PROTEST: A COMPUTER SYSTEM FOR THE ANALYSIS OF COMPUTATIONAL C-ETC(U)

UNCLASSIFIED

UNCLASSIFIED

CAAD-60-1
DISCLAIMER

The findings in this document are not be construed as an official Department of the Army position unless so designated by other authorized documents. Comments or suggestions should be addressed to:

Commander
USA Concepts Analysis Agency
ATTN: Director of Methodology and Computer Support
8120 Woodmont Avenue
Bethesda, Maryland 20014
This system assists the testing and debugging of computational computer programs by generating test points for the tested program at each test point. This information may be used to infer the operation of, or to predict output values of, an operational program as well. The performance data highlighting input variable sensitivities is analyzed further by one of the components of the system.
This writeup contains complete documentation of the system, including description of the methodology, sample outputs, flowcharts and listings. The package is written in UNIVAC's FORTRAN V.
PROGTEST: A COMPUTER SYSTEM FOR THE ANALYSIS OF
COMPUTATIONAL COMPUTER PROGRAMS

April 1980

Prepared by
Methodology and Computer Support Directorate
US Army Concepts Analysis Agency
8120 Woodmont Avenue
Bethesda, Maryland 20014
ABSTRACT

1. PROGTEST. This Computer System for the Analysis of Computational Computer Programs consists of three interrelated subsystems:

   POINTCOMP consisting of the major components POINTCOMP and VARVARY1
   ANALYZ containing the ANALYZ routine
   GRID consisting of the GRID, REARRANGE and DIFFQUOT routines as major components

2. POINTCOMP

   a. The POINTCOMP routine of the POINTCOMP subsystem utilizes the method of steepest ascent/descent to generate input variable values maximizing/minimizing the output of the program being tested (one output at a time). The points generated by the routine are printed out and may be used by the analyst to determine whether increasing or decreasing the variable increases/decreases the output, or whether a maximum/minimum is reached. The routine also prints out sensitivity information usable by the analyst in determining the comparative effect of each variable upon the output variable. Points generated by this routine are used as input by some of the other subsystems.

   b. The VARVARY1 component of the POINTCOMP subsystem uses input variable values produced by POINTCOMP or chosen by the analyst. For each combination produced or chosen by the analyst, each variable is varied uniformly through a range while keeping the other variables fixed. The tested routine is evaluated at each new point and the gradient is computed at each new point. In addition, statistics showing the average marginal return over various portions of the variables range are printed out. The output of this program can be used to generate curves showing the effect of the variable being varied upon the output.

3. ANALYZ. The ANALYZ subsystem uses a file of gradients as input. This type of file is produced by several subsystems. The subsystem computes and outputs sensitivity statistics derived from the gradients.
4. GRID. The GRID routine of the GRID subsystem utilizes a user defined grid to intensively analyze a small area of input values. The subsystem evaluates the routine to be tested at each node in the grid, and prints out the input values, the evaluated values, and the gradients at each point. The REARRANGE routine of the GRID subsystem rearranges the data to show the variable by variable variation, and the DIFFQUOT routine computes difference quotients for the generated points for one variable at a time.
# CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>OUTLINE OF SUBSYSTEMS</td>
<td>1-1</td>
</tr>
<tr>
<td></td>
<td>Subsystem Descriptions</td>
<td>1-1</td>
</tr>
<tr>
<td></td>
<td>POINTCOMP Subsystem</td>
<td>1-1</td>
</tr>
<tr>
<td></td>
<td>ANALYZ Subsystem</td>
<td>1-2</td>
</tr>
<tr>
<td></td>
<td>GRID Subsystem</td>
<td>1-3</td>
</tr>
<tr>
<td></td>
<td>Description of Subsystems</td>
<td>1-5</td>
</tr>
<tr>
<td></td>
<td>Subsystem Inputs and Outputs</td>
<td>1-5</td>
</tr>
<tr>
<td></td>
<td>Standard File Structure</td>
<td>1-7</td>
</tr>
<tr>
<td>2</td>
<td>THE POINTCOMP SUBSYSTEM</td>
<td>2-1</td>
</tr>
<tr>
<td></td>
<td>Section I. THE POINTCOMP ROUTINE</td>
<td>2-1</td>
</tr>
<tr>
<td></td>
<td>Introduction</td>
<td>2-1</td>
</tr>
<tr>
<td></td>
<td>Background</td>
<td>2-1</td>
</tr>
<tr>
<td></td>
<td>Discussion of Methodology</td>
<td>2-3</td>
</tr>
<tr>
<td></td>
<td>Limitations</td>
<td>2-3</td>
</tr>
<tr>
<td></td>
<td>Run Setups</td>
<td>2-4</td>
</tr>
<tr>
<td></td>
<td>Output Descriptions and Sample Output</td>
<td>2-8</td>
</tr>
<tr>
<td></td>
<td>POINTCOMP Routine Listing</td>
<td>2-12</td>
</tr>
<tr>
<td></td>
<td>POINTCOMP Routine Flowchart</td>
<td>2-18</td>
</tr>
<tr>
<td></td>
<td>Section II. THE VARVARY1 ROUTINE</td>
<td>2-25</td>
</tr>
<tr>
<td></td>
<td>Introduction</td>
<td>2-25</td>
</tr>
<tr>
<td></td>
<td>Limitations</td>
<td>2-25</td>
</tr>
<tr>
<td></td>
<td>Run Setup</td>
<td>2-25</td>
</tr>
<tr>
<td></td>
<td>Output Description and Sample Output</td>
<td>2-29</td>
</tr>
<tr>
<td></td>
<td>VARVARY1 Routine Listing</td>
<td>2-38</td>
</tr>
<tr>
<td></td>
<td>VARVARY1 Routine Flowchart</td>
<td>2-40</td>
</tr>
<tr>
<td>3</td>
<td>THE ANALYZ SUBSYSTEM</td>
<td>3-1</td>
</tr>
<tr>
<td></td>
<td>Introduction</td>
<td>3-1</td>
</tr>
<tr>
<td></td>
<td>Discussion of Statistics</td>
<td>3-1</td>
</tr>
<tr>
<td></td>
<td>Limitations</td>
<td>3-1</td>
</tr>
<tr>
<td></td>
<td>Run Setups</td>
<td>3-1</td>
</tr>
<tr>
<td></td>
<td>Output Description and Sample Output</td>
<td>3-3</td>
</tr>
<tr>
<td></td>
<td>ANALYZ Routine Listing</td>
<td>3-6</td>
</tr>
<tr>
<td></td>
<td>ANALYZ Routine Flowchart</td>
<td>3-11</td>
</tr>
</tbody>
</table>
# CHAPTER 4 THE GRID SUBSYSTEM

## Section I. THE GRID ROUTINE
- Introduction .................................................. 4-1
- Limitations .................................................. 4-1
- Run Setups .................................................. 4-1
- Output Descriptions and Sample Output ..................... 4-4
- GRID Routine Listing ........................................ 4-8
- GRID Routine Flowchart .................................... 4-11

## Section II. THE REARRANGE ROUTINE
- Introduction .................................................. 4-16
- Limitations .................................................. 4-16
- Run Setups .................................................. 4-16
- Output Descriptions and Sample Output ..................... 4-19
- REARRANGE Routine Listing ................................ 4-23
- REARRANGE Routine Flowchart ............................... 4-25

## Section III. THE DIFFQUOT ROUTINE
- Introduction .................................................. 4-29
- Background .................................................... 4-29
- Limitations .................................................. 4-29
- Run Setups .................................................. 4-29
- DIFFQUOT Routine Listing .................................. 4-33
- DIFFQUOT Routine Flowchart ............................... 4-35

# CHAPTER 5 COMMON SUBROUTINES

## Section I. THE PARTL SUBROUTINE
- Introduction .................................................. 5-1
- Background .................................................... 5-1
- Discussion of Methodology .................................. 5-1
- Limitations .................................................. 5-2
- Calling Sequence ............................................. 5-2
- PARTL Subroutine Listing ................................... 5-3
- PARTL Subroutine Flowchart ............................... 5-4

## Section II. THE PREPR DRIVER SUBROUTINE
- Introduction .................................................. 5-6
- Discussion .................................................... 5-6
- PREPR 'Layout' ............................................... 5-6

# APPENDIX

## A CONTRIBUTORS .......................................... A-1

## B DISTRIBUTION ........................................... B-1
CHAPTER 1

OUTLINE OF SUBSYSTEMS

1-1. SUBSYSTEM DESCRIPTIONS

a. POINTCOMP Subsystem

(1) POINTCOMP. The POINTCOMP routine of the POINTCOMP subsystem creates input variable values by using maximum ascent/descent techniques to maximize/minimize the output of the program being tested. The output from POINTCOMP includes:

(a) A file in standard format for which data lines are comprised of variable values and the corresponding value of the program being tested. This file is created on FORTRAN unit 10 and is one possible input to VARVARY1. FORTRAN allows usage of a disk file as on I/O unit.

(b) A file of the corresponding gradients in standard format on FORTRAN unit 11.

(c) A listing containing the new points and the corresponding program values, the gradients, and the computed increments is produced on the printer.

(d) The subsystem includes the following routines:

1. POINTCOMP.
2. PARTL.
3. PREPR.
4. The program to be tested.

(2) VARVARY1. The VARVARY1 component of the POINTCOMP subsystem takes the file created by POINTCOMP on unit 10 as input or a similarly structured user provided file. The user must also input a file containing higher and lower bounds and increments for each variable. VARVARY1 will read in each line of the unit 10 file and from each point inputted VARVARY1 creates new points by varying each variable in turn through the given range using the
given increment keeping the other values fixed. For each new point, the program to be tested is evaluated and the point and the corresponding tested program value are printed out.

(3) In addition, the VARVARY1 routine computes the gradient at each new point and prints its components. The new points and the corresponding tested routine values are written out on unit 12 and the corresponding gradients are written out on unit 11. When one variable is varied and the others held constant, the difference in the output values divided by the difference in the varied variable values is called the difference quotient. As the VARVARY1 routine varies a variable through its range, various combinations of difference quotients are computed and printed for that variable.

(4) The output from the VARVARY1 routine includes:

(a) A listing of the new input points and the associated output values of the routine being tested. This listing also contains the difference quotients.

(b) A listing of the corresponding gradients.

(c) A file containing the new gradients in standard format on unit 11.

(d) A file containing the new generated points in standard format on unit 12.

(e) A scratch file containing input variable values and the associated output values on unit 14.

(f) The subsystem includes the following routines:

1. VARVARY1.
2. PARTL.
3. PREPR.
4. The program being tested.

b. ANALYZ Subsystem. This subsystem analyzes a file of gradients in standard format on unit 11 produced by the POINTCOMP, VARVARY1, or GRID subsystem. The statistics produced include:

(1) For each gradient, the ratio of every component to the minimum component in absolute value.
(2) For each component of each gradient the change needed in the corresponding variable in order to change the output by one unit.

(3) For each component of each gradient:
   
   (a) If the component is not the largest in absolute value, the ratio of the component to the absolute value of the maximal component.

   (b) For the maximal component, the ratio of the absolute value of the maximal component to the absolute value of the next largest components.

(4) The gradients are also aggregated component by component, and statistics (1) to (3) described above are computed for the aggregate vector.

(5) The output from ANALYZ includes a listing of the statistics described above.

(6) The subsystem includes the following routine: ANALYZ.

c. GRID Subsystem. This subsystem is comprised of three linked but independent components--GRID, REARRANGE, and DIFFQUOT.

   (1) GRID. The GRID subsystem needs a range and step size for each variable. The subsystem varies each variable from the lower bound to the upper bound, incrementing each value by a given step size to obtain the next value, thereby producing all combinations of variable values. The routine to be tested is evaluated at each point and the gradient is computed at each point as well.

   (a) The outputs from GRID include:

      1. A listing of the new points and the associated routine values.

      2. A listing of the gradients.

      3. A file in standard format containing the new points and the associated values on unit 10. This file is one possible input to REARRANGE.

      4. A file in standard format containing the corresponding gradients on unit 11. This file is the other possible input to REARRANGE.
5. A scratch file containing the new points, the associated values and some pointers on unit 12. This file is used as an input to REARRANGE.

6. A scratch file containing the gradients and tested routine values on unit 13.

(b) The components of this subsystem are:

1. GRID.
2. PARTL.
3. PREPR.
4. The program to be tested.

(2) REARRANGE. This routine reads from unit 14, one of two files produced by GRID on units 12 or 13. The routine also utilizes the file produced by GRID on unit 12. The output data is rearranged variable by variable to facilitate further analyses.

(a) The output from REARRANGE includes:

1. A listing of the rearranged input file, where adjoining lines in the same section differ only in one variable value, each point is numbered to facilitate linkage of this output to the GRID output.

2. A file containing similar information produced on unit 15.

(b) The subsystem has one component: REARRANGE.

(3) DIFFQUOT. This routine reads the GRID output files produced on units 10 and 12. The routine outputs a list of difference quotients for the variable requested. The outputted quotients are headed by the numbers of the relevant GRID points, facilitating the linkage between the GRID output and this output. The subsystem has one component: DIFFQUOT.
1-2. DESCRIPTION OF SUBSYSTEMS. Subsystem components are shown in Table 1-1.

Table 1-1. Subsystem Composition

<table>
<thead>
<tr>
<th>Subsystem</th>
<th>Major components</th>
<th>Minor components</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINTCOMP</td>
<td>POINTCOMP</td>
<td>PARTL</td>
</tr>
<tr>
<td></td>
<td>VARVARY1</td>
<td>PREPR</td>
</tr>
<tr>
<td>ANALYZ</td>
<td>ANALYZ</td>
<td>--</td>
</tr>
<tr>
<td>GRID</td>
<td>GRID</td>
<td>PARTL</td>
</tr>
<tr>
<td></td>
<td>REARRANGE</td>
<td>PREPR</td>
</tr>
<tr>
<td></td>
<td>DIFFQUOT</td>
<td></td>
</tr>
</tbody>
</table>

1-3. SUBSYSTEM INPUTS AND OUTPUTS. Figure 1-1 and Table 1-2 show inputs, outputs, and intercommunication between the subsystems.

NOTES: POINTCOMP, VARVARY1, and GRID routines each may be run independently of the other major routines. The other major routines may be run independently once POINTCOMP or GRID have been run to get up the required input files, as indicated above.

The head of the arrow denotes input; the tail of the arrow, output.

Figure 1-1. Data Flow
Table 1-2. Unit Utilization (all routines utilize Units 5 and 6 in addition to those listed below.)

<table>
<thead>
<tr>
<th>Routine</th>
<th>Unit number</th>
<th>Type</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINTCOMP</td>
<td>10</td>
<td>Output</td>
<td>Standard</td>
<td>Each line contains input variable values and the corresponding value.</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>Output</td>
<td>Standard</td>
<td>Each line contains the corresponding gradients.</td>
</tr>
<tr>
<td>VARVARY1</td>
<td>10</td>
<td>Input</td>
<td>Standard</td>
<td>File created by POINTCOMP, containing the variable values.</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>Output</td>
<td>Standard</td>
<td>Gradients corresponding to new points generated by VARVARY1.</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>Output</td>
<td>Standard</td>
<td>New points generated by VARVARY1 and the associated value of the tested routine.</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>Output</td>
<td>Nonstandard</td>
<td>Scratch file containing gradients.</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>Output</td>
<td>Nonstandard</td>
<td>Scratch file containing input variable values.</td>
</tr>
<tr>
<td>ANALYZ</td>
<td>11</td>
<td>Input</td>
<td>Standard</td>
<td>Gradient file created by POINTCOMP, VARVARY1, or GRID</td>
</tr>
<tr>
<td>GRID</td>
<td>10</td>
<td>Output</td>
<td>Standard</td>
<td>File containing input variable values and the associated values of the tested routine.</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>Output</td>
<td>Standard</td>
<td>File containing the corresponding gradients.</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>Output</td>
<td>Nonstandard</td>
<td>Scratch file containing the input variable values tested routine value and a pointer to the variable varied to create the point. Used as input to REARRANGE.</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>Output</td>
<td>Nonstandard</td>
<td>Scratch file containing the corresponding gradients and the value of the tested routine. Used as input to REARRANGE.</td>
</tr>
<tr>
<td>REARRANGE</td>
<td>10</td>
<td>Input</td>
<td>Standard</td>
<td>File produced by GRID, containing the input variable values and tested routine value on Unit 12, containing the indication of which variables were varied.</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>Input</td>
<td>Nonstandard</td>
<td>File produced by GRID on Unit 12, containing the indication of which variable was varied. Used in the reordering computations by REARRANGE.</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>Input</td>
<td>Nonstandard</td>
<td>The file to be reordered, this file could be the file produced by GRID on Units 12 or 13.</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>Output</td>
<td>Nonstandard</td>
<td>A file containing the rearranged output.</td>
</tr>
<tr>
<td>DIFFQUOT</td>
<td>10</td>
<td>Input</td>
<td>Standard</td>
<td>File containing input variable values, produced by GRID.</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>Input</td>
<td>Nonstandard</td>
<td>File produced by GRID, indicating which variables were varied to create each point.</td>
</tr>
</tbody>
</table>
The following (Figure 1-2) outlines the units utilized by the different routines.

NOTE: Numbers denote I/O unit numbers; primes denote same unit but different formats.

Figure 1-2. Subsystem Unit Utilization

1-4. STANDARD FILE STRUCTURE. A standard file has the following structure:

a. Line 1 is Comprised of Fields A1, A2, and A3
   (1) A1 is the number of points or gradients in I5 format.
   (2) A2 is the number of variables in I3 format.
   (3) A3 is a positive number in F10.7 format. Any vector components smaller than this number in absolute value will be considered to be zero.

b. Line 2 (format). This line indicates the format that was used to write each data line in the file and which may be used to read the subsequent lines in the file.

c. Other Lines. These lines contain data consisting either of variable value vectors (points) or gradients. The leftmost column gives data on variable 1, the next column to the right on variable 2, etc.
CHAPTER 2
THE POINTCOMP SUBSYSTEM

Section I. THE POINTCOMP ROUTINE

2-1. INTRODUCTION. This routine uses the gradient to pick new points which minimize/maximize (locally) the value of the output of the routine being tested. In addition, the gradient at each point is also printed out. The analyst may use the output to check out the routine being tested in the following manner:

a. The analyst can examine each variable to see if it must be increased/decreased to increase/decrease the output. The analyst can also check for attainment of values from which no increase/decrease of the output variable is possible. These results should be explainable intuitively.

b. The analyst may examine each gradient to determine the relative effect of each variable at that point.

c. The analyst may examine the components of the gradients for all points to analyze the marginal returns.

d. Changes in the sign of a component of a gradient from one point to the next indicate that a potential local optimum for that variable lies between the two successive values of the coordinate.

e. POINTCOMP calls PARTL to compute partial derivatives numerically. POINTCOMP calls PREPR, a user provided driver routine which obtains an output value from the program being tested.

2-2. BACKGROUND
If \( f(x_1, \ldots, x_n) \) is a real valued function of several variables, \( \nabla f(x_1, \ldots, x_n) = \left( \frac{\partial f}{\partial x_1}(x_1, \ldots, x_n), \ldots, \frac{\partial f}{\partial x_n}(x_1, \ldots, x_n) \right) \) points in the direction of greatest local increase in \( f \) from \( (x_1, \ldots, x_n) \), and \( -\nabla f(x_1, \ldots, x_n) \) points in the direction of greatest local decrease in \( f \) from \( (x_1, \ldots, x_n) \). We justify this fact by the following argument:
If \((y_1, \ldots, y_n)\) is a point "close" to \((x_1, \ldots, x_n)\),
f\((y_1, \ldots, y_n) - f(x_1, \ldots, x_n)\) is approximated by:

\[
\sum_{i=1}^{n} \frac{\partial f}{\partial x_i} (x_1, \ldots, x_n) (y_i - x_i) = \nabla f(x_1, \ldots, x_n) \cdot (y - x) = ||\nabla f(x_1, \ldots, x_n)|| \cdot ||y - x|| \cos \theta
\]

where: \(y = (y_1, \ldots, y_n)\);
\(x = (x_1, \ldots, x_n)\);

\(||A||\) is the magnitude of a vector \(A\); and

\(\theta\) is the angle between \(\nabla f(x_1, \ldots, x_n)\) and \((y - x)\).

So

\[
f(y_1, \ldots, y_n) - f(x_1, \ldots, x_n) = ||\nabla f(x_1, \ldots, x_n)|| \cdot ||y - x|| \cos \theta,
\]

and therefore:

\[
\frac{f(y_1, \ldots, y_n) - f(x_1, \ldots, x_n)}{||y - x||} = ||\nabla f(x_1, \ldots, x_n)|| \cos \theta
\]

Now \(-1 \leq \cos \theta \leq 1\), so

\[
\max_{||y - x|| \neq 0} \frac{f(y_1, \ldots, y_n) - f(x_1, \ldots, x_n)}{||y - x||} = ||\nabla f(x_1, \ldots, x_n)||.
\]

Therefore the greatest change in the function \(f\) per unit distance between \(y\) and \(x\) (i.e., \(||y - x||\)) is the magnitude of the gradient \(||\nabla f(x_1, \ldots, x_n)||\). This maximum is reached for \(\cos \theta = 1\) or \(\theta = 0\), so the direction of greatest increase is \(\theta = 0\), so \(y - x\) and \(f(x_1, \ldots, x_n)\) are collinear. Therefore, \(\nabla f(x_1, \ldots, x_n)\) gives both the magnitude and direction of the maximal change.

Likewise:

\[
\min_{||y - x|| \neq 0} \frac{f(y_1, \ldots, y_n) - f(x_1, \ldots, x_n)}{||y - x||} = -||\nabla f(x_1, \ldots, x_n)||
\]
In this case, \( \cos \theta = -1 \), so \( \theta = 180^\circ \) and the direction of greatest decrease (least increase) per unit distance between \( y \) and \( x \) (\( ||y - x|| \)) is \( -\nabla f(x_1, \ldots, x_n) \), the magnitude of greatest decrease is \( ||\nabla f(x_1, \ldots, x_n)|| \).

2-3. DISCUSSION OF METHODOLOGY. Several criteria are used in the program to decide upon the distance to be moved along the gradient in order to obtain the next point. These criteria are:

a. The maximum change in one step for each variable (an input parameter) divided by the size of the component of the gradient corresponding to that variable.

b. First, compute a certain fraction of the difference between the current variable value and the bound on the variable in the appropriate direction (an input parameter) whenever the bound exists. When the variable isn't bounded, use twice the current value as a bound. Next, the number just described is divided by the size of the corresponding component of the gradient.

c. Compute the minimum of the numbers developed in parts a and b and the default multiplier read in as an input parameter on the 8th input card (see pg 2-6). This step computes a gradient multiplier.

d. A candidate increment is determined by multiplying the gradient by the number derived by the process described in c. If the gradient multiplier is too small, use the A7 field on card 1. If necessary, the increment is multiplied by a number sufficient to increase each component to the minimum threshold. These minimum thresholds are also input.

e. Lastly, the increment vector is added to the original point, obtaining a new point candidate. The components of this new point exceeding the bounds of the corresponding variable (if any) are set equal to the value of the bound, and the result is taken to be the next point.

2-4. LIMITATIONS. As currently compiled, the POINTCOMP routine is subject to the following limitations:

a. Testing of the program must proceed by testing one output variable at a time if the program has several output variables.

b. If any variables input to the program being tested are integers, it is possible that their incrementation and effect upon the output variables will be reduced due to truncation.
c. Current limits are set to 20 variables and 500 generated points at most. This is easily changed.

d. Testing Monte Carlo programs is feasible, but potentially time consuming. The PREPR routine, called by POINTCOMP, must call the routine being tested repetitively and average the output values before returning. Replication is necessary in order to average out random effects. A better approach is to use the Monte Carlo variables as ordinary input variables and let POINTCOMP assign their values with no replication.

e. The function represented by the routine to be tested should be well-behaved (i.e., differentiable).

f. The order in which the criteria for step size are examined is fixed, so there is a hard wired priority among the criteria.

2-5. RUN SETUPS

a. Developing an Absolute File in ASCII

```
@MAP,S
IN O3PROGTEST. POINTCOMP
IN O3PROGTEST. PARTIAL
IN Element containing PREPR subroutine
IN Programs to be tested
LIB LIB$*FTNG.
END
```

b. To Execute

```
@USE 10, file name to contain variable values and values of the tested routine
@USE 11, data file name for gradient file in standard format
@ASG,A First file name
@ASG,A Second file name
@XQT Absolute element created by the @MAP
Input
Deck
```

c. Input Deck. The following tables describe the input deck (Table 2-1), show a sample input deck (Table 2-2), and a sample run deck (Table 2-3).
Table 2-1. Description of Input Deck
(page 1 of 2 pages)

Line 1

Field: A1  A2  A3  A4  A5  A6  A7  A8

Format: I3  I3  I2  F10.7 F10.7 F12.0 F12.5 I3

where the fields are defined as follows:

A1 is the maximum number of points to be generated.
A2 is the number of variables.
A3 is +1 if output is to be increased, -1 if output is to be decreased.
A4 is a positive real number. Any number smaller than this number will be considered to be zero.
A5 is a positive real number, gradients smaller than this number will be considered to be zero.
A6 is a code number. This number can be used to indicate that a variable is unbounded above or below.
A7 describes the minimal multiplier desired for the gradient.
A8 the run is executed in a debugging mode (more printout) when this field contains a -1.

Line 2. (Format for reading in each of the following six lines, lines 3-8 which follow. Parentheses must bound the format as shown.)

Line 3. Initial values, one for each variable, in the format described by line 2.
Table 2-3. Description of Input Deck
(page 2 of 2 pages)

Line 4. Lower bounds allowed for each variable, in the format described by line 2. If some variable is unbounded below, indicate this by using the number from field A6 of the first line.

Line 5. Upper bounds allowed for each variable. Rest of description as in line 4.

Line 6. The preferred maximum single step change in each variable, one number for each variable.

Line 7. The preferred minimum single step change in each variable, one for each variable.

Line 8. Default gradient multiplier for each variable, one number for each variable. These will only be used for unbounded variables and are only useful to reduce the number by which the gradient will be multiplied.

Line 9. (Format for writing one line of variable values and the corresponding program output value on unit 10. This format should accommodate at least n+3 values, where n is the number of variables.)

Line 10. (Format for writing out one line of variable values and the corresponding program output value on the printer.)
Table 2-2. Sample Input Data

<table>
<thead>
<tr>
<th></th>
<th>4a</th>
<th>gb+1c</th>
<th>.001d</th>
<th>.001e</th>
<th>1001.f</th>
<th>.1000g+1h</th>
</tr>
</thead>
<tbody>
<tr>
<td>(13F5.0) format for inputting the following six lines</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.0</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>Initial value</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>Lower bounds</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
<td>0.9</td>
<td>Upper bounds</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10.0</td>
<td>10.0</td>
<td>10.0</td>
<td>0.5</td>
<td>Preferred maximum step</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>Preferred minimum step</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>Default multiplier</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(not really needed for bounded variables)</td>
<td></td>
</tr>
</tbody>
</table>

First column describes variable 1.
Second column describes variable 2.
Etc.

(6(7X,F13.5)) Format for writing values and output on unit 10

(1X,6(fX,F13.5)) Format for writing values and output on the printer

---

a Number of variables.
b Maximum number of points.
c Increase the output.
d Numbers lower than this are considered zero.
e Gradients smaller than this are considered zero.
f Code number indicating no bounds.
g Minimal gradient multiplier.
h Not in debug mode.
Table 2-3. Sample Run Deck

@USE 10,03MAT1.
@ASG,A 03MAT1.
@USE 11,03MAT2.
@ASG,A 03MAT2.
@XQT absolute deck created by @MAP

<table>
<thead>
<tr>
<th># # # # +1</th>
<th>.001</th>
<th>.001</th>
<th>1001.</th>
<th>.1000 +1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(13F6.0)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.0</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
<td>0.9</td>
<td></td>
</tr>
<tr>
<td>10.0</td>
<td>10.0</td>
<td>10.0</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.01</td>
<td></td>
</tr>
</tbody>
</table>
(6(7X,F13.5))
(1X,6(7X,F13.5))

2-6. OUTPUT DESCRIPTIONS AND SAMPLE OUTPUT

a. Tuples. Each row corresponds to a point where the rightmost column is the value of the routine being tested and the leftmost column represents variable 1, the column to its right variable 2, etc.

b. Gradients. Each row is the gradient at the point described by the corresponding row of the tuples output, e.g., the first row is the gradient at the point described by the first row of the tuples printout.

c. Increments. Each row is the computed vector difference of the corresponding tuples (except where boundaries are encountered). The first row is the second tuple minus the first, the second row is the third tuple minus the second, etc. The following (Table 2-4) exhibits sample printer output.
### Table 2-4. Sample Output to the Printer

#### Tuples

<table>
<thead>
<tr>
<th>Value</th>
<th>Value2</th>
<th>Value3</th>
<th>Value4</th>
<th>Value5</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.00000</td>
<td>0.50000</td>
<td>0.50000</td>
<td>0.50000</td>
<td>0.73821</td>
</tr>
<tr>
<td>5.00000</td>
<td>0.61902</td>
<td>0.75809</td>
<td>0.63333</td>
<td>0.99916</td>
</tr>
<tr>
<td>5.00119</td>
<td>0.73820</td>
<td>0.97516</td>
<td>0.76667</td>
<td>1.24159</td>
</tr>
<tr>
<td>5.00270</td>
<td>0.80924</td>
<td>1.08781</td>
<td>0.84667</td>
<td>1.37902</td>
</tr>
<tr>
<td>5.00270</td>
<td>0.84061</td>
<td>1.13401</td>
<td>0.88222</td>
<td>1.43833</td>
</tr>
<tr>
<td>5.00270</td>
<td>0.85178</td>
<td>1.14995</td>
<td>0.89492</td>
<td>1.45926</td>
</tr>
<tr>
<td>5.00270</td>
<td>0.85513</td>
<td>1.15468</td>
<td>0.89873</td>
<td>1.46551</td>
</tr>
<tr>
<td>5.00270</td>
<td>0.85513</td>
<td>1.15590</td>
<td>0.89873</td>
<td>1.46626</td>
</tr>
<tr>
<td>5.00270</td>
<td>0.85513</td>
<td>1.15715</td>
<td>0.89975</td>
<td>1.46753</td>
</tr>
</tbody>
</table>

#### Gradients

<table>
<thead>
<tr>
<th>Value</th>
<th>Value2</th>
<th>Value3</th>
<th>Value4</th>
<th>Value5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00048</td>
<td>0.30919</td>
<td>0.67048</td>
<td>0.34638</td>
<td></td>
</tr>
<tr>
<td>0.00359</td>
<td>0.35940</td>
<td>0.65458</td>
<td>0.40207</td>
<td></td>
</tr>
<tr>
<td>0.00852</td>
<td>0.40086</td>
<td>0.63565</td>
<td>0.45144</td>
<td></td>
</tr>
<tr>
<td>0.01236</td>
<td>0.42295</td>
<td>0.62294</td>
<td>0.47942</td>
<td></td>
</tr>
<tr>
<td>0.01428</td>
<td>0.43221</td>
<td>0.61698</td>
<td>0.49139</td>
<td></td>
</tr>
<tr>
<td>0.01500</td>
<td>0.43545</td>
<td>0.61481</td>
<td>0.49560</td>
<td></td>
</tr>
<tr>
<td>0.01522</td>
<td>0.43641</td>
<td>0.61415</td>
<td>0.49686</td>
<td></td>
</tr>
<tr>
<td>0.01523</td>
<td>0.43631</td>
<td>0.61415</td>
<td>0.49674</td>
<td></td>
</tr>
<tr>
<td>0.01527</td>
<td>0.43670</td>
<td>0.61405</td>
<td>0.49663</td>
<td></td>
</tr>
</tbody>
</table>

#### Increments

<table>
<thead>
<tr>
<th>Value</th>
<th>Value2</th>
<th>Value3</th>
<th>Value4</th>
<th>Value5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00000</td>
<td>0.11902</td>
<td>0.25809</td>
<td>0.13333</td>
<td></td>
</tr>
<tr>
<td>0.00119</td>
<td>0.11919</td>
<td>0.21707</td>
<td>0.13333</td>
<td></td>
</tr>
<tr>
<td>0.00151</td>
<td>0.07104</td>
<td>0.11264</td>
<td>0.08000</td>
<td></td>
</tr>
<tr>
<td>0.00000</td>
<td>0.03137</td>
<td>0.04620</td>
<td>0.03556</td>
<td></td>
</tr>
<tr>
<td>0.00000</td>
<td>0.01117</td>
<td>0.01594</td>
<td>0.01270</td>
<td></td>
</tr>
<tr>
<td>0.00000</td>
<td>0.00335</td>
<td>0.00473</td>
<td>0.00381</td>
<td></td>
</tr>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00122</td>
<td>0.00000</td>
<td></td>
</tr>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00126</td>
<td>0.00102</td>
<td></td>
</tr>
</tbody>
</table>
d. Output File Descriptions

(1) File on Unit 10

(a) Line 1

<table>
<thead>
<tr>
<th>Fields</th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Formats</td>
<td>15</td>
<td>13</td>
<td>F10.7</td>
</tr>
</tbody>
</table>

where

A1 is the number of points in the file.

A2 is the number of variables.

A3 is the number determining when small numbers are considered to be zero.

(b) Line 2. (Format for reading in one point and its corresponding program value.)

(c) Other Lines. Each line contains a set of variable values and the corresponding tested program value.

(2) File on Unit 11

(a) Same as file on unit 10.

(b) Same as file on unit 10.

(c) Other lines - Each line contains the corresponding gradient.
e. Sample Output File--Unit 10

<table>
<thead>
<tr>
<th>9 4 .0010000 (6(7X,F13,5))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuples</td>
</tr>
<tr>
<td>5.00000 50000 50000 50000 73821</td>
</tr>
<tr>
<td>5.00000 61902 75809 63333 99916</td>
</tr>
<tr>
<td>5.00119 73820 97516 76667 124159</td>
</tr>
<tr>
<td>5.00270 80924 1.08781 84667 1.37902</td>
</tr>
<tr>
<td>5.00270 84061 1.13401 88222 1.43833</td>
</tr>
<tr>
<td>5.00270 85178 1.14995 89492 1.45926</td>
</tr>
<tr>
<td>5.00270 85513 1.15468 89873 1.46551</td>
</tr>
<tr>
<td>5.00270 85513 1.15590 89873 1.46626</td>
</tr>
<tr>
<td>5.00270 85513 1.15715 89975 1.46753</td>
</tr>
</tbody>
</table>

f. Sample Output File--Unit 11

<table>
<thead>
<tr>
<th>9 4 .0010000 (6(7X,F13,5))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00048 0.30919 0.67048 0.34638</td>
</tr>
<tr>
<td>0.00359 0.35940 0.65458 0.40207</td>
</tr>
<tr>
<td>0.00852 0.40086 0.63565 0.45144</td>
</tr>
<tr>
<td>0.01236 0.42295 0.62294 0.47942</td>
</tr>
<tr>
<td>0.01428 0.43221 0.61598 0.49139</td>
</tr>
<tr>
<td>0.01500 0.43545 0.61481 0.49560</td>
</tr>
<tr>
<td>0.01522 0.43641 0.61415 0.49686</td>
</tr>
<tr>
<td>0.01523 0.43631 0.61415 0.49674</td>
</tr>
<tr>
<td>0.01527 0.43670 0.61405 0.49663</td>
</tr>
</tbody>
</table>
2-7. POINTCOMP ROUTINE LISTING

PARAMETER FILE1=10, FILE2=11, NOVALS=20, NOPTS=500, INFILE=5, OUTFIL=6
PARAMETER NOVAL1=NOVALS+1

C CHARACTER*80 FORMAT1, FORMT2, FORMT3
C
DIMENSION FS1VAL(NOVALS), LOWBD(NOVALS), HYBDS(NOVALS)
DIMENSION MAXCHG(NOVALS), GRAD(NOPTS, NOVALS), TUPLES(NOPTS, NOVALS)
DIMENSION DELTA1(NOVALS), DELTA2(NOVALS), DELTA(NOPTS, NOVALS)
DIMENSION SIGN(NOVALS), INPUT(NOVALS), MINNOVALS, NOMULT(NOVALS)
C
INTEGER NOVARS, UPOWNS, UPDOWN, EPSLF, FLATNOLIM, EPSLON, FLATNOLIM, NOPTSL, CHG
REAL EPSLON, FLATNOLIM, FS1VAL, LOWBD, HYBDS, MAXCHG
REAL GRAD, SUM, INPUT, INCR, PARTY, DELTA1, DELTA2, DELTA
REAL TUPLES, VALUE, NOMULT, MIN, MIN1, MINMLT, MAX
C
REWIND FILE1
REWIND FILE2
READ(INFILE, 10000) NOVARS, MAXPTS, UPDOWN, EPSLON, FLATNOLIM, EPSLON, SWITCH
C
10000 FORMAT(2I3, 2F10.7, 2F12.0, 1I3)
C
**** IF (SWITCH NE. -1) GOTO 30010
WRITE(OUTFIL, 20060) NOVARS, MAXPTS, UPDOWN, EPSLON, FLATNOLIM, EPSLON, MINMLT
20060 FORMAT(15S, *NOVALS.., /1X, 2I3, 2F10.7, 2F12.0, 2I3)
30010 CONTINUE
C
**** READ(INFILE, 10010) FORMT1
10010 FORMAT(180)
READ(INFILE, FORMT1) (FS1VAL(I), I=1, NOVARSI)
READ(INFILE, FORMT1) (LOWBD(I), I=1, NOVARSI)
READ(INFILE, FORMT1) (HYBDS(I), I=1, NOVARSI)
READ(INFILE, FORMT1) (MAXCHG(I), I=1, NOVARSI)
READ(INFILE, FORMT1) (MIN1, I=1, NOVARSI)
READ(INFILE, FORMT1) (NOMULT(I), I=1, NOVARSI)
READ(INFILE, 10010) FORMT2
READ(INFILE, 10010) FORMT3
C
**** IF (SWITCH NE. -1) GOTO 30020
WRITE(OUTFIL, 21000) (FS1VAL(I), I=1, NOVARSI)
21000 FORMAT(15S, **FS1VAL** *10F/24, 1X, 6F20.10)
WRITE(OUTFIL, 21010) (LOWBD(I), I=1, NOVARSI)
21010 FORMAT(15S, **LOWBD** *10F/24, 1X, 6F20.10)
WRITE(OUTFIL, 21020) (HYBDS(I), I=1, NOVARSI)
21020 FORMAT(15S, **HYBDS** *10F/24, 1X, 6F20.10)
WRITE(OUTFIL, 21030) (MAXCHG(I), I=1, NOVARSI)
21030 FORMAT(15S, **MAXCHG** *10F/24, 1X, 6F20.10)
WRITE(OUTFIL, 20065) (MIN1, I=1, NOVARSI)
20065 FORMAT(15S, **MIN1** *10F/24, 1X, 6F20.10)
WRITE(OUTFIL, 20067) (NOMULT(I), I=1, NOVARSI)
20067 FORMAT(15S, **NOMULT** *10F/24, 1X, 6F20.10)
30020 CONTINUE
C
****
C  INITIALIZATION
   DO 100 I=1,NOVARS
   TUPLES(I,I)=FSTVAL(I)
   INPUT(I)=FSTVAL(I)
100  CONTINUE
   CALL PPR(INPVALUE)
   TUPLES(1,NOVAL1)=VALUE
C  LOOP TO CREATE POINTS
   DO 300 P=2,MAXPTS
   DO 300 I=1,NOVARS
   INPUT(I)=TUPLES(P-1,I)
300  CONTINUE
   DO 400 I=1,NOVARS
   CALL PARTLYNOVARS,INP,EPSON,PARTY,INCR,TUPLES(P-1,NOVAL1)
C  ****
   IF (SWITCH .NE. -1) GOTO 30030
   WRITE(OUTFIL,20000) (INPUT(J),J=1,NOVARS),PARTY,INCR,
   TUPLES(P-1,NOVAL1)
20000 FORMAT(///,...**INPUT PAR**ial***,10(/,1X,6F20.10))
30030 CONTINUE
C  ****
   GRAD(P-1,I)=PARTY
400  CONTINUE
   DO 500 I=1,NOVARS
   SIGN(I)=UPDOWN
   IF (GRAD(P-1,I) .LT. 0.) SIGN(I)=-UPDOWN
500  CONTINUE
C  ****
   IF (SWITCH .NE. -1) GOTO 30040
   WRITE(OUTFIL,20010) (GRAD(P-1,I),I=1,NOVARS)
20010 FORMAT(///,**GRAD***,**10(/,1X,6F20.10))
   WRITE(OUTFIL,20015) (SIGN(I),I=1,NOVARS)
20015 FORMAT(///,**SIGN***,**10(/,1X,10I3))
30040 CONTINUE
C  ****
C COMPUTE CANDIDATE MULTIPLIER BASED MACH CHANGE PER COORDINATE
   DO 600 I=1,NOVARS
   DELTA(I)=ABS(NOMULT(I))
   IF (ABS(GRAD(P-1,I)) .LT. EPSLON) GOTO 600
   DELTA(I)=ABS(MAXCHG(I))/ABS(GRAD(P-1,I))
600  CONTINUE
C  ****
   IF (SWITCH .NE. -1) GOTO 30050
   WRITE(OUTFIL,20020) (DELTA(I),I=1,NOVARS)
20020 FORMAT(///,**DELTA***,**10(/,1X,6F20.10))
30050 CONTINUE
C ******
C COMPUTE CANDIDATE MULTIPLIER BASED ON DISTANCE FROM BOUNDS
C OR DOUBLING THE VALUE IF NO BOUND EXISTS
DO 700 I=1,NVARS
FORMAT (A2,I=1,NVARS)
DELTA(I)=MIN1(ABS(TUPLES(I-1,I)/ABS(�I(P-1,I))*&NOVLUTI))
IF (SIGN(I)*LT. 0) GOTO 700
IF (ABS(�(I-NOLIM)) .GE. EPSLON) DELTA2(I)=
1 ((FLOAT(P)-1)/(FLOAT(P)-1))*(ABS(�(I)-TUPLES(I-1,I)))
2 /ABS(�(P-1,I))
GOTO 700
700 IF (ABS(LOWBSDS(I)-NOLIM)) .GE. EPSLON DELTA2(I)=
1 ((FLOAT(P)-1)/(FLOAT(P)-1))*(ABS(LOWBSDS(I)-TUPLES(P-1,I)))
2 /ABS(�(P-1,I))
800 CONTINUE
C ******
C IF (SWITCH .NE. -1) GOTO 30060
WRITE (OUTFIL,20046) (DELTA2(I),I=1,NVARS)
20046 FORMAT (//,* DELTA2**10(/1x,4F30.1))
30060 CONTINUE
C ******
C COMPUTE THE INCREMENTATION VECTOR
DO 900 I=1,NVARS
DELTA(I)=MIN1(DELTA1(I),DELTA2(I))
900 CONTINUE
C ******
C IF (SWITCH .NE. -1) GOTO 30070
WRITE (OUTFIL,2005) (DELTA(I-1,I),I=1,NVARS)
2005 FORMAT (//,* DELTA**10(/1x,4F30.1))
30070 CONTINUE
C ******
C TO COMPUTE THE MINIMUM GRADIENT MULTIPLIER.
MINI=DELTA(P-1,I)
DO 925 I=1,NVARS
IF (DELTA(I-1,I) .LT. EPSLON) GOTO 925
IF (DELTA(I-1,I) .LT. MIN1) MIN1=DELTA(P-1,I)
925 CONTINUE
C ******
C IF (SWITCH .NE. -1) GOTO 30080
WRITE (OUTFIL,20044) (DELTA(I-1,I),I=1,NVARS),MIN1
20044 FORMAT (//,* DELTA**10(/1x,6F20.10))
30080 CONTINUE
C ******
C TO COMPUTE A CANDIDATE INCREMENT
IF (MIN1 .LT. EPSLON) MIN1=MINMLT
DO 950 I=1,NVARS
DELTA(I-1,I)=MIN1*ABS(�(P-1,I))
950 CONTINUE
C ******
IF (SWITCH .NE. -1) GOTO 30090
WRITE(OUTFIL=20040) (DELTA(P-1*I)+I=1+NOVARS),MIN1
20040 FORMAT(/'**DELTA.2*MINI**.10F/1X/1X*6F20.10I))
30090 CONTINUE

C *****
C TO GUARANTEE A MINIMAL STEP IN EACH COORDINATE
MAX=0.
DO 990 I=1+NOVARS
C SET MINISCULLE STEPS TO ZERO
IF (DELTA(P-1*I)+GE. EPSLON) GOTO 980
DELTA(P-1*I)=0.
GOTO 990
980 CONTINUE
C CHECK TO SEE IF CANDIDATE INCREMENT EXCEEDS MINIMUM
990 IF (DELTA(P-1*I)+GE. MIN(I)) GOTO 990
C *****
IF (SWITCH .NE. -1) GOTO 30200
WRITE(OUTFIL=20200) I,MAX,MIN(I),ABS(DELTA(P-1*I))
20200 FORMAT(/'**MAX**.15F,15F4F20.10I))
30200 CONTINUE
C *****
IF ((MIN(I)/DELTA(P-1*I)+LE. MAX) GOTO 990
MAX=MIN(I)/DELTA(P-1*I)
990 CONTINUE

C INCREASE THE STEP SIZE
DO 970 I=1+NOVARS
DELTA(P-1*I)=MAX*DELTA(P-1*I)
970 CONTINUE

C *****
IF (SWITCH .NE. -1) GOTO 30110
WRITE(OUTFIL=20041) (DELTA(P-1*I)+I=1+NOVARS),MAX
20041 FORMAT(/'**DELTA.3*MAX**.10F/1X/1X*6F20.10I))
30110 CONTINUE
C *****
975 DO 985 I=1+NOVARS
IF (DELTA(P-1*I)+GT. MAXCHG(I)) DELTA(P-1*I)=MAXCHG(I)
DELTA(P-1*I)=DELTA(P-1*I)*SIGN(I)
985 CONTINUE
C *****
IF (SWITCH .NE. -1) GOTO 30120
WRITE(OUTFIL=20043) (DELTA(P-1*I)+I=1+NOVARS)
20043 FORMAT(/'**DELTA.4**.10F/1X*6F20.10I))
30120 CONTINUE
C *****
C CHECK TUPLE VALUES AGAINST THE BOUNDS
C IF ANY
DO 1100 I=1+NOVARS
TUPLES(I)=TUPLES(P-1,I)+DELTA(P-1,I)
IF (ABS(HYBDS(I)-NOLIM) LT EPSLON) GOTO 1000
IF (TUPLES(I) > HYBDS(I)) TUPLES(I)=HYBDS(I)
1000 CONTINUE
IF (ABS(LWBSDS(I)-NOLIM) LT EPSLON) GOTO 1100
IF (TUPLES(I) > LOWBSDS(I)) TUPLES(I)=LOWBSDS(I)
1100 CONTINUE

C ****
WRITE(OUTFIL,20030) (TUPLES(I),I=1,NOVARS)
20030 FORMAT(11H00,1UPLES4#1011/91X9GF2U*103)

C ****
DO 3600 I=1,NOVARS
INPUT(I)=TUPLES(I)
3600 CONTINUE

C CHECK FOR A SMALL GRADIENT
SUM=0.
DO 1300 I=1,NOVARS
SUM=SUM+GRAD(I)**2
1300 CONTINUE
Q=SUM**0.5
IF (SUM LT FLAT) Q=0.1
IF (SUM LT FLAT) GOTO 3100

C TO EXIT IF NO SIGNIFICANT CHANGE FROM PREVIOUS VALUES
DO 3600 I=1,NOVARS
IF (ABS(TUPLES(I)-TUPLES(P-1,I)) GE EPSLON) GOTO 3700
3600 CONTINUE
Q=0.1
GOTO 3100
3700 CONTINUE
C END OF POINT COMPUTATION LOOP
3000 CONTINUE

P=0.1
DO 1400 I=1,NOVARS
INPUT(I)=TUPLES(I)
1400 CONTINUE

DO 1500 I=1,NOVARS
CALL PRL(NOVARS,I,INPUT,EPSON,PARTYL,INCRTUPLES(P,NOVAL1))
1500 CONTINUE

C ****
WRITE(OUTFIL,20030) (TUPLES(P,NOVAL),P=1,NOVAL)
20030 FORMAT(11H00,1UPLES4#1011/91X9GF2U*103)
300150 CONTINUE
C ******

GRAD(P,J)=PARTYL

1500 CONTINUE
IF (I+1) I=1
C ******

IF (SWITCH *ME* = 1) GOTO 30130
WRITE(OUTFIL=20050) SUM

30130 CONTINUE
C ******

C BEGIN OUTPUT OF COMPUTATIONS
C WRITE FILE FOR VARIABLE VARIATION ROUTINE (VARYVAR)
WRITE(FILE1=10020) P,NOVARS, EPSLON

10020 FORMAT(I5+5*F10.7)
WRITE(FILE1=10030) FORMT2

10030 FORMAT(A80)
DO 3200 I=1,P
WRITE(FILE1=FORMT2) (TUPLES(I,J)+1.NOVAR)+,TUPLES(I,NOVAL)

3200 CONTINUE
C OUTPUT TO PRINTER
WRITE(OUTFIL9=10040)

10040 FORMAT(/,12X,*TUPLES*+/)
DO 3300 I=1,P
WRITE(OUTFIL9=FORMAT3) (TUPLES(I,J)+1.NOVAR)+,TUPLES(I,NOVAL)

3300 CONTINUE
WRITE(OUTFIL1=10050)

10050 FORMAT(/,21X,*GRADIENTS*+/)
DO 3400 I=1,P
WRITE(OUTFIL=FORMAT3) (GRAD(I,J)+1.NOVAR)+

3400 CONTINUE
WRITE(OUTFIL9=10060)

10060 FORMAT(/,21X,*INCREMENTs*+/)
DO 3500 I=1,P
WRITE(OUTFIL9=FORMAT3) (DELTA(I,J)+1.NOVAR)+

3500 CONTINUE
WRITE(FILE2=10070) P,NOVARS, EPSLON

10070 FORMAT(I5+5*F10.7)
WRITE(FILE2=10090) FORMT2
DO 3800 I=1,P
WRITE(FILE2=FORMAT2) (GRAD(I,J)+1.NOVAR)+

3800 CONTINUE
WRITE(OUTFIL9=10090)

10090 FORMAT(/)
END
2-8. POINTCOMP ROUTINE FLOWCHART

Start

Read in parameters

Debug mode?

Yes

Write out parameters

No

Read in a format read in initial point, variable bounds, min and max step sizes & default gradient multipliers

Read in format for disk and format for printed output

Debug mode?

Yes

Write out initial point bounds, step sizes and default multipliers

No
1. **PREPR**
   Call program to be tested to evaluate it on the initial point

2. **PARTL**
   Call routine to compute partial

3. **Debug mode?**
   - Yes
     - Print out partial and point computed at
   - No
     - Compute the sign of the increments

4. **Debug mode?**
   - Yes
     - Print out gradient and sign of each component of increment
   - No
     - Compute candidate gradient multiplier based on max step size (MAXCHG)

5. **Debug mode?**
   - Yes
     - Print out the candidate multiplier
   - No
     - For variables with gradient component too small, take the default multiplier (NOMULT)
Compute candidate multiplier based on distance to bound or doubling.

If no bound, set appropriate component to the minimum of the corresponding default multiplier (NOMULT) and the quotient of the current component and the partial for that variable.

Debug mode?

Yes

Print out this candidate multiplier.

No

Compute component by component minimum of both candidates.

Debug mode?

Yes

Print out vector of minima.

No

Find minimum of components of minima vector.

Debug mode?

Yes

Print out minima vector and minimal component.

No
Multiply gradient by minimum to obtain candidate increment.

Use the minimal multiplier provided as an input parameter if minimum is too small (MINML).

Compute an increase factor if components are smaller than MIN(I). Find the maximal such factor if any exist.

Print out candidate increment.

Debug mode...

No

Print all values of minimum sign due to candidate increment component for components below the minimum.

Increase increment if necessary in order to exceed MIN(I) values.

Debug mode...

No

Print the increment and multiplier if increase occurred.

Multiply increment by the appropriate sign.

Debug mode...

No

Print the candidate increment.
Increment current values to obtain new candidate values

Set values exceeding bounds to the value of the bound

Debug mode?
Yes

Print new values

Evaluate at new point

Gradient small?
Yes

Delete last point

New values differ significantly from the previous values?
Yes

More points needed?
Yes

No
Compute the gradient of the last point

Debug mode?

Yes

Write the point coordinates and each partial

No

Debug mode?

Yes

Print out magnitude of gradient

No

Write file of points on unit 10 and gradient file on unit 11

Print out points, gradients, and increments

End
Section II. THE VARVARY1 ROUTINE

2-9. INTRODUCTION. The purpose of this routine is to take each of the points generated by the POINTCOMP routine or chosen by the user and to vary each variable in turn (using the bound and increment) while keeping the other variables fixed. The program to be tested is evaluated at each of these new points, and the new points and their associated program values and gradients are printed out. Difference quotients are computed and printed for each variable as it is varied. The points at which the difference quotient is computed are those defined by the user provided increments and lower bounds. The new points can be used to study the effects of varying just one variable at a time and to provide points for graphs.

2-10. LIMITATIONS

a. When varying a single variable, the only values tested are those of the form: lower bound + integer x increment ~ higher bound. The current routine will not accept a list of values of the form 1, 3, 5, 17, 98.

b. All variables are real.

c. Currently restricted to 100 variables.

d. Currently restricted to a maximum of 500 steps per variable being varied.

e. When this routine is used on output from POINTCOMP or GRID, the limitations of these routines apply as well.

2-11. RUN SETUP

a. Developing an Absolute File in ASCI

@MAP,S , name of absolute element

IN 03PROGTEST.VARVARY1

IN 03PROGTEST.PARTIAL

IN element containing PREPR

IN program to be tested

LIB$*FTN.B.

END

2-24
b. To Execute

@USE 10, name of file containing input points

@USE 11, name of file into which gradients are placed in standard format

@USE 12, name of file into which new points are placed in standard format

@USE 13, name of scratch file

@USE 14, name of scratch file

@ASG,A name of file containing input points

@ASG,A name of file into which gradients will be placed

@ASG,A name of file into which new points are placed

@ASG,A name of scratch file

@ASG,A name of scratch file

@XQT absolute element

[Input deck]

c. Description of Input Deck

(1) Line 1. (Format to read in each of the following three lines, one at a time.)

(2) Line 2. Lower bounds for the variables; the leftmost bound pertains to the first variable. The line must adhere to the above format.

(3) Line 3. Upper bounds for the variables.

(4) Line 4. Increments for the variables.

(5) Line 5. (The format to print out the point identification number, the new variable values and the output value from the routine being tested, one point at a time.) This format is also used to print out the gradient identification number and the gradient coordinates.
d. Sample Input Data for VARVARY1

(13F5.2) Format for reading in the next three lines, one at a time.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>Lower bounds.</td>
</tr>
<tr>
<td>18.0</td>
<td>9.0</td>
<td>48.0</td>
<td>1.0</td>
<td>Upper bounds.</td>
</tr>
<tr>
<td>3.5</td>
<td>1.5</td>
<td>10.0</td>
<td>0.1</td>
<td>Increments</td>
</tr>
</tbody>
</table>

The leftmost column in rows 1-4 pertains to the first variable, the middle column to the second variable, etc.

e. Sample Input File for VARVARY1--Unit 10

(' POINT NUMBER ',15,2X,8F10.5) Format for printing out new points and gradients, one to a line.

(1X,13F10.5) Format to print out one line of difference quotients:

```
INPUT VALUES

9 4 .0010000
(6(7X,F13.5))
5.00000 .50000 .50000 .50000 .73821
5.00000 .61902 .75809 .63333 .99916
5.00119 .73820 .97516 .76667 1.24159
5.00270 .80924 1.08781 .84667 1.37902
5.00270 .84061 1.13401 .88222 1.43833
5.00270 .85178 1.14995 .89492 1.45926
5.00270 .85513 1.15468 .89873 1.46551
5.00270 .85513 1.15590 .89873 1.46626
5.00270 .85513 1.15715 .89975 1.46753
```

This file was created by POINTCOMP on unit 10.

(1) Line 1. The first number is the number of points in the 15 format and the second is the number of variables in an 13 format. The third number is used to determine when a number is essentially zero, in F10.7 format.
(2) Line 2. The format with which the following lines were written, and with which they may be read.

(3) Lines 3-7. The leftmost four columns represent the values of variables one to four, reading from left to right. Each entry in the rightmost column gives the tested program value when run with the variable values printed on the same row. Each row represents an input point. The leftmost four columns are values of the input variable. The last column is the output value of the routine being evaluated.

f. Sample VARVARY1 Run Setup

@USE 10,03MAT1.
@ASG,A 03MAT1.
@USE 11,03MAT3.
@ASG,A 03MAT3.
@USE 12,03MAT4.
@ASG,A 03MAT4.
@USE 13,03MAT5.
@ASG,A 03MAT5.
@USE 14,03MAT6.
@ASG,A 03MAT6.
@XQT 03PROGTEST.VARVARY1

Input cards:
   (13F5.2)
   5.0  0.5  0.5  0.5
   18.0  9.0  48.0  1.0
   3.5  1.5  10.0  0.1
   (' POINT NUMBER ',15,2X,8F10.5)
   (1X,13F10.5)
CAA-D-80-1

2-12. OUTPUT DESCRIPTION AND SAMPLE OUTPUT

a. **Printer**

(1) **Sample VARVARY1 Output--Points**

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>VARYING VARIABLE NUMBER 1</th>
<th>VARYING VARIABLE NUMBER 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.00000 .50000 .50000 .50000 .73821</td>
<td>5.00000 .50000 .50000 .50000 .73821</td>
</tr>
<tr>
<td>2</td>
<td>8.50000 .50000 .50000 .50000 .73934</td>
<td>5.00000 .50000 .50000 .50000 .73934</td>
</tr>
<tr>
<td>3</td>
<td>12.00000 .50000 .50000 .50000 .73985</td>
<td>5.00000 .50000 .50000 .50000 .73985</td>
</tr>
<tr>
<td>4</td>
<td>15.50000 .50000 .50000 .50000 .74014</td>
<td>5.00000 .50000 .50000 .50000 .74014</td>
</tr>
<tr>
<td>5</td>
<td>19.00000 .50000 .50000 .50000 .74033</td>
<td>5.00000 .50000 .50000 .50000 .74033</td>
</tr>
</tbody>
</table>

**DIFFERENCE QUOTIENT MATRIX**

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>POINT NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>5.00000</td>
</tr>
<tr>
<td>7</td>
<td>5.00000</td>
</tr>
<tr>
<td>8</td>
<td>5.00000</td>
</tr>
<tr>
<td>9</td>
<td>5.00000</td>
</tr>
<tr>
<td>10</td>
<td>5.00000</td>
</tr>
<tr>
<td>11</td>
<td>5.00000</td>
</tr>
<tr>
<td>12</td>
<td>5.00000</td>
</tr>
</tbody>
</table>

**DIFFERENCE QUOTIENT MATRIX**

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.00032</td>
<td></td>
<td>.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.00023</td>
<td>.00015</td>
<td>.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.00018</td>
<td>.00011</td>
<td>.00008</td>
<td>.0000</td>
<td></td>
</tr>
<tr>
<td>.00015</td>
<td>.00009</td>
<td>.00007</td>
<td>.00005</td>
<td>.0000</td>
</tr>
</tbody>
</table>

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>.0000</td>
<td>.0032</td>
<td>.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.0023</td>
<td>.0015</td>
<td>.0000</td>
<td></td>
<td>.0000</td>
</tr>
<tr>
<td>.0018</td>
<td>.0011</td>
<td>.0008</td>
<td>.0000</td>
<td>.0000</td>
</tr>
<tr>
<td>.0015</td>
<td>.0009</td>
<td>.0007</td>
<td>.0005</td>
<td>.0000</td>
</tr>
</tbody>
</table>
(2) **Sample VARVARY1 Output--Gradients**

**THE VARIABLE VARIED IS NUMBER 1**

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>1</th>
<th>0.00048</th>
<th>0.30919</th>
<th>0.67048</th>
<th>0.34638</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>2</td>
<td>0.00019</td>
<td>0.32471</td>
<td>0.68200</td>
<td>0.34773</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>3</td>
<td>0.00010</td>
<td>0.33081</td>
<td>0.68706</td>
<td>0.34836</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>4</td>
<td>0.00006</td>
<td>0.33495</td>
<td>0.68991</td>
<td>0.34871</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>5</td>
<td>0.00004</td>
<td>0.33762</td>
<td>0.69173</td>
<td>0.34894</td>
</tr>
</tbody>
</table>

**THE VARIABLE VARIED IS NUMBER 2**

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>6</th>
<th>0.00048</th>
<th>0.30919</th>
<th>0.67048</th>
<th>0.34638</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>7</td>
<td>0.02158</td>
<td>0.18565</td>
<td>0.60644</td>
<td>1.08123</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>8</td>
<td>0.05104</td>
<td>0.12313</td>
<td>0.56450</td>
<td>1.55265</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>9</td>
<td>0.07778</td>
<td>0.08812</td>
<td>0.53491</td>
<td>1.88027</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>10</td>
<td>0.10199</td>
<td>0.06543</td>
<td>0.51291</td>
<td>2.12195</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>11</td>
<td>0.12182</td>
<td>0.05001</td>
<td>0.49591</td>
<td>2.30725</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>12</td>
<td>0.13869</td>
<td>0.04014</td>
<td>0.48239</td>
<td>2.45384</td>
</tr>
</tbody>
</table>

(3) **Description of Points Output.** Each row describes a point. The leftmost four columns are values of variables one through four, reading from left to right. The rightmost entry on each row is the output value when the inputs are those in columns one through four.

(4) **Description of Gradients Output.** Each row is the gradient of the routine being tested at the point whose number is listed on the left. Again, the values of variables one to four are listed from left to right. The point numbers link the printout of the points and variable values to the gradient printout.

(5) **Sample VARVARY1 Output--Difference Quotients**

**GIVEN SAMPLE OUTPUT**

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>51</th>
<th>5.00000</th>
<th>0.73820</th>
<th>0.97516</th>
<th>0.76667</th>
<th>1.24158</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>52</td>
<td>8.50000</td>
<td>0.73820</td>
<td>0.97516</td>
<td>0.76667</td>
<td>1.26175</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>53</td>
<td>12.00000</td>
<td>0.73820</td>
<td>0.97516</td>
<td>0.76667</td>
<td>1.27087</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>54</td>
<td>15.50000</td>
<td>0.73820</td>
<td>0.97516</td>
<td>0.76667</td>
<td>1.27607</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>55</td>
<td>19.00000</td>
<td>0.73820</td>
<td>0.97516</td>
<td>0.76667</td>
<td>1.27942</td>
</tr>
</tbody>
</table>
We define the following labels for some of the variable values and outputs:

<table>
<thead>
<tr>
<th>Values</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_1$ = 5.00000</td>
<td>$Y_1$ = 1.24158</td>
</tr>
<tr>
<td>$X_2$ = 8.50000</td>
<td>$Y_2$ = 1.26175</td>
</tr>
<tr>
<td>$X_3$ = 12.00000</td>
<td>$Y_3$ = 1.27087</td>
</tr>
<tr>
<td>$X_4$ = 15.50000</td>
<td>$Y_4$ = 1.27607</td>
</tr>
<tr>
<td>$X_5$ = 19.00000</td>
<td>$Y_5$ = 1.27942</td>
</tr>
</tbody>
</table>

The associated difference quotients are

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>.00000</td>
<td>.00000</td>
<td>.00000</td>
<td>.00000</td>
<td>.00000</td>
</tr>
<tr>
<td>.00576</td>
<td>.00261</td>
<td>.00144</td>
<td>.00096</td>
<td>.00000</td>
</tr>
<tr>
<td>.00418</td>
<td>.00204</td>
<td>.00122</td>
<td>.00096</td>
<td>.00000</td>
</tr>
<tr>
<td>.00328</td>
<td>.00168</td>
<td>.00122</td>
<td>.00096</td>
<td>.00000</td>
</tr>
<tr>
<td>.00270</td>
<td>.00168</td>
<td>.00122</td>
<td>.00096</td>
<td>.00000</td>
</tr>
</tbody>
</table>

If we denote the varying coordinate of point numbers 51-55 by $x_1$ to $x_5$, and denote the corresponding values (the rightmost column) by $Y_1$ to $Y_5$, the second row in the matrix has the nonzero entry:

$$\frac{y_2 - y_1}{x_2 - x_1} = .00576.$$ 

Note that the difference in $y$ is in the numerator and the difference in $x$ is in the denominator. The third row has the nonzero entries:

$$\frac{y_3 - y_1}{x_3 - x_1} = .00418$$

$$\frac{y_3 - y_2}{x_3 - x_2} = .00261$$

Note that the third row has the first term of numerators and denominators indexed by 3 and the second terms are indexed by 1 and 2 (i.e., 3-1). The fourth row has the nonzero entries:
\[
\frac{y_4 - y_1}{x_4 - x_1} = 0.00328
\]

\[
\frac{y_4 - y_2}{x_4 - x_2} = 0.00204
\]

\[
\frac{y_4 - y_3}{x_4 - x_3} = 0.00148
\]

Note that the fourth row has the first term in each numerator and denominator indexed by 4 and the second terms are indexed by 1, 2, and 3 (i.e., 4-1). In general, the nth row is comprised of the ordered set:

\[
\left(\frac{y_n - y_j}{x_n - x_j} \bigg| j=1,...,n-1\right)
\]

To show the geometrical significance of these computations given the following points:

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>X</th>
<th>Y1</th>
<th>Y2</th>
<th>Y3</th>
<th>Y4</th>
</tr>
</thead>
<tbody>
<tr>
<td>51</td>
<td>5.00000</td>
<td>0.73820</td>
<td>0.97516</td>
<td>0.76667</td>
<td>1.24158</td>
</tr>
<tr>
<td>52</td>
<td>8.50000</td>
<td>0.73820</td>
<td>0.97516</td>
<td>0.76667</td>
<td>1.26175</td>
</tr>
<tr>
<td>53</td>
<td>12.00000</td>
<td>0.73820</td>
<td>0.97516</td>
<td>0.76667</td>
<td>1.27087</td>
</tr>
<tr>
<td>54</td>
<td>12.50000</td>
<td>0.73820</td>
<td>0.97516</td>
<td>0.76667</td>
<td>1.27607</td>
</tr>
<tr>
<td>55</td>
<td>19.00000</td>
<td>0.73820</td>
<td>0.97516</td>
<td>0.76667</td>
<td>1.27942</td>
</tr>
</tbody>
</table>
and the difference quotients:

\[
\begin{array}{ccc}
0.00576 & 0.0000 & 0.0000 \\
0.00418 & 0.00122 & 0.0000 \\
0.00328 & 0.00204 & 0.00143 \\
0.00270 & 0.00168 & 0.00096
\end{array}
\]

Table 2-1. Table of Difference Quotients

The purpose of these computations will now be explained. The import of the quotients in triangle A is illustrated in Figure 2-1 (drawn not to scale).
The figures in triangle B show that:

Figure 2-2. Triangle B in Table 2-1

The fourth line gives the following picture (not to scale):

Figure 2-3. Slopes on the Fourth Line of Table 2-1
Note that since this is the fourth row, all lines have the fourth point as terminus, reading from left to right. The second column gives the following picture (not to scale):

\[ \text{slope .00261} \]
\[ \text{slope .00204} \]
\[ \text{slope .00168} \]

![Figure 2-4. Slopes on the Second Column of Table 2-1](image)

Note that since this is the second column, all lines start at point number 52, reading from left to right.
b. Sample VARVARY1 Output Files and Descriptions

(1) Standard Format File on Unit 12--Points

225 4 .0010000
(6(7X,F13.5))

5.00000 .50000 .50000 .50000 .73821
8.50000 .50000 .50000 .50000 .73934
12.00000 .50000 .50000 .50000 .74014
15.50000 .50000 .50000 .50000 .74033
19.00000 .50000 .50000 .50000 .74033

The first row indicates that there are 225 points in the file, that there are four variables (the rightmost column gives the output values), and that numbers smaller than .001 are considered to be zero. The second row gives the format in which the file was written, which may be used for reading in the file. The 225 points and values commence at line 3 and comprise the remainder of the file. (Only five points are illustrated.)

(2) Standard Format File on Unit 11--Gradients

225 4 .0010000
(6(7X,F13.5))

.00048 .30919 .67048 .34638
.00019 .32471 .68200 .34773
.00010 .33081 .68706 .34836
.00006 .33495 .68991 .34871
.00004 .33762 .69173 .34894

The first two rows of this file are described as above. The gradients comprise the following rows.

(3) Scratch Files

(a) Unit 13

-19.00100 4.00100 .00048 .30919 .67048 .34638
20.00100 4.00100 .00055 .37265 .66454 .34684
21.00100 4.00100 .00061 .43536 .65859 .34731
22.00100 4.00100 .00067 .49823 .65264 .34778
23.00100 4.00100 .00072 .56128 .64668 .34825
24.00100 4.00100 .00076 .62451 .64071 .34872
25.00100 4.00100 .00079 .68990 .63473 .34920
-26.00100 1.00100 .00359 .35940 .65458 .40207
The first column is a point count. A negative sign indicates a
new variable is being varied. The second column indicates which
variable is being varied. Columns 3 through 6 are the gradient
values for variables 1 to 4, respectively.

<table>
<thead>
<tr>
<th>Values</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.73821</td>
</tr>
<tr>
<td>8.5000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.73934</td>
</tr>
<tr>
<td>12.0000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.73985</td>
</tr>
<tr>
<td>15.5000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.74014</td>
</tr>
<tr>
<td>19.0000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.74033</td>
</tr>
<tr>
<td>5.0000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.73821</td>
</tr>
<tr>
<td>2.0000</td>
<td>8.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>1.10039</td>
</tr>
</tbody>
</table>

The first four columns indicate variable values. The fifth column
gives the output values, each one corresponding to the input val-
ues listed on the same row.
2-13. VARVARY1 ROUTINE LISTING

PARAMETER FILE1=10, FILE4=13, NOVALS=100, INFILE=5, OUTFIL=6
PARAMETER POINTS=500, FILE2=11, FILE3=12, FILE5=14
C
DIMENSION TUPLE3(NOVALS), LOWBDS(NOVALS), HYRDS(NOVALS), INCR(NOVALS)
DIMENSION VALUEC(NOVALS), GRAD(NOVALS), VALU(POINTS), LINE(POINTS)
C
REAL TUPLES, LOWBDS, HYRDS, INCR, VALUES, VALUE, VALU, LINE, INCR1
REAL PARTY=, F=, EQEPSLON
C
INTEGER NOVARS, NOPTS=I+J+K+L+P+Q
C
CHARCTFR=90, FORM1, FORM2, FORM3, FORM4
C
REWIN FILEI
REWIN FILE2
REWIN FILE3
REWIN FILE4
REWIN FILE5
G=0
READ(FILE1, 10000) NOPTS, NOVARs, EPSLON
10000 FORMAT(5, 13, F10.7)
READ(FILE1, 10010) FORM1
10010 FORMAT(A80)
READ(FILE1, FORM2) TUPLE3(I)
READ(FILE1, FORM3) LOWBDS(I)
READ(FILE1, FORM4) HYRDS(I)
READ(FILE1, FORM5) INCR(I)
READ(FILE1, FORM6) VALUE(I)
WRITE(OUTFILE, 10050)
10050 FORMAT(10, 10)
DO 400 P=1, NOPTS
READ(FILE1, FORM1) TUPLE3(I)
DO 300 I=1, NOVARs
WRITE(OUTFILE, 10070) 1
300 CONTINUE

DO 100 J=1, NOVARs
VALUES(J)=TUPLE3(J)
100 CONTINUE

K=INT(HYRDS(I)-LOWBDS(I))/INCR(I)+1
DO 200 J=0, K
VALUES(I)=LOWBDS(I)+J*INCR(I)
CALL PRFPRIVR(VALUES, VALUE)
G=G+1
WRITE(OUTFILE, FORM3) G*VALUES(I), VALUES(I), VALUE
WRITE(FILES, FORM1) VALUES(I), VALUES(I), VALUE
VALUES(J+1)=VALUE
C A A - D - 8 0 - 1

F 0 = 0 + E P S L O N
F I = I + E P S L O N
D O 1 5 0 1 = 1 + N O V A R S
C A L L P A R I L ( N O V A R S , L + V A L U E S + E P S L O N , P A T Y L , I N C R 1 , V A L U E )
C R A D I L = P A T Y L
1 5 0 C O N T I N U E
I F ( J = E Q . 0 ) F 0 = - F 3
W R I T E ( F I L E 4 * F O R M T 1 ) F 0 + F I , ( C R A D I L ) + L = 1 + N O V A R S
2 0 0 C O N T I N U E
W R I T E ( O U T F I L * 1 0 0 5 0 )
1 0 0 5 0 F O R M A T ( / 1 4 4 0 I H ) + " D I F F E R E N C E O U T C I E N T M A T R I X * 2 0 ( 2 1 H ) + / / , D O 7 0 0 J = 0 + K
D O 6 0 0 L = 0 + J
L I N E ( L + 1 ) = ( V A L U E ( J + 1 ) - V A L U E ( L + 1 ) ) / ( M A X D I S T + 0 + L + 1 + I N C R 1 1 )
6 0 0 C O N T I N U E
W R I T E ( O U T F I L * F O R M T 4 1 ) ( L I N E ( L + 1 ) , L = 0 + J)
7 0 0 C O N T I N U E
W R I T E ( O U T F I L * 1 0 0 6 0 )
1 0 0 6 0 F O R M A T ( / )
3 0 0 C O N T I N U E
W R I T E ( O U T F I L * 1 0 0 3 0 ) P
1 0 0 3 0 F O R M A T ( / 3 4 0 I H ) + " E N D O F C O M P U T A T I O N S F O R P O I N T N U M B E R * * , 1 1 5 + 3 0 ( 1 4 1 H ) / / , /
4 0 0 C O N T I N U E
W R I T E ( F I L E 2 * 1 0 0 0 0 ) 3 + N O V A R S * E P S L O N
W R I T E ( F I L E 2 * 1 0 0 1 0 ) * F O R M T 1
W R I T E ( F I L E 3 * 1 0 0 0 0 ) 0 + N O V A R S * E P S L O N
W R I T E ( F I L E 3 * 1 0 0 1 0 ) * F O R M T 3
R E W I N D F I L E 4
R E W I N D F I L E 5
W R I T E ( O U T F I L * 1 0 0 9 0 )
1 6 0 9 0 F O R M A T ( / 1 4 4 0 I H ) + " T H E C O R R E S P O N D I N G G R A D I E N T S * / , D O 8 0 0 P = 0 + 0
R E A D ( F I L E 4 * F O R M T 1 ) F 0 + F I , ( C R A D I L ) + L = 1 + N O V A R S
P E A D ( F I L E 5 * F O R M T 1 ) ( V A L U E S ( L ) , L = 1 + N O V A R S ) + V A L U E
I F ( F G * G T . 0 . 0 ) G O T O 7 5 0
F 0 = - F 4
K = I N T ( F I )
W R I T E ( O U T F I L * 1 0 0 0 ) K
1 0 1 0 0 F O R M A T ( / 1 0 1 1 H ) + " T H E V A R I A B L E V A R I E D I S N U M B E R * , 1 5 + / , D O 8 0 0 P = 0 + 0
7 5 0 J = I N T ( F I )
W R I T E ( O U T F I L * F O R M T 3 1 ) J , ( C R A D I L ) + L = 1 + N O V A R S
W R I T E ( F I L E 3 * F O R M T 1 ) ( V A L U E S ( L ) , L = 1 + N O V A R S ) + V A L U E
W R I T E ( F I L E 2 * F O R M T 1 ) ( C R A D I L ) + L = 1 + N O V A R S
9 0 0 C O N T I N U E
W R I T E ( O U T F I L * 1 0 0 4 0 )
1 0 0 4 0 F O R M A T ( / )
E N D

2 - 3 8
2-14. VARVARY1 ROUTINE FLOWCHART

Start

Read in the files

Read in the no.
of points, the
no. of values,
and the zero
approximator
from the file

Read in the file
S/0 format

Read in the load
data format

Read in the bounds
and the
increment

Read in the location
print format

Read in the format for the
difference
coefficients

Read in the
coordinates of a point

Define the
variable to be varied
next

Print message
describing
variable

(1)

(1)

(1)
3) All prints processed?

Yes → Write out headers for standard files

No → Read in a point and its gradient

New variable being varied?

Yes → Print message describing variable

No → Print the point number and gradient

Write the point value into one standard file, the gradient into the other

Completed the last point?

Yes → End

No → Yes
CHAPTER 3
THE ANALYZ SUBSYSTEM

3-1. INTRODUCTION. This routine computes variable sensitivity statistics from gradients in a standard format file on unit 11. Statistics are initially computed on each gradient. The gradients are summed, component by component, and the same statistics are computed for the sum vector.

3-2. DISCUSSION OF STATISTICS

   a. The first statistic is the ratio of each component of the gradient to the component with the smallest absolute value.

   b. The second statistic is the change in the component required to achieve a unit change in the output for each component of the gradient. For the summed vector, a change in the output equal to the number of gradients rather than a unit change is utilized.

   c. The third statistic is the ratio of each component to the component with the largest absolute value; except that for the largest component, its ratio with the next largest component is taken.

3-3. LIMITATIONS

   a. The gradients input to ANALYZ must all be real.

   b. No more than 500 gradients can be analyzed in a single run by the currently compiled version of ANALYZ.

   c. No more than 20 variables may be analyzed by the current version of ANALYZ.

3-4. RUN SETUPS

   a. To Execute

      @USE 11, name of file containing gradients in standard format.

      @ASG,A filename.

      @XQ1 U3PRGTEST.ANALYZ

      [Input deck]
b. Description of Input Deck

(1) Line 1. Field: Al
Format: I3

Al is +1 if no debugging information is desired, -1 if debugging is desired.

(2) Line 2. (Format for printing the statistics.)

(3) Line 3. (Format for debug printouts.)

c. Sample Input Data

+1 Indicates no debugging output requested.

\[(6(7X,F13.5),/,,100(21X,5(F13.5,7X),/))\] Format for statistical output

\[(6(7X,F13.5),/,,100(21X,5(F13.5,7X),/))\] Debugging data output format

d. Input File Description on Unit 11

Produced by PDP-6 on Unit 11

9 4 .00010000
\[(6(7X,F13.5))\]

.00048 .30919 .67048 .34638
.00359 .35940 .65458 .40207
.00852 .40086 .63565 .45144
.01236 .42295 .62294 .47942
.01428 .43221 .61698 .49139
.01500 .43545 .61481 .49560
.01522 .43641 .61415 .49686
.01523 .43631 .61415 .49674
.01527 .43670 .61405 .49663

(1) Line 1

(a) Format: I5, I3, F10.7.
(b) Meaning: Nine lines, four variables, zero approximator .001.

(2) Line 2. Format for reading in the following lines of data, one at a time.

(3) Other Lines. Each row is a gradient, variable 1 values are in the leftmost column, variable 2 values are in the next column, etc.

e. Sample Run Setup

@USE 11,03MAT2.

@ASG,A 03MAT2.

@XT 03PROGTEST.ANALYZ +1

(6(7X,F13.5),/100(21X,5(F13.5,7X),/))

(6(7X,F13.5),/100(21X,5(F13.5,7X),/))

3-5. OUTPUT DESCRIPTION AND SAMPLE OUTPUT

a. Sample Output--Gradient by Gradient

STATISTICS ON EACH GRADIENT

******************EQUIVALENT CHANGES**************

<table>
<thead>
<tr>
<th></th>
<th>Variable 1</th>
<th>Variable 2</th>
<th>Variable 3</th>
<th>Variable 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0000</td>
<td>3.23426</td>
<td>1.48147</td>
<td>2.88700</td>
<td></td>
</tr>
<tr>
<td>279.55153</td>
<td>2.78242</td>
<td>1.52770</td>
<td>2.48713</td>
<td></td>
</tr>
<tr>
<td>117.37089</td>
<td>2.49464</td>
<td>1.57319</td>
<td>2.21513</td>
<td></td>
</tr>
<tr>
<td>80.90615</td>
<td>2.36435</td>
<td>1.60529</td>
<td>2.08585</td>
<td></td>
</tr>
<tr>
<td>70.02001</td>
<td>2.31369</td>
<td>1.62080</td>
<td>2.03504</td>
<td></td>
</tr>
<tr>
<td>66.66667</td>
<td>2.29647</td>
<td>1.62652</td>
<td>2.01776</td>
<td></td>
</tr>
<tr>
<td>65.70302</td>
<td>2.29142</td>
<td>1.62827</td>
<td>2.01264</td>
<td></td>
</tr>
<tr>
<td>65.65988</td>
<td>2.29195</td>
<td>1.62827</td>
<td>2.01313</td>
<td></td>
</tr>
<tr>
<td>65.48788</td>
<td>2.28990</td>
<td>1.62853</td>
<td>2.01357</td>
<td></td>
</tr>
</tbody>
</table>
b. Description. The preceding statistics are described in paragraph 3-2. For each statistic, each row corresponds to a gradient.

c. Sample Output—Sum of Gradients

SUMMED GRADIENT COMPUTATIONS

***SUMMED GRADIENTS***

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>.09947</td>
<td>3.66948</td>
<td>5.65779</td>
<td>4.15653</td>
</tr>
</tbody>
</table>

***EQUIVALENT SUMMED CHANGES***

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>90.47954</td>
<td>2.45266</td>
<td>1.59073</td>
<td>2.16427</td>
</tr>
</tbody>
</table>
d. Description. The first row is the sum of the gradients. The last three rows are the same statistics as before, in the same order, but applied to the sum vector only.
3-6. ANALYZ ROUT. LISTING

PARAMETER FILE2=INFILE, OUTFILE=E
PARAMETER NOVALS=20, NOGRC=500

DIMENSION GRAD(NOVALS), RMAX(NOVALS), CMAX(NOVALS)

DIMENSION RMAX(NOGRC+2), CMAX(NOGRC+2), PRINT(NOVALS)

C ANALYZE GRADIENT VALUES
C READ THE INPUTS
C READ(INFILE, 100) SWITCH
C READ(INFILE, 200) FMTRC
C READ(INFILE, 100) NOVAR
C CONTINUE
C REWIND FILE2
C READFILE2(300) NOGRD, NOVARS, EPSLON
C READFILE2(200) FMTRC
C READFILE2(FMTRC) (GRAD[6*V]*V=1, NOVAR)
C CONTINUE
C IF (SWITCH =FL.A. -1) READ(INFILE, 200) FMT90
C IF (SWITCH =NE. -1) GOTO 20200
C WRITE(OUTFILE, 2000) NOGRD, NOVARS
C 20000 FORMAT(/ **INPUT** 
C DO 2100 C=1, NOGRC
C WRITE(OUTFILE, FMT95) (GRAD[6*V]*V=1, NOVAR)
C 2100 CONTINUE
C DO 2200 C=1, NOGRC
C WRT
C CONTINUE
C *** COMPUTE ROW MINIMA
C DO 750 G=1, NOGRC
C DO 500 V=1, NOVAR
C RMINS(G)=0.0
C IF (ABS(GRAD(G)*V) .LT. EPSLON) GOTO 500
C RMINS(G)=ABS(GRAD(G)*V)

3-6
50. CONTINUE
51. 600 DO 700 V=1+NOVARS
52. IF (ABS(1GRAD(G,V)) .LT. EPSLON) CPAD(G,V)=U*0
53. IF (ABS(1GRAD(G,V)) .LT. EPSLON) GOTO 700
54. IF (ABS(1GRAD(G,V)) .LT. RMSN(G) RMSN(C)=ABS(1GRAD(G,V))
55. 700 CONTINUE
56. 750 CONTINUE
57. C ************
58. IF (SWITCH NE. -1) GOTO 204000
59. WRITE(OUTFIL,20300)
60. WRITE(OUTFIL,FMT=203200) (RMSN(G),G=1,NQRD)
61. 20300 CONTINUE
62. C ************
63. C COMPUTE THE COLUMN S U M S
64. DO 900 V=1+NOVARS
65. CSUMS(V)=G.
66. DO 800 G=1+NQRD
67. CSUMS(V)=CSUMS(V)+GRAD(G,V)
68. 800 CONTINUE
69. 900 CONTINUE
70. CMIN=ABS(CSUMS(V))
71. DO 1000 V=2+NOVARS
72. IF (ABS(CSUMS(V)) .LT. RMSN(V)) GOTO 1000
73. 1000 CONTINUE
74. C
75. C TO COMPUTE THE TWO DISTINCT LARGEST PARTIALS (IN ABSOLUTE VALUE)
76. C IN EACH GRADIENT
77. IF (NOVARS .LT. 2) GOTO 1600
78. 1150 DO 1150 G=1+NQRD
79. RMAX(G+1)=ABS(1GRAD(G+1))
80. 1150 CONTINUE
81. IF (ABS(1GRAD(G+1)) .LT. RMAX(G+1)) RMAX(G+1)=ABS(1GRAD(G+1))
82. 1100 CONTINUE
83. DO 1100 V=1+NOVARS
84. IF (ABS(1GRAD(G,V)) .LT. EPSLON) GOTO 1150
85. IF (ABS(1GRAD(G,V)) .LT. RMAX(G+1)) GOTO 1150
86. RMAX(G+2)=ABS(1GRAD(G+2))
87. 1150 CONTINUE
88. RMAX(G+2)=RMAX(G+1)
89. COTO 1350
LAA-D-80-1

1200 DO 1100 V=1+NOVARs
99. IF (ABS(CMAXS(V)) .GT. RMAXS(V)) AND. (ABS(CGRAD(V)) .LT.
100. 1 RMAXS(V)) RMAXS(V)=ABS(CGRAD(V))
101. CONTINUE
102. 1350 CONTINUE
103. C **************
104. IF (SWITCH .NE. -1) GOTO 21000
105. WRITE(OUTFILE,20700)
106. 20700 FORMAT(/** ROW MAXIMA///**)
107. DO 2U90U G=1+NOVARs
108. WRITE(OUTFILE,FMTD1G) (GRAD(V),V=1+NOVARs),RMAXS(V),RMAXS(V)
109. CONTINUE
110. 21000 CONTINUE
111. C **************
112. C TO COMPUTE THE TWO LARGEST COMPONENTS OF CSUMS IN ABSOLUTE VALUE
113. C
114. C
115. CSMAXS(1)=ABS(CSUMS(V))
116. DO 1400 V=2+NOVARs
117. IF (ABS(CSUMS(V)) .GT. CSMAXS(1)) CSMAXS(1)=ABS(CSUMS(V))
118. 1400 CONTINUE
119. CONTINUE
120. IF (ABS(CSUMS(V)) .LT. CSMAXS(1)) GOTO 1425
121. GOTO 1425
122. 1425 CONTINUE
123. C TO COMPUTE THE TWO LARGEST COMPONENTS OF CSUMS IN RELATIVE VALUE
124. C
125. CMAXS(1)=ABS(CSUMS(V))
126. DO 1500 V=1+NOVARs
127. IF (ABS(CSUMS(V)) .LT. CMAXS(1)) AND. (ABS(CSUMS(V)) .LT.
128. CMAXS(V)) CMAXS(1)=ABS(CSUMS(V))
129. 1500 CONTINUE
130. C
131. 1550 CONTINUE
132. C **************
133. IF (SWITCH .NE. -1) GOTO 21300
134. WRITE(OUTFILE,21100)
135. 21100 FORMAT(/** CSUMS HAYIMA///**)
136. WRITE(OUTFILE,FMTD1G) (CSUMS(V),V=1+NOVARs),CSMAXS(V),CSMAXS(V)
137. 21300 CONTINUE
138. C **************
139. 1600 CONTINUE
140. C
141. C TO COMPUTE THE RELATIVE RATIOS:
142. C
143. WRITE(OUTFILE,10000)
144. 10000 FORMAT(/** IX3L(14)**, "RELATIVE RATIOS",1D1H4.1/)
145. DO 1800 G=1+NOVARs

3-8
DO 1700 V=1,NOVAR$S
146. PRINT(V)=.U
147. IF (RMINS(G) .GE. EPSLON ) PRINT(V) = GRAD(G;V)/RMINS(G)
148. 1700 CONTINUE
149. WRITE(OUTFIL,FMTWRT) (PRINT(V);V=1,NOVAR$S)
150. 1600 CONTINUE
151. C PRINT THE EQUIVALENT CHANCE-
152. C
153. IF (GRAD(G;V) .GE. EPSLON ) PRINT(V) = GRAD(G;V)
154. C
155. 1660 CONTINUE
156. IF (NGVAR$S .LT. 21) 2000 2200
157. WRITE(OUTFIL,FMTWRT) (PRINT(V),V=1,NOVAR$S)
158.
159. C PRINT THE COMPARISONS
160. C
161. IF (RMAXS(G;1) .GE. EPSLON ) PRINT(V) = GRAD(G;V)/RMAXS(G;1)
162. C
163. 2100 CONTINUE
164. WRITE(OUTFIL,FMTWRT) (PRINT(V),V=1,NOVAR$S)
165. 2200 CONTINUE
166. C SUMMED GRADIENT COMPUTATION:
167. C
168. WRITE(OUTFIL,FMTWRT) (SUMS(V),V=1,NOVAR$S)
169. C
170. C WRITE OUT THE SUMMED GRADIENTS
171. C
172. WRITE(OUTFIL,FMTWRT) (SUMS(V),V=1,NOVAR$S)
173. C COMPUTE THE SUMMED RATIOS
174. C
193. WRITE(OUTFIL,10400)
194. 10400 FORMAT(//,'RELATIVE SUMMED PATIOS',3(JH,1,1))
195. DO 10500 V=1,NOVARS
196. PRINT(V)=0.0
197. IF (CSMIN + EPSLON 1 PRINT(V)=CSUMS(V)/CSMIN
198. 10500 CONTINUE
199. WRITE(OUTFIL,FMTWRT) (PRINT(V),V=1,NOVARS)
200. C COMPUTE THE SUMMED EQUIVALENT CHANCES
201. C
202. C
203. WRITE(OUTFIL,10600)
204. 10600 FORMAT(//,'SUMMED COMPARISONS',30(JH,1,1))
205. DO 10700 V=1,NOVARS
206. PRINT(V)=0.0
207. IF (CSUMS(V) .GE. EPSLON) PRINT(V)=NOVARS/CSUMS(V)
208. 10700 CONTINUE
209. WRITE(OUTFIL,FMTWRT) (PRINT(V),V=1,NOVARS)
210. C COMPUTE SUMMED COMPARISONS
211. C
212. IF (NOVARS .LT. 2) GOTO 11000
213. WRITE(OUTFIL,11000)
214. 11000 FORMAT(//,'SUMMED COMPARISONS',30(JH,1,1))
215. DO 11100 V=1,NOVARS
216. PRINT(V)=0.0
217. IF (CMAXS(V) .GE. EPSLON) PRINT(V)=CMAXS(V)/CMAXS(1)
218. IF (ABS(PRINT(V)) .EQ. 1.0) PRINT(V)=CMAXS(V)/CMAXS(2)
219. 11100 CONTINUE
220. WRITE(OUTFIL,FMTWRT) (PRINT(V),V=1,NOVARS)
221. 11100 FORMAT(/)
222. 1100 CONTINUE
223. WRITE(OUTFIL,11100)
224. 11100 FORMAT(/)
225. END

3-10
3-7. ANALYZ ROUTINE FLOWCHART

- Start
- Read debug/nu debug switch
- Read printout format
- Read in the number of gradients, no. of variables and smallness indicator
- Read in the format for file i/o
- Read in the gradients
- In debug mode?
  - Yes
  - Read in the debug mode format
  - In debug mode?
    - Yes
    - Write out the numbers of gradients and variables, and the smallness indicator
  - No

(1) (2)
Find the first non-vial.
Set small.
Find the material.
Set small.
Plot out.
Plot out.
Plot out.
Set the next largest component equal to the largest component to the gradient.

Yes: Use it as a starting value.

No: Find the next largest component in the gradient.

Done all gradients?

Yes: Next.

No: Done.

In debug mode?

Yes: Print the gradients and largest and next largest components.

No: Find the largest component of the sum vector.

Done the vector have a smaller non-trivial component?

Yes: Use it as a starting value.

No: Find the next largest component of the gradients.

(1)
To examine the next gradient

Only one component?

To examine first component

Is the minimum component trivial?

Compute the ratio of the component to the maximum component?

Compute the ratio of the component to the next largest component

To examining the next component

To examining the next gradient

Last gradient?
Compute and print the same statistics for the sum vector

Stop
CHAPTER 4
THE GRID SUBSYSTEM

Section I. THE GRID ROUTINE

4-1. INTRODUCTION. This routine utilizes a user specified subdivision of each variable in order to generate a grid of input variable values. The routine to be tested is evaluated at each node of the grid, and the gradients are also computed at each node. The output of the grid routine is used by REARRANGE and DIFFQUOT.

4-2. LIMITATIONS

a. As currently compiled, the routine is restricted to 20 input variables at the most.

b. All input variables (being varied) must be real.

c. Only one output from the routine being tested may be checked out at one time.

d. Testing time-consuming Monte Carlo routines may be too expensive. Rather than making several runs per point and averaging the output values, it is better to run one component of the system at a time, testing random variables no differently from deterministic variables.

e. The function represented by the routine being tested must be well-behaved.

4-3. RUN SETUPS

a. To Develop an Absolute ASCII Program File

@MAP,S name to be given to absolute element

IN 03PROGTEST.GRID.

IN 03PROGTEST.PARTIAL.

IN element containing the driver PREPR.

IN programs to be tested.

LIBS*FTN8.

END.
b. To Execute

@USE 10, name of file into which points will be stored.
@ASG,A name of this file.

@USE 11, name of file into which gradients will be stored.
@ASG,A name of this file.

@USE 12, name of scratch file (used as input to REARRANGE and DIFFQUOT).
@ASG,A name of this file.

@USE 13, name of scratch file.
@ASG,A name of this file.

@XQT name of absolute deck created by @MAP.

[Input deck]

c. Description of the Input Deck

(1) Line 1. Number of variables - I3

Zero approximator - F10.7

Debug mode field - I3

The zero approximator is a threshold: numbers smaller in absolute value are considered to be zero. If the debug mode field contains the number -1, the run will be in debug mode. Any other value implies run will not print debugging information.

(2) Line 2. (Format for reading in one line of variable variation data.)

(3) Line 3. Initial variable values. Variable one's initial value is leftmost, variable two's initial value is to the right of variable one, etc.

(4) Line 4. Bounds for variable values.

(5) Line 5. Increment (step) value for each variable.
(6) Line 6. Format for one line of output. First field should be 15, other fields should be real. The number of fields specified must be no less than number of variables + 1.

(7) Line 7. File I/O format for one line of output. All fields should be real and at least (number of variables + 2) fields must be specified.

(8) Line 8. Optional; should contain a format for writing one line of debugging data if in debug mode.

d. Sample Input Data

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>.001</td>
<td>+1</td>
<td></td>
</tr>
</tbody>
</table>

Four variables. Zero approximator value of .001. Not in debug mode

(13F6.0) Format for reading variable variation data

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Initial values

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>18.00</td>
<td>9.00</td>
<td>48.00</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Terminating values for incrementation

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5</td>
<td>1.5</td>
<td>10.0</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Step values for each variable

( ' POINT NUMBER ',16,5(7X,F13.5) ) Format for output

(6(7X,F13.5) ) File I/O format

Not a debugging run, so debugging format is omitted.

e. Sample Run Setup

@USE 10,03MAT5.
@ASG,A 03MAT5.
@USE 11,03MAT6.
@ASG,A 03MAT6.
@USE 12,03MAT7.
@ASG,A 03MAT7.
### 4-4. OUTPUT DESCRIPTIONS AND SAMPLE OUTPUT

These outputs are of the same types as the outputs of POINTCOMP and are comprised of variable values and gradients. The two routines differ in the point selection algorithm.

#### a. Sample Output—Variable Values and Output Values

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>5.00000</th>
<th>.50000</th>
<th>50.50000</th>
<th>.50000</th>
<th>34.26218</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>32</td>
<td>5.00000</td>
<td>.50000</td>
<td>50.50000</td>
<td>.60000</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>33</td>
<td>5.00000</td>
<td>.50000</td>
<td>50.50000</td>
<td>.70000</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>34</td>
<td>5.00000</td>
<td>.50000</td>
<td>50.50000</td>
<td>.80000</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>35</td>
<td>5.00000</td>
<td>.50000</td>
<td>50.50000</td>
<td>.90000</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>36</td>
<td>5.00000</td>
<td>.50000</td>
<td>50.50000</td>
<td>1.00000</td>
</tr>
</tbody>
</table>

### VARIABLE NUMBER 2 HAS BEEN INCREMENTED

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>5.00000</th>
<th>2.00000</th>
<th>.50000</th>
<th>.50000</th>
<th>1.10039</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>38</td>
<td>5.00000</td>
<td>2.00000</td>
<td>.50000</td>
<td>1.20860</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>39</td>
<td>5.00000</td>
<td>2.00000</td>
<td>.50000</td>
<td>1.31727</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>40</td>
<td>5.00000</td>
<td>2.00000</td>
<td>.50000</td>
<td>1.42641</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>41</td>
<td>5.00000</td>
<td>2.00000</td>
<td>.50000</td>
<td>1.53502</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>42</td>
<td>5.00000</td>
<td>2.00000</td>
<td>.50000</td>
<td>1.64610</td>
</tr>
</tbody>
</table>
VARIABLE NUMBER 3 HAS BEEN INCREMENTED

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>5.00000</th>
<th>2.00000</th>
<th>10.50000</th>
<th>2.00000</th>
<th>10.50000</th>
<th>7.16476</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>44</td>
<td>5.00000</td>
<td>2.00000</td>
<td>10.50000</td>
<td>.60000</td>
<td>7.08334</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>45</td>
<td>5.00000</td>
<td>2.00000</td>
<td>10.50000</td>
<td>.70000</td>
<td>7.00158</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>46</td>
<td>5.00000</td>
<td>2.00000</td>
<td>10.50000</td>
<td>.80000</td>
<td>6.91946</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>47</td>
<td>5.00000</td>
<td>2.00000</td>
<td>10.50000</td>
<td>.90000</td>
<td>6.83699</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>48</td>
<td>5.00000</td>
<td>2.00000</td>
<td>10.50000</td>
<td>1.00000</td>
<td>6.75416</td>
</tr>
</tbody>
</table>

In each row, the four columns to the right of the point number contain the four input variable values, and the fifth contains the corresponding output value from the routine being tested. The first variable, as usual, is the leftmost, i.e., in the column to the right of the point number. Note that in each group, the last (fourth) variable is being varied through its range while the others remain fixed. The headings explain how this group differs from the preceding group. Whenever the first, second, or third variable is incremented, the variables to its right are reset to their initial values.

b. **Sample Output---Gradients**

VARIABLE NUMBER 3 HAS BEEN INCREMENTED

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>.26871</th>
<th>-2.38501</th>
<th>.67048</th>
<th>-2.62300</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>.32286</td>
<td>-2.86548</td>
<td>.66454</td>
<td>-2.62653</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.37827</td>
<td>-3.34741</td>
<td>.65860</td>
<td>-2.63007</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.43280</td>
<td>-3.83228</td>
<td>.65264</td>
<td>-2.63362</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.48479</td>
<td>-4.31738</td>
<td>.64668</td>
<td>-2.63718</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.54242</td>
<td>-4.80347</td>
<td>.64071</td>
<td>-2.64073</td>
</tr>
</tbody>
</table>

VARIABLE NUMBER 2 HAS BEEN INCREMENTED

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>.02158</th>
<th>.18565</th>
<th>.60644</th>
<th>1.08123</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>.02576</td>
<td>.22479</td>
<td>.58747</td>
<td>1.08586</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.03090</td>
<td>.26340</td>
<td>.56843</td>
<td>1.09051</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.03513</td>
<td>.30235</td>
<td>.54930</td>
<td>1.09520</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.03931</td>
<td>.34164</td>
<td>.53010</td>
<td>1.09991</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.04344</td>
<td>.38127</td>
<td>.51081</td>
<td>1.10466</td>
</tr>
</tbody>
</table>

VARIABLE NUMBER 3 HAS BEEN INCREMENTED

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>.15683</th>
<th>-.14934</th>
<th>.60644</th>
<th>-.81352</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>.18874</td>
<td>-.17997</td>
<td>.58747</td>
<td>-.81700</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.22084</td>
<td>-.21085</td>
<td>.56843</td>
<td>-.82051</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.25426</td>
<td>-.24200</td>
<td>.54930</td>
<td>-.82403</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.28688</td>
<td>-.27446</td>
<td>.53010</td>
<td>-.82758</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>.31969</td>
<td>-.30626</td>
<td>.51081</td>
<td>-.83115</td>
</tr>
</tbody>
</table>
The point numbers link the gradients to the variable values at which the gradients were computed.

c. **Sample Output Files.** On unit 10--points in standard format.

```
1260 4 .001000
(6(7X,F13.5))
5.00000 .50000 .50000 .50000 .73821
5.00000 .50000 .50000 .50000 .77276
5.00000 .50000 .50000 .70000 .80735
5.00000 .50000 .50000 .80000 .84199
5.00000 .50000 1.00000 .90000 .87667
5.00000 .50000 10.50000 .50000 7.44301
5.00000 .50000 10.50000 .60000 7.41817
5.00000 .50000 10.50000 .70000 7.39330
```

The first line is the standard heading, component of:

(1) The number of points in I5 format.

(2) The number of variables in I3 format.

(3) The zero approximator in F10.7 format.

The second line is the format used to write/read the file. The following lines comprise the variable values and corresponding output values.

d. **On Unit 11--Gradients in Standard Format**

```
1260 4 .001000
(6(7X,F13.5))
  .00048  .30919  .67048  .34638
  .00055  .37265  .66454  .34684
  .00061  .43536  .65859  .34731
  .00067  .49823  .65264  .34778
  .00072  .56128  .64668  .34825
  .00076  .62541  .64071  .34872
  .00083  -.22734 -.37265 -.34894
  .00086  -.27318 -.37265 -.34905
  .00093  -.31915 -.37265 -.34938
  .07469  -.31915  .65859  -.24972
```

The first two lines are as described previously. The following lines contain gradient values corresponding to the variable values contained in the previously described file.
e. On Unit 12--File Used to Communicate with DIFFQUOT and REARRANGE

<table>
<thead>
<tr>
<th>Variable Values</th>
<th>Output Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.00000</td>
<td>.50000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.50000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.50000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.50000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.50000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.50000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.50000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.60000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.70000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.80000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.90000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.10.00000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.10.50000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.10.50000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.10.50000</td>
</tr>
<tr>
<td>5.00000</td>
<td>.10.50000</td>
</tr>
<tr>
<td>.73821</td>
<td>4.00100</td>
</tr>
<tr>
<td>.77276</td>
<td>4.00100</td>
</tr>
<tr>
<td>.80735</td>
<td>4.00100</td>
</tr>
<tr>
<td>.84199</td>
<td>4.00100</td>
</tr>
<tr>
<td>.87667</td>
<td>4.00100</td>
</tr>
<tr>
<td>.91140</td>
<td>4.00100</td>
</tr>
<tr>
<td>7.44301</td>
<td>3.00100</td>
</tr>
<tr>
<td>7.41817</td>
<td>4.00100</td>
</tr>
<tr>
<td>7.39330</td>
<td>4.00100</td>
</tr>
</tbody>
</table>

In each row, the first four columns represent variable values. The fifth column contains output values. The integer portion of the sixth number is the variable being varied in obtaining the values for that row. The rightmost column is used by REARRANGE and DIFFQUOT. This file is written and may be read by the format located at the second line of the standard format files on units 10 and 11.

f. On Unit 13--Scratch File Containing Gradients

<table>
<thead>
<tr>
<th>Gradient Components</th>
<th>Variable Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>.00048</td>
<td>.30919</td>
</tr>
<tr>
<td>.00055</td>
<td>.37265</td>
</tr>
<tr>
<td>.00061</td>
<td>.43536</td>
</tr>
<tr>
<td>.00067</td>
<td>.49823</td>
</tr>
<tr>
<td>.00072</td>
<td>.56128</td>
</tr>
<tr>
<td>.00076</td>
<td>.62451</td>
</tr>
<tr>
<td>.005325</td>
<td>-.22734</td>
</tr>
<tr>
<td>.006396</td>
<td>-.27318</td>
</tr>
<tr>
<td>.007469</td>
<td>-.31915</td>
</tr>
<tr>
<td>.34638</td>
<td>.67048</td>
</tr>
<tr>
<td>.34684</td>
<td>.66454</td>
</tr>
<tr>
<td>.34731</td>
<td>.65859</td>
</tr>
<tr>
<td>.34778</td>
<td>.65264</td>
</tr>
<tr>
<td>.34825</td>
<td>.64668</td>
</tr>
<tr>
<td>.34872</td>
<td>.64071</td>
</tr>
<tr>
<td>-.24905</td>
<td>.67048</td>
</tr>
<tr>
<td>-.24938</td>
<td>.66454</td>
</tr>
<tr>
<td>-.24972</td>
<td>.65859</td>
</tr>
<tr>
<td>.73821</td>
<td>.34638</td>
</tr>
<tr>
<td>.77276</td>
<td>.34684</td>
</tr>
<tr>
<td>.80735</td>
<td>.34731</td>
</tr>
<tr>
<td>.84199</td>
<td>.34778</td>
</tr>
<tr>
<td>.87667</td>
<td>.34825</td>
</tr>
<tr>
<td>.91140</td>
<td>.34872</td>
</tr>
<tr>
<td>7.44301</td>
<td>.67048</td>
</tr>
<tr>
<td>7.41817</td>
<td>.66454</td>
</tr>
<tr>
<td>7.39330</td>
<td>.65859</td>
</tr>
</tbody>
</table>

The first four columns contain the gradient components. The fifth column contains the variable values corresponding to the input variable values at which the gradient was evaluated. This file was also written and may be read using the format on the second line of the files on units 10 and 11.
PARAMETER INFILE=5,OUTFILE=6,FILE1=10,FILE2=11,FILE3=12,FILE4=13
PARAMETER NOVALS=20,NOVAL1=NOVALS+1,NOVAL2=NOVAL1+1

DIMENSION FSTVAL(NOVAL1),LSTVAL(NOVALS),STEPS(NOVALS),VALUE(NOVALS)
DIMENSION COUNTS(NOVALS),LINE(NOVAL2),GRAD(NOVALS)

INTEGER NOVAR,SWITCH,POINTS,COUNTS
INTEGER PTR1,PTR2

REAL FSTVAL,LSTVAL,STEPS,VALUE,LINE
REAL EPSLON,PARTYL,INCR,PTR

CHARACTER*80 FMTRO,FMTFRT,FMTFIL,FMTDBG

C

C  Initialization

REWIND FILE1
REWIND FILE2
REWIND FILE3
REWIND FILE4
READ(INFILE, 10000) NOVAR, EPSLON, SWITCH

10000 FORMAT (I3,F10.7) READ(INFILE, 10010) FMTRO

10010 FORMAT(A80)

READ(INFILE,FMTRO) (FSTVAL(V),V=1,NOVAR)
READ(INFILE,FMTRO) (LSTVAL(V),V=1,NOVAR)
READ(INFILE,FMTRO) (STEPS(V),V=1,NOVAR)
POINTS=0
READ(INFILE, 10011) FMTFRT
READ(INFILE, 10010) FMTFIL

IF (SWITCH .GT. -1) READ(INFILE, 10011) FMTFIL
DU 100 V=1,NOVAR
VALUE(V)=FSTVAL(V)

100 CONTINUE

C **************

IF (SWITCH .NE. -1) CONTINUE

100 CONTINUE

C  ***...

IF (SWITCH .GT. -1) CONTINUE

1000 FORMAT(A1X,'** FIRST VALUES **') WRITE(OUTFILE,10020)

10020 FORMAT(A1X,'** ECHO PRINT OUT **') WRITE(OUTFILE, 10025) NOVAR, EPSLON, SWITCH

10025 FORMAT(A1X,'** ECHO PRINT OUT **') WRITE(OUTFILE, 10030)

10030 FORMAT(A1X, '** FIRST VALUES **') WRITE(OUTFILE,FMTDBG) (FSTVAL(V),V=1,NOVAR)

10040 FORMAT(A1X, '** LAST VALUES **') WRITE(OUTFILE,FMTDBG) (LSTVAL(V),V=1,NOVAR)

10050 FORMAT(A1X, '** INCREASES **') WRITE(OUTFILE,FMTDBG)
WRITE(OUTFIL,FMTDBG) (STEPS(V),V=1,NOVARS)
20000 CONTINUE
C ************
  FPTR=FLOAT(NOVAR$1)*EPSLON
  GOTO 500
C
C TO COMPUTE THE NEXT SET OF VARIABLE VALUES
C
200 DO 30 PTR=NOVAR$1+1
  PTR1=PTR
  FPTR=FLOAT(PTR1)*EPSLON
  IF (VALUI(PTR1) *LT. LSTVAL(PTR)) GOTO 400
  VALUI(PTR1)=FSTVAL(PTR)
300 CONTINUE
  GOTO 600
400 VALU(PTR1)=VALU(PTR1)+TEPSIPT1IV
500 CONTINUE
  POINTS=POINTS+1
  CALL PREPR(VALUI,VALUE)
  WRITE(OUTFIL,FMTFIL) (VALU(V),V=1,NOVAR$1,VALUE,FPTR)
DO 550 PZ1,POINTS
  CALL PARTLY(NOVAR$1,VALUE,EP$10N,PARTY,INCR,VALUE)
  GRAD(V)=PARTY
C ************
  IF (SWITCH .NE. -1) GOTO 20010
  WRITE(OUTFIL,FMTDBG) PARTY,VALUE,INCR
20010 CONTINUE
C ************
550 CONTINUE
  WRITE(OUTFIL,FMTFIL) (GRAD(V),V=1,NOVAR$1,VALUE)
C RETURN TO COMPUTE THE NEXT SET OF VARIABLE VALUES
  GOTO 200
E00 REWIND FILE3
  REWIND FILE4
  WRITE(OUTFIL,FMTDBG) POINTS,NOVAR$1,EP$10N
10070 FORMAT(15,F10.7) WRITE(OUTFIL,FMTDBG) POINTS,NOVAR$1,EP$10N
WRITE(OUTFIL,FMTDBG) POINTS
WRITE(OUTFIL,FMTDBG) POINTS
WRITE(OUTFIL,FMTDBG) POINTS
10075 FORMAT(/,10X,'THE *15* POINTS AND OUTPUT VALUES*'/)
DO 800 PZ1,POINTS
  READ(OUTFIL,FMTDBG) (LINV(V),V=1,NOVAR$1,LINV(NOVAR$1),LINV(NOVAR$1),LINV(NOVAR$1))
  IF (PTR .EQ. NOVAR$1) GOTO 700
4-9
WRITE(OUTFIL,'(U035)') PTR
10030 FORMAT(*1X**VARIABLE NUMBER **I3,** HAS BEEN INCREMENTED**,
1 /)
700 WRITE(OUTFIL,FMTFIL) P*(LINE(V),V=1,NOVARS),LINE(NOVAL1)
WRITE(FILE1,FMTFIL) (LINE(V),V=1,NOVARS),LINE(NOVAL1)
800 CONTINUE
WRITE(OUTFIL,'(U035)')
9035 FORMAT(*31X**THE CORRESPONDING GRADIENTS**,
REWIND FILE3
DO 300 P=1,POINTS
READ(FILE1,FMTFIL) (LINE(V),V=1,NOVARS),LINE(NOVAL1),LINE(NOVAL2)
PTR=INT(LINE(NOVAL2))
IF (.NOT. EQ. NOVARS) GOTO 850
WRITE(OUTFIL,'(U035)') PTR
850 READ(FILE4,FMTFIL) (LINE(V),V=1,NOVARS)
WRITE(OUTFIL,FMTFIL) P*(LINE(V),V=1,NOVARS)
WRITE(FILE2,FMTFIL) (LINE(V),V=1,NOVARS)
900 CONTINUE
WRITE(OUTFIL,'(U035)')
10030 FORMAT(*)
END
4-6. GRID ROUTINE FLOWCHART

Start

Rewind all files used

Read in the number of variables, zero estimator and debug mode indicator

Read in the format for reading the input deck

Read in the initial variable values, outer bounds and steps

Set the point count to zero

Read the print and file I/O formats

In debug mode?

Yes

Read in the debug format

No

Set the variable value vector to the initial values
In debug mode

Yes

Write out the input values

Set up to write the flotted value of the last variable

Set index to point to the last component

Store index value

Float index value incremented slightly

Reset vector component to initial value

Index below bound?

Yes

Increment variable value by step

Increment the point count

No

Index points to first variable?

Yes

Set index to point to next lowest variable

No
Evaluate the tested routine

Write out the variable values, output value and variable no. to unit 12 file

Set pointer to first component

Compute a partial

In debug mode?

Yes

Write partial and other debugging information

Last partial?

Yes

Write gradient to unit 13 file

Set to next variable

No
Rewind the scratch file.
Write the standard file headings.
Read the variable values, output values, and variable varied indicator for next point.
Set pointer to point to the variable varied.

Last variable?

Yes

Write heading on print file. Start new output group.
Write out the print no., the variable values and the output value.
Write the variable and output value into the point file.

No

Complete last point?

Yes

No
Print gradient heading

Rewind FILE3

Read in the next variable varied indicator

Read in the next gradient

Write the point number and gradient

Write the gradient into FILE4

Liest gradient?

Space three lines

Stop
4-7. INTRODUCTION. The output of the GRID routine is so organized that within each group of output lines, the lines agree in all variable values except the last, i.e., the last variable is varied most frequently. The REARRANGE routine reorders the output, grouping on each variable in turn, except the last. This re-ordering facilitates the analysis of the output.

4-8. LIMITATIONS

a. The currently compiled version is limited to 20 variables.

b. Since this routine utilizes output from GRID, the limitations of the GRID routine are applicable.

c. This routine uses much I/O, so the smaller the number of points produced by GRID, the better.

4-9. RUN SETUPS

a. To Execute

@USE 10, name of standard format file produced by GRID on unit 10.

@ASG,A name of above file.

@USE 12, name of communication file produced by GRID on unit 12.

@ASG,A name of above file.

@USE 14, name of file to be rearranged. This file should be the nonstandard format file produced by GRID on units 12 (points) or 13 (gradients).

@ASG,A name of this file.

@USE 15, name of scratch file into which REARRANGE places outputs.

@ASG,A name of scratch file.
@XQT name of absolute file developed earlier.

[Input deck]

b. Description and Sample of Input Deck

(1) Line 1. ('POINT NUMBER',15,2X,7(F10.5,5X),/21X,8(F10.5,5X))

This format will bring out the point number and regrouped data.

(2) Line 2. (15,2X,7(F10.5,5X),/21X,8(F10.5,5X))

This format will be used to printout the file of reordered data.

c. Sample Input File

(1) Unit 10

1260 4 .0010000
(6(7X,F13.5))

5.00000 .50000 .50000 .50000 .50000 .73821
5.00000 .50000 .50000 .50000 .60000 .77276
5.00000 .50000 .50000 .70000 .70000 .80735
5.00000 .50000 .50000 .80000 .80000 .84199
5.00000 .50000 .50000 .90000 .90000 .87667
5.00000 .50000 1.00000 1.00000 1.00000 .91140
5.00000 .50000 10.50000 10.50000 .50000 7.44301
5.00000 .50000 10.50000 10.50000 .60000 7.41817
5.00000 .50000 10.50000 10.50000 .70000 7.39330
5.00000 .50000 10.50000 10.50000 .80000 7.36839
5.00000 .50000 10.50000 10.50000 .90000 7.34346

This is a standard format file of variable values and an output value. This file is developed by GRID on unit 10, and is described in the GRID documentation.
This communication file is developed by GRID on unit 12 and is described in the GRID documentation.

(3) Unit 14. This is the file to be rearranged. It may be the GRID file produced on unit 12 described earlier or the analogous nonstandard format gradient file produced by GRID on unit 13. A portion of the gradient file follows:

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>.00048</td>
<td>.30919</td>
<td>.67048</td>
<td>.34638</td>
<td>.73821</td>
<td></td>
</tr>
<tr>
<td>.00055</td>
<td>.37265</td>
<td>.66454</td>
<td>.34684</td>
<td>.77276</td>
<td></td>
</tr>
<tr>
<td>.00061</td>
<td>.43536</td>
<td>.65859</td>
<td>.34731</td>
<td>.80735</td>
<td></td>
</tr>
<tr>
<td>.00067</td>
<td>.49023</td>
<td>.65264</td>
<td>.34778</td>
<td>.84199</td>
<td></td>
</tr>
<tr>
<td>.00072</td>
<td>.56128</td>
<td>.64668</td>
<td>.34825</td>
<td>.87667</td>
<td></td>
</tr>
<tr>
<td>.00076</td>
<td>.62451</td>
<td>.64071</td>
<td>.34872</td>
<td>.91140</td>
<td></td>
</tr>
<tr>
<td>.05325</td>
<td>-.22734</td>
<td>.67048</td>
<td>-.24905</td>
<td>7.44301</td>
<td></td>
</tr>
<tr>
<td>.06396</td>
<td>-.27318</td>
<td>.66454</td>
<td>-.24938</td>
<td>7.41817</td>
<td></td>
</tr>
</tbody>
</table>

d. Sample Run Stream

@USE 10,03MAT5.
@UASG,A 03MAT5.
@USE 12,03MAT7.
@ASG,A 03MAT7.
@USE 14,03MAT7.
@ASG,A 03MAT7.
@USE 15,03MAT9.
@ASG,A 03MAT9.
4-10. OUTPUT DESCRIPTIONS AND SAMPLE OUTPUT

a. Printed Output

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>VARIABLE 1</th>
<th>VARIABLE 2</th>
<th>VARIABLE 3</th>
<th>VARIABLE 4</th>
<th>VARIABLE 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>1</td>
<td>0.00048</td>
<td>0.30919</td>
<td>0.67048</td>
<td>0.34638</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>253</td>
<td>0.00019</td>
<td>0.32471</td>
<td>0.68200</td>
<td>0.34773</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>505</td>
<td>0.00010</td>
<td>0.33081</td>
<td>0.68706</td>
<td>0.34836</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>757</td>
<td>0.00006</td>
<td>0.33495</td>
<td>0.68991</td>
<td>0.34871</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>1009</td>
<td>0.00004</td>
<td>0.33762</td>
<td>0.69173</td>
<td>0.34894</td>
</tr>
</tbody>
</table>

END OF VARIATION FOR INITIAL POINT 1

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>VARIABLE 1</th>
<th>VARIABLE 2</th>
<th>VARIABLE 3</th>
<th>VARIABLE 4</th>
<th>VARIABLE 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>2</td>
<td>0.00055</td>
<td>0.37265</td>
<td>0.66454</td>
<td>0.34684</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>254</td>
<td>0.00022</td>
<td>0.38998</td>
<td>0.67838</td>
<td>0.34892</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>506</td>
<td>0.00012</td>
<td>0.39826</td>
<td>0.68446</td>
<td>0.34856</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>758</td>
<td>0.00007</td>
<td>0.40214</td>
<td>0.68788</td>
<td>0.34887</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>1010</td>
<td>0.00005</td>
<td>0.40531</td>
<td>0.69006</td>
<td>0.34908</td>
</tr>
</tbody>
</table>

END OF VARIATION FOR INITIAL POINT 2

This particular output file is a file of gradients, a reordered version of the gradient file produced by GRID. The gradients in the first group were evaluated at points whose values for variables 2-4 are identical. Adjacent gradients were evaluated at points whose variable 1 coordinates differ by the step value for variable 1. The gradients in the second group are similar, these gradients were also evaluated at points whose coordinates 2 through 4 are identical and where adjacent gradients were evaluated at points differing only in the variable 1 coordinate and the difference is the step size. Note that the fifth column contains the corresponding output values. The points at which these gradients were evaluated are exhibited below.
<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>VARIABLE NUMBER 1</th>
<th>VARIABLE NUMBER 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.00000 .50000 .50000 .50000 .73821</td>
<td></td>
</tr>
<tr>
<td>253</td>
<td>8.50000 .50000 .50000 .50000 .73934</td>
<td></td>
</tr>
<tr>
<td>505</td>
<td>12.00000 .50000 .50000 .50000 .73985</td>
<td></td>
</tr>
<tr>
<td>757</td>
<td>15.50000 .50000 .50000 .50000 .74014</td>
<td></td>
</tr>
<tr>
<td>1009</td>
<td>19.00000 .50000 .50000 .50000 .74033</td>
<td></td>
</tr>
</tbody>
</table>

END OF VARIATION FOR INITIAL POINT 1

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>VARIABLE NUMBER 1</th>
<th>VARIABLE NUMBER 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5.00000 .50000 .50000 .60000 .77276</td>
<td></td>
</tr>
<tr>
<td>254</td>
<td>8.50000 .50000 .50000 .60000 .77406</td>
<td></td>
</tr>
<tr>
<td>506</td>
<td>12.00000 .50000 .50000 .60000 .77465</td>
<td></td>
</tr>
<tr>
<td>758</td>
<td>15.50000 .50000 .50000 .60000 .77498</td>
<td></td>
</tr>
<tr>
<td>1010</td>
<td>19.00000 .50000 .50000 .60000 .77520</td>
<td></td>
</tr>
</tbody>
</table>

END OF VARIATION FOR INITIAL POINT 2

Note that within each group, only variable 1 varies. The following printout is a later segment of the same output.

FINISHED VARIABLE NUMBER 1

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>VARIABLE NUMBER 1</th>
<th>VARIABLE NUMBER 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.00048 .30819 .67048 .34638 .73821</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>.02158 .18565 .60644 1.08123 1.10039</td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>.05104 .12313 .56450 1.55265 1.32998</td>
<td></td>
</tr>
<tr>
<td>109</td>
<td>.07778 .08812 .53491 1.88027 1.48843</td>
<td></td>
</tr>
<tr>
<td>145</td>
<td>.10199 .06543 .51291 2.12195 1.60434</td>
<td></td>
</tr>
<tr>
<td>181</td>
<td>.12181 .05001 .49491 2.30725 1.69280</td>
<td></td>
</tr>
<tr>
<td>217</td>
<td>.13869 .04014 .48239 2.45384 1.76252</td>
<td></td>
</tr>
</tbody>
</table>

NEW INITIAL POINT

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>VARIABLE NUMBER 1</th>
<th>VARIABLE NUMBER 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>253</td>
<td>.00019 .32471 .68200 .34773 .73934</td>
<td></td>
</tr>
<tr>
<td>289</td>
<td>.00997 .23380 .63786 1.18644 1.15472</td>
<td></td>
</tr>
<tr>
<td>325</td>
<td>.02581 .17577 .60447 1.81117 1.46142</td>
<td></td>
</tr>
<tr>
<td>361</td>
<td>.04306 .13756 .57832 2.29508 1.69706</td>
<td></td>
</tr>
<tr>
<td>397</td>
<td>.05990 .10961 .55728 2.67928 1.88374</td>
<td></td>
</tr>
<tr>
<td>433</td>
<td>.07674 .09007 .54000 2.99334 2.03525</td>
<td></td>
</tr>
<tr>
<td>469</td>
<td>.09131 .07532 .52555 3.25328 2.16066</td>
<td></td>
</tr>
</tbody>
</table>

NEW INITIAL POINT
These gradients were evaluated on point groups where only variable 2 varied within each group. The "NEW INITIAL POINT" message indicates that variable 1 varied between points 217 and 253, so these points vary in two coordinates—hence it's time to start a new grouping. This situation will not occur for variable 1 because variable 1 is varied the least. The following printout exhibits the points at which the gradients were evaluated.

FINISHED VARIABLE NUMBER 1

+++)VARIABLE NUMBER 2+++-------------------------------------------------

NEW INITIAL POINT

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>1</th>
<th>5.00000</th>
<th>.50000</th>
<th>.50000</th>
<th>.50000</th>
<th>.73821</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>37</td>
<td>5.00000</td>
<td>2.00000</td>
<td>.50000</td>
<td>.50000</td>
<td>1.10037</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>73</td>
<td>5.00000</td>
<td>3.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>1.32998</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>109</td>
<td>5.00000</td>
<td>5.00000</td>
<td>.50000</td>
<td>.50000</td>
<td>1.48843</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>145</td>
<td>5.00000</td>
<td>6.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>1.60434</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>181</td>
<td>5.00000</td>
<td>8.00000</td>
<td>.50000</td>
<td>.50000</td>
<td>1.69280</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>211</td>
<td>5.00000</td>
<td>9.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>1.76252</td>
</tr>
</tbody>
</table>

NEW INITIAL POINT

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>253</th>
<th>8.50000</th>
<th>.50000</th>
<th>.50000</th>
<th>.50000</th>
<th>.73934</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>289</td>
<td>8.50000</td>
<td>2.00000</td>
<td>.50000</td>
<td>.50000</td>
<td>1.15472</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>325</td>
<td>8.50000</td>
<td>3.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>1.46142</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>361</td>
<td>8.50000</td>
<td>5.00000</td>
<td>.50000</td>
<td>.50000</td>
<td>1.69706</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>397</td>
<td>8.50000</td>
<td>6.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>1.88374</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>433</td>
<td>8.50000</td>
<td>8.00000</td>
<td>.50000</td>
<td>.50000</td>
<td>2.03525</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>469</td>
<td>8.50000</td>
<td>9.50000</td>
<td>.50000</td>
<td>.50000</td>
<td>2.16066</td>
</tr>
</tbody>
</table>

Note that all points in each group vary only in the second variable. Note also that, as before, while the point numbers continue increasing, points 217 and 253 differ in two coordinates since variables 1 and 2 change simultaneously. This fact necessitates creating a new group.

b. Sample Output File Segment

<table>
<thead>
<tr>
<th>POINT NUMBER</th>
<th>252</th>
<th>5.00000</th>
<th>9.50000</th>
<th>50.50000</th>
<th>1.00000</th>
<th>15.70270</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT NUMBER</td>
<td>504</td>
<td>8.50000</td>
<td>9.50000</td>
<td>50.50000</td>
<td>1.00000</td>
<td>21.01859</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>756</td>
<td>12.00000</td>
<td>9.50000</td>
<td>50.50000</td>
<td>1.00000</td>
<td>24.55868</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>1008</td>
<td>15.50000</td>
<td>9.50000</td>
<td>50.50000</td>
<td>1.00000</td>
<td>27.08553</td>
</tr>
<tr>
<td>POINT NUMBER</td>
<td>1260</td>
<td>19.00000</td>
<td>9.50000</td>
<td>50.50000</td>
<td>1.00000</td>
<td>28.97974</td>
</tr>
</tbody>
</table>

252+++)-------------------------------------------------------------------------------------

1 //////////////////////////////////////////////////////////////////////////////////////
The leftmost integers are point numbers.

(1) The line
252 + + + ... indicates the end of the grouping whose initial point was 252, i.e., if there was a point 253, the point numbers, now at 1260, would decrease to 253 next and commence to increase from that value, i.e., the next point numbers would be:

253
289
325
etc.

(2) The line
1 /// ... indicates the end of regroupings for which only variable 1 varies within each preceding group.

(3) The line
VARIABLE 2 indicates that now groupings where only variable 2 varies will be derived.

(4) The line
---- . . . ----NEW INITIAL POINT---- . . . indicates that while the point numbers may continue to increase, a new grouping must nevertheless begin.
4-11. REARRANGE ROUTINE LISTING

PARAMETER INFILE=5,OUTFIL=6,FILE1=10,FILE3=12,SRTFIL=14,GRPFILE=15
PARAMETER NOVALS=20,NOVAL2=NOVALS+1,NOVAL3=NOVALS+2
C
CHARACTER*80 FMTFIL,FMTGRP,FMTGRP
C
INTEGER POINTS,PTR,STEP,STEPN,PERIOD,FI,GI,QI,P,V
INTEGER NOVRM1,NOVARS,NOVRP1,NOVRP2
C
REAL LINE
C
DIMENSION PERIOD(NOVALS),LINE(NOVAL2)
C
C INITIALIZATION
C
REWIND FILE1
REWIND FILE3
REWIND SRTFIL
REWIND GRPFILE
READ(FILE1,10000) POINTS,NOVARS
10000 FORMAT(15I1I)
READ(FILE1,10100) FMTFIL
10100 FORMAT(60I1)
READ(INFILE,10200) FMTPRT
READ(INFILE,10210) FMTGRP
NOVRP1=NOVARS+1
NOVRP2=NOVARS+2
NOVRM1=NOVARS-1
C
C TO FIND THE PERIOD
C
DO 100 V=1,NOVARS
PERIOD(V)=0 100 CONTINUE
DO 200 P=1,POINT
READ(FILE3,200) LINE(I),I=1,NOVRP2
PRT=INT(LINE(NOVRP2))
IF(PRTR=EQ.0) PERIOD(PRT)=MAX(P-1,1)
IF(PERIOD(PRT) .NE. 0) GOTO 300
200 CONTINUE
C
C TO PRODUCE A REORDERED FILE, VARYING EACH VARIABLE THROUGHOUT ITS RANGE IN TURN
C
300 DO 600 F=1,NOVRM1
WRITE(OUTFIL,10020) F
600 FORMAT(15X,'VARIABLE NUMBER *',I5,2X,'TOTAL REMAINING *',I5)
WRITE(SRTFIL,10030) F
10030 FORMAT('VARIABLE *',I5)
FUTEST: A COMPUTER SYSTEM FOR THE ANALYSIS OF COMPUTATIONAL C--ETC(U)

UNCLASSIFIED CAA-D-80-1

END
DATE
TIME
12-80
DTC
STEPM=0
STEP=PERIOD(F)
IF IF .NE. 1) STEPM=PERIOD(F-1)

C
C TO PICK UP THE Q TH ENTRY FROM EACH BLOCK OF SIZE STEP
C
DO 500 Q=1,STEP
REWIND SRTFIL
Q1=Q
IF (Q1 .EQ. STEP) Q1=0
GO TO 300 P=1+POINTS
PEAD(SRTFIL,FMTFIL) (LINE(V),V=1,NOVRP1)
IF (MOD(P,STEP) .NE. 01) GOTO 400
IF (Q1 .EQ. 0) GOTO 350

C
C CHECK FOR VARIATION IN THE NEXT VAP TABLE
C
IF (IF .NE. 11 .AND. (MOD(P,STEP) .EQ. Q1)) WRITE(OUTFIL,10032)
10032 FORMAT(*/NEW INITIAL POINT*/)
IF (IF .NE. 11 .AND. (MOD(P,STEP) .EQ. Q1)) WRITE(OUTFIL,10036)
10036 FORMAT(55(H=14-),*NEW INITIAL POINT*,55(H=1-))
350 WRITE(OUTFIL,FMTFIL) P,(LINE(V),V=1,NOVRP1)
WRITE(OUTFIL,FMTFIL) P,(LINE(V),V=1,NOVRP1)
IF (Q1 .NE. 0 ) GOTO 400
IF (IF .NE. 11 .AND. (MOD(P,STEP) .EQ. Q1)) WRITE(OUTFIL,10032)
IF (IF .NE. 11 .AND. (MOD(P,STEP) .EQ. Q1)) WRITE(OUTFIL,10036)
400 CONTINUE
WRITE(OUTFIL,10040) 2
10040 FORMAT(*/END OF VARIATION FOR INITIAL POINT */)
WRITE(OUTFIL,10050) 0
10050 FORMAT(15+120(H+))
C
C FINISHED PASS FOR THE Q TH ENTRY IN EACH BLOCK
C
500 CONTINUE
WRITE(OUTFIL,10060) F
10060 FORMAT(*/FINISHED VARIABLE NUMBER */)
WRITE(OUTFIL,10070) F
10070 FORMAT(75,H=5+120(H+))
500 CONTINUE
WRITE(OUTFIL,10080) 0
10080 FORMAT(///)
END
4-12. REARRANGE ROUTINE FLOWCHART

Start

Rewind files

Read in the no. of points and the no. of variables

Read in the printer and file I/O formats

Initialize periodicity vector

Read next variable varied indicator from FILE3

Has this variable's period been determined?

Yes

No

Increase the point count

Use point count to set the period

All periods known

Yes

No
Set up to analyze first variable

Print the number of the variable grouped upon

Write the number of the variable being grouped upon into the output file

Determine the period of the varied variable and its lower numbered neighbor

Initialize initial pointer

Rewind file to be rearranged

Set up to read first line of file to be rearranged

Read the indicated line of file

Is this the next point?

No

Yes
Yes

Print new initial point heading

Write new initial point indicator on output file

Print the point number and data

Write the point number and data on the output file

Yes

Last initial point

New initial point

No

Set up to read next point

No

End of file to be reordered?

Yes

(1)
Print "End of variation for initial point" message

Write the analogous line into the output file

Set up for next initial point

Last initial point

Print "Finished variable number" message

Write the analogous line on the output file

Set up for next variable

Last variable?

Space three lines

Stop...
Section III. THE DIFFQUOT ROUTINE

4-13. INTRODUCTION. This routine utilizes the output developed by GRID on unit 10 in order to compute difference quotients for one variable. The point numbers listed in the output identify the points used to compute the difference quotients. These points are developed by GRID, and the point numbers link the GRID output points to the difference quotient computations.

4-14. BACKGROUND. See the discussion of difference quotients in the VARVARY1 documentation (Chapter 5).

4-15. LIMITATIONS

   a. The current compiled version is limited to 20 variables.

   b. The current compiled version is limited to a maximum of 500 lines of difference quotient computations in each difference quotient block.

   c. Since this routine utilizes GRID output, the limitations applicable to GRID apply.

4-16. RUN SETUPS

   a. To Develop an Absolute ASCI Program File

      @MAP,S name of absolute element.

      IN 03PROGTEST.DIFFQUOT.

      IN 03PROGTEST.PARTIAL.

      IN element containing the driver PREPR.

      IN programs to be tested.

      LIB$*FTN8.

      END
CAA-D-80-1

b. To Execute

@USE 10, name of file produced by GRID on unit 10.
@ASG,A name of above file.
@USE 12, name of file produced by GRID on unit 12.
@ASG,A name of this file.
@XQT name of absolute deck created earlier.
[Input deck]

c. Sample Input Deck and Description

(1) Line 1.

2

The number of the variable for which difference quotients are to be computed, in I3 format.

(2) Line 2.

(13F6.0)

The format for reading each line of input data.

(3) Line 3.

5.0  0.5  0.5  0.5

Initial values for each variable, where variable 1 is leftmost.

(4) Line 4.

18.00  9.00  48.00  0.9

Terminal values for each variable, variable 1 leftmost.

(5) Line 5.

3.5  1.5  10.0  0.1

Step values for each variable.

(6) Line 6.

4-30
(1X,8F13.5)
Format for one line of difference quotient printouts.

(7) Line 7.
(1X,8(5X,I6))
Format for printing out the point numbers whose difference quotients are being computed.

d. Sample Input Files and Descriptions

(1) Unit 10

1260 4 .0010000
(6(7X,F13.5))

5.00000 .50000 .50000 .50000 .73821
5.00000 .50000 .50000 .60000 .77276
5.00000 .50000 .50000 .70000 .80735
5.00000 .50000 .50000 .80000 .84199
5.00000 .50000 .50000 .90000 .87667
5.00000 .50000 .50000 1.00000 .91140
5.00000 .50000 10.50000 .50000 7.44301
5.00000 .50000 10.50000 .60000 7.41817
5.00000 .50000 10.50000 .70000 7.39330

This file is in standard format; it contains variable values (points) and output. This file is produced by GRID on unit 10, see GRID documentation (Chapter 4) for a more detailed description.

(2) Unit 12

5.00000 .50000 .50000 .50000 .73821 4.00100
5.00000 .50000 .50000 .60000 .77276 4.00100
5.00000 .50000 .50000 .70000 .80735 4.00100
5.00000 .50000 .50000 .80000 .84199 4.00100
5.00000 .50000 .50000 .90000 .87667 4.00100
5.00000 .50000 10.50000 .50000 7.44301 3.00100
5.00000 .50000 10.50000 .60000 7.41817 4.00100
5.00000 .50000 10.50000 .70000 7.39330 4.00100

This file is produced by GRID on unit 12 and is used by DIFFQUOT to determine the periodicity of each variable. For a more detailed description, see the GRID documentation (Chapter 7).
e. Sample RUN SETUP

@USE 10,03MAT5.
@ASG,A 03MAT5.
@USE 12,03MAT7.
@ASG,A 03MAT7.
@XQT 03PROGTEST.DIFFQUOT

1 (13F6.0)
  5.0  0.5  0.5  0.5
18.00  9.00 48.00  0.9
  3.5  1.5 10.0  0.1
(1X,8F13.5)
(1X,8(5X,16))

4-17. DESCRIPTION AND SAMPLE OUTPUT

embre DIFFERENCE QUOTIENTS FOR VARIABLE 1--------- AND POINTS

<table>
<thead>
<tr>
<th></th>
<th>253</th>
<th>505</th>
<th>757</th>
<th>1009</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.00000</td>
<td>.00000</td>
<td>.00000</td>
<td>.00000</td>
</tr>
<tr>
<td>.00032</td>
<td>.00000</td>
<td>.00015</td>
<td>.00000</td>
<td>.00000</td>
</tr>
<tr>
<td>.00023</td>
<td>.00011</td>
<td>.00008</td>
<td>.00000</td>
<td>.00000</td>
</tr>
<tr>
<td>.00018</td>
<td>.00009</td>
<td>.00007</td>
<td>.00005</td>
<td>.00000</td>
</tr>
</tbody>
</table>

The following figure (4-1) may help explain the output:

The slope of the line to point

<table>
<thead>
<tr>
<th>From point</th>
<th>1</th>
<th>253</th>
<th>505</th>
<th>757</th>
<th>1009</th>
</tr>
</thead>
<tbody>
<tr>
<td>/1</td>
<td>-.0032</td>
<td>.00023</td>
<td>.00018</td>
<td>.00015</td>
<td></td>
</tr>
<tr>
<td>253</td>
<td>-.00015</td>
<td>.00011</td>
<td>.00009</td>
<td>.00007</td>
<td></td>
</tr>
<tr>
<td>505</td>
<td>-.00008</td>
<td>.00007</td>
<td>.00005</td>
<td>.00005</td>
<td></td>
</tr>
<tr>
<td>757</td>
<td>-.00005</td>
<td>.00005</td>
<td>.00005</td>
<td>.00009</td>
<td></td>
</tr>
<tr>
<td>1009</td>
<td>.Slope of the line from point no. 1 to point no. 253</td>
<td>.Slope of the line from point no. 253 to point no. 757</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 4-1. Explanatory Figure
For more details, examine the description of difference quotients in the VARVARY1 documentation (Chapter 2, Section II).

4-18. DIFFQUOT ROUTINE LISTING

```fortran
PARAMETER INFILE=3,OUTFILE=C,NOPTS=300,NOVALS=20,FILE1=10
PARAMETER FILE3=12,NOVAL1=NOVAL2=1,NOVAL2=NOVAL1+1

DIMENSION FSTVAL(NOVALS),LSTVAL(NOVALS),STEPS(NOVALS)
DIMENSION INDEX(NOVALS),VALU(NOVALS),COORDS(NOPTS,NOVALS)
DIMENSION DIFF(NOVALS),POINTS(NOPTS),CLK(NOVALS)
DIMENSION PERIOD(NOVALS)

REAL EPSLON,FSTVAL,LSTVAL,STEPS,VALU,DIFF,COORDS

INTEGER INDEX,POINTS,NOVAR,NOVAR1,NOVAR2,WHICH,VARVAR1
INTEGER LOC,PCTC,I,J,CLK,CLOCK,FMT,FMT2,NUMBER

CHARACTER*80 FMTRO,FMTROPT,FMTPG2,FMTFIL

REWIND FILE1
REWIND FILE3
READ(INFILE,10040) WHICH,VARVAR1
READ(INFILE,10040) FMTFIL

READ(INFILE,10040) WHICH
READ(INFILE,10040) FMTFIL

READ(INFILE,10040) WHICH
READ(INFILE,10040) FMTFIL

READ(INFILE,10040) WHICH
READ(INFILE,10040) FMTFIL

READ(INFILE,10040) WHICH
READ(INFILE,10040) FMTFIL

READ(INFILE,10040) WHICH
READ(INFILE,10040) FMTFIL

NOVAR1=NOVAR+1
NOVAR2=NOVAR1+1
NOVAR1=NOVAR-1
DO 100 I=1,NOVAR
CLK(I)=1
INDEX(I)=I
VALU(I)=FSTVAL(I)

100 CONTINUE
IF WHICH .EQ. NOVAR OR GOTO 110
DO 200 I=WHICH,NOVAR1
INDEX(I)=INDEX(I+1)

200 CONTINUE
INDEX(NOVAR)=WHICH

100 CONTINUE
TO 1200 I=1,NOVAR
PERIOD(I)=D

1200 CONTINUE
```

4-33
DO 1300 I=1,NUMBER
READ(FILE3,FMTF,LI)
VALU(J),J=1,NOVRP1
PTR=IN(VALU(NOVRP1))
IF (PERIOD(PTR),GE,0) PERIOD(PTR)=MAXU(I-1+1)
IF (PERIOD(I),NE,0) GOTO 1?5U
1300 CONTINUE
STOP
1150 DO 1375 I=1,NOVARS
VALU(I)=FSTVAL(I)
1375 CONTINUE
GOTO 1400
700 DO 400 VAR=NOVAPS+1+1
VAR=VAR
IF (VALU(INDEX(VAR)) ,LT, LASTVAL(INDEX(VAR))) GOTO 700
CLOCK(INDEX(VAR))=1
VALU(INDEX(VAR))=FSTVAL(INDEX(VAR))
C IF LOC IS ZERO, HAVE ALREADY PRINTED THE LAST 5FT
C OF DIFFERENCE:
IF (LOC ,EQ, 0) GOTO 400
WRITE(OUTFIL+1002U) WHICH
10020 FORMAT('/*1X*2U(IH-1),"DIFFERENCE QUOTIENTS FOR VARIABLE ",
1 */5*2X*2U(IH-1)
WRITE(OUTFIL+10025)
10025 FORMAT('IX,"AND POINTS")
WRITE(OUTFIL+FMTF,2) (POINTS(J)+J=1,LCC)
WRITE(OUTFIL+10028)
10038 FORMAT(/)
DO 500 T=1,LOC
DIFF(J)=U,
DO 600 J=1,I
IF ( J ,EQ, 1 ) GOTO 600
DIFF(J)=(COORDS(I+NOVRP1)-COORDS(J,NOVRP1))/
1 (COORDS(I+WHICH)-COORDS(J+WHICH))
600 CONTINUE
WRITE(OUTFIL+FMTF,2) (DIFF(J)+J=1,I)
500 CONTINUE
LOC=U
400 CONTINUE
WRITE(OUTFIL+10030)
10030 FORMAT(/)
STOP
700 VALU(INDEX(VAR))=VALU(INDEX(VAR))+STEPS(INDEX(VAR))
CLOCK(INDEX(VAR))=CLOCK(INDEX(VAR))+1
1000 CALL PRFPR(VALU,VALU(NOVRP1))
LOC=LOC+1
POINTS(LOC)=1
DO 1100 I=1,NOVRP1
POINTS(LOC)=POINTS(LOC)+(CLOCK(I)-1)*PERIOD(I)
1100 CONTINUE
DO 900 I=1,NOVRP1
COORDS(LOC+I)=VALU(I)
900 CONTINUE
GOTO 900
END

4-34
4-19. DIFFQUOT ROUTINE FLOWCHART

Start

Rewind files

Read in the number of points, of variables, and the zero approximator

Read in the file input format

Read in the number of the variable for which difference quotients must be computed

Read in the format for the input data

Read in the initial, terminal and step values

Read in the format for one line of difference quotient output

Read in the format for printing the point numbers

Initialize the point descriptor, indirect addressing and value vectors
1

Is indirect addressing needed?

Yes

Shift indirect addressing vector to last location points to variable being varied.

Initialize the period vector.

Read in the next variable varied indicator.

Has this variable's period been determined?

Yes

Use the line count to set the period.

No

Are all the periods determined?

Yes

Initialize the value vector.

First time?

Yes

2

No

3
1. Set up to point to last variable.

2. Is value below bound? Yes/No
   - If Yes, initialize the components of the point descriptor and value vectors.
   - If No, increment the point descriptor.

3. Is there anything to print? Yes/No
   - If Yes, write name of variable for which difference quotients were computed.
   - If No, evaluate at the output vector.

4. Write the point numbers.
   - Point to the next point.

5. Compute a line of difference quotients.

6. Write a line of difference quotients.
   - Any points left? Yes/No
     - If Yes, point to the next point.
     - If No, set quotient line count to zero.
Any variables left?

Yes
Point to the next variable

No
Space three lines

Stop
5-1. INTRODUCTION. Two subroutines are used by both the POINTCOMP and GRID subsystems:

a. PARTL
b. PREPR

Detailed writeups of these subroutines follow.

Section I. THE PARTL SUBROUTINE

5-2. INTRODUCTION TO THE PARTL SUBROUTINE. This routine computes the partial derivative at a point numerically by using the definition of the right partial derivative. PARTL is called by POINTCOMP and other routines repetitively to construct the gradient, but it may be used by anyone as a standalone subroutine. PARTL calls PREPR which must be a user-provided driver subroutine whose function is to obtain an output value from the program being tested.

5-3. BACKGROUND. Given a real valued function \( f(x_1, \ldots, x_n) \), the jth right partial derivative of \( f \) at \( (x_1, \ldots, x_n) \) is defined to be:

\[
\frac{\partial f}{\partial x_j}(x_1, \ldots, x_n) = \lim_{h \to 0^+} \frac{f(x_1, \ldots, x_{j-1}, x_j + h, x_{j+1}, \ldots, x_n) - f(x_1, \ldots, x_n)}{h}
\]

5-4. DISCUSSION OF METHODOLOGY

a. The methodology is derived directly from the definition:

For \( h = 1/2 \), we approximate the jth partial derivative at \( (x_1, \ldots, x_n) \) to be:

\[
\frac{f(x_1, \ldots, x_j-1, x_j+1/2, x_{j+1}, \ldots, x_n) - f(x_1, \ldots, x_n)}{1/2}
\]

b. We repeat this procedure for \( h = 1/4, 1/8, 1/16, \text{ etc.} \). When the difference between two successive approximations is sufficiently small (as determined by an input parameter), the process
is terminated and the approximation is returned as the value of the partial derivative. The process terminates automatically after 50 approximations.

5-5. LIMITATIONS
a. As currently compiled, all inputs to PARTL must be real.
b. The function whose partials are being computed should be differentiable.

5-6. CALLING SEQUENCE

Call PARTL (I1, I2, R1, R2, R3, R4, R5)

where

I1 is an integer input variable containing the number of variables.

I2 is an integer input variable containing the index value of the variable whose partial is to be found. The index value refers to the subscript locating the variable in the input array R1.

R1 is a real input array containing the values of the variables at the point at which the partial is to be evaluated.

R2 is a real input variable containing a positive number. Any number smaller than this number will be considered to be zero.

R3 is a real output variable into which the approximate value of the partial will be placed by PARTL.

R4 A real output variable into which the last increment tested will be placed by PARTL.

R5 is a real input variable containing the value returned by PREPR for the variable values in R1.
SUBROUTINE PARTLI(NOVAR,WHICH,VAL,EPSPARTY,INCR,BASE)

PARAMETER NOVALS=20,INFILE=5,OUTFILE=6

DIMENSION VAL(NOVALS),INCR(NOVALS)

INTEGER NOVAR,WHICH,FIRST,J

REAL VALUE,VALUE1,VALUE2,EPSPARTY,EPSPARTY,EPSPARTY

FIRST=1
INCR=1
DO 100 J=1,NOVAR
   VALUE(J)=VAL(J)
100  CONTINUE
DO 200 J=1,50
   INCR=INCR+5
   VALUE(WHICH)=VALUE(WHICH)+INCR
   CALL PREPR(VALUE)
   IF (FIRST .NE. 1) GOTO 300
   EPSPARTY=(VALUE-BASE)/INCR
   FIRST=0
   GOTO 200
200  CONTINUE
   IF (ABS((VALUE-BASE)/INCR)-EPSPARTY) .LT. EPS) RETURN
   EPSPARTY=(VALUE-BASE)/INCR
   CONTINUE
RETURN
END
5-8. PARTL SUBROUTINE FLOWCHART

Start

Set increment to 1

Halve the increment

Compute incremented point

PREPR Evaluate at incremented point

First partial approximation? Yes

Save partial value

Compare to previous approximation

Sufficient change? No

Return

Yes

Replace former by current estimate

2
Were fifty approximations made? 

Yes

Return

No

3
5-9. INTRODUCTION. This subroutine must be written by the user to provide an interface between the PROGTEST, PARTL, VARVARY1, GRID, and DIFFQUOT routines and the routine being tested. PREPR receives variable values from POINTCOMP or the other routines and returns the output value determined by calling the program being tested with the given variable values as input.

5-10. DISCUSSION. The input variable values are passed to PREPR in array V, in the same order as the values read in, i.e., the leftmost variable defined in the input is in V(1), etc.

5-11. PREPR LAYOUT

SUBROUTINE PREPR(V, VALU)
PARAMETER NOVALS = 20
DIMENSION V(NOVALS)
REAL V, VALU

Pass input values given in V to the program being tested.

Call program being tested as a subroutine using input values passed in V.

VALU = value returned by the program.

END
APPENDIX A

CONTRIBUTORS

1. DOCUMENTALIST
   Dr. Steve Bravy, Methodology and Computer Support Directorate

2. TECHNICAL SUPPORT PERSONNEL
   Mr. Joseph M. Tessmer

3. OTHER SUPPORT PERSONNEL
   Ms Carrie Allen, Word Processing Center
   Ms Julie Fuller, Word Processing Center
   Ms Bobbie Guenthner, Word Processing Center
   Ms Joyce Garris, Word Processing Center
   SSG Richard Loller, Graphic Arts Branch
   Ms Linda Prieto, Word Processing Center
## APPENDIX B

### DISTRIBUTION

<table>
<thead>
<tr>
<th>Addressee</th>
<th># of Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defense Technical Information Center</td>
<td>2</td>
</tr>
<tr>
<td>Building 5</td>
<td></td>
</tr>
<tr>
<td>Cameron Station</td>
<td></td>
</tr>
<tr>
<td>Alexandria, VA 22314</td>
<td></td>
</tr>
<tr>
<td>The Army Central Library (ASDIRS)</td>
<td>1</td>
</tr>
<tr>
<td>Room 1A600</td>
<td></td>
</tr>
<tr>
<td>Pentagon Washington, DC 20310</td>
<td></td>
</tr>
<tr>
<td>Federal Software Exchange Center</td>
<td>1</td>
</tr>
<tr>
<td>5285 Port Royal Rd.</td>
<td></td>
</tr>
<tr>
<td>Springfield, VA 22161</td>
<td></td>
</tr>
<tr>
<td>National Technical Information Service</td>
<td>1</td>
</tr>
<tr>
<td>Springfield, VA 22161</td>
<td></td>
</tr>
<tr>
<td>Command and Control Technical Center (CCTC)</td>
<td>1</td>
</tr>
<tr>
<td>Room BE685</td>
<td></td>
</tr>
<tr>
<td>ATTN: Technical Library</td>
<td></td>
</tr>
<tr>
<td>Pentagon Washington, DC 20301</td>
<td></td>
</tr>
<tr>
<td>Dept of Defense Computer Institute</td>
<td>1</td>
</tr>
<tr>
<td>Technical Library</td>
<td></td>
</tr>
<tr>
<td>Washington Navy Yard</td>
<td></td>
</tr>
<tr>
<td>Building 175, Room 37</td>
<td></td>
</tr>
<tr>
<td>Washington, DC 20374</td>
<td></td>
</tr>
<tr>
<td>Assistant for Automation</td>
<td>1</td>
</tr>
<tr>
<td>OJCS</td>
<td></td>
</tr>
<tr>
<td>Room 2A932</td>
<td></td>
</tr>
<tr>
<td>ATTN: CPT Goddard (Navy)</td>
<td></td>
</tr>
<tr>
<td>Pentagon Washington, DC 20301</td>
<td></td>
</tr>
<tr>
<td>Defense Sciences Office, Cybernetics Technology Division (DARPA)</td>
<td>1</td>
</tr>
<tr>
<td>DSO/CTD</td>
<td></td>
</tr>
<tr>
<td>1400 Wilson Blvd</td>
<td></td>
</tr>
<tr>
<td>Arlington, VA 22209</td>
<td></td>
</tr>
</tbody>
</table>
Defense Communication Engineering Center
R801 (Lt. Col. Tufts)
1860 Wichle Ave.
Reston, VA 22090

Defense Communication Agency
Command Control Technical Center/C440
11440 Isaac Newton Square, North
Reston, VA 22090

Director, Defense Nuclear Agency
ATTN: NASD Lt. Cdr Gladwin
Washington, DC 20305

Defense Advanced Research Project Agency (DARPA)
Cybernetics Technology Division
ATTN: Dr. Fields
1400 Wilson Blvd
Arlington, VA 22209

Commander
US Army and Electronic Research and
Development Command
Fort Monmouth, NJ 07703

Director
US Army TRADOC Systems Analysis Activity
ATTN: ATAA-SL
ATAA-T
White Sands Missile Range, NM 88002

Director
US Army Tank-Automotive Research and
Development Command
Warren, MI 48090

Commander
US Army Mobility Equipment Research and
Development Command
Fort Belvoir, VA 22060

Commander
US Army Computer Systems Command
Fort Belvoir, VA 22060
<table>
<thead>
<tr>
<th>Post</th>
<th>Address</th>
</tr>
</thead>
</table>
| Commander | US Army Natick Research and Development Cmd  
Natick, MA 01760 |
| Chief, | Defense Logistics Studies Information Exchange  
US Army Logistics Management Center  
Fort Lee, VA 23801 |
| Commander | US Army Logistics Center  
Fort Lee, VA 23801 |
| Commander | Army TRADOC Systems Analysis Activity  
White Sands Missile Range  
White Sands, NM 88002 |
| Office of the Deputy Chief of Staff for Research, Development and Acquisition |  
Headquarters, Department of the Army  
Pentagon  
Washington, DC 20310 |
| Director of Army Automation |  
Office Chief of Staff, Army  
Pentagon  
Washington, DC 20310 |
| Commander | Rock Island Arsenal  
ATTN: Technical Library  
Rock Island, IL 61201 |
| Commander | US Army Armament Research and Development Cmd.  
Dover, NJ 07801 |
| Commander | Harry Diamond Laboratories  
2800 Power Mill Rd  
Adelphi, MD 20783 |
Commander
US Army Test & Evaluation Command
Dugway Proving Ground, MD 84022

Commander
US Army Aviation Research and Development Cmd
P.O. Box 209
St. Louis, MO 63166

Commander
Army Training and Doctrine Command
Fort Monroe, VA 23651

Commander
Army Material Systems Analysis Agency
Aberdeen Proving Ground
Aberdeen, MD 21005

Commander
Computer Systems Command
Fort Belvoir, VA 22060

Office of the Deputy Under Secretary of the Army
for Operations Research
ATTN: Mr. Dick Lester
Pentagon Room 2E621
Washington, DC 20310

Communications Research and Development Command
Center for Tactical Computer Systems
Division for Software Engineering
ATTN: Mr. Joe Kernan
Fort Monmouth, NJ 07703

Ballistic Research Laboratory
Building 393
ATTN: Mr. Jerry Thomas
Aberdeen Proving Ground
Aberdeen, MD 21005

Ballistic Research Laboratory Technical Library
Building 305
Aberdeen Proving Ground
Aberdeen, MD 21005
Mr. Glen Ingram  
Scientific Computing Division  
Room A151 Technology Building  
National Bureau of Standards  
Washington, DC  20234  

US Army Concepts Analysis Agency  

Chief, AD, ATTN: Library Branch  
Director, FA  
Director, JF  
Director, RQ  
Director, SM  
Director, MC