating radius, it is allowed to ride up the highwall only until the interburden bench crest is reached. Beyond this point REH2 will not allow the calculations to proceed. When the dragline is operating on the interburden bench, the following site specific methods may be applied: (1) An extended bench is used if necessary. (2) If the dragline's stacking capability is inadequate, two dragline positions are used on the extended bench to lower the required stacking height of the two resulting peaks.

Cut and spoil volumes are calculated for both SEAM2 and REH2, with REH2 also reporting values for rehandle and extended bench volumes. The default data files for SEAM2 and REH2 are SEAM2.DAT and REH2.DAT respectively. The input data can be modified interactively as described in chapter 3.

5.1 SEAM2

The input variables used in SEAM2 are listed in Table 5.1 and displayed in Figure 5.1. A sample edit screen for SEAM2 is shown in Figure 5.2. When the space bar is pressed to calculate output values, all of the error checks described in chapter 4 are carried out in addition to the following check: The width of the interburden cut must be less than or equal to the combined width of the overburden cut and the safety bench. This forces all of the interburden to be uncovered so that a realistic pit can be designed.

Table 5.1 Input Variables used in SEAM2

VARIABLE NAME	UNITS	VARIABLE
overburden cut width	m	WCO
interburden cut width	m	WC
spoil pit width	m	WP
chopcut width	m	WCH
safety bench width	m	WSB
overburden thickness	m	TOB
interburden thickness	m	TIB
chopcut thickness	m	TCH
top seam thickness	m	TTC
lower seam thickness	m	TBC
spoil pile angle	degrees	THETA (θ)
overburden highwall angle	degrees	BETA (β)
interburden highwall angle	degrees	ALPHA (α)
overburden swell factor	(decimal)	SWOB
interburden swell factor	(decimal)	SWIB
overburden positioning factor	(decimal)	PO
interburden positioning factor	(decimal)	PI
tub diameter	m	D

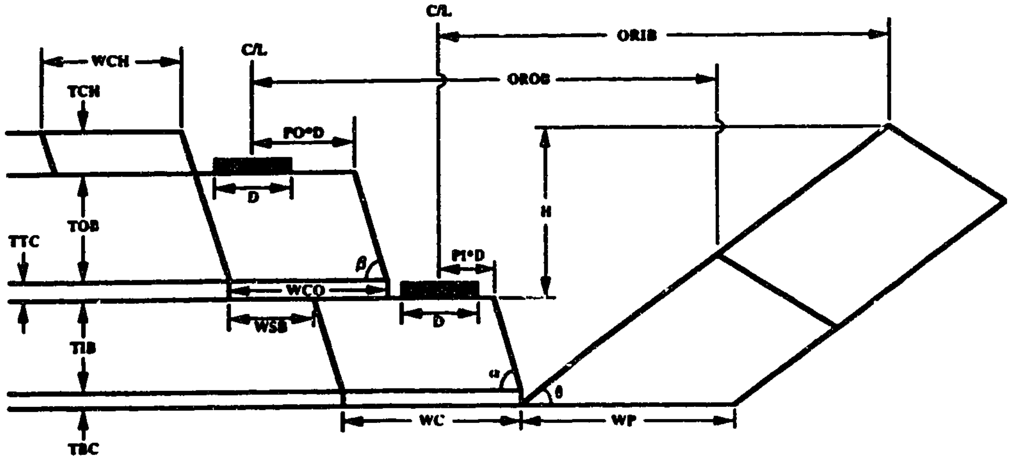
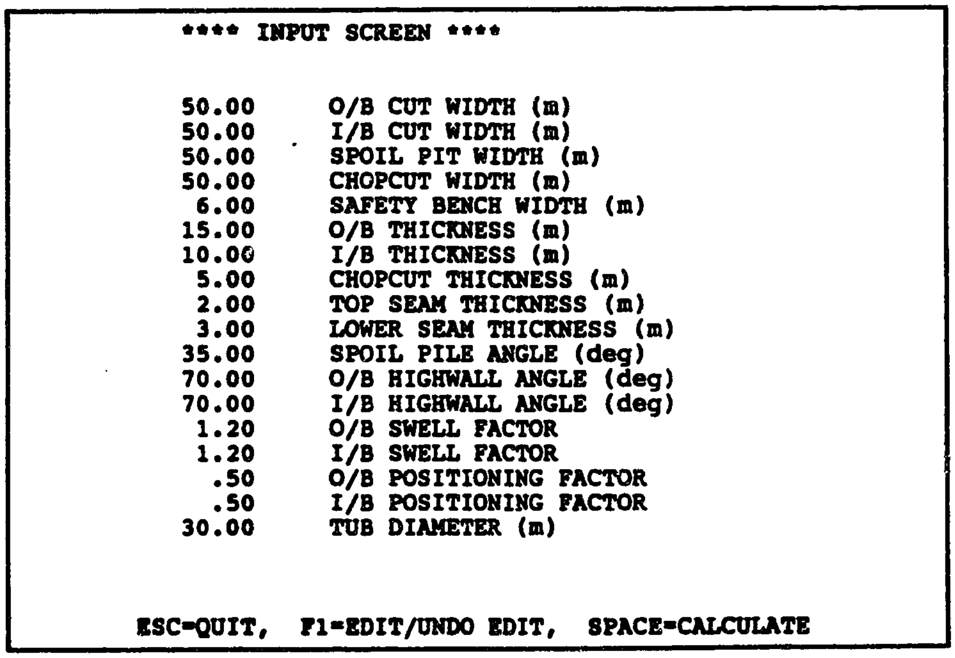


Figure 5.1 SEAM2 pit variables



```
                **** INPUT SCREEN ****


                50.00      O/B CUT WIDTH (m)
                50.00      I/B CUT WIDTH (m)
                50.00      SPOIL PIT WIDTH (m)
                50.00      CHOPCUT WIDTH (m)
                 6.00      SAFETY BENCH WIDTH (m)
                15.00      O/B THICKNESS (m)
                10.00      I/B THICKNESS (m)
                 5.00      CHOPCUT THICKNESS (m)
                 2.00      TOP SEAM THICKNESS (m)
                 3.00      LOWER SEAM THICKNESS (m)
                35.00      SPOIL PILE ANGLE (deg)
                70.00      O/B HIGHWALL ANGLE (deg)
                70.00      I/B HIGHWALL ANGLE (deg)
                 1.20      O/B SWELL FACTOR
                 1.20      I/B SWELL FACTOR
                  .50      O/B POSITIONING FACTOR
                  .50      I/B POSITIONING FACTOR
                30.00      TUB DIAMETER (m)


           ESC=QUIT,  F1=EDIT/UNDO EDIT,  SPACE=CALCULATE
```

Figure 5.2 Input screen for SEAM2

The bank volumes of the chopcut (VCHOP), overburden (VOB) and interburden (VIB) are calculated by multiplying their respective thicknesses by their widths as follows:

$$VOB = TOB \cdot WCO \tag{5.1}$$

$$VIB = TIB \cdot WC \tag{5.2}$$

$$VCHOP = WCH \cdot TCH \tag{5.3}$$

The loose volumes of the overburden (VOBSP), interburden (VIBSP) and total spoil (VSPOIL) are calculated by using the previous bank volumes and accounting for the swell as follows:

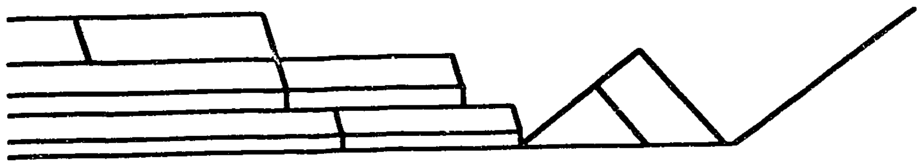
$$VOBSP = (VCHOP + VOB) \cdot SWOB \tag{5.4}$$

$$VIBSP = VIB \cdot SWIB \tag{5.5}$$

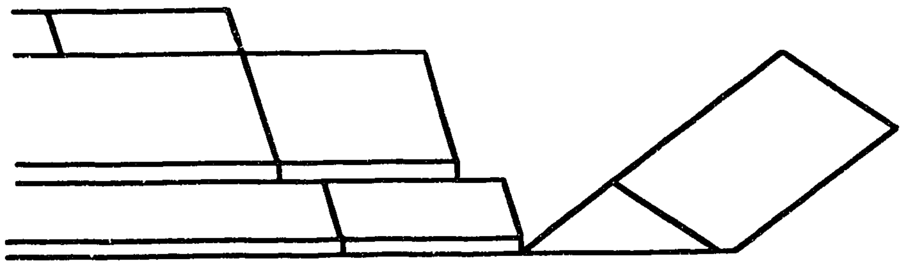
$$VSPOIL = VIBSP + VOBSP \tag{5.6}$$

The next step in SEAM2 is to calculate the required dragline operating radius from both benches and the dragline stacking height from the interburden bench. Figure 5.3 shows the possible pit geometries that are used by SEAM2 in this calculation.

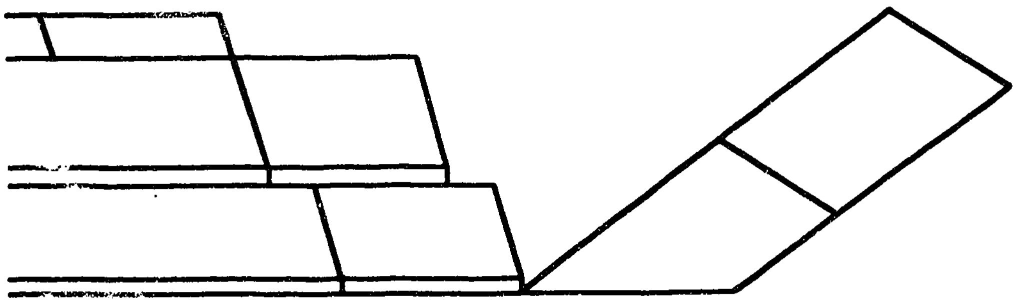
The variable, VTEST, as defined in Equation 4.4 and shaded in Figure 4.3, is used again to determine which pit geometry is required. Two internal variables, DH1 and DH2, are calculated according to which geometrical configuration is required. DH1 and DH2 are the heights above the lower coal seam of the overburden and total spoil respectively. If the total spoil (VSPOIL) is smaller than VTEST, the pit geometry of Figure 5.3a is used and DH1 and DH2 are calculated as in Equations 5.7 and 5.8.



5.3a VTEST > VSPOIL



5.3b VOBSP $\leq$ VTEST $\leq$ VSPOIL



5.3c VTEST < VOBSP

Figure 5.3 Possible pit geometries for SEAM2

$$DH1 = \sqrt{VOBSP \cdot \tan(\theta)} \tag{5.7}$$

$$DH2 = \sqrt{VSPOIL \cdot \tan(\theta)} \tag{5.8}$$

When the overburden spoil (VOBSP) is smaller than VTEST but the total spoil (VSPOIL) is larger, the pit geometry of Figure 5.3b is used and DH1 and DH2 are calculated as

$$DH1 = \sqrt{VOBSP \cdot \tan(\theta)} \tag{5.9}$$

$$DH2 = \frac{VSPOIL + VEST}{WP} \tag{5.10}$$

If the overburden spoil (VOBSP) is larger than VTEST, the pit geometry of Figure 5.3c is used with DH1 and DH2 defined as

$$DH1 = \frac{VOBSP + VEST}{WP} \tag{5.11}$$

$$DH2 = \frac{VSPOIL + VEST}{WP} \tag{5.12}$$

When the dragline is operating on the overburden bench, the required operating radius (OROB) is calculated as in Equation 5.13.

$$OROB = PO \cdot D + \frac{TOB}{\tan(\beta)} + WSB + \frac{TIB}{\tan(\alpha)} + \frac{DH1}{\tan(\theta)} \tag{5.13}$$

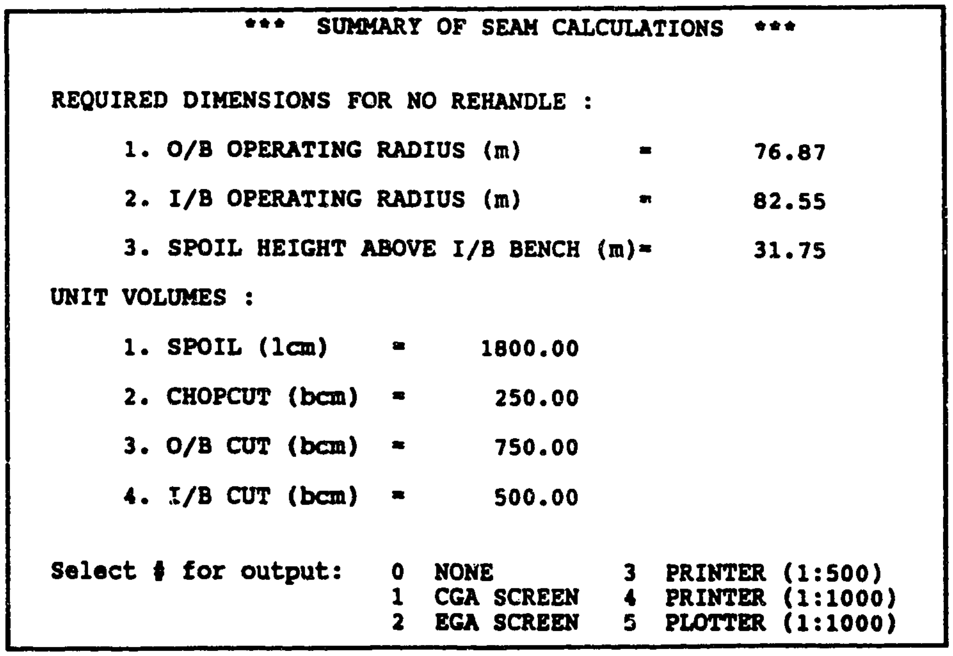
Equation 5.14 gives the dragline operating radius (ORIB) for the dragline operating on the interburden bench.

$$ORIB = PI \cdot D + \frac{TIB}{\tan(\alpha)} + \frac{DH2}{\tan(\theta)} \tag{5.14}$$

The required stacking height (H) for the dragline while it is operating on the interburden bench is

$$H = DH2 - TIB - TBC \tag{5.15}$$

A sample output screen is shown in Figure 5.4. From this screen, the user can choose a suitable output device and create output as shown in Figure 5.5. The graphical output shown in Figure 5.5 is calculated using the point locations in Figure 5.6.

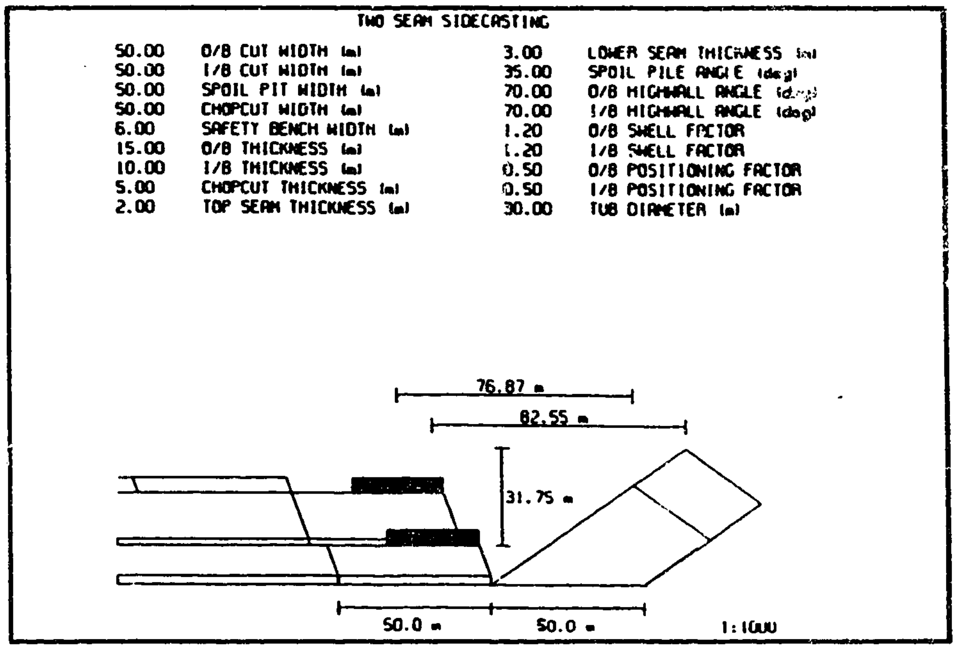


```
                *** SUMMARY OF SEAM CALCULATIONS ***

REQUIRED DIMENSIONS FOR NO REHANDLE :

     1. O/B OPERATING RADIUS (m)        =      76.87

     2. I/B OPERATING RADIUS (m)        =      82.55

     3. SPOIL HEIGHT ABOVE I/B BENCH (m)=      31.75

UNIT VOLUMES :

     1. SPOIL (lcm)    =     1800.00

     2. CHOPCUT (bcm)  =      250.00

     3. O/B CUT (bcm)  =      750.00

     4. I/B CUT (bcm)  =      500.00

Select # for output:   0  NONE          3  PRINTER (1:500)
                       1  CGA SCREEN    4  PRINTER (1:1000)
                       2  EGA SCREEN    5  PLOTTER (1:1000)
```

Figure 5.4 SEAM2 output screen

5.2 REH2

In addition to variables used by SEAM2, REH2 requires the additional input variables listed in Table 5.2. The nature of these variables is illustrated in Figure 5.7. A sample input screen is given in Figure 5.8.



```
                          TWO SEAM SIDECASTING

50.00    O/B CUT WIDTH (m)                  3.00    LOWER SEAM THICKNESS (m)
50.00    I/B CUT WIDTH (m)                 35.00    SPOIL PILE ANGLE (deg)
50.00    SPOIL PIT WIDTH (m)               70.00    O/B HIGHWALL ANGLE (deg)
50.00    CHOPCUT WIDTH (m)                 70.00    I/B HIGHWALL ANGLE (deg)
6.00     SAFETY BENCH WIDTH (m)             1.20    O/B SWELL FACTOR
15.00    O/B THICKNESS (m)                  1.20    I/B SWELL FACTOR
10.00    I/B THICKNESS (m)                  0.50    O/B POSITIONING FACTOR
5.00     CHOPCUT THICKNESS (m)              0.50    I/B POSITIONING FACTOR
2.00     TOP SEAM THICKNESS (m)            30.00    TUB DIAMETER (m)
```

Figure 5.5 SEAM2 graphical output

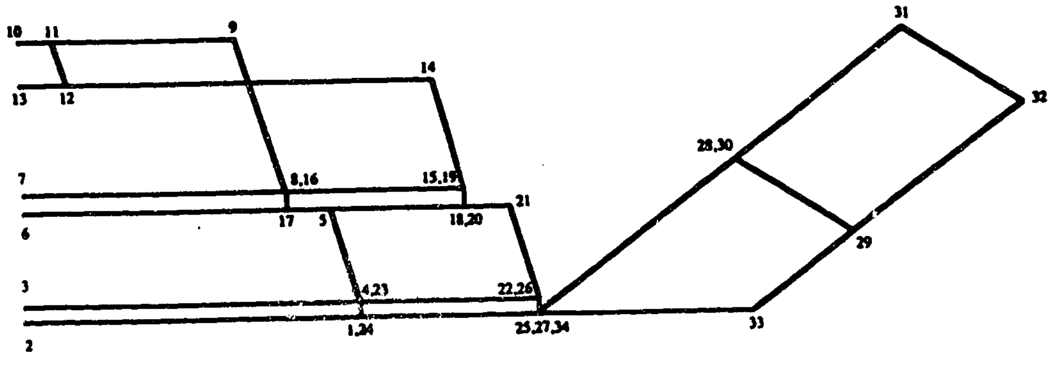


Figure 5.6 Point locations used for SEAM2 graphical output

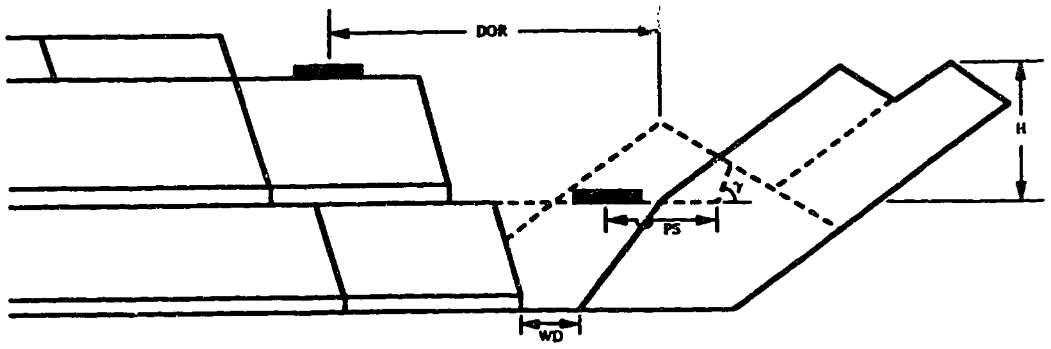
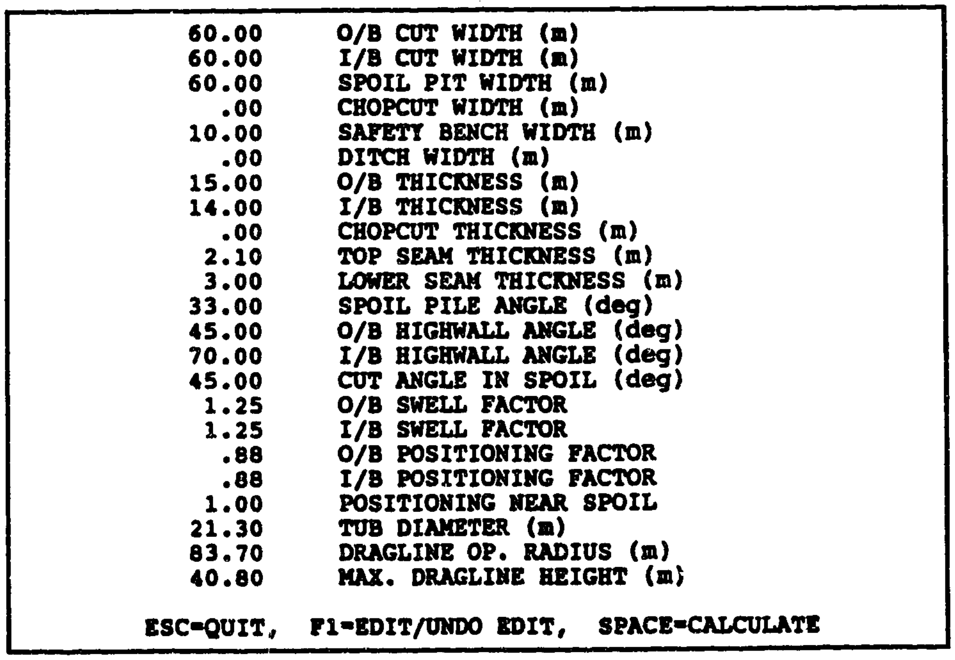


Figure 5.7 REH2 pit variables



```
       60.00     O/B CUT WIDTH (m)
       60.00     I/B CUT WIDTH (m)
       60.00     SPOIL PIT WIDTH (m)
         .00     CHOPCUT WIDTH (m)
       10.00     SAFETY BENCH WIDTH (m)
         .00     DITCH WIDTH (m)
       15.00     O/B THICKNESS (m)
       14.00     I/B THICKNESS (m)
         .00     CHOPCUT THICKNESS (m)
        2.10     TOP SEAM THICKNESS (m)
        3.00     LOWER SEAM THICKNESS (m)
       33.00     SPOIL PILE ANGLE (deg)
       45.00     O/B HIGHWALL ANGLE (deg)
       70.00     I/B HIGHWALL ANGLE (deg)
       45.00     CUT ANGLE IN SPOIL (deg)
        1.25     O/B SWELL FACTOR
        1.25     I/B SWELL FACTOR
         .88     O/B POSITIONING FACTOR
         .88     I/B POSITIONING FACTOR
        1.00     POSITIONING NEAR SPOIL
       21.30     TUB DIAMETER (m)
       83.70     DRAGLINE OP. RADIUS (m)
       40.80     MAX. DRAGLINE HEIGHT (m)

   ESC=QUIT,  F1=EDIT/UNDO EDIT,  SPACE=CALCULATE
```

Figure 5.8 REH2 input screen

Table 5.2 Additional variables used in REH2

VARIABLE NAME	UNITS	VARIABLE
ditch width	m	WD
cut angle in spoil	degrees	GAMA (γ)
positioning near spoil	decimal	PS
dragline operating radius	m	DOR
maximum dragline height	m	H

The input data checks of SEAM2 are imposed as well as the following: (1) An additional positioning factor (PS) is used where the dragline is positioned a distance away from the overburden spoil cut face if an extended bench is required. Its minimum value is also 0.5. (2) The interburden highwall angle (α) must be greater than the extended bench angle. This is achieved by requiring that the overburden highwall angle is greater than the spoil pile angle (θ) and then setting the extended bench angle equal to θ. (3) The spoil cut angle (γ) must be larger than the spoil pile angle (θ). (4) The maximum allowable ditch width occurs when a vertical spoil slope is obtained by over-steepening from the spoil pit floor up to the interburden bench elevation.

If calculations are allowed to proceed, the following checks will take place during calculations: (1) The dragline must be able to spoil beyond the crest of the interburden crest. (2) If overburden spoil rides up onto the interburden bench, calculations are not allowed to proceed. (3) The combined overburden and interburden spoil volume must be large enough so that the spoil peak has an elevation above the interburden bench. If only the overburden range diagram is going to be generated, this restriction is not required. (4) The routine AREA is used within REH2 to calculate several areas using the method of coordinates. A check for a simple closed polygon was employed as a guard against invalid areas being reported. (5) The maximum allowable spoil material occurs when an extended bench is used and the spoil pile has two spoil peaks. Two peaks are built when the dragline uses two positions to spoil onto the spoil pile.

The cut volumes (VCHOP, VOB and VIB) and spoil volumes (VOBSP, VIBSP and VSPOIL) are calculated in the same manner as shown in Equations 5.1 to 5.6. The next step in REH2 is to check if the dragline can spoil beyond the interburden highwall crest when it is operating on the overburden bench. The distance from the dragline centerline to the interburden crest is calculated as shown in Equation 5.16 with the value being stored in D1. If DOR < D1, an error message is written to the screen and the user must change some variables before being allowed to calculate any further. Otherwise the dragline has enough reach and then is checked to see if the spoil volume will spill onto the interburden bench.

$$D1 = PO{\cdot}D + \frac{TOB}{\tan(\beta)} + WSB \tag{5.16}$$

$$D2 = D1 + \frac{TIB}{\tan(\alpha)} \tag{5.17}$$

$$D3 = D2 + WP \tag{5.18}$$

This check is accomplished by using three possible regions for spoiling. These regions are bounded by the three distances (D1, D2 and D3) shown in Figure 5.9. When the spoil does not ride up the previous spoil pile, the maximum overburden volume (VOLMAX) that can be spoiled at a given dragline operating radius (DOR) is defined in Equation 5.20.

$$DX = DOR - D1 \tag{5.19}$$

$$VOLMAX = \frac{(TIB + TBC + DX \cdot \tan(\theta))^2}{\tan(\theta)} - \frac{(TBC + TIB)^2\left(\frac{1}{\tan(\theta)} + \frac{1}{\tan(\alpha)}\right)}{2} + \frac{(TBC)^2}{2 \cdot \tan(\theta)} \tag{5.20}$$

If the maximum spoil volume rides up the previous spoil pile, VOLMAX is reduced by the following volume:

$$\frac{\tan(\theta)}{4}\left[2\frac{TIB + TBC + DX \cdot \tan(\theta)}{\tan(\theta)} - (TBC + TIB)\left(\frac{1}{\tan(\theta)} + \frac{1}{\tan(\alpha)}\right) + \frac{TBC}{\tan(\theta)} - WP\right]^2 \tag{5.21}$$

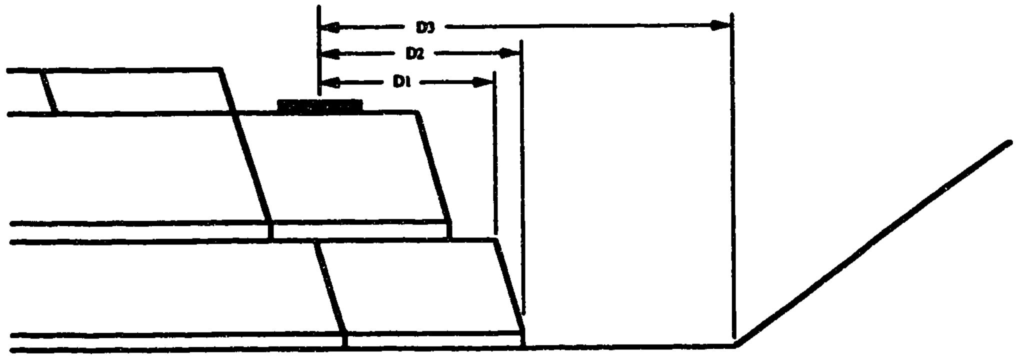


Figure 5.9 Distances used to bound spoil region in REH2

If the overburden spoil (VOBSP) is smaller than VOLMAX, the text in Figure 5.10 appears on the screen. The user has three alternatives at this point: (1) return to the input screen (2) calculate and plot the range diagram with only the overburden spoil on it or (3) calculate and plot the final range diagram which includes all of the spoil. If (1) is selected, the user is returned to the input screen and can change any of the input variables again. If (2) is selected, the user is prompted for an output device selection so that a range diagram can be plotted. This screen is displayed in Figure 5.11. If (3) is selected, the user is again prompted for an output device selection, but a rehandle volume and extended bench width are also printed to the screen. Figure 5.12 shows this screen. An error message would have been displayed on the screen if VOBSP was greater than VOLMAX.

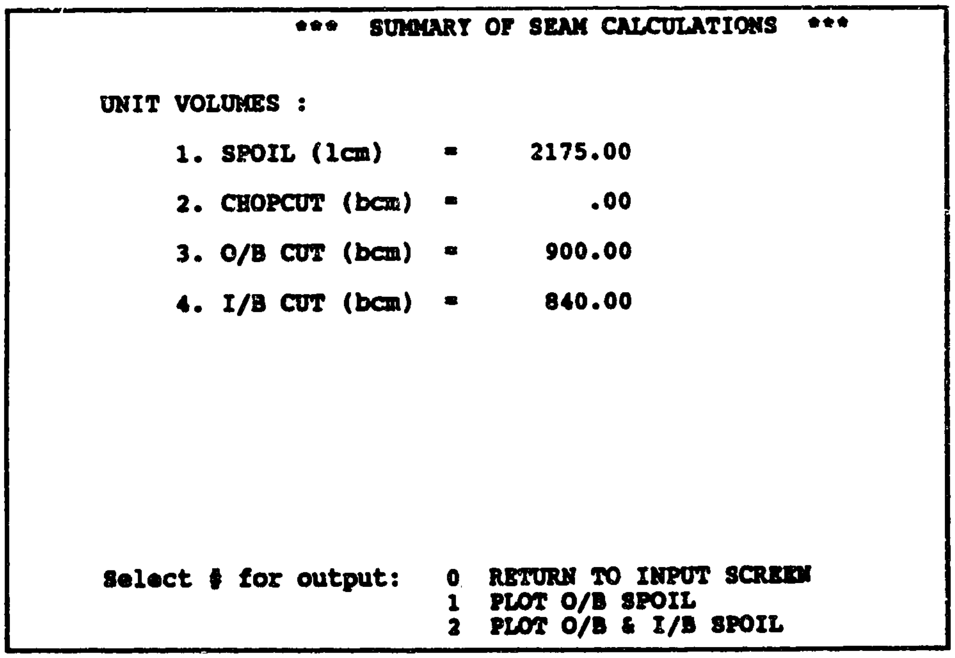
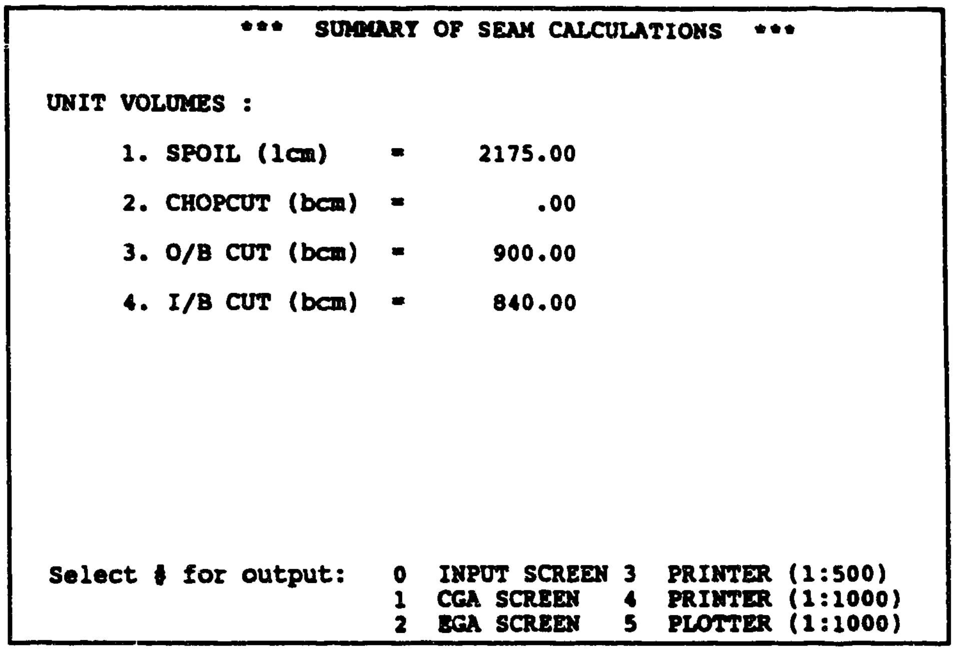
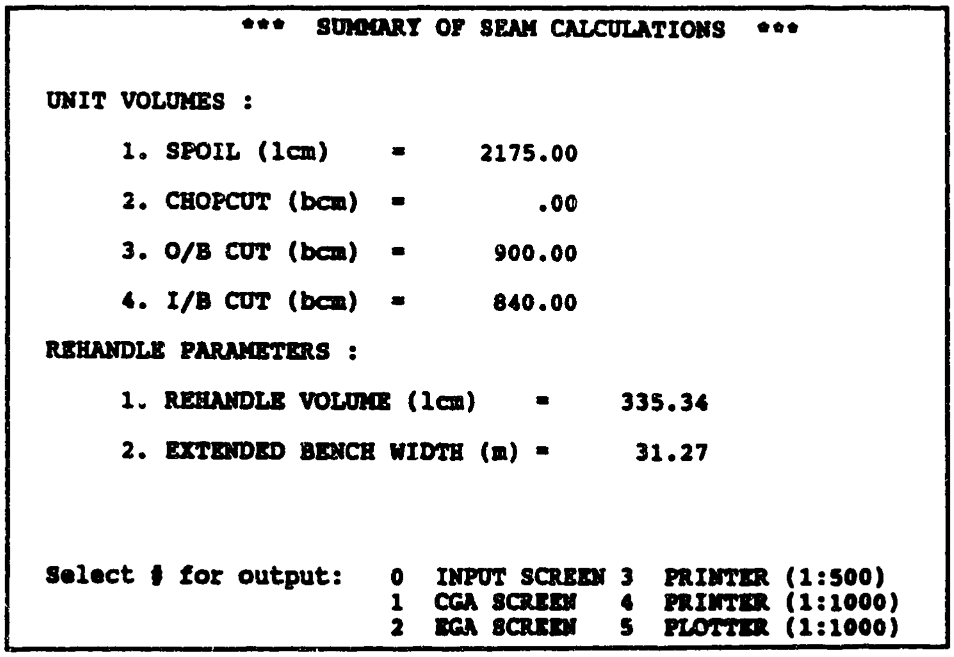

```
                    ***  SUMMARY OF SEAM CALCULATIONS  ***

     UNIT VOLUMES :
          1. SPOIL (lcm)      =     2175.00
          2. CHOPCUT (bcm)    =         .00
          3. O/B CUT (bcm)    =      900.00
          4. I/B CUT (bcm)    =      840.00







     Select # for output:     0  RETURN TO INPUT SCREEN
                              1  PLOT O/B SPOIL
                              2  PLOT O/B & I/B SPOIL
```

Figure 5.10 First REH2 output screen



```
                  ***  SUMMARY OF SEAM CALCULATIONS  ***

UNIT VOLUMES :

     1. SPOIL (lcm)     =    2175.00

     2. CHOPCUT (bcm)   =        .00

     3. O/B CUT (bcm)   =     900.00

     4. I/B CUT (bcm)   =     840.00





Select # for output:   0  INPUT SCREEN 3  PRINTER (1:500)
                       1  CGA SCREEN   4  PRINTER (1:1000)
                       2  EGA SCREEN   5  PLOTTER (1:1000)
```

Figure 5.11 Overburden output screen



```
                  ***  SUMMARY OF SEAM CALCULATIONS  ***

UNIT VOLUMES :

     1. SPOIL (lcm)     =    2175.00

     2. CHOPCUT (bcm)   =        .00

     3. O/B CUT (bcm)   =     900.00

     4. I/B CUT (bcm)   =     840.00

REHANDLE PARAMETERS :

     1. REHANDLE VOLUME (lcm)      =     335.34

     2. EXTENDED BENCH WIDTH (m) =       31.27



Select # for output:   0  INPUT SCREEN 3  PRINTER (1:500)
                       1  CGA SCREEN   4  PRINTER (1:1000)
                       2  EGA SCREEN   5  PLOTTER (1:1000)
```

Figure 5.12 Overburden and interburden output screen

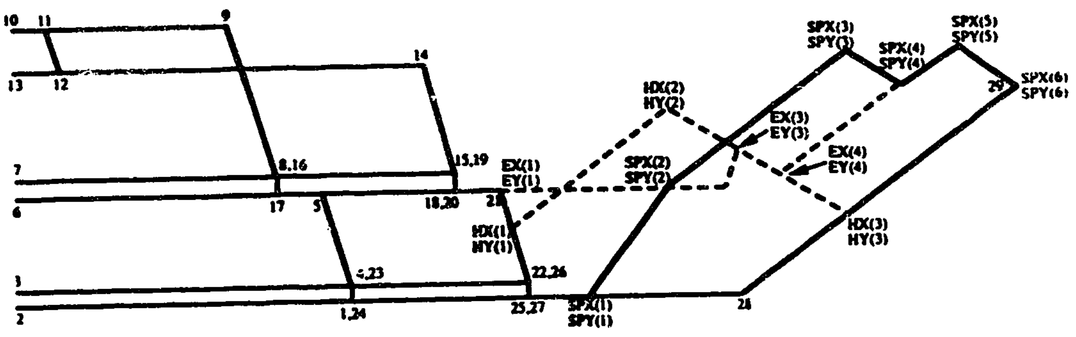
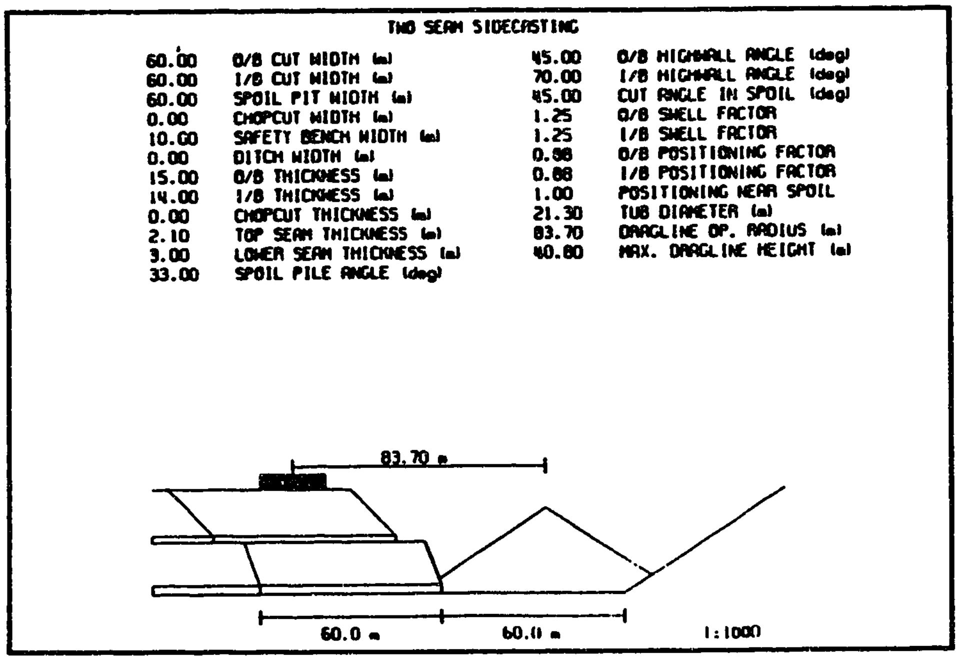
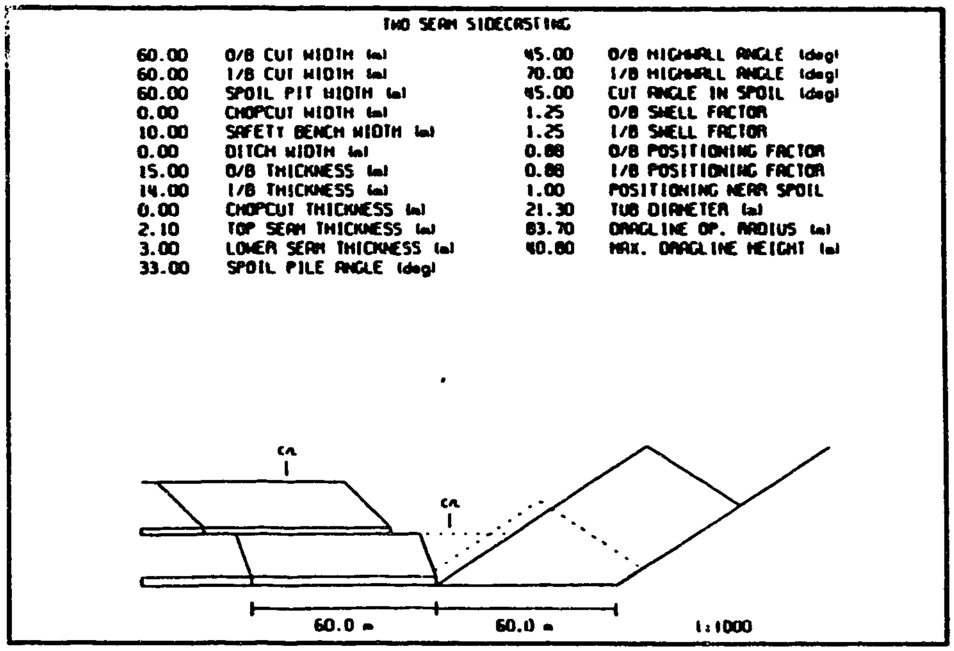


Figure 5.13 Point locations for REH2



```
                         TWO SEAM SIDECASTING

60.00   O/B CUT WIDTH (m)               45.00   O/B HIGHWALL ANGLE (deg)
60.00   I/B CUT WIDTH (m)               70.00   I/B HIGHWALL ANGLE (deg)
60.00   SPOIL PIT WIDTH (m)             45.00   CUT ANGLE IN SPOIL (deg)
0.00    CHOPCUT WIDTH (m)               1.25    O/B SWELL FACTOR
10.00   SAFETY BENCH WIDTH (m)          1.25    I/B SWELL FACTOR
0.00    DITCH WIDTH (m)                 0.88    O/B POSITIONING FACTOR
15.00   O/B THICKNESS (m)               0.88    I/B POSITIONING FACTOR
14.00   I/B THICKNESS (m)               1.00    POSITIONING NEAR SPOIL
0.00    CHOPCUT THICKNESS (m)           21.30   TUB DIAMETER (m)
2.10    TOP SEAM THICKNESS (m)          83.70   DRAGLINE OP. RADIUS (m)
3.00    LOWER SEAM THICKNESS (m)        40.80   MAX. DRAGLINE HEIGHT (m)
33.00   SPOIL PILE ANGLE (deg)
```

Figure 5.14 REH2 graphical output for overburden



```
                              TWO SEAM SIDECASTING
60.00   O/B CUT WIDTH (m)                     45.00   O/B HIGHWALL ANGLE (deg)
60.00   I/B CUT WIDTH (m)                     70.00   I/B HIGHWALL ANGLE (deg)
60.00   SPOIL PIT WIDTH (m)                   45.00   CUT ANGLE IN SPOIL (deg)
0.00    CHOPCUT WIDTH (m)                     1.25    O/B SWELL FACTOR
10.00   SAFETY BENCH WIDTH (m)                1.25    I/B SWELL FACTOR
0.00    DITCH WIDTH (m)                       0.88    O/B POSITIONING FACTOR
15.00   O/B THICKNESS (m)                     0.88    I/B POSITIONING FACTOR
14.00   I/B THICKNESS (m)                     1.00    POSITIONING NEAR SPOIL
0.00    CHOPCUT THICKNESS (m)                 21.30   TUB DIAMETER (m)
2.10    TOP SEAM THICKNESS (m)                83.70   DRAGLINE OP. RADIUS (m)
3.00    LOWER SEAM THICKNESS (m)              40.80   MAX. DRAGLINE HEIGHT (m)
33.00   SPOIL PILE ANGLE (deg)
```

Figure 5.15 REH2 graphical output for overburden and interburden

The coordinates in Figure 5.13 are calculated in order that scaled graphical output may be made. Figures 5.14 and 5.15 are examples of the pit configurations sent to the printer by the overburden and total spoil selections respectively. After discussion with engineers at the Obed mine site, it was agreed to use a full extended bench on the interburden if one was required. Since the stacking height is a major concern when operating the dragline on the interburden bench, REH2 can use up to two spoil peaks in order to lower the spoil pile height. Two spoil peaks can be achieved by operating from two different positions on the extended bench. If two peaks can not provide adequate spoil area, an error message is written to the screen and the user is then returned to the input screen. Another assumption in REH2 is that the total spoil must have a peak above the elevation of the interburden bench. This is only done to accommodate the manner in which calculations were performed. This is also a fairly small volume and should not restrict its use in the Obed mine operating procedures.

As REH2 attempts to be the most realistic dragline routine of the four, it also has many more possible pit configurations. The numerous possible pit c nfigurations are documented more fully in Appendix D.

While all of the programs have common output options, REH2 the additional option of plotting the overburden by itself. This was done in order that if the Obed engineers decide to change their mining method, they can manually modify these range diagrams to accommodate different interburden handling procedures. The main advantage is that the engineer is starting with a dragline pit which is already drawn to scale.

5.3 Summary

A large number of additional constraints make the REH2 subroutine the most difficult model to program in an easily understood manner. This is why Appendix D has been allocated for the description of the many possible pit configurations. SEAM2, on the other hand, is more easily followed, but is less applicable to the operations at Obed.

6. Probabilistic Design

6.1 Introduction

Once a dragline pit geometry has been selected, it is desirable to perform further analysis in order that the sensitivity of a dependent variable can be observed with respect to different design variables. When many component variables exist within a function, the practical methods for estimating system variability are limited to approximate techniques such as Monte Carlo simulation or moment estimation. Exact methods, such as the transformation of variables, become unfeasible as systems become complex.

Probabilistic design is considered here as a method of quantifying uncertainty within the design to provide a logical and systematic analysis of uncertainty. Deterministic models are often inadequate for designing complex systems because component differences or fluctuations in operating conditions are not taken into account. A common method of dealing with uncertainty in engineering design is to either implement a high factor of safety or combine worst cases. For structures in many civil engineering projects, this may be acceptable since relatively high probabilities of failure are not tolerated and high safety factors are used. In mining situations, however, designs are not usually permanent and the cost associated with high safety factors would make many projects unfeasible.

6.2 General Concepts

Design parameter values are never exactly known due to a natural randomness, imprecise measurements, or inaccurate assumptions. Parameters that exhibit such variations will here be termed random variables and will be represented by a probability density function. Continuous random variables result when dealing with measured, rather than counted data. The random variables discussed within this thesis will be considered continuous since they all can be measured or interpreted as real values. The probability distribution of a continuous random variable can always be described by its cumulative distribution function. The cumulative function $F_X(x)$ is equal to the probability that the random variable X will have a value less than or equal to x. That is

$$F_X(x) = P(X \le x) \tag{6.1}$$

A probability density function $f_X(x)$ is the first derivative of the cumulative function $F_X(x)$ and therefore $F_X(x)$ is

$$F_X(x) = \int_{-\infty}^{x} f_X(y)\, dy \tag{6.2}$$

When considering a function result Y of a continuous random variable X such that $Y = g(X)$, the expectation operator is defined

$$E(Y) = \int_{-\infty}^{+\infty} g(x)\, f_X(x)\, dx \tag{6.3}$$

The probability density function, $f_X(x)$ is usually not well known in practice. For this reason, the moments of a function are usually approximated by other methods rather than calculated by the exact method of inverse transformation. The nth order moment is defined as

$$E(X^n) = \int_{-\infty}^{+\infty} x^n\, f_X(x)\, dx \tag{6.4}$$

The first moment is often referred to as the mean while the second central moment is termed the variance. The second central moment is a measure of dispersion where deviations are measured with respect to the expected value.

$$\begin{aligned} VAR(X) &= E[(X-E(X))^2] \\ &= E(X^2) - (E(X))^2 \end{aligned} \tag{6.5}$$

The joint second central moment of two random variables X and Y is called the covariance of X and Y and is defined as

$$\begin{aligned} COV(X,Y) &= E[(X-E(X))\,(Y-E(Y))] \\ &= E(XY) - E(X)E(Y) \end{aligned} \tag{6.6}$$

If the covariance of X and Y is equal to zero, then the X and Y are statistically independent.

6.3 Moment Appr[illegible] Function of a Single Random Variable

In practice, the density [illegible]ion $f_X(x)$ is often unknown and data may be limited to the first moment (m[illegible] second central moment (variance) of the random variable. The integrations may be difficult to perform even if the density function is known, so it is often useful to obtain the approximate first moment and second central moment values of the function using a Taylor series expansion about the expected value of the random variable. When terms are only retained up to second order, this Taylor series expansion is defined as follows

$$Y \approx g(E(X)) + (X-E(X))\frac{dg}{dX} + \frac{1}{2}(X-E(X))^2\frac{d^2g}{dX^2} \tag{6.7}$$

When the moments of the dependent variable are estimated, the derivatives are evaluated at the expected value of the random variable. The second order approximation of the first moment is

$$E(Y) = g(E(X)) + \frac{1}{2} VAR(X) \frac{d^2g}{dX^2} \tag{6.8}$$

The second order approximation of the second central moment is

$$\begin{aligned} VAR(Y) = {} & VAR(X)\left(\frac{dg}{dX}\right)^2 + E(X-E(X))^3 \frac{dg}{dX}\frac{d^2g}{dX^2} \\ & + \frac{1}{4}E(X-E(X))^4\left(\frac{d^2g}{dX^2}\right)^2 - \frac{1}{4}(VAR(X))^2\left(\frac{d^2g}{dX^2}\right)^2 \end{aligned} \tag{6.9}$$

however, the third and fourth order terms are usually dropped [16,43] and the function variance is estimated as follows,

$$VAR(Y) = VAR(X)\left(\frac{dg}{dX}\right)^2 \tag{6.10}$$

The third order central moment of the function is a measure of how symmetric the distribution of the function is, so if a fairly symmetric function distribution exists, the third order central moment will be approximately equal to zero. If the third central moment term is negative, the function distribution is said to be skewed to the left. Alternatively, the function distribution is skewed to the right if the third central moment is positive. The fourth central moment is related to the kurtosis (degree of peakedness) of the function distribution. Equations 6.8 and 6.10 are often used since they provide satisfactory results without having to calculate third or higher moment and derivative terms with respect to the random variable. If the function is linear, the second and higher derivative terms will be equal to zero and therefore exactly agree with 6.8 and 6.10.[2,16,43]

6.4 Moment Approximation of a Function of Multiple Random Variables

When Y is a function of several random variables ($Y=g(X_1,X_2,...,X_n)$), the Taylor series expansion about the expected variate values is

$$Y = g(E(X_1),E(X_2),...E(X_n)) + \sum_{i=1}^{n}(X_i-E(X_i))\frac{\partial g}{\partial X_i} + \frac{1}{2}\sum_{i=1}^{n}(X_i-E(X_i))^2\frac{\partial^2 g}{\partial X_i^2} + \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(x_i-E(x_i))(x_j-E(x_j))\frac{\partial^2 g}{\partial X_i \partial X_j} \qquad (6.11)$$

Once again, only the second order terms have been retained in the Taylor series expansion. By using the expectation operator, the second order approximation of the first moment with uncorrelated variables is

$$E(Y) = g(E(X_1),E(X_2),...,E(X_n)) + \frac{1}{2}\sum_{i=1}^{n} VAR(X_i)\frac{\partial^2 g}{\partial X_i^2} \qquad (6.12)$$

while the second order approximation of the second central moment with uncorrelated variables is defined in equation 6.13.

$$VAR(Y) = \sum_{i=1}^{n} VAR(X_i)\left(\frac{\partial g}{\partial X_i}\right)^2 + \sum_{i=1}^{n} E(X_i-E(X_i))^3\frac{\partial g}{\partial X_i}\frac{\partial^2 g}{\partial X_i^2} + \frac{1}{4}\sum_{i=1}^{n} E(X_i-E(X_i))^4\left(\frac{\partial^2 g}{\partial X_i^2}\right)^2 - \frac{1}{4}\sum_{i=1}^{n}(VAR(X_i))^2\left(\frac{\partial^2 g}{\partial X_i^2}\right)^2 \qquad (6.13)$$

A satisfactory estimate of the second central moment will again be possible without the third and higher order terms of the basic variates so that the variance can be defined as shown in equation 6.14.

$$VAR(Y) = \sum_{i=1}^{n} VAR(X_i)\left(\frac{\partial g}{\partial X_i}\right)^2 \qquad (6.14)$$

6.5 Transformation of Variables

The transformation of variable technique is exact in that it gives explicit analytical solutions to the general problem of probabilistic analysis. The simplest case is when there is only one independent variable such that y=g(x). The infinitesimal event in x is related to that in y as shown in Figure 6.1.

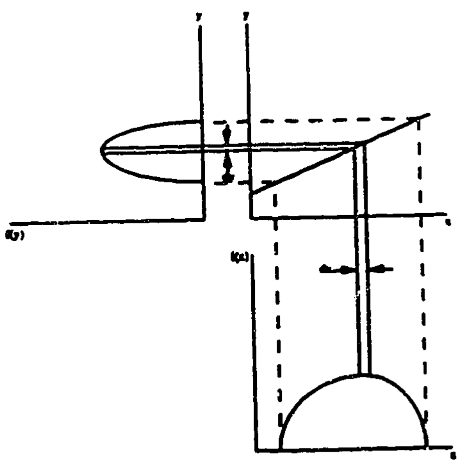


Figure 6.1 Transformation of variables (after [43])

The events of x lying in dx and y in dy must occur together and have the same probability such that,

$$f(y)dy = f(x)dx \tag{6.15}$$

and

$$f(y) = \frac{f(x)}{\left|\frac{dy}{dx}\right|} \tag{6.16}$$

The absolute value sign is used to ensure that f(y) is always positive. When $x = g^{-1}(y)$, the properly derived density function is defined as

$$f_Y(y) = f_X(g^{-1}(y)) \left| \frac{dg^{-1}(y)}{dy} \right| \tag{6.17}$$

Many continuous distribution functions F(x) do not have analytic inverse functions. Therefore, the main difficulty with the inverse transformation method lies in finding the inverse transformation $F^{-1}(x)$. When more than one independent variable exists in the function, the value of $F^{-1}(x)$ becomes increasingly difficult to determine.

6.6 Monte Carlo Simulation

Monte Carlo simulation is another way of obtaining information about a function from component data. Monte Carlo simulation consists of calculating many sample systems and evaluating the distribution of the results. If the distribution of each component variable is known, it is possible to obtain synthetic measurements on these components and then combine these measurements to determine the function value. It is necessary to simulate a function many times in order to achieve adequate accuracy of the estimates. The availability of high-speed computers has led to the popularization of Monte Carlo procedures. Figure 6.2 shows a flow chart of the Monte Carlo simulation method.

Monte Carlo simulation involves generating random values for each of the component distributions and then calculating the function estimate. The usual method of obtaining random samples from prescribed distributions is to first generate uniform random samples between 0 and 1 and then transform the uniform sample into a new sample with the desired distribution. A good summary of methods and equations for representing random samples from different distributions is given by Pritaker.[37] The uniform random numbers between 0 and 1 are usually determined by a deterministic mathematical expression rather than by some physical mechanism, so the resulting values are often called "pseudorandom".[16]

As with the other methods discussed, the validity of the answers depends on the adequacy of the input data. Careful attention must be provided when the distributions of the input data are estimated so that a reliable result of the function is simulated.

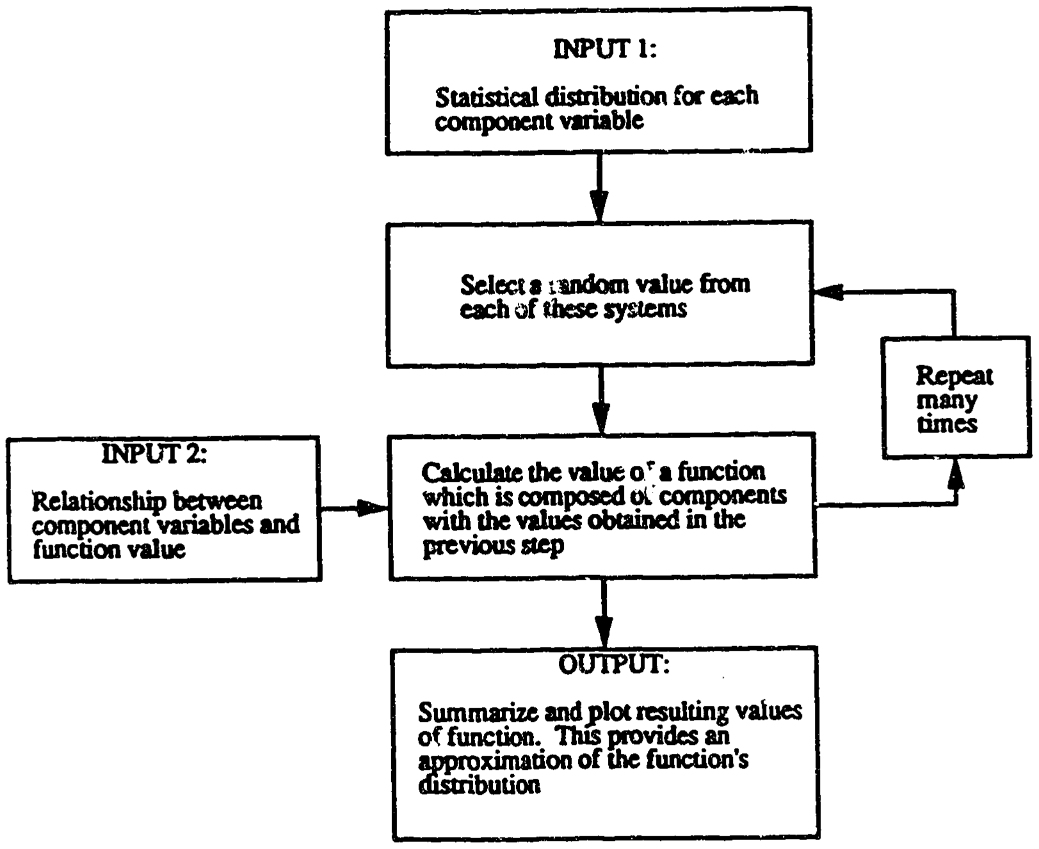

INPUT 1:
Statistical distribution for each component variable
Select a random value from each of these systems
Repeat many times
INPUT 2:
Relationship between component variables and function value
Calculate the value of a function which is composed of components with the values obtained in the previous step
OUTPUT:
Summarize and plot resulting values of function. This provides an approximation of the function's distribution

Figure 6.2 Flow chart of Monte Carlo simulation (after [16])

6.7 Application of Probabilistic Methods

A common method of sensitivity analysis is to individually vary each component variable at a baseline $\pm$ 10% and observe how much of an effect this has on the function estimate. This method can be used as a rough check on whether the function estimate is relatively insensitive or not to changes in different input variables. An input variable may also be varied over a range of values to observe changes in the function estimates. It must be clearly understood that these two methods do provide additional information which may be valuable to the user, however, they do not attempt to quantify the effect of the inherent randomness of the random variables and they completely ignore any combined effects of variables.

The methods presented in sections 6.2 to 6.5 attempt to explain how the random variables effect the function estimate. The remainder of this section will be used to discuss the advantages and disadvantages of these probabilistic methods.

One important advantage of the Taylor series expansion method is that it does not require that a frequency distribution model be estimated for each independent random variable. Both the inverse transformation and Monte Carlo methods have this requirement. A poor estimate of the component distributions may have a direct effect upon the accuracy of the function estimate and its statistical moments.

It is seldom the case that enough information is available to determine whether or not variables are correlated. This is why analyses are often carried out assuming that stochastic variables are independent and covariances are zero. For this reason, stochastic variables will be regarded as independent in this thesis.

The omission of higher order terms in the Taylor series expansion about the expected value causes the method of moment approximation to be an approximate procedure. In practical problems, however, the estimation of the mean and standard deviation of a component variable may introduce more error than the omission of the higher order terms.[16] This method is very dependent upon the ease of which partial derivatives may be obtained from a function. Numerical estimation may eliminate the problem of obtaining derivatives, but it is subject to instability when functions change rapidly or pass over discontinuities. If difficulties are encountered when trying to obtain derivatives, the Rosenblueth method of point estimates [17,35,39] may be used as a substitute since it does not require taking derivatives. Rosenblueth's method is an approximate numerical integration approach which can be used to crudely estimate the moments of a function. Calculations are limited to evaluating the function values of the variables one standard deviation on either side of the mean. Calculations are therefore quite simple, but no information is provided about individual variable contributions to the moments as in the Taylor Series methods.

The transformation of variables provides an exact result provided that the component distributions are known. The main problem with this method is that it is restricted to relatively simple problems and this limits its application to most engineering models. This method only provides a result as exact as the estimator of the component distributions.

Monte Carlo simulation may have more appeal because it is more easily understood than the moment approximation technique. It is a very flexible method and it is easily applied to complex functions, but it is impossible to determine quantitatively which variables dominate the results. In contrast, the moment approximation method gives the user a direct quantitative measure of importance of each component variable by examining the individual portions of equations of equations 6.12 and 6.14. This is a desirable attribute of a method because corrective actions may be undertaken on those components that contribute heavily on the estimate of the function variance.

There is no "optimal" method for the determination of statistical estimates of any function of random variables. Choosing a good method for a given analysis depends on the nature and complexity of the problem and what is required from the analysis. The results from a chosen method may often be complimented by the additional information from another method. For example, the Monte Carlo method may be used to compliment the results of the moment approximation technique. If only the mean and standard deviation of the function are estimated using the moment approximation technique, Monte Carlo simulation can provide information about how well the function distribution fits the normal distribution or any other distribution.

In a dragline analysis, the estimates of several different dependent variables may be determined with respect to independent random variables. Rehandle, operating radius, cut width and even owning and operating costs could be used as dependent variables in one of the above probabilistic analysis methods.

6.8 VARSIM Library Description

A general program was written to provide numerical variance and bias component analysis of a user-supplied function. An option to simulate the function using normal or lognormal variates is also provided in the program. As with the dragline programs, an interactive user environment has been used so that the variable values can be manipulated until the user is satisfied that the mean and standard deviation values of the independent variables are representative and/or economically achievable for the function.

By putting all of the compiled routines into a library called VARSIM, the user only has to write a Fortran function routine and then link its object code with the library. A sample function routine is shown in appendix E. Sample input and output are also documented in chapter 7. All of the source code for the VARSIM library is fully documented in Appendix C.

The VARSIM library contains numerous error checks during execution. When outputs are written to files, the user is notified if the file already exists and has the option of overwriting the file or entering a new file name. When entering new mean and standard deviation values for the independent variables, only the number and decimal keys on the keyboard will respond to being pressed. The only exceptions to this is that the ESC and F1 keys may also be pressed. The ESC key allows the user to quit the program and the F1 key allows the user to restore the previous value. Only direct decimal notation is allowed when entering data.

Figure 6.3 shows a flow chart of the program's execution. Once the mean and standard deviation values of the variables have been changed, the user can choose to simulate the function or to perform variance and bias component analysis using the method of moment approximation. After either of these options has been completed, the user is returned to the edit screen and can modify the variable values again if desired.

The component analysis requires input values of mean and standard deviation for each variable. In order to determine a second order Taylor series estimate of the expected function value and a first order Taylor series estimate of the function's standard deviation, the first and second partial derivatives are estimated numerically for each variable. The methodology described in section 6.4 applies to VARSIM because multiple variables can be used in the function.

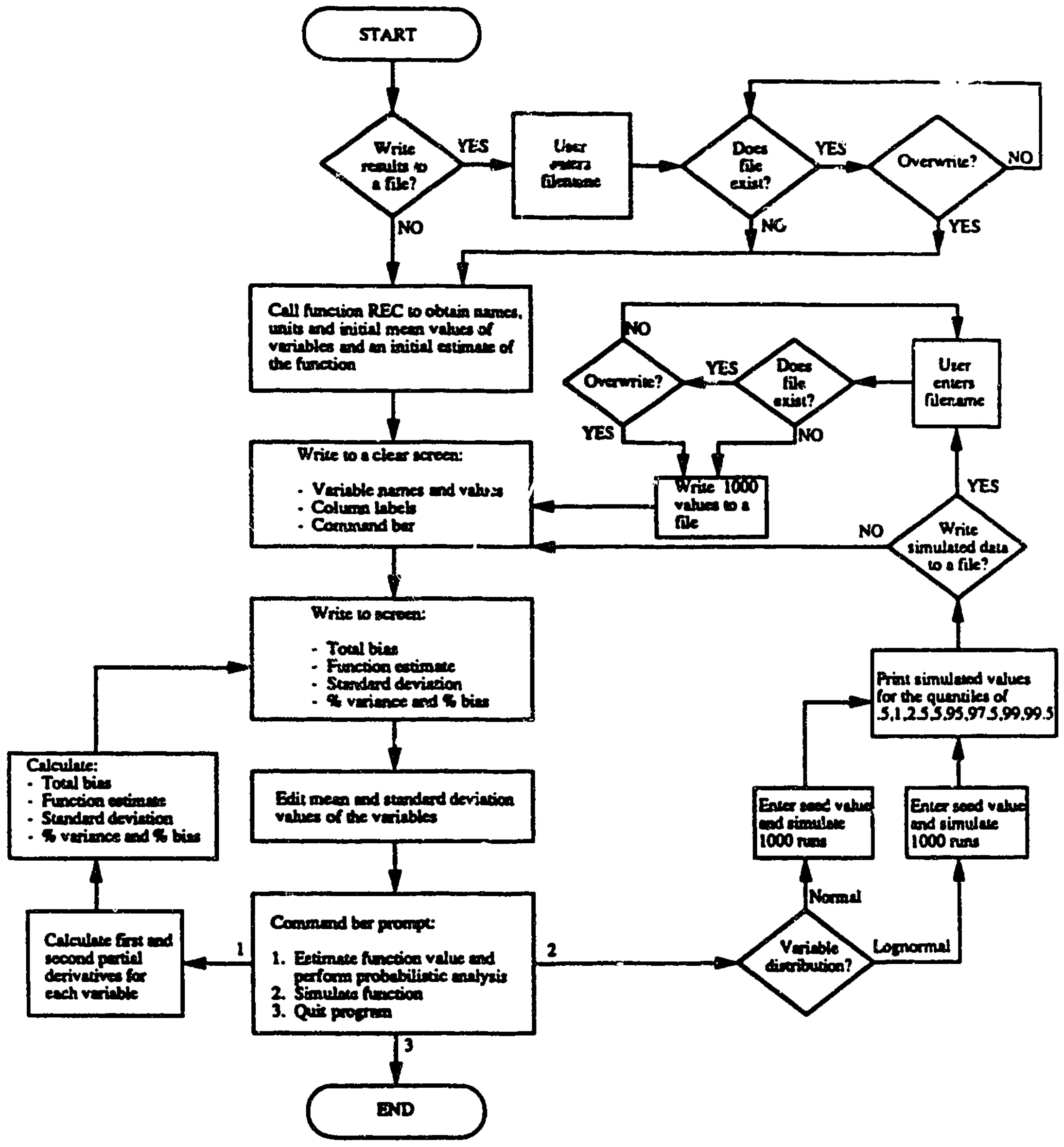

START
Write results to a file?
YES
NO
User enters filename
Does file exist?
YES
NO
Overwrite?
NO
YES
Call function REC to obtain names, units and initial mean values of variables and an initial estimate of the function
Write to a clear screen:
- Variable names and values
- Column labels
- Command bar
Write to screen:
- Total bias
- Function estimate
- Standard deviation
- % variance and % bias
Edit mean and standard deviation values of the variables
Command bar prompt:
1. Estimate function value and perform probabilistic analysis
2. Simulate function
3. Quit program
1
2
3
END
Calculate first and second partial derivatives for each variable
Calculate:
- Total bias
- Function estimate
- Standard deviation
- % variance and % bias
Variable distribution?
Normal
Lognormal
Enter seed value and simulate 1000 runs
Enter seed value and simulate 1000 runs
Print simulated values for the quantiles of .5,1,2.5,5,95,97.5,99,99.5
Write simulated data to a file?
YES
NO
User enters filename
Does file exist?
YES
NO
Overwrite?
NO
YES
Write 1000 values to a file

Figure 6.3 Flow chart of VARSIM

The bias component of the expected function value will be considered first. The bias component is the second derivative term in equation 6.12 and is defined as

$$\mathit{BIAS\ DUE\ TO\ VARIABLE\ i} = \frac{1}{2}\ VAR(X_i)\frac{\partial^2 g}{\partial X_i^2} \qquad (6.18)$$

$$\mathit{TOTAL\ BIAS\ OF\ FUNCTION} = \frac{1}{2}\sum_{i=1}^{N}\ VAR(X_i)\frac{\partial^2 g}{\partial X_i^2} \qquad (6.19)$$

$$\%\ \mathit{BIAS\ FROM\ VARIABLE\ i} = \frac{\mathit{BIAS\ FROM\ VARIABLE\ i}}{\mathit{TOTAL\ BIAS}}\cdot 100\% \qquad (6.20)$$

The % bias values are printed to the screen in a column beside the variable values and the total bias value is printed below the variable names with the function estimate. A large standard deviation of an independent variable may cause the total bias term to be large. If the total bias value is large, the estimated expected value of the function will be considerably different from an estimate of the function using only the mean values of the independent variables. The function estimate is obtained using equation 6.12.

The variance components are described in equation 6.14 where

$$\mathit{VARIANCE\ DUE\ TO\ VARIABLE\ i} = VAR(X_i)\left(\frac{\partial g}{\partial X_i}\right)^2 \qquad (6.21)$$

$$\mathit{TOTAL\ VARIANCE\ OF\ FUNCTION} = \sum_{i=1}^{N}\ VAR(X_i)\left(\frac{\partial g}{\partial X_i}\right)^2 \qquad (6.22)$$

$$\%\ \mathit{VARIANCE\ FROM\ VARIABLE\ i} = \frac{\mathit{VARIANCE\ FROM\ VARIABLE\ i}}{\mathit{TOTAL\ VARIANCE}}\cdot 100\% \qquad (6.23)$$

These values may show that a certain independent variable contributes more than others to the variance of the function. A large standard deviation of one of the variables may actually contribute very little to the function variance. If further sampling is allowed to improve the estimate of a function value, the component variance values can provide important information about which variables could give the most improvement.

The simulation option in VARSIM allows the user to simulate the function with either normal or lognormal variables. VARSIM was intended to be a general program for many possible functions, so a practical approach to simulation was to allow all of the random variables to have the same distribution. The normal distribution is probably used more often than any other distribution, while the lognormal distribution allows all of its distribution values to be positive. More distributions could have been made available, but information is often not available for each variable and the purpose of the simulation option is mainly to see how applicable the normal distribution is to the function. If the simulated results show that the function distribution is not highly skewed, then the confidence of obtaining certain function values may be estimated based upon the normal distribution.

Chapter 7. Example Calculations Using VARSIM

This chapter contains an example of how VARSIM may be applied with the program DRAG. Hand calculations are also shown to verify the calculations performed by VARSIM. Subroutine SEAM1, as described in chapter 4, can be used to calculate the operating radius and height requirements of a dragline which operates above a single coal seam and requires no rehandle. The following example looks at how the input values affect the required dragline operating radius. The values in Table 7.1 are used in SEAM1 and an operating radius of 80.80 metres is required so that no rehandle occurs. The coal seam thickness is arbitrarily set equal to 2 metres since a value is required for SEAM1. However, this value is not used when calculating the operating radius in this case.

Table 7.1 Variable Description for SEAM1 Example

VARIABLE NAME	UNITS	VARIABLE	MEAN
cut width	m	WC	50
spoil pit width	m	WP	40
chopcut width	m	WCH	50
overburden thickness	m	TOB	15
chopcut thickness	m	TCH	5
spoil pile angle	degrees	θ	35
overburden highwall angle	degrees	β	70
overburden swell factor	(decimal)	SWOB	1.20
overburden positioning factor	(decimal)	PO	0.75
tub diameter	m	D	30

The first step in VARSIM is to find the first and second partial derivatives of each independent variable with respect to the operating radius. The required operating radius is represented as

$$OROB = PO \cdot D + \frac{TOB}{\tan\beta} + \frac{SWOB}{WP\ \tan\theta}(WC \cdot TOB + TCH \cdot WCH) + \frac{WP}{4} \tag{7.1}$$

The number of digits in the following calculations are not all significant, but are purposely carried for verification of VARSIM calculations. The mean values from Table 7.1 were substituted into Equation 7.1 and OROB was estimated as 80.80399372 meters using a calculator. The first partial derivatives are as follows,

$$\frac{\partial\ OROB}{\partial\ WC} = \frac{TOB \cdot SWOB}{WP\ \tan\theta} = 0.642666603$$

$$\frac{\partial\ OROB}{\partial\ WP} = \frac{-SWOB}{WP^2\ \tan\theta}(WC \cdot TOB + TCH \cdot WCH) + 0.25 = -0.821111005$$

$$\frac{\partial\ OROB}{\partial\ WCH} = \frac{TCH \cdot SWOB}{WP\ \tan\theta} = 0.214222201$$

$$\frac{\partial\ OROB}{\partial\ TOB} = \frac{1}{\tan\beta} + \frac{WC \cdot SWOB}{WP\ \tan\theta} = 2.506192244$$

$$\frac{\partial\ OROB}{\partial\ TCH} = \frac{WCH \cdot SWOB}{WP\ \tan\theta} = 2.142222010$$

$$\frac{\partial\ OROB}{\partial\ \theta} = \frac{-SWOB}{WP\ \sin^2\theta}(WC \cdot TOB + TCH \cdot WCH) = -91.18820185$$

$$\frac{\partial\, OROB}{\partial\, \beta} = \frac{-TOB}{\sin^2\beta} = -16.98711497$$

$$\frac{\partial\, OROB}{\partial\, SWOB} = \frac{WC \cdot TOB + TCH \cdot WCH}{WP\ \tan\theta} = 35.70370017$$

$$\frac{\partial\, OROB}{\partial\, PO} = D = 30$$

$$\frac{\partial\, OROB}{\partial\, D} = PO = 0.75$$

All of the second partial derivatives are equal to zero except for the following three,

$$\frac{\partial^2 OROB}{\partial\, WP^2} = \frac{2 \cdot SWOB}{WP^3\ \tan\theta}(WC \cdot TOB + TCH \cdot WCH) = 0.053555550$$

$$\frac{\partial^2 OROB}{\partial\, \theta^2} = \frac{2 \cdot SWOB}{WP\ \tan\theta\ \sin^2\theta}(WC \cdot TOB + TCH \cdot WCH) = 260.4604975$$

$$\frac{\partial^2 OROB}{\partial\, \beta^2} = \frac{2 \cdot TOB}{\tan\beta\ \sin^2\beta} = 12.36560843$$

The bias term in Equation 6.12 is calculated in Table 7.2 along with the individual component bias terms. The bias term (1.006692933) is combined with the mean OROB value (80.80399372) to obtain an estimate of the expected operating radius (81.81068665). The bias terms do not have to be positive values since the second derivative term is not squared.

Standard deviation values have been assumed to be equal to 10% of the mean values (coefficient of variation = 10%) for every variable except for the tub diameter. The tub diameter has been included in the input screen so that different tub sizes can be tried, but it is not treated as a random variable.

Table 7.2 Calculation of Bias Components by Calculator

X_i	δ^2OROB/ δX_i^2 (1)	σ_{Xi} (2)	bias component $(1)(2)^2/2$	% bias component
WC	0	5	0	0
WP	0.053555550	4	0.428444400	42.56
WCH	0	5	0	0
TOB	0	1.5	0	0
TCH	0	0.5	0	0
θ	260.4604975	$3.5\pi/180$	0.485962429	48.27
β	12.36560843	$7\pi/180$	0.092286104	9.17
SWOB	0	0.12	0	0
PO	0	0.075	0	0
D	0	0	0	0
TOTAL			1.006692933	100.00

The variance of OROB is calculated to be 96.29502437 in Table 7.3. The variance value represents the summation of individual variance components as represented by the general Equation 6.14. These component variance terms are always positive so their combined influence cannot be reduced by terms offsetting each other as can be the case with the bias terms. By taking the square root of this variance term, the standard deviation can be approximated as 9.813002821.

Table 7.3 Calculation of Variance Components by Calculator

X_i	$\delta OROB / \delta X_i$	σ_{Xi}	variance	% variance
			component	component
	(1)	(2)	$(1)^2(2)^2$	
WC	0.642666603	5	10.32550907	10.72
WP	-0.821111005	4	10.78757252	11.20
WCH	0.214222201	5	1.147278785	1.19
TOB	2.506192244	1.5	14.13224902	14.68
TCH	2.142222010	0.5	1.147278785	1.19
θ	-91.18820185	$3.5\pi/180$	31.02902490	32.22
β	-16.98711497	$7\pi/180$	4.307150705	4.47
SWOB	35.70370017	0.12	18.35646057	19.06
PO	30	0.075	5.0625	5.26
D	0.75	0	0	0
TOTAL			96.29502437	100.00

A summary of the estimates calculated by calculator is as follows

- bias term = 1.006692933
- expected OROB = 81.81068665
- standard deviation = 9.813002821

The estimates obtained using VARSIM are listed in Figure 7.1. The bias, expected OROB and standard deviation values obtained using a calculator agree exactly with the display accuracy of these estimates. The function REC listing is located in appendix E. Even though the final answers agree very well between VARSIM and the analytic values using a calculator, the partial derivative values obtained in VARSIM are not always correct for the angle variables. All of the other variables use the correct procedure to calculate partial derivatives. Trigonometric calculations in Fortran are performed using angles in radian measurements. The function REC has been set up to use angles with degree units of measure. This was done simply because most users prefer to work with angles in degree rather than radian units. The estimated bias and variance component values are not affected by this choice of units because VARSIM uses the same units for the input standard deviation value and the increment of X_i when calculating derivatives. These units cancel each other out when the variance and bias values are calculated. If the user decides to set up the function REC so that the user has to input radian values for the angles, then the derivative values will all be calculated correctly. If derivatives are desired to be correct throughout the program, the main program VARSM would have to be modified so that all angle values are always converted into radians prior to derivative calculation. This step was not included in the program so that it would remain more general and more numerically efficient.

```
 #          FACTOR                      MEAN      STD DEVIATION       % BIAS  % VAR

 1  cut width (m)                     50.000000     .00000000            .00    .00
 2  spoil pit width (m)               40.000000     .00000000            .00    .00
 3  chopcut width (m)                 50.000000     .00000000            .00    .00
 4  O/B thickness (m)                 15.000000     .00000000            .00    .00
 5  chopcut thickness (m)              5.0000000    .00000000            .00    .00
 6  spoil pile angle (deg)            35.000000     .00000000            .00    .00
 7  O/B highwall angle (deg)          70.000000     .00000000            .00    .00
 8  O/B swell factor                   1.2000000    .00000000            .00    .00
 9  O/B positioning factor             .75000000    .00000000            .00    .00
10  tub diameter (m)                  30.000000     .00000000            .00    .00
TOTAL BIAS IN                   required OROB    .0000000   meters
ESTIMATED                       required OROB   80.80399    meters
STANDARD DEV                    required OROB    .0000000   meters

 #          FACTOR                      MEAN      STD DEVIATION       % BIAS  % VAR

 1  cut width (m)                     50.000000     5.0000000           .00  10.72
 2  spoil pit width (m)               40.000000     4.0000000         42.56  11.20
 3  chopcut width (m)                 50.000000     5.0000000           .00   1.19
 4  O/B thickness (m)                 15.000000     1.5000000           .00  14.68
 5  chopcut thickness (m)              5.0000000    .50000000           .00   1.19
 6  spoil pile angle (deg)            35.000000     3.5000000         48.27  32.22
 7  O/B highwall angle (deg)          70.000000     7.0000000          9.17   4.47
 8  O/B swell factor                   1.2000000    .12000000           .00  19.06
 9  O/B positioning factor             .75000000    .75000000E-01       .00   5.26
10  tub diameter (m)                  30.000000     .00000000           .00    .00
TOTAL BIAS IN                   required OROB    1.006693   meters
ESTIMATED                       required OROB   81.81069    meters
STANDARD DEV                    required OROB    9.813003   meters
```

Figure 7.1 VARSIM component analysis output for SEAM1 example

In this example, the total bias is not very significant and there is no clearly dominant variable in the variance components. The spoil pile angle variable seems to have more influence on the total variance than the other variables, but it is not overwhelming. It is important to check for a dominant variable because of the sensitive nature of the derivative calculations.

If the user wants to make a statement about the function (OROB) concerning confidence limits, it is necessary to determine which distribution best describes it. The VARSIM simulation option allows the user to generate 1000 random function estimates using the density function of either the normal or lognormal distribution for all the variables. One thousand random samples of OROB were generated using each of the two density functions for the component values from Figure 7.1 and then summarized as in Figure 7.2.

```
NORMAL
DSEED =    .99876321000000D+08

QUANTILE     .5 % required OROB    58.907     meters
QUANTILE    1.0 % required OROB    60.651     meters
QUANTILE    2.5 % required OROB    64.503     meters
QUANTILE    5.0 % required OROB    66.555     meters
QUANTILE   95.0 % required OROB    100.52     meters
QUANTILE   97.5 % required OROB    104.57     meters
QUANTILE   99.0 % required OROB    108.82     meters
QUANTILE   99.5 % required OROB    112.70     meters

LOGNORMAL
DSEED =    .99876321000000D+08

QUANTILE     .5 % required OROB    58.677     meters
QUANTILE    1.0 % required OROB    60.227     meters
QUANTILE    2.5 % required OROB    64.443     meters
QUANTILE    5.0 % required OROB    66.688     meters
QUANTILE   95.0 % required OROB    100.53     meters
QUANTILE   97.5 % required OROB    104.44     meters
QUANTILE   99.0 % required OROB    109.33     meters
QUANTILE   99.5 % required OROB    113.03     meters
```

Figure 7.2 VARSIM simulation analysis output for SEAM1 example

The quantiles of the two different sample sets are very similar, so the choice of probability density functions for the variates did not make much difference in this case. Only the 1000 data values generated using normal variates will be considered further.

The PC-FIT software [44] was then used to see whether a normal or lognormal distribution would fit the data adequately. The first comparison was to observe the simulated data on a normal probability plot, take the natural logarithm of the data values, and then observe the normal probability plot again for the transformed data. These plots suggested that the lognormal distribution would represent the data better because it fit the straight line better on the plot.

PC-FIT was also used to perform a χ^2 goodness-of-fit test for the data. Using the default data class sizes and a 5% level of significance, the lognormal distribution was accepted as a good fit while the normal distribution was rejected. Since there is subjectivity in the choice of the significance level and the class size, the χ^2 goodness-of-fit test does not provide absolute information on the acceptance of a specific distribution. For this reason, another goodness-of-fit test was used to supplement the previous results.

The stabilized probability plot, proposed by Michael [30], was chosen because of its ease of interpretation. It uses a transformation to stabilize the variances of the plotted points. The abscissa axis value, r_i, and the ordinate axis value, s_i, are defined as follows

$$r_i = \frac{2}{\pi}\cdot arcsin\left(\sqrt{\frac{i-\frac{1}{2}}{n}}\right) \tag{7.2}$$

$$s_i = \frac{2}{\pi}\cdot arcsin\left(\sqrt{F_o\cdot\left(\frac{y_i-\mu}{\sigma}\right)}\right) \tag{7.3}$$

where n = number of data samples

y_i = ith random sample value

F_o = hypothesized distribution

Michael states that 0.95 acceptance regions can be plotted on the plots so that it is easy to see if the data falls outside of the acceptance region. Michael only provides the acceptance values for up to 100 samples, so these are the two lines that are plotted in Figures 7.3 and 7.4. For the 1000 simulated values of OROB, these acceptance regions must therefore be observed with caution. Similar to the previous normal probability plots, the goodness-of-fit is determined by the amount of departure from the straight line at 45 degrees from each axis. The lognormal distribution appears to provide the best fit again.

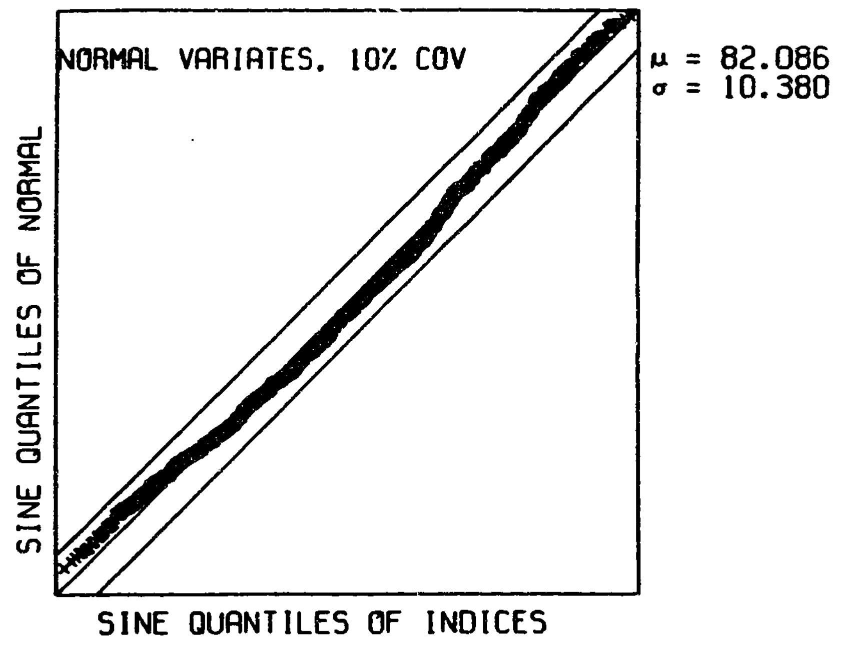


Figure 7.3 Stabilized probability plot using a hypothesized normal distribution

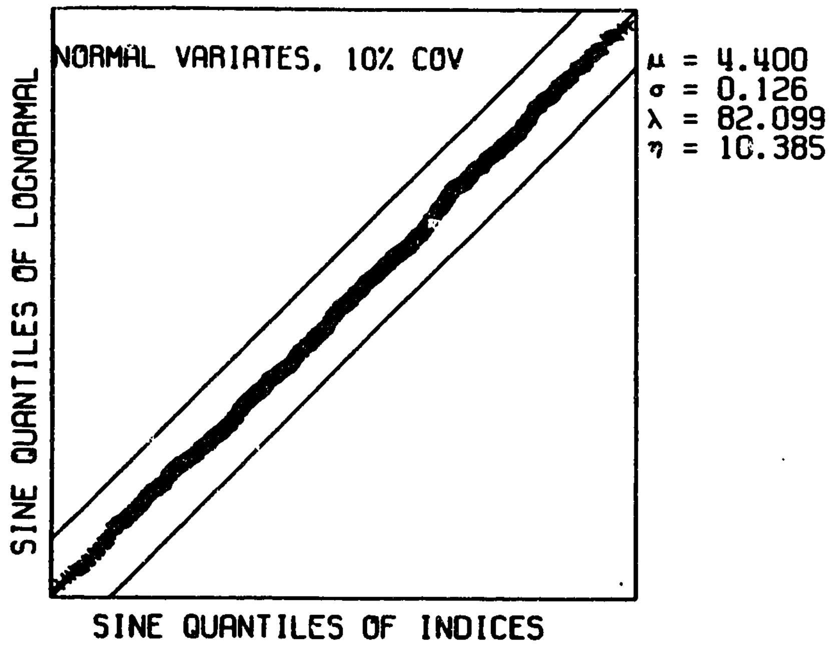


Figure 7.4 Stabilized probability plot using a hypothesized lognormal distribution

Now that the lognormal distribution has been selected as a good fit, the next step is to determine a value of the dragline operating radius which will satisfy the pit requirements 95% of the time. This is a one-tailed test because only the upper tail of the distribution will be considered as a failure 5% of the time. Parameter statistics may be used to estimate the value at which the area under the lognormal probability distribution curve is equal to 0.95. The probability density function of X is defined by Hahn and Shapiro[16] as

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma x}\exp\left[-\frac{1}{2}\left(\frac{\ln x - \mu}{\sigma}\right)^2\right] \qquad x > 0, -\infty<\mu<\infty, \sigma>0 \qquad (7.4)$$
$$= 0 \quad elsewhere$$

The 95% confidence interval is found by applying Normal theory to the logarithms and then transforming. As shown below, a dragline operating radius of 100.27m would be adequate 95% of the time.

$$95\%\ CI = [e^{\mu-1.96\sigma}, e^{\mu+1.96\sigma}] = [63.63m, 104.27m]$$

$$one\ tailed\ upper\ CI = e^{\mu+1.65\sigma} = 100.27m$$

In summary, it is the user's choice as to what information is the most relevant to a particular problem. Some users may simply want to see which variables are contributing the most to a function so that they can direct more attention to further refinement of available data. Others may be concerned only with the final function's distribution. The main strength of VARSIM is that it is a very flexible tool which can be manipulated in whatever form the user wants to analyze a problem.

8. Conclusions

1. The computer programs developed by the research described in this thesis provide Obed Mountain Coal Co. Ltd. with the ability to quickly develop dragline range diagrams, and to obtain output at a desired scale. The program DRAG provides a total of four different dragline pit designs for one- and two-seam mines with horizontal coal seams. The interactive nature of DRAG makes it an effective tool, because users are allowed to control the program flow, and to receive almost immediate feedback when making decisions.

2. The VARSIM library is used to provide numerical variance and bias component analysis of the user-supplied function. By obtaining component contributions to a function's bias and variance, a mining engineer can pinpoint problems requiring component analysis. Users also have the option of simulating the function using normal or lognormal variates. VARSIM is interactive, and, therefore, the user can easily try many different scenarios. Although it is required to formulate the source code for the function, this requirement provides users with the freedom to impose any restrictions that may be necessary at a given minesite.

3. Calculations from the program DRAG were used with the VARSIM library to show how VARSIM could be used to compliment the results of DRAG when an operating decision is to be made. The sensitivity analysis information could have a large impact on the decision if production costs were to be examined.

9. Recommendations for further research

1. In DRAG, three of the models are very general, while the two-seam extended bench model (REH2) is very specific to Obed's operating methods. It is easy to recommend that more general programs be developed for N-seams; however, it is unrealistic to hope to accommodate the many different types of working conditions and equipment .

2. The applicability of expert systems to pit design may deserve more recognition. This would allow the mine engineer to define some of the common working rules, and then examine all of the designs that satisfy the rules. Such a program would require rule changes, as conditions change in the pit.

3. The limitations of a two-dimensional model were noted in chapter 3. A three-dimensional model may help alleviate the problem of determining the effects of haul roads through the spoil on the pit design.

4. A further refinement to the VARSIM library could be to allow it to take into account the correlations between different variables. This would make the program more applicable in some cases, but would not affect its use with others, because this information is not usually available.

5. The VARSIM library could be used to perform sensitivity analysis on many other problems such as safety factors in rock mechanics, cash flows in project feasibility studies, and material balances in mineral processing. VARSIM should not be restricted to use in mining problems since applies to whatever problems can utilize some form of sensitivity analysis.

References

1. Abramowitz, M. and Stegun, I.A., "Handbook of Mathematical Functions", Dover Publications Inc., New York, 1968

2. Ang, A.H. and Tang, W.H., "Probability Concepts in Engineering Planning and Design: Volume 1-Basic Principles", John Wiley & Sons, Inc., 1975

3. Bandopadhyay, S. and Ramani, R.V., "A Computer Simulation Model for Surface Mine Reclamation Planning - Volume II: Dragline Model", Pennsylvania State University, Contract J0295005, April 1985

4. Bandopadhyay, S. and Ramani, R.V., "Digital Simulation of Dragline Deployment Schemes", Proc., 16th International Symposium on the Applications of Computers and Operations Research in the Mineral Industries, O'Neil, T.J., ed., SME/AIME, 1979

5. Bandopadhyay, S. and Ramani, R.V., "Simulation of a Dragline Operation in an Eastern Kentucky Mine", CIM Bulletin, Vol. 78, No. 882, pp. 52-58, Oct. 1985

6. Bandopadhyay, S. and Sundararajan, A., "Computer Simulation of Mining and Reclamation Operations of a Sub-Arctic Coal Mine", Proc., Second Annual Workshop, Generic Mineral Technology Center, Mine Systems Design and Ground Control, Reno, NV, Nov. 12-13, 1984

7. Bandopadhyay, S. and Sundararajan, A., "Simulation of a multi-seam dragline operation in a sub-arctic mine", CIM Bulletin, Vol. 79, No. 893, pp. 47-54, Sept. 1986

8. Bucyrus Erie Co., "Surface Mining Supervisory Training Program", South Milwaukee, Wisconsin, 1977

9. Chatterjee, P.K., Rowlands, D., and Siller, K.C., "Simulation of Dragline Operations", Proc., 14th International Symposium on the Application of Computer Methods in the Mineral Industry, Ramani, R.V., ed., SME/AIME, 1977

10. Chironis, N.P., "Computer Aids Dragline Operator", Coal Age, Vol. 83, No. 3, pp. 58-64, March 1978

11. Chironis, N.P., "Perpendicular stripping pays in hilly Northern Appalachia", Coal Age, Vol. 91, No. 11, pp. 48-50, Nov. 1986

12. Chironis, N.P., "Spoil-side stripping succeeds", Coal Age, Vol. 88, No. 4, pp. 48-53, April 1983

13. Cook, H.C. Jr., "Graphic Pit Design for Two Seam Strip Mining Operations using a Microcomputer", M.Sc. thesis, Colorado School of Mines, Dec. 1986

14. Dagdelen, K., "Dragline Simulation and Selection for Single Flat-Lying Coal Seams", M.Sc. thesis, Colorado School of Mines, June 1979

15. Dunlap, J.W. and Jacobs, H.H., "How operations research solved the dragline problem", Engineering and Mining Journal, Vol. 156, No. 8, pp. 79-83, Aug. 1955

16. Hahn, G.J. and Shapiro, S.S., "Statistical Models in Engineering", John Wiley & Sons, Inc., New York, 1967

17. Harr, M.E., "Reliability-Based Design in Civil Engineering", McGraw-Hill Book Company, New York, 1987

18. Hrebar, M.J., "Application of microcomputers in surface coal mine design", Mining Engineering, Vol. 39, No. 10, pp. 937-941, Oct. 1987

19. Hrebar, M.J. and Cook, H.C., "Estimating dragline productivity using a graphic microcomputer program", International Journal of Surface Mining, Vol. 1, No. 4, pp. 252-255, 1987

20. Hrebar, M.J. and Dagdelen, K., "Equipment selection using simulation of dragline stripping methods", Proc., 16th International Symposium on the Applications of Computers and Operations Research in the Mineral Industries, O'Neil, T.J., ed., SME/AIME, 1979

21. Hrebar, M.J. and Sherer, H.E., "How to evaluate the economics of dragline stripping alternatives", Coal Mining & Processing, Vol. 18, No. 7, pp. 60-67, July 1981

22. Huddart, H. and Runge, I.C., "Pit design optimized for dragline productivity", Institute of Mining and Metallurgy, London. Transactions. Section A. Mining Industry, Vol. 88, pp. A6-A12, 1979

23. IMSL, Inc., "User's Manual: Math/Library, FORTRAN Subroutines for Mathematical Applications", Houston, Texas, 1987

24. Isles, P.T., "Australia's Bowen Basin", Engineering and Mining Journal, Vol. 187, No. 4, pp. 24-30, April 1986

25. Isles, P.T., Hagen, T.N. and Smith, G.H., "Strip Mining", Australasian Coal Mining Practice, Monograph Series No. 12, Martin, C.H., ed., The Australian Institute of Mining and Metallurgy, Victoria, Australia, 1986

26. Learmont, T., "Productivity Improvement in Large Stripping Machines", Transactions, Society of Mining Engineering, AIME, Vol. 258, No. 3, pp. 76-82, 1975

27. Loy, M.D., "Using mine design modules to map your mining approach", Coal Mining & Processing, Vol. 20, No. 7, pp. 46-48, July 1983

28. Martin, J.W., Martin, T.J., Bennett, T.P. and Martin, K.M., "Surface Mining Equipment", First Edition, Martin Consultants, Inc., Colorado, 1982

29. Mathtech Inc., "Evaluation of Current Surface Coal Mining Overburden Techniques and Reclamation Practices", Final Report to U.S. Bureau of Mines on Contract No. S0144081, OFR 28-77, NTIS PB-264 111, Dec. 1976

30. Michael, J.R., "The stabilized probability plot", Biometrika, Vol. 70, No. 1, pp. 11-17, 1983

31. Microsoft Corporation, "Microsoft Fortran Reference, Version 5.0", U.S.A., 1989

32. Mooney, E.L. and Gibson, D.F., "A mathematical programming approach to the selection of stripping technique and dragline size for area surface mines", Proc., 17th International Symposium on the Applications of Computers and Operations Research in the Mineral Industries, Johnson, T.B. and Barnes, R.J., eds., SME/AIME, pp. 500-521, 1982

33. Mooney, E.L. and Gibson, D.F., "Formulation and solution of the dragline and pit width selection problem allowing variable pit widths for varying overburden depth and coal seam thickness", Proc., 17th International Symposium on the Applications of Computers and Operations Research in the Mineral Industries, Johnson, T.B. and Barnes, R.J., eds., SME/AIME, pp. 522-531, 1982

34. Morey, P.G. and Lee, C.D., "Optimization of Dragline Operation", Mining Congress Journal, Vol. 65, No. 6, pp. 27-30, June 1977

35. Nguyen, V.U. and Chowdhury, R.N., "Probabilistic Study of Spoil Pile Stability in Strip Coal Mines - Two Techniques Compared", Int. J. Rock Mech. Min. Sci. & Geomech. Abstr., Vol. 21, No. 6, pp. 303-312, Dec. 1984

36. Plotworks Inc., "PLOT88 Software Library Reference Manual, Second Edition", La Jolla, CA, June 1986

37. Pritsker, A.A.B., "Introduction to Simulation and SLAM II, Third Edition", John Wiley & Sons, Inc., New York, 1986

38. Rodriguez, R., Berlanga, J.M. and Ibarra, M.A., "Mathematical expressions for simulating a dragline mining system", Proc., Mine Planning and Equipment Selection, Singhal, R., ed., Balkema, pp. 409-420, 1988

39. Rosenblueth, E., "Point estimates for probability moments", Proc. Nat. Acad. Sci. USA, Vol. 72, No. 10, pp. 3812-3814, Oct. 1975

40. Rumfelt, H., "Computer method for estimating... Proper machinery mass for stripping overburden", Mining Engineering, Vol. 13, No. 5, pp. 480-487, May 1961

41. Sadri, R.J. and Lee, C.D., "Optimization of single and multiple seam dragline mines through simulation", Proc., 17th International Symposium on the Applications of Computers and Operations Research in the Mineral Industries, Johnson, T.B. and Barnes, R.J., eds., SME/AIME, pp. 642-654, 1982

42. Seymour, C.A., "Dragline Stripping: Extended Bench Method", World Coal, Vol. 5, No. 4, pp. 23-26, April 1979

43. Siddall, J.N., "Probabilistic Engineering Design - Principles and Applications", Marcel Dekker, Inc., New York, 1983

44. Smith, W.P., "PC-FIT : A Statistical Distribution Fitting Package", Statistical Policy Branch, Office of Policy and Evaluation, Environmental Protection Agency, Washington, D.C., April 1985

45. Software Labs, "Fortran Utilities, Version 2.0", 1221 Matisse St., Sunnyvale, CA, Nov. 1983

46. Speake, C.J., Finch, T.E. and Haley, D.R., "Calculating dragline reach requirements for western surface mines", Mining Engineering, Vol. 29, No. 5, pp. 35-37, May 1977

47. Stuart, N.J. and Cobb, Q., "Two approaches to the computerized planning of dragline operations", Proc., Computer Applications in the Mineral Industry, Fytas, Collins & Singhal, eds., Balkema, pp. 23-31, 1988

48. Weimer, W.H., "Production Engineering in Surface Coal Mines", Surface Mining, Pfleider, E.P., ed., AIME, New York, pp. 232-238, 1968

49. White, G.P. and Jones, J.M., "Computer Simulation of Draglines", Mining Engineering, Vol. 36, No. 9, pp. 1349-1352, Sept. 1984

50. Williams, C., Engineering Supervisor, Obed Mountain Coal Co. Ltd., Hinton, Alberta, Personal Communication and Unpublished Data, 1989

Appendix A - Program DRAG

This appendix contains the listing of all the routines used in the program DRAG. These routines can be used to analyze different configurations for single seam and two seam dragline pits. No COMMON blocks are used in these routines so all global variables are passed in the call statements. The purpose of using PARAMETER statements in some of the subroutines is simply to have the convenience of being able to define several variable values at the same time. The values in the PARAMETER statements are not to be modified by the user unless all of the consequences of doing so are investigated thoroughly. Global and local variables are defined within the routines in which they are used.

This appendix is set up in the following manner:

- an index of routines with page numbers
- flow charts of the routines showing how they link together
- listing of all the routines

Index of Routines

SUBROUTINE/FUNCTION Page

Page

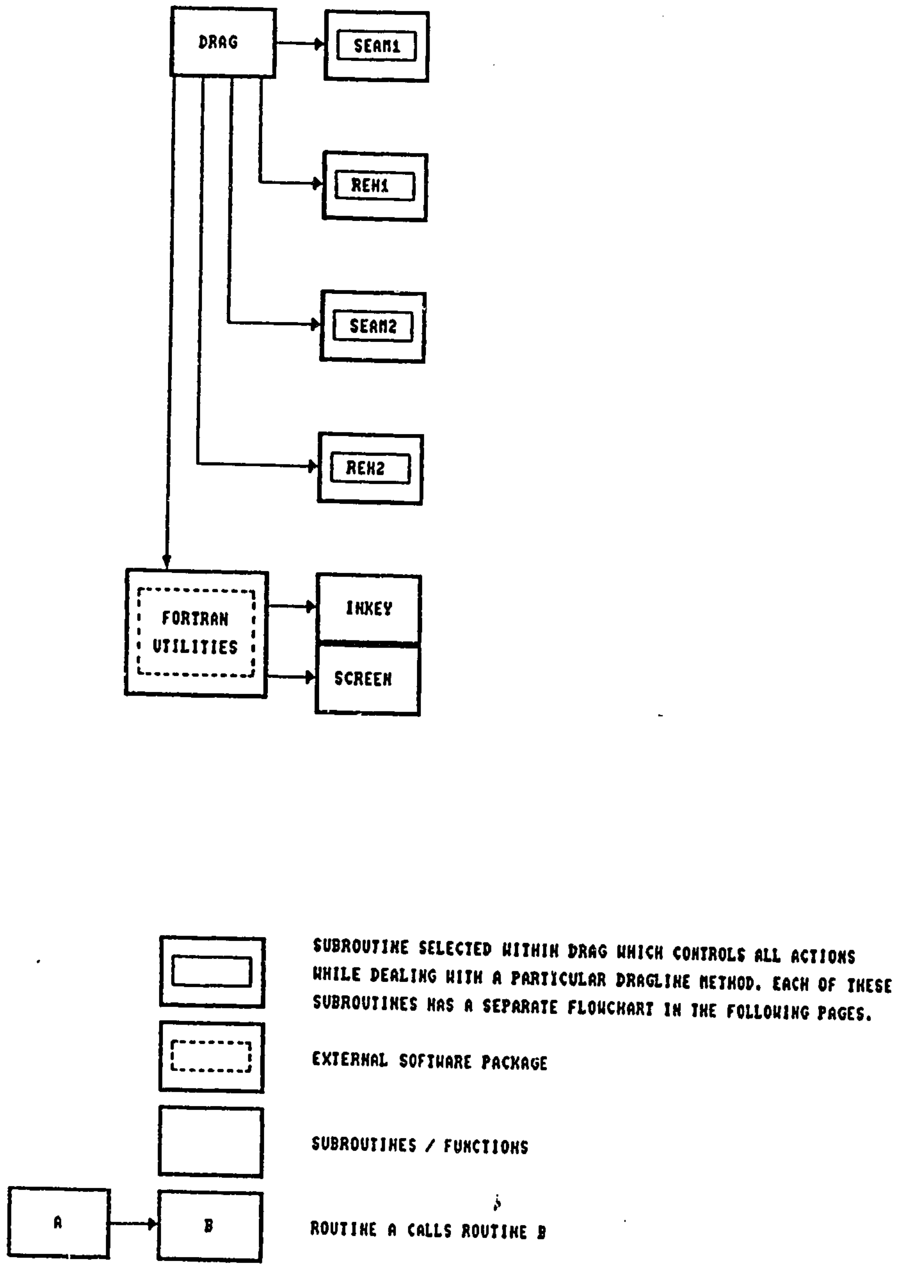

DRAG
SEAM1
REH1
SEAM2
REH2
FORTRAN UTILITIES
INKEY
SCREEN
SUBROUTINE SELECTED WITHIN DRAG WHICH CONTROLS ALL ACTIONS WHILE DEALING WITH A PARTICULAR DRAGLINE METHOD. EACH OF THESE SUBROUTINES HAS A SEPARATE FLOWCHART IN THE FOLLOWING PAGES.
EXTERNAL SOFTWARE PACKAGE
SUBROUTINES / FUNCTIONS
A
B
ROUTINE A CALLS ROUTINE B

Figure A.1 Flow chart showing routines used by DRAG

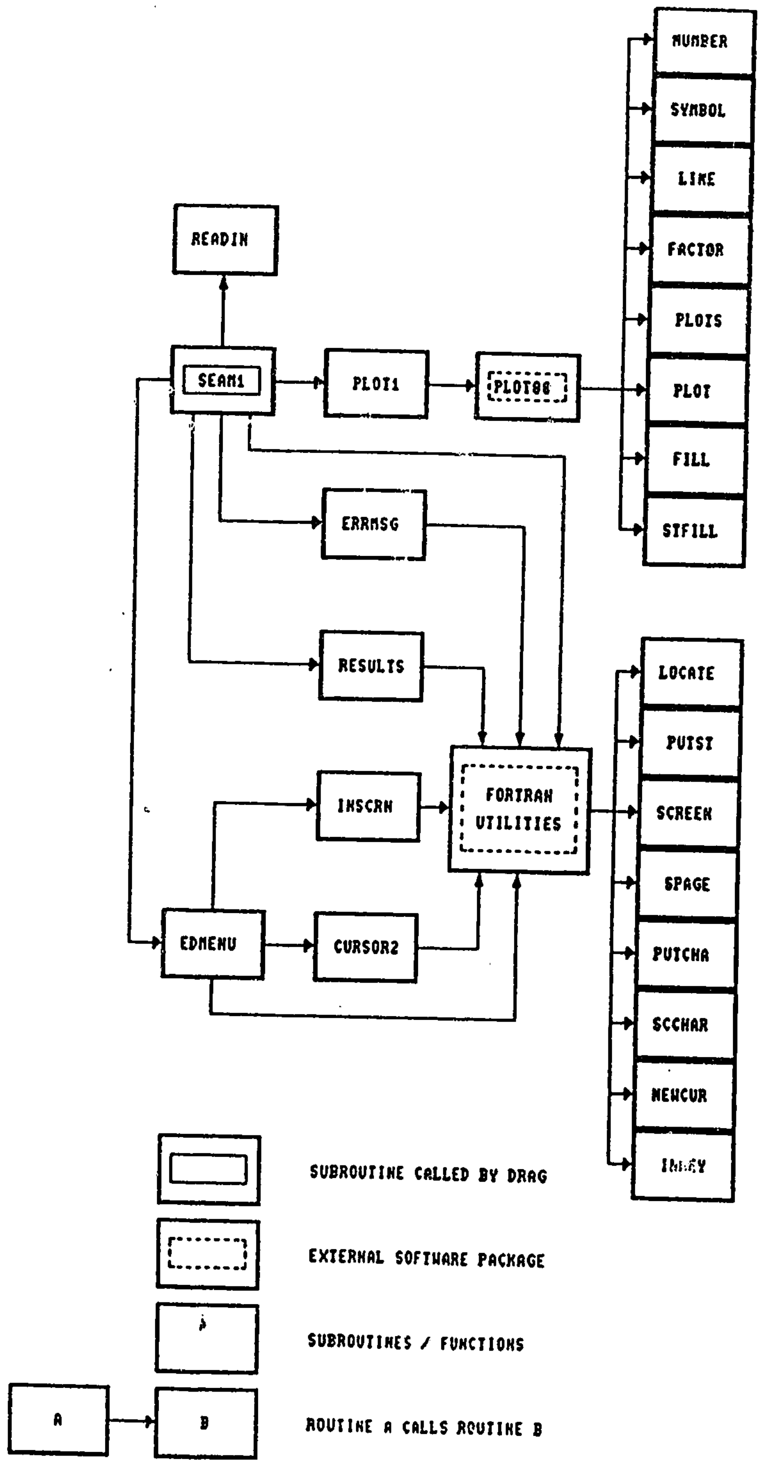


Figure A.2 Flow chart of routines used by SEAM1

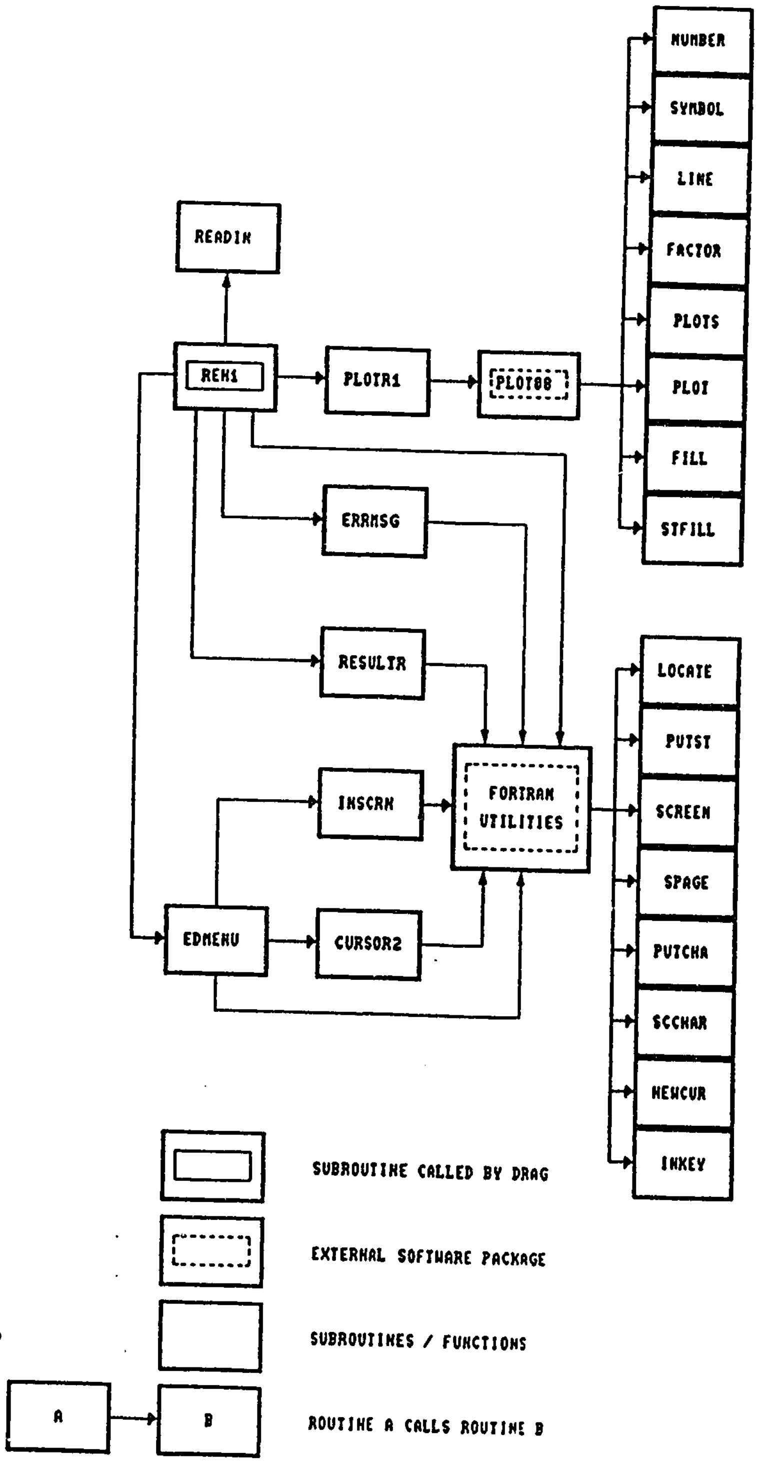


Figure A.3 Flow chart of routines used by REH1

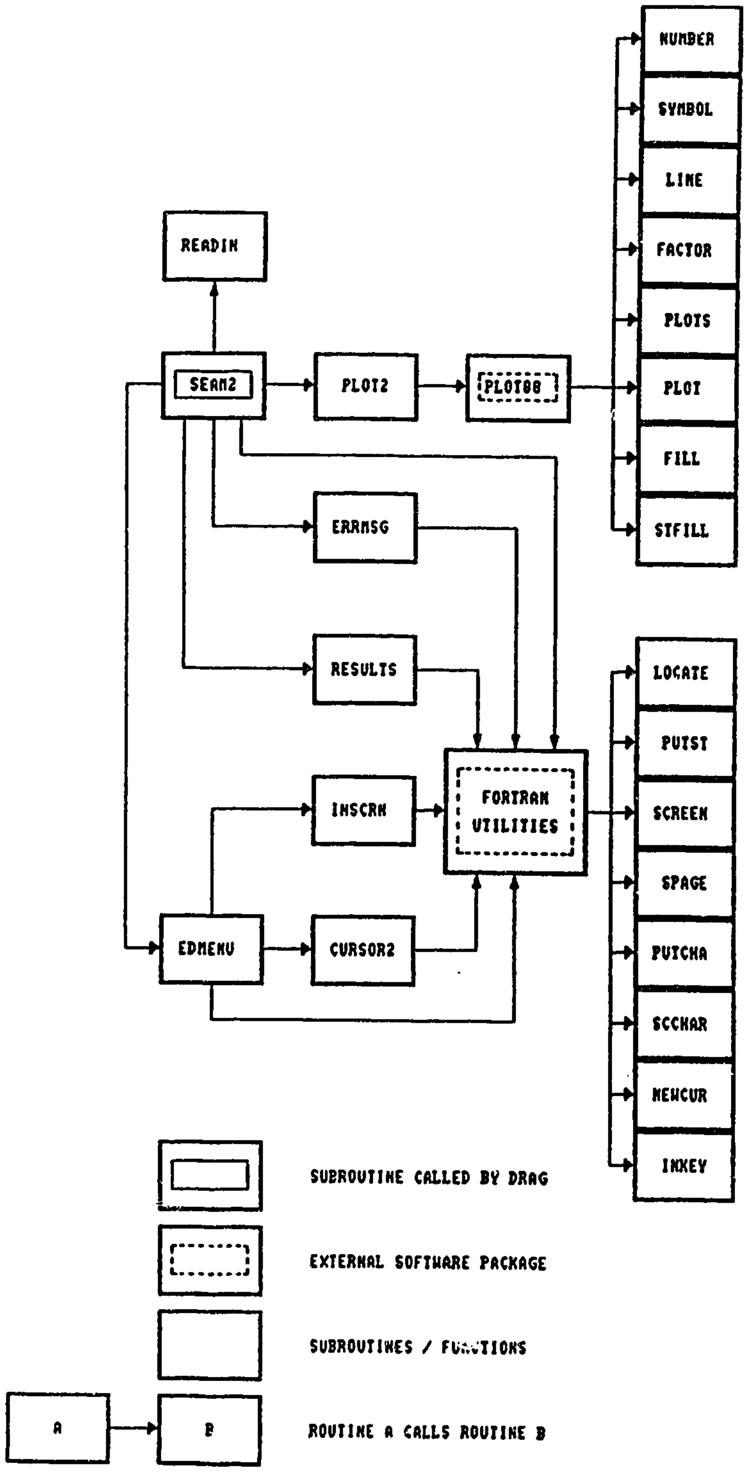


Figure A.4 Flow chart of routines used by SEAM2

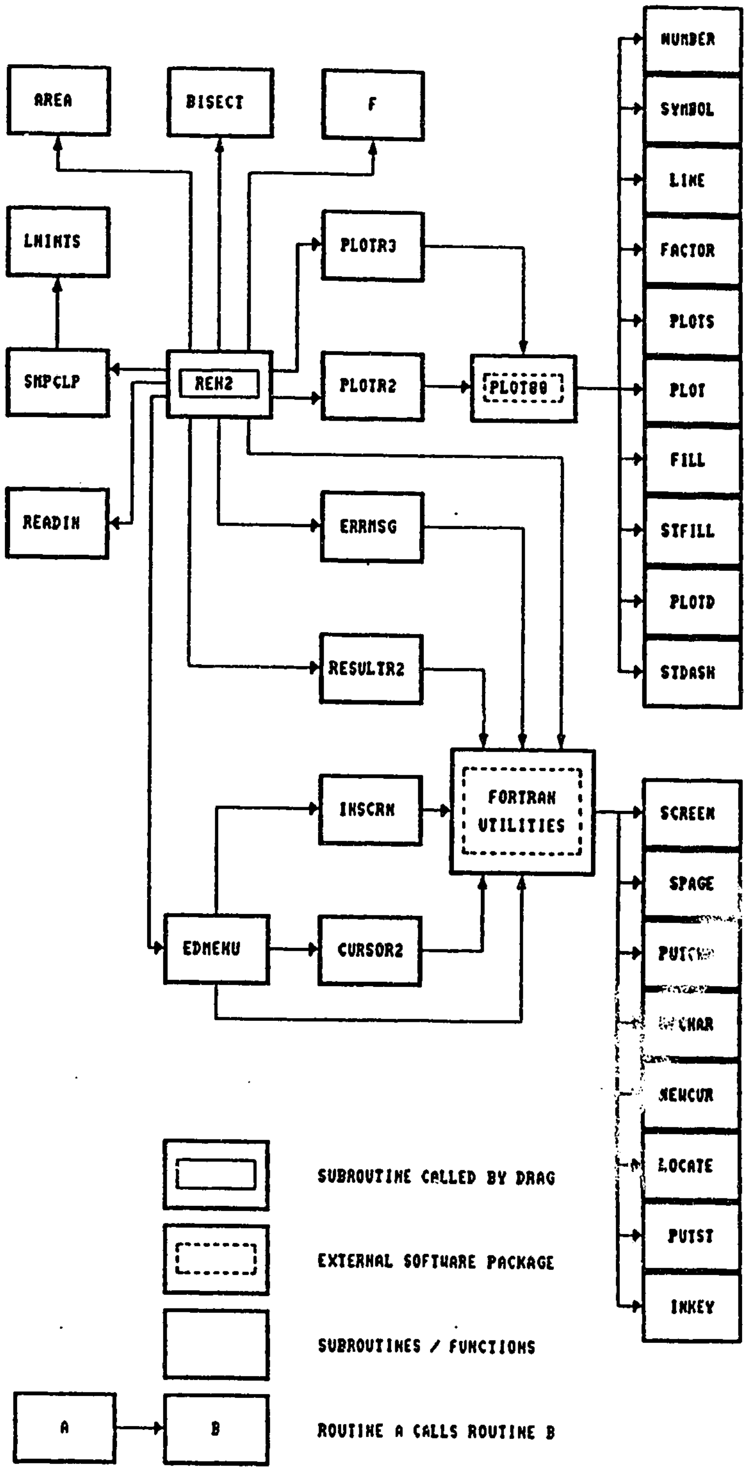


Figure A.5 Flow chart of routines used by REH2

```
      PROGRAM DRAG
C
C  PURPOSE:
C
C  THIS IS THE MAIN PROGRAM.  THE PURPOSE OF THIS PROGRAM
C  IS TO WRITE A DRAGLINE METHOD SELECTION MENU TO THE
C  SCREEN AND THEN ALLOW THE USER TO CHOOSE THE DESIRED
C  METHOD.  ONCE THE USER HAS MADE A SELECTION, THE
C  APPROPRIATE SUBROUTINE IS CALLED.
C
C  INPUT:
C
C  THE ONLY INPUT INTO THIS MAIN PROGRAM OCCURS WHEN A KEY
C  IS PRESSED TO SELECT A METHOD.  THE ONLY FEASIBLE KEYS
C  ARE:
C        1,2,3,4 - TO SELECT A METHOD
C        ESC     - TO QUIT PROGRAM AND RETURN TO DOS
C
C  OTHER KEYS THAT ARE PRESSED WILL BE IGNORED.
C  ALL OF THE REQUIRED INPUT FOR EACH OF THE INDIVIDUAL
C  METHODS WILL BE HANDLED WITHIN THE RESPECTIVE SUBROUTINES
C
C  INTERNAL VARIABLES:
C
C  CH=  CHARACTER*1 VARIABLE CONTAINING THE ASCII CHARACTER
C     READ FROM THE KEYBOARD
C  SCAN=  INTEGER*2 VARIABLE CONTAINING THE EXTENDED CODE
C     VALUE READ FROM THE KEYBOARD
C
C  OUTPUT:
C
C NONE - ALL OUTPUT IS ALSO HANDLED BY THE METHOD SUBROUTINES
C
C  ROUTINES REQUIRED:
C
C    SUBROUTINE SEAM1
C  THIS SUBROUTINE IS USED TO CALCULATE THE REACH AND HEIGHT
C  REQUIREMENTS WHEN MINING A SINGLE SEAM WITH NO REHANDLE
C
C    SUBROUTINE SEAM2
C  THIS SUBROUTINE IS USED TO CALCULATE THE REACH AND HEIGHT
C  REQUIREMENTS WHEN MINING TWO SEAMS WITH NO REHANDLE
C
C    SUBROUTINE REH1
C  THIS SUBROUTINE IS USED TO CALCULATE VOLUMES FOR DIFFERENT
C  SINGLE SEAM PIT DESIGNS AND TO USE AN EXTENDED BENCH IF
C  NECESSARY.
C
C    SUBROUTINE REH2
C  THIS SUBROUTINE IS USED TO CALCULATE VOLUMES FOR DIFFERENT
```

```
C  TWO SEAM PIT DESIGNS AND TO USE AN EXTENDED BENCH ON THE
C  INTERBURDEN IF NECESSARY.
C
C    INTEGER*2 FUNCTION INKEY
C  THIS FUNCTION IS A FORTRAN UTILITIES ROUTINE WHICH IS
C  CALLED TO DETERMINE WHETHER A KEY FROM THE KEYBOARD IS
C  PRESSED AND WHAT IT IS.
C
C
C NOTE:ALL OF THE DRAGLINE METHODS ASSUME THAT THE COAL
C     SEAMS ARE HORIZONTAL AND THAT THE COAL FACE IS VERTICAL
C
C
      INTEGER*2 SCAN,INKEY
      CHARACTER*1 CH
C
C
5     CALL SCREEN(2)
      WRITE(*,*)
     +'                    ** CHOOSE A DRAGLINE METHOD **'
      WRITE(*,*)
      WRITE(*,*)
      WRITE(*,*)
     +'                   1  SIDECASTING     - SINGLE SEAM'
      WRITE(*,*)
     +'                   2  EXTENDED BENCH - SINGLE SEAM'
      WRITE(*,*)
     +'                   3  SIDECASTING     - TWO SEAMS'
      WRITE(*,*)
     +'                   4  EXTENDED BENCH - TWO SEAMS'
      WRITE(*,*)
      WRITE(*,*)
      WRITE(*,*)
     +'                          ESC TO QUIT'
10    IF(INKEY(CH,SCAN).EQ.0) GO TO 10
      SELECT CASE(ICHAR(CH))
      CASE(49)
        CALL SEAM1()
      CASE(50)
        CALL REH1()
      CASE(51)
        CALL SEAM2()
      CASE(52)
        CALL REH2()
      CASE(27)
        CALL SCREEN(2)
        STOP 'NORMAL EXIT BY USER'
      CASE DEFAULT
        GO TO 10
      ENDSELECT
```

```
GO TO 5
STOP
END
```

```
$STORAGE:2
      SUBROUTINE SEAM1()
C
C   PURPOSE:
C
C THIS SUBROUTINE IS USED TO CALCULATE THE REACH AND HEIGHT
C REQUIREMENTS WHEN MINING A SINGLE SEAM WITH NO REHANDLE.
C THE COAL SEAM FACE IS ASSUMED TO BE VERTICAL.  THE FINAL
C SPOIL SLOPE IS ASSUMED TO START AT THE BASE OF THE COAL
C SEAM. IT IS ALSO ASSUMED THAT A HORIZONTAL SEAM IS BEING
C MINED.
C
C   INPUT:
C
C AN INTERACTIVE INPUT SCREEN IS USED SO THAT AFTER THE INITIAL
C VALUES ARE READ IN FROM A PARAMETER FILE, THE VALUES OF THE
C DIFFERENT VARIABLES MAY BE CHANGED TO TRY DIFFERENT
C COMBINATIONS.
C
C INTERNAL VARIABLES:
C
C NAME=  CHARACTER*25 VECTOR CONTAINING THE NAMES OF THE DATA
C     VARIABLES
C VALUE=  REAL*4 VECTOR CONTAINING THE VALUES OF THE DATA
C       VARIABLES
C NDAT=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF DATA
C      VALUES WHICH WILL APPEAR ON THE INPUT SCREEN
C CH=CHARACTER*1 VARIABLE CONTAINING THE ASCII CHARACTER
C    READ FROM THE KEYBOARD
C SCAN=INTEGER*2 VARIABLE CONTAINING THE EXTENDED CODE VALUE
C       READ FROM THE KEYBOARD
C FILE1=  CHARACTER*30 STRING CONTAINING THE NAME OF THE DATA
C       FILE WHICH WILL CONTAIN THE DEFAULT DATA VALUES
C WC=  REAL*4 VARIABLE CONTAINING THE WIDTH OF DRAGLINE CUT (m)
C WP=  REAL*4 VARIABLE CONTAINING THE WIDTH OF SPOIL PIT (m)
C WCH=  REAL*4 VARIABLE CONTAINING THE WIDTH OF CHOPCUT (m)
C TOB=  REAL*4 VARIABLE CONTAINING THE THICKNESS OF THE O/B (m)
C TCH=  REAL*4 VARIABLE CONTAINING THE CHOPCUT THICKNESS (m)
C TTC=  REAL*4 VARIABLE CONTAINING THE COAL SEAM THICKNESS (m)
C THETA=  REAL*4 VARIABLE CONTAINING THE SPOIL ANGLE (DEGREES)
C BETA=REAL*4 VARIABLE CONTAINING THE HIGHWALL ANGLE
C      (DEGREES)
C SWOB=  REAL*4 VARIABLE CONTAINING THE O/B SWELL FACTOR
C PO=  REAL*4 VARIABLE CONTAINING THE POSITIONING FACTOR
C D=  REAL*4 VARIABLE CONTAINING THE DRAGLINE TUB DIAMETER (m)
C X,Y=  REAL*4 VECTORS CONTAINING THE (X,Y) POINT PAIRS OF
C         THE RANGE DIAGRAM
C
C   OUTPUT:
C
```

```
C CALCULATED VALUES ARE WRITTEN TO THE SCREEN AND THEN A
C CHOICE IS GIVEN IN MENU FORMAT AS TO THE TYPE OF OUTPUT
C DESIRED.  THE PROGRAM IS SET UP TO SUPPORT BOTH EGA AND CGA
C GRAPHICS.  THE RANGE DIAGRAM CAN ALSO BE SENT TO A PRINTER OR
C PLOTTER FOR A HARD COPY.  THE DEFAULT PRINTER IS THE EPSON
C LQ500 WHILE THE DEFAULT PLOTTER IS THE HEWLETT PACKARD
C HP7475A. IF ANOTHER PRINTER OR PLOTTER IS DESIRED, A FILE
C CALLED PRINT.DAT CAN BE USED TO OVER-RIDE THE DEFAULT
C SPECIFICATIONS.  PRINT.DAT SHOULD CONTAIN 3 LINES:
C
C        LINE 1 - IOPORT VALUE (USUALLY SET = 0)
C        LINE 2 - MODEL VALUE (LQ500 = 42)
C        LINE 3 - WIDTH OF OUTPUT CONTROL
C                     (0 = NARROW CARRIAGE, 1 = WIDE CARRIAGE)
C
C  ALL OF THE VALUES ARE READ FROM PRINT.DAT IN FREE FORMAT.
C
C  ERRORS:
C
C  PROGRAM DISPLAYS THE FOLLOWING ERROR MESSAGES:
C
C       1. THE DATA FILE DOES NOT CONTAIN ## VARIABLES
C
C       2. DRAGLINE POSITIONING FACTOR MUST BE AT LEAST 0.5
C
C       3. DRAGLINE MUST BE POSITIONED ON CHOP OR MAIN CUT
C
C       4. VALID ANGLE RANGE (DEGREES):  0 < ANGLE < 90
C
C  SUBROUTINES REQUIRED:
C
C  READIN= THIS SUBROUTINE IS USED TO READ VARIABLE NAMES AND
C        VALUES FROM FILE1
C  EDMENU= THIS SUBROUTINE IS USED TO CONTROL THE EDITING OF
C     VALUES ON THE INPUT SCREEN.  THE ENTRY CALLED ED IS USED
C       TO INITIALIZE THE LOCATIONS OF THE DATA ON THE SCREEN.
C  CURSOR2= THIS SUBROUTINE IS REQUIRED BY THE SUBROUTINE EDIT
C        TO CONTROL CURSOR MOVEMENT AND DATA ENTRY ON THE
C        INPUT SCREEN
C  INSCRN= THIS SUBROUTINE IS REQUIRED BY THE SUBROUTINE
C      EDMENU TO REWRITE TEXT AND VALUES ON THE SCREEN IF
C      NECESSARY
C  ERRMSG= THIS SUBROUTINE IS USED TO DISPLAY MESSAGES ON THE
C        SCREEN
C  RESULTS= THIS SUBROUTINE IS USED TO WRITE THE CALCULATED
C        RESULTS TO THE SCREEN
C  PLOT1= THIS SUBROUTINE IS USED TO PLOT THE RANGE DIAGRAM
C        USING THE PLOT88 SOFTWARE GRAPHICS PACKAGE.  THE ENTRY
C        INITP1 IS USED TO INITIALIZE THE PRINTER SPECIFICATIONS.
C  SPAGE= THIS FORTRAN UTILITIES ROUTINE IS CALLED TO SELECT
```

```
C       ONE SCREEN PAGE FOR DISPLAYING
C   SCREEN= THIS FORTRAN UTILITIES ROUTINE IS CALLED TO SET THE
C        SCREEN MODE AND CLEAR THE SCREEN
C
C   FUNCTIONS REQUIRED:
C
C INKEY=AN INTEGER*2 FORTRAN UTILITIES ROUTINE CALLED TO
C      DETERMINE WHETHER A KEY FROM THE KEYBOARD IS PRESSED
C      AND WHAT IT IS
C
C
     INTEGER*2 N,NUM,NDAT,SCAN,INKEY,ISEAM,NN,IMENU
     PARAMETER(N=20,NN=11)
     REAL*4 WCH,WC,WP
     REAL*4 PO,D
     REAL*4 TCH,TOB,TTC
     REAL*4 BETA,TANB,THETA,TANT,RAD
     REAL*4 SWOB
     REAL*4 H,DH,OROB,ORIB,VOB,VIB,VCHOP,VSPOIL,VTEST
     REAL*4 XDIST,YDIST,X(N+2),Y(N+2),VALUE(NN)
     CHARACTER NAME(NN)*25, FILE1*30, CH*1
     LOGICAL*2 ISW,ISW2
C
     FILE1='SEAM1.DAT'
     ISW=.TRUE.
     ISW2=.TRUE.
     ISEAM=1
C
C--READ DATA VALUES FROM A DATA FILE
C
     CALL READIN(NAME,VALUE,NDAT,FILE1)
C
C--CHECK IF DATA FILE CONTAINS THE CORRECT NUMBER OF DATA
C  VALUES
C
     IF(NDAT.NE.NN)THEN
        WRITE(*,*)'** ERROR **'
        WRITE(*,5)NN
   5 FORMAT(' THE DATA FILE DOES NOT CONTAIN ',I2,' VARIABLES')
        STOP
     ENDIF
C
C--INITIALIZE PLOT1 AND EDMENU SUBROUTINES
C
     CALL INITP1()
     CALL ED(NDAT,1)
C
C--SET THE SCREEN MODE:  2 SELECTS AN 80x25 BLACK AND WHITE
C  DISPLAY WITH 4 TEXT PAGES
C
```

```
      CALL SCREEN(2)
C
   10 CALL EDMENU(NAME,VALUE,ISW,IMENU)
      IF(IMENU.EQ.0)RETURN
      CALL SPAGE(1)
C
      WC   = VALUE(1)
      WP   = VALUE(2)
      WCH  = VALUE(3)
      TOB  = VALUE(4)
      TCH  = VALUE(5)
      TTC  = VALUE(6)
      THETA= VALUE(7)
      BETA = VALUE(8)
      SWOB = VALUE(9)
      PO   = VALUE(10)
      D    = VALUE(11)
C
C--ANGLE AND DRAGLINE POSITIONING CHECK
C
      IF(PO.LT.0.5)THEN
        CALL ERRMSG(
     +     'DRAGLINE POSITIONING FACTOR MUST BE AT LEAST 0.5',48)
      ELSEIF((PO+.5)*D.GT.WCH+WC)THEN
        CALL ERRMSG(
     +     'DRAGLINE MUST BE POSITIONED ON CHOP OR MAIN CUT ',48)
      ELSEIF(THETA.GE.90..OR.BETA.GE.90..OR.THETA.LE.0..OR.BETA.LE.0.)
     +              THEN
        CALL ERRMSG(
     +             'VALID ANGLE RANGE (DEGREES):   0 < ANGLE < 90   ',48)
      ELSE
        GO TO 20
      ENDIF
      GO TO 10
20    RAD=.0174532925
      TANT=TAN(THETA*RAD)
      TANB=TAN(BETA*RAD)
C
C--DETERMINE CUT AND SPOIL VOLUMES
C
      VCHOP=WCH*TCH
      VOB=TOB*WC
      VSPOIL=(VCHOP+VOB)*SWOB
      VIB=0.
C
C--FIRST TEST IF THE SPOIL COVERS THE SPOIL PIT, THEN
C  DETERMINE THE OPERATING RADIUS AND HEIGHT REQUIREMENTS OF
C  THE DRAGLINE WHEN IT IS SPOILING FROM THE O/B BENCH
C
      VTEST=WP*WP*0.25*TANT
      IF(VSPOIL.GT.VTEST)THEN
        DH=(VSPOIL+VTEST)/WP
      ELSE
        DH=SQRT(VSPOIL*TANT)
      ENDIF
      OROB=PO*D + TOB/TANB + DH/TANT
      ORIB=0.
      H=DH - TOB - TTC
C
C--WRITE OUTPUT VALUES TO SCREEN
C
      CALL RESULTS(VOB,VIB,VCHOP,VSPOIL,OROB,ORIB,H,ISW2,ISEAM)
C
C--SELECT MODE OF OUTPUT
C
   80 IF(INKEY(CH,SCAN).EQ.0) GO TO 80
      SELECT CASE(ICHAR(CH))
         CASE(48)
           GO TO 10
         CASE(49)
           NUM=1
         CASE(50)
           NUM=2
         CASE(51)
           NUM=3
         CASE(52)
           NUM=4
         CASE(53)
           NUM=5
         CASE DEFAULT
           GO TO 80
      END SELECT
C
C--DETERMINE X AND Y COORDINATES OF RANGE DIAGRAM
C
      XDIST=5.0
      YDIST=15.0
      X(1)=XDIST+WCH+(TOB+TCH)/TANB
      Y(1)=YDIST
      X(2)=0.0
      Y(2)=YDIST
      X(3)=0.0
      Y(3)=TTC+YDIST
      X(4)=X(1)
      Y(4)=TTC+YDIST
      X(5)=XDIST+WCH
      Y(5)=TTC+TOB+TCH+YDIST
      X(6)=0.0
      Y(6)=Y(5)
      X(7)=XDIST
      Y(7)=Y(5)
      X(8)=XDIST+TCH/TANB
      Y(8)=TTC+TOB+YDIST
      X(9)=0.0
      Y(9)=Y(8)
      X(10)=X(8)+WCH+WC
      Y(10)=Y(8)
      X(11)=X(1)+WC
      Y(11)=TTC+YDIST
      X(12)=X(1)
      Y(12)=TTC+YDIST
      X(13)=X(1)
      Y(13)=YDIST
      X(14)=X(11)
      Y(14)=YDIST
      X(15)=X(11)
      Y(15)=Y(11)
      X(16)=X(14)
      Y(16)=YDIST
      X(17)=X(16)+DH/TANT
      Y(17)=DH+YDIST
C
C--TEST IF SPOIL COVERS SPOIL PIT WIDTH
C
      IF(VSPOIL.GT.VTEST)THEN
        X(18)=X(17)+0.5*WP
        Y(18)=Y(17)-0.5*WP*TANT
      ELSE
        X(18)=X(16)+DH*2./TANT
        Y(18)=YDIST
      ENDIF
      X(19)=X(16)+WP
      Y(19)=YDIST
      X(20)=X(16)
      Y(20)=YDIST
      X(21)=0.0
      Y(21)=0.0
C
      CALL SPAGE(3)
      CALL PLOT1(X,Y,N,NUM,VALUE,NAME,H,OROB)
C
      IF(NUM.EQ.1.OR.NUM.EQ.2)THEN
        ISW=.TRUE.
        ISW2=.TRUE.
      ENDIF
      GO TO 10
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE SEAM2()
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO CALCULATE THE REACH AND HEIGHT
C  REQUIREMENTS WHEN MINING TWO SEAMS WITH NO REHANDLE.  THIS
C  PROGRAM ASSUMES THAT THE COAL FACE IS VERTICAL AND THE
C  ANGLE OF REPOSE IS THE SAME FOR BOTH I/B AND O/B.  THE FINAL
C  SPOIL SLOPE IS ASSUMED TO START AT THE BASE OF THE COAL
C  SEAM. IT IS ALSO ASSUMED THAT HORIZONTAL SEAMS ARE BEING
C  MINED.
C
C  INPUT:
C
C AN INTERACTIVE INPUT SCREEN IS USED SO THAT AFTER THE INITIAL
C  VALUES ARE READ IN FROM A PARAMETER FILE, THE VALUES OF THE
C  DIFFERENT VARIABLES MAY BE CHANGED TO TRY DIFFERENT
C  COMBINATIONS.
C
C  INTERNAL VARIABLES:
C
C  NAME=CHARACTER*25 VECTOR CONTAINING THE NAMES OF THE
C      DATA VARIABLES
C  VALUE= REAL*4 VECTOR CONTAINING THE VALUES OF THE DATA
C      VARIABLES
C  NDAT= INTEGER*2 VARIABLE CONTAINING THE NUMBER OF DATA
C    VALUES WHICH WILL APPEAR ON THE INPUT SCREEN
C  CH=CHARACTER*1 VARIABLE CONTAINING THE ASCII CHARACTER
C    READ FROM THE KEYBOARD
C  SCAN=INTEGER*2 VARIABLE CONTAINING THE EXTENDED CODE
C      VALUE READ FROM THE KEYBOARD
C  FILE1= CHARACTER*30 STRING CONTAINING THE NAME OF THE DATA
C      FILE WHICH WILL CONTAIN THE DEFAULT DATA VALUES
C  WCO= REAL*4 VARIABLE CONTAINING THE O/B CUT WIDTH (m)
C  WC= REAL*4 VARIABLE CONTAINING THE I/B CUT WIDTH (m)
C  WP= REAL*4 VARIABLE CONTAINING THE WIDTH OF SPOIL PIT (m)
C  WCH= REAL*4 VARIABLE CONTAINING THE WIDTH OF CHOPCUT (m)
C  WSB= REAL*4 VARIABLE CONTAINING THE WIDTH OF THE SAFETY
C      BENCH (m)
C  TOB= REAL*4 VARIABLE CONTAINING THE THICKNESS OF THE O/B (m)
C  TIB= REAL*4 VARIABLE CONTAINING THE THICKNESS OF THE I/B (m)
C  TCH= REAL*4 VARIABLE CONTAINING THE CHOPCUT THICKNESS (m)
C  TTC= REAL*4 VARIABLE CONTAINING THE THICKNESS OF THE UPPER
C      COAL SEAM (m)
C  TBC= REAL*4 VARIABLE CONTAINING THE THICKNESS OF THE LOWER
C      COAL SEAM (m)
C  THETA= REAL*4 VARIABLE CONTAINING THE SPOIL ANGLE (DEGREES)
C  BETA= REAL*4 VARIABLE CONTAINING THE O/B HIGHWALL
C      ANGLE (DEGREES)
```

```
C  ALPHA= REAL*4 VARIABLE CONTAINING THE I/B HIGHWALL
C       ANGLE (DEGREES)
C  SWOB= REAL*4 VARIABLE CONTAINING THE O/B SWELL FACTOR
C  SWIB= REAL*4 VARIABLE CONTAINING THE I/B SWELL FACTOR
C  PO= REAL*4 VARIABLE CONTAINING THE O/B POSITIONING FACTOR
C  PI= REAL*4 VARIABLE CONTAINING THE I/B POSITIONING FACTOR
C  D= REAL*4 VARIABLE CONTAINING THE DRAGLINE TUB DIAMETER (m)
C  X,Y= REAL*4 VECTORS CONTAINING THE (X,Y) POINT PAIRS OF
C      THE RANGE DIAGRAM
C
C  OUTPUT:
C
C  CALCULATED VALUES ARE WRITTEN TO THE SCREEN AND THEN A
C  CHOICE IS GIVEN IN MENU FORMAT AS TO THE TYPE OF OUTPUT
C  DESIRED.  THE PROGRAM IS SET UP TO SUPPORT BOTH EGA AND CGA
C  GRAPHICS.THE RANGE DIAGRAM CAN ALSO BE SENT TO A PRINTER OR
C  PLOTTER FOR A HARD COPY.  THE DEFAULT PRINTER IS THE EPSON
C  LQ500 WHILE THE DEFAULT PLOTTER IS THE HEWLETT PACKARD
C  HP7475A. IF ANOTHER PRINTER OR PLOTTER IS DESIRED, A FILE
C  CALLED PRINT.DAT CAN BE USED TO OVER-RIDE THE DEFAULT
C  SPECIFICATIONS.
C  PRINT.DAT SHOULD CONTAIN 3 LINES:
C
C       LINE 1 - IOPORT VALUE (USUALLY SET = 0)
C       LINE 2 - MODEL VALUE (LQ500 = 42)
C       LINE 3 - WIDTH OF OUTPUT CONTROL
C                    (0 = NARROW CARRIAGE, 1 = WIDE CARRIAGE)
C
C  ALL OF THE VALUES ARE READ FROM PRINT.DAT IN FREE FORMAT.
C
C  ERRORS:
C
C  PROGRAM DISPLAYS THE FOLLOWING ERROR MESSAGES:
C
C      1. THE DATA FILE DOES NOT CONTAIN ## VARIABLES
C
C      2. DRAGLINE POSITIONING FACTOR MUST BE AT LEAST 0.5
C
C      3. DRAGLINE MUST BE POSITIONED WITHIN CHOPCUT ON O/B
C
C      4. DRAGLINE MUST BE POSITIONED WITHIN CUT ON I/B
C
C      5. I/B WIDTH MUST BE <= WIDTH OF O/B + SAFETY BENCH
C
C      6. VALID ANGLE RANGE (DEGREES):  0 < ANGLE < 90
C
C  SUBROUTINES REQUIRED:
C
C  READIN= THIS SUBROUTINE IS USED TO READ VARIABLE NAMES AND
C        VALUES FROM FILE1
```

```
C  EDMENU= THIS SUBROUTINE IS USED TO CONTROL THE EDITING OF
C        VALUES ON THE INPUT SCREEN. THE ENTRY CALLED ED IS USED
C        TO INITIALIZE THE LOCATIONS OF THE DATA ON THE SCREEN.
C  CURSOR2= THIS SUBROUTINE IS REQUIRED BY THE SUBROUTINE
C        EDMENU TO CONTROL CURSOR MOVEMENT AND DATA ENTRY ON
C        THE INPUT SCREEN
C INSCRN=THIS SUBROUTINE IS REQUIRED BY THE SUBROUTINE EDMEMU
C        TO REWRITE TEXT AND VALUES ON THE SCREEN IF NECESSARY
C  ERRMSG= THIS SUBROUTINE IS USED TO DISPLAY MESSAGES ON THE
C        SCREEN
C  RESULTS= THIS SUBROUTINE IS USED TO WRITE THE CALCULATED
C        RESULTS TO THE SCREEN
C  PLOT2= THIS SUBROUTINE IS USED TO PLOT THE RANGE DIAGRAM
C        USING THE PLOT88 SOFTWARE GRAPHICS PACKAGE.  THE ENTRY
C        INITP2 IS USED TO INITIALIZE THE PRINTER SPECIFICATIONS.
C  SPAGE= A FORTRAN UTILITIES ROUTINE CALLED TO SELECT ONE
C       SCREEN PAGE FOR DISPLAYING
C  SCREEN=A FORTRAN UTILITIES ROUTINE CALLED TO SET THE SCREEN
C       MODE AND CLEAR THE SCREEN
C
C  FUNCTIONS REQUIRED:
C
C  INKEY= AN INTEGER*2 FORTRAN UTILITIES ROUTINE CALLED TO
C       DETERMINE WHETHER A KEY FROM THE KEYBOARD IS PRESSED
C       AND WHAT IT IS
C
C
   INTEGER*2 N,NUM,NDAT,SCAN,INKEY,ISEAM,NN,IMENU
   PARAMETER(N=34,NN=18)
   REAL*4 WCH,WC,WCO,WP,WSB
   REAL*4 PO,D,PI
   REAL*4 TCH,TOB,TTC,TIB,TBC
   REAL*4 BETA,THETA,ALPHA
   REAL*4 SWOB,SWIB
   REAL*4 RAD,TANB,TANA,TANT,VTEST
   REAL*4 H,OROB,ORIB,VOB,VIB,VCHOP,VSPOIL,DH1,DH2,VOBSP,VIBSP
   REAL*4 XDIST,YDIST,X(N+2),Y(N+2),VALUE(NN)
   CHARACTER NAME(NN)*25, FILE1*30, CH*1
   LOGICAL*2 ISW,ISW2
C
   FILE1='SEAM2.DAT'
   ISW=.TRUE.
   ISW2=.TRUE.
   ISEAM=2
C
C--READ DATA VALUES FROM A DATA FILE
C
   CALL READIN(NAME,VALUE,NDAT,FILE1)
C
C--CHECK IF DATA FILE CONTAINS THE CORRECT NUMBER OF DATA
```

```
C  VALUES
C
     IF(NDAT.NE.NN)THEN
        WRITE(*,*)'** ERROR **'
        WRITE(*,5)NN
   5 FORMAT(' THE DATA FILE DOES NOT CONTAIN ',I2,' VARIABLES')
        STOP
     ENDIF
C
C--INITIALIZE PLOT2 AND EDMENU SUBROUTINES
C
     CALL INITP2()
     CALL ED(NDAT,1)
C
C--SET THE SCREEN MODE:  2 SELECTS AN 80x25 BLACK AND WHITE
C  DISPLAY WITH 4 TEXT PAGES
C
     CALL SCREEN(2)
C
  10 CALL EDMENU(NAME,VALUE,ISW,IMENU)
     IF(IMENU.EQ.0)RETURN
     CALL SPAGE(1)
C
     WCO  = VALUE(1)
     WC   = VALUE(2)
     WP   = VALUE(3)
     WCH  = VALUE(4)
     WSB  = VALUE(5)
     TOB  = VALUE(6)
     TIB  = VALUE(7)
     TCH  = VALUE(8)
     TTC  = VALUE(9)
     TBC  = VALUE(10)
     THETA= VALUE(11)
     BETA = VALUE(12)
     ALPHA= VALUE(13)
     SWOB = VALUE(14)
     SWIB = VALUE(15)
     PO   = VALUE(16)
     PI   = VALUE(17)
     D    = VALUE(18)
C
C--ANGLE AND DRAGLINE POSITIONING CHECK
C
     IF(PO.LT.0.5.OR.PI.LT.0.5)THEN
        CALL ERRMSG(
    +   'DRAGLINE POSITIONING FACTOR MUST BE AT LEAST 0.5',48)
     ELSEIF((PO+.5)*D.GT.WCH+WCO)THEN
        CALL ERRMSG(
    +  'DRAGLINE MUST BE POSITIONED WITHIN CHOPCUT ON O/B',49)
```

```
      ELSEIF((PI+.5)*D.GT.WC)THEN
         CALL ERRMSG(
     +     'DRAGLINE MUST BE POSITIONED WITHIN CUT ON I/B',45)
      ELSEIF(WC.GT.WCO+WSB)THEN
         CALL ERRMSG(
     +    'I/B WIDTH MUST BE <= WIDTH OF O/B + SAFETY BENCH',48)
      ELSEIF(THETA.GE.90..OR.BETA.GE.90..OR.THETA.LE.0..OR.BETA.LE.0.
     +.OR.ALPHA.GE.90..OR.ALPHA.LE.0.)THEN
         CALL ERRMSG(
     +     'VALID ANGLE RANGE (DEGREES):  0 < ANGLE < 90',45)
      ELSE
        GO TO 20
      ENDIF
      GO TO 10
C
20     RAD=.0174532925
      TANB=TAN(BETA*RAD)
      TANA=TAN(ALPHA*RAD)
      TANT=TAN(THETA*RAD)
C
C--DETERMINE CUT AND SPOIL VOLUMES
C
      VCHOP=WCH*TCH
      VOB=TOB*WCO
      VIB=TIB*WC
      VOBSP=(VCHOP+VOB)*SWOB
      VIBSP=VIB*SWIB
      VSPOIL=VIBSP+VOBSP
C
C--THERE ARE THREE DIFFERENT CASES THAT MUST BE CONSIDERED
C  WHEN DEALING WITH THE DRAGLINE SPOIL
C
C    CASE 1: ALL OF THE SPOIL (O/B + I/B) FALLS WITHIN THE WIDTH
C            OF THE SPOIL PIT
C    CASE 2: ALL OF THE O/B SPOIL FALLS WITHIN THE WIDTH OF THE
C            SPOIL, HOWEVER THE I/B SPOIL RIDES UP THE PREVIOUS
C            SPOIL SLOPE
C    CASE 3: THE O/B SPOIL COVERS THE ENTIRE SPOIL PIT WIDTH AND
C            RIDES UP THE SPOIL SLOPE
C
C  THREE DIFFERENT VALUES WILL NOW BE CALCULATED ACCORDING TO
C  THE SPOIL CONFIGURATION
C
C  THE OPERATING RADIUS REQUIREMENTS WILL BE DETERMINED FOR
C  THE DRAGLINE WHEN IT IS SPOILING FROM THE O/B AND I/B
C
C  THE HEIGHT REQUIREMENTS WILL ALSO BE DETERMINED FOR WHEN
C  THE DRAGLINE IS OPERATING ON THE I/B BENCH
C
      VTEST=WP*WP*0.25*TANT
      IF(VSPOIL.LE.VTEST)THEN
         DH1=SQRT(VOBSP*TANT)
         DH2=SQRT(VSPOIL*TANT)
      ELSEIF(VSPOIL.GT.VTEST.AND.VOBSP.LE.VTEST)THEN
         DH1=SQRT(VOBSP*TANT)
         DH2=(VSPOIL+VTEST)/WP
      ELSE
         DH1=(VOBSP+VTEST)/WP
         DH2=(VSPOIL+VTEST)/WP
      ENDIF
      OROB=PO*D+TOB/TANB+WSB+TIB/TANA+DH1/TANT
      ORIB=PI*D+TIB/TANA+DH2/TANT
      H=DH2-TIB-TBC
C
C--WRITE OUTPUT VALUES TO SCREEN
C
      CALL RESULTS(VOB,VIB,VCHOP,VSPOIL,OROB,ORIB,H,ISW2,ISEAM)
C
C--SELECT MODE OF OUTPUT
C
   80 IF(INKEY(CH,SCAN).EQ.0) GO TO 80
      SELECT CASE(ICHAR(CH))
         CASE(48)
           GO TO 10
         CASE(49)
           NUM=1
         CASE(50)
           NUM=2
         CASE(51)
           NUM=3
         CASE(52)
           NUM=4
         CASE(53)
           NUM=5
         CASE DEFAULT
           GO TO 80
      END SELECT
C
C--DETERMINE X AND Y COORDINATES OF RANGE DIAGRAM
C
      XDIST=5.0
      YDIST=15.0
      X(1)=XDIST+WCH+(TOB+TCH)/TANB+(WSB+WCO-WC)+TIB/TANA
      Y(1)=YDIST
      X(2)=0.0
      Y(2)=YDIST
      X(3)=0.0
      Y(3)=TBC+YDIST
      X(4)=X(1)
      Y(4)=TBC+YDIST
```

```
X(5)=X(4)-TIB/TANA
Y(5)=Y(4)+TIB
X(6)=0.0
Y(6)=Y(5)
X(7)=0.0
Y(7)=Y(6)+TTC
X(8)=X(5)-(WSB+WCO-WC)
Y(8)=Y(7)
X(9)=XDIST+WCH
Y(9)=Y(8)+TCH+TOB
X(10)=0.0
Y(10)=Y(9)
X(11)=XDIST
Y(11)=Y(9)
X(12)=X(11)+TCH/TANB
Y(12)=Y(9)-TCH
X(13)=0.0
Y(13)=Y(12)
X(14)=X(12)+WCH+WCO
Y(14)=Y(12)
X(15)=X(8)+WCO
Y(15)=Y(8)
X(16)=X(8)
Y(16)=Y(8)
X(17)=X(8)
Y(17)=Y(5)
X(18)=X(17)+WCO
Y(18)=Y(5)
X(19)=X(15)
Y(19)=Y(15)
X(20)=X(18)
Y(20)=Y(18)
X(21)=X(20)+WSB
Y(21)=Y(20)
X(22)=X(4)+WC
Y(22)=Y(4)
X(23)=X(4)
Y(23)=Y(4)
X(24)=X(1)
Y(24)=YDIST
X(25)=X(1)+WC
Y(25)=YDIST
X(26)=X(22)
Y(26)=Y(22)
X(27)=X(25)
Y(27)=YDIST
X(28)=X(27)+DH1/TANT
Y(28)=DH1+YDIST
X(30)=X(28)
Y(30)=Y(28)
```

```
      X(31)=X(27)+DH2/TANT
      Y(31)=DH2+YDIST
      IF(VSPOIL.LE.VTEST)THEN
        X(29)=X(27)+DH1*2./TANT
        Y(29)=YDIST
        X(32)=X(27)+DH2*2./TANT
        Y(32)=YDIST
      ELSEIF(VSPOIL.GT.VTEST.AND.VOBSP.LE.VTEST)THEN
        X(29)=X(27)+DH1*2./TANT
        Y(29)=YDIST
        X(32)=X(31)+0.5*WP
        Y(32)=Y(31)-0.5*WP*TANT
      ELSE
        X(29)=X(28)+0.5*WP
        Y(29)=Y(28)-0.5*WP*TANT
        X(32)=X(31)+0.5*WP
        Y(32)=Y(31)-0.5*WP*TANT
      ENDIF
      X(33)=X(27)+WP
      Y(33)=YDIST
      X(34)=X(25)
      Y(34)=YDIST
      X(35)=0.0
      Y(35)=0.0
C
      CALL SPAGE(3)
      CALL PLOT2(X,Y,N,NUM,VALUE,NAME,H,OROB,ORIB)
C
      IF(NUM.EQ.1.OR.NUM.EQ.2)THEN
         ISW=.TRUE.
         ISW2=.TRUE.
      ENDIF
      GO TO 10
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE REH10
C
C   PURPOSE:
C
C   THIS SUBROUTINE IS USED TO CALCULATE VOLUMES FOR DIFFERENT
C   SINGLE SEAM PIT DESIGNS AND TO USE AN EXTENDED BENCH IF
C   NECESSARY.  IT IS ASSUMED THAT THE COAL FACE IS VERTICAL,
C   THE EXTENDED BENCH SLOPE IS THE SAME AS THAT OF THE SPOIL
C   SLOPE, THE FINAL SPOIL SLOPE STARTS AT THE BASE OF THE COAL
C   SEAM, AND THE COAL SEAM IS HORIZONTAL.
C
C   INPUT:
C
C  AN INTERACTIVE INPUT SCREEN IS USED SO THAT AFTER THE INITIAL
C   VALUES ARE READ IN FROM A PARAMETER FILE, THE VALUES OF THE
C   DIFFERENT VARIABLES MAY BE CHANGED TO TRY DIFFERENT
C   COMBINATIONS.
C
C   INTERNAL VARIABLES:
C
C   NAME= CHARACTER*25 VECTOR CONTAINING THE NAMES OF THE DATA
C      VARIABLES
C   VALUE= REAL*4 VECTOR CONTAINING THE VALUES OF THE DATA
C       VARIABLES
C   NDAT= INTEGER*2 VARIABLE CONTAINING THE NUMBER OF DATA
C     VALUES WHICH WILL APPEAR ON THE INPUT SCREEN
C   CH= CHARACTER*1 VARIABLE CONTAINING THE ASCII CHARACTER READ
C       FROM THE KEYBOARD
C   SCAN= INTEGER*2 VARIABLE CONTAINING THE EXTENDED CODE VALUE
C       READ FROM THE KEYBOARD
C   FILE1= CHARACTER*30 STRING CONTAINING THE NAME OF THE DATA
C       FILE WHICH WILL CONTAIN THE DEFAULT DATA VALUES
C   WC= REAL*4 VARIABLE CONTAINING THE WIDTH OF DRAGLINE CUT (m)
C   WP= REAL*4 VARIABLE CONTAINING THE WIDTH OF SPOIL PIT (m)
C   WCH= REAL*4 VARIABLE CONTAINING THE WIDTH OF CHOPCUT (m)
C   TOB= REAL*4 VARIABLE CONTAINING THE THICKNESS OF THE O/B (m)
C   TCH= REAL*4 VARIABLE CONTAINING THE CHOPCUT THICKNESS (m)
C   TTC= REAL*4 VARIABLE CONTAINING THE COAL SEAM THICKNESS (m)
C   THETA= REAL*4 VARIABLE CONTAINING THE SPOIL ANGLE (DEGREES)
C   BETA=REAL*4 VARIABLE CONTAINING THE HIGHWALL ANGLE (DEGREES)
C   SWOB= REAL*4 VARIABLE CONTAINING THE O/B SWELL FACTOR
C   PO= REAL*4 VARIABLE CONTAINING THE POSITIONING FACTOR
C   D= REAL*4 VARIABLE CONTAINING THE DRAGLINE TUB DIAMETER (m)
C   DOR= REAL*4 VARIABLE CONTAINING THE DRAGLINE OPERATING
C       RADIUS (m)
C   X,Y= REAL*4 VECTORS CONTAINING THE (X,Y) POINT PAIRS OF
C       THE RANGE DIAGRAM
C
C   OUTPUT:
```

```
C
C  CALCULATED VALUES ARE WRITTEN TO THE SCREEN AND THEN A
C  CHOICE IS GIVEN IN MENU FORMAT AS TO THE TYPE OF OUTPUT
C  DESIRED.  THE PROGRAM IS SET UP TO SUPPORT BOTH EGA AND CGA
C  GRAPHICS.THE RANGE DIAGRAM CAN ALSO BE SENT TO A PRINTER OR
C  PLOTTER FOR A HARD COPY.  THE DEFAULT PRINTER IS THE EPSON
C  LQ500 WHILE THE DEFAULT PLOTTER IS THE HEWLETT PACKARD
C  HP7475A.  IF ANOTHER PRINTER OR PLOTTER IS DESIRED, A FILE
C  CALLED PRINT.DAT CAN BE USED TO OVER-RIDE THE DEFAULT
C  SPECIFICATIONS.
C  PRINT.DAT SHOULD CONTAIN 3 LINES:
C
C       LINE 1 - IOPORT VALUE (USUALLY SET = 0)
C       LINE 2 - MODEL VALUE (LQ500 = 42)
C       LINE 3 - WIDTH OF OUTPUT CONTROL
C                    (0 = NARROW CARRIAGE, 1 = WIDE CARRIAGE)
C
C  ALL OF THE VALUES ARE READ FROM PRINT.DAT IN FREE FORMAT.
C
C  ERRORS:
C
C  PROGRAM DISPLAYS THE FOLLOWING ERROR MESSAGES:
C
C      1. THE DATA FILE DOES NOT CONTAIN ## VARIABLES
C
C      2. DRAGLINE POSITIONING FACTOR MUST BE AT LEAST 0.5
C
C      3. DRAGLINE MUST BE POSITIONED ON CHOP OR MAIN CUT
C
C      4. VALID ANGLE RANGE (DEGREES):   0 < ANGLE < 90
C
C  SUBROUTINES REQUIRED:
C
C  READIN= THIS SUBROUTINE IS USED TO READ VARIABLE NAMES AND
C      VALUES FROM FILE1
C  EDMENU= THIS SUBROUTINE IS USED TO CONTROL THE EDITING OF
C      VALUES ON THE INPUT SCREEN.  THE ENTRY CALLED ED IS USED
C      TO INITIALIZE THE LOCATIONS OF THE DATA ON THE SCREEN.
C  CURSOR2= THIS SUBROUTINE IS REQUIRED BY THE SUBROUTINE
C      EDMENU TO CONTROL CURSOR MOVEMENT AND DATA ENTRY ON THE
C      INPUT SCREEN
C INSCRN=THIS SUBROUTINE IS REQUIRED BY THE SUBROUTINE EDMENU
C      TO REWRITE TEXT AND VALUES ON THE SCREEN IF NECESSARY
C  ERRMSG= THIS SUBROUTINE IS USED TO DISPLAY MESSAGES ON THE
C      SCREEN
C  RESULTR= THIS SUBROUTINE IS USED TO WRITE THE CALCULATED
C      RESULTS TO THE SCREEN
C  PLOTR1= THIS SUBROUTINE IS USED TO PLOT THE RANGE DIAGRAM
C      USING THE PLOT88 SOFTWARE GRAPHICS PACKAGE.  THE ENTRY
C      INITR1 IS USED TO INITIALIZE THE PRINTER SPECIFICATIONS.
```

```
C  SPAGE= A FORTRAN UTILITIES ROUTINE CALLED TO SELECT ONE
C     SCREEN PAGE FOR DISPLAYING
C  SCREEN=A FORTRAN UTILITIES ROUTINE CALLED TO SET THE SCREEN
C     MODE AND CLEAR THE SCREEN
C  LOCATE= A FORTRAN UTILITIES ROUTINE CALLED TO POSITION THE
C     CURSOR ON THE SELECTED SCREEN PAGE
C  PUTST= A FORTRAN UTILITIES ROUTINE CALLED TO WRITE A STRING
C     WITH ATTRIBUTE AT THE CURRENT CURSOR POSITION
C
C  FUNCTIONS REQUIRED:
C
C  INKEY= AN INTEGER*2 FORTRAN UTILITIES ROUTINE CALLED TO
C    DETERMINE WHETHER A KEY FROM THE KEYBOARD IS PRESSED AND
C     WHAT IT IS
C
C
      INTEGER*2 N,NUM,NDAT,SCAN,INKEY,NN,IMENU
      PARAMETER(N=20,NN=12)
      REAL*4 WCH,WC,WP
      REAL*4 PO,D
      REAL*4 TCH,TOB,TTC
      REAL*4 BETA,TANB,THETA,TANT,RAD
      REAL*4 SWOB
      REAL*4 ET,EB,E1,E2,RVOL,EVOL,VTEMP,DOR
      REAL*4 H,DH,OROB,ORBAK,VOB,VCHOP,VSPOIL,VTEST
      REAL*4 XDIST,YDIST,X(N+2),Y(N+2),VALUE(NN)
      CHARACTER NAME(NN)*25, FILE1*30, CH*1
      LOGICAL*2 ISW,ISW2
C
      FILE1='REH1.DAT'
      ISW=.TRUE.
      ISW2=.TRUE.
C
C--READ DATA VALUES FROM A DATA FILE
C
      CALL READIN(NAME,VALUE,NDAT,FILE1)
C
C--CHECK IF DATA FILE CONTAINS THE CORRECT NUMBER OF DATA
C  VALUES
C
      IF(NDAT.NE.NN)THEN
        WRITE(*,*)'** ERROR **'
        WRITE(*,5)NN
    5 FORMAT(' THE DATA FILE DOES NOT CONTAIN ',I2,' VARIABLES')
        STOP
      ENDIF
C
C--INITIALIZE PLOT1 AND EDMENU SUBROUTINES
C
      CALL INITR1()
      CALL ED(NDAT,1)
C
C--SET THE SCREEN MODE:  2 SELECTS AN 80x25 BLACK AND WHITE
C  DISPLAY WITH 4 TEXT PAGES
C
      CALL SCREEN(2)
C
   10 CALL EDMENU(NAME,VALUE,ISW,IMENU)
      IF(IMENU.EQ.0)RETURN
C
      WC   = VALUE(1)
      WP   = VALUE(2)
      WCH  = VALUE(3)
      TOB  = VALUE(4)
      TCH  = VALUE(5)
      TTC  = VALUE(6)
      THETA= VALUE(7)
      BETA = VALUE(8)
      SWOB = VALUE(9)
      PO   = VALUE(10)
      D    = VALUE(11)
      DOR  = VALUE(12)
C
C--ANGLE AND DRAGLINE POSITIONING CHECK
C
      IF(PO.LT.0.5)THEN
        CALL ERRMSG(
     +    'DRAGLINE POSITIONING FACTOR MUST BE AT LEAST 0.5',48)
      ELSEIF((PO+.5)*D.GT.WCH+WC)THEN
        CALL ERRMSG(
     +    'DRAGLINE MUST BE POSITIONED ON CHOP OR MAIN CUT',47)
      ELSEIF(THETA.GE.90..OR.BETA.GE.90..OR.THETA.LE.0..OR.BETA.LE.0.)
     +       THEN
        CALL ERRMSG(
     +          'VALID ANGLE RANGE (DEGREES):   0 < ANGLE < 90',45)
      ELSE
        GO TO 20
      ENDIF
      GO TO 10
   20 RAD=.0174532925
      TANT=TAN(THETA*RAD)
      TANB=TAN(BETA*RAD)
C
C--DETERMINE CUT AND SPOIL VOLUMES
C
      VCHOP=WCH*TCH
      VOB=TOB*WC
      VSPOIL=(VCHOP+VOB)*SWOB
C
C--FIRST TEST IF THE SPOIL COVERS THE SPOIL PIT, THEN
```

```
C  DETERMINE THE OPERATING RADIUS AND HEIGHT REQUIREMENTS OF
C  THE DRAGLINE WHEN IT IS SPOILING FROM THE O/B BENCH
C
      VTEST=WP*WP*0.25*TANT
      IF(VSPOIL.GT.VTEST)THEN
        DH=(VSPOIL+VTEST)/WP
      ELSE
        DH=SQRT(VSPOIL*TANT)
      ENDIF
      OROB=PO*D + TOB/TANB + DH/TANT
      H=DH - TOB - TTC
C
C--CHECK IF REHANDLING IS REQUIRED
C
      IF(OROB.GT.DOR)THEN
        ET=OROB-DOR
        ORBAK=OROB
        OROB=DOR
C
C--CHECK IF EXTENDED BENCH CUTS INTO FINAL SPOIL SLOPE
C
        IF(ET.GT.TOB/TANB+(TOB+TTC)/TANT)THEN
          CALL SPAGE(0)
          CALL LOCATE(0,10,15)
          CALL PUTST(0,7,48,
     +         'EXTENDED BENCH CUTS INTO FINAL SPOIL SLOPE      ')
          CALL LOCATE(0,15,15)
          CALL PUTST(0,7,34,'EXTEND BENCH ONLY TO SPOIL ? (Y/N)')
   90     IF(INKEY(CH,SCAN).EQ.0) GO TO 90
          CALL LOCATE(0,15,15)
          CALL PUTST(0,7,34,'                                  ')
          IF(CH.EQ.'Y'.OR.CH.EQ.'y')THEN
            ET=TOB/TANB+(TOB+TTC)/TANT
            OROB=ORBAK-ET
          ELSE
            GO TO 10
          ENDIF
        ENDIF
        EB=ET+(TOB+TTC)*(1./TANT-1./TANB)
        E1=TTC/TANB
C
C--CHECK IF EXTENDED BENCH MATERIAL RIDES UP NEXT SPOIL SLOPE
C
        IF(E1+EB.GT.WP)THEN
          E2=E1+EB-WP
        ELSE
          E2=0.0
        ENDIF
C
C--CALCULATE REHANDLE AND EXTENDED BENCH VOLUMES
```

```
C
       VTEMP=0.5*((ET+EB)*(TOB+TTC)+TTC*E1)
       RVOL=VTEMP-(EB+E1)*(EB+E1)*0.25*TANT
       EVOL=VTEMP-0.25*E2*E2*TANT
     ELSE
       RVOL=0.0
       EVOL=0.0
       ET=0.0
     ENDIF
     CALL SPAGE(1)
C
C--WRITE OUTPUT VALUES TO SCREEN
C
     CALL RESULTR(VOB,VCHOP,VSPOIL,OROB,H,ET,RVOL,EVOL,ISW2)
C
C--SELECT MODE OF OUTPUT
C
   80 IF(INKEY(CH,SCAN).EQ.0) GO TO 80
     SELECT CASE(ICHAR(CH))
        CASE(48)
          GO TO 10
        CASE(49)
          NUM=1
        CASE(50)
          NUM=2
        CASE(51)
          NUM=3
        CASE(52)
          NUM=4
        CASE(53)
          NUM=5
        CASE DEFAULT
          GO TO 80
     END SELECT
C
C--DETERMINE X AND Y COORDINATES OF RANGE DIAGRAM
C
     XDIST=5.0
     YDIST=15.0
     X(1)=XDIST+WCH+(TOB+TCH)/TANB
     Y(1)=YDIST
     X(2)=0.0
     Y(2)=YDIST
     X(3)=0.0
     Y(3)=TTC+YDIST
     X(4)=X(1)
     Y(4)=TTC+YDIST
     X(5)=XDIST+WCH
     Y(5)=TTC+TOB+TCH+YDIST
     X(6)=0.0
```

```
      Y(6)=Y(5)
      X(7)=XDIST
      Y(7)=Y(5)
      X(8)=XDIST+TCH/TANB
      Y(8)=TTC+TOB+YDIST
      X(9)=0.0
      Y(9)=Y(8)
      X(10)=X(8)+WCH+WC
      Y(10)=Y(8)
      X(11)=X(1)+WC
      Y(11)=TTC+YDIST
      X(12)=X(1)
      Y(12)=TTC+YDIST
      X(13)=X(1)
      Y(13)=YDIST
      X(14)=X(11)
      Y(14)=YDIST
      X(15)=X(11)
      Y(15)=Y(11)
      X(16)=X(14)
      Y(16)=YDIST
      X(17)=X(16)+DH/TANT
      Y(17)=DH+YDIST
C
C--TEST IF SPOIL COVERS SPOIL PIT WIDTH
C
      IF(VSPOIL.GT.VTEST)THEN
        X(18)=X(17)+0.5*WP
        Y(18)=Y(17)-0.5*WP*TANT
      ELSE
        X(18)=X(16)+DH*2./TANT
        Y(18)=YDIST
      ENDIF
      X(19)=X(16)+WP
      Y(19)=YDIST
      X(20)=X(16)
      Y(20)=YDIST
      X(21)=0.0
      Y(21)=0.0
C
      CALL SPAGE(3)
      CALL PLOTR1(X,Y,N,NUM,VALUE,NAME,H,OROB,ET,EB,E1,E2,TANT)
C
      IF(NUM.EQ.1.OR.NUM.EQ.2)THEN
        ISW=.TRUE.
        ISW2=.TRUE.
      ENDIF
      GO TO 10
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE REH2O
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO DETERMINE FEASIBLE DRAGLINE
C  POSITIONNING WHEN MINING TWO SEAMS WITH THE OPTION OF USING
C  AN EXTENDED BENCH. THIS PROGRAM ASSUMES THAT THE COAL SEAM
C  FACE IS VERTICAL, THE ANGLE OF REPOSE IS THE SAME FOR BOTH
C  THE I/B AND O/B, AND HORIZONTAL SEAMS ARE BEING MINED.
C
C  INPUT:
C
C AN INTERACTIVE INPUT SCREEN IS USED SO THAT AFTER THE INITIAL
C  VALUES ARE READ IN FROM A PARAMETER FILE, THE VALUES OF THE
C  DIFFERENT VARIABLES MAY BE CHANGED TO TRY DIFFERENT
C  COMBINATIONS.
C
C  INTERNAL VARIABLES:
C
C  NAME= CHARACTER*25 VECTOR CONTAINING THE NAMES OF THE DATA
C     VARIABLES
C  VALUE= REAL*4 VECTOR CONTAINING THE VALUES OF THE DATA
C      VARIABLES
C NDAT=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF DATA VALUES
C      WHICH WILL APPEAR ON THE INPUT SCREEN
C  CH= CHARACTER*1 VARIABLE CONTAINING THE ASCII CHARACTER READ
C      FROM THE KEYBOARD
C  SCAN= INTEGER*2 VARIABLE CONTAINING THE EXTENDED CODE VALUE
C      READ FROM THE KEYBOARD
C  FILE1= CHARACTER*30 STRING CONTAINING THE NAME OF THE DATA
C      FILE WHICH WILL CONTAIN THE DEFAULT DATA VALUES
C  WCO= REAL*4 VARIABLE CONTAINING THE O/B CUT WIDTH (m)
C  WC= REAL*4 VARIABLE CONTAINING THE I/B CUT WIDTH (m)
C  WP= REAL*4 VARIABLE CONTAINING THE WIDTH OF SPOIL PIT (m)
C  WCH= REAL*4 VARIABLE CONTAINING THE WIDTH OF CHOPCUT (m)
C  WSB= REAL*4 VARIABLE CONTAINING THE WIDTH OF THE SAFETY
C      BENCH (m)
C  WD= REAL*4 VARIABLE CONTAINING THE DITCH WIDTH (m)
C  TOB= REAL*4 VARIABLE CONTAINING THE THICKNESS OF THE O/B (m)
C  TIB= REAL*4 VARIABLE CONTAINING THE THICKNESS OF THE I/B (m)
C  TCH= REAL*4 VARIABLE CONTAINING THE CHOPCUT THICKNESS (m)
C  TTC= REAL*4 VARIABLE CONTAINING THE THICKNESS OF THE UPPER
C      COAL SEAM (m)
C  TBC= REAL*4 VARIABLE CONTAINING THE THICKNESS OF THE LOWER
C      COAL SEAM (m)
C  THETA= REAL*4 VARIABLE CONTAINING THE SPOIL ANGLE (DEGREES)
C  BETA= REAL*4 VARIABLE CONTAINING THE O/B HIGHWALL
C      ANGLE (DEGREES)
C  ALPHA= REAL*4 VARIABLE CONTAINING THE I/B HIGHWALL
```

```
C       ANGLE (DEGREES)
C   GAMA= REAL*4 VARIABLE CONTAINING THE CUT ANGLE INTO THE
C       O/B SPOIL (DEGREES)
C   SWOB= REAL*4 VARIABLE CONTAINING THE O/B SWELL FACTOR
C   SWIB= REAL*4 VARIABLE CONTAINING THE I/B SWELL FACTOR
C   PO= REAL*4 VARIABLE CONTAINING THE O/B POSITIONING FACTOR
C   PI= REAL*4 VARIABLE CONTAINING THE I/B POSITIONING FACTOR
C   PS= REAL*4 VARIABLE CONTAINING THE POSITIONING FACTOR ON
C       THE EXTENDED BENCH
C   D= REAL*4 VARIABLE CONTAINING THE DRAGLINE TUB DIAMETER (m)
C   DOR= REAL*4 VARIABLE CONTAINING THE DRAGLINE OPERATING
C       RADIUS (m)
C   H=REAL*4 VARIABLE CONTAINING THE MAXIMUM DRAGLINE HEIGHT (m)
C   X,Y= REAL*4 VECTORS CONTAINING THE (X,Y) POINT PAIRS OF
C       THE RANGE DIAGRAM
C
C   OUTPUT:
C
C   CALCULATED VALUES ARE WRITTEN TO THE SCREEN AND THEN A
C   CHOICE IS GIVEN IN MENU FORMAT AS TO THE TYPE OF OUTPUT
C   DESIRED.  THE FIRST DECISION IS TO CHOOSE WHETHER THE PLOT
C   SHOULD SHOW ONLY WHERE THE O/B IS TO BE SPOILED.THIS ALLOWS
C THE PROGRAM TO BE USED WITH OTHER REHANDLING TECHNIQUES. THE
C   SECOND OPTION OF THE FIRST DECISION IS TO PLOT THE FINAL
C   SPOIL DIAGRAM SHOWING ALL OF THE DRAGLINE POSITIONS.  THIS
C   PLOT WILL STILL SHOW THE O/B SPOIL, BUT IT WILL NOW BE IN
C   DASHED LINES.  THIS PROGRAM ONLY CONSIDERS UP TO 2 PEAKS TO
C   BE USE IN THE FINAL SPOIL PILE.
C   THE SECOND DECISION ALLOWS THE USER TO EITHER SEND THE PLOT
C   TO A PLOTTER, PRINTER OR SCREEN, OR RETURN TO THE INPUT
C   SCREEN.  THE PROGRAM IS SET UP TO SUPPORT BOTH EGA AND CGA
C   GRAPHICS.THE RANGE DIAGRAM CAN ALSO BE SENT TO A PRINTER OR
C   PLOTTER FOR A HARD COPY.  THE DEFAULT PRINTER IS THE EPSON
C   LQ500 WHILE THE DEFAULT PLOTTER IS THE HEWLETT PACKARD
C   HP7475A. IF ANOTHER PRINTER OR PLOTTER IS DESIRED, A FILE
C   CALLED PRINT.DAT CAN BE USED TO OVER-RIDE THE DEFAULT
C   SPECIFICATIONS.
C   PRINT.DAT SHOULD CONTAIN 3 LINES:
C
C       LINE 1 - IOPORT VALUE (USUALLY SET = 0)
C       LINE 2 - MODEL VALUE (LQ500 = 42)
C       LINE 3 - WIDTH OF OUTPUT CONTROL
C                   (0 = NARROW CARRIAGE, 1 = WIDE CARRIAGE)
C
C   ALL OF THE VALUES ARE READ FROM PRINT.DAT IN FREE FORMAT.
C
C   ERRORS:
C
C   PROGRAM DISPLAYS THE FOLLOWING ERROR MESSAGES:
C
```

```
C        1. THE DATA FILE DOES NOT CONTAIN ## VARIABLES
C
C        2. DRAGLINE POSITIONING FACTORS MUST BE AT LEAST 0.5
C
C        3. DRAGLINE MUST BE POSITIONED WITHIN CHOPCUT ON O/B
C
C        4. DRAGLINE MUST BE POSITIONED WITHIN CUT ON I/B
C
C        5. I/B WIDTH MUST BE <= WIDTH OF O/B + SAFETY BENCH
C
C        6. I/B HIGHWALL ANGLE MUST BE > SPOIL PILE ANGLE
C
C        7. CUT ANGLE IN SPOIL MUST BE > SPOIL PILE ANGLE
C
C        8. WIDTH OF DITCH MUST BE >= 0.0
C
C        9. WIDTH OF DITCH IS TOO LARGE
C
C       10. VALID ANGLE RANGE (DEGREES):  0 < ANGLE < 90
C
C       11. DRAGLINE BOOM DOES NOT EVEN REACH I/B HIGHWALL
C
C       12. O/B SPOIL SPILLS ONTO THE I/B BENCH
C
C       13. TOTAL SPOIL PEAK MUST BE HIGHER THAN I/B BENCH
C
C       14. AREA IS NOT A SIMPLE CLOSED POLYGON
C
C       15. TOO MUCH SPOIL FOR ONLY 2 SPOIL PEAKS
C
C  SUBROUTINES REQUIRED:
C
C  READIN= THIS SUBROUTINE IS USED TO READ VARIABLE NAMES AND
C      VALUES FROM FILE1
C  EDMENU= THIS SUBROUTINE IS USED TO CONTROL THE EDITING OF
C      VALUES ON THE INPUT SCREEN.  THE ENTRY CALLED ED IS USED
C      TO INITIALIZE THE LOCATIONS OF THE DATA ON THE SCREEN.
C  CURSOR2= THIS SUBROUTINE IS REQUIRED BY THE SUBROUTINE
C      EDMENU TO CONTROL CURSOR MOVEMENT AND DATA ENTRY ON THE
C     INPUT SCREEN
C INSCRN=THIS SUBROUTINE IS REQUIRED BY THE SUBROUTINE EDMENU
C      TO REWRITE TEXT AND VALUES ON THE SCREEN IF NECESSARY
C  ERRMSG= THIS SUBROUTINE IS USED TO DISPLAY MESSAGES ON THE
C      SCREEN
C  RESULTR2= THIS SUBROUTINE IS USED TO WRITE THE CALCULATED
C      RESULTS TO THE SCREEN
C  PLOTR2= THIS SUBROUTINE IS USED TO PLOT THE RANGE DIAGRAM
C      DISPLAYING ONLY THE O/B SPOIL.
C      THE PLOT88 SOFTWARE GRAPHICS PACKAGE IS USED.
C      THE ENTRY INITR2 IS USED TO INITIALIZE THE PRINTER
```

```
C       SPECIFICATIONS.
C  PLOTR3= THIS SUBROUTINE IS USED TO PLOT THE RANGE DIAGRAM
C       DISPLAYING BOTH THE O/B SPOIL AND THE I/B SPOIL.
C       THE PLOT88 SOFTWARE GRAPHICS PACKAGE IS USED.
C       THE ENTRY INITR3 IS USED TO INITIALIZE THE PRINTER
C       SPECIFICATIONS.
C  SPAGE= A FORTRAN UTILITIES ROUTINE CALLED TO SELECT ONE
C     SCREEN PAGE FOR DISPLAYING
C  SCREEN=A FORTRAN UTILITIES ROUTINE CALLED TO SET THE SCREEN
C     MODE AND CLEAR THE SCREEN
C  LOCATE= A FORTRAN UTILITIES ROUTINE CALLED TO POSITION THE
C     CURSOR ON THE SELECTED SCREEN PAGE
C  PUTST= A FORTRAN UTILITIES ROUTINE CALLED TO WRITE A STRING
C     WITH ATTRIBUTE AT THE CURRENT CURSOR POSITION
C  BISECT= THIS SUBROUTINE USES THE BISECTION METHOD TO SOLVE
C       FOR A ROOT OF A CONTINUOUS FUNCTION, GIVEN TWO
C       BOUNDING STARTING POINTS
C  SMPCLP= THIS SUBROUTINE IS USED TO CHECK IF A SET OF (X,Y)
C       POINTS REPRESENT A SIMPLE CLOSED POLYGON
C  LNINTS= THIS SUBROUTINE IS REQUIRED BY SUBROUTINE SMPCLP
C       AND IS USED TO TEST IF TWO GIVEN LINE SEGMENTS
C       INTERSECT ONE ANOTHER
C
C  FUNCTIONS REQUIRED:
C
C  INKEY= AN INTEGER*2 FORTRAN UTILITIES ROUTINE CALLED TO
C    DETERMINE WHETHER A KEY FROM THE KEYBOARD IS PRESSED AND
C       WHAT IT IS
C  F= REAL*4 FUNCTION USED TO SOLVE THE VALUE OF A POLYNOMIAL
C       F(X) WHEN X IS GIVEN
C  AREA= REAL*4 FUNCTION USED TO SOLVE THE AREA OF A POLYGON
C       BY THE METHOD OF COORDINATES
C
C
   INTEGER*2 N,NUM,NDAT,SCAN,INKEY,NN,NC,NCOEF,LIMIT,NLIM
   INTEGER*2 NIT,PNUM,I,IFLAG,LTST,IJS,KLS,IMENU
   PARAMETER(N=29,NN=23,NC=5,LIMIT=100)
   REAL*4 WCH,WC,WCO,WP,WSB,WD,WI,HI,HR,HTEST,HR1
   REAL*4 PO,D,PI,PS
   REAL*4 TCH,TOB,TTC,TIB,TBC
   REAL*4 BETA,THETA,ALPHA,GAMA
   REAL*4 SWOB,SWIB,TANG
   REAL*4 RAD,TANB,TANA,TANT,COST,SINT,SINAT,SIN18A,SIN9A,SIN9T
   REAL*4 SINAPT,TAN9T
   REAL*4 H,VOB,VIB,VCHOP,VSPOIL,VOBSP,VIBSP,X1,X2,Y2,Y3
   REAL*4 A1,A2,A3,A4,B1,B2,B3,B4,D1,D2,D3,DX,DX2,VOLMAX
   REAL*4 XDIST,YDIST,X(N+2),Y(N+2),VALUE(NN),DOR,YD,EX(4),EY(4)
   REAL*4 F,COEF(NC),YLOW,YHIGH,YR,FR,RATIO,AX1,AY1,AX2,AY2
   REAL*4 A,XTEMP,HX(3),HY(3),VOL1,VOL2,XR,XR2,DIST,WA,AREA
   REAL*4 SPX(6),SPY(6),EB,REH,ANG,YTEST,XA(10),YA(10),S1,S2,S3
```

```
      CHARACTER NAME(NN)*25, FILE1*30, CH*1
      LOGICAL*2 ISW,ISW2
      CHARACTER*10 DUMFF, BLANK
      EXTERNAL F
C
      DATA BLANK/'          '/
C
      FILE1='REH2.DAT'
      ISW=.TRUE.
      ISW2=.TRUE.
C
C--READ DATA VALUES FROM A DATA FILE
C
      CALL READIN(NAME,VALUE,NDAT,FILE1)
C
C--CHECK IF DATA FILE CONTAINS THE CORRECT NUMBER OF DATA
C  VALUES
C
      IF(NDAT.NE.NN)THEN
        WRITE(*,*)'** ERROR **'
        WRITE(*,5)NN
    5 FORMAT(' THE DATA FILE DOES NOT CONTAIN ',I2,' VARIABLES')
        STOP
      ENDIF
C
C--INITIALIZE PLOT2 AND EDMENU SUBROUTINES
C
      CALL INITR2()
      CALL INITR3()
      CALL ED(NDAT,0)
C
C--SET THE SCREEN MODE:  2 SELECTS AN 80x25 BLACK AND WHITE
C  DISPLAY WITH 4 TEXT PAGES
C
      CALL SCREEN(2)
C
   10 CALL EDMENU(NAME,VALUE,ISW,IMENU)
      IF(IMENU.EQ.0)RETURN
      CALL SPAGE(1)
C
      WCO  = VALUE(1)
      WC   = VALUE(2)
      WP   = VALUE(3)
      WCH  = VALUE(4)
      WSB  = VALUE(5)
      IF(VALUE(6).EQ.0.00)VALUE(6)=0.0001
      WD   = VALUE(6)
      TOB  = VALUE(7)
      TIB  = VALUE(8)
      TCH  = VALUE(9)
      TTC  = VALUE(10)
      TBC  = VALUE(11)
      THETA= VALUE(12)
      BETA = VALUE(13)
      ALPHA= VALUE(14)
      GAMA = VALUE(15)
      SWOB = VALUE(16)
      SWIB = VALUE(17)
      PO   = VALUE(18)
      PI   = VALUE(19)
      PS   = VALUE(20)
      D    = VALUE(21)
      DOR  = VALUE(22)
      H    = VALUE(23)
C
      RAD=.0174532925
      TANB=TAN(BETA*RAD)
      TANA=TAN(ALPHA*RAD)
      TANT=TAN(THETA*RAD)
      TANG=TAN(GAMA*RAD)
      COST=COS(THETA*RAD)
      SINT=SIN(THETA*RAD)
C
C--INPUT DATA ERROR CHECKING
C
      IF(PO.LT.0.5.OR.PI.LT.0.5.OR.PS.LT.0.5)THEN
        CALL ERRMSG(
     +   'DRAGLINE POSITIONING FACTORS MUST BE AT LEAST 0.5',49)
      ELSEIF((PO+.5)*D.GT.WCH+WCO)THEN
        CALL ERRMSG(
     +  'DRAGLINE MUST BE POSITIONED WITHIN CHOPCUT ON O/B',49)
      ELSEIF((PI+.5)*D.GT.WC)THEN
     CALL ERRMSG(
     +   'DRAGLINE MUST BE POSITIONED WITHIN CUT ON I/B',45)
      ELSEIF(WC.GT.WCO+WSB)THEN
        CALL ERRMSG(
     + 'I/B WIDTH MUST BE <= WIDTH OF O/B + SAFETY BENCH',48)
      ELSEIF(ALPHA.LE.THETA)THEN
        CALL ERRMSG(
     + 'I/B HIGHWALL ANGLE MUST BE > SPOIL PILE ANGLE',45)
      ELSEIF(GAMA.LE.THETA)THEN
        CALL ERRMSG(
     +'CUT ANGLE IN SPOIL MUST BE > SPOIL PILE ANGLE',45)
      ELSEIF(WD.LT.0.0)THEN
        CALL ERRMSG('WIDTH OF DITCH MUST BE >= 0.0',29)
      ELSEIF(WD.GT.(TIB+TBC)/TANT)THEN
        CALL ERRMSG('WIDTH OF DITCH IS TOO LARGE',27)
      ELSEIF(THETA.LE.0..OR.BETA.LE.0..OR.ALPHA.LE.0..OR.THETA.GE.90.
     +          .OR.BETA.GE.90..OR.ALPHA.GE.90..OR.GAMA.GE.90)THEN
        CALL ERRMSG(
     + 'VALID ANGLE RANGE (DEGREES):  0 < ANGLE < 90',45)
      ELSE
        GO TO 20
      ENDIF
      GO TO 10
C
C
C--DETERMINE CUT AND SPOIL VOLUMES
C
20    VCHOP=WCH*TCH
      VOB=TOB*WCO
      VIB=TIB*WC
      VOBSP=(VCHOP+VOB)*SWOB
      VIBSP=VIB*SWIB
      VSPOIL=VIBSP+VOBSP
C
C--THERE ARE THREE DIFFERENT CASES THAT MUST BE CONSIDERED
C  WHEN DEALING WITH THE DRAGLINE O/B SPOIL
C
C    CASE 1: THE DRAGLINE REACH ENDS ABOVE THE I/B HIGHWALL
C    CASE 2: THE DRAGLINE REACH ENDS OVER THE SPOIL PIT
C    CASE 3: THE DRAGLINE REACH ENDS BEYOND THE SPOIL PIT
C
C  THREE DIFFERENT VALUES WILL NOW BE CALCULATED ACCORDING TO
C  THE SPOIL CONFIGURATION
C
C  THE OPERATING RADIUS REQUIREMENTS WILL BE DETERMINED FOR
C  THE DRAGLINE WHEN IT IS SPOILING FROM THE O/B AND I/B
C
C  THE HEIGHT REQUIREMENTS WILL ALSO BE DETERMINED FOR WHEN
C  THE DRAGLINE IS OPERATING ON THE I/B BENCH
C
C
      D1=PO*D+TOB/TANB+WSB
      IF(DOR.LE.D1)THEN
        CALL ERRMSG(
     +  'DRAGLINE BOOM DOES NOT EVEN REACH I/B HIGHWALL',46)
        GO TO 10
      ENDIF
      D2=D1+TIB/TANA
      D3=D2+WP
      DX=DOR-D1
      B1=2.*(TIB+TBC+DX*TANT)/TANT
      B2=(TBC+TIB)*(1./TANT+1./TANA)
      B3=TBC/TANA
      B4=B1-B2+B3-WP
      IF(B4.LT.0.)THEN
        B4=0.
        A4=0.
      ELSE
      A4=B4*B4*0.25*TANT
    ENDIF
    A1=0.5*B1*(TIB+TBC+DX*TANT)
    A2=0.5*(TBC+TIB)*B2
    A3=0.5*TBC*B3
    VOLMAX=A1-A2+A3-A4
    IF(VOBSP.GT.VOLMAX)THEN
      CALL ERRMSG('O/B SPILLS ONTO THE I/B BENCH',35)
      GO TO 10
    ENDIF
C
C--WRITE OUTPUT VALUES TO SCREEN
C
    CALL RESULTR2(VOB,VIB,VCHOP,VSPOIL,ISW2)
C
  25 IF(INKEY(CH,SCAN).EQ.0) GO TO 25
    SELECT CASE(ICHAR(CH))
      CASE(48)
        GO TO 10
      CASE(49)
        PNUM=1
      CASE(50)
        PNUM=2
      CASE DEFAULT
        GO TO 25
    END SELECT
C
    VMIN=.5*(TIB+TBC)*(2.*(TIB+TBC)/TANT-WD)
    IF(PNUM.EQ.2.AND.VSPOIL.LT.VMIN)THEN
      CALL ERRMSG(
   + 'TOTAL SPOIL PEAK MUST BE HIGHER THAN I/B BENCH',43)
      GO TO 10
    ENDIF
    NCOEF=3
    RATIO=.001
    NLIM=LIMIT
    SINAT=SIN((ALPHA-THETA)*RAD)
    SIN18A=SIN((180.-ALPHA)*RAD)
    SIN9A=SIN((90.-ALPHA)*RAD)
    SIN9T=SIN((90.-THETA)*RAD)
    TAN9T=TAN((90.-THETA)*RAD)
    SINAPT=SIN((ALPHA+THETA)*RAD)
C-------------------------------------------------------------
    IF(DOR.LE.D2)THEN
C-------------------------------------------------------------
C--CALCULATE VOL1
    XTEMP=TBC/TANT
    VOL1=0.5*TBC*XTEMP
    IF(XTEMP.GT.WP)VOL1=VOL1-0.25*TANT*(XTEMP-WP)*(XTEMP-WP)
C--CALCULATE VOL2
```

```
     XR=(D2-DOR)*((TANA/TANT)-1.)
     VOL2=XR*(TBC+0.5*TANA*(D2-DOR))+VOL1
     IF(XR+XTEMP.GT.WP.AND.XTEMP.LE.WP)THEN
       VOL2=VOL2-0.25*(XTEMP+XR-WP)*(XTEMP+XR-WP)*TANT
     ELSEIF(XTEMP.GT.WP)THEN
       VOL2=VOL2-XR*XR*0.25*TANT-0.25*XR*(XTEMP-WP)/(TANT*COST)
     ENDIF
     IF(VOBSP.LE.VOL1)THEN
        A=WP*WP*TANT*0.5
       IF(VOBSP.LE.A)THEN
CASE A(1)
        HX(1)=0.
        HX(2)=SQRT(2.*VOBSP/TANT)
        HY(1)=HX(2)*TANT
        HY(2)=0.
       ELSE
CASE A(2)
        COEF(1)=A-VOBSP
        COEF(2)=WP
        COEF(3)=0.25/TANT
        YLOW=0.
        YHIGH=TBC
        CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
        HX(1)=0.
        HY(1)=WP*TANT+YR
        HX(2)=WP+0.5*YR/TANT
        HY(2)=0.5*YR
       ENDIF
       HX(3)=HX(2)
       HY(3)=HY(2)
     ELSEIF(VOBSP.LE.VOL2.AND.VOBSP.GT.VOL1)THEN
       IF(XTEMP+XR.LE.WP)THEN
CASE A(3)
        COEF(1)=VOL1-VOBSP
        COEF(2)=TBC
        COEF(3)=0.5*SINT*SIN18A/SINAT
        YLOW=0.
        YHIGH=XR
        CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
       HY(1)=TBC+YR*SINT*SIN18A/SINAT
       HX(1)=-(HY(1)-TBC)/TANA
       HX(2)=XTEMP+YR
       HY(2)=0.0
       ELSEIF(XTEMP.LE.WP.AND.XTEMP+XR.GT.WP)THEN
A=VOL1+(WP-XTEMP)*TBC+0.5*(WP-XTEMP)*(WP-XTEMP)*SIN18A*SINT/SINAT
        IF(VOBSP.LE.A)THEN
CASE A(4)
        COEF(1)=VOL1-VOBSP
        COEF(2)=TBC
        COEF(3)=0.5*SINT*SIN18A/SINAT
```

```
          YLOW=0.
          YHIGH=XR
          CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
          HY(1)=TBC+YR*SINT*SIN18A/SINAT
          HX(1)=-(HY(1)-TBC)/TANA
          HX(2)=XTEMP+YR
          HY(2)=0.0
          ELSE
 CASE A(5)
          COEF(1)=A-VOBSP-0.25*TANT*(XTEMP*XTEMP+WP*WP-2.*WP*XTEMP)
          COEF(2)=TBC-0.5*TANT*(XTEMP-WP)
          COEF(3)=0.5*SIN18A*SINT/SINAT-0.25*TANT
          YLOW=0.
          YHIGH=XTEMP+XR-WP
          CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
          HY(1)=TBC+(WP-XTEMP+YR)*SINT/SINAT
          HX(1)=-(HY(1)-TBC)/TANA
          HX(2)=WP+0.5*YR
          HY(2)=YR*TANT*0.5
          ENDIF
        ELSE
 CASE A(6)
          COEF(1)=VOL1-VOBSP+((XTEMP-WP)*(XTEMP-WP)
     +                -(XTEMP*XTEMP+WP*WP-2.*WP*XTEMP))*0.25*TANT
          COEF(2)=TBC+0.5*TANT*(WP-XTEMP)
          COEF(3)=0.5*SIN18A*SINT/SINAT-0.25*TANT
          YLOW=0.
          YHIGH=XR
          CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
        HY(1)=TBC+YR*SINT/SINAT
        HX(1)=-(HY(1)-TBC)/TANA
        HX(2)=WP+0.5*(YR+XTEMP-WP)
        HY(2)=(YR+XTEMP-WP)*TANT*0.5
        ENDIF
        HX(3)=HX(2)
        HY(3)=HY(2)
      ELSE
        DIST=TANT*SIN9A*SIN9T/SINAPT
        XR2=(TIB-TANA*(D2-DOR))/(TANA*DIST)
        IF(XTEMP+XR+XR2.LE.WP)THEN
 CASE A(7)
          COEF(1)=VOL2-VOBSP
          COEF(2)=(D2-DOR+XTEMP+XR)*TANT
          COEF(3)=0.5*TANT*(1.+DIST)
          YLOW=0.
          YHIGH=XR2
          CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
          HX(1)=DOR-D2-YR*DIST
          HY(1)=TBC+(D2-DOR+YR*DIST)*TANA
          HX(2)=DOR-D2
          HY(2)=TBC+(D2-DOR)*TANA+YR*TANT
          HX(3)=XTEMP+XR+YR
          HY(3)=0.
        ELSEIF(XTEMP+XR.LE.WP.AND.XTEMP+XR+XR2.GT.WP)THEN
          WA=WP-XTEMP-XR
          A=VOL2+(D2-DOR+XTEMP+XR+0.5*WA*(1.+DIST))*TANT*WA
          IF(VOBSP.LE.A)THEN
CASE A(8)
            COEF(1)=VOL2-VOBSP
            COEF(2)=(D2-DOR+XTEMP+XR)*TANT
            COEF(3)=0.5*TANT*(1.+DIST)
            YLOW=0.
            YHIGH=XR2
          CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
            HX(1)=DOR-D2-YR*DIST
            HY(1)=TBC+(D2-DOR+YR*DIST)*TANA
            HX(2)=DOR-D2
            HY(2)=TBC+(D2-DOR)*TANA+YR*TANT
            HX(3)=XTEMP+XR+YR
            HY(3)=0.
          ELSE
CASE A(9)
            COEF(1)=VOL2-VOBSP-WA*WA*TANT*0.25
            COEF(2)=(D2-DOR+XTEMP+XR+0.5*WA)*TANT
            COEF(3)=0.5*TANT*(.5+DIST)
            YLOW=0.
            YHIGH=XR2
          CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
            HX(1)=DOR-D2-YR*DIST
            HY(1)=TBC+(D2-DOR+YR*DIST)*TANA
            HX(2)=DOR-D2
            HY(2)=TBC+(D2-DOR)*TANA+YR*TANT
            HX(3)=WP+0.5*(YR-WA)
            HY(3)=0.5*(YR-WA)*TANT
          ENDIF
        ELSE
CASE A(10)
          WA=XTEMP+XR-WP
          COEF(1)=VOL2-VOBSP
          COEF(2)=TANT*(D2-DOR+XTEMP+XR-0.5*WA)
          COEF(3)=0.5*TANT*(0.5+DIST)
          YLOW=0.
          YHIGH=XR2
          CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
          HX(1)=DOR-D2-YR*DIST
          HY(1)=TBC-HX(1)*TANA
          HX(2)=DOR-D2
          HY(2)=TBC+(D2-DOR)*TANA+YR*TANT
          HX(3)=WP+0.5*(YR+WA)
          HY(3)=0.5*(YR+WA)*TANT
        ENDIF
      ENDIF
C---------------------------------------------------------------
      ELSEIF(DOR.GT.D2.AND.DOR.LE.D3)THEN
C---------------------------------------------------------------
      DX2=DOR-D2
      XTEMP=2.*DX2
      VOL1=0.25*XTEMP*XTEMP*TANT
      IF(XTEMP.GT.WP)VOL1=VOL1-0.25*TANT*(XTEMP-WP)*(XTEMP-WP)
      XR=TBC/TANT
      VOL2=(DX2+XR)*(DX2+XR)*TANT-0.5*TBC*TBC/TANT
      IF(XTEMP+XR.GT.WP)
     +VOL2=VOL2-0.25*(XTEMP+XR-WP)*(XTEMP+XR-WP)*TANT
      IF(VOBSP.LE.VOL1)THEN
        IF(XTEMP.LE.WP)THEN
CASE A(11)
          YR=SQRT(VOBSP/TANT)
          HX(1)=DX2-YR
          HY(1)=0.0
          HX(2)=DX2
          HY(2)=YR*TANT
          HX(3)=DX2+YR
          HY(3)=0.0
        ELSE
          A=TANT*(WP-DX2)*(WP-DX2)
          IF(VOBSP.GE.A)THEN
CASE A(12)
            COEF(1)=0.25*TANT*(2.*DX2*WP-DX2*DX2-WP*WP)-VOBSP
            COEF(2)=0.5*TANT*(WP-DX2)
            COEF(3)=0.75*TANT
            YLOW=WP-DX2
            YHIGH=DX2
          CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
            HX(1)=DX2-YR
            HY(1)=0.0
            HX(2)=DX2
            HY(2)=YR*TANT
            HX(3)=WP+(DX2+YR-WP)*0.5
            HY(3)=(DX2+YR-WP)*0.5*TANT
          ELSE
CASE A(13)
            YR=SQRT(VOBSP/TANT)
            HX(1)=DX2-YR
            HY(1)=0.0
            HX(2)=DX2
            HY(2)=YR*TANT
            HX(3)=DX2+YR
            HY(3)=0.0
          ENDIF
        ENDIF
```

```
    ELSEIF(VOBSP.LE.VOL2.AND.VOBSP.GT.VOL1)THEN
      IF(XTEMP+XR.LE.WP)THEN
CASE A(14)
         COEF(1)=DX2*DX2*TANT-VOBSP
         COEF(2)=2.0*TANT*DX2
         COEF(3)=0.5*TANT
         YLOW=0.0
         YHIGH=XR
        CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
         HX(1)=0.0
         HY(1)=YR*TANT
         HX(2)=DX2
         HY(2)=(DX2+YR)*TANT
         HX(3)=XTEMP+YR
         HY(3)=0.0
       ELSE
        IF(WP.GE.XTEMP)THEN
         A=TANT*((WP-DX2)*(WP-DX2)-0.5*(WP-XTEMP)*(WP-XTEMP))
        ELSE
         A=0.
        ENDIF
        IF(VOBSP.GE.A)THEN
CASE A(15)
         COEF(1)=-VOBSP+DX2*DX2*TANT-0.25*TANT*(XTEMP*XTEMP+WP*WP
   +                      -2.*WP*XTEMP)
         COEF(2)=(2.*DX2-0.5*(XTEMP-WP))*TANT
         COEF(3)=0.25*TANT
         IF(WP.GE.XTEMP)THEN
          YLOW=WP-XTEMP
         ELSE
          YLOW=0.0
         ENDIF
         YHIGH=XR
        CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
         HX(1)=0.0
         HY(1)=YR*TANT
         HX(2)=DX2
         HY(2)=(DX2+YR)*TANT
         HX(3)=WP+0.5*(YR+XTEMP-WP)
         HY(3)=0.5*(YR+XTEMP-WP)*TANT
        ELSE
CASE A(16)
         COEF(1)=DX2*DX2*TANT-VOBSP
         COEF(2)=2.0*TANT*DX2
         COEF(3)=0.5*TANT
         YLOW=0.0
         YHIGH=WP-XTEMP
        CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
         HX(1)=0.0
         HY(1)=YR*TANT
```

```
          HX(2)=DX2
          HY(2)=(DX2+YR)*TANT
          HX(3)=XTEMP+YR
          HY(3)=0.0
         ENDIF
        ENDIF
     ELSE
        DIST=TANT*SIN9A*SIN9T/SINAPT
        XR2=TIB/(TANA*DIST)
        IF(WP.GE.XTEMP+XR+XR2)THEN
CASE A(17)
          COEF(1)=(0.5*(XTEMP*XTEMP+XR*XR+2.*XR*XTEMP)-DX2*DX2)*TANT
     +                     -VOBSP
          COEF(2)=TANT*(XTEMP+XR)
          COEF(3)=0.5*TANT*(1.+DIST)
          YLOW=0.0
          YHIGH=XR2
         CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
          HX(1)=-YR*DIST
          HY(1)=TBC+YR*DIST*TANA
          HX(2)=DX2
          HY(2)=(DX2+XR+YR)*TANT
          HX(3)=XTEMP+XR+YR
          HY(3)=0.0
        ELSEIF(WP.GE.XTEMP+XR.AND.WP.LT.XTEMP+XR+XR2)THEN
         IF(WP.LE.XTEMP+XR)THEN
          A=0.
         ELSE
          DX=WP-XTEMP-XR
          A=VOL2+DX*TANT*(XTEMP+XR+0.5*DX*(1.+DIST))
         ENDIF
         IF(VOBSP.LE.A)THEN
CASE A(18)
          COEF(1)=(0.5*(XTEMP*XTEMP+XR*XR+2.*XR*XTEMP)-DX2*DX2)*TANT
     +                     -VOBSP
          COEF(2)=TANT*(XTEMP+XR)
          COEF(3)=0.5*TANT*(1.+DIST)
          YLOW=0.0
          YHIGH=XR2
         CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
          HX(1)=-YR*DIST
          HY(1)=TBC+YR*DIST*TANA
          HX(2)=DX2
          HY(2)=(DX2+XR+YR)*TANT
          HX(3)=XTEMP+XR+YR
          HY(3)=0.0
         ELSE
CASE A(19)
          COEF(1)=0.25*TANT*(XTEMP*XTEMP+XR*XR-WP*WP+2.*XR*XTEMP
     +                     +2.*WP*XTEMP+2.*XR*WP)-VOBSP-DX2*DX2*TANT
```

```
          COEF(2)=TANT*0.5*(XTEMP+XR+WP)
          COEF(3)=TANT*(0.25+0.5*DIST)
          YLOW=0.0
          YHIGH=XR2
         CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
          HX(1)=-YR*DIST
          HY(1)=TBC+YR*DIST*TANA
          HX(2)=DX2
          HY(2)=(DX2+XR+YR)*TANT
          HX(3)=WP+0.5*(XTEMP+XR+YR-WP)
          HY(3)=0.5*(XTEMP+XR+YR-WP)*TANT
         ENDIF
        ELSE
CASE A(20)
          COEF(1)=0.25*TANT*(XTEMP*XTEMP+XR*XR-WP*WP+2.*XR*XTEMP
   +                    +2.*WP*XTEMP+2.*XR*WP)-VOBSP-DX2*DX2*TANT
          COEF(2)=TANT*0.5*(XTEMP+XR+WP)
          COEF(3)=TANT*(0.25+0.5*DIST)
          YLOW=0.0
          YHIGH=XR2
         CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
          HX(1)=-YR*DIST
          HY(1)=TBC+YR*DIST*TANA
          HX(2)=DX2
          HY(2)=(DX2+XR+YR)*TANT
          HX(3)=WP+0.5*(XTEMP+XR+YR-WP)
          HY(3)=0.5*(XTEMP+XR+YR-WP)*TANT
        ENDIF
      ENDIF
C-----------------------------------------------------------------
      ELSE
C-----------------------------------------------------------------
        DX=DOR-D3
        DX2=TBC/TANT
        VOL1=TANT*WP*(0.75*WP+DX)
        VOL2=TANT*(0.25*DX2*DX2+0.75*WP*WP+1.5*DX2*WP+DX2*DX+WP*DX)
        IF(VOBSP.LE.VOL1)THEN
CASE A(21)
         COEF(1)=-VOBSP
         COEF(2)=TANT*DX
         COEF(3)=TANT*0.75
         YLOW=0.0
         YHIGH=WP
         CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
         HX(1)=WP-YR
         HY(1)=0.0
         HX(2)=WP+DX
         HY(2)=(YR+DX)*TANT
         HX(3)=WP+0.5*(YR+2.*DX)
         HY(3)=0.5*(2.*DX+YR)*TANT
      ELSEIF(VOBSP.GT.VOL1.AND.VOBSP.LE.VOL2)THEN
CASE A(22)
        COEF(1)=-VOBSP+TANT*WP*(0.75*WP+DX)
        COEF(2)=TANT*(1.5*WP+DX)
        COEF(3)=TANT*0.25
        YLOW=0.0
        YHIGH=DX2
        CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
        HX(1)=0.0
        HY(1)=YR*TANT
        HX(2)=WP+DX
        HY(2)=(YR+WP+DX)*TANT
        HX(3)=WP+0.5*(YR+WP+2.*DX)
        HY(3)=0.5*(2.*DX+WP+YR)*TANT
      ELSE
CASE A(23)
        DIST=TANT*SIN9A*SIN9T/SINAPT
        COEF(1)=VOL2-VOBSP
        COEF(2)=1.5*WP+DX+0.5*DX2
        COEF(3)=0.25*TAN9T+0.5*DIST/TANT
        YLOW=0.0
        YHIGH=TIB*TANT/(TANA*DIST)
        CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
        HX(1)=-YR*DIST/TANT
        HY(1)=TBC+YR*DIST*TANA/TANT
        HX(2)=WP+DX
        HY(2)=TBC+YR+(WP+DX)*TANT
        HX(3)=WP+0.5*(YR/TANT+WP+2.*DX+DX2)
        HY(3)=0.5*(YR/TANT+WP+2.*DX+DX2)*TANT
      ENDIF
C-----------------------------------------------------------
    ENDIF
C-----------------------------------------------------------
C
C--SELECT MODE OF OUTPUT
C
    CALL LOCATE(1,22,5)
    CALL PUTST(1,7,57,
    +           'Select # for output:   0  INPUT SCREEN 3  PRINTER (1:500)')
    CALL LOCATE(1,23,5)
    CALL PUTST(1,7,58,
    + '                    1  CGA SCREEN   4  PRINTER (1:1000)')
    CALL LOCATE(1,24,5)
    CALL PUTST(1,7,58,
    + '                    2  EGA SCREEN   5  PLOTTER (1:1000)')
C
C--DETERMINE X AND Y COORDINATES OF RANGE DIAGRAM
C
    XDIST=5.0
    YDIST=15.0
```

```
X(1)=XDIST+WCH+(TOB+TCH)/TANB+(WSB+WCO-WC)+TIB/TA
Y(1)=YDIST
X(2)=0.0
Y(2)=YDIST
X(3)=0.0
Y(3)=TBC+YDIST
X(4)=X(1)
Y(4)=TBC+YDIST
X(5)=X(4)-TIB/TANA
Y(5)=Y(4)+TIB
X(6)=0.0
Y(6)=Y(5)
X(7)=0.0
Y(7)=Y(6)+TTC
X(8)=X(5)-(WSB+WCO-WC)
Y(8)=Y(7)
X(9)=XDIST+WCH
Y(9)=Y(8)+TCH+TOB
X(10)=0.0
Y(10)=Y(9)
X(11)=XDIST
Y(11)=Y(9)
X(12)=X(11)+TCH/TANB
Y(12)=Y(9)-TCH
X(13)=0.0
Y(13)=Y(12)
X(14)=X(12)+WCH+WCO
Y(14)=Y(12)
X(15)=X(8)+WCO
Y(15)=Y(8)
X(16)=X(8)
Y(16)=Y(8)
X(17)=X(8)
Y(17)=Y(5)
X(18)=X(17)+WCO
Y(18)=Y(5)
X(19)=X(15)
Y(19)=Y(15)
X(20)=X(18)
Y(20)=Y(18)
X(21)=X(20)+WSB
Y(21)=Y(20)
X(22)=X(4)+WC
Y(22)=Y(4)
X(23)=X(4)
Y(23)=Y(4)
X(24)=X(1)
Y(24)=YDIST
X(25)=X(1)+WC
Y(25)=YDIST
```

```
      X(26)=X(22)
      Y(26)=Y(22)
      X(27)=X(25)
      Y(27)=YDIST
      X(28)=X(27)+WP
      Y(28)=YDIST
      X(29)=X(28)+AMAX1(Y(9)-YDIST,HY(2))/TANT
      Y(29)=YDIST+AMAX1(Y(9)-YDIST,HY(2))
      X(30)=0.0
      Y(30)=0.0
C
C--ADJUST CORDINATES OF O/B SPOIL TO SAME AXES AS RANGE DIAGRAM
C
      HX(1)=X(27)+HX(1)
      HX(2)=X(27)+HX(2)
      HX(3)=X(27)+HX(3)
      HY(1)=YDIST+HY(1)
      HY(2)=YDIST+HY(2)
      HY(3)=YDIST+HY(3)
C
      IF(PNUM.EQ.2)THEN
C--HR=MAX HEIGHT OF SPOIL WITH DRAGLINE ON I/B BENCH
        HR=AMIN1(H,TANT*(DOR-PI*D-TIB/TANA-(TIB+TBC)/TANT))
        HTEST=(VSPOIL+0.25*WP*WP*TANT+0.5*WD*(TIB+TBC))/WP-TBC-TIB
          SPX(1)=X(27)+WD
          SPY(1)=YDIST
          SPX(2)=X(27)+(TIB+TBC)/TANT
          SPY(2)=YDIST+TIB+TBC
          SPX(3)=X(27)+(TIB+TBC+HTEST)/TANT
          SPY(3)=YDIST+TIB+TBC+HTEST
          SPX(4)=SPX(3)+0.5*WP
          SPY(4)=SPY(3)-0.5*WP*TANT
          IF(SPX(4).LT.HX(3))THEN
            WI=HX(1)-X(27)
            HI=(VIBSP+0.25*WI*WI*TANT+0.5*WD*(TIB+TBC))/WI-TBC-TIB
            SPX(3)=X(27)+(TIB+TBC+HI)/TANT
            SPY(3)=YDIST+TIB+TBC+HI
            SPX(4)=SPX(3)+0.5*WI
            SPY(4)=SPY(3)-0.5*WI*TANT
            SPX(5)=HX(2)
            SPY(5)=HY(2)
            SPX(6)=HX(3)
            SPY(6)=HY(3)
          ELSE
            SPX(5)=SPX(4)
            SPY(5)=SPY(4)
            SPX(6)=SPX(4)
            SPY(6)=SPY(4)
          ENDIF
          IF(SPY(3).GT.Y(29))THEN
          Y(29)=SPY(3)
          X(29)=X(28)+(Y(29)-YDIST)/TANT
         ENDIF
C--HR1=MAX HEIGHT OF SPOIL WITH DRAGLINE ON EXTENDED BENCH
        HR1=AMIN1(H,TANT*(DOR-PS*D))
C
C--CALCULATE REHANDLE VOLUME WHEN NO EXTENDED BENCH NEEDED
C
        IF(HR.GE.HTEST)THEN
         IFLAG=1
         EB=0.0
         ANG=ATAN2(TIB+TBC,SPX(2)-SPX(1))
         YTEST=(SPX(1)-HX(1))/(1./TANT-1./TAN(ANG))
         IF(HX(1).GE.X(27)+WD)THEN
CASES B(1)-B(3)
          REH=0.0
         ELSEIF(HX(3).LT.X(27)+WD)THEN
CASES B(4)-B(9)
          REH=VOBSP
         ELSEIF(HY(1).EQ.YDIST.AND.HX(2).GT.SPX(1).AND.
     +            HY(2).GE.YTEST+YDIST)THEN
CASES B(10)-B(12)
          REH=0.5*(SPX(1)-HX(1))*YTEST
         ELSE
CASES B(13)-B(26)
          XA(1)=SPX(1)
          YA(1)=SPY(1)
          IF(HY(2).GE.HY(1))THEN
           AX1=HX(2)
           AY1=HY(2)
           I=3
          ELSE
           AX1=HX(1)
           AY1=HY(1)
           I=2
          ENDIF
          AX2=HX(I)
          AY2=HY(I)
          S1=(AY1-AY2)/(AX1-AX2)
          IF(AX1.LT.SPX(2).AND.AY1-(SPX(2)-AX1)*TANT.LT.SPY(2))THEN
           S2=(SPY(1)-SPY(2))/(SPX(1)-SPX(2))
           XA(2)=(AY1+AX1*TANT-SPY(1)+SPX(1)*TAN(ANG))/(S2-S1)
           I=2
          ELSE
           S3=(SPY(2)-SPY(3))/(SPX(2)-SPX(3))
           XA(2)=SPX(2)
           YA(2)=SPY(2)
           XA(3)=(AY1+AX1*TANT-SPY(2)+SPX(2)*TANT)/(S3-S1)
           I=3
          ENDIF
          YA(I) = AY1 + AX1*TANT + XA(I)*S1
          I=I+1
          XA(I) = AX2
          YA(I) = AY2
          IF(AY2.GT.YDIST)THEN
            I=I+1
            XA(I) = X(27) + WP
            YA(I) = YDIST
          ENDIF
C--CHECK IF POLYGON IS SIMPLE CLOSED
          CALL SMPCLP(XA,YA,I,IJS,KLS,LTST)
          IF(LTST.EQ.1)STOP 'AREA IS NOT A SIMPLE CLOSED POLYGON'
          REH = VOBSP-AREA(XA,YA,I)
        ENDIF
      ELSEIF(HR1.GT.HTEST)THEN
        IFLAG = 2
        EX(1) = X(21)
        EY(1) = Y(21)
        EX(2) = SPX(2)
        EY(2) = SPY(2)
        IF(HY(2).GE.HY(1))THEN
          AX1 = HX(2)
          AY1 = HY(2)
          I=3
        ELSE
          AX1 = HX(1)
          AY1 = HY(1)
          I=2
        ENDIF
        AX2 = HX(I)
        AY2 = HY(I)
        S1 = (AY1-AY2)/(AX1-AX2)
        IF(AY1 + AX1*TANT + SPX(2)*S1.GT.SPY(2))THEN
          EX(3) = SPX(2)
          EY(3) = SPY(2)
        ELSE
          DX = SPX(2)-X(27)
          IF(SPX(2) + DX.LE.X(28))THEN
            EX(3) = SPX(2) + DX
            EY(3) = Y(27)
          ELSE
            EX(3) = X(28) + DX-0.5*WP
            EY(3) = Y(28) + (DX-0.5*WP)*TANT
          ENDIF
        ENDIF
        XA(1) = X(27)
        YA(1) = YDIST
        XA(2) = X(27)
        YA(2) = YDIST + TBC
        XA(3) = X(21)
        YA(3)=Y(21)
        XA(4)=SPX(2)
        YA(4)=SPY(2)
        XA(5)=SPX(1)
        YA(5)=YDIST
        I=5
C--CHECK IF POLYGON IS SIMPLE CLOSED
        CALL SMPCLP(XA,YA,I,IJS,KLS,LTST)
        IF(LTST.EQ.1)STOP 'AREA IS NOT A SIMPLE CLOSED POLYGON'
        REH=AREA(XA,YA,I)
        EB=SPX(2)-X(21)
        YD=HY(2)-Y(21)
        IF(YD.GT.0..AND.HY(1).GT.YDIST)THEN
          YTEST=YD/TANT
          REH=REH+YD*YTEST
          IF(HX(2)+YTEST.GT.SPX(2))THEN
CASES C(1)-C(4)
          REH=REH-(HX(2)+YTEST-SPX(2))*(HX(2)+YTEST-SPX(2))*TANT*.25
          ENDIF
CASES C(5)-C(8)
        ENDIF
CASES C(9)-C(26)
      ELSE
        VOLMAX=0.125*WP*WP*TANT
        XA(1)=SPX(2)+HR1/TANT
        YA(1)=SPY(2)+HR1
        XA(2)=SPX(3)
        YA(2)=SPY(3)
        XA(3)=XA(2)+0.5*WP
        YA(3)=YA(2)-TANT*0.5*WP
        XA(4)=XA(1)+0.5*WP
        YA(4)=YA(1)-TANT*0.5*WP
        I=4
C--CHECK IF POLYGON IS SIMPLE CLOSED
        CALL SMPCLP(XA,YA,I,IJS,KLS,LTST)
        IF(LTST.EQ.1)STOP 'AREA IS NOT A SIMPLE CLOSED POLYGON'
        VOL1=AREA(XA,YA,I)
        IF(VOL1.GE.VOLMAX)THEN
          CALL ERRMSG('TOO MUCH SPOIL FOR ONLY 2 SPOIL PEAKS',37)
          GO TO 10
        ENDIF
          IFLAG=3
        XA(1)=X(27)
        YA(1)=YDIST
        XA(2)=X(27)
        YA(2)=YDIST+TBC
        XA(3)=X(21)
        YA(3)=Y(21)
        XA(4)=SPX(2)
        YA(4)=SPY(2)
      XA(5)=SPX(1)
      YA(5)=YDIST
      I=5
C--CHECK IF POLYGON IS SIMPLE CLOSED
      CALL SMPCLP(XA,YA,I,IJS,KLS,LTST)
      IF(LTST.EQ.1)STOP 'AREA IS NOT A SIMPLE CLOSED POLYGON'
      REH=AREA(XA,YA,I)
      YD=HY(2)-Y(21)
      IF(YD.GT.0..AND.HY(1).GT.YDIST)THEN
        YTEST=YD/TANT
        REH=REH+YD*YTEST
        IF(HX(2)+YTEST.GT.SPX(2))THEN
        REH=REH-(HX(2)+YTEST-SPX(2))*(HX(2)+YTEST-SPX(2))*TANT*.25
        ENDIF
      ENDIF
C--USE 2 SPOIL PEAKS TO FIT SPOIL
        COEF(1)=2.*VOL1/TANT
        COEF(2)=-WP
        COEF(3)=1.
        YLOW=0.
        YHIGH=0.5*WP
        CALL BISECT(F,COEF,NCOEF,YLOW,YHIGH,YR,FR,RATIO,NLIM,NIT)
        SPX(4)=SPX(3)+0.5*YR
        SPY(4)=SPY(3)-0.5*YR*TANT
        SPX(5)=SPX(3)+YR
        SPY(5)=SPY(3)
        SPX(6)=SPX(5)+0.5*(WP-YR)
        SPY(6)=SPY(5)-0.5*(WP-YR)*TANT
        EB=SPX(2)-X(21)+YR
        EX(1)=X(21)
        EY(1)=Y(21)
        EX(2)=SPX(2)+YR
        EY(2)=SPY(2)
      IF(HY(2).GE.HY(1))THEN
        AX1=HX(2)
        AY1=HY(2)
        I=3
      ELSE
        AX1=HX(1)
        AY1=HY(1)
        I=2
      ENDIF
      AX2=HX(I)
      AY2=HY(I)
      S1=(AY1-AY2)/(AX1-AX2)
      IF(AY1+AX1*TANT+EX(2)*S1.GT.EY(2))THEN
        Y2=HY(1)-HX(1)*TANT
        Y3=SPY(2)-SPX(2)*TANT
        EX(3)=EX(2)
        EY(3)=EY(2)
        IF(Y3.GT.Y2)THEN
C--CUT ANGLE INTERSECTS ONE OF THE O/B SPOIL SIDES
         X2=(EY(2)-EX(2)*TANG-Y2)/(TANT-TANG)
         X1=(EY(2)-EX(2)*TANG-AY1-AX1*TANT)/(S1-TANG)
         XA(1)=EX(2)
         YA(1)=EY(2)
         XA(2)=(EY(2)-Y2)/TANT
         YA(2)=EY(2)
         IF(X2.GT.X1)THEN
C--CUT ANGLE INTERSECTS NEGATIVE O/B SLOPE
CASE D(1)
            EX(3)=X1
            EY(3)=AY1+AX1*TANT+X1*S1
            XA(3)=HX(2)
            YA(3)=HY(2)
            XA(4)=X1
            YA(4)=EY(3)
            REH=REH+AREA(XA,YA,4)
         ELSE
C--CUT ANGLE INTERSECTS POSITIVE O/B SLOPE
CASE D(2)
            EX(3)=X2
            EY(3)=Y2+X2*TANT
            XA(3)=X2
            YA(3)=EY(3)
            REH=REH+AREA(XA,YA,3)
         ENDIF
        ELSE
C--CUT ANGLE INTERSECTS EITHER THE FINAL SPOIL SIDE OR THE 2ND
C  O/B SIDE
         X2=(EY(2)-EX(2)*TANG-Y3)/(TANT-TANG)
         X1=(EY(2)-EX(2)*TANG-AY1-AX1*TANT)/(S1-TANG)
         XA(1)=EX(2)
         YA(1)=EY(2)
         XA(2)=SPX(2)
         YA(2)=SPY(2)
         IF(X2.GT.X1)THEN
C--CUT ANGLE INTERSECTS NEGATIVE O/B SLOPE
CASES D(3)-D(4)
            EX(3)=X1
            EY(3)=AY1+AX1*TANT+X1*S1
            XA(3)=(AY1+AX1*TANT-Y3)/(TANT-S1)
            YA(3)=Y3+XA(3)*TANT
            XA(4)=X1
            YA(4)=EY(3)
            REH=REH+AREA(XA,YA,4)
         ELSE
C--CUT ANGLE INTERSECTS FINAL SPOIL SLOPE
CASES D(5)-D(6)
            EX(3)=X2
```

```
                EY(3)=Y3+X2*TANT
                XA(3)=X2
                YA(3)=EY(3)
                REH=REH+AREA(XA,YA,3)
              ENDIF
            ENDIF
            S2=(SPY(4)-EY(2))/(SPX(4)-EX(2))
            EX(4)=(AY1-EY(2)+TANT*(AX1+EX(2)))/(S2-S1)
            EY(4)=AY1+AX1*TANT+EX(4)*S1
          ELSE
            DX=SPX(2)-X(27)
            IF(SPX(2)+DX+YR.LE.X(28))THEN
              EX(3)=SPX(2)+DX+YR
              EY(3)=Y(27)
            ELSE
              EX(3)=X(28)+DX+.5*YR-0.5*WP
              EY(3)=Y(28)+(DX+.5*YR-0.5*WP)*TANT
            ENDIF
CASE D(8)
            EX(4)=EX(2)
            EY(4)=EY(2)
            IF(AY1+AX1*TANT+SPX(2)*S1.GT.EY(2))THEN
CASE D(7)
              X1=(EY(2)-AY1-AX1*TANT)/S1
              REH=REH+(X1-SPX(2))*(X1-SPX(2))*0.25*TANT
            ENDIF
          ENDIF
        ENDIF
C--WRITE REHANDLE AND EXTENDED BENCH VALUES ON THE SCREEN
        CALL LOCATE(1,13,5)
        CALL PUTST(1,7,21,'REHANDLE PARAMETERS :')
        CALL LOCATE(1,15,10)
        CALL PUTST(1,7,29,'1. REHANDLE VOLUME (lcm)     =')
        CALL LOCATE(1,17,10)
        CALL PUTST(1,7,29,'2. EXTENDED BENCH WIDTH (m)  =')
        DUMFF=BLANK
        WRITE(DUMFF,'(F10.2)')REH
        CALL LOCATE(1,15,40)
        CALL PUTST(1,120,10,DUMFF)
        DUMFF=BLANK
        WRITE(DUMFF,'(F10.2)')EB
        CALL LOCATE(1,17,40)
        CALL PUTST(1,120,10,DUMFF)
      ENDIF
   80 IF(INKEY(CH,SCAN).EQ.0) GO TO 80
      SELECT CASE(ICHAR(CH))
        CASE(48)
          NUM=0
          IF(PNUM.EQ.2)THEN
       CALL LOCATE(1,13,5)
```

```
  CALL PUTST(1,7,21,'                    ')
  CALL LOCATE(1,15,10)
  CALL PUTST(1,7,41,'                                        ')
  CALL LOCATE(1,17,10)
  CALL PUTST(1,7,41,'                                        ')
    ENDIF
    GO TO 10
  CASE(49)
    NUM=1
  CASE(50)
    NUM=2
  CASE(51)
    NUM=3
  CASE(52)
    NUM=4
  CASE(53)
    NUM=5
  CASE DEFAULT
    GO TO 80
END SELECT
  CALL SPAGE(3)
IF(PNUM.EQ.1)THEN
  CALL PLOTR2(X,Y,N,NUM,VALUE,NAME,HX,HY)
ELSE
  CALL PLOTR3(X,Y,N,NUM,VALUE,NAME,SPX,SPY,HX,HY,EX,EY,IFLAG)
  IF(NUM.GT.2)THEN
  CALL LOCATE(1,13,5)
  CALL PUTST(1,7,21,'                    ')
  CALL LOCATE(1,15,10)
  CALL PUTST(1,7,41,'                                        ')
  CALL LOCATE(1,17,10)
  CALL PUTST(1,7,41,'                                        ')
  ENDIF
ENDIF
IF(NUM.EQ.1.OR.NUM.EQ.2)THEN
   ISW=.TRUE.
   ISW2=.TRUE.
ENDIF
GO TO 10
RETURN
END
```

```
$STORAGE:2
      SUBROUTINE EDMENU(NAME,VALUE,ISW,IMENU)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO CONTROL THE EDITING OF THE DATA
C  ON AN INPUT SCREEN.  THE ENTRY ED MUST BE CALLED PRIOR TO
C  CALLING THIS SUBROUTINE SO THAT ALL OF THE DATA LOCATIONS
C  ARE INITIALIZED AND RETAINED THROUGHOUT THE USEAGE OF THE
C  CALLING PROGRAM.  THERE ARE BASICALLY TWO DIFFERENT
C  ALTERNATIVES THAT CAN BE CARRIED OUT WHEN CALLING THIS
C  SUBROUTINE:
C                (1) REWRITE THE ENTIRE SCREEN WHEN CALLING
C                    AND THEN EDIT THE DATA VALUES
C                (2) EDIT THE DATA VALUES
C
C  INPUT VARIABLES:
C
C  NAME= CHARACTER*25 VECTOR CONTAINING THE NAMES OF THE DATA
C      VARIABLES
C  VALUE= REAL*4 VECTOR CONTAINING THE VALUES OF THE DATA
C       VARIABLES
C  ISW= LOGICAL*2 VARIABLE CONTAINING WHETHER OR NOT THE
C    SCREEN INFORMATION MUST BE REWRITTEN
C ITEST=INTEGER*2 VARIABLE CONTAINING WHICH DATA POSITIONS WILL
C       BE USED.    IF ITEST=1  A TITLE WILL BE WRITTEN ABOVE THE
C       VALUES, OTHERWISE, THE DATA VALUES WILL START FROM THE
C       TOP ROW
C NDAT=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF DATA VALUES
C       WHICH WILL APPEAR ON THE INPUT SCREEN
C
C  INTERNAL VARIABLES:
C
C  ROW= INTEGER*2 VECTOR OF LENGTH 23 CONTAINING THE SEPARATE
C     ROW LOCATION OF EACH DATA VALUE
C COL=INTEGER*2 VARIABLE CONTAINING THE COLUMN LOCATION OF THE
C       FIRST COLUMN OF THE DATA VALUES
C  ILOC= INTEGER*2 VARIABLE CONTAINING THE NUMBER OF THE DATA
C       VALUE WHICH IS TO BE HIGHLITED IF THE SCREEN IS
C       REWRITTEN
C  CH= CHARACTER*1 VARIABLE CONTAINING THE ASCII CHARACTER READ
C       FROM THE KEYBOARD
C  SCAN= INTEGER*2 VARIABLE CONTAINING THE EXTENDED CODE VALUE
C       READ FROM THE KEYBOARD
C
C  OUTPUT:
C
C  NAME,VALUE,NDAT= UNCHANGED
C  ISW= .FALSE.
C
```

```
C  SUBROUTINES REQUIRED:
C
C  SPAGE= A FORTRAN UTILITIES ROUTINE CALLED TO SELECT ONE
C     SCREEN PAGE FOR DISPLAYING
C  SCREEN=A FORTRAN UTILITIES ROUTINE CALLED TO SET THE SCREEN
C     MODE AND CLEAR THE SCREEN
C  INSCRN= THIS SUBROUTINE IS USED TO REWRITE VALUES ON THE
C      INPUT SCREEN
C  CURSOR2= THIS SUBROUTINE IS USED TO MOVE THE HIGHLITED CELL
C      AROUND THE INPUT SCREEN AND ALSO TO EDIT THE VALUES
C
C  FUNCTIONS REQUIRED:
C
C  INKEY= AN INTEGER*2 FORTRAN UTILITIES ROUTINE CALLED TO
C    DETERMINE WHETHER A KEY FROM THE KEYBOARD IS PRESSED AND
C      WHAT IT IS
C
C
      CHARACTER*1 NULL, CH
      INTEGER*2 SCAN, INKEY, ILOC, N, NDAT, IMENU
      INTEGER*2 ROW(23), COL, J, ITEST
      CHARACTER NAME(*)*25
      REAL*4 VALUE(*)
      LOGICAL*2 ISW
C
      IMENU=1
      CALL SPAGE(2)
C
C--CHECK IF THE INPUT SCREEN NEEDS TO BE REWRITTEN
C
      IF(ISW)THEN
        ILOC=1
        CALL INSCRN(ILOC,N,NAME,VALUE,COL,ROW)
        ISW=.FALSE.
      ENDIF
   10 IF (INKEY(CH,SCAN) .EQ. 0) GO TO 10
C
C--CHECK IF ESC WAS PRESSED TO QUIT
C
      IF(ICHAR(CH).EQ.27)THEN
         IMENU=0
         RETURN
      ENDIF
C
C--CHECK IBM EXTENDED CHARACTER CODE WAS PRESSED
C
      IF(CH.EQ.NULL)
     +CALL CURSOR2(CH,SCAN,ILOC,VALUE,N,ROW,COL,IMENU)
      IF(IMENU.EQ.0)RETURN
C
```

```
C--CHECK IF A NUMBER HAS BEEN ENTERED DIRECTLY WITHOUT USING
C  F1
C
     SELECT CASE(ICHAR(CH))
     CASE(46,48:57)
       CALL CURSOR2(CH,SCAN,ILOC,VALUE,N,ROW,COL,IMENU)
     IF(IMENU.EQ.0)RETURN

     ENDSELECT
C
C--CHECK IF SPACE BAR WAS PRESSED TO CONTINUE
C
     IF(ICHAR(CH).NE.32)GO TO 10
C
     RETURN
C
C--ENTRY TO INITIALIZE THE LOCATIONS OF THE DATA VALUES ON THE
C  SCREEN
C
     ENTRY ED(NDAT,ITEST)
C
     NULL=CHAR(+8#00)
     N=NDAT
     COL=15
     DO 100 J=1,N
       IF(ITEST.EQ.1)THEN
         ROW(J)=J+2
       ELSE
         ROW(J)=J-1
       ENDIF
 100 CONTINUE
C
     RETURN
     END
```

```
$STORAGE:2
      SUBROUTINE CURSOR2(CH,SCAN,ILOC,VALUE,N,ROW,COL,IMENU)
C
C   PURPOSE:
C
C   THIS SUBROUTINE IS USED TO MOVE THE HIGHLITED AREA AROUND
C   AND TO EDIT THE DATA VALUES IN THE HIGHLITED AREA.
C
C   INPUT:
C
C THE DIMENSION VALUE FIELD SHOULD BE CHECKED BEFORE USING
C THIS SUBROUTINE.IT DEFINES THE FIELD WIDTH OF THE HIGHLITED
C AREA AND IS SET = 10.
C
C   INPUT VARIABLES:
C
C   CH= CHARACTER*1 VARIABLE CONTAINING THE ASCII CHARACTER READ
C       FROM THE KEYBOARD
C   SCAN= INTEGER*2 VARIABLE CONTAINING THE EXTENDED CODE VALUE
C        READ FROM THE KEYBOARD
C   VALUE= REAL*4 VECTOR CONTAINING THE VALUES OF THE DATA
C        VARIABLES
C   N= INTEGER*2 VARIABLE CONTAINING THE NUMBER OF DATA VALUES
C        WHICH WILL APPEAR ON THE INPUT SCREEN
C   ROW= INTEGER*2 VECTOR OF LENGTH 23 CONTAINING THE SEPARATE
C       ROW LOCATION OF EACH DATA VALUE
C  COL=INTEGER*2 VARIABLE CONTAINING THE COLUMN LOCATION OF THE
C        FIRST COLUMN OF THE DATA VALUES
C   ILOC= INTEGER*2 VARIABLE CONTAINING THE NUMBER OF THE DATA
C        VALUE WHICH IS TO BE HIGHLITED IF THE SCREEN IS
C        REWRITTEN
C
C   INTERNAL VARIABLES:
C
C   DUMF= CHARACTER*1 VECTOR OF LENGTH=FIELD WHICH IS USED TO
C        WRITE AND READ REAL VALUES TO
C  DUMFF= CHARACTER*10 VARIABLE WHICH IS ALSO USED TO WRITE AND
C        READ REAL VALUES TO.  THE EQUIVALENCE STATEMENT IS
C        USED WITH THE DUMFF STRING AND THE DUMF CHARACTER ARRAY
C        SO THAT A EITHER THE ENTIRE STRING OR AN INDIVIDUAL
C        ELEMENT CAN BE REFERENCED
C
C   OUTPUT:
C
C   ALL OUTPUT IS WRITTEN TO THE SCREEN ON PAGE 2
C
C   OUTPUT VARIABLES:
C
C   ILOC,VALUE,N,ROW,COL= UNCHANGED
C
```

```
C  ERRORS:
C
C    -THE ALPHABET CHARACTERS ARE LOCKED OUT WHEN THE USER IS
C      CHANGING VALUES IN EDIT MODE.
C    - ONLY ONE DECIMAL POINT CAN BE ENTERED WHEN EDITING.
C    - ESC CAN BE PRESSED AT ANY TIME TO QUIT THE PROGRAM
C
C  SUBROUTINES REQUIRED:
C
C  SCREEN=A FORTRAN UTILITIES ROUTINE CALLED TO SET THE SCREEN
C     MODE AND CLEAR THE SCREEN
C  LOCATE= A FORTRAN UTILITIES ROUTINE CALLED TO POSITION THE
C     CURSOR ON THE SELECTED SCREEN PAGE
C  PUTST= A FORTRAN UTILITIES ROUTINE CALLED TO WRITE A STRING
C     WITH ATTRIBUTE AT THE CURRENT CURSOR POSITION
C  PUTCHA= A FORTRAN UTILITIES ROUTINE CALLED TO WRITE A
C     CHARACTER WITH ATTRIBUTE AT THE CURRENT CURSOR POSITION
C NEWCUR=A FORTRAN UTILITIES ROUTINE CALLED TO DEFINE THE SIZE
C     AND SHAPE OF THE TEXT CURSOR
C
C  FUNCTIONS REQUIRED:
C
C  INKEY= AN INTEGER*2 FORTRAN UTILITIES ROUTINE CALLED TO
C    DETERMINE WHETHER A KEY FROM THE KEYBOARD IS PRESSED AND
C      WHAT IT IS
C
C
   INTEGER*2 FIELD
   INTEGER*2 SCAN, ILOC, SCAN2, INKEY, IDEC, N, IMENU
   INTEGER*2 ITEMP, II
   PARAMETER(FIELD=10)
   INTEGER*2 ROW(*), COL, I
   CHARACTER*10 DUMFF, BLANK
   CHARACTER*1 DUMF(FIELD), CH, CH2, NULL
   EQUIVALENCE (DUMF(1),DUMFF)
   REAL*4 VALUE(*)
   LOGICAL*2 ISW
C
C   SCAN VALUES: 72 = UP ARROW
C                80 = DOWN ARROW
C                59 = FUNCTION KEY 1
C
   NULL=CHAR(+8#00)
C
   IF(CH.EQ.NULL)THEN
     IF(SCAN.NE.72 .AND. SCAN.NE.80 .AND. SCAN.NE.59) RETURN
     WRITE(DUMFF,'(F10.2)')VALUE(ILOC)
   ELSE
     SCAN=59
   ENDIF
```

```
      ISW=.TRUE.
      DATA BLANK/'          '/
      CALL LOCATE(2,ROW(ILOC),COL)
C
C--CHECK IF F1 IS PRESSED IN ORDER TO EDIT DATA
C
      IF(SCAN.EQ.59)THEN
C--WRITE EDIT SPACE ON SCREEN
         CALL PUTST(2,120,FIELD,BLANK)
C--WRITE PREVIOUS VALUE BESIDE EDIT SPACE
         WRITE(DUMFF,'(F10.2)')VALUE(ILOC)
         CALL LOCATE(2,ROW(ILOC),COL-12)
         CALL PUTST(2,7,FIELD,DUMFF)
         CALL NEWCUR(10,10)
C
      DUMFF=BLANK
      IDEC=0
      ITEMP=1
      II=1
C--BACKSPACING IN EDIT SPACE
    5 IF(ITEMP.GE.1)THEN
         II=ITEMP
         IF(IDEC.EQ.II)IDEC=0
         DUMF(II)=' '
         CALL LOCATE(2,ROW(ILOC),COL-1+II)
         CALL PUTCHA(2,120,1,' ')
      ENDIF
C--LOOP TO READ THE CHARACTERS ENTERED IN EDIT SPACE
      DO 10 I=II,FIELD
         CALL LOCATE(2,ROW(ILOC),COL-1+I)
C
C--CHECK IF A NUMBER WAS ENTERED DIRECTLY WITHOUT THE USE OF
C  F1
C
        IF(CH.NE.NULL.AND.I.EQ.1)THEN
         CH2=CH
         CH=NULL
         GO TO 25
        ENDIF
   20  IF(INKEY(CH2,SCAN2).EQ. 0) GO TO 20
C
C--CHECK IF ESC WAS PRESSED TO QUIT
C
      IF(ICHAR(CH2).EQ.27)THEN
         IMENU=0
         RETURN
      ENDIF
         IF(CH2.EQ.NULL)THEN
          SELECT CASE(SCAN2)
C
```

```
C--CHECK IF F1 IS PRESSED IN ORDER TO QUIT EDITING AND USE THE
C  PREVIOUS VALUE
C
          CASE(59)
            GO TO 35
C
C--CHECK IF A CURSOR KEY WAS PRESSED TO SAVE THE VALUE ENTERED
C  AND MOVE THE CURSOR
C
          CASE(72,80)
            ISW=.FALSE.
            IF(IDEC.EQ.0)THEN
              IDEC=I
              DUMF(I)='.'
            ENDIF
            GO TO 30
          END SELECT
        ENDIF
C
C--CHECK FOR 0-9 AND DECIMAL
  25 SELECT CASE(ICHAR(CH2))
        CASE(48:57,46)
C--CHECK FOR MORE THAN ONE DECIMAL
           IF(ICHAR(CH2).EQ.46.AND.IDEC.NE.0)GO TO 20
C--MARK LOCATION OF DECIMAL IN STRING
           IF(ICHAR(CH2).EQ.46)IDEC=I
C--ONLY ALLOW 7 CHARACTERS TO THE LEFT OF THE DECIMAL
           IF(ICHAR(CH2).NE.46.AND.IDEC.EQ.0.AND.I.EQ.8)GO TO 20
C--WRITE CHARACTER IN EDIT SPACE
           WRITE(DUMF(I),'(A)')CH2
           CALL PUTCHA(2,120,1,CH2)
C--CHECK FOR CARRIAGE RETURN
        CASE(13)
           IF(IDEC.EQ.0)THEN
             IDEC=I
             DUMF(I)='.'
           ENDIF
           GO TO 30
C--CHECK FOR BACKSPACE KEY PRESSED IN ORDER TO ERASE PREVIOUS
C  CHARACTER
        CASE(8)
           ITEMP=I-1
           GO TO 5
        CASE DEFAULT
           GO TO 20
        END SELECT
 10  CONTINUE
 30     READ(DUMFF,'(F10.2)')VALUE(ILOC)
 35     WRITE(DUMFF,'(F10.2)')VALUE(ILOC)
        CALL LOCATE(2,ROW(ILOC),COL)
        CALL NEWCUR(32,0)
C--ERASE VALUE BESIDE EDIT SPACE
        CALL LOCATE(2,ROW(ILOC),COL-FIELD)
        CALL PUTST(2,7,FIELD,BLANK)
        IF(ISW)THEN
          CALL PUTST(2,120,FIELD,DUMFF)
          RETURN
        ENDIF
      ENDIF
      CALL PUTST(2,7,FIELD,DUMFF)
C
C--CHECK IF UP OR DOWN ARROW KEYS HAVE BEEN PRESSED
C
      IF(ISW)SCAN2=SCAN
      IF(SCAN2.EQ.72)THEN
        ILOC=ILOC-1
        IF(ILOC.LT.1)ILOC=N
      ELSEIF(SCAN2.EQ.80)THEN
        ILOC=ILOC+1
        IF(ILOC.GT.N)ILOC=1
      ENDIF
C
C--HIGHLIGHT CURRENT VALUE SELECTED
C
      WRITE(DUMFF,'(F10.2)')VALUE(ILOC)
      CALL LOCATE(2,ROW(ILOC),COL)
      CALL PUTST(2,120,FIELD,DUMFF)
C
      RETURN
      END
      SUBROUTINE ERRMSG(MESS,N)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS CALLED TO DISPLAY MESSAGES ON PAGE 0
C  AND THEN WAIT FOR THE USER TO PRESS A KEY TO CONTINUE
C
C  INPUT:
C
C  THE MESSAGE LENGTH IS RESTRICTED TO 50 CHARACTERS
C
C  INPUT VARIABLES:
C
C  N= INTEGER*2 VARIABLE CONTAINING THE NUMBER OF CHARACTERS
C       WITHIN THE MESSAGE CONTAINED IN MESS
C  MESS= CHARACTER*1 ARRAY OF LENGTH N WHICH CONTAINS THE
C     MESSAGE THAT IS TO BE DISPLAYED
C
C  OUTPUT:
C
C  OUTPUT IS WRITTEN TO THE SCREEN ON PAGE 0
C
C  INTERNAL VARIABLE:
C
C M= CHARACTER*1 ARRAY OF LENGTH 50 WHICH IS USED TO STORE THE
C       MESSAGE IN.  IF THE MESSAGE IS LESS THAN 50 CHARACTERS IN
C       LENGTH, THE REST OF THE ARRAY IS FILLED WITH BLANKS
C
C  ERRORS:
C
C  THE PROGRAM WILL BE TERMINATED IF THE MESSAGE LENGTH (N) IS
C  LARGER THAN 50.
C
C  SUBROUTINES REQUIRED:
C
C  SPAGE= A FORTRAN UTILITIES ROUTINE CALLED TO SELECT ONE
C     SCREEN PAGE FOR DISPLAYING
C  LOCATE= A FORTRAN UTILITIES ROUTINE CALLED TO POSITION THE
C     CURSOR ON THE SELECTED SCREEN PAGE
C  PUTST= A FORTRAN UTILITIES ROUTINE CALLED TO WRITE A STRING
C     WITH ATTRIBUTE AT THE CURRENT CURSOR POSITION
C
C  FUNCTIONS REQUIRED:
C
C  INKEY= A FORTRAN UTILITIES ROUTINE CALLED TO DETERMINE
C   WHETHER A KEY FROM THE KEYBOARD IS PRESSED AND WHAT IT IS
C
C
      INTEGER*2 N,I,SCAN,INKEY
      CHARACTER MESS(N)*1,CH*1,M(50)*1
```

```
C
     IF(N.GT.50)STOP' ERROR MESSAGE IS TOO LONG'
     DO 10 I=1,50
       IF(I.LE.N)THEN
         M(I)=MESS(I)
       ELSE
         M(I)=' '
       ENDIF
10    CONTINUE
     CALL SPAGE(0)
     CALL LOCATE(0,10,15)
     CALL PUTST(0,7,50,M)
     CALL LOCATE(0,15,15)
     CALL PUTST(0,7,23,'PRESS A KEY TO CONTINUE')
20    IF(INKEY(CH,SCAN).EQ.0)GO TO 20
     RETURN
     END
```

```
$STORAGE:2
      SUBROUTINE INSCRN(ILOC,N,NAME,VALUE,COL,ROW)
C
C   PURPOSE:
C
C   THIS SUBROUTINE IS USED TO REWRITE THE VALUES OF THE INPUT
C   SCREEN.THE INPUT SCREEN GETS ERASED WHEN THE PROGRAM USES
C   GRAPHICS MODE.
C
C   INPUT VARIABLES:
C
C   NAME=  CHARACTER*[illegible] VECTOR CONTAINING THE NAMES OF THE DATA
C        PARAMETERS
C   VALUE=  REAL*4 VECTORS CONTAINING THE VALUES OF THE DATA
C        PARAMETERS
C   N=  INTEGER*2 VARIABLE CONTAINING THE NUMBER OF DATA VALUES
C COL=INTEGER*2 VARIABLE CONTAINING THE COLUMN NUMBER THAT THE
C       DATA VALUES START THEIR DISPLAY ON THE SCREEN
C  ROW=INTEGER*2 VECTOR OF LENGTH N CONTAINING THE ROW NUMBERS
C        OF THE DATA VALUE POSITIONS
C  ILOC=INTEGER*2 VARIABLE CONTAINING THE ELEMENT NUMBER IN THE
C        ROW ARRAY WHICH WILL BE HIGHLITED UPON RETURN.
C
C   OUTPUT:
C
C THE INPUT SCREEN VALUES ARE WRITTEN TO THE SCREEN ON PAGE 2
C
C   SUBROUTINES REQUIRED:
C
C   SCCHAR=  A FORTRAN UTILITIES ROUTINE CALLED TO DETERMINE THE
C       CHARACTER VARIABLE ON THE SPECIFIED PAGE AT THE CURSOR
C       POSITION
C   LOCATE=  A FORTRAN UTILITIES ROUTINE CALLED TO POSITION THE
C       CURSOR ON THE SELECTED SCREEN PAGE
C   PUTST=  A FORTRAN UTILITIES ROUTINE CALLED TO WRITE A STRING
C       WITH ATTRIBUTE AT THE CURRENT CURSOR POSITION
C NEWCUR=  A FORTRAN UTILITIES ROUTINE CALLED TO DEFINE THE SIZE
C       AND SHAPE OF THE TEXT CURSOR
C
C
      CHARACTER*10 BLANK,DUMFF
      CHARACTER DUMF(10)*1, NAME(*)*25
      EQUIVALENCE(DUMF(1),DUMFF)
      INTEGER*2 COL,ROW(*),I
      INTEGER*2 N,J, ILOC, ATTRIB
      REAL*4 VALUE(*)
      DATA BLANK/'          '/
C
C--WRITE HEADING TO SCREEN
C
```

```
      IF(ROW(1).GT.1)THEN
        CALL LOCATE(2,0,COL+5)
        CALL PUTST(2,7,22,'**** INPUT SCREEN ****')
      ENDIF
C
C--WRITE VARIABLE NAMES AND VALUES TO SCREEN
C
      DO 10 J=1,N
        CALL LOCATE(2,ROW(J),30)
        CALL PUTST(2,7,25,NAME(J))
        DUMFF=BLANK
        WRITE(DUMFF,'(F10.2)')VALUE(J)
        CALL LOCATE(2,ROW(J),COL)
        CALL PUTST(2,7,10,DUMFF)
 10   CONTINUE
C
      DO 20 I=1,10
        CALL LOCATE(2,ROW(ILOC),COL-1+I)
20      CALL SCCHAR(2,DUMF(I),ATTRIB)
      CALL LOCATE(2,ROW(ILOC),COL)
      CALL PUTST(2,120,10,DUMFF)
      CALL NEWCUR(32,0)
C
      CALL LOCATE(2,24,17)
      CALL PUTST(2,7,46,
     +'ESC=QUIT,  F1=EDIT/UNDO EDIT,  SPACE=CALCULATE')
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE READIN(NAME,VALUE,NDAT,FILE1)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO READ VARIABLE NAMES AND VALUES
C  FROM FILE1.
C
C  INPUT:
C
C  THE FOLLOWING DIMENSION PARAMETER SHOULD BE CHECKED IN THIS
C  PROGRAM:
C
C       MAXDAT=MAXIMUM NUMBER OF DATA VALUES (SET AT 24)
C
C  ONE INPUT FILE (UNIT 5) IS REQUIRED.
C  IF THE INPUT FILE (FILE1) DOES NOT EXIST, ANOTHER ONE WILL
C  BE PROMPTED FOR.
C  DATA IS READ IN AS FREE FORMAT.
C  THE END OF FILE IS USED TO SIGNAL THE END OF DATA.
C
C  INPUT VARIABLES:
C
C  NAME= CHARACTER*25 VECTOR CONTAINING THE NAMES OF THE DATA
C     PARAMETERS
C  VALUE= REAL*4 VECTORS CONTAINING THE VALUES OF THE DATA
C     PARAMETERS
C  FILE1= CHARACTER*30 VARIABLE CONTAINING THE INPUT DATA FILE
C     NAME
C
C  INTERNAL VARIABLES:
C
C  EX= LOGICAL*2 VARIABLE CONTAINING THE TEST VALUE OF WHETHER
C     OR NOT FILE1 EXISTS
C
C  OUTPUT:
C
C  INPUT ARRAYS ARE FILLED WITH VALUES AS THEY ARE READ IN
C
C  OUTPUT VARIABLES:
C
C NDAT= INTEGER*2 VARIABLE CONTAINING THE NUMBER OF DATA VALUES
C     READ IN
C
C  ERRORS:
C
C  THE FOLLOWING ERROR MESSAGES ARE PRINTED TO THE SCREEN AND
C  THEN THE PROGRAM IS TERMINATED IF CERTAIN CONDITIONS ARE
C  SATISFIED:
C
```

```
C  1. DATA FILE MUST CONTAIN THE SAME NUMBER OF VALUES AND
C    NAMES
C  2. ALLOWABLE NO. OF VARIABLES EXCEEDED
C
C   ROUTINES REQUIRED: NONE
C
C
      CHARACTER NAME(*)*25, FILE1*30
      REAL*4 VALUE(*)
      INTEGER*2 NDAT,MAXDAT
      PARAMETER(MAXDAT=24)
      LOGICAL*2 EX
C
   10 INQUIRE(FILE=FILE1,EXIST=EX)
      IF(.NOT.EX)THEN
        WRITE(*,*)' '
        WRITE(*,*)' ENTER THE DATA FILE NAME'
        READ(*,'(A)')FILE1
        GO TO 10
      ENDIF
      OPEN(5,FILE=FILE1)
C
      NDAT=1
   20 READ(5,'(A)',END=40)NAME(NDAT)
      READ(5,*,END=30)VALUE(NDAT)
      NDAT=NDAT+1
      GO TO 20
C
   30 WRITE(*,*)'** ERROR **'
      STOP
      'DATA FILE MUST CONTAIN THE SAME NO. OF VALUES AND NAMES'
   40 NDAT=NDAT-1
      IF(NDAT.GT.MAXDAT)
     + STOP ' ALLOWABLE NO. OF VARIABLES EXCEEDED'
      CLOSE(5)
C
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE RESULTS(VOB,VIB,VCHOP,VSPOIL,OROB,ORIB,H,ISW,ISEAM)
C
C     PURPOSE:
C
C     THIS SUBROUTINE IS USED TO WRITE THE CALCULATED RESULTS OF
C     EITHER 1SEAM.FOR OR 2SEAM.FOR TO THE SCREEN ON PAGE 1.
C     THERE ARE TWO ALTERNATIVES WHEN USING THIS SUBROUTINE:
C
C          (1) WRITE ALL VALUES AND LABELS TO THE SCREEN
C          (2) WRITE ONLY NEW VALUES TO THE SCREEN - THIS WOULD BE
C              USED IF THE LABELS ALREADY EXIST ON THE SCREEN FROM
C             A PREVIOUS CALL AND ONLY THE VALUES HAVE CHANGED
C
C     INPUT VARIABLES:
C
C     VOB,VIB,VCHOP= REAL*4 VARIABLES CONTAINING THE UNIT BANK
C        VOLUME OF THE O/B, I/B AND CHOP CUTS RESPECTIVELY
C     VSPOIL= REAL*4 VARIABLE CONTAINING THE UNIT LOOSE VOLUME OF
C         THE TOTAL SPOIL
C OROB,ORIB=REAL*4 VARIABLES CONTAINING THE REQUIRED REACHES OF
C         THE DRAGLINE WHEN IT IS OPERATING ON THE O/B AND I/B
C         BENCHES RESPECTIVELY
C     H= REAL*4 VARIABLE CONTAINING THE REQUIRED HEIGHT OF THE
C         DRAGLINE WHEN IT IS OPERATING ON THE I/B BENCH
C     ISW= LOGICAL*2 VARIABLE CONTAINING WHETHER OR NOT THE TEXT
C         NEEDS TO BE WRITTEN TO THE SCREEN
C     ISEAM= INTEGER*2 VARIABLE CONTAINING WHICH RESULTS PAGE
C        MUST BE WRITTEN WHERE 1=1SEAM AND 2=2SEAM.
C
C     OUTPUT:
C
C     ALL VALUES ARE WRITTEN TO THE SCREEN ON PAGE 1
C
C     OUTPUT VARIABLES:
C
C     VOB,VIB,VCHOP,VSPOIL,OROB,ORIB,H,ISEAM=  UNCHANGED
C     ISW= .FALSE.
C
C     SUBROUTINES REQUIRED:
C
C     LOCATE= A FORTRAN UTILITIES ROUTINE CALLED TO POSITION THE
C        CURSOR ON THE SELECTED SCREEN PAGE
C     PUTST= A FORTRAN UTILITIES ROUTINE CALLED TO WRITE A STRING
C        WITH ATTRIBUTE AT THE CURRENT CURSOR POSITION
C
C
      REAL*4 VCHOP, VSPOIL, OROB, ORIB, H, VIB, VOB
      CHARACTER*10 DUMFF, BLANK
      LOGICAL*2 ISW
      INTEGER*2 ISEAM
C
      DATA BLANK/'          '/
      IF(ISEAM.NE.1.AND.ISEAM.NE.2)RETURN
C
C--WRITE TEXT ON PAGE
C
      IF(ISW)THEN
      CALL LOCATE(1,0,18)
      CALL PUTST(1,7,38,'***  SUMMARY OF SEAM CALCULATIONS  ***')
      CALL LOCATE(1,3,5)
      CALL PUTST(1,7,37,'REQUIRED DIMENSIONS FOR NO REHANDLE :')
      CALL LOCATE(1,5,10)
      CALL PUTST(1,7,36,'1. O/B OPERATING RADIUS (m)        =')
      CALL LOCATE(1,11,5)
      CALL PUTST(1,7,14,'UNIT VOLUMES :')
      CALL LOCATE(1,13,10)
      CALL PUTST(1,7,19,'1. SPOIL (lcm)     =')
      CALL LOCATE(1,15,10)
      CALL PUTST(1,7,19,'2. CHOPCUT (bcm)  =')
      CALL LOCATE(1,17,10)
      CALL PUTST(1,7,19,'3. O/B CUT (bcm)  =')
      CALL LOCATE(1,22,5)
      CALL PUTST(1,7,57,
     + 'Select # for output:  0  NONE        3  PRINTER (1:500)')
      CALL LOCATE(1,23,5)
      CALL PUTST(1,7,58,
     + '                      1  CGA SCREEN   4  PRINTER (1:1000)')
      CALL LOCATE(1,24,5)
      CALL PUTST(1,7,58,
     + '                      2  EGA SCREEN   5  PLOTTER (1:1000)')
      IF(ISEAM.EQ.1)THEN
        CALL LOCATE(1,7,10)
        CALL PUTST(1,7,36,'2. SPOIL HEIGHT ABOVE O/B BENCH (m)=')
      ELSE
        CALL LOCATE(1,7,10)
        CALL PUTST(1,7,36,'2. I/B OPERATING RADIUS (m)        =')
        CALL LOCATE(1,9,10)
        CALL PUTST(1,7,36,'3. SPOIL HEIGHT ABOVE I/B BENCH (m)=')
        CALL LOCATE(1,19,10)
        CALL PUTST(1,7,19,'4. I/B CUT (bcm)  =')
      ENDIF
      ISW=.FALSE.
      ENDIF
C
C--WRITE VALUES IN HIGHLIGHTED CELLS
C
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')OROB
      CALL LOCATE(1,5,48)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')VSPOIL
      CALL LOCATE(1,13,31)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')VCHOP
      CALL LOCATE(1,15,31)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')VOB
      CALL LOCATE(1,17,31)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      IF(ISEAM.EQ.1)THEN
        DUMFF=BLANK
        WRITE(DUMFF,'(F10.2)')H
        CALL LOCATE(1,7,48)
        CALL PUTST(1,120,10,DUMFF)
      ELSE
        DUMFF=BLANK
        WRITE(DUMFF,'(F10.2)')ORIB
        CALL LOCATE(1,7,48)
        CALL PUTST(1,120,10,DUMFF)
        DUMFF=BLANK
        WRITE(DUMFF,'(F10.2)')H
        CALL LOCATE(1,9,48)
        CALL PUTST(1,120,10,DUMFF)
        DUMFF=BLANK
        WRITE(DUMFF,'(F10.2)')VIB
        CALL LOCATE(1,19,31)
        CALL PUTST(1,120,10,DUMFF)
      ENDIF
C
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE RESULTR(VOB,VCHOP,VSPOIL,OROB,H,ET,RVOL EVOL,ISW)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO WRITE THE CALCULATED RESULTS OF
C  1R.FOR TO THE SCREEN ON PAGE 1.  1R.FOR IS A PROGRAM WHICH
C  DETERMINES THE DRAGLINE VOLUMES WHICH OCCUR WHEN MINING A
C  SINGLE SEAM AND ALLOWING AN EXTENDED BENCH TO BE USED.
C  THERE ARE TWO ALTERNATIVES WHEN USING THIS SUBROUTINE:
C
C       (1) WRITE ALL VALUES AND LABELS TO THE SCREEN
C       (2) WRITE ONLY NEW VALUES TO THE SCREEN - THIS WOULD BE
C           USED IF THE LABELS ALREADY EXIST ON THE SCREEN FROM
C          A PREVIOUS CALL AND ONLY THE VALUES HAVE CHANGED
C
C  INPUT VARIABLES:
C
C  VOB,VCHOP= REAL*4 VARIABLES CONTAINING THE UNIT BANK VOLUME
C     OF THE O/B AND CHOP CUTS RESPECTIVELY
C  VSPOIL= REAL*4 VARIABLE CONTAINING THE UNIT LOOSE VOLUME OF
C       THE TOTAL SPOIL
C  OROB= REAL*4 VARIABLE CONTAINING THE REQUIRED REACH OF THE
C       THE DRAGLINE WHEN IT IS OPERATING ON THE O/B BENCH
C  H= REAL*4 VARIABLE CONTAINING THE REQUIRED HEIGHT OF THE
C       DRAGLINE WHEN IT IS OPERATING ON THE I/B BENCH
C  ISW= LOGICAL*2 VARIABLE CONTAINING WHETHER OR NOT THE TEXT
C       NEEDS TO BE WRITTEN TO THE SCREEN
C ET=REAL*4 VARIABLE CONTAINING THE WIDTH OF THE EXTENDED BENCH
C  RVOL= REAL*4 VARIABLE CONTAINING THE REHANDLE VOLUME
C  EVOL= REAL*4 VARIABLE CONTAINING THE EXTENDED BENCH VOLUME
C
C  OUTPUT:
C
C  ALL VALUES ARE WRITTEN TO THE SCREEN ON PAGE 1
C
C  OUTPUT VARIABLES:
C
C  VOB,VCHOP,VSPOIL,OROB,H,ET,RVOL,EVOL= UNCHANGED
C  ISW= .FALSE.
C
C  SUBROUTINES REQUIRED:
C
C  LOCATE= A FORTRAN UTILITIES ROUTINE CALLED TO POSITION THE
C     CURSOR ON THE SELECTED SCREEN PAGE
C  PUTST= A FORTRAN UTILITIES ROUTINE CALLED TO WRITE A STRING
C     WITH ATTRIBUTE AT THE CURRENT CURSOR POSITION
C
C
      REAL*4 VCHOP, VSPOIL, OROB, H, VOB
      REAL*4 ET,RVOL,EVOL
      CHARACTER*10 DUMFF, BLANK
      LOGICAL*2 ISW
C
      DATA BLANK/'          '/
C
C--WRITE TEXT ON PAGE
C
      IF(ISW)THEN
      CALL LOCATE(1,0,18)
      CALL PUTST(1,7,38,'***  SUMMARY OF SEAM CALCULATIONS  ***')
      CALL LOCATE(1,2,5)
      CALL PUTST(1,7,21,'REQUIRED DIMENSIONS :')
      CALL LOCATE(1,4,10)
      CALL PUTST(1,7,37,'1. O/B OPERATING RADIUS (m)         =')
      CALL LOCATE(1,6,10)
      CALL PUTST(1,7,37,'2. SPOIL HEIGHT ABOVE O/B BENCH (m) =')
      CALL LOCATE(1,8,10)
      CALL PUTST(1,7,37,'3. WIDTH OF EXTENDED BENCH (m)      =')
      CALL LOCATE(1,10,5)
      CALL PUTST(1,7,14,'UNIT VOLUMES :')
      CALL LOCATE(1,12,10)
      CALL PUTST(1,7,25,'1. SPOIL (lcm)          =')
      CALL LOCATE(1,14,10)
      CALL PUTST(1,7,25,'2. REHANDLE (lcm)       =')
      CALL LOCATE(1,16,10)
      CALL PUTST(1,7,25,'3. EXTENDED BENCH (lcm) =')
      CALL LOCATE(1,18,10)
      CALL PUTST(1,7,25,'4. CHOPCUT (bcm)        =')
      CALL LOCATE(1,20,10)
      CALL PUTST(1,7,25,'5. O/B CUT (bcm)        =')
      CALL LOCATE(1,22,5)
      CALL PUTST(1,7,57,
     +'Select # for output:   0  NONE         3  PRINTER (1:500)')
      CALL LOCATE(1,23,5)
      CALL PUTST(1,7,58,
     +'                       1  CGA SCREEN   4  PRINTER (1:1000)')
      CALL LOCATE(1,24,5)
      CALL PUTST(1,7,58,
     +'                       2  EGA SCREEN   5  PLOTTER (1:1000)')
      ISW=.FALSE.
      ENDIF
C
C--WRITE VALUES IN HIGHLIGHTED CELLS
C
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')OROB
      CALL LOCATE(1,4,49)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')H
      CALL LOCATE(1,6,49)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')ET
      CALL LOCATE(1,8,49)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')VSPOIL
      CALL LOCATE(1,12,37)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')RVOL
      CALL LOCATE(1,14,37)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')EVOL
      CALL LOCATE(1,16,37)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')VCHOP
      CALL LOCATE(1,18,37)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')VOB
      CALL LOCATE(1,20,37)
      CALL PUTST(1,120,10,DUMFF)
C
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE RESULTR2(VOB,VIB,VCHOP,VSPOIL,ISW)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO WRITE THE CALCULATED RESULTS OF
C  2R.FOR TO THE SCREEN ON PAGE 1.  2R.FOR IS A PROGRAM WHICH
C  DETERMINES THE DRAGLINE VOLUMES WHICH OCCUR WHEN MINING TWO
C  SEAMS AND ALLOWING AN EXTENDED BENCH TO BE USED.
C  THERE ARE TWO ALTERNATIVES WHEN USING THIS SUBROUTINE:
C
C        (1) WRITE ALL VALUES AND LABELS TO THE SCREEN
C        (2) WRITE ONLY NEW VALUES TO THE SCREEN - THIS WOULD BE
C            USED IF THE LABELS ALREADY EXIST ON THE SCREEN FROM
C           A PREVIOUS CALL AND ONLY THE VALUES HAVE CHANGED
C
C  INPUT VARIABLES:
C
C  VOB,VIB,VCHOP= REAL*4 VARIABLES CONTAINING THE UNIT BANK
C     VOLUME OF THE O/B,I/B AND CHOP CUTS RESPECTIVELY
C  VSPOIL= REAL*4 VARIABLE CONTAINING THE UNIT LOOSE VOLUME OF
C      THE TOTAL SPOIL
C  ISW= LOGICAL*2 VARIABLE CONTAINING WHETHER OR NOT THE TEXT
C      NEEDS TO BE WRITTEN TO THE SCREEN
C
C  OUTPUT:
C
C  ALL VALUES ARE WRITTEN TO THE SCREEN ON PAGE 1
C
C  OUTPUT VARIABLES:
C
C  VOB,VIB,VCHOP,VSPOIL= UNCHANGED
C  ISW= .FALSE.
C
C  SUBROUTINES REQUIRED:
C
C  LOCATE= A FORTRAN UTILITIES ROUTINE CALLED TO POSITION THE
C     CURSOR ON THE SELECTED SCREEN PAGE
C  PUTST= A FORTRAN UTILITIES ROUTINE CALLED TO WRITE A STRING
C     WITH ATTRIBUTE AT THE CURRENT CURSOR POSITION
C
C
      REAL*4 VCHOP, VSPOIL, VIB, VOB
      CHARACTER*10 DUMFF, BLANK
      LOGICAL*2 ISW
C
      DATA BLANK/'          '/
C
C--WRITE TEXT ON PAGE
C
```

```
      IF(ISW)THEN
        CALL LOCATE(1,0,18)
        CALL PUTST(1,7,38,'***  SUMMARY OF SEAM CALCULATIONS  ***')
        CALL LOCATE(1,3,5)
        CALL PUTST(1,7,14,'UNIT VOLUMES :')
        CALL LOCATE(1,5,10)
        CALL PUTST(1,7,19,'1. SPOIL (lcm)     =')
        CALL LOCATE(1,7,10)
        CALL PUTST(1,7,19,'2. CHOPCUT (bcm)  =')
        CALL LOCATE(1,9,10)
        CALL PUTST(1,7,19,'3. O/B CUT (bcm)  =')
        CALL LOCATE(1,11,10)
        CALL PUTST(1,7,19,'4. I/B CUT (bcm)  =')
        ISW=.FALSE.
      ENDIF
      CALL LOCATE(1,22,5)
      CALL PUTST(1,7,57,
     +'Select # for output:   0  RETURN TO INPUT SCREEN          ')
      CALL LOCATE(1,23,5)
      CALL PUTST(1,7,58,
     +'                        1  PLOT O/B SPOIL                  ')
      CALL LOCATE(1,24,5)
      CALL PUTST(1,7,58,
     +'                        2  PLOT O/B & I/B SPOIL            ')
C
C--WRITE VALUES IN HIGHLIGHTED CELLS
C
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')VSPOIL
      CALL LOCATE(1,5,31)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')VCHOP
      CALL LOCATE(1,7,31)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')VOB
      CALL LOCATE(1,9,31)
      CALL PUTST(1,120,10,DUMFF)
      DUMFF=BLANK
      WRITE(DUMFF,'(F10.2)')VIB
      CALL LOCATE(1,11,31)
      CALL PUTST(1,120,10,DUMFF)
C
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE PLOT1(X,Y,N,NUM,VALUE,NAME,H,OROB)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO PLOT THE RANGE DIAGRAM FOR A
C  SINGLE SEAM DRAGLINE CUT. THE MINIMUM HEIGHT AND REACH
C  REQUIREMENTS ARE DISPLAYED ON THE DRAWING.  THIS SUBROUTINE
C  USES SEVERAL ROUTINES FROM THE PLOT88 GRAPHICS SOFTWARE
C  LIBRARY TO DISPLAY THE RANGE DIAGRAMS.
C
C  INPUT VARIABLES:
C
C N=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF POINTS DEFINING
C       THE RANGE DIAGRAM
C  X,Y= REAL*4 VECTORS CONTAINING THE COORDINATES OF THE RANGE
C     DIAGRAM
C  NUM= INTEGER*2 VARIABLE CONTAINING THE NUMBER WHICH DEFINES
C       THE MODE OF OUTPUT WHERE:
C
C                 NUM=1  -->  CGA SCREEN
C                 NUM=2  -->  EGA SCREEN
C                 NUM=3  -->  PRINTER (1:500)
C                 NUM=4  -->  PRINTER (1:1000)
C                 NUM=5  -->  PLOTTER (1:1000)
C
C  NAME= CHARACTER*25 VECTOR CONTAINING THE NAMES OF THE DATA
C       PARAMETERS
C  VALUE= REAL*4 VECTORS CONTAINING THE VALUES OF THE DATA
C       PARAMETERS
C  OROB= REAL*4 VARIABLE CONTAINING THE REQUIRED REACH OF THE
C       DRAGLINE WHEN IT IS OPERATING ON THE O/B BENCH
C  H= REAL*4 VARIABLE CONTAINING THE REQUIRED HEIGHT OF THE
C       DRAGLINE WHEN IT IS OPERATING ON THE O/B BENCH
C
C  INTERNAL VARIABLES:
C
C  RECTX(4),RECTY(4)= REAL*4 VECTORS CONTAINING THE COORDINATES
C       OF THE RECTANGLE REPRESENTING THE DRAGLINE
C MAXX,MAXY=REAL*4 VARIABLES CONTAINING THE MAXIMUM COORDINATE
C       IN THE X AND Y DIRECTION RESPECTIVELY
C XSCALE,YSCALE= REAL*4 VARIABLES CONTAINING THE SCALE FACTORS
C       IN THE X AND Y DIRECTION RESPECTIVELY
C  PWIDTH= INTEGER*2 VARIABLE CONTAINING A NUMBER REPRESENTING
C       THE WIDTH OF THE PRINTER BEING USED.
C       IF PWIDTH=0 --> NARROW PRINTER CARRIAGE
C       IF PWIDTH=1 --> WIDE PRINTER CARRIAGE
C  HITE= REAL*4 VARIABLE CONTAINING THE HEIGHT THAT THE TEXT
C       WILL BE DRAWN AT.
C
```

```
C  OUTPUT VARIABLES:
C
C  X,Y,N,NUM,VALUE,NAME,H,OROB= UNCHANGED
C
C  SUBROUTINES REQUIRED:
C
C  PLOTS= A PLOT88 ROUTINE WHICH MUST BE CALLED TO INITIALIZE
C      THE PLOT88 SOFTWARE
C  FACTOR= A PLOT88 ROUTINE WHICH IS USED TO ENLARGE OR REDUCE
C      THE SIZE OF THE ENTIRE DRAWING.
C  WINDOW= A PLOT88 ROUTINE WHICH DEFINES THE WINDOW IN WHICH
C      THE DRAWING WILL BE MADE.
C  PLOT= A PLOT88 ROUTINE WHICH MOVES THE DRAWING UTENSIL FROM
C      THE CURRENT POSITION TO A NEW POSITION.  THE 'PEN' MAY
C      BE UP OR DOWN AS IT IS BEING MOVED.
C  LINE= A PLOT88 ROUTINE WHICH DRAWS A LINE THROUGH (X,Y) POINT
C      PAIRS
C  STFILL= A PLOT88 ROUTINE WHICH IS USED TO SELECT THE SHADING
C      PATTERN TO USE IN FUTURE CALLS TO THE FILL ROUTINE
C  FILL= A PLOT88 ROUTINE WHICH PROVIDES SHADING OF USER
C      SPECIFIED POLYGONAL AREAS
C  SYMBOL= A PLOT88 ROUTINE WHICH IS USED TO DRAW CHARACTER
C      STRINGS AND SPECIAL CENTERED SYMBOLS.
C  NUMBER= A PLOT88 ROUTINE WHICH IS USED TO CONVERT A SINGLE
C      PRECISION REAL NUMBER TO ITS DECIMAL EQUIVALENT AS A
C      CHARACTER STRING AND TO DRAW THE STRING USING SYMBOL.
C
C
   REAL*4 X(*),Y(*),FACT,MAXX,MAXY,VALUE(*),RECTX(4),RECTY(4)
   REAL*4 XSCALE,YSCALE,X1,X2,H,OROB,YVAL,HITE,Y1,Y2
   INTEGER*2 IOPORT,MODEL,N,NUM,PMODEL,PIOPRT,I,PWIDTH
   LOGICAL*2 EX
   CHARACTER NAME(*)*25
C
   FACT=1.0
   HITE=.15
   SELECT CASE(NUM)
     CASE(1)
C
C      SCALE TO FIT CGA SCREEN
C
       MAXX=AMAX1(X(18),X(19))
       MAXY=AMAX1(Y(5),Y(17))
       IF(.444*MAXX.GT.MAXY)THEN
         X(22)=MAXX/9.
       ELSE
         X(22)=MAXY/4.
       ENDIF
       Y(22)=X(22)
       IOPORT = 99
```

```
        MODEL  = 99
        YVAL=7.1
        FACT=0.8
        HITE=.2
      CASE(2)
C
C       SCALE TO FIT EGA SCREEN
C
        MAXX=AMAX1(X(18),X(19))
        MAXY=AMAX1(Y(5),Y(17))
        IF(.444*MAXX.GT.MAXY)THEN
          X(22)=MAXX/9.
        ELSE
          X(22)=MAXY/4.
        ENDIF
        Y(22)=X(22)
        IOPORT = 97
        MODEL  = 97
        YVAL=7.1
      CASE(3)

        PRINTER SETTING (1:500)
        NARROW PRINTER (80 COL)
C
        IOPORT = PIOPRT
        MODEL  = PMODEL
        X(22) = 12.70
        Y(22) = 12.70
        YVAL=7.8
      CASE(4)
C
C       PRINTER SETTING (1:1000)
C       NARROW PRINTER (80 COL)
C
        IOPORT = PIOPRT
        MODEL  = PMODEL
        X(22) = 25.40
        Y(22) = 25.40
        YVAL=7.8
      CASE(5)
C
C       HP7475A PLOTTER SETTING (1:1000)
C       ASSUMING 8.5"x11" PAPER USED
C
        IOPORT = 9602
        MODEL  = 30
        X(22) = 25.40
        Y(22) = 25.40
        YVAL=7.8
      CASE DEFAULT
```

```
        RETURN
      END SELECT
      XSCALE=X(22)
      YSCALE=Y(22)
C
      CALL PLOTS(0,IOPORT,MODEL)
      CALL FACTOR(FACT)
C
C--INDICATE ON THE SCREEN THAT THE DRAWING IS BEING GENERATED
C
      WRITE(*,*)'DRAWING IS BEING GENERATED ...'
      WRITE(*,*)' '
C
C--ALLOW PRINTER TO PRINT ON MORE THAN ONE PAGE AND SET PAGE
C  HEIGHT
C
      IF(NUM.EQ.3.OR.NUM.EQ.4) THEN
        IF(PWIDTH.EQ.1)YVAL=YVAL+5.
        CALL WINDOW(0.,0.,22.,YVAL+.5)
      ENDIF
C
C   MOVE TO (0.0,0.0), NO LINE IS DRAWN, AND DEFINE A NEW ORIGIN
C
      CALL PLOT(0.0,0.0,-3)
      CALL LINE(X,Y,N,1,0,2)
C
C--DRAW RECTANGLE TO REPRESENT DRAGLINE      VALUE(10) = P
C                                           VALUE(11) = D
C
      RECTX(1)=(X(10)-VALUE(11)*(VALUE(10)-0.5))/XSCALE
      RECTX(2)=RECTX(1)
      RECTX(3)=RECTX(2)-(VALUE(11)/XSCALE)
      RECTX(4)=RECTX(3)
      RECTY(1)=Y(10)/YSCALE
      RECTY(2)=RECTY(1)+(5./YSCALE)
      RECTY(3)=RECTY(2)
      RECTY(4)=RECTY(1)
      CALL STFILL(1)
      CALL FILL(RECTX,RECTY,4)
C
C--WRITE TITLE ON SCREEN
C
      CALL SYMBOL(3.0,YVAL,HITE,'SINGLE SEAM SIDECASTING',0.,23)
C
C--WRITE INPUT VALUES ON SCREEN FOR COLUMN #1
C
      DO 100 I=1,6
      CALL NUMBER(0.,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,VALUE(I),0.,2)
100   CALL SYMBOL(1.1,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,NAME(I),0.,25)
C
```

```
C--WRITE INPUT VALUES ON SCREEN FOR COLUMN #2
C
     DO 200 I=1,5
     CALL NUMBER(5.0,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,VALUE(I+6),0.,2)
200  CALL SYMBOL(6.1,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,NAME(I+6),0.,25)
C
C--WRITE SCALE VALUE ON OUTPUT
C
     SELECT CASE(NUM)
       CASE(3)
         CALL SYMBOL((X(19)+25.)/XSCALE,0.,HITE,'1:500 ',0.,6)
       CASE(4,5)
         CALL SYMBOL((X(19)+25.)/XSCALE,0.,HITE,'1:1000',0.,6)
     END SELECT
C
C--DRAW OPERATING RADIUS VALUE ON RANGE DIAGRAM
C
     X1=(X(10)-VALUE(10)*VALUE(11))/XSCALE
     CALL PLOT(X1,(Y(17)+5.)/YSCALE,3)
     CALL PLOT(X1,(Y(17)+10.)/YSCALE,2)
     CALL PLOT(X1,(Y(17)+7.5)/YSCALE,3)
     X2=X(17)/XSCALE
     CALL PLOT(X2,(Y(17)+7.5)/YSCALE,2)
     CALL PLOT(X2,(Y(17)+5.)/YSCALE,3)
     CALL PLOT(X2,(Y(17)+10.)/YSCALE,2)
     CALL NUMBER(X1-.5+(X2-X1)*.5,(Y(17)+8.5)/YSCALE,HITE,OROB,0.,2)
     CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
C
C--DRAW HEIGHT VALUE ON RANGE DIAGRAM
C
     Y1=Y(10)/YSCALE
     CALL PLOT((X(10)+5.)/XSCALE,Y1,3)
     CALL PLOT((X(10)+10.)/XSCALE,Y1,2)
     CALL PLOT((X(10)+7.5)/XSCALE,Y1,3)
     Y2=Y(17)/YSCALE
     CALL PLOT((X(10)+7.5)/XSCALE,Y2,2)
     CALL PLOT((X(10)+5.)/XSCALE,Y2,3)
     CALL PLOT((X(10)+10.)/XSCALE,Y2,2)
     CALL NUMBER((X(10)+8.5)/XSCALE,Y1-.07+(Y2-Y1)*.5,HITE,H,0.,2)
     CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
C
C--LABEL DIFFERENT PIT WIDTHS
C
     Y1=10./YSCALE
     Y2=5./YSCALE
     CALL PLOT(X(1)/XSCALE,Y1,3)
     CALL PLOT(X(1)/XSCALE,Y2,2)
     CALL PLOT(X(14)/XSCALE,Y1,3)
     CALL PLOT(X(14)/XSCALE,Y2,2)
     CALL PLOT(X(19)/XSCALE,Y1,3)
```

```
      CALL PLOT(X(19)/XSCALE,Y2,2)
      CALL PLOT(X(19)/XSCALE,7.5/YSCALE,3)
      CALL PLOT(X(1)/XSCALE,7.5/YSCALE,2)
      CALL NUMBER((X(1)+.5*(X(14)-X(1)))/XSCALE-.4,0.,HITE,
     +                    VALUE(1),0.,1)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
      CALL NUMBER((X(14)+.5*(X(19)-X(14)))/XSCALE-.4,0.,HITE,
     +                    VALUE(2),0.,1)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
      CALL PLOT(0.0,0.0,999)
      RETURN
C
      ENTRY INITP1()
C
C--IF A PRINT.DAT FILE IS PRESENT, THE PARAMETERS WILL BE READ
C  FROM THAT FILE, OTHERWISE DEFAULT PARAMETERS WILL BE USED
C
      INQUIRE(FILE='PRINT.DAT',EXIST=EX)
      IF(EX)THEN
        OPEN(5,FILE='PRINT.DAT')
        READ(5,*)PIOPRT
        READ(5,*)PMODEL
        READ(5,*)PWIDTH
        CLOSE(5)
      ELSE
C
C--IF WIDE CARRIAGE IS DESIRED FOR OUTPUT, SPECIFY IN PRINT.DAT
C  MODEL VALUE FOR LQ500 IS 42
C
        PIOPRT=0
        PMODEL=42
        PWIDTH=0.
      ENDIF
C
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE PLOT2(X,Y,N,NUM,VALUE,NAME,H,OROB,ORIB)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO PLOT THE RANGE DIAGRAM FOR A
C  TWO SEAM DRAGLINE CUT. THE MINIMUM HEIGHT AND REACH
C  REQUIREMENTS ARE DISPLAYED ON THE DRAWING.  THIS SUBROUTINE
C  USES SEVERAL ROUTINES FROM THE PLOT88 GRAPHICS SOFTWARE
C  LIBRARY TO DISPLAY THE RANGE DIAGRAMS.
C
C  INPUT VARIABLES:
C
C N= INTEGER*2 VARIABLE CONTAINING THE NUMBER OF POINTS DEFINING
C      THE RANGE DIAGRAM
C  X,Y= REAL*4 VECTORS CONTAINING THE COORDINATES OF THE RANGE
C       DIAGRAM
C  NUM= INTEGER*2 VARIABLE CONTAINING THE NUMBER WHICH DEFINES
C       THE MODE OF OUTPUT WHERE:
C
C                NUM=1  -->  CGA SCREEN
C                NUM=2  -->  EGA SCREEN
C                NUM=3  -->  PRINTER (1:500)
C                NUM=4  -->  PRINTER (1:1000)
C                NUM=5  -->  PLOTTER (1:1000)
C
C  NAME= CHARACTER*25 VECTOR CONTAINING THE NAMES OF THE DATA
C       PARAMETERS
C  VALUE= REAL*4 VECTORS CONTAINING THE VALUES OF THE DATA
C       PARAMETERS
C  OROB,ORIB= REAL*4 VARIABLES CONTAINING THE REQUIRED REACHES
C       OF A DRAGLINE WHEN IT IS OPERATING ON THE O/B
C       AND I/B BENCHES RESPECTIVELY
C  H= REAL*4 VARIABLE CONTAINING THE REQUIRED HEIGHT OF THE
C       DRAGLINE WHEN IT IS OPERATING ON THE I/B BENCH
C
C  INTERNAL VARIABLES:
C
C  RECTX(4),RECTY(4)= REAL*4 VECTORS CONTAINING THE COORDINATES
C       OF THE RECTANGLE REPRESENTING THE DRAGLINE
C MAXX,MAXY= REAL*4 VARIABLES CONTAINING THE MAXIMUM COORDINATE
C       IN THE X AND Y DIRECTION RESPECTIVELY
C XSCALE,YSCALE= REAL*4 VARIABLES CONTAINING THE SCALE FACTORS
C       IN THE X AND Y DIRECTION RESPECTIVELY
C  PWIDTH= INTEGER*2 VARIABLE CONTAINING A NUMBER REPRESENTING
C       THE WIDTH OF THE PRINTER BEING USED.
C       IF PWIDTH=0 --> NARROW PRINTER CARRIAGE
C       IF PWIDTH=1 --> WIDE PRINTER CARRIAGE
C  HITE= REAL*4 VARIABLE CONTAINING THE HEIGHT THAT THE TEXT
C       WILL BE DRAWN AT.
```

```
C
C  OUTPUT VARIABLES:
C
C  X,Y,N,NUM,VALUE,NAME,H,OROB,ORIB=  UNCHANGED
C
C  SUBROUTINES REQUIRED:
C
C  PLOTS= A PLOT88 ROUTINE WHICH MUST BE CALLED TO INITIALIZE
C       THE PLOT88 SOFTWARE
C  FACTOR= A PLOT88 ROUTINE WHICH IS USED TO ENLARGE OR REDUCE
C       THE SIZE OF THE ENTIRE DRAWING.
C  WINDOW= A PLOT88 ROUTINE WHICH DEFINES THE WINDOW IN WHICH
C       THE DRAWING WILL BE MADE.
C  PLOT= A PLOT88 ROUTINE WHICH MOVES THE DRAWING UTENSIL FROM
C       THE CURRENT POSITION TO A NEW POSITION.  THE 'PEN' MAY
C       BE UP OR DOWN AS IT IS BEING MOVED.
C  LINE= A PLOT88 ROUTINE WHICH DRAWS A LINE THROUGH (X,Y) POINT
C       PAIRS
C  STFILL= A PLOT88 ROUTINE WHICH IS USED TO SELECT THE SHADING
C       PATTERN TO USE IN FUTURE CALLS TO THE FILL ROUTINE
C  FILL= A PLOT88 ROUTINE WHICH PROVIDES SHADING OF USER
C       SPECIFIED POLYGONAL AREAS
C  SYMBOL= A PLOT88 ROUTINE WHICH IS USED TO DRAW CHARACTER
C       STRINGS AND SPECIAL CENTERED SYMBOLS.
C  NUMBER= A PLOT88 ROUTINE WHICH IS USED TO CONVERT A SINGLE
C       PRECISION REAL NUMBER TO ITS DECIMAL EQUIVALENT AS A
C       CHARACTER STRING AND TO DRAW THE STRING USING SYMBOL.
C
C
      REAL*4 X(*),Y(*),FACT,MAXX,MAXY,VALUE(*),RECTX(4),RECTY(4)
      REAL*4 XSCALE,YSCALE,X1,X2,H,OROB,YVAL,HITE,Y1,Y2
      INTEGER*2 IOPORT,MODEL,N,NUM,PMODEL,PIOPRT,I,PWIDTH
      LOGICAL*2 EX
      CHARACTER NAME(*)*25
C
      FACT=1.0
      HITE=.15
      SELECT CASE(NUM)
        CASE(1)
C
C         SCALE TO FIT CGA SCREEN
C
          MAXX=AMAX1(X(32),X(33))
          MAXY=AMAX1(Y(9),Y(31))
          IF(.4*MAXX.GT.MAXY)THEN
            X(36)=MAXX/9.
          ELSE
            X(36)=MAXY/3.6
          ENDIF
          Y(36)=X(36)
        IOPORT = 99
        MODEL  = 99
        YVAL=7.1
        FACT=0.8
        HITE=.2
      CASE(2)
C
C       SCALE TO FIT EGA SCREEN
C
        MAXX=AMAX1(X(32),X(33))
        MAXY=AMAX1(Y(9),Y(31))
        IF(.444*MAXX.GT.MAXY)THEN
          X(36)=MAXX/9.
        ELSE
          X(36)=MAXY/4.
        ENDIF
        Y(36)=X(36)
        IOPORT = 97
        MODEL  = 97
        YVAL=7.1
      CASE(3)
C
C       PRINTER SETTING (1:500)
C       NARROW PRINTER (80 COL)
C
        IOPORT = PIOPRT
        MODEL  = PMODEL
        X(36) = 12.70
        Y(36) = 12.70
        YVAL=7.8
      CASE(4)
C
C       PRINTER SETTING (1:1000)
C       NARROW PRINTER (80 COL)
C
        IOPORT = PIOPRT
        MODEL  = PMODEL
        X(36) = 25.40
        Y(36) = 25.40
        YVAL=7.8
      CASE(5)
C
C       HP7475A PLOTTER SETTING (1:1000)
C       ASSUMING 8.5"x11" PAPER USED
C
        IOPORT = 9602
        MODEL  = 30
        X(36) = 25.40
        Y(36) = 25.40
        YVAL=7.8
        CASE DEFAULT
          RETURN
      END SELECT
      XSCALE=X(36)
      YSCALE=Y(36)
C
      CALL PLOTS(0,IOPORT,MODEL)
      CALL FACTOR(FACT)
C
C--INDICATE ON THE SCREEN THAT THE DRAWING IS BEING GENERATED
C
      WRITE(*,*)'DRAWING IS BEING GENERATED ...'
      WRITE(*,*)' '
C
C--ALLOW PRINTER TO PRINT ON MORE THAN ONE PAGE AND SET PAGE
C  HEIGHT
C
      IF(NUM.EQ.3.OR.NUM.EQ.4) THEN
        IF(PWIDTH.EQ.1)YVAL=YVAL+5.
        CALL WINDOW(0.,0.,22.,YVAL+.5)
      ENDIF
C
C     MOVE TO (0.0,0.0), NO LINE IS DRAWN, AND DEFINE A NEW ORIGIN
C
      CALL PLOT(0.0,0.0,-3)
      CALL LINE(X,Y,N,1,0,2)
C
C--DRAW RECTANGLE TO REPRESENT DRAGLINE ON O/B    VALUE(16) = PO
C                                                 VALUE(18) = D
C
      RECTX(1)=(X(14)-VALUE(18)*(VALUE(16)-0.5))/XSCALE
      RECTX(2)=RECTX(1)
      RECTX(3)=RECTX(2)-(VALUE(18)/XSCALE)
      RECTX(4)=RECTX(3)
      RECTY(1)=Y(14)/YSCALE
      RECTY(2)=RECTY(1)+(5./YSCALE)
      RECTY(3)=RECTY(2)
      RECTY(4)=RECTY(1)
      CALL STFILL(1)
      CALL FILL(RECTX,RECTY,4)
C
C--DRAW RECTANGLE TO REPRESENT DRAGLINE ON I/B   VALUE(17) = PI
C                                                VALUE(18) = D
C
      RECTX(1)=(X(21)-VALUE(18)*(VALUE(17)-0.5))/XSCALE
      RECTX(2)=RECTX(1)
      RECTX(3)=RECTX(2)-(VALUE(18)/XSCALE)
      RECTX(4)=RECTX(3)
      RECTY(1)=Y(21)/YSCALE
      RECTY(2)=RECTY(1)+(5./YSCALE)
      RECTY(3)=RECTY(2)
      RECTY(4)=RECTY(1)
      CALL FILL(RECTX,RECTY,4)
C
C--WRITE TITLE ON SCREEN
C
      CALL SYMBOL(3.1,YVAL,HITE,'TWO SEAM SIDECASTING',0.,20)
C
C--WRITE INPUT VALUES ON SCREEN FOR COLUMN #1
C
      DO 100 I=1,9
      CALL NUMBER(0.,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,VALUE(I),0.,2)
100   CALL SYMBOL(1.1,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,NAME(I),0.,25)
C
C--WRITE INPUT VALUES ON SCREEN FOR COLUMN #2
C
      DO 200 I=1,9
      CALL NUMBER
     + (5.0,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,VALUE(I+9),0.,2)
200   CALL SYMBOL(6.1,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,NAME(I+9),0.,25)
C
C--WRITE SCALE VALUE ON OUTPUT
C
      SELECT CASE(NUM)
        CASE(3)
          CALL SYMBOL((X(33)+25.)/XSCALE,0.,HITE,'1:500 ',0.,6)
        CASE(4,5)
          CALL SYMBOL((X(33)+25.)/XSCALE,0.,HITE,'1:1000',0.,6)
      END SELECT
C
C--DRAW O/B OPERATING RADIUS VALUE ON RANGE DIAGRAM
C
      X1=(X(14)-VALUE(18)*VALUE(16))/XSCALE
      CALL PLOT(X1,(Y(31)+15.)/YSCALE,3)
      CALL PLOT(X1,(Y(31)+20.)/YSCALE,2)
      CALL PLOT(X1,(Y(31)+17.5)/YSCALE,3)
      X2=X(28)/XSCALE
      CALL PLOT(X2,(Y(31)+17.5)/YSCALE,2)
      CALL PLOT(X2,(Y(31)+15.)/YSCALE,3)
      CALL PLOT(X2,(Y(31)+20.)/YSCALE,2)
      CALL NUMBER(X1-.5+(X2-X1)*.5,(Y(31)+18.5)/YSCALE,HITE,OROB,0.,2)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
C
C--DRAW I/B OPERATING RADIUS VALUE ON RANGE DIAGRAM
C
      X1=(X(21)-VALUE(18)*VALUE(17))/XSCALE
      CALL PLOT(X1,(Y(31)+5.)/YSCALE,3)
      CALL PLOT(X1,(Y(31)+10.)/YSCALE,2)
      CALL PLOT(X1,(Y(31)+7.5)/YSCALE,3)
      X2=X(31)/XSCALE
      CALL PLOT(X2,(Y(31)+7.5)/YSCALE,2)
      CALL PLOT(X2,(Y(31)+5.)/YSCALE,3)
      CALL PLOT(X2,(Y(31)+10.)/YSCALE,2)
      CALL NUMBER(X1-.5+(X2-X1)*.5,(Y(31)+8.5)/YSCALE,HITE,ORIB,0.,2)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
C
C--DRAW HEIGHT VALUE ON RANGE DIAGRAM
C
      Y1=Y(21)/YSCALE
      CALL PLOT((X(21)+5.)/XSCALE,Y1,3)
      CALL PLOT((X(21)+10.)/XSCALE,Y1,2)
      CALL PLOT((X(21)+7.5)/XSCALE,Y1,3)
      Y2=Y(31)/YSCALE
      CALL PLOT((X(21)+7.5)/XSCALE,Y2,2)
      CALL PLOT((X(21)+5.)/XSCALE,Y2,3)
      CALL PLOT((X(21)+10.)/XSCALE,Y2,2)
      CALL NUMBER((X(21)+8.5)/XSCALE,Y1-.07+(Y2-Y1)*.5,HITE,H,0.,2)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
C
C--LABEL DIFFERENT PIT WIDTHS
C
      Y1=10./YSCALE
      Y2=5./YSCALE
      CALL PLOT(X(1)/XSCALE,Y1,3)
      CALL PLOT(X(1)/XSCALE,Y2,2)
      CALL PLOT(X(25)/XSCALE,Y1,3)
      CALL PLOT(X(25)/XSCALE,Y2,2)
      CALL PLOT(X(33)/XSCALE,Y1,3)
      CALL PLOT(X(33)/XSCALE,Y2,2)
      CALL PLOT(X(33)/XSCALE,7.5/YSCALE,3)
      CALL PLOT(X(1)/XSCALE,7.5/YSCALE,2)
      CALL NUMBER((X(1)+.5*(X(25)-X(1)))/XSCALE-.4,0.,HITE,
     +                   VALUE(2),0.,1)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
      CALL NUMBER((X(25)+.5*(X(33)-X(25)))/XSCALE-.4,0.,HITE,
     +                   VALUE(3),0.,1)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
C
      CALL PLOT(0.0,0.0,999)
      RETURN
C
      ENTRY INITP20
C
C--IF A PRINT.DAT FILE IS PRESENT, THE PARAMETERS WILL BE READ
C  FROM THAT FILE, OTHERWISE DEFAULT PARAMETERS WILL BE USED
C
      INQUIRE(FILE='PRINT.DAT',EXIST=EX)
      IF(EX)THEN
        OPEN(5,FILE='PRINT.DAT')
        READ(5,*)PIOPRT
        READ(5,*)PMODEL
        READ(5,*)PWIDTH
        CLOSE(5)
      ELSE
C
C--IF WIDE CARRIAGE IS DESIRED FOR OUTPUT, SPECIFY IN PRINT.DAT
C  MODEL VALUE FOR LQ500 IS 42
        PIOPRT=0
        PMODEL=42
        PWIDTH=0.
      ENDIF
      RETURN
        [illegible]
```

```
$STORAGE:2
   SUBROUTINE PLOTR1(X,Y,N,NUM,VALUE,NAME,H,OROB,ET,EB,E1,E2,TANT)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO PLOT THE RANGE DIAGRAM FOR A
C  SINGLE SEAM DRAGLINE CUT WITH EXTENDED BENCHING ALLOWED.
C  THE HEIGHT AND REACH REQUIREMENTS ARE DISPLAYED ON THE
C  DRAWING.  THIS SUBROUTINE USES SEVERAL ROUTINES FROM THE
C  PLOT88 GRAPHICS SOFTWARE LIBRARY TO DISPLAY THE RANGE
C  DIAGRAMS.
C
C  INPUT VARIABLES:
C
C N=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF POINTS DEFINING
C      THE RANGE DIAGRAM
C  X,Y= REAL*4 VECTORS CONTAINING THE COORDINATES OF THE RANGE
C      DIAGRAM
C  NUM= INTEGER*2 VARIABLE CONTAINING THE NUMBER WHICH DEFINES
C      THE MODE OF OUTPUT WHERE:
C
C                 NUM=1 --> CGA SCREEN
C                 NUM=2 --> EGA SCREEN
C                 NUM=3 --> PRINTER (1:500)
C                 NUM=4 --> PRINTER (1:1000)
C                 NUM=5 --> PLOTTER (1:1000)
C
C  NAME= CHARACTER*25 VECTOR CONTAINING THE NAMES OF THE DATA
C      PARAMETERS
C  VALUE= REAL*4 VECTORS CONTAINING THE VALUES OF THE DATA
C      PARAMETERS
C  OROB= REAL*4 VARIABLE CONTAINING THE REQUIRED REACH OF THE
C      DRAGLINE WHEN IT IS OPERATING ON THE O/B BENCH
C  H= REAL*4 VARIABLE CONTAINING THE REQUIRED HEIGHT OF THE
C      DRAGLINE WHEN IT IS OPERATING ON THE O/B BENCH
C  ET= REAL*4 VARIABLE CONTAINING THE EXTENDED BENCH WIDTH
C  E1= REAL*4 VARIABLE CONTAINING THE DISTANCE GAINED BY
C      ASSUMING THE VERTICAL COAL SEAM INSTEAD OF THE HIGHWALL
C      ANGLE
C  EB= REAL*4 VARIABLE CONTAINING THE MAXIMUM EXTENDED BENCH
C      MATERIAL WIDTH ON THE PIT FLOOR LESS E1
C  E2= REAL*4 VARIABLE CONTAINING THE DISTANCE PAST THE SPOIL
C      PIT WIDTH THAT THE EXTENDED BENCH MATERIAL COULD EXTEND
C  TANT= REAL*4 VARIABLE CONTAINING THE TANGENT OF THE ANGLE
C      OF REPOSE OF THE SPOIL PILE
C
C  INTERNAL VARIABLES:
C
C  RECTX(4),RECTY(4)= REAL*4 VECTORS CONTAINING THE COORDINATES
C      OF THE RECTANGLE REPRESENTING THE DRAGLINE
```

```
C MAXX,MAXY= REAL*4 VARIABLES CONTAINING THE MAXIMUM COORDINATE
C      IN THE X AND Y DIRECTION RESPECTIVELY
C XSCALE,YSCALE= REAL*4 VARIABLES CONTAINING THE SCALE FACTORS
C      IN THE X AND Y DIRECTION RESPECTIVELY
C  PWIDTH= INTEGER*2 VARIABLE CONTAINING A NUMBER REPRESENTING
C      THE WIDTH OF THE PRINTER BEING USED.
C      IF PWIDTH=0 --> NARROW PRINTER CARRIAGE
C      IF PWIDTH=1 --> WIDE PRINTER CARRIAGE
C  HITE= REAL*4 VARIABLE CONTAINING THE HEIGHT THAT THE TEXT
C      WILL BE DRAWN AT.
C  XE,YE= REAL*4 VECTORS OF LENGTH=5 CONTAINING THE (X,Y) POINT
C      PAIRS DEFINING THE REHANDLE AREA
C
C  OUTPUT VARIABLES:
C
C  X,Y,N,NUM,VALUE,NAME,H,OROB,ET,EB,E1,E2,TANT= UNCHANGED
C
C  SUBROUTINES REQUIRED:
C
C  PLOTS= A PLOT88 ROUTINE WHICH MUST BE CALLED TO INITIALIZE
C      THE PLOT88 SOFTWARE
C  FACTOR= A PLOT88 ROUTINE WHICH IS USED TO ENLARGE OR REDUCE
C      THE SIZE OF THE ENTIRE DRAWING.
C  WINDOW= A PLOT88 ROUTINE WHICH DEFINES THE WINDOW IN WHICH
C      THE DRAWING WILL BE MADE.
C  PLOT= A PLOT88 ROUTINE WHICH MOVES THE DRAWING UTENSIL FROM
C      THE CURRENT POSITION TO A NEW POSITION. THE 'PEN' MAY
C      BE UP OR DOWN AS IT IS BEING MOVED.
C  LINE= A PLOT88 ROUTINE WHICH DRAWS A LINE THROUGH (X,Y) POINT
C      PAIRS
C  STFILL= A PLOT88 ROUTINE WHICH IS USED TO SELECT THE SHADING
C      PATTERN TO USE IN FUTURE CALLS TO THE FILL ROUTINE
C  FILL= A PLOT88 ROUTINE WHICH PROVIDES SHADING OF USER
C      SPECIFIED POLYGONAL AREAS
C  SYMBOL= A PLOT88 ROUTINE WHICH IS USED TO DRAW CHARACTER
C      STRINGS AND SPECIAL CENTERED SYMBOLS.
C  NUMBER= A PLOT88 ROUTINE WHICH IS USED TO CONVERT A SINGLE
C      PRECISION REAL NUMBER TO ITS DECIMAL EQUIVALENT AS A
C      CHARACTER STRING AND TO DRAW THE STRING USING SYMBOL.
C
C
      REAL*4 X(*),Y(*),FACT,MAXX,MAXY,VALUE(*),RECTX(4),RECTY(4)
      REAL*4 XSCALE,YSCALE,X1,X2,H,OROB,YVAL,HITE,Y1,Y2
      REAL*4 ET,EB,E1,E2,TANT,XE(5),YE(5)
      INTEGER*2 IOPORT,MODEL,N,NUM,PMODEL,PIOPRT,I,PWIDTH
      LOGICAL*2 EX
      CHARACTER NAME(*)*25
C
      FACT=1.0
      HITE=.15
      SELECT CASE(NUM)
        CASE(1)
C
C         SCALE TO FIT CGA SCREEN
C
          MAXX=AMAX1(X(18),X(19))
          MAXY=AMAX1(Y(5),Y(17))
          IF(.444*MAXX.GT.MAXY)THEN
            X(22)=MAXX/9.
          ELSE
            X(22)=MAXY/4.
          ENDIF
          Y(22)=X(22)
          IOPORT = 99
          MODEL  = 99
          YVAL=7.1
          FACT=0.8
          HITE=.2
        CASE(2)
C
C         SCALE TO FIT EGA SCREEN
C
          MAXX=AMAX1(X(18),X(19))
          MAXY=AMAX1(Y(5),Y(17))
          IF(.444*MAXX.GT.MAXY)THEN
            X(22)=MAXX/9.
          ELSE
            X(22)=MAXY/4.
          ENDIF
          Y(22)=X(22)
          IOPORT = 97
          MODEL  = 97
          YVAL=7.1
        CASE(3)
C
C         PRINTER SETTING (1:500)
C         NARROW PRINTER (80 COL)
C
          IOPORT = PIOPRT
          MODEL  = PMODEL
          X(22) = 12.70
          Y(22) = 12.70
          YVAL=7.8
        CASE(4)
C
C         PRINTER SETTING (1:1000)
C         NARROW PRINTER (80 COL)
C
          IOPORT = PIOPRT
          MODEL  = PMODEL
```

```
         X(22) = 25.40
         Y(22) = 25.40
         YVAL=7.8
        CASE(5)
C
C        HP7475A PLOTTER SETTING (1:1000)
C        ASSUMING 8.5"x11" PAPER USED
C
         IOPORT = 9602
         MODEL  = 30
         X(22) = 25.40
         Y(22) = 25.40
         YVAL=7.8
        CASE DEFAULT
         RETURN
      END SELECT
      XSCALE=X(22)
      YSCALE=Y(22)
C
      CALL PLOTS(0,IOPORT,MODEL)
      CALL FACTOR(FACT)
C
C--INDICATE ON THE SCREEN THAT THE DRAWING IS BEING GENERATED
C
      WRITE(*,*)'DRAWING IS BEING GENERATED ...'
      WRITE(*,*)' '
C
C--ALLOW PRINTER TO PRINT ON MORE THAN ONE PAGE AND SET PAGE
C  HEIGHT
C
      IF(NUM.EQ.3.OR.NUM.EQ.4) THEN
        IF(PWIDTH.EQ.1)YVAL=YVAL+5.
        CALL WINDOW(0.,0.,22.,YVAL+.5)
      ENDIF
C
C     MOVE TO (0.0,0.0), NO LINE IS DRAWN, AND DEFINE A NEW ORIGIN
C
      CALL PLOT(0.0,0.0,-3)
      CALL LINE(X,Y,N,1,0,2)
C
C--CHECK IF EXTENDED BENCH IS REQUIRED
C
      IF(ET.GT.0.)THEN
C
C--FILL IN REHANDLE AREA
C
        XE(1)=X(10)/XSCALE
        YE(1)=Y(10)/YSCALE
        XE(2)=X(11)/XSCALE
        YE(2)=Y(11)/YSCALE
        XE(3)=X(16)/XSCALE
        YE(3)=Y(16)/YSCALE
        XE(4)=(X(16)+0.5*(E1+EB))/XSCALE
        YE(4)=(Y(16)+0.5*(E1+EB)*TANT)/YSCALE
        XE(5)=XE(1)+ET/XSCALE
        YE(5)=YE(1)
        CALL STFILL(3)
        CALL FILL(XE,YE,5)
C
C--DRAW EXTENDED BENCH ON PLOT
C
        CALL PLOT(XE(1),YE(1),3)
        CALL PLOT(XE(5),YE(5),2)
        CALL PLOT((X(16)+E1+EB-E2*.5)/XSCALE,
     +                (Y(16)+0.5*E2*TANT)/YSCALE,2)
      ENDIF
C
C--DRAW RECTANGLE TO REPRESENT DRAGLINE WHERE :  VALUE(10) = P
C                                                VALUE(11) = D
C
      RECTX(1)=(X(10)+ET-VALUE(11)*(VALUE(10)-0.5))/XSCALE
      RECTY(1)=Y(10)/YSCALE
      RECTX(2)=RECTX(1)
      RECTX(3)=RECTX(2)-(VALUE(11)/XSCALE)
      RECTX(4)=RECTX(3)
      RECTY(2)=RECTY(1)+(5./YSCALE)
      RECTY(3)=RECTY(2)
      RECTY(4)=RECTY(1)
      CALL STFILL(1)
      CALL FILL(RECTX,RECTY,4)
C
C--WRITE TITLE ON SCREEN
C
      CALL SYMBOL(3.0,YVAL,HITE,'SINGLE SEAM SIDECASTING',0.,23)
C
C--WRITE INPUT VALUES ON SCREEN FOR COLUMN #1
C
      DO 100 I=1,6
      CALL NUMBER(0.,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,VALUE(I),0.,2)
100   CALL SYMBOL(1.1,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,NAME(I),0.,25)
C
C--WRITE INPUT VALUES ON SCREEN FOR COLUMN #2
C
      DO 200 I=1,6
      CALL NUMBER
     +(5.0,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,VALUE(I+6),0.,2)
200   CALL SYMBOL(6.1,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,NAME(I+6),0.,25)
C
C--WRITE SCALE VALUE ON OUTPUT
C
```

```
      SELECT CASE(NUM)
        CASE(3)
          CALL SYMBOL((X(19)+25.)/XSCALE,0.,HITE,'1:500 ',0.,6)
        CASE(4,5)
          CALL SYMBOL((X(19)+25.)/XSCALE,0.,HITE,'1:1000',0.,6)
      END SELECT
C
C--DRAW OPERATING RADIUS VALUE ON RANGE DIAGRAM
C
      X1=(X(10)+ET-VALUE(10)*VALUE(11))/XSCALE
      CALL PLOT(X1,(Y(17)+5.)/YSCALE,3)
      CALL PLOT(X1,(Y(17)+10.)/YSCALE,2)
      CALL PLOT(X1,(Y(17)+7.5)/YSCALE,3)
      X2=X(17)/XSCALE
      CALL PLOT(X2,(Y(17)+7.5)/YSCALE,2)
      CALL PLOT(X2,(Y(17)+5.)/YSCALE,3)
      CALL PLOT(X2,(Y(17)+10.)/YSCALE,2)
      CALL NUMBER(X1-.5+(X2-X1)*.5,(Y(17)+8.5)/YSCALE,HITE,OROB,0.,2)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
C
C--DRAW HEIGHT VALUE ON RANGE DIAGRAM
C
      Y1=Y(10)/YSCALE
      CALL PLOT((X(10)+ET+5.)/XSCALE,Y1,3)
      CALL PLOT((X(10)+ET+10.)/XSCALE,Y1,2)
      CALL PLOT((X(10)+ET+7.5)/XSCALE,Y1,3)
      Y2=Y(17)/YSCALE
      CALL PLOT((X(10)+ET+7.5)/XSCALE,Y2,2)
      CALL PLOT((X(10)+ET+5.)/XSCALE,Y2,3)
      CALL PLOT((X(10)+ET+10.)/XSCALE,Y2,2)
      CALL NUMBER((X(10)+ET+8.5)/XSCALE,Y1-.07+(Y2-Y1)*.5,HITE,H,0.,2)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
C
C--LABEL DIFFERENT PIT WIDTHS
C
      Y1=10./YSCALE
      Y2=5./YSCALE
      CALL PLOT(X(1)/XSCALE,Y1,3)
      CALL PLOT(X(1)/XSCALE,Y2,2)
      CALL PLOT(X(14)/XSCALE,Y1,3)
      CALL PLOT(X(14)/XSCALE,Y2,2)
      CALL PLOT(X(19)/XSCALE,Y1,3)
      CALL PLOT(X(19)/XSCALE,Y2,2)
      CALL PLOT(X(19)/XSCALE,7.5/YSCALE,3)
      CALL PLOT(X(1)/XSCALE,7.5/YSCALE,2)
      CALL NUMBER((X(1)+.5*(X(14)-X(1)))/XSCALE-.4,0.,HITE,
     +                  VALUE(1),0.,1)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
      CALL NUMBER((X(14)+.5*(X(19)-X(14)))/XSCALE-.4,0.,HITE,
     +                  VALUE(2),0.,1)
```

```
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
C
      CALL PLOT(0.0,0.0,999)
      RETURN
C
      ENTRY INITR1()
C
C--IF A PRINT.DAT FILE IS PRESENT, THE PARAMETERS WILL BE READ
C  FROM THAT FILE, OTHERWISE DEFAULT PARAMETERS WILL BE USED
C
      INQUIRE(FILE='PRINT.DAT',EXIST=EX)
      IF(EX)THEN
        OPEN(5,FILE='PRINT.DAT')
        READ(5,*)PIOPRT
        READ(5,*)PMODEL
        READ(5,*)PWIDTH
        CLOSE(5)
      ELSE
C
C--IF WIDE CARRIAGE IS DESIRED FOR OUTPUT, SPECIFY IN PRINT.DAT
C  MODEL VALUE FOR LQ500 IS 42
C
        PIOPRT=0
        PMODEL=11
        PWIDTH=0.
      ENDIF
C
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE PLOTR2(X,Y,N,NUM,VALUE,NAME,HX,HY)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO PLOT THE RANGE DIAGRAM FOR A
C  TWO SEAM DRAGLINE CUT. THE DRAGLINE REACH DIMENSIONS ARE
C  DISPLAYED ON THE DRAWING AS WELL AS THE O/B LOCATION.
C  THIS SUBROUTINE USES SEVERAL ROUTINES FROM THE PLOT88
C  GRAPHICS SOFTWARE LIBRARY TO DISPLAY THE RANGE DIAGRAMS.
C
C  INPUT VARIABLES:
C
C N=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF POINTS DEFINING
C       THE RANGE DIAGRAM
C  X,Y= REAL*4 VECTORS CONTAINING THE COORDINATES OF THE RANGE
C     DIAGRAM
C  NUM= INTEGER*2 VARIABLE CONTAINING THE NUMBER WHICH DEFINES
C       THE MODE OF OUTPUT WHERE:
C
C                 NUM=1  -->  CGA SCREEN
C                 NUM=2  -->  EGA SCREEN
C                 NUM=3  -->  PRINTER (1:500)
C                 NUM=4  -->  PRINTER (1:1000)
C                 NUM=5  -->  PLOTTER (1:1000)
C
C  NAME= CHARACTER*25 VECTOR CONTAINING THE NAMES OF THE DATA
C     PARAMETERS
C  VALUE= REAL*4 VECTORS CONTAINING THE VALUES OF THE DATA
C       PARAMETERS
C  HX,HY= REAL*4 VECTORS CONTAINING THE (X,Y) POINT PAIRS OF
C       THE O/B SPOIL PILE
C
C  INTERNAL VARIABLES:
C
C  RECTX(4),RECTY(4)= REAL*4 VECTORS CONTAINING THE COORDINATES
C       OF THE RECTANGLE REPRESENTING THE DRAGLINE
C MAXX,MAXY= REAL*4 VARIABLES CONTAINING THE MAXIMUM COORDINATE
C       IN THE X AND Y DIRECTION RESPECTIVELY
C XSCALE,YSCALE= REAL*4 VARIABLES CONTAINING THE SCALE FACTORS
C       IN THE X AND Y DIRECTION RESPECTIVELY
C  PWIDTH= INTEGER*2 VARIABLE CONTAINING A NUMBER REPRESENTING
C       THE WIDTH OF THE PRINTER BEING USED.
C       IF PWIDTH=0 --> NARROW PRINTER CARRIAGE
C       IF PWIDTH=1 --> WIDE PRINTER CARRIAGE
C  HITE= REAL*4 VARIABLE CONTAINING THE HEIGHT THAT THE TEXT
C       WILL BE DRAWN AT.
C
C  OUTPUT VARIABLES:
C
```

```
C  X,Y,N,NUM,VALUE,NAME,HX,HY=  UNCHANGED
C
C  SUBROUTINES REQUIRED:
C
C  PLOTS=  A PLOT88 ROUTINE WHICH MUST BE CALLED TO INITIALIZE
C      THE PLOT88 SOFTWARE
C  FACTOR=  A PLOT88 ROUTINE WHICH IS USED TO ENLARGE OR REDUCE
C      THE SIZE OF THE ENTIRE DRAWING.
C  WINDOW=  A PLOT88 ROUTINE WHICH DEFINES THE WINDOW IN WHICH
C      THE DRAWING WILL BE MADE.
C  PLOT=  A PLOT88 ROUTINE WHICH MOVES THE DRAWING UTENSIL FROM
C      THE CURRENT POSITION TO A NEW POSITION.  THE 'PEN' MAY
C      BE UP OR DOWN AS IT IS BEING MOVED.
C  LINE=  A PLOT88 ROUTINE WHICH DRAWS A LINE THROUGH (X,Y) POINT
C      PAIRS
C  STFILL=  A PLOT88 ROUTINE WHICH IS USED TO SELECT THE SHADING
C      PATTERN TO USE IN FUTURE CALLS TO THE FILL ROUTINE
C  FILL=  A PLOT88 ROUTINE WHICH PROVIDES SHADING OF USER
C      SPECIFIED POLYGONAL AREAS
C  SYMBOL=  A PLOT88 ROUTINE WHICH IS USED TO DRAW CHARACTER
C      STRINGS AND SPECIAL CENTERED SYMBOLS.
C  NUMBER=  A PLOT88 ROUTINE WHICH IS USED TO CONVERT A SINGLE
C      PRECISION REAL NUMBER TO ITS DECIMAL EQUIVALENT AS A
C      CHARACTER STRING AND TO DRAW THE STRING USING SYMBOL.
C
C
   REAL*4 X(*),Y(*),FACT,MAXX,MAXY,VALUE(*),RECTX(4),RECTY(4)
   REAL*4 XSCALE,YSCALE,X1,X2,YVAL,HITE,Y1,Y2,HX(*),HY(*),YT
   INTEGER*2 IOPORT,MODEL,N,NUM,PMODEL,PIOPRT,I,PWIDTH
   LOGICAL*2 EX
   CHARACTER NAME(*)*25
C
   FACT=1.0
   HITE=.15
   SELECT CASE(NUM)
     CASE(1)
C
C      SCALE TO FIT CGA SCREEN
C
      MAXY=AMAX1(Y(9),Y(29))
      MAXX=X(29)
      IF(.3*MAXX.GT.MAXY)THEN
        X(31)=MAXX/9.
      ELSE
        X(31)=MAXY/2.7
      ENDIF
      Y(31)=X(31)
      IOPORT = 99
      MODEL  = 99
      YVAL=7.1
```

```
        FACT=0.8
        HITE=.2
       CASE(2)
C
C       SCALE TO FIT EGA SCREEN
C
        MAXY=AMAX1(Y(9),Y(29))
        MAXX=X(29)
        IF(.39*MAXX.GT.MAXY)THEN
          X(31)=MAXX/9.
        ELSE
          X(31)=MAXY/3.5
        ENDIF
        Y(31)=X(31)
        IOPORT = 97
        MODEL  = 97
        YVAL=7.1
       CASE(3)
C
C       PRINTER SETTING (1:500)
C       NARROW PRINTER (80 COL)
C
        IOPORT = PIOPRT
        MODEL  = PMODEL
        X(31) = 12.70
        Y(31) = 12.70
        YVAL=7.8
       CASE(4)
C
C       PRINTER SETTING (1:1000)
C       NARROW PRINTER (80 COL)
C
        IOPORT = PIOPRT
        MODEL  = PMODEL
        X(31) = 25.40
        Y(31) = 25.40
        YVAL=7.8
       CASE(5)
C
C       HP7475A PLOTTER SETTING (1:1000)
C       ASSUMING 8.5"x11" PAPER USED
C
        IOPORT = 9602
        MODEL  = 30
        X(31) = 25.40
        Y(31) = 25.40
        YVAL=7.8
       CASE DEFAULT
        RETURN
      END SELECT
      XSCALE=X(31)
      YSCALE=Y(31)
C
      CALL PLOTS(0,IOPORT,MODEL)
      CALL FACTOR(FACT)
C
C--INDICATE ON THE SCREEN THAT THE DRAWING IS BEING GENERATED
C
      WRITE(*,*)'DRAWING IS BEING GENERATED ...'
      WRITE(*,*)' '
C
C--ALLOW PRINTER TO PRINT ON MORE THAN ONE PAGE AND SET PAGE
C  HEIGHT
C
      IF(NUM.EQ.3.OR.NUM.EQ.4) THEN
        IF(PWIDTH.EQ.1)YVAL=YVAL+5.
        CALL WINDOW(0.,0.,22.,YVAL+.5)
      ENDIF
C
C     MOVE TO (0.0,0.0), NO LINE IS DRAWN, AND DEFINE A NEW ORIGIN
C
      CALL PLOT(0.0,0.0,-3)
      CALL LINE(X,Y,N,1,0,2)
C
C--DRAW RECTANGLE TO REPRESENT DRAGLINE ON O/B   VALUE(18) = PO
C                                                VALUE(21) = D
C
      RECTX(1)=(X(14)-VALUE(21)*(VALUE(18)-0.5))/XSCALE
      RECTX(2)=RECTX(1)
      RECTX(3)=RECTX(2)-(VALUE(21)/XSCALE)
      RECTX(4)=RECTX(3)
      RECTY(1)=Y(14)/YSCALE
      RECTY(2)=RECTY(1)+(5./YSCALE)
      RECTY(3)=RECTY(2)
      RECTY(4)=RECTY(1)
      CALL STFILL(1)
      CALL FILL(RECTX,RECTY,4)
C
      CALL SYMBOL(3.1,YVAL,HITE,'TWO SEAM SIDECASTING',0.,20)
C
C--WRITE INPUT VALUES ON SCREEN FOR COLUMN #1
C
      DO 100 I=1,12
      CALL NUMBER(0.,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,VALUE(I),0.,2)
100   CALL SYMBOL(1.1,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,NAME(I),0.,25)
C
C--WRITE INPUT VALUES ON SCREEN FOR COLUMN #2
C
      DO 200 I=1,11
      CALL NUMBER(5.0,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,
    +                    VALUE(I+12),0.,2)
200   CALL SYMBOL(6.1,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,
    +                    NAME(I+12),0.,25)
C
C--WRITE SCALE VALUE ON OUTPUT
C
      SELECT CASE(NUM)
        CASE(3)
          CALL SYMBOL((X(28)+25.)/XSCALE,0.,HITE,'1:500 ',0.,6)
        CASE(4,5)
          CALL SYMBOL((X(28)+25.)/XSCALE,0.,HITE,'1:1000',0.,6)
      END SELECT
C
C--DRAW O/B OPERATING RADIUS VALUE ON RANGE DIAGRAM
C
      X1=(X(14)-VALUE(21)*VALUE(18))/XSCALE
      IF(Y(14).GT.HY(2))THEN
        YT=Y(14)
      ELSE
        YT=HY(2)
      ENDIF
      CALL PLOT(X1,(YT+5.)/YSCALE,3)
      CALL PLOT(X1,(YT+10.)/YSCALE,2)
      CALL PLOT(X1,(YT+7.5)/YSCALE,3)
      IF(HY(2).GT.HY(1))THEN
        X2=HX(2)/XSCALE
      ELSE
        X2=HX(1)/XSCALE
      ENDIF
      CALL PLOT(X2,(YT+7.5)/YSCALE,2)
      CALL PLOT(X2,(YT+5.)/YSCALE,3)
      CALL PLOT(X2,(YT+10.)/YSCALE,2)
      CALL NUMBER(X1-.5+(X2-X1)*.5,(YT+8.5)/YSCALE,HITE,
     +                    VALUE(22),0.,2)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
C
C--DRAW O/B SPOIL
C
      CALL PLOT(HX(1)/XSCALE,HY(1)/YSCALE,3)
      CALL PLOT(HX(2)/XSCALE,HY(2)/YSCALE,2)
      CALL PLOT(HX(3)/XSCALE,HY(3)/YSCALE,2)
C
C--LABEL DIFFERENT PIT WIDTHS
C
      Y1=10./YSCALE
      Y2=5./YSCALE
      CALL PLOT(X(1)/XSCALE,Y1,3)
      CALL PLOT(X(1)/XSCALE,Y2,2)
      CALL PLOT(X(25)/XSCALE,Y1,3)
      CALL PLOT(X(25)/XSCALE,Y2,2)
      CALL PLOT(X(28)/XSCALE,Y1,3)
      CALL PLOT(X(28)/XSCALE,Y2,2)
      CALL PLOT(X(28)/XSCALE,7.5/YSCALE,3)
      CALL PLOT(X(1)/XSCALE,7.5/YSCALE,2)
      CALL NUMBER((X(1)+.5*(X(25)-X(1)))/XSCALE-.4,0.,HITE,
     +                   VALUE(2),0.,1)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
      CALL NUMBER((X(25)+.5*(X(28)-X(25)))/XSCALE-.4,0.,HITE,
     +                   VALUE(3),0.,1)
      CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
C
      CALL PLOT(0.0,0.0,999)
      RETURN
C
      ENTRY INITR2()
C
C--IF A PRINT.DAT FILE IS PRESENT, THE PARAMETERS WILL BE READ
C  FROM THAT FILE, OTHERWISE DEFAULT PARAMETERS WILL BE USED
C
      INQUIRE(FILE='PRINT.DAT',EXIST=EX)
      IF(EX)THEN
        OPEN(5,FILE='PRINT.DAT')
        READ(5,*)PIOPRT
        READ(5,*)PMODEL
        READ(5,*)PWIDTH
        CLOSE(5)
      ELSE
C
C--IF WIDE CARRIAGE IS DESIRED FOR OUTPUT, SPECIFY IN PRINT.DAT
C  MODEL VALUE FOR LQ500 IS 42
C
        PIOPRT=0
        PMODEL=42
        PWIDTH=0.
      ENDIF
C
      RETURN
      END
```

```
$STORAGE:2
   SUBROUTINE PLOTR3
        (X,Y,N,NUM,VALUE,NAME,SPX,SPY,HX,HY,EX,EY,IFLAG)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO PLOT THE RANGE DIAGRAM FOR A
C  TWO SEAM DRAGLINE CUT WHICH ALLOWS AN EXTENDED BENCH TO BE
C  USED.  THIS SUBROUTINE ALSO ALLOWS 2 DIFFERENT POSITIONS TO
C  BE USED WHEN REHANDLING MATERIAL IF IT IS REQUIRED.
C  THIS SUBROUTINE USES SEVERAL ROUTINES FROM THE PLOT88
C  GRAPHICS SOFTWARE LIBRARY TO DISPLAY THE RANGE DIAGRAMS.
C
C  INPUT VARIABLES:
C
C  N=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF POINTS DEFINING
C      THE RANGE DIAGRAM
C  X,Y= REAL*4 VECTORS CONTAINING THE COORDINATES OF THE RANGE
C      DIAGRAM
C  NUM= INTEGER*2 VARIABLE CONTAINING THE NUMBER WHICH DEFINES
C      THE MODE OF OUTPUT WHERE:
C
C                NUM=1  -->  CGA SCREEN
C                NUM=2  -->  EGA SCREEN
C                NUM=3  -->  PRINTER (1:500)
C                NUM=4  -->  PRINTER (1:1000)
C                NUM=5  -->  PLOTTER (1:1000)
C
C  NAME= CHARACTER*25 VECTOR CONTAINING THE NAMES OF THE DATA
C      PARAMETERS
C  VALUE= REAL*4 VECTORS CONTAINING THE VALUES OF THE DATA
C      PARAMETERS
C  HX,HY= REAL*4 VECTORS CONTAINING THE (X,Y) POINT PAIRS OF
C      THE O/B SPOIL PILE
C  SPX,SPY= REAL*4 VECTORS CONTAINING THE (X,Y) POINT PAIRS OF
C      THE FINAL SPOIL PILE
C  EX,EY= REAL*4 VECTORS CONTAINING THE (X,Y) POINT PAIRS OF
C      THE EXTENDED BENCH MATERIAL
C  IFLAG= INTEGER*2 VARIABLE CONTAINING A NUMBER REPRESENTING
C      THE TYPE OF POSITIONNING ON THE I/B
C
C        IF IFLAG=1, NO EXTENDED BENCH REQUIRED
C        IF IFLAG=2, EXTENDED BENCH IS REQUIRED BUT THE DRAGLINE
C                    CAN SPOIL ALL THE SPOIL FROM ONE POSITION
C        IF IFLAG=3, EXTENDED BENCH IS REQUIRED BUT THE DRAGLINE
C                    NEEDS TWO POSITIONS TO SPOIL ALL OF THE
C                    MATERIAL
C
C  INTERNAL VARIABLES:
C
```

```
C MAXX,MAXY= REAL*4 VARIABLES CONTAINING THE MAXIMUM COORDINATE
C       IN THE X AND Y DIRECTION RESPECTIVELY
C  XSCALE,YSCALE= REAL*4 VARIABLES CONTAINING THE SCALE FACTORS
C       IN THE X AND Y DIRECTION RESPECTIVELY
C  PWIDTH= INTEGER*2 VARIABLE CONTAINING A NUMBER REPRESENTING
C       THE WIDTH OF THE PRINTER BEING USED.
C       IF PWIDTH=0 --> NARROW PRINTER CARRIAGE
C       IF PWIDTH=1 --> WIDE PRINTER CARRIAGE
C  HITE= REAL*4 VARIABLE CONTAINING THE HEIGHT THAT THE TEXT
C       WILL BE DRAWN AT.
C
C  OUTPUT VARIABLES:
C
C  X,Y,N,NUM,VALUE,NAME,SPX,SPY,HX,HY,EX,EY,IFLAG= UNCHANGED
C
C  SUBROUTINES REQUIRED:
C
C  PLOTS= A PLOT88 ROUTINE WHICH MUST BE CALLED TO INITIALIZE
C       THE PLOT88 SOFTWARE
C  FACTOR= A PLOT88 ROUTINE WHICH IS USED TO ENLARGE OR REDUCE
C       THE SIZE OF THE ENTIRE DRAWING.
C  WINDOW= A PLOT88 ROUTINE WHICH DEFINES THE WINDOW IN WHICH
C       THE DRAWING WILL BE MADE.
C  PLOT= A PLOT88 ROUTINE WHICH MOVES THE DRAWING UTENSIL FROM
C       THE CURRENT POSITION TO A NEW POSITION.  THE 'PEN' MAY
C       BE UP OR DOWN AS IT IS BEING MOVED.
C  LINE= A PLOT88 ROUTINE WHICH DRAWS A LINE THROUGH (X,Y) POINT
C       PAIRS
C  STFILL= A PLOT88 ROUTINE WHICH IS USED TO SELECT THE SHADING
C       PATTERN TO USE IN FUTURE CALLS TO THE FILL ROUTINE
C  FILL= A PLOT88 ROUTINE WHICH PROVIDES SHADING OF USER
C       SPECIFIED POLYGONAL AREAS
C  SYMBOL= A PLOT88 ROUTINE WHICH IS USED TO DRAW CHARACTER
C       STRINGS AND SPECIAL CENTERED SYMBOLS.
C  NUMBER= A PLOT88 ROUTINE WHICH IS USED TO CONVERT A SINGLE
C       PRECISION REAL NUMBER TO ITS DECIMAL EQUIVALENT AS A
C       CHARACTER STRING AND TO DRAW THE STRING USING SYMBOL.
C  STDASH= A PLOT88 ROUTINE WHICH IS USED TO DEFINE THE
C       CHARACTERISTICS OF A DASHED LINE
C PLOTD= A PLOT88 ROUTINE WHICH IS USED TO PRODUCE DASHED LINES
C
C
      REAL*4 X(*),Y(*),FACT,MAXX,MAXY,VALUE(*)
      REAL*4 XSCALE,YSCALE,X1,YVAL,HITE,Y1,Y2,HX(*),HY(*)
      REAL*4 SPX(*),SPY(*),EX(*),EY(*)
      INTEGER*2 IOPORT,MODEL,N,NUM,PMODEL,PIOPRT,I,PWIDTH,IFLAG
      LOGICAL*2 EXIS
      CHARACTER NAME(*)*25
C
      FACT=1.0
      HITE=.15
      SELECT CASE(NUM)
        CASE(1)
C
C         SCALE TO FIT CGA SCREEN
C
          MAXY=AMAX1(Y(9),Y(29))
          MAXX=X(29)
          IF(.3*MAXX.GT.MAXY)THEN
            X(31)=MAXX/9.
          ELSE
            X(31)=MAXY/2.7
          ENDIF
          Y(31)=X(31)
          IOPORT = 99
          MODEL  = 99
          YVAL=7.1
          FACT=0.8
          HITE=.2
        CASE(2)
C
C         SCALE TO FIT EGA SCREEN
C
          MAXY=AMAX1(Y(9),Y(29))
          MAXX=X(29)
          IF(.39*MAXX.GT.MAXY)THEN
            X(31)=MAXX/9.
          ELSE
            X(31)=MAXY/3.5
          ENDIF
          Y(31)=X(31)
          IOPORT = 97
          MODEL  = 97
          YVAL=7.1
        CASE(3)
C
C         PRINTER SETTING (1:500)
C         NARROW PRINTER (80 COL)
C
          IOPORT = PIOPRT
          MODEL  = PMODEL
          X(31) = 12.70
          Y(31) = 12.70
          YVAL=7.8
        CASE(4)

          PRINTER SETTING (1:1000)
          NARROW PRINTER (80 COL)

          IOPORT = PIOPRT
        MODEL  = PMODEL
        X(31) = 25.40
        Y(31) = 25.40
        YVAL=7.8
      CASE(5)
C
C       HP7475A PLOTTER SETTING (1:1000)
C       ASSUMING 8.5"x11" PAPER USED
C
        IOPORT = 9602
        MODEL  = 30
        X(31) = 25.40
        Y(31) = 25.40
        YVAL=7.8
      CASE DEFAULT
        RETURN
    END SELECT
    XSCALE=X(31)
    YSCALE=Y(31)
C
    CALL PLOTS(0,IOPORT,MODEL)
    CALL FACTOR(FACT)
C
C--INDICATE ON THE SCREEN THAT THE DRAWING IS BEING GENERATED
C
    WRITE(*,*)'DRAWING IS BEING GENERATED ...'
    WRITE(*,*)' '
C
C--ALLOW PRINTER TO PRINT ON MORE THAN ONE PAGE AND SET PAGE
C  HEIGHT
C
    IF(NUM.EQ.3.OR.NUM.EQ.4) THEN
      IF(PWIDTH.EQ.1)YVAL=YVAL+5.
      CALL WINDOW(0.,0.,22.,YVAL+.5)
    ENDIF
C
C    MOVE TO (0.0,0.0), NO LINE IS DRAWN, AND DEFINE A NEW ORIGIN
C
    CALL PLOT(0.0,0.0,-3)
    CALL LINE(X,Y,N,1,0,2)
C
C--WRITE TITLE ON SCREEN
C
    CALL SYMBOL(3.1,YVAL,HITE,'TWO SEAM SIDECASTING',0.,20)
C
C--WRITE INPUT VALUES ON SCREEN FOR COLUMN #1
C
    DO 100 I=1,12
    CALL NUMBER(0.,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,VALUE(I),0.,2)
100  CALL SYMBOL(1.1,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,NAME(I),0.,25)
```

```
C
C--WRITE INPUT VALUES ON SCREEN FOR COLUMN #2
C
    DO 200 I=1,11
    CALL NUMBER(5.0,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,
   +                  VALUE(I+12),0.,2)
200  CALL SYMBOL(6.1,YVAL-.4-FLOAT(I-1)*(HITE+.1),HITE,
   +                  NAME(I+12),0.,25)
C
C--WRITE SCALE VALUE ON OUTPUT
C
    SELECT CASE(NUM)
      CASE(3)
        CALL SYMBOL((X(28)+25.)/XSCALE,0.,HITE,'1:500 ',0.,6)
      CASE(4,5)
        CALL SYMBOL((X(28)+25.)/XSCALE,0.,HITE,'1:1000',0.,6)
    END SELECT
C
C--DRAW O/B SPOIL
C
    CALL STDASH(.0556,.1110)
    CALL PLOTD(HX(1)/XSCALE,HY(1)/YSCALE,3)
    CALL PLOTD(HX(2)/XSCALE,HY(2)/YSCALE,2)
    CALL PLOTD(HX(3)/XSCALE,HY(3)/YSCALE,2)
C
C--DRAW EXTENDED BENCH IF NECESSARY
C
    IF(IFLAG.EQ.2.OR.IFLAG.EQ.3)THEN
      CALL PLOTD(EX(1)/XSCALE,EY(1)/YSCALE,3)
      CALL PLOTD(EX(2)/XSCALE,EY(2)/YSCALE,2)
      CALL PLOTD(EX(3)/XSCALE,EY(3)/YSCALE,2)
    ENDIF
C
C--DRAW FINAL SPOIL
C
    CALL PLOT(SPX(1)/XSCALE,SPY(1)/YSCALE,3)
    CALL PLOT(SPX(2)/XSCALE,SPY(2)/YSCALE,2)
    CALL PLOT(SPX(3)/XSCALE,SPY(3)/YSCALE,2)
    CALL PLOT(SPX(4)/XSCALE,SPY(4)/YSCALE,2)
    CALL PLOT(SPX(5)/XSCALE,SPY(5)/YSCALE,2)
    CALL PLOT(SPX(6)/XSCALE,SPY(6)/YSCALE,2)
C
C--LABEL DRAGLINE POSITIONS
C
    X1=X(14)-VALUE(18)*VALUE(21)
    CALL PLOT(X1/XSCALE,(Y(14)+2.)/YSCALE,3)
    CALL PLOT(X1/XSCALE,(Y(14)+7.)/YSCALE,2)
    CALL SYMBOL(X1/XSCALE-.1,(Y(14)+9.)/YSCALE,HITE*.65,'C/L',0.,3)
    IF(IFLAG.EQ.1)THEN
      X1=X(21)-VALUE(19)*VALUE(21)
      CALL PLOT(X1/XSCALE,(Y(21) + 2.)/YSCALE,3)
      CALL PLOT(X1/XSCALE,(Y(21) + 7.)/YSCALE,2)
      CALL SYMBOL(X1/XSCALE-.1,(Y(21) + 9.)/YSCALE,HITE*.65,'C/L',0.,3)
    ELSEIF(IFLAG.EQ.2)THEN
      X1 = SPX(2)-VALUE(20)*VALUE(21)
      CALL PLOT(X1/XSCALE,(Y(21) + 2.)/YSCALE,3)
      CALL PLOT(X1/XSCALE,(Y(21) + 7.)/YSCALE,2)
      CALL SYMBOL(X1/XSCALE-.1,(Y(21) + 9.)/YSCALE,HITE*.65,'C/L',0.,3)
    ELSEIF(IFLAG.EQ.3)THEN
      X1 = SPX(2)-VALUE(20)*VALUE(21)
      CALL PLOT(X1/XSCALE,(Y(21) + 2.)/YSCALE,3)
      CALL PLOT(X1/XSCALE,(Y(21) + 7.)/YSCALE,2)
      CALL SYMBOL(X1/XSCALE-.2,(Y(21) + 9.)/YSCALE,HITE*.65,'C/L',0.,3)
      X1 = EX(2)-VALUE(20)*VALUE(21)
      CALL PLOT(X1/XSCALE,(Y(21) + 2.)/YSCALE,3)
      CALL PLOT(X1/XSCALE,(Y(21) + 7.)/YSCALE,2)
      CALL SYMBOL(X1/XSCALE,(Y(21) + 9.)/YSCALE,HITE*.65,'C/L',0.,3)
C
C--LABEL SPOIL FROM SECOND PEAK
C
      CALL PLOTD(SPX(4)/XSCALE,SPY(4)/YSCALE,3)
      CALL PLOTD(EX(4)/XSCALE,EY(4)/YSCALE,2)
    ENDIF
C--LABEL DIFFERENT PIT WIDTHS
    Y1 = 10./YSCALE
    Y2 = 5./YSCALE
    CALL PLOT(X(1)/XSCALE,Y1,3)
    CALL PLOT(X(1)/XSCALE,Y2,2)
    CALL PLOT(X(25)/XSCALE,Y1,3)
    CALL PLOT(X(25)/XSCALE,Y2,2)
    CALL PLOT(X(28)/XSCALE,Y1,3)
    CALL PLOT(X(28)/XSCALE,Y2,2)
    CALL PLOT(X(28)/XSCALE,7.5/YSCALE,3)
    CALL PLOT(X(1)/XSCALE,7.5/YSCALE,2)
    CALL NUMBER((X(1) + .5*(X(25)-X(1)))/XSCALE-.4,0.,HITE,
    +                     VALUE(2),0.,1)
    CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
    CALL NUMBER((X(25) + .5*(X(28)-X(25)))/XSCALE-.4,0.,HITE,
    +                     VALUE(3),0.,1)
    CALL SYMBOL(999.0,999.0,HITE,' m',0.,2)
    CALL PLOT(0.0,0.0,999)
    RETURN
C
    ENTRY INITR3()
C--IF A PRINT.DAT FILE IS PRESENT, THE PARAMETERS WILL BE READ
C  FROM THAT FILE, OTHERWISE DEFAULT PARAMETERS WILL BE USED
    INQUIRE(FILE = 'PRINT.DAT',EXIST = EXIS)
    IF(EXIS)THEN
      OPEN(5,FILE = 'PRINT.DAT')
      READ(5,*)PIOPRT
```

```
        READ(5,*)PMODEL
        READ(5,*)PWIDTH
        CLOSE(5)
      ELSE
C--IF WIDE CARRIAGE IS DESIRED FOR OUTPUT, SPECIFY IN PRINT.DAT
C  MODEL VALUE FOR LQ500 IS 42
        PIOPRT=0
        PMODEL=42
        PWIDTH=0.
      ENDIF
      RETURN
      END
      REAL*4 FUNCTION F(X,COEF,N)
C
C  PURPOSE:
C
C  THIS FUNCTION IS USED TO SOLVE FOR THE VALUE OF A (N-1)TH
C POLYNOMIAL AT X. IT IS MEANT TO BE USED AS A GENERAL FUNCTION
C  SUCH THAT THE NUMBER OF COEFFICIENTS AS WELL AS THE VALUES
C  OF THE COEFFICIENTS CAN BE PASSED AS INPUT VARIABLES.
C
C  INPUT VARIABLES:
C
C  X= REAL*4 VARIABLE CONTAINING THE VALUE AT WHICH THE
C      POLYNOMIAL IS BEING EVALUATED AT
C  COEF= REAL*4 ARRAY OF A LENGTH DECLARED IN THE CALLING
C      PROGRAM. THIS ARRAY CONTAINS THE COEFFICIENTS OF THE
C      POLYNOMIAL
C N=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF COEFFICIENTS IN
C        THE POLYNOMIAL
C
C  OUTPUT VARIABLES:
C
C  X,COEF,N= UNCHANGED
C  F= REAL*4 VARIABLE CONTAINING THE VALUE OF THE POLYNOMIAL AT
C      X
C
C  ROUTINES REQUIRED: NONE
C
C
      REAL*4 X, COEF(*), XX
      INTEGER*2 N,I
      F=0.
      XX=1.
      DO 10 I=1,N
        F=F+XX*COEF(I)
   10 XX=XX*X
      RETURN
      END
```

```
$STORAGE:2
      REAL*4 FUNCTION AREA(X,Y,N)
C
C  PURPOSE:
C
C  THIS FUNCTION USES THE METHOD OF COORDINATES THE CALCULATE
C  THE AREA OF A SIMPLE CLOSED PLANE POLYGON.  A SIMPLE CLOSED
C  POLYGON IS DEFINED AS AN AREA IN A PLANE WHOSE BOUNDS ARE
C  CONNECTED, NON-INTERSECTING, STRAIGHT LINE SEGMENTS.
C  THE USER IS RESPONSIBLE FOR ASSURING THAT THE POLYGON IS
C  SIMPLE CLOSED.
C
C  INPUT VARIABLES:
C
C  X,Y= REAL*4 VECTORS OF LENGTH N CONTAINING THE ORDERED
C       COORDINATE LOCATIONS OF THE POLYGON SIDES.  THE DIRECTION
C       OF ORDERING DOES NOT MATTER
C N= INTEGER*2 VARIABLE CONTAINING THE NUMBER OF POINTS DEFINING
C       THE POLYGON
C
C  OUTPUT:
C
C  INPUT IS UNCHANGED
C
C  OUTPUT VARIABLES:
C
C  AREA= REAL*4 VARIABLE CONTAINING THE CALCULATED AREA OF THE
C       POLYGON
C
C  ERRORS:
C
C  THE ONLY ERROR CHECK IS THAT THERE MUST BE AT LEAST 3 SIDES
C  BEFORE AN AREA CALCULATION WILL BE ATTEMPTED
C
C  ROUTINES REQUIRED: NONE
C
C  NOTES:
C
C  1. THE USER IS RESPONSIBLE FOR ASSURING THAT THE POLYGON IS
C       SIMPLE CLOSED.
C  2. ALL INTERNAL CALCULATIONS ARE IN DOUBLE PRECISION.
C
C
      REAL*4 X(*),Y(*)
      REAL*8 X1,Y1,X2,Y2,SUM
      INTEGER*2 I,N
C
      SUM=0.D0
      AREA = 0.0
      IF(N.LT.3)
```

```
     STOP'POLYGON HAS LESS THAN 3 SIDES IN AREA CALCULATION'
   X1=X(N)
   Y1=Y(N)
   DO 10 I=1,N
     X2=X(I)
     Y2=Y(I)
     SUM = SUM + X1*Y2 - X2*Y1
     X1=X2
10   Y1=Y2
   AREA = DABS( SUM/2.D0)
C
   RETURN
   END
```

```
$STORAGE:2
      INTEGER*2 FUNCTION LNINTS(XI,YI,XJ,YJ,XK,YK,XL,YL)
C
C
C   PURPOSE:
C
C   THIS FUNCTION IS TO TEST WHETHER LINE SEGMENT XI,YI TO XJ,YJ
C INTERCEPTS LINE XK,YK TO XL,YL.  ONLY TRUE CROSSING INTERCEPTS
C   ARE CONSIDERED.  THE FOLLOWING RETURNED VALUES FOR LNINTS
C   WILL ELUCIDATE.
C
C
C   INPUT:
C
C   XI,YI,XJ,YJ,XK,YK,XL,YL = REAL VARIABLES OR CONSTANTS THAT
C      DEFINE TWO LINES I-J AND K-L IN TWO SPACE.
C
C   OUTPUT:
C
C   INPUT IS UNCHANGED
C
C   RETURNED VALUES:
C
C   LNINTS=-1 THE LINE SEGMENTS HAVE A TRUE CROSSING
C   LNINTS=0 ONE OR BOTH LINES DEGENERATE TO A POINT
C   LNINTS=1 LINES ARE PARALLEL
C   LNINTS=2 POINTS I AND K ARE COINCIDENT
C   LNINTS=3 POINTS I AND L ARE COINCIDENT
C   LNINTS=4 POINTS J AND K ARE COINCIDENT
C   LNINTS=5 POINTS J AND L ARE COINCIDENT
C   LNINTS=6 I-J LIES TO RIGHT OF K-L
C   LNINTS=7 I-J LIES ABOVE K-L
C   LNINTS=8 I-J LIES TO LEFT OF K-L
C   LNINTS=9 I-J LIES BELOW K-L
C   LNINTS=10 X INTERCEPT IS RIGHT OF I-J
C   LNINTS=11 X INTERCEPT IS LEFT OF I-J
C   LNINTS=12 X INTERCEPT IS RIGHT OF K-L
C   LNINTS=13 X INTERCEPT IS LEFT OF K-L
C
C   ROUTINES REQUIRED:
C
C   ANSI FORTRAN MAXIMUM AND MINIMUM FUNCTIONS AMAX1 AND AMIN1
C
C   ERRORS:  NONE
C
C   INTERNAL VARIABLES:
C
C   DXIJ,DYIJ,DXKL,DYKL= THE REAL VECTOR COMPONENTS OF THE
C      TWO LINES
C   CRSPRD= THE REAL CROSS PRODUCT OF THE TWO LINES AS VECTORS
```

```
C  XMNIJ,XMXIJ,YMNIJ,YMXIJ,XMKL,XMXKL,YMNKL,YMXKL= THE MAXIMUM
C      (AND MINIMUM) OF EACH LINE COORDINATE PAIR
C  X,Y= THE COORDINATES AT WHICH THE LINES INTERCEPT
C
C        ************ USER CAVEAT ************
C
C  ALL COMPARES, ESPECIALLY THE EQUALITY TYPES ARE MADE
C  ASSUMING THAT REAL ARITHMETIC IS INFINITELY CORRECT.
C  IF ONE DESIRES TO INCLUDE A FUZZ FACTOR, COMPLETE RECODING
C  WILL BE REQUIRED.
C
C
      REAL*4 XI, YI, XJ, YJ, XK, YK, XL, YL, DXIJ,
     #     DYIJ, DXKL, DYKL, CRSPRD
      REAL*4 XMNIJ, XMXIJ, YMNIJ, YMXIJ, XMNKL, XMXKL,
     #     YMNKL, YMXKL, X
C
C
      LNINTS = 0
      DXIJ = XJ - XI
      DYIJ = YJ - YI
      DXKL = XK - XL
      DYKL = YK - YL
      IF ((DXIJ .EQ. 0.) .AND. (DYIJ .EQ. 0.))
     #      RETURN
C
      IF ((DXKL .EQ. 0.) .AND. (DYKL .EQ. 0.))
     #      RETURN
C
      LNINTS = LNINTS + 1
      CRSPRD = DXIJ * DYKL - DYIJ * DXKL
      IF (CRSPRD .EQ. 0.) RETURN
      LNINTS = LNINTS + 1
      IF ((XI .EQ. XK) .AND. (YI .EQ. YK))
     #      RETURN
      LNINTS = LNINTS + 1
      IF ((XI .EQ. XL) .AND. (YI .EQ. YL))
     #      RETURN
      LNINTS = LNINTS + 1
      IF ((XJ .EQ. XK) .AND. (YJ .EQ. YK))
     #      RETURN
      LNINTS = LNINTS + 1
      IF ((XJ .EQ. XL) .AND. (YJ .EQ. YL))
     #      RETURN
      LNINTS = LNINTS + 1
      XMNIJ = AMIN1(XI,XJ)
      XMXIJ = AMAX1(XI,XJ)
      YMNIJ = AMIN1(YI,YJ)
      YMXIJ = AMAX1(YI,YJ)
      XMNKL = AMIN1(XK,XL)
      XMXKL = AMAX1(XK,XL)
      YMNKL = AMIN1(YK,YL)
      YMXKL = AMAX1(YK,YL)
      IF (XMNIJ .GT. XMXKL) RETURN
      LNINTS = LNINTS + 1
      IF (YMNIJ .GT. YMXKL) RETURN
      LNINTS = LNINTS + 1
      IF (XMXIJ .LT. XMNKL) RETURN
      LNINTS = LNINTS + 1
      IF (YMXIJ .LT. YMNKL) RETURN
      LNINTS = LNINTS + 1
      X = (DXKL*(XJ*YI - XI*YJ) - DXIJ*(XK*YL - XL*
     #YK)) / CRSPRD
      IF (X .GT. XMXIJ) RETURN
      LNINTS = LNINTS + 1
      IF (X .LT. XMNIJ) RETURN
      LNINTS = LNINTS + 1
      IF (X .GT. XMXKL) RETURN
      LNINTS = LNINTS + 1
      IF (X .LT. XMNKL) RETURN
C
      LNINTS = -1
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE SMPCLP(X, Y, N, IJS, KLS, LTST)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO CHECK AN ORDERED SET OF DYADS FOR
C  THE DEFINITION OF A SIMPLE CLOSED POLYGON.
C
C  INPUT:
C
C  X,Y= REAL ARRAYS(N) WHICH CONTAIN ORDERED COORDINATE PAIRS
C     THAT ARE TO DEFINE A SIMPLE CLOSED POLYGON WHOSE FINAL
C     SIDE IS POINT N CONNECTED TO POINT 1. IE CLOSURE.
C  N= THE DIMENSION OF INPUT. MUST BE GREATER OR EQUAL TO 3.
C
C  OUTPUT:
C
C  INPUT IS UNCHANGED.
C
C  LTST, IJS, KLS= INTEGER FLAGS AS A RESULT OF SIMPLE
C     CLOSURE TESTING.
C     LSTS = 0
C     LSTS = 1 POLYGON IS NOT SIMPLE CLOSED
C        WHEN POLYGON IS NOT SIMPLE CLOSED THEN FLAGS IJS AND
C        KLS MAY BE TESTED TO ASCERTAIN THE FIRST CAUSE OF
C        NON-CLOSURE.
C     IJS = KLS = 0 NO POLYGON IS POSSIBLE AS N LE 2
C     IJS + 1 = KLS EITHER POINTS IJS AND KLS ARE
C        COINCIDENT OR SIDE IJS TO KLS FOLDED
C        BACK ON SIDE IJS - 1 TO IJS.
C        OTHERWISE SIDE KLS-1 TO KLS INTERSECTED
C        OR WAS COINCIDENT WITH A TERMINUS OF
C        SIDE IJS-1 TO IJS.
C        NOTE: IF IJS OR KLS EQUALS ZERO THEN IJS-1 OR
C           KLS-1 EQUALS N.
C
C  ERRORS: NONE
C
C  ROUTINES REQUIRED:
C
C  INTEGER*2 FUNCTION LNINTS= A FUNCTION WHICH TESTS FOR LINE
C     INTERSECTION.
C
       REAL*4 X(*), Y(*), XK, YK, XJ, YJ, XI, YI, XL, YL
       REAL*4 XJLXK, XJLXI, YJLYK, YJLYI
       INTEGER*2 N, IJS, KLS, LTST, LNINTS, L, NL2, J, JP2
C
C
C      INTERNAL VARIABLES:
C         XK,YK,XJ,YJ,XI,YI,XL,YL - TEMPORARY REAL VARIABLES
```

```
C          XJLXK,XJLXI,YJLYK,YJLYI - TEMPORARY REAL VARIABLES
C
C          L,J - INDEXING INTEGER VARIABLES
C
C          NLP2,JP2 - DO LOOP CONTROL INTEGERS
C
    IJS = 0
    KLS = 0
C
C     TEST THE DIMENSION N
C
    IF (N .LE. 2) GO TO 70
C
C CHECK THE CONNECTED PAIRS TO ENSURE THAT THEY ARE NOT
C REVERSED AND OVERLAY EACH OTHER OR THAT SIDES DEGENERATE TO
C A POINT.
C
    YI = Y(N)
    XI = X(N)
    XJ = X(1)
    YJ = Y(1)
C
    DO 30 L = 2, N
      XK = X(L)
      YK = Y(L)
      XJLXK = XJ - XK
      YJLYK = YJ - YK
      XJLXI = XJ - XI
      YJLYI = YJ - YI
      IF (XJLXK .EQ. 0.0 .AND. YJLYK .EQ. 0.0)
     #    GO TO 10
      IF (XJLXI .EQ. 0.0 .AND. YJLYI .EQ. 0.0)
     #    GO TO 10
      IF (ATAN2(XJLXK,YJLYK) .NE. ATAN2(XJLXI,
     #    YJLYI)) GO TO 20
   10  KLS = L
      IJS = L - 1
      GO TO 70
   20  XI = XJ
      YI = YJ
      XJ = XK
      YJ = YK
   30 CONTINUE
C
    IF (N .EQ. 3) GO TO 80
C
C THERE ARE MORE THAN 3 SIDES THEREFORE WE MUST NOW TEST THE
C REMAINING TO WHICH IT IS CONNECTED.  WE LOOP THROUGH THE SIDES
C TESTING FOR INTERSECTION OR COINCIDENCE.  SET STARTING LINE
C SEGMENT POINTS TEMPORARY VARIABLES AND SWITCH AS LOOPS
```

```
C PROCEDE. WE USE TEMPORARY VARIABLES THROUGHOUT TO MINIMIZE
C INDEXING.
C
    NL2 = N - 2
    XI = X(N)
    YI = Y(N)
C
    DO 60 J = 1, NL2
      XJ = X(J)
      YJ = Y(J)
      XK = X(J + 1)
      YK = Y(J + 1)
      JP2 = J + 2
C
      DO 50 L = JP2, N
        IF (J .EQ. 1 .AND. L .EQ. N)
   #       GO TO 50
        XL = X(L)
        YL = Y(L)
        LTST = LNINTS(XI,YI,XJ,YJ,XK,YK,XL,YL)
        IF (LTST .GT. 5) GO TO 40
        IF (LTST .EQ. 1) GO TO 40
        LJS = J
        KLS = L
        GO TO 70
  40     XK = XL
        YK = YL
  50   CONTINUE
C
      XI = XJ
      YI = YJ
  60 CONTINUE
C
    GO TO 80
C
C   POLYGON IS NOT SIMPLE CLOSED
C
  70 LTST = 1
    RETURN
C
C   POLYGON IS SIMPLE CLOSED
C
  80 LTST = 0
    RETURN
    END
```

```
      SUBROUTINE BISECT(FCN,COEF,N,X1,X2,XR,FR,RATIO,NLIM,NIT)
C
C  PURPOSE:
C
C  THIS SUBROUTINE USES THE NUMERICAL PROCEDURE CALLED THE
C  BISECTION METHOD (OR INTERVAL HALVING) TO DETERMINE A ROOT
C  OF F(X)=0 OF A CONTINUOUS FUNCTION. THE BISECTION METHOD
C  REQUIRES THAT STARTING VALUES ARE OBTAINED BEFORE THE
C  METHOD CAN BEGIN.
C
C  INPUT VARIABLES:
C
C  FCN= REAL*4 VARIABLE CONTAINING THE FUNCTION BEING
C  EVALUATED FOR ITS ROOT. THE CALLING PROGRAM WILL DECLARE
C  THIS FUNCTION AS AN EXTERNAL FUNCTION SO THAT IT CAN BE
C  PASSED
C  COEF= REAL*4 ARRAY OF A LENGTH DECLARED IN THE CALLING
C     PROGRAM. THIS ARRAY CONTAINS THE COEFFICIENTS OF THE
C     POLYNOMIAL
C N=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF COEFFICIENTS IN
C     THE POLYNOMIAL
C  X1,X2= REAL*4 VARIABLES CONTAINING THE STARTING VALUES
C  RATIO= REAL*4 VARIABLE CONTAINING THE DIFFERENCE RATIO
C     TOLERANCE
C  NLIM= INTEGER*2 VARIABLE CONTAINING THE MAXIMUM NUMBER OF
C     ITERATIONS THAT CAN BE CARRIED OUT WHILE SEARCHING FOR
C     THE ROOT.
C
C  OUTPUT:
C
C  INPUT VARIABLES ARE UNCHANGED
C
C  OUTPUT VARIABLES:
C
C  XR= REAL*4 VARIABLE CONTAINING THE ROOT VALUE
C FR= REAL*4 VARIABLE CONTAINING THE FUNCTION VALUE AT THE ROOT
C  NIT= INTEGER*2 VARIABLE CONTAINING THE NUMBER OF ITERATIONS
C     CARRIED OUT IN THIS ROUTINE
C
C  ERRORS:
C
C THE FOLLOWING ERROR DESCRIPTIONS MAY BE WRITTEN TO THE
C SCREEN DURING PROGRAM EXECUTION:
C
C      1. BISECTION ERROR: X1 AND X2 CONTAIN THE SAME VALUE
C      2. BISECTION ERROR: ASSIGNED RELATIVE ERROR TOO SMALL
C      3. BISECTION ERROR: F(X1) AND F(X2) HAVE SAME SIGN
C      4. BISECTION ERROR: NLIM MUST BE > ZERO
C      5. BISECTION ERROR: TOLERANCE NOT MET
C         ## ITERATIONS CARRIED OUT
```

```
C
C   SUBROUTINES REQUIRED: NONE
C
C   FUNCTIONS REQUIRED:
C
C FCN=REAL*4 FUNCTION (EXTERNAL) WHICH CALCULATES THE VALUE OF
C       THE CONTINUOUS FUNCTION AT A GIVEN VALUE OF X.
C
C
     REAL*4 FCN, X1, X2, XR, RATIO, F1, F2, FR, XLAST, COEF(*)
     INTEGER*2 NLIM, J, NIT, N
C
C--CHECK IF X1 AND X2 ARE EQUAL TO EACH OTHER
C
     IF(X1.EQ.X2)THEN
    WRITE(*,*)'BISECTION ERROR: X1 AND X2 CONTAIN THE SAME VALUE'
       RETURN
     ENDIF
C
C--CHECK IF RATIO IS LESS THAN OR EQUAL TO ZERO
C
     IF(RATIO.LT.EPSILON(RATIO))THEN
    WRITE(*,*)'BISECTION ERROR: ASSIGNED RELATIVE ERROR TOO SMALL'
       RETURN
     ENDIF
C
C--CHECK IF FUNCTION HAS THE SAME SIGN AT X1 & X2
C
     F1 = FCN(X1,COEF,N)
     F2 = FCN(X2,COEF,N)
     IF (F1*F2 .GT. 0.) THEN
       WRITE(*,*)'BISECTION ERROR: F(X1) AND F(X2) HAVE SAME SIGN'
       RETURN
     END IF
C
C--CHECK IF NUMBER OF MAXIMUM ITERATIONS IS GREATER THAN ZERO
C
     IF (NLIM.LE.0) THEN
       WRITE(*,*)'BISECTION ERROR: NLIM MUST BE > ZERO'
       RETURN
     END IF
C
C--NOW BISECT THE INTERVAL UP TO A MAXIMUM OF NLIM TIMES TO
C  DETERMINE THE ROOT
C
     IF(X1.NE.0.)THEN
       XLAST=X1
     ELSE
       XLAST=0.000001
     ENDIF
```

```
      DO 10 J = 1, NLIM
        XR = (X1 + X2) / 2.
        FR = FCN(XR,COEF,N)
        IF (ABS((XLAST-XR)/XLAST) .LE. RATIO) THEN
          NIT=J
          RETURN
        ENDIF
        IF (FR*F1 .GT. 0.) THEN
          XLAST=XR
          X1 = XR
        ELSE
          XLAST=XR
          X2 = XR
        ENDIF
   10 CONTINUE
C
C--CONVERGENCE DOES NOT OCCUR IN NLIM ITERATIONS
C
      WRITE(*,*)'BISECTION ERROR: TOLERANCE NOT MET '
      WRITE(*,*)NLIM,' ITERATIONS CARRIED OUT'
      RETURN
      END
      SUBROUTINE PLOTS(0,IOPORT,MODEL)
C
C  PURPOSE:
C
C THIS PLOT88 SUBROUTINE INITIALIZES THE PLOT88 SOFTWARE.IT MUST
C  BE CALLED BEFORE ANY OTHER PLOT88 SUBROUTINES ARE CALLED.
CPLOTS DEFINES THE DEVICE SPECIFIC DEFAULT PARAMETERS BASED ON
C  THE DEVICE TYPE AND SETS EACH NON DEVICE SPECIFIC PARAMETERS
C  TO ITS VALUE.
C
C  VARIABLES:
C
C 0=INTEGER*2 VARIABLE WHICH IS ALWAYS SET EQUAL TO ZERO AND IS
C      INCLUDED FOR COMPATIBILITY
C IOPORT = INTEGER*2 VARIABLE CONTAINING THE HARDWARE INTERFACE
C      TYPE.
C                 IOPORT=0,  PRN: (EQUIVALENT TO LPT1:)
C                 IOPORT=97, EGA: IBM ENHANCED GRAPHICS ADAPTER
C                 IOPORT=99, CGA: IBM COLOUR GRAPHICS ADAPTER
C                 IOPORT=9600, COM1:9600,E,7,1
C
C  MODEL = INTEGER*2 VARIABLE CONTAINING THE OUTPUT DEVICE
C      IDENTIFICATION
C
C           MODEL=0, EPSON FX-80 PRINTER, SINGLE DENSITY
C           MODEL=30, HP 7475A GRAPHICS PLOTTER
C           MODEL=42, EPSON LQ1500, DOUBLE SPEED, DUAL DENSITY
C           MODEL=97, ENHANCED GRAPHICS ADAPTER (EGA)
C           MODEL=99, COLOUR GRAPHICS ADAPTER (CGA)
C
      RETURN
      END
C------------------------------------------------------------
      SUBROUTINE FACTOR(FACT)
C
C  PURPOSE:
C
C  THIS PLOT88 SUBROUTINE IS USED TO ENLARGE OR REDUCE THE SIZE
C  OF THE ENTIRE DRAWING
C
C  VARIABLE:
C
C  FACT = REAL*4 VARIABLE CONTAINING THE RATIO OF THE DESIRED
C       DRAWING SIZE TO THE NORMAL DRAWING SIZE
C
      RETURN
      END
C------------------------------------------------------------
      SUBROUTINE WINDOW(XMIN,YMIN,XMAX,YMAX)
C
```

```
C PURPOSE:
C
C THIS PLOT88 SUBROUTINE DEFINES THE WINDOW IN WHICH THE
C DRAWING WILL BE MADE.
C
C VARIABLES:
C
C XMIN = REAL*4 VARIABLE CONTAINING THE MINIMUM X AXIS VALUE
C YMIN = REAL*4 VARIABLE CONTAINING THE MINIMUM Y AXIS VALUE
C XMAX = REAL*4 VARIABLE CONTAINING THE MAXIMUM X AXIS VALUE
C YMAX = REAL*4 VARIABLE CONTAINING THE MAXIMUM Y AXIS VALUE
C
   RETURN
   END
C------------------------------------------------------------
   SUBROUTINE PLOT(X,Y,IPEN)
C
C PURPOSE:
C
C THIS PLOT88 SUBROUTINE MOVES THE PEN FROM THE CURRENT
C POSITION TO A NEW POSITION, EITHER BY DRAWING A STRAIGHT LINE
C TO THE NEW POSITION (PEN DOWN) OR BY MOVING TO THE NEW
C POSITION WITH THE PEN UP.
C
C VARIABLES:
C
C X,Y=REAL*4 VARIABLES CONTAINING THE POSITION TO WHICH THE PEN
C    WILL BE MOVED.  (X,Y) ARE RELATIVE TO THE EXISTING ORIGIN.
C IPEN=INTEGER*2 VARIABLE CONTAINING THE PEN STATE DURING MOVE.
C           IPEN=1, MOVE/DRAW TO (X,Y) USING THE LAST PEN STATE
C      IPEN=2, DRAW SOLID LINE TO (X,Y)
C           IPEN=3, MOVE TO (X,Y), NO LINE DRAWN
C           IPEN=999, TERMINATE THIS DRAWING. TERMINATE PLOT88
C                         LIBRARY USEAGE.
   RETURN
   END
C------------------------------------------------------------
   SUBROUTINE LINE(X,Y,N,INC,LINTYP,INTEQ)
C
C PURPOSE:
C
C THIS PLOT88 SUBROUTINE DRAWS A LINE BASED ON (X,Y) POINTS
C USING THE OFFSET AND SCALING VALUES THAT WERE DETERMINED
C USING THE SCALE SUBROUTINE.
C
C VARIABLES:
C
C X,Y = REAL*4 VECTORS CONTAINING THE UNSCALED DATA VALUES FOR
C    X AND Y AXES.  THE AXIS VALUES 'FIRST' AND 'DELTA' VALUES
C    MUST BE STORED AFTER THE LAST ELEMENT OF EACH VECTOR.
```

```
C  N=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF DATA POINTS IN
C     EACH VECTOR.  THIS NUMBER SHOULD NOT INCLUDE THE EXTRA
C     LOCATIONS FOR THE TWO SCALING PARAMETERS.
C  INC = INTEGER*2 VARIABLE CONTAINING THE INCREMENT USED IN
C     INDEXING THE TWO VECTORS.
C  LINTYP = INTEGER*2 VARIABLE USED TO DESCRIBE THE LINE TYPE TO
C     BE DRAWN.
C                    LINTYP=0, DATA POINTS CONNECTED WITH STRAIGHT
C                   LINES WITHOUT SYMBOLS DRAWN
C  INTEQ = INTEGER*2 VARIABLE CONTAINING THE INDEX VALUE OF THE
C     SYMBOL WHICH MAY BE DRAWN.
C
      RETURN
      END
C-----------------------------------------------------------
      SUBROUTINE FILL(XVERT,YVERT,NVERT)
C
C  PURPOSE:
C
C  THIS PLOT88 SUBROUTINE PROVIDES SHADING OF USER SPECIFIED
C  POLYGONAL AREAS.
C
C  VARIABLES:
C
C  XVERT,YVERT = REAL*4 VECTORS CONTAINING THE (X,Y) COORDINATES
C     OF EACH VERTICE OF THE POLYGON TO FILL
C  NVERT = INTEGER*2 VARIABLE CONTAINING THE NUMBER OF VERTICES
C     DEFINING THE POLYGON
C
      RETURN
      END
C-----------------------------------------------------------
      SUBROUTINE STFILL(NFILL)
C
C  PURPOSE:
C
C  THIS PLOT88 SUBROUTINE IS USED TO SELECT THE SHADING PATTERN
C  TO USE IN SUBSEQUENT CALLS TO THE FILL SUBROUTINE.
C
C  VARIABLE:
C
C  NFILL = INTEGER*2 VARIABLE CONTAINING THE INDEX NUMBER OF A
C     SPECIFIED SHADING PATTERN TO USE
C
      RETURN
      END
C-----------------------------------------------------------
      SUBROUTINE SYMBOL(X,Y,HEIGHT,CTEXT,ANGLE,NC)
C
C  PURPOSE:
```

```
C
C  THIS PLOT88 SUBROUTINE IS USED TO DRAW CHARACTER STRINGS AND
C  SPECIAL CENTERED SYMBOLS
C
C  VARIABLES:
C
C  X,Y=REAL*4 VARIABLES CONTAINING THE COORDINATES OF THE LOWER
C       LEFT HAND CORNER OF THE FIRST CHARACTER TO BE DRAWN
C  HEIGHT = REAL*4 VARIABLE CONTAINING THE HEIGHT OF EACH
C       CHARACTER IN INCHES
C  CTEXT = CHARACTER STRING TO BE DRAWN
C  ANGLE=REAL*4 VARIABLE CONTAINING THE ANGLE IN DEGREES ABOUT
C       THE X AXIS AT WHICH THE TEXT IS DRAWN
C  NC=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF CHARACTERS IN
C       CTEXT TO BE DRAWN
C
   RETURN
   END
C-----------------------------------------------------------
   SUBROUTINE NUMBER(X,Y,HEIGHT,FPN,ANGLE,NDEC)
C
C  PURPOSE:
C
C  THIS PLOT88 SUBROUTINE IS USED TO CONVERT A SINGLE PRECISION
C  REAL NUMBER TO ITS DECIMAL EQUIVALENT AS A CHARACTER STRING
C  AND TO DRAW THE STRING USING SUBROUTINE SYMBOL.
C
C  VARIABLES:
C
C  X,Y=REAL*4 VARIABLES CONTAINING THE COORDINATES OF THE LOWER
C       LEFT HAND CORNER OF THE FIRST CHARACTER TO BE DRAWN
C  HEIGHT = REAL*4 VARIABLE CONTAINING THE HEIGHT OF EACH
C       CHARACTER IN INCHES
C  FPN = FLOATING POINT NUMBER TO BE DRAWN
C  ANGLE=REAL*4 VARIABLE CONTAINING THE ANGLE IN DEGREES ABOUT
C       THE X AXIS AT WHICH THE TEXT IS DRAWN
C  NDEC = INTEGER*2 VARIABLE CONTAINING THE NUMBER OF DIGITS TO
C        THE RIGHT OF THE DECIMAL POINT TO BE DRAWN
C
   RETURN
   END
C-----------------------------------------------------------
   SUBROUTINE STDASH(XLINE,XSPACE)
C
C  PURPOSE:
C
C  THIS PLOT88 SUBROUTINE IS USED TO DEFINE THE CHARACTERISTICS
C  OF A DASHED LINE.
C
C  VARIABLES:
```

```
C
C XLINE = REAL*4 VARIABLE CONTAINING THE LENGTH OF THE DASH IN
C      INCHES
C XSPACE=REAL*4 VARIABLE CONTAINING THE LENGTH IN INCHES OF THE
C      GAP BETWEEN LINE SEGMENTS OF THE DASHED LINE.
C
   RETURN
   END
C-----------------------------------------------------------
   SUBROUTINE PLOTD(X,Y,IPEN)
C
C PURPOSE:
C
C THIS PLOT88 SUBROUTINE PRODUCES DASHED LINES USING THE DASH
C SPECIFICATIONS PRESET BY SUBROUTINE STDASH.
C
C VARIABLES:
C
C X,Y = REAL*4 VARIABLES CONTAINING THE COORDINATES OF THE (X,Y)
C    LOCATIONS WHERE THE NEW POSITION WILL BE
C IPEN = INTEGER*2 VARIABLE CONTAINING THE PEN STATE
C                    IPEN=2, DRAW DASHED LINE TO (X,Y)
C                    IPEN=3, MOVE TO (X,Y) WITH PEN UP
C
   RETURN
   END
C-----------------------------------------------------------
   SUBROUTINE LOCATE(PAGE, ROW, COL)
C
C PURPOSE:
C
C THIS FORTRAN UTILITIES SUBROUTINE POSITIONS THE CURSOR ON
C THE SELECTED SCREEN PAGE
C
C VARIABLES:
C
C PAGE = INTEGER*2 VARIABLE CONTAINING THE TEXT PAGE
C ROW = INTEGER*2 VARIABLE CONTAINING THE SCREEN LINE NUMBER
C    WHERE YOU WANT TO PLACE THE CURSOR. THE LINES ARE
C    NUMBERED 0 TO 24, FROM TOP TO BOTTOM.
C COL = INTEGER*2 VARIABLE CONTAINING THE SCREEN COLUMN NUMBER
C    WHERE YOU WANT TO PLACE THE CURSOR. THE COLUMNS ARE
C    NUMBERED 0 TO 79 FROM LEFT TO RIGHT.
C
   RETURN
   END
C-----------------------------------------------------------
   SUBROUTINE SPAGE(PAGE)
C
C PURPOSE:
```

```
C
C THIS FORTRAN UTILITIES SUBROUTINE SELECTS ONE SCREEN PAGE
C FOR DISPLAYING
C
C VARIABLES:
C
C PAGE = INTEGER*2 VARIABLE CONTAINING THE TEXT PAGE
C
   RETURN
   END
C-----------------------------------------------------------
   SUBROUTINE SCCHAR(PAGE, CH, ATTRIB)
C
C PURPOSE:
C
C THIS FORTRAN UTILITIES SUBROUTINE RETURNS THE CHARACTER
C AND ITS ATTRIBUTE ON THE SPECIFIED PAGE AT CURSOR POSITION
C
C VARIABLES:
C
C PAGE = INTEGER*2 VARIABLE CONTAINING THE TEXT PAGE
C CH = CHARACTER*1 VARIABLE CONTAINING THE CHARACTER WHICH
C     IS RETURNED AFTER BEING READ AT THE CURSOR POSITION
C ATTRIB = INTEGER*2 VARIABLE CONTAINING THE COLOR ATTRIBUTE
C     FOR THE CHARACTER
C
   RETURN
   END
C-----------------------------------------------------------
   SUBROUTINE PUTCHA(PAGE, ATTRIB, COUNT, CH)
C
C PURPOSE:
C
C THIS FORTRAN UTILITIES SUBROUTINE WRITES A CHARACTER
C AND ITS ATTRIBUTE AT THE CURRENT CURSOR POSITION
C
C VARIABLES:
C
C PAGE = INTEGER*2 VARIABLE CONTAINING THE TEXT PAGE
C CH = CHARACTER*1 VARIABLE CONTAINING THE CHARACTER WHICH
C     IS TO BE WRITTEN AT THE CURSOR POSITION
C COUNT = INTEGER*2 VARIABLE CONTAINING THE NUMBER OF
C     CHARACTERS TO WRITE
C ATTRIB = INTEGER*2 VARIABLE CONTAINING THE COLOR ATTRIBUTE
C     FOR THE CHARACTER
C     IF ATTRIB=2, NORMAL DISPLAY
C     IF ATTRIB=120, REVERSE VIDEO
C     IF ATTRIB=130, BLINKING NORMAL
C     IF ATTRIB=248, BLINKING REVERSE VIDEO
C
```

```
      RETURN
      END
C-----------------------------------------------------------
      SUBROUTINE SCREEN(MODE)
C
C  PURPOSE:
C
C  THIS FORTRAN UTILITIES SUBROUTINE SETS THE SCREEN MODE AND
C  AND CLEARS THE SCREEN
C
C  VARIABLES:
C
C  MODE  =  INTEGER*2 VARIABLE CONTAINING A NUMERIC EXPRESSION IN
C       THE RANGE 0 TO 6 FOR THE COLOR/GRAPHICS ADAPTER .
C    IF MODE=2, SCREEN IS SET AS AN 80x25 BW DISPLAY WITH 4 TEXT
C              PAGES LABELED AS 0 TO 3.
C    IF MODE=3, SCREEN IS SET AS AN 80x25 COLOR DISPLAY WITH 4
C              TEXT PAGES LABELED AS 0 TO 3.
C
      RETURN
      END
C-----------------------------------------------------------
      SUBROUTINE NEWCUR(START, STOP)
C
C  PURPOSE:
C
C  THIS FORTRAN UTILITIES SUBROUTINE DEFINES THE SIZE AND SHAPE
C  OF THE TEXT CURSOR.
C
C  VARIABLES:
C
C  START=INTEGER*2 VARIABLE CONTAINING A NUMERIC EXPRESSION IN
C       THE RANGE 0 TO 32.  IT INDICATES THE CURSOR STARTING SCAN
C       LINE.
C  STOP  =  INTEGER*2 VARIABLE CONTAINING A NUMERIC EXPRESSION IN
C       THE RANGE 0 TO 31.  IT INDICATES THE CURSOR STOP SCAN
C       LINE.
C
C  THE SCAN LINES ARE NUMBERED FROM 0 AT THE TOP OF THE
C  POSITION.  THE FOLLOWING EXAMPLE MAKES THE CURSOR INVISIBLE
C
C      CALL NEWCUR(32,0)
C
      RETURN
      END
C-----------------------------------------------------------
      SUBROUTINE PUTST(PAGE, ATTRIB, LENGTH, STRING)
C
C  PURPOSE:
C
```

```
C  THIS FORTRAN UTILITIES SUBROUTINE WRITES A STRING
C  AND ITS ATTRIBUTE AT THE CURRENT CURSOR POSITION
C
C  VARIABLES:
C
C  PAGE = INTEGER*2 VARIABLE CONTAINING THE TEXT PAGE
C  STRING = CHARACTER STRING CONTAINING THE CHARACTERS WHICH
C      ARE TO BE WRITTEN AT THE CURSOR POSITION
C  LENGTH = INTEGER*2 VARIABLE CONTAINING THE NUMBER OF
C      CHARACTERS IN THE STRING TO BE WRITTEN
C  ATTRIB = INTEGER*2 VARIABLE CONTAINING THE COLOR ATTRIBUTE
C      FOR THE CHARACTER AS DESCRIBED IN THE PUTCHA SUBROUTINE
C
   RETURN
   END
C-----------------------------------------------------------
   INTEGER*2 FUNCTION INKEY(CH,SCAN)
C
C  PURPOSE:
C
C  THIS FORTRAN UTILITIES FUNCTION RETURNS WHETHER A KEY IS
C  PRESSED FROM THE KEYBOARD AND WHAT IT IS
C
C  VARIABLES:
C
C  INKEY= RETURNS 1 IF AT LEAST ONE KEY IS PRESSED, OTHERWISE 0
C  CH = CHARACTER*1 STRING CONTAINING THE ASCII CHARACTER READ
C      FROM THE KEYBOARD. A NULL CHARACTER INDICATES A SPECIAL
C      KEY SUCH AS A FUNCTION KEY.
C SCAN=INTEGER*2 VARIABLE CONTAINING THE EXTENDED CODE OF A KEY
C
   RETURN
   END
```

Appendix B - Sample Data Files for DRAG

The following are sample input files for the computer program DRAG. DRAG can call four different subroutines and each of them requires a separate data file to initialize the variable names and values which are displayed on the input menu. Each of these different subroutines (SEAM1, REH1, SEAM2 and REH2) represents a different model for dragline analysis. The initial variable values read from a data file can be changed during program operation due to the interactive input menus. This allows many different combinations of variable values to be tested during the execution of DRAG.

Each default data file name has a name similar to its calling subroutine. For example, the default data file for subroutine SEAM1 is SEAM1.DAT. If the default data file does not exist, the user is prompted for another file name.

The data files are read in list directed format. Line 1 of each data file contains a variable name character string, followed by its corresponding value in line 2. This sequence is repeated for all of the remaining variables.

Subroutine READIN is used to read in the data files for all of the different models. The end of file is used to signal the end of data. Only two error checks are made in READIN. The first check is to make sure that the number of variable names read in is the same as that of the variable values. The second check is to make sure that the maximum number of variables (24) is not exceeded for the screen display. The exact number of variables is checked in the model subroutine (ie. SEAM1).

Input file for subroutine SEAM1 - filename: SEAM1.DAT

```
CUT WIDTH  (m)
50.
SPOIL PIT WIDTH  (m)
50.
CHOPCUT  WIDTH  (m)
50.
O/B THICKNESS  (m)
15.
CHOPCUT  THICKNESS  (m)
5.
COAL  SEAM  THICKNESS  (m)
2.0
SPOIL PILE  ANGLE  (deg)
35.
O/B HIGHWALL   ANGLE  (deg)
70.
O/B SWELL  FACTOR
1.20
O/B POSITIONING   FACTOR
0.50
TUB DIAMETER   (m)
30.
```

Input file for subroutine REH1 - filename: REH1.DAT

```
CUT WIDTH (m)
50.
SPOIL PIT WIDTH (m)
50.
CHOPCUT WIDTH (m)
50.
O/B THICKNESS (m)
15.
CHOPCUT THICKNESS (m)
5.
COAL SEAM THICKNESS (m)
2.0
SPOIL PILE ANGLE (deg)
35.
O/B HIGHWALL ANGLE (deg)
70.
O/B SWELL FACTOR
1.20
O/B POSITIONING FACTOR
0.50
TUB DIAMETER (m)
30.
DRAGLINE OP. RADIUS (m)
80
```

Input file for subroutine SEAM2 - filename: SEAM2.DAT

```
O/B CUT WIDTH (m)
50.
I/B CUT WIDTH (m)
50.
SPOIL PIT WIDTH (m)
50.
CHOPCUT WIDTH (m)
50.
SAFETY BENCH WIDTH (m)
6.0
O/B THICKNESS (m)
15.
I/B THICKNESS (m)
10.
CHOPCUT THICKNESS (m)
5.
TOP SEAM THICKNESS (m)
2.0
LOWER SEAM THICKNESS (m)
3.0
SPOIL PILE ANGLE (deg)
35.
O/B HIGHWALL ANGLE (deg)
70.
I/B HIGHWALL ANGLE (deg)
70.
O/B SWELL FACTOR
1.20
I/B SWELL FACTOR
1.20
O/B POSITIONING FACTOR
0.50
I/B POSITIONING FACTOR
0.5
TUB DIAMETER (m)
30.
```

Input file for subroutine REH2 - filename: REH2.DAT

```
O/B CUT WIDTH (m)
60.
I/B CUT WIDTH (m)
60.
SPOIL PIT WIDTH (m)
60.
CHOPCUT WIDTH (m)
0.
SAFETY BENCH WIDTH (m)
10.0
DITCH WIDTH (m)
0.
O/B THICKNESS (m)
15.
I/B THICKNESS (m)
14.
CHOPCUT THICKNESS (m)
0.
TOP SEAM THICKNESS (m)
2.1
LOWER SEAM THICKNESS (m)
3.0
SPOIL PILE ANGLE (deg)
33.
O/B HIGHWALL ANGLE (deg)
45.
I/B HIGHWALL ANGLE (deg)
70.
CUT ANGLE IN SPOIL (deg)
45.
O/B SWELL FACTOR
1.25
I/B SWELL FACTOR
1.25
O/B POSITIONING FACTOR
0.88
I/B POSITIONING FACTOR
0.88
POSITIONING NEAR SPOIL
1.0
TUB DIAMETER (m)
21.3
```

DRAGLINE OP. RADIUS (m)
83.7
MAX. DRAGLINE HEIGHT (m)
40.8

Appendix C - VARSIM Library

This appendix contains the listing of all the routines in the VARSIM library. The purpose of this library is to allow numerical variance and bias component analysis of a user-supplied function. This external function is used to define all of the initial values, names and units of the independent variables within the function. This Fortran function is the only input required. An example of the required function format is given in appendix D. The program was set up in this general manner so that it could be easily applied to different cases. Global and local variables are defined within their respective programs. This appendix is set up in the following manner:

- list and description of COMMON blocks
- an index of routines with page numbers
- a flow chart of the routines showing how they link together
- a discussion of differentiation alternatives
- listing of the subroutines and functions

The COMMON blocks used in this library are:

FUNC - used to pass the function name, function units and variable names from the user function to the main program called VARSM

XVAL - used to pass a real vector from subroutine MNVRCP to function DFCN so that partial derivatives may be obtained (using subroutine DDERIV) when there is more than one independent variable

SWITCH - used to pass an integer constant from MNVRCP to DFCN. This integer defines the vector element to be used in the partial derivative evaluation with respect to the function.

Index of Routines

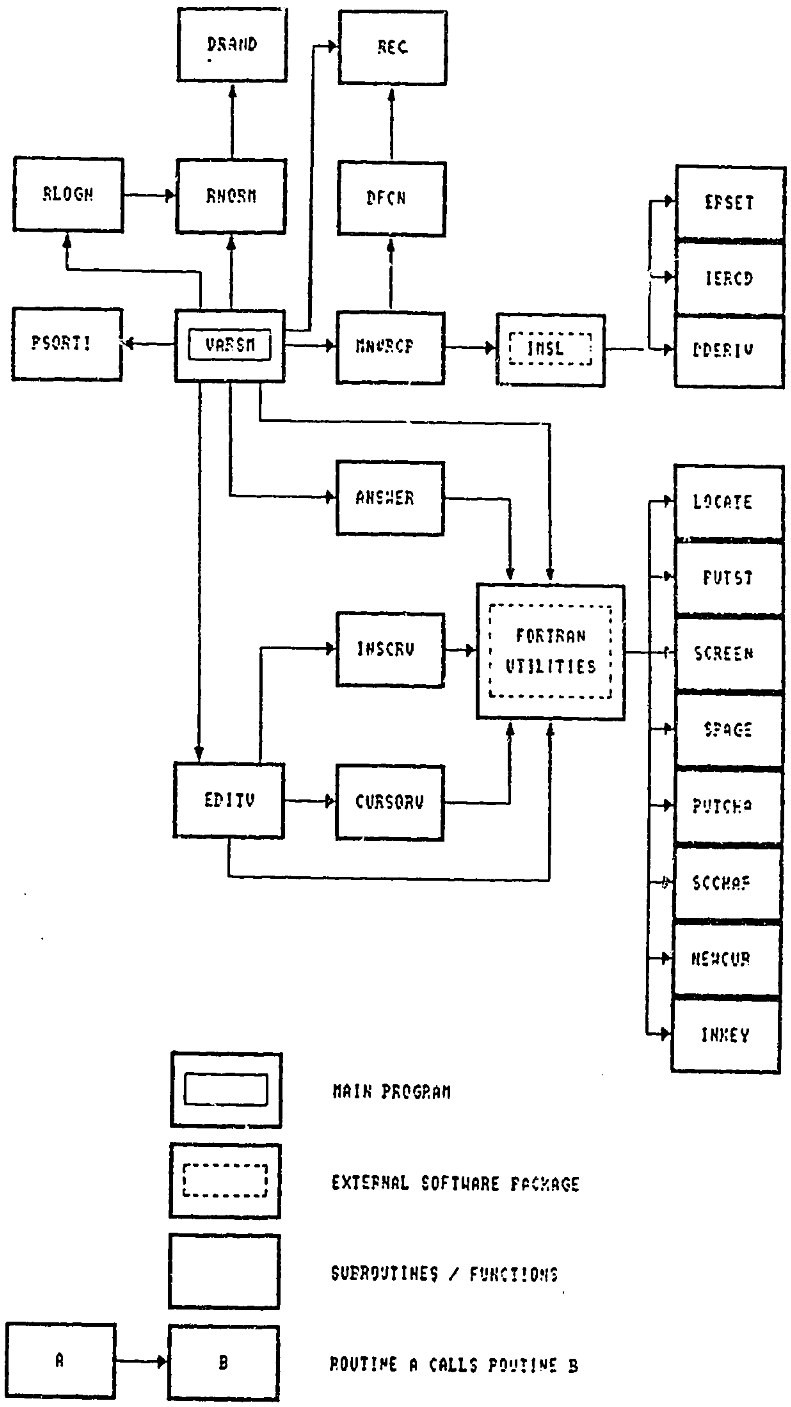


Figure C.1 Flowchart of VARSIM Routines

Several different algorithms can be used to determine derivatives if the user decides to change the subroutine MNVRCP. One method is to use a number of points to estimate the derivative of a function with specific coefficients in a differentiation formula. The coefficients are determined according which derivative is desired and how many points are to be used when estimating the derivative. Abramowitz and Stegun [1] list a table of these coefficients for a differentiation formula. As an example, the desired formula of a second derivative of a function using three point estimation is as follows

$$F''(X) \approx \frac{A_1 \cdot F(X-H) + A_2 \cdot F(X) + A_3 \cdot F(X+H)}{H^2} \qquad \text{(C.1)}$$

where :

$A_1 = 1$
$A_2 = -2$
$A_3 = 1$
H = step size to be used
F = function whose derivative is desired
X = point at which the derivative is to be calculated

Numerical differentiation programs are often documented within reputable texts on applied numerical analysis. When using a three point difference formula as above, values of F(X) are used at the distance H on either side of the point where the derivative is desired. Most differentiation algorithms require an initial step size. The truncation error can be reduced by reducing the value of H, however, roundoff error will become more significant. Rou[illegible]ff error occurs when it is required to subtract function values that are nearly the same [illegible]s. The best accuracy therefore occurs at some intermediate value of H. The best step s[illegible] to use changes when the argument values change within a function. For this reason it is advisable to include the capacity to change the step size to improve accuracy when differentiating. The routine used in VARSIM, IMSL [23] routine

DDERIV, uses finite difference approximations where the initial step size is adaptively changed to enhance accuracy.

```
$STORAGE:2
      PROGRAM VARSM
C
C  PURPOSE:
C
C  TO PROVIDE A GENERAL CODE FOR NUMERICAL VARIANCE AND BIAS
C  COMPONENT ANALYSIS PLUS SIMULATION,OF THE DOUBLE PRECISION
C  FUNCTION "REC(X,N)".
C
C  PROGRAM OPERATION:
C
C     THE NUMBER OF FUNCTION ARGUMENTS "N",THE NAMES OF THESE
C  ARGUMENTS AND THEIR UNITS (FACT(I),I=1,N),THE NAME OF THE
C  FUNCTION RESULT "FRNM", THE FUNHxION RESULT UNITS "FRUNTS",
C  INITIAL VALUES FOR THE FUNCTION ARGUMENTS (X(I),I=1,N) AND AN
C  INITIAL FUNCTION RESULT "R" ARE SET BY THE STATEMENT
C  "R=REC(X,N)". A VARIANCE AND BIAS COMPONENT ANALYSIS TABLE IS
C  NOW WRITTEN TO THE SCREEN AND THE USER IS PROMPTED FOR
C  MODIFICATIONS OF THE ARGUMENTS (X(I),I=1,N) AND THE ESTIMATED
C  STANDARD DEVIATION OF THESE ARGUMENTS (SDX(I),I=1,N). INTERNAL
C  VALUES OF ARGUMENTS ARE STORED IN DOUBLE PRECISION.
C      VALUES MAY BE CHANGED BY MOVING THE HILITED CELL TO THE
C  VALUE TO BE CHANGED AND SIMPLY TYPING IN A NEW VALUE. NEW
C  VALUES ARE CALCULATED FOR THE VARIANCE AND BIAS ANALYSIS
C  TABLE WHENEVER THE SPACE BAR IS PRESSED.
C      AT THE START OF THE PROGRAM'S OPERATION, THE USER IS
C  PROMPTED WHETHER OR NOT ALL USER INTERACTION IS TO BE
C  RECORDED IN A FILE. ONLY A REPLY OF "Y" OR "y" WILL CAUSE
C  RESULTS TO BE WRITTEN TO A FILE. THE DEFAULT WILL BE NO AND
C  IS AUTOMATICALLY SELECTED IF ONLY A CARRIAGE RETURN IS
C  PRESSED AT THE PROMPT.
C      IF AN ERROR OCCURS WHILE THE USER IS ENTERING A VALUE INTO
C  A CELL, THE PREVIOUS VALUE MAY BE RESTORED BY PRESSING
C  FUNCTION KEY F1.
C     THE PROGRAM MAY BE QUIT AT ANY TIME BY PRESSING THE ESCAPE
C  KEY. THE ONLY EXCEPTIONS ARE WHEN THE ANALYSIS TABLE VALUES
C  ARE BEING CALCULATED OR IF THE USER HAS SELECTED TO USE THE
C  SIMULATE OPTION.
C      THE SIMULATE OPTION MAY BE ACTIVATED BY PRESSING THE "S" OR
C  "s" CHARACTERS. IN THIS OPTION, THE USER IS PROMPTED FOR THE
C DECISION TO USE NORMAL OR LOGNORMAL VARIATES.AN ANSWER
C WITH A WORD WHOSE FIRST CHARACTER IS NOT "L" OR "l" WILL CAUSE
C ALL ARGUMENTS TO BE SIMULATED AS NORMAL (GUASSIAN) VARIATES.
C  OTHERWISE THE ARGUMENTS WILL BE TREATED AS LOGNORMAL
C  VARIATES. THE FINAL INPUT PROMPT IS FOR "DSEED ?", WHICH IS THE
C STARTING DOUBLE PRECISION NUMBER FOR PSEUDO RANDOM NUMBER
C  GENERATION. A CARRIAGE RETURN RESPONSE WILL "DSEED" AT ITS
C  INTERNAL VALUE. IF THIS OPTION IS NOT TAKEN THE USER SHOULD
C  INPUT A POSITIVE ODD NUMBER OF AT LEAST 10 DIGITS TO THE RIGHT
C  OF THE DECIMAL. THE PROGRAM THEN GOES THROUGH 1000 MONTE
```

```
C CARLO SAMPLES OF "R=REC(X,N)". THE 1000 SIMULATED VALUES ARE
C PARTIALLY SORTED AND THEN VALUES OF THE POSITIONS 5, 10, 25, 50,
C 950, 975, 990 AND 995 ARE WRITTEN TO THE SCREEN. ANALYSIS OF
C THESE VALUES MAY INDICATE WHETHER OR NOT THE FUNCTION IS
C "SKEWED" TO ONE SIDE OF THE MEAN.
C    THE USER IS PROMPTED FOR A FILE NAME IN WHICH TO STORE
C SIMULATED DATA VALUES. IF A FILE IS REQUESTED, THE SIMULATED
C VALUES ARE WRITTEN TO THE FILE IN DOUBLE PRECISION LIST
C DIRECTED FORMAT. THE PROGRAM THEN RETURNS TO THE INPUT
C SCREEN. OTHER COMBINATIONS OF VALUES MAY BE TRIED OR THE
C USER MAY DECIDE TO TERMINATE PROGRAM EXECUTION.
C
C VARIABLES:
C
C  ISW2 = LOGICAL*2 VARIABLE WHERE:
C       IF ISW2=.TRUE., VALUES ARE WRITTEN TO A FILE
C       IF ISW2=.FALSE., VALUES ARE NOT WRITTEN TO A FILE
C  X(I) = REAL*8 VECTOR CONTAINING THE MEAN VALUE OF ARGUMENT I
C  SDX(I) = REAL*8 VECTOR CONTAINING THE STANDARD DEVIATION OF
C         ARGUMENT I
C  BIAS(I) = REAL*8 VECTOR CONTAINING THE PERCENTAGE BIAS
C        COMPONENT OF ARGUMENT I
C  VAR(I) = REAL*8 VECTOR CONTAINING THE PERCENTAGE VARIANCE
C        COMPONENT OF ARGUMENT I
C  FACT(I)=CHARACTER*28 VECTOR CONTAINING THE NAME OF
C   ARGUMENT I
C  TBIAS = REAL*8 VARIABLE CONTAINING THE TOTAL BIAS VALUE
C  R= REAL*8 VARIABLE CONTAINING THE ESTIMATED FUNCTION VALUE
C  SDR = REAL*8 VARIABLE CONTAINING THE ESTIMATED STANDARD
C        DEVIATION OF THE FUNCTION
C  ISW = LOGICAL*2 VARIABLE WHERE:
C      IF ISW=.TRUE., THE SIMULATE OPTION HAS BEEN SELECTED
C      IF ISW=.FALSE., THE SIMULATE OPTION HAS NOT BEEN SELECTED
C
C ROUTINES REQUIRED:
C
C     DOUBLE PRECISION FUNCTION REC(X,N)
C  A USER WRITTEN ROUTINE WHICH RETURNS A DOUBLE PRECISION
C  RESULT FROM THE PRECISION ARGUMENTS IN THE VECTOR "X". IT IS
C  ASSUMED THAT THE FUNCTION IS DEFINED THROUGHOUT THE
C  PRACTICAL SIMULATION RANGE "X(I)+OR-4*SDX(I),I=1,N" AND IS
C  SMOOTH ENOUGH THAT NUMERICAL DIFFERENTIATION MAY BE
C  CARRIED OUT. IN ADDITION THE CHARACTER*28 STRINGS DEFINING C   THE
ARGUMENT NAMES
C  AND UNITS, THE CHARACTER*15 FUNCTION RETURN NAME, THE
C  CHARACTER*7 FUNCTION RETURN UNITS , INITIAL VALUES FOR
C  "X(I),I=1,N", AND "N" ARE RETURNED ON THE FIRST INVOKATION OF
C "R=REC(X,N)".THE CHARACTER NAMES ARE RETURNED VIA THE
C  COMMON BLOCK "FACT" WHOSE ORDER IS
C  "FRNM","FRUNTS","FACT(I),I=1,20".
```

```
C  THE MAXIMUM NUMBER OF ARGUMENTS IN REC IS 20.
C
C     SUBROUTINE MNVRCP
C  THIS SUBROUTINE ACCEPTS "X(I),SDX(I),I=1,N" AND EXTERNAL
C  FUNCTION "REC", DOES NUMERICAL DIFFERENTIATION AND RETURNS
C  THE ABSOLUTE BIAS AND VARIANCE APPROXIMATIONS FROM SECOND
C  ORDER EXPECTATION TAYLOR EXPANSION UNDER SECOND MOMENT
C  INDEPENDENCE AND NULL HIGHER MOMENT ASSUMPTION. THE
C  METHODS USED ARE DETAILED IN THE ROUTINE.
C
C     FUNCTION DRAND
C  THIS IS A MUTIPLICATIVE CONGRUENTIAL GENERATOR OF SINGLE
C  PRECISION UNIFORMLY DISTRIBUTED (0. TO 1.) PSEUDO RANDOM
C  NUMBERS.
C
C     FUNCTION RNORM
C  THIS FUNCTION RETURNS A DOUBLE PRECISION NORMAL DEVIATE OF
C  MEAN "U" AND STANDARD DEVIATION "S".
C
C     FUNCTION RLOGN
C  THIS FUNCTION RETURNS A DOUBLE PRECISION LOGNORMAL DEVIATE
C  OF MEAN "U" AND STANDARD DEVIATION "S".
C
C     SUBROUTINE PSORTI
C  THIS ROUTINE PARTIAL SORTS A REAL VECTOR.
C
C     SUBROUTINE SCREEN
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO SET THE SCREEN MODE AND CLEAR THE SCREEN
C
C     SUBROUTINE SPAGE
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO SELECT ONE SCREEN PAGE FOR DISPLAYING
C
C     SUBROUTINE LOCATE
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO POSITION THE CURSOR ON THE SELECTED SCREEN PAGE
C
C     SUBROUTINE PUTST
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO WRITE A STRING WITH ATTRIBUTE AT THE CURRENT
C  CURSOR LOCATION
C
C     SUBROUTINE EDITV
C  THIS SUBROUTINE CONTROLS THE INTERACTIVE EDIT SCREEN
C
C     ENTRY ED
C  THIS ENTRY IS LOCATED AT THE END OF SUBROUTINE EDITV AND IS
C  CALLED TO INITIALIZE THE DATA LOCATIONS AND TO WRITE THE
C  VARIABLE NAMES TO THE SCREEN
```

```
C
C      SUBROUTINE ANSWER
C THIS SUBROUTINE WRITES THE CALCULATED VALUES ONTO THE
C SCREEN
C
C      ENTRY ANS
C THIS ENTRY IS LOCATED AT THE END OF SUBROUTINE ANSWER AND IS
C   CALLED TO WRITE THE FUNCTION, TOTAL BIAS AND STANDARD
C   DEVIATION NAMES AND UNITS TO THE SCREEN
C
C
C
    INTEGER*2 NN,IND(8),NF
    PARAMETER(NN=1000,NF=20)
    REAL*8 X(NF),BIAS(NF),SDX(NF),VAR(NF),R,SDR,TBIAS,REC
    REAL*8 XX(NF),SDXX(NF),DSEED,RNORM,RLOGN,RDSEED
    REAL*4 SIM(NN)
    INTEGER*2 N , IER, I, J
    EXTERNAL REC
    CHARACTER*24 FLNM,FLNM2
    CHARACTER*28 FACT(NF)
    CHARACTER*15 FRNM
    CHARACTER*7 FRUNTS
    CHARACTER*1 YES
    CHARACTER*30 EST,TBC,SDV
    LOGICAL*2 LN,ISW,ISW2,EX
    COMMON /FUNC/FRNM,FRUNTS,FACT
    DATA X,BIAS,SDX,VAR,R,SDR,TBIAS,XX,SDXX/123*0.D0/
    DATA IND/5,10,25,50,950,975,990,995/
    DATA TBC/' TOTAL BIAS IN '/
    DATA EST/' ESTIMATED      '/
    DATA SDV/' STANDARD DEV  '/
C
C  THE USER IS PROMPTED WHETHER OR NOT TO WRITE RESULTS TO A
C  FILE
C
    WRITE(*,*)' '
    WRITE(*,*)'WRITE RESULTS TO A FILE ? (Y/N)    <CR> = N'
    READ(*,7)YES
    IF(YES.EQ.'Y'.OR.YES.EQ.'y')THEN
      WRITE(*,*)' '
      WRITE(*,*)'ENTER FILE NAME :'
      READ(*,7)FLNM
  1   INQUIRE(FILE=FLNM,EXIST=EX)
      IF(EX)THEN
        WRITE(*,*)'** FILE ALREADY EXISTS **'
        WRITE(*,*)' '
        WRITE(*,*)'ENTER ANOTHER FILE NAME.   <CR> TO OVERWRITE'
        FLNM2 = '                         '
        READ(*,7)FLNM2
        IF(FLNM2(1:1).NE.' ')THEN
          FLNM=FLNM2
          GOTO 1
        ENDIF
      ENDIF
      OPEN(9,FILE=FLNM)
      ISW2=.TRUE.
    ELSE
      ISW2=.FALSE.
    ENDIF
  4 FORMAT(' #',9X,'FACTOR',18X'MEAN    STD DEVIATION',
   #'       % BIAS  % VAR',/)
  5 FORMAT(I2,2X,A28,2G15.8,2F7.2)
  6 FORMAT(A,A,G14.7,A)
  7 FORMAT(A)
    R=REC(X,N)
 99 CALL SCREEN(2)
    CALL SPAGE(2)
    CALL ED(N,FACT,X,SDX)
    CALL ANS(FRNM,FRUNTS,N)
C
C  ALL VALUES AND SCREEN POSITIONS HAVE BEEN INITIALIZED BY THIS
C  POINT AND ALL PROGRAM OPERATION WILL ONLY LOOP BACK TO LINE 10
C  WHEN VALUES ARE CHANGED ON THE SCREEN
C  AN EXCEPTION OCCURS WHEN THE SIMULATE OPTION HAS BEEN
C  SELECTED,THE PROGRAM LOOPS BACK TO LINE 99 SO THAT THE
C  VALUES
C  MAY BE REWRITTEN TO THE SCREEN
C
 10 CALL ANSWER(TBIAS,R,SDR,BIAS,VAR)
    IF(ISW2)THEN
      WRITE(9,4)
      DO 50 I=1,N
 50   WRITE(9,5)I,FACT(I),X(I),SDX(I),BIAS(I),VAR(I)
      WRITE(9,6)TBC,FRNM,TBIAS,FRUNTS
      WRITE(9,6)EST,FRNM,R,FRUNTS
      WRITE(9,6)SDV,FRNM,SDR,FRUNTS
      WRITE(9,*)' '
    ENDIF
    ISW=.FALSE.
    CALL EDITV(X,SDX,ISW)
C
C  CHECK IF THE SIMULATE OPTION HAS BEEN SELECTED
C
    IF(ISW)GO TO 90
C
C  WRITE A HILIGHTED WAIT STATEMENT AT THE BOTTOM RIGHT
C  CORNER OF THE SCREEN DURING CALCULATION
C
    CALL LOCATE(2,24,70)
```

```
      CALL PUTST(2,120,4,'WAIT')
C
      CALL MNVRCP(X, SDX, BIAS, VAR, N, IER)
      R = 0.D0
      SDR = 0.D0
C
C  CALCULATE THE TOTAL BIAS AND VARIANCE VALUES
C  OF THE ESTIMATE
C
      DO 30 I = 1, N
        R = R + BIAS(I)
   30 SDR = SDR + VAR(I)
C
C  CALCULATE THE PERCENTAGE COMPONENT OF BOTH THE BIAS
C  AND VARIANCE FOR ARGUMENT I
C
      DO 40 I = 1, N
        IF (R .NE. 0.D0) BIAS(I) = BIAS(I) * 100.D0 / R
        IF (SDR .NE. 0.D0) VAR(I) = VAR(I) * 100.D0 / SDR
      XX(I)=X(I)
      SDXX(I)=SDX(I)
   40 CONTINUE
      TBIAS=R
C
C  CALCULATE THE ESTIMATE OF THE FUNCTION VALUE AND THE
C  STANDARD DEVIATION
C
      R = R + REC(X,N)
      SDR = DSQRT(SDR)
C
C  REMOVE HILIGHTED WAIT STATEMENT
C
      CALL LOCATE(2,24,70)
      CALL PUTST(2,7,4,'    ')
      GO TO 10
C
C  SIMULATE THE FUNCTION HERE
C
   90 CALL SCREEN(2)
   95 WRITE(*,*)'NORMAL (N) OR LOGNORMAL (L) ?    <CR> = N'
      READ(*,'(A)')YES
      LN=.FALSE.
      IF(YES.EQ.'L'.OR.YES.EQ.'l')LN=.TRUE.
      WRITE(*,*)' '
      DSEED=99876321.D0
      WRITE(*,*)'RANDOM SEED ?   <CR> =99876321.D0'
      READ(*,'(D20.0)')RDSEED
      IF(RDSEED.NE.0.D0)DSEED=RDSEED
      RDSEED=DSEED
      WRITE(*,*)' '
      WRITE(*,*)'SIMULATION IN PROGRESS ...'
      WRITE(*,*)' '
      DO 100 J=1,NN
        DO 110 I=1,N
          IF(SDX(I).LE.0.D0)GO TO 110
          IF(LN)THEN
            XX(I)=RLOGN(X(I),SDX(I),DSEED)
          ELSE
            XX(I)=RNORM(X(I),SDX(I),DSEED)
          ENDIF
  110   CONTINUE
  100 SIM(J)=REC(XX,N)
C
C  PARTIAL SORT THE SIMULATED VALUES AND WRITE SELECTED
C  VALUES TO SCREEN
C
      CALL PSORTI(SIM,NN,IND,8)
      DO 120 I=1,8
  120 WRITE(*,8)' QUANTILE ',IND(I)/10.,' % ',FRNM,SIM(IND(I)),FRUNTS
    8 FORMAT(A,F5.1,A,A,G12.5,A)
C
C--WRITE SIMULATION SUMMARY TO A DATA FILE IF REQUESTED
C
      IF(ISW2)THEN
        WRITE(9,*)' '
        IF(LN)THEN
          WRITE(9,*)'LOGNORMAL'
        ELSE
          WRITE(9,*)'NORMAL'
        ENDIF
        WRITE(9,135)RDSEED
  135   FORMAT(' DSEED = ',D20.13)
        WRITE(9,*)' '
        DO 140 I=1,8
  140   WRITE(9,8)' QUANTILE ',IND(I)/10.,' % ',FRNM,SIM(IND(I)),FRUNTS
        WRITE(9,*)' '
      ENDIF
C
      WRITE(*,*)' '
      WRITE(*,*)'WRITE SIMULATED VALUES TO A FILE ? (Y/N)   <CR> = N'
      READ(*,7)YES
      IF(YES.EQ.'Y'.OR.YES.EQ.'y')THEN
        WRITE(*,*)' '
        WRITE(*,*)'ENTER FILE NAME :'
        READ(*,7)FLNM
    2   INQUIRE(FILE=FLNM,EXIST=EX)
        IF(EX)THEN
          WRITE(*,*)'** FILE ALREADY EXISTS **'
          WRITE(*,*)' '
          WRITE(*,*)'ENTER ANOTHER FILE NAME. <CR> TO OVERWRITE'
```

```
       FLNM2 = '                    '
       READ(*,7)FLNM2
       IF(FLNM2(1:1).NE.' ')THEN
        FLNM=FLNM2
        GOTO 2
       ENDIF
      ENDIF
      OPEN(7,FILE=FLNM)
      DO 130 I=1,NN
 130  WRITE(7,*)SIM(I)
     ENDIF
C
C  SIMULATION COMPLETE, RETURN TO INPUT SCREEN
C
     GO TO 99
     END
```

```
$STORAGE:2
      SUBROUTINE MNVRCP( X, SDX, BIAS, VAR, N, IER)
C
C  PURPOSE:
C
C TO APPROXIMATE THE BIAS AND VARIANCE COMPONENTS OF A
C FUNCTION OF "N" INDEPENDENT VARIATES BY MEANS OF PARTIAL
C DERIVATIVE ESTIMATION
C
C  INPUT:
C
C  X = REAL*8 VECTOR OF LENGTH "N" OF THE MEAN VALUES OF THE
C     ARGUMENTS FOR A USER-SUPPLIED FUNCTION "REC"
C  SDX = REAL*8 VECTOR OF STANDARD DEVIATION ESTIMATES OF THE
C       VECTOR "X".  THESE VALUES MUST ALL BE NON NEGATIVE
C  BIAS = REAL*8 VECTOR OF LENGTH "N" TO HOLD THE ESTIMATES OF
C       THE COMPONENTS THAT MUST BE ADDED TO THE FUNCTION "REC"
C       EVALUATED AT THE MEAN "X" TO APPROXIMATE, TO SECOND
C     ORDER RESOLUTION, THE TRUE EXPECTATION OF "REC".
C  VAR = REAL*8 VECTOR OF LENGTH "N" TO HOLD THE ESTIMATES OF
C       THE COMPONENTS OF VARIANCE THAT MUST BE SUMMED TO
C   APPROXIMATE TO SECOND ORDER RESOLUTION, THE TRUE VARIANCE
C       OF "REC"
C
C  N = INTEGER*2 VARIABLE OR CONSTANT GIVING THE DIMENSIONS OF
C     "X","SDX","BIAS" AND "VAR".
C  IER = INTEGER*2 VARIBLE TO HOLD AN ERROR RETURN CODE.
C
C  OUTPUT:
C
C  "X","SDX" AND "N" ARE UNCHANGED.
C  "BIAS" AND "VAR" CONTAIN THE BIAS AND VARIANCE COMPONENTS
C  AS DETAILED UNDER "INPUT".
C
C  ROUTINES REQUIRED:
C
C        REAL*8 FUNCTION DDERIV
C  THIS IMSL FUNCTION COMPUTES THE FIRST, SECOND OR THIRD ORDER
C  DERIVATIVE OF A USER-SUPPLIED FUNCTION.  A RELATIVE ERROR
C  (DTOL) OF 0.01 IS USED IN THIS PROGRAM FROM THE DERIVATIVE
C  ESTIMATES.  CONVERGENCE IS ASSUMED WHEN
C  ABS(D2-D1)/MAX(1.0,SQRT(D1**2+D2**2)) .LT. DTOL
C
C        EXTERNAL REAL*8 FUNCTION "DFCN"
C  THIS FUNCTION IS PASSED INTO IMSL FUNCTION DDERIV SO THAT
C  PARTIAL DERIVATIVES MAY BE EVALUATED USING FUNCTION "REC"
C
C        SUBROUTINE ERSET
C  THIS IMSL SUBROUTINE IS CALLED TO SET THE ACTIONS TO BE TAKEN
C WHEN ERRORS OCCUR IN FUNCTION DERRIV.THE PARAMETERS USED
```

```
C IN THIS SUBROUTINE INSTRUCT THE COMPUTER NOT TO PRINT ERRORS
C OR TERMINATE THE PROGRAM IF ERRORS OCCUR.
C
C      INTEGER*4 FUNCTION IERCD
C  THIS IMSL FUNCTION IS CALLED TO RETRIEVE THE INTEGER CODE
C  FOR AN INFORMATIONAL ERROR.  IF IERCD() RETURNS A VALUE OF 1,
C SUBROUTINE DDERIV WAS UNABLE TO ACHIEVE DESIRED TOLERANCE
C IN DERIVATIVE ESTIMATION.  IF IERCD() RETURNS A VALUE OF 2,
C  ROUNDOFF ERROR BECAME DOMINANT.
C
C  ERRORS:
C
C  IER=0 NO ERRORS IN "SDX" OR "N"
C  IER=1 "N" LT 1
C  IER=2 AN ELEMENT OF "SDX" IS NEGATIVE.
C
C  METHOD:
C
C  IF "SDX(I)=0." THEN  "BIAS(I)" AND "VAR(I)" ARE 0.
C  THE "BIAS" AND "VAR" COMPONENT ESTIMATION EQUATIONS ARE
C  DEVELOPED FROM A TAYLOR SERIES EXPANSION OF "REC" ABOUT "X",
C  TAKING EXPECTATIONS AND TRUNCATING TERMS IN COVARIANCE AND
C  THIRD ORDER PARTIAL DERIVATIVES.  DETAILS MAY BE FOUND IN
C  ANY REPUTABLE APPLIED STATISTICS TEXT.
C
C ****************************************************************
C                  ! NOTE !
C  AS ONLY THE STANDARD DEVIATION OF THE VECTOR "X" IS
C  GIVEN IT HAS BEEN IMPLICITLY ASSUMED THAT ALL COVARIANCE AND
C  HIGHER PRODUCT MOMENTS WITHIN "X" ARE ZERO. THAT IS ALL THE
C  "X(I)" ARE STATISTICALLY INDEPENDENT.
C ****************************************************************
C
C
      REAL*8 X(*), SDX(*), BIAS(*), VAR(*), XREC(20)
      INTEGER*2 I, N, IER, NREC,NSW
      INTEGER*4 KORDER,IERCD,NER
      REAL*8 DTOL, DBGSTE, D1,D2, DDERIV,DSV
      EXTERNAL DFCN, DDERIV,IERCD,ERSET
      COMMON/XVAL/XREC,NREC
      COMMON/SWITCH/NSW
C
      DO 1 I=1,N
        XREC(I)=X(I)
    1 CONTINUE
      NREC=N
C TESTS FOR INVALID "N" OR "SDX"
      IER = 1
      IF (N .LT. 1) RETURN
      IER = 2
      DO 10 I = 1, N
        IF (SDX(I) .LT. 0.D0) RETURN
   10 CONTINUE
      IER = 0
C
C TURN OFF PRINTING AND STOPPING FOR ALL ERRORS ENCOUNTERED
C IN IMSL
      NER=0
      CALL ERSET(NER,NER,NER)
      DTOL=0.01D0
C LOOP FOR "BIAS" AND "VAR" EVALUATION VIA PARTIAL DERIVATIVES.
      DO 20 I = 1, N
        BIAS(I) = 0.D0
        VAR(I) = 0.D0
C IF "SDX(I)" IS 0. THEN BOTH "BIAS(I)" AND "VAR(I)" ARE 0.
        IF (SDX(I) .EQ. 0.D0) GO TO 20
C
C NOW DERIVATIVES ARE TAKEN USING IMSL ROUTINE DDERIV
C IF ERRORS RESULT DUE TO ROUNDOFF BECOMING TOO DOMINANT OR
C TOLERANCE NOT BEING MET, STEP SIZE IS INCREASED BY A FACTOR OF
C 2 AND DERIVATIVES ARE CALCULATED AGAIN
C
        NSW=I
        DBGSTE=SDX(I)
        DSV=DBGSTE
        DO 40 KORDER=1,2
   30     D2 = DDERIV(DFCN,KORDER,X(I),DBGSTE,DTOL)
          IF(IERCD().EQ.1.OR.IERCD().EQ.2)THEN
            DBGSTE=DBGSTE*2.D0
          IF(DBGSTE.GT.10.D0*X(I))STOP' ROUNDOFF OR TOLERANCE ERROR !!'
            GO TO 30
          ENDIF
          IF(KORDER.EQ.1)D1=D2
          DBGSTE=DSV
   40   CONTINUE
C
        IF(DABS(D2).LE.DABS(D1/100.))D2=0.D0
C BIAS COMPONENT IS ONE HALF THE PRODUCT OF THE SECOND ORDER
C PARTIAL AND THE SQUARE OF THE STANDARD DEVIATION.
C VARIANCE COMPONENT IS THE PRODUCT OF THE FIRST ORDER
C PARTIAL AND THE STANDARD DEVIATION SQUARED.
        BIAS(I) = (D2/2.D0) * (SDX(I)**2)
        VAR(I) = (D1*SDX(I)) ** 2
C
   20 CONTINUE
      RETURN
      END
      REAL*8 FUNCTION DFCN(XX)
C
C  PURPOSE:
C
C  REAL*8 FUNCTION DFCN IS USED TO MAKE IT POSSIBLE TO USE
C  IMSL SUBROUTINE DERRIV TO OBTAIN PARTIAL DERRIVATIVES.
C SUBROUTINE DERRIV USES A USER-SUPPLIED FUNCTION TO CALCULATE
C  A FIRST, SECOND OR THIRD ORDER DERIVATIVE AT X.
C  BY USING A USER-SUPPLIED FUNCTION WITHIN FUNCTION DFCN,
C  PARTIAL DERIVATIVES MAY BE EVALUATED WITHIN A FUNCTION FOR
C MORE THAN ONE INDEPENDENT VARIABLE.ALL THAT IS REQUIRED IS
C THAT THE ELEMENT NUMBER (NSW) OF THE INDEPENDENT VARIABLE
C  ARRAY X IS KEPT TRACK OF SO THAT EACH VARIABLE CAN HAVE
C  PARTIAL DERIVATIVES TAKEN WITH RESPECT TO THE FUNCTION.
C  OBTAINING PARTIAL DERIVATIVES USING DERRIV IN THIS MANNER
C  ALLOWS THE STEP SIZE TO BE ADAPTIVELY CHANGED TO ENHANCE
C  ACCURACY FOR EACH VARIABLE SEPARATELY.
C
C  INPUT:
C
C  XX = REAL*8 VARIABLE CONTAINING THE POINT AT WHICH THE
C     FUNCTION DFCN IS TO BE EVALUATED AT FOR A DERIVATIVE
C
C  COMMON BLOCK XVAL :
C        X = REAL*8 ARRAY OF LENGTH 20 CONTAINING THE VALUES
C            OF THE INDEPENDENT VARIABLES
C        N = INTEGER*2 VARIABLE CONTAINING THE NUMBER OF
C            INDEPENDENT VARIABLES IN THE FUNCTION
C
C  COMMON BLOCK SWITCH :
C        NSW = INTEGER*2 VARIABLE CONTAINING THE ELEMENT NUMBER
C                OF THE INDEPENDENT VARIABLE ARRAY FOR WHICH THE
C                THE PARTIAL DERIVATIVE IS BEING CALCULATED AT
C                BY IMSL SUBROUTINE DERRIV
C
C  INTERNAL VARIABLE :
C
C   XSV = REAL*8 VARIABLE USED TO STORE THE VALUE OF X(NSW)
C        WHILE THE VARIABLE IS BEING CHANGED DUE TO STEP
C        SIZE ALTERATIONS DURING THE OPERATION OF DERRIV.
C
C  OUTPUT:
C
C  DFCN = REAL*8 VALUE OF THE FUNCTION REC WHEN X(NSW) IS SET
C          EQUAL TO XX
C  X,N,NSW = UNCHANGED IN OUTPUT
C
C  FUNCTIONS REQUIRED :
C
C  REC = REAL*8 USER-SUPPLIED FUNCTION CONTAINING A FUNCTION
```

```
C        DEFINED WITH N INDEPENDENT VARIABLES IN REAL*8 ARRAY X
C
C
      REAL*8 XX, X(20), XSV, REC
      INTEGER*2 N, NSW
      EXTERNAL REC
      COMMON/XVAL/X,N
      COMMON/SWITCH/NSW
C
      XSV = X(NSW)
      X(NSW) = XX
      DFCN = REC(X,N)
      X(NSW) = XSV
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE EDITV(VALUE1,VALUE2,ISW)
C
C  PURPOSE:
C
    THIS SUBROUTINE CONTROLS THE USER INTERACTION WHEN THE
C  INPUT SCREEN IS DISPLAYED.
C
C  INPUT:
C
C VALUE1 = REAL*8 VECTOR CONTAINING THE ESTIMATED MEAN VALUES
C     OF THE ARGUMENTS
C  VALUE2 = REAL*8 VECTOR CONTAINING THE ESTIMATED STANDARD
C       DEVIATION VALUES OF THE ARGUMENTS
C  ISW = LOGICAL*2 VARIABLE CONTAINING WHETHER OR NOT THE USER
C       HAS SELECTED THE SIMULATE OPTION
C
C  INTERNAL VARIABLES:
C
C IROW=INTEGER*2 VARIABLE CONTAINING THE ARGUMENT NUMBER TO
C       HIGHLITE
C  ICOL = INTEGER*2 VARIABLE CONTAINING THE COLUMN TO HIGHLITE
C       WHERE A VALUE OF 1 WILL HIGHLITE THE MEAN CELL AND A
C       VALUE OF 2 WILL HIGHLITE THE STANDARD DEVIATION CELL
C N=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF ARGUMENTS IN
C       THE FUNCTION
C  NAME = CHARACTER*28 VECTOR CONTAINING THE NAMES OF THE
C       ARGUMENTS
C  COL = INTEGER*2 VECTOR CONTAINING THE COLUMN LOCATIONS FOR
C       SCREEN OUTPUT
C  ROW = INTEGER*2 VECTOR CONTAINING THE ROW LOCATIONS FOR
C       SCREEN OUTPUT
C  CH = CHARACTER*1 VARIABLE CONTAINING THE ASCII CHARACTER
C       READ FROM THE KEYBOARD
C  SCAN = INTEGER*2 VARIBLE CONTAINING THE IBM EXTENDED CODE
C
C  OUTPUT:
C
C VALUE1 AND VALUE2 VECTORS WILL CONTAIN THE NEW VALUES OF
C MEAN AND STANDARD DEVIATION THAT HAVE BEEN ENTERED ON THE
C SCREEN.
C ISW WILL RETURN A VALUE OF .TRUE. IF THE SIMULATE OPTION HAS
C BEEN SELECTED, OTHERWISE .FALSE. IS RETURNED.
C
C  ROUTINES REQUIRED:
C
C       SUBROUTINE SCREEN
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO SET THE SCREEN MODE AND CLEAR THE SCREEN
C
```

```
C       INTEGER*2 FUNCTION INKEY
C  THIS FUNCTION IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO DETERMINE WHETHER A KEY IS PRESSED FROM THE
C  KEYBOARD AND WHAT IT IS.
C  INKEY RETURNS A VALUE OF 1 IF A KEY HAS BEEN PRESSED
C  INKEY RETURNS A VALUE OF 0 IF A KEY HAS NOT BEEN PRESSED
C
C       SUBROUTINE INSCRV
C  THIS SUBROUTINE IS USED TO WRITE THE ARGUMENT NAMES AND THE
C  INITIAL MEAN AND STANDARD DEVIATION VALUES TO THE SCREEN.
C  IT ALSO WRITES TO THE SCREEN THE HEADING AND COMMAND BAR
C  LABELS.
C
C       SUBROUTINE CURSORV
C  THIS SUBROUTINE CONTOLS CURSOR MOVEMENT, HIGHLITING OF
C  CELLS AND STORAGE OF NEW VALUES ENTERED ONTO THE SCREEN
C
C
      CHARACTER*1 NULL, CH
      INTEGER*2 SCAN, INKEY, IROW, ICOL, N, NDAT
      INTEGER*2 ROW(20), COL(2), J
      CHARACTER NAME(*)*28
      REAL*8 VALUE1(*),VALUE2(*)
      LOGICAL*2 IS[illegible]
C
   20 IF (INKEY(CH,SCAN) .EQ. 0) GO TO 20
C
C--CHECK IF ESC WAS PRESSED TO QUIT
C
      IF(ICHAR(CH).EQ.27)THEN
         CALL SCREEN(2)
         STOP '      NORMAL EXIT BY USER'
      ENDIF
C
C-CHECK IF A SPECIAL KEY SUCH AS A FUNCTION KEY HAS BEEN PRESSED
C
      IF(CH.EQ.NULL)
     + CALL CURSORV(CH,SCAN,IROW,ICOL,VALUE1,VALUE2,N,ROW,COL)
C
C-CHECK IF A NUMBER HAS BEEN ENTERED DIRECTLY WITHOUT USING
C F1
C
      SELECT CASE(ICHAR(CH))
      CASE(46,48:57)
         CALL CURSORV(CH,SCAN,IROW,ICOL,VALUE1,VALUE2,N,ROW,COL)
      ENDSELECT
C
C--CHECK IF 'S' IS PRESSED TO SIMULATE        SCAN VALUES: S=83
C                                                          s=115
C
```

```
      IF(ICHAR(CH).EQ.83.OR.ICHAR(CH).EQ.115)THEN
        ISW=.TRUE.
        RETURN
      ENDIF
C
C  CHECK IF SPACE BAR WAS PRESSED TO CALCULATE
C
      IF(ICHAR(CH).NE.32)GO TO 20
C
C  IF A KEY IS PRESSED WHICH IS NOT SUPPOSED TO BE USED ON THE
C  INPUT SCREEN, IT IS IGNORED AND THE PROGRAM RETURNS TO LINE 20
C  AND WAITS FOR ANOTHER KEY TO BE PRESSED
C
      RETURN
C
C  THIS ENTRY IS USED TO INITIALIZE THE DATA LOCATIONS ON THE
C  SCREEN. IT ALSO WRITES OUT ALL OF THE INITIAL VALUES AND
C  LABELS
C
      ENTRY ED(NDAT,NAME,VALUE1,VALUE2)
C
      NULL=CHAR(+8#00)
      N=NDAT
      COL(1)=33
      COL(2)=48
      DO 100 J=1,N
  100   ROW(J)=J+1
      IROW=1
      ICOL=1
      CALL INSCRV(IROW,ICOL,N,NAME,VALUE1,VALUE2,COL,ROW)
C
      RETURN
      END
```

```
$STORAGE:2
 SUBROUTINE CURSORV(CH,SCAN,IROW,ICOL,VALUE1,VALUE2,N,ROW,COL)
C
C  PURPOSE:
C
C THIS SUBROUTINE CONTROLS CURSOR MOVEMENT, HIGHLITING OF
C CELLS AND STORAGE OF NEW VALUES ENTERED ONTO THE SCREEN
C
C  INPUT:
C
C  CH = CHARACTER*1 VARIABLE CONTAINING THE ASCII CHARACTER
C    READ FROM THE KEYBOARD
C  SCAN = INTEGER*2 VARIBLE CONTAINING THE IBM EXTENDED CODE
C IROW=INTEGER*2 VARIABLE CONTAINING THE ARGUMENT NUMBER TO
C      HIGHLITE
C  ICOL = INTEGER*2 VARIABLE CONTAINING THE COLUMN TO HIGHLITE
C      WHERE A VALUE OF 1 WILL HIGHLITE THE MEAN CELL AND A
C      VALUE OF 2 WILL HIGHLITE THE STANDARD DEVIATION CELL
C N=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF ARGUMENTS IN
C      THE FUNCTION
C  NAME = CHARACTER*28 VECTOR CONTAINING THE NAMES OF THE
C      ARGUMENTS
C VALUE1 = REAL*8 VECTOR CONTAINING THE ESTIMATED MEAN VALUES
C    OF THE ARGUMENTS
C  VALUE2 = REAL*8 VECTOR CONTAINING THE ESTIMATED STANDARD
C      DEVIATION VALUES OF THE ARGUMENTS
C  COL = INTEGER*2 VECTOR CONTAINING THE COLUMN LOCATIONS FOR
C      SCREEN OUTPUT
C  ROW = INTEGER*2 VECTOR CONTAINING THE ROW LOCATIONS FOR
C      SCREEN OUTPUT
C
C  OUTPUT:
C
C  CH, SCAN, IROW, ICOL, N, ROW AND COL ARE UNCHANGED
C  VALUE1 OR VALUE2 MAY BE CHANGED, DEPENDING ON USER
C INTERACTION WHILE ENTERING NUMBERS INTO THE CELL
C
C  ROUTINES REQUIRED:
C
C       SUBROUTINE SCREEN
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO SET THE SCREEN MODE AND CLEAR THE SCREEN
C
C       INTEGER*2 FUNCTION INKEY
C  THIS FUNCTION IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO DETERMINE WHETHER A KEY IS PRESSED FROM THE
C  KEYBOARD AND WHAT IT IS.
C  INKEY RETURNS A VALUE OF 1 IF A KEY HAS BEEN PRESSED
C  INKEY RETURNS A VALUE OF 0 IF A KEY HAS NOT BEEN PRESSED
C
```

```
C      SUBROUTINE LOCATE
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO POSITION THE CURSOR ON THE SELECTED SCREEN PAGE
C
C      SUBROUTINE PUTST
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO WRITE A STRING WITH ATTRIBUTE AT THE CURRENT
C  CURSOR LOCATION
C
C      SUBROUTINE NEWCUR
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO DEFINE THE SHAPE AND SIZE OF THE TEXT CURSOR.
C  THE ARGUMENT VALUES OF 32 AND 0 DEFINE AN INVISIBLE CURSOR.
C
C      SUBROUTINE PUTCHA
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO WRITE A CHARACTER WITH ATTRIBUTE AT THE
C  CURRENT CURSOR POSITION
C
C
    INTEGER*2 FIELD
    INTEGER*2 SCAN, IROW, ICOL, SCAN2, INKEY, IDEC, N
    INTEGER*2 ITEMP, II
    PARAMETER(FIELD=14)
    INTEGER*2 ROW(*), COL(2), I
    CHARACTER*14 DUMFF, BLANK
    CHARACTER*1 DUMF(FIELD), CH, CH2, NULL
    EQUIVALENCE (DUMF(1),DUMFF)
    REAL*8 VALUE1(*), VALUE2(*)
    LOGICAL*2 ISW
C
C    SCAN VALUES: 72 = UP ARROW
C                 75 = LEFT ARROW
C                 77 = RIGHT ARROW
C                 80 = DOWN ARROW
C                 59 = FUNCTION KEY 1
C
    NULL=CHAR(+8#00)
    IF(CH.EQ.NULL)THEN
      SELECT CASE(SCAN)
      CASE(59,72,75,77,80)
      CASE DEFAULT
        RETURN
      END SELECT
    ELSE
      SCAN=59
    ENDIF
C
    ISW=.TRUE.
    DATA BLANK/'              '/
```

```
      CALL LOCATE(2,ROW(IROW),COL(ICOL))
C
C  CHECK IF F1 IS PRESSED IN ORDER TO EDIT DATA
C
      IF(SCAN.EQ.59)THEN
C  WRITE EDIT SPACE ON SCREEN
        CALL PUTST(2,120,FIELD,BLANK)
        CALL NEWCUR(10,10)
C
      DUMFF=BLANK
      IDEC=0
      ITEMP=1
      II=1
C  BACKSPACING IN EDIT SPACE
    5 IF(ITEMP.GE.1)THEN
        II=ITEMP
        IF(IDEC.EQ.II)IDEC=0
        DUMF(II)=' '
        CALL LOCATE(2,ROW(IROW),COL(ICOL)-1+II)
        CALL PUTCHA(2,120,1,' ')
      ENDIF
C  LOOP TO READ THE CHARACTERS ENTERED IN EDIT SPACE
      DO 10 I=II,FIELD
        CALL LOCATE(2,ROW(IROW),COL(ICOL)-1+I)
C
C  CHECK IF A NUMBER WAS ENTERED DIRECTLY WITHOUT THE USE OF
C  F1
C
        IF(CH.NE.NULL.AND.I.EQ.1)THEN
          CH2=CH
          CH=NULL
          GO TO 25
        ENDIF
C
C  READ KEYSTROKES WHILE EDITING A HIGHLITED DATA CELL
C
   20   IF(INKEY(CH2,SCAN2).EQ. 0) GO TO 20
        IF(CH2.EQ.NULL)THEN
          SELECT CASE(SCAN2)
C
C  CHECK IF F1 IS PRESSED IN ORDER TO QUIT EDITING AND RESTORE
C  THE PREVIOUS VALUE
C
          CASE(59)
            GO TO 35
C
C  CHECK IF A CURSOR KEY WAS PRESSED TO SAVE THE VALUE ENTERED
C  AND MOVE THE CURSOR
C
          CASE(72,75,77,80)
```

```
            ISW=.FALSE.
            IF(IDEC.EQ.0)THEN
              IDEC=I
              DUMF(I)='.'
            ENDIF
            GO TO 30
          END SELECT
        ENDIF
C
C  CHECK IF ESC WAS PRESSED TO QUIT
C
      IF(ICHAR(CH2).EQ.27)THEN
        CALL SCREEN(2)
        STOP '      NORMAL EXIT BY USER'
      ENDIF
C
C  CHECK FOR 0-9 AND DECIMAL
   25   SELECT CASE(ICHAR(CH2))
        CASE(48:57,46)
C  CHECK FOR MORE THAN ONE DECIMAL
          IF(ICHAR(CH2).EQ.46.AND.IDEC.NE.0)GO TO 20
C  MARK LOCATION OF DECIMAL IN STRING
          IF(ICHAR(CH2).EQ.46)IDEC=I
C  WRITE CHARACTER IN EDIT SPACE
          WRITE(DUMF(I),'(A)')CH2
          CALL PUTCHA(2,120,1,CH2)
C  CHECK FOR CARRIAGE RETURN
        CASE(13)
          IF(IDEC.EQ.0)THEN
            IDEC=I
            DUMF(I)='.'
          ENDIF
          GO TO 30
C  CHECK FOR BACKSPACE KEY PRESSED IN ORDER TO ERASE PREVIOUS
C  CHARACTER
        CASE(8)
          ITEMP=I-1
          GO TO 5
        CASE DEFAULT
          GO TO 20
        END SELECT
 10   CONTINUE
 30     IF(ICOL.EQ.1)THEN
          READ(DUMFF,'(G14.7)')VALUE1(IROW)
        ELSE
          READ(DUMFF,'(G14.7)')VALUE2(IROW)
        ENDIF
 35     IF(ICOL.EQ.1)THEN
          WRITE(DUMFF,'(G14.7)')VALUE1(IROW)
        ELSE
```

```
          WRITE(DUMFF,'(G14.7)')VALUE2(IROW)
        ENDIF
        CALL LOCATE(2,ROW(IROW),COL(ICOL))
        CALL NEWCUR(32,0)
        IF(ISW)THEN
          CALL PUTST(2,120,FIELD,DUMFF)
          RETURN
        ENDIF
      ENDIF
      IF(ISW)THEN
        IF(ICOL.EQ.1)THEN
          WRITE(DUMFF,'(G14.7)')VALUE1(IROW)
        ELSE
          WRITE(DUMFF,'(G14.7)')VALUE2(IROW)
        ENDIF
      ENDIF
      CALL PUTST(2,7,FIELD,DUMFF)
C
C CHECK IF UP, DOWN, LEFT OR RIGHT ARROW KEYS HAVE BEEN
C PRESSED
C
      IF(ISW)THEN
        SCAN2=SCAN
      ENDIF
      SELECT CASE(SCAN2)
      CASE(72)
        IROW=IROW-1
        IF(IROW.LT.1)IROW=N
      CASE(75)
        ICOL=ICOL-1
        IF(ICOL.LT.1)ICOL=2
      CASE(77)
        ICOL=ICOL+1
        IF(ICOL.GT.2)ICOL=1
      CASE(80)
        IROW=IROW+1
        IF(IROW.GT.N)IROW=1
      ENDSELECT
C
C  HIGHLIGHT CURRENT VALUE SELECTED
C
      IF(ICOL.EQ.1)THEN
        WRITE(DUMFF,'(G14.7)')VALUE1(IROW)
      ELSE
        WRITE(DUMFF,'(G14.7)')VALUE2(IROW)
      ENDIF
      CALL LOCATE(2,ROW(IROW),COL(ICOL))
      CALL PUTST(2,120,FIELD,DUMFF)
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE ANSWER(TBIAS,R,SDR,BIAS,VAR)
C
C  PURPOSE:
C
C  THIS SUBROUTINE WRITES THE ESTIMATED COMPONENT VALUES OF
C  BIAS AND VARIANCE ONTO THE SCREEN.  THE FUNCTION ESTIMATE,
C  TOTAL BIAS AND VARIANCE ARE ALSO WRITTEN TO THE SCREEN.
C
C  INPUT:
C
C BIAS(I)=REAL*8 VECTOR OF LENGTH "N" CONTAINING THE PERCENTAGE
C      BIAS COMPONENT OF ARGUMENT I
C VAR(I)=REAL*8 VECTOR OF LENGTH "N" CONTAINING THE PERCENTAGE
C       VARIANCE  COMPONENT OF ARGUMENT I
C  TBIAS = REAL*8 VARIABLE CONTAINING THE TOTAL BIAS VALUE
C  R = REAL*8 VARIABLE CONTAINING THE ESTIMATED FUNCTION VALUE
C  SDR = REAL*8 VARIABLE CONTAINING THE ESTIMATED STANDARD
C       DEVIATION OF THE FUNCTION
C  FRNM = CHARACTER*15 VARIABLE CONTAINING THE FUNCTION NAME
C  FNUNTS = CHARACTER*7 VARIABLE CONTAINING THE FUNCTION UNITS
C  N=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF AGUMENTS IN
C       THE FUNCTION
C
C  OUTPUT:
C
C  TBIAS, R, SDR, BIAS, VAR, [illegible] NUNTS ARE UNCHANGED
C
C  ROUTINES REQUIRED:
C
C       SUBROUTINE LOCATE
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO POSITION THE CURSOR ON THE SELECTED SCREEN PAGE
C
C       SUBROUTINE PUTST
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO WRITE A STRING WITH ATTRIBUTE AT THE CURRENT
C  CURSOR LOCATION
C
C
      CHARACTER*30 TBC,EST,SDV
      CHARACTER*15 FRNM
      CHARACTER*7 FRUNTS
      CHARACTER*7 DUM,BLANK
      CHARACTER*14 DUMF,BL
      REAL*8 TBIAS,R,SDR,BIAS(*),VAR(*)
      INTEGER*2 N,I,NP
      DATA BLANK/'       '/
      DATA BL/'              '/
      DATA TBC/' TOTAL BIAS IN '/
      DATA EST/' ESTIMATED      '/
      DATA SDV/' STANDARD DEV  '/
C
C  THE FOLLOWING LOOP WRITES THE COMPONENT BIAS AND VARIANCE
C  TERMS TO THE SCREEN IN THEIR RESPECTIVE COLUMNS
C
      DO 10 I=1,NP
        DUM=BLANK
        WRITE(DUM,'(F7.2)')BIAS(I)
        CALL LOCATE(2,I+1,63)
        CALL PUTST(2,7,7,DUM)
        DUM=BLANK
        WRITE(DUM,'(F7.2)')VAR(I)
        CALL LOCATE(2,I+1,71)
        CALL PUTST(2,7,7,DUM)
   10 CONTINUE
C
C  WRITE TO THE SCREEN THE ESTIMATED VALUES OF TOTAL BIAS,
C  FUNCTION AND STANDARD DEVIATION
C
      DUMF=BL
      CALL LOCATE(2,NP+2,47)
      WRITE(DUMF,'(G14.7)')TBIAS
      CALL PUTST(2,7,14,DUMF)
      DUMF=BL
      CALL LOCATE(2,NP+3,47)
      WRITE(DUMF,'(G14.7)')R
      CALL PUTST(2,7,14,DUMF)
      DUMF=BL
      CALL LOCATE(2,NP+4,47)
      WRITE(DUMF,'(G14.7)')SDR
      CALL PUTST(2,7,14,DUMF)
      RETURN
C
C THIS ENTRY IS USED TO WRITE TO THE SCREEN THE LABELS AND UNITS
C OF THE FUNCTION, TOTAL BIAS AND STANDARD DEVIATION ESTIMATES.
C THIS ENTRY IS USED SO THAT THIS IS CARRIED OUT ONLY ONCE.
C
      ENTRY ANS(FRNM,FRUNTS,N)
      NP=N
      CALL LOCATE(2,NP+2,1)
      CALL PUTST(2,7,30,TBC)
      CALL LOCATE(2,NP+3,1)
      CALL PUTST(2,7,30,EST)
      CALL LOCATE(2,NP+4,1)
      CALL PUTST(2,7,30,SDV)
      DO 20 I=1,3
        CALL LOCATE(2,NP+I+1,31)
        CALL PUTST(2,7,15,FRNM)
        CALL LOCATE(2,NP+I+1,62)
```

```
    CALL PUTST(2,7,7,FRUNTS)
20 CONTINUE
   RETURN
   END
```

```
$STORAGE:2
    SUBROUTINE INSCRV(IROW,ICOL,N,NAME,VALUE1,VALUE2,COL,ROW)
C
C  PURPOSE:
C
C  THIS SUBROUTINE IS USED TO WRITE THE ARGUMENT NAMES AND THE
C  INITIAL MEAN AND STANDARD DEVIATION VALUES TO THE SCREEN.
C  IT ALSO WRITES TO THE SCREEN THE HEADING AND COMMAND BAR
C  LABELS.
C
C  INPUT:
C
C IROW=INTEGER*2 VARIABLE CONTAINING THE ARGUMENT NUMBER TO
C     HIGHLITE
C  ICOL = INTEGER*2 VARIABLE CONTAINING THE COLUMN TO HIGHLITE
C      WHERE A VALUE OF 1 WILL HIGHLITE THE MEAN CELL AND A
C      VALUE OF 2 WILL HIGHLITE THE STANDARD DEVIATION CELL
C N=INTEGER*2 VARIABLE CONTAINING THE NUMBER OF ARGUMENTS IN
C      THE FUNCTION
C  NAME = CHARACTER*28 VECTOR CONTAINING THE NAMES OF THE
C      ARGUMENTS
C  VALUE1= REAL*8 VECTOR CONTAINING THE ESTIMATED MEAN VALUES
C     OF THE ARGUMENTS
C  VALUE2 = REAL*8 VECTOR CONTAINING THE ESTIMATED STANDARD
C      DEVIATION VALUES OF THE ARGUMENTS
C  COL = INTEGER*2 VECTOR CONTAINING THE COLUMN LOCATIONS FOR
C      SCREEN OUTPUT
C  ROW = INTEGER*2 VECTOR CONTAINING THE ROW LOCATIONS FOR
C      SCREEN OUTPUT
C
C  OUTPUT:
C
C IROW,ICOL, N, NAME, VALUE1, VALUE2, COL AND ROW ARE UNCHANGED
C
C  ROUTINES REQUIRED:
C
C      SUBROUTINE LOCATE
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO POSITION THE CURSOR ON THE SELECTED SCREEN PAGE
C
C      SUBROUTINE PUTST
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO WRITE A STRING WITH ATTRIBUTE AT THE CURRENT
C  CURSOR LOCATION
C
C      SUBROUTINE SCCHAR
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO READ A CHARACTER AND ITS ATTRIBUTE FROM THE
C  CURRENT CURSOR LOCATION ON THE SCREEN
C
```

```
C      SUBROUTINE NEWCUR
C  THIS SUBROUTINE IS PART OF THE FORTRAN UTILITIES LIBRARY AND
C  IS CALLED TO DEFINE THE SHAPE AND SIZE OF THE TEXT CURSOR.
C  THE ARGUMENT VALUES OF 32 AND 0 DEFINE AN INVISIBLE CURSOR.
C
C
      INTEGER*2 FIELD,F2
      PARAMETER(FIELD=14,F2=2)
      CHARACTER*14 BLANK,DUMFF
      CHARACTER DUMF(FIELD)*1, NAME(*)*28, NUM*2, BL
      EQUIVALENCE(DUMF(1),DUMFF)
      INTEGER*2 COL(F2),ROW(*),I
      INTEGER*2 N, J, IROW, ICOL, ATTRIB
      REAL*8 VALUE1(*),VALUE2(*)
      DATA BLANK/'              '/,BL/'  '/
C
C  WRITE HEADING TO THE SCREEN
C
      CALL LOCATE(2,0,1)
      CALL PUTST(2,7,17,' #   VARIABLE NAME')
      CALL LOCATE(2,0,39)
      CALL PUTST(2,7,38,'MEAN      STD DEVIATION    % BIAS    % VAR')
C
C  WRITE VARIABLE NAMES AND VALUES TO SCREEN
C
      DO 90 J=1,N
        NUM=BL
        WRITE(NUM,'(I2)')J
        CALL LOCATE(2,ROW(J),1)
        CALL PUTST(2,7,F2,NUM)
        CALL LOCATE(2,ROW(J),5)
        CALL PUTST(2,7,28,NAME(J))
        DUMFF=BLANK
        WRITE(DUMFF,'(G14.7)')VALUE1(J)
        CALL LOCATE(2,ROW(J),COL(1))
        CALL PUTST(2,7,FIELD,DUMFF)
        DUMFF=BLANK
        WRITE(DUMFF,'(G14.7)')VALUE2(J)
        CALL LOCATE(2,ROW(J),COL(2))
        CALL PUTST(2,7,FIELD,DUMFF)
 90   CONTINUE
C HIGHLITE A VALUE SPECIFIED IN THE CALL ROUTINE BY IROW AND
C ICOL
      DO 100 I=1,FIELD
         CALL LOCATE(2,ROW(IROW),COL(ICOL)-1+I)
100      CALL SCCHAR(2,DUMF(I),ATTRIB)
      CALL LOCATE(2,ROW(IROW),COL(ICOL))
      CALL PUTST(2,120,FIELD,DUMFF)
      CALL NEWCUR(32,0)
C  WRITE THE COMMAND BAR TO THE SCREEN
```

```
      CALL LOCATE(2,24,5)
      CALL PUTST(2,7,59,
     +'ESC=QUIT,  F1=EDIT/UNDO EDIT,  S=SIMULATE, SPACE=CALCULATE')
      RETURN
      END
```

```
$STORAGE:2
      SUBROUTINE PSORTI(A, N, IND, NI)
C
C  PURPOSE:
C
C  THIS IS  ACM ALGORITHM 410,PARTIAL SORTING.
C  RETURNING THE REAL VECTOR PARTIALLY SORTED WITH RESPECT TO
C  THE ASCENDING ORDER POINTERS OF THE INDICATOR VECTOR
C
C  INPUT:
C
C  A = A REAL*4 VECTOR TO BE PARTIAL SORTED
C  N = THE DIMENSIONS OF THE REAL VECTOR. INTEGER VARIABLE
C      OR CONSTANT GREAT OR EQUAL TO 1
C  IND = THE ORDER STATISTIC SET THAT THE REAL VECTOR IS TO
C    PARTIALLY SORTED INTO. INTEGER*2 VECTOR WHOSE VALUES MUST
C      BE GREATER THAN ZERO, LESS OR EQUAL TO N AND ALL
C      DIFFERENT. THEY MUST BE IN ASCENDING ORDER.
C  NI = THE DIMENSIONS OF IND. INTEGER VARIABLE OR CONSTANT
C      GREATER THAN ZERO, LESS THAN OR EQUAL TO N
C
C  OUTPUT:
C
C  A = THE REAL*4 VECTOR IS REORDERED AS PER THE NI ITEMS OF IND
C     THAT IS A(IND(I)),I=1,NI IS IN THE (IND(I) NTH ORDERED POSITION
C  N,NI & IND = REMAIN UNCHANGED
C
C  INTERNAL VARIABLES:
C
C  T,TT = TEMPORARY REAL STORAGE FOR A SWITCHING
C  INDU,INDL,IU,IL = INDEXING VECTORS TO ACT AS POINTERS
C   THESE VECTORS ARE DIMENSIONED AT 16 AND THEREFORE LIMIT THE
C     VALUE OF INPUT PARAMETER N TO 2**(16+1) = 131072
C  I,J,M,K,P,JL,JU & IJ = INTEGER INDEXING AND TEMPORARY
C     STORAGE VARIABLES
C
C  SUBROUTINES REQUIRED: NONE
C
C
      REAL*4 A(1), T, TT
      INTEGER*2 N, IND(1), NI, INDU(16), INDL(16), IU(16), IL(16)
      INTEGER*2 P, JL, JU, I, J, M, K, IJ,  L
C
      IF (NI .LT. 1 .OR. NI .GT. N) RETURN
      M = IND(1)
      IF (M .LT. 1 .OR. M .GT. N) RETURN
      IF (NI .EQ. 1) GO TO 20
      DO 10 I = 2, NI
        IF (IND(I) .LE. M .OR. IND(I) .GT. N) RETURN
   10 M = IND(I)
   20 CONTINUE
C
C FROM HERE ON THE CODE SHOULD BE EXACTLY EQUAL TO THAT
C GIVEN ON PAGES 410-412 OF COLLECTED ALGORITHMS FROM ACM VOL
C II
C
      JL = 1
      JU = NI
      INDL(1) = 1
      INDU(1) = 1
C
C VECTORS INDL,INDU KEEP ACCOUNT OF THE PORTION OF IND RELATED
C  TO THE CURRENT SEGMENT OF DATA BEING ORDERED
C
      I = 1
      J = N
      M = 1
   40 IF (I .GE. J) GO TO 150
C
C  FIRST ORDER A(I),A(J),A((I+J)/2),AND USE MEDIAN TO SPLIT DATA
C
   50 K = I
      IJ = (I + J) / 2
      T = A(IJ)
      IF (A(I) .LE. T) GO TO 60
      A(IJ) = A(I)
      A(I) = T
      T = A(IJ)
   60 L = J
      IF (A(J) .GE. T) GO TO 80
      A(IJ) = A(J)
      A(J) = T
      T = A(IJ)
      IF (A(I) .LT. T) GO TO 80
      A(IJ) = A(I)
      A(I) = T
      T = A(IJ)
      GO TO 80
   70 A(L) = A(K)
      A(K) = TT
   80 L = L - 1
      IF (A(L) .GT. T) GO TO 80
      TT = A(L)
C
C  SPLIT THE DATA INTO A(I TO L).LT.T, A(K TO J) .GT.T
C
   90 K = K + 1
      IF (A(K) .LT. T) GO TO 90
      IF (K .LE. L) GO TO 70
      INDL(M) = JL
      INDU(M) = JU
      P = M
      M = M + 1
C
C  SPLIT THE LARGER OF THE SEGMENTS
C
      IF (L - I .LE. J - K) GO TO 120
      IL(P) = I
      IU(P) = L
      I = K
C
C  SKIP ALL SEGMENTS NOT CORRESPONDING TO AN ENTRY IN IND
C
  100 IF (JL .GT. JU) GO TO 150
      IF (IND(JL) .GE. I) GO TO 110
      JL = JL + 1
      GO TO 100
  110 INDU(P) = JL - 1
      GO TO 160
  120 IL(P) = K
      IU(P) = J
      J = L
  130 IF (JL .GT. JU) GO TO 150
      IF (IND(JU) .LE. J) GO TO 140
      JU = JU - 1
      GO TO 130
  140 INDL(P) = JU + 1
      GO TO 160
  150 M = M - 1
      IF (M .EQ. 0) RETURN
      I = IL(M)
      J = IU(M)
      JL = INDL(M)
      JU = INDU(M)
      IF (JL .GT. JU) GO TO 150
  160 IF (J - I .GT. 10) GO TO 50
      IF (I .EQ. 1) GO TO 40
      I = I - 1
  170 I = I + 1
      IF (I .EQ. J) GO TO 150
      T = A(I + 1)
      IF (A(I) .LT. T) GO TO 170
      K = I
  180 A(K + 1) = A(K)
      K = K - 1
      IF (T .LT. A(K)) GO TO 180
      A(K + 1) = T
      GO TO 170
      END
```

```
$STORAGE:2
      REAL*4 FUNCTION DRAND(DSEED)
C
C  PURPOSE:
C
C  DRAND PRODUCES A UNIFORM RANDOM NUMBER BETWEEN 0 AND 1
C
C  INPUT :
C
C  DSEED  =  REAL*8 VARIABLE CONTAINING THE SEED VALUE
C
C  OUTPUT :
C
C  DRAND  =  REAL*8 VARIABLE CONTAINING THE PSEUDORANDOM
C     UNIFORM NUMBER BETWEEN 0 AND 1
C
      REAL*8 DSEED,D1,D2,D3
      DATA D1,D2,D3/2147483647.D0,2147483648.D0,16807.D0/
      DSEED=DMOD(D3*DSEED,D1)
      DRAND=SNGL(DSEED/D2)
      RETURN
      END
```

```
$STORAGE:2
      REAL*8 FUNCTION RNORM(U,S,DSEED)
C
C  PURPOSE:
C
C  RNORM PRODUCES A NORMAL VARIATE OF MEAN U AND STD DEV S
C
C  INPUT :
C
C  U = REAL*8 VARIABLE CONTAINING THE MEAN VALUE OF
C       A VARIABLE
C  S = REAL*8 VARIABLE CONTAINING THE STANDARD DEVIATION
C       OF A VARIABLE
C  DSEED = REAL*8 VARIABLE CONTAINING THE SEED VALUE
C
C  OUTPUT :
C
C  RNORM = REAL*8 NORMAL VARIATE VALUE OF MEAN U
C      AND STANDARD DEVIATION S
C  U,S = UNCHANGED IN OUTPUT
C  DSEED = CHANGED EVERY OTHER TIME IT IS PASSED INTO RNORM
C
C  ROUTINE REQUIRED:
C
C      REAL*4 FUNCTION DRAND
C  DRAND RETURNS A UNIFORM RANDOM NUMBER BETWEEN 0 AND 1
C
C
C  ALGORITHM FOR RNORM MAY BE REFERENCED ON PAGE 713 IN
C  'INTRODUCTION TO SIMULATION AND SLAM II',THIRD EDITION
C   A. ALAN B. PRITSKER
C   JOHN WILEY & SONS, 1986
C
      REAL*8 DSEED,U,S,A,B,W
      REAL*4 DRAND
      LOGICAL*2 SW
      DATA SW/.FALSE./
      IF(SW)GO TO 2
    1 A=2.D0*DBLE(DRAND(DSEED))-1.D0
      B=2.D0*DBLE(DRAND(DSEED))-1.D0
      W=A*A+B*B
      IF(W.GT.1.D0)GO TO 1
      W=DSQRT(-2.D0*DLOG(W)/W)
      RNORM=A*W*S+U
      SW=.TRUE.
      RETURN
    2 RNORM=B*W*S+U
      SW=.FALSE.
      RETURN
      END
```

```
$STORAGE:2
      REAL*8 FUNCTION RLOGN(U,S,DSEED)
C
C  PURPOSE:
C
C  RLOGN PRODUCES A LOGNORMAL VARIATE OF MEAN U AND STD DEV S
C
C  INPUT :
C
C  U = REAL*8 VARIABLE CONTAINING THE MEAN VALUE OF
C       A VARIABLE
C  S = REAL*8 VARIABLE CONTAINING THE STANDARD DEVIATION
C       OF A VARIABLE
C  DSEED = REAL*8 VARIABLE CONTAINING THE SEED VALUE
C
C  OUTPUT :
C
C  RLOGN = REAL*8 LOGNORMAL VARIATE VALUE OF MEAN U
C      AND STANDARD DEVIATION S
C  U,S = UNCHANGED IN OUTPUT
C  DSEED = CHANGED INSIDE OF FUNCTION RNORM EVERY OTHER
C       TIME IT IS PASSED INTO RNORM
C
C  ROUTINES REQUIRED :
C
C      REAL*8 FUNCTION RNORM
C  RNORM RETURNS A REAL*8 NORMAL VARIATE VALUE OF
C  MEAN U AND STANDARD DEVIATION S
C
C
C  ALGORITHM FOR RLOGN MAY BE REFERENCED ON PAGE 714 IN
C  'INTRODUCTION TO SIMULATION AND SLAM II',THIRD EDITION
C   A. ALAN B. PRITSKER
C   JOHN WILEY & SONS, 1986
C
C
      REAL*8 U,S,DSEED,RLGN,RNORM
      RLGN=DLOG(((S*S)/(U*U))+1.D0)
      RLGN=RNORM(DLOG(U)-RLGN/2.D0,DSQRT(RLGN),DSEED)
      RLOGN=DEXP(RLGN)
      RETURN
      END
      REAL*8 FUNCTION DDERIV(DFCN, KORDER, X, DBGSTEP, DTOL)
C
C  PURPOSE:
C
C  THIS IMSL FUNCTION RETURNS THE FIRST, SECOND OR THIRD
C  DERIVATIVE OF A USER-SUPPLIED FUNCTION.
C
C  VARIABLES:
C
C DFCN=REAL*8 FUNCTION WHOSE DERIVATIVE AT X WILL BE
C COMPUTED THE FORM IS DFCN(X) WHERE X IS THE INDEPENDENT
C VARIABLE PASSED INTO FUNCTION DFCN
C KORDER=INTEGER*4 VARIABLE CONTAINING THE ORDER OF THE
C DESIRED DERIVATIVE (1, 2 OR 3)
C X=REAL*8 VARIABLE CONTAINING THE POINT AT WHICH THE
C DERIVATIVE IS TO BE EVALUATED
C  DBGSTEP = REAL*8 VARIABLE CONTAINING THE BEGINNING STEP SIZE
C     USED IN THE FINITE DIFFERENCE SCHEME.
C  DTOL=REAL*8 VARIABLE CONTAINING THE RELATIVE ERROR DESIRED
C     IN THE DERIVATIVE ESTIMATE
      RETURN
      END
      SUBROUTINE ERSET(IERSVR, IPACT, ISACT)
C
C PURPOSE:
C
C THIS IMSL SUBROUTINE IS USED TO ALLOW THE USER TO INTERACT
C WITH THE IMSL ERROR HANDLING SYSTEM BY SETTING THE ACTIONS
C TO BE TAKEN WHEN ERRORS OCCUR
C
C VARIABLES:
C
C IERSVR = INTEGER*4 VARIABLE CONTAINING THE ERROR SEVERITY
C     LEVEL INDICATOR.
C     IF IERSVR = 0, ACTIONS ARE SET FOR LEVELS 1 TO 5.
C     IF IERSVR IS 1 TO 5, ERRORS ARE SET FOR ERRORS OF
C     THE SPECIFIED LEVEL.
C IPACT = INTEGER*4 VARIABLE CONTAINING THE PRINTING ACTION
C     IPACT ACTION:
C     -1   DO NOT CHANGE CURRENT SETTING(S)
C      0   DO NOT PRINT
C      1   PRINT
C      2   RESTORE THE DEFAULT SETTING(S)
C ISACT = INTEGER*4 VARIABLE CONTAINING THE STOPPING ACTION
C     ISACT ACTION:
C     -1   DO NOT CHANGE CURRENT SETTING(S)
C      0   DO NOT STOP
C      1   STOP
C      2   RESTORE THE DEFAULT SETTING(S)
      RETURN
      END
      INTEGER*4 FUNCTION IERCD()
C
C  PURPOSE:
C
C  THIS IMSL FUNCTION RETRIEVES THE INTEGER CODE FOR AN
C  INFORMATIONAL ERROR.  SINCE IT HAS NO ARGUMENTS, IT MAY
C  BE USED IN THE FOLLOWING WAY:  ICODE=IERCD()
C
C  INFORMATIONAL ERRORS WHEN USING DDERIV:
C  IF IERCD()=1, UNABLE TO ACHIEVE DESIRED TOLERANCE IN
C              DERIVATIVE ESTIMATION.
C  IF IERCD()=2, ROUNDOFF ERROR BECAME DOMINANT BEFORE
C              ESTIMATES CONVERGED
C
      RETURN
      END
```

Appendix D - Description of REH2

This appendix is used to describe the loose figures in the back of the thesis. Figures D.1 and D.2 both are used to describe subroutine REH2. Figure D.1 is a flow diagram for the REH2 subroutine. This flow diagram is not always self-explanatory; it was made to follow the FORTRAN source code and possibly allow future modifications more easily. Figure D.1 is made on two separate sheets due to its size. The possible cases are shown on the diagrams as well as in the source code. Figure D.2 shows drawings of all of the cases. The cases are grouped using the letters A, B, C and D. The cases with the 'A' classification are used to represent all of the possible overburden configurations. These cases do not consider the interburden at all. The cases with the 'B' classification represent when the dragline is working on the interburden and an extended bench is not required. 'C' cases represent the dragline operating on an extended bench and only requiring one spoil peak. 'D' cases represent the dragline when it operates on the extended bench and requires two spoil peaks.

```
      DOUBLE PRECISION FUNCTION REC(X, N)
C
C PURPOSE:
C
C This REAL*8 function is used to calculate the required
C dragline operating radius so that no rehandle occurs.
C This function is to be used for a single horizontal seam
C and it is assumed that the spoil does not ride up the
C vertical coal face.
C
C INPUT:
C
C N = INTEGER*2 variable containing the number of variables
C     used in the function. For this function: N=10
C X = REAL*8 vector of length N whose elements represent
C     the independent variables used in the function.
C     The maximum length allowed for this vector is 20.
C
C VARIABLES IN COMMON BLOCK FUNC:
C
C FRNM = character string of length 15 containing the name
C        of the function
C FRUNTS = character string of length 15 containing the units
C        of the function
C FACT = character string vector of length 20 containing the
C        variable names which can be up to 28 characters long.
C
C INTERNAL VARIABLE:
C
C SW = LOGICAL*2 variable which is initially set .FALSE. so
C    all of the variable values and names will be defined
C    during the first call to REC. In subsequent calls, SW
C    will be .TRUE. so that no more variable definition occurs.
C
C OUTPUT:
C
C Input is unchanged
C REC = REAL*8 variable containing the function value
C
C ROUTINES REQUIRED: none
C
C
      REAL*8 X(*)
      INTEGER*2 N
      CHARACTER*28 FACT(20)
      CHARACTER*15 FRNM
      CHARACTER*7 FRUNTS
      COMMON /FUNC/FRNM,FRUNTS,FACT
      LOGICAL*2 SW
      DATA SW/.FALSE./
C
      IF(SW)GO TO 1
      FACT(1)= 'cut width (m)'
      FACT(2)= 'spoil pit width (m)'
      FACT(3)= 'chopcut width (m)'
      FACT(4)= 'O/B thickness (m)'
      FACT(5)= 'chopcut thickness (m)'
      FACT(6)= 'spoil pile angle (deg)'
      FACT(7)= 'O/B highwall angle (deg)'
      FACT(8)= 'O/B swell factor'
      FACT(9)= 'O/B positioning factor'
      FACT(10)= 'tub diameter (m)'
      FRNM='required OROB'
      FRUNTS='meters'
      X(1)=50.D0
      X(2)=40.D0
      X(3)=50.D0
      X(4)=15.D0
      X(5)=5.D0
      X(6)=35.D0
      X(7)=70.D0
      X(8)=1.2D0
      X(9)=0.75D0
      X(10)=30.D0
      N=10
      SW=.TRUE.
C
    1 REC=X(9)*X(10) + X(4)/DTAN(.01745329251994D0*X(7))
     #   + X(8)/(X(2)*DTAN(.01745329251994D0*X(6)))*
     #   (X(1)*X(4)+X(5)*X(3)) + 0.25D0*X(2)
      RETURN
      END
```