ANL-7495

ANL-7495

Argonne National Laboratory

SPECIAL-PURPOSE LANGUAGE FOR LEAST-SQUARES FITS

by

Marian Gabriel

The facilities of Argonne National Laboratory are owned by the United States Government. Under the terms of a contract (W-31-109-Eng-38) between the U. S. Atomic Energy Commission, Argonne Universities Association and The University of Chicago, the University employs the staff and operates the Laboratory in accordance with policies and programs formulated, approved and reviewed by the Association.

MEMBERS OF ARGONNE UNIVERSITIES ASSOCIATION

The University of Arizona
Carnegie-Mellon University
Case Western Reserve University
The University of Chicago
University of Cincinnati
Illinois Institute of Technology
University of Illinois
Indiana University
Iowa State University
The University of Iowa
Kansas State University
The University of Kansas
Loyola University
Marquette University
Michigan State University
The University of Michigan
University of Minnesota
University of Missouri
Northwestern University
University of Notre Dame
The Ohio State University
Ohio University
The Pennsylvania State University
Purdue University
Saint Louis University
Southern Illinois University
University of Texas
Washington University
Wayne State University
The University of Wisconsin

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

Printed in the United States of America
Available from
Clearinghouse for Federal Scientific and Technical Information
National Bureau of Standards, U. S. Department of Commerce
Springfield, Virginia 22151
Price: Printed Copy $3.00; Microfiche $0.65

ANL-7495
Mathematics and Computers

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

SPECIAL-PURPOSE LANGUAGE FOR LEAST-SQUARES FITS

by

Marian Gabriel

Applied Mathematics Division

September 1968

TABLE OF CONTENTS

	Page
I. INTRODUCTION	5
II. FORM OF THE LANGUAGE	5
III. INFORMATION SUPPLIED BY THE USER	6
IV. PROGRAMMING THE PROCESSOR	7
V. ORGANIZATION OF THE PROGRAM	7
VI. RESULTS	8
APPENDIXES	
A. Description of the Input	9
B. Notes on the FORTRAN Language	20
C. Programming Notes and Flow Charts	23
D. Data Sets Used	48
E. Sections from the Report on the Chemical Equation Translator	50
F. Source Listings for All Codes and Cataloged Procedure Listings	52
ACKNOWLEDGMENTS	79
REFERENCES	79

LIST OF FIGURES

No.	Title	Page
1.	Flow Chart 1, for CLASS	29
2.	Flow Chart 2, for STORE	33
3.	Flow Chart 3, for BUBBLE	34
4.	Flow Chart 4, List-generating Logic: The EQUIVALENCE Statement	35
5.	Flow Chart 5, Constructing the READ Statement	36
6.	Flow Chart 6, Counting Points in FCN$ When a Data Format Was Specified	39
7.	Flow Chart 7, Generating the Inner Loop in FCN$	40
8.	Flow Chart 8, for MKFMT	44
9.	Flow Chart for ORDSP	51

LIST OF TABLES

No.	Title	Page
I.	Functions Commonly Used in FORTRAN	22

SPECIAL-PURPOSE LANGUAGE FOR LEAST-SQUARES FITS

by

Marian Gabriel

I. INTRODUCTION

This report describes a problem-oriented language developed to deal with the following situation in the Digital Computing Center at Argonne: many of the routine problems coded by the Computing Center require least-squares fits to data. That is, a scientific user has several sets of data that obey an algebraic equation depending on various parameters and relating one variable to the others and to the parameters. He wants to know which values of the parameters give the best fit in the least-squares sense to his data. When such a problem arises, a programmer writes the necessary code to adapt one of several general least-squares fitting procedures to the user's problem. The difficulty arises when the user does not know exactly what function to use. He may originally ask to have his data fitted, for instance, to $y = ae^{cx_1} + be^{x_2}$, where x_1, x_2, and y are observed and a, b, and c are the fitting parameters. After seeing the results, he may decide that $y = ae^{cx_1} + be^{x_2} + dx_1$ would be better. Still later, he may want to try several other forms.

When this project was begun, each time the user wanted a change, he had to return to the programmer who made the original adaptation. The programmer would then change the code appropriately. This system was unsuitable to everyone: to the programmer, because the work is a complex, tedious, largely clerical, uninteresting task; and to the user, because programming was slow, impeding his research, and expensive. What was needed was a simple way of specifying the function so that even a user who had hardly seen a computer could do it himself: in other words, a special-purpose language was needed. This report describes such a language for doing least-squares fits to data.

II. FORM OF THE LANGUAGE

The first problem in defining such a language was to choose a fitting procedure with which it would work. The one chosen was Davidon's method[1] for minimizing a function of several variables. This method was chosen partly because it is the most comprehensive of the fitting procedures commonly used at Argonne, and partly because, as presently organized and coded, the problems of adapting it to be used as a least-squares fitting processor are the most inconvenient.

The Davidon procedure finds a minimum of a given function of several variables. For a least-squares fit, the function to be minimized is the sum of the squared residuals,

$$F = \sum_{i=1}^{n} w_i(f_{obs_i} - f_{calc_i})^2,$$

where n = the number of data points, f_{obs_i} = the dependent variable at the ith observed point, f_{calc_i} = the value of the function to be fitted at the ith point (e.g., if the fitting form is $y = ax + b$, where a and b are parameters, x the independent variable, and the ith observed point is (x_i, f_{obs_i}), then $f_{calc_i} = ax_i + b$), and w_i = the weight of the ith data point, often given by 1/(error).[2] The Davidon program as currently used requires, at least, FORTRAN statements to evaluate the function F and its first partial derivatives with respect to each of the fitting parameters. Statements to read the data and print the final results are also required.

The first partial derivatives are given by the formula

$$\frac{\partial F}{\partial p} = \sum_{i=1}^{n} 2w_i(f_{obs_i} - f_{calc_i})\left(-\frac{\partial f_{calc_i}}{\partial p}\right),$$

where p is a parameter. In the formulas for the function and its derivatives, as in the reading and printing, many parts remain fixed from problem to problem. Once f_{calc} and $\partial f_{calc}/\partial p$ are defined, the same FORTRAN statements can be used to calculate F and $\partial F/\partial p$, no matter what f_{calc} and $\partial f_{calc}/\partial p$ are. Therefore the language was organized as a set of inserts into a fixed, precoded section of FORTRAN program. The entire section of program could then be executed easily with the current version of the Davidon procedure. (In practice, however, some of this version was rewritten to simplify the input.)

III. INFORMATION SUPPLIED BY THE USER

Once the form is chosen, it becomes possible to decide what information the user must supply to process a particular problem. To specify a function to be fitted, the user identifies all his symbols as names of parameters, independent variables, or the dependent variable. He gives the order in which numbers appear in his data points and the formula for his function. In the present version, he must specify the partial derivative of his function with respect to each parameter. A more sophisticated user may insert additional FORTRAN statements at six points in the precoded skeleton. To run a case once the function is defined, the user must give estimates for each parameter along with data points, consisting of values for the dependent and independent variables and possibly weights or errors for

each point. The processor then generates a valid FORTRAN code from the function definition and converts the parameter guesses and other data into acceptable input for the generated FORTRAN program.

IV. PROGRAMMING THE PROCESSOR

It was assumed throughout the programming task that people's time is always more valuable than machine time. Thus, although some care was taken to avoid gross inefficiencies, no special care was taken to make either the processor or the FORTRAN code generated maximally efficient. The 3-7 min required to generate the FORTRAN code, compile it, and execute the Davidon procedure costs much less than even 2 days of a programmer's time.

The processor is divided into three sections: a translator, which generates the required FORTRAN code; a data processor, which rearranges all the numerical data into valid input to FORTRAN programs; and the Davidon procedure.

PL/I was chosen as a programming language for the translator and data processor because it is an extremely convenient language for doing the kind of analysis required to interpret the user's definitions and to generate the proper FORTRAN code.

Because the Davidon procedure was written in FORTRAN, this language was used for the last step.

V. ORGANIZATION OF THE PROGRAM

The translator reads all the cards defining the function, identifies the purpose of each card or group of cards, marks each card image, and rearranges the marked card images into the order in which they are needed by the program-generating section of the translator. It then combines the precoded sections of the required FORTRAN code with statements derived from the function definition to produce the finished code. Finally, the translator passes the parameter guesses and data, together with information about the handling of the data, to the data processor.

The data processor arranges the parameter guesses in an order and format acceptable as input for FORTRAN programs and provides estimates of their standard deviations, if no such estimates were given. If the user wanted special reading techniques, the unaltered card images containing the data are passed to the Davidon procedure. Otherwise, the data points are counted and arranged in a standard form before the information is transmitted. The Davidon procedure is then called directly by the data processor.

(Appendixes A and B contain a more detailed description of the input. Appendixes C-F contain detailed descriptions of the programming.)

To make the code easier to use and harder to alter, it is kept on a direct-access storage device, such as a disk. Two cataloged procedures are provided for easy access to the processor.

VI. RESULTS

The processor has been in limited use for several months. It has saved programmer time and effort, but it is not yet entirely satisfactory for the following reason: Most users want graphs and extra calculations in addition to values of the fitting parameters and other standard information. Since these features must be incorporated by using FORTRAN-coded inserts, a programmer is still needed. Writing the inserts is much easier than writing the FORTRAN to solve the entire fitting problem, but the average user cannot do it himself.

The language can be extended to eliminate some of these problems. A statement can be included to specify plotting, for example. And plans are being made to include a derivative taker which will analyze the user's function and produce correct FORTRAN statements for calculating the partial derivatives of that function with respect to each fitting parameter.

APPENDIX A

Description of the Input

1. General Principles

Input is on cards or card images.

Unless explicit instructions to the contrary are given, input may appear anywhere on a card.

The equal sign (=) is used as a key word. Thus it should not be used except where specifically required.

Names should not begin with the letters I, J, K, L, M, or N or end with the dollar sign ($).

The word END, appearing in columns 1-3, indicates the end of a section of data or of a case.

2. Minimal Input

a. Function Definition

The translator uses the following cards to generate a FORTRAN subroutine FCN$:

1) A title identifying the function. This title must be the first card and is printed every time the program is run.

The following cards may be in any order, but all must appear:

2) The symbol PARAMETERS=, followed by a list of parameter names, each having at most six characters. The names must be separated by commas. The list may occupy more than one card, but no name should appear more than once in the list.

3) The symbol IV=, followed by a list of the names of the independent variables. These names may have at most six characters each and must be separated by commas. The list may appear on more than one card, but no name should appear more than once in the list.

4) The symbol DV=, followed by the name of the dependent variable.

5) The symbol DATA=, followed by a list of independent and dependent variable names in the order in which they appear in the data to be fitted. If weights are given, the word WEIGHT appears in the list. If error estimates are given, the word ERROR appears in the list.

6) The formula for the dependent variable. If DVNAME represents the name of the dependent variable, this appears as DVNAME= any valid FORTRAN formula. It may occupy more than one card; but if it does, no continuation marks of any kind should appear. The formula may contain the names of the independent variables and parameters. If a formula for the partial derivative of this function with respect to a parameter named PNAME appeared before this formula in the input, DPNAME, denoting this derivative, may also be used. (See Appendix B for further notes on FORTRAN.)

7) For each parameter, the partial derivative of the function with respect to that parameter. If PNAME is the name of the parameter, this derivative appears as DPNAME= any valid FORTRAN formula, which may contain names of parameters and independent variables. The DVNAME and the DPNAME's may also be used if the appropriate formulas have already been given. The formula may occupy more than one card, but each derivative should start on a new card.

8) An end-of-section card, containing the word END in columns 1-3 and blanks in all other columns. This card must follow the function-definition cards.

b. Data for Cases

1) The first card must be a title card to identify the case. It is printed with the output for each case.

2) For the first case, an estimated value and possibly a standard deviation for each parameter. For cases following the first, only those values that differ from the first case must appear. Each parameter estimate must appear on one card.

One of the following forms is used:

a) If no standard deviation is specified:

PNAME=value

In this case, a standard deviation of 0.1 of the value is assumed.

b) PNAME=value; standard deviation

c) PNAME=value; CONSTANT

In this case, the value is held constant for the duration of the case.

3) After the estimates for the parameters have been given, an END card (with END appearing in columns 1-3) must appear.

4) The observed data to be fitted then follow. For each point, numbers must be in the order given in the statement DATA=, (as specified in 2.a.5) above. Each number must be separated from other numbers by at least one blank and may not contain embedded blanks.

5) Two END cards must follow the last data card. If more cases are desired, more sets of cards (starting with the case title in this section) are included.

c. Job Control Cards Needed (to run on System 360 50-75 under ASP).

1) To compile the FORTRAN subroutine FCN$ and run one or more cases, the following job control cards are needed:

```
/*SETUP DDNAME=PRGG,DEVICE=2314,ID=ODDJOB
// EXEC VMMCLG
//GO.SYSIN DD *
    Data as described above.
/*
```

Punched cards containing the names of the parameters and other information are always produced in this process.

2) To compile and run as above and to get an object deck of the FORTRAN subroutine, the following job control cards are needed:

```
/*SETUP DDNAME=PRGG,DEVICE=2314,ID=ODDJOB
// EXEC VMMCLG,PARM.FTH='DECK'
//GO.SYSIN DD *
    Data cards as described in a and b above.
/*
```

The structure of the punched deck is as follows:

a) Header card.

b) Two to five cards always obtained from the special purpose compiler when a FORTRAN program is generated.

c) Object deck for the generated FORTRAN subroutine FCN$. This part of the punched output contains the characters FCN$ in columns 73-76 and a four-digit sequence number, starting with 0000, in columns 77-80.

d) Header card.

Only the object deck is valid input to the linkage editor. Occasionally, because of a human or operating-system error, the extra cards described under b) are not returned with the deck. If this happens, the simplest thing to do is to run the program again without the PARM.FTH='DECK' option, i.e., as in Part 1) above.

3) To execute the program using the object deck obtained in Part c), the following cards are needed:

```
/*SETUP DDNAME=LIB,DEVICE=2314,ID=ODDJOB
// EXEC VMMLG
//EDT.SYSIN DD *
    Object deck from Part 2) above.
/*
//GO.SYSIN DD *
```

a) The two to five cards obtained when the FORTRAN program was generated.

b) Case definition data given under Part b, "Data for Cases" above.

```
/*
```

3. Sample Case Using Minimal Input

Suppose it is desired to find the values of a and b that give the best least-squares fit of y = ax + b to the following sets of data:

Case 1	y	x	Error in y
	-1.0	0.0	0.02
	-0.8	0.2	0.03
	-0.25	1.0	0.01
	0.4	2.0	0.04
	1.2	0.3	0.03
	1.9	4.0	0.02
	2.6	5.0	0.01

Case 2	y	x	Error in y
	-1.0	0.0	0.01
	-1.4	1.0	0.01
	-1.8	2.0	0.02
	-2.2	3.0	0.03
	-2.7	4.0	0.06

The entire input might look like this (each line is a card):

```
JOB card
Accounting card
/* SETUP DDNAME=PRGG,DEVICE=2314,ID=ODDJOB
// EXEC VMMCLG
// GO.SYSIN DD *
LINEAR FIT TO SAMPLE DATA
PARAMETERS=A,B
IV=X
DV=Y
DATA=Y,X,ERROR
Y=A*X+B
DA=X
DB=1.
END
CASE 1
A=.833
B=-1.0
END
    -1.0      0.0      0.02
    -0.8      0.2      0.03
    -0.25     1.0      0.01
     0.4      2.0      0.04
     1.2      0.3      0.03
     1.9      4.0      0.02
     2.6      5.0      0.01
END
END
```

```
CASE 2
A=-.4
END
    -1.0      0.0      0.01
    -1.4      1.0      0.01
    -1.8      2.0      0.02
    -2.2      3.0      0.03
    -2.7      4.0      0.06
END
END
/*
```

After an object deck for the FORTRAN subroutine has been obtained, the following input deck is used:

```
JOB card
Accounting card
/*SETUP DDNAME=LIB,DEVICE=2314,ID=ODDJOB
// EXEC VMMLG
//EDT.SYSIN DD*
    Object deck as described under c.2)c) above.
/*
//GO.SYSIN DD *
    Extra cards generated by a run above, described
    under c.2)b) above.
CASE 1
A=.833
B=-1.0
END
Data as before, ending with
END
END
/*
```

4. Expanded Input

The following statements may appear in the function definition section of the input. Except for the DATA=INSERT option, which replaces the DATA=list statement, these statements appear in addition to the function definition cards described under "Minimal Input."

a. DATA FORMAT Option

A user may, if he wishes, give his own FORTRAN format for reading the observed data. The statement for specifying the format is

DATA FORMAT = any valid FORTRAN format.

The format may be continued on more than one card. The equal sign is optional here; the compiler supplies enclosing parentheses for the format. This statement causes information for each data point to be read according to the given format in the order specified in the statement DATA=list. WARNING. This option is extremely slow. If at all possible, the format specification should be omitted or a DATA=INSERT option used. (See Part c below.)

b. FORTRAN Inserts

A programmer may insert FORTRAN statements at six "break points" in the subprogram FCN$.

1) The form of the inserts is:

Column	Description
1	An integer from 0 to 5 indicating the placement of the insert. ("C" for comment may also be included if it immediately follows a numbered insert.)
2-80	Valid FORTRAN statement. To avoid duplicate statement numbers, user's statement numbers should be between 100 and 999. User's names known to the program are the names of the dependent variable, independent variables, and parameters. D followed by a parameter name is the derivative of the user's function with respect to that parameter. All other names should be defined by the user. No name should end with $.

When the card image is inserted, the integer in column 1 is replaced by a blank. The rest of the card image is not changed in any way.

2) The six inserts are:

Insert Level	Placement and Contents
0	These cards are placed between a DIMENSION and a DATA statement and therefore must contain similar nonexecutable statements.
1	These cards are placed immediately after the reading of the fitting data and are executed only once for each case. Extra data may be read here. If extra data are present, they must appear immediately before the last END statement.
2	This set of statements is executed once each time FCN$ is called. It precedes the loop defining the sum of the squared residuals and its first partial derivatives.
3	These statements are internal to the loop defining the sum of the squared residuals. Their exact placement depends partly on the order, relative to the function definition and derivative statements, in which they appear in the input. Thus, 3 P=EXP (Z*5.) DVNAME=P+2 DZ=5.*P 3 Q=5.+6.*A is compiled: P=EXP (Z*5.) Y$=P+2. (for dependent variable definition) GGG$(1)=5.*P (if Z is the first parameter) Q=5.+6.*A In any case, however, cards for Insert 3 appear before the sum of the squared residuals and the derivatives of the sum are defined. The statements DVNAME= and DPNAME= may be omitted from the function definition if the value of the function and its partial derivatives are given with Insert 3 cards.
4	These inserts appear after all calculations and printing for a case are completed. The statements are executed once for each case.
5	These statements are the last to appear in the generated program (except for RETURN and END and also some formats). They are executed once after all the cases have been completed.

3) Inserts of a particular level appear in the code in the same order they appeared in the input stream. For example, suppose the following cards appear in the input:

```
1      LEVEL 1, CARD 1
2      LEVEL 2, CARD 1
1      LEVEL 1, CARD 2
3      LEVEL 3, CARD 1
3      LEVEL 3, CARD 2
2      LEVEL 2, CARD 2
```

These appear as follows:

Block of code preceding place for Insert 1

```
LEVEL 1, CARD 1
LEVEL 1, CARD 2
```

Code between place for Insert 1 and place for Insert 2

```
LEVEL 2, CARD 1
LEVEL 2, CARD 2
```

Code between place for Insert 2 and place for Insert 3

```
LEVEL 3, CARD 1
LEVEL 3, CARD 2
  rest of code
```

Insert cards may appear anywhere in the function definition section after the title and before the END card.

4) WARNING. Because of the way the job steps are organized and the data processed, the step name of the job step in which the Davidon procedure is actually executed is different in the procedures VMMCLG and VMMLG. Thus, DD cards referring to extra data sets used on these inserts will have ddnames qualified by different stepnames in the two procedures. For example, suppose plotting is desired. The deck structures, including a DD statement for a Calcomp tape, are as follows:

a) To generate a FORTRAN code and execute:

```
/*SETUP DDNAME=PRGG,DEVICE=2314,ID=ODDJOB
/*SETUP DDNAME=PLOTTAPE,DEVICE=2400-7,ID=(,,SAVE,NL)
```

```
// EXEC VMMCLG
//GO.SYSIN DD *
    Data to generate FORTRAN code and execute.
/*
//RUN.PLOTTAPE DD DSNAME=PLOT780,DISP=(NEW,KEEP),   C
// UNIT=(2400-2,,DEFER),LABEL=(,NL)
/*
```

b) To execute, using an object deck for the generated FORTRAN code:

```
/*SETUP DDNAME=LIB,DEVICE=2314,ID=ODDJOB
/*SETUP DDNAME=PLOTTAPE,DEVICE=2400-7,ID=(,,SAVE,NL)
// EXEC VMMLG
//EDT.SYSIN DD *
    Object deck.
/*
//GO.PLOTTAPE DD DSNAME=PLOT780,DISP=(NEW,KEEP),   C
// UNIT=(2400-2,,DEFER),LABEL=(,NL)
//GO.SYSIN DD *
    Data as appropriate for executing the program using an
    object deck.
/*
```

Thus, for a compilation the stepname is RUN, and cards referring to it follow the data. For execution from an object module, the stepname is GO, and cards referring to it immediately precede the data.

c. DATA=INSERT Option

A programmer may use Insert 1 cards to write his own READ statements for the observed data by replacing the statement DATA=list with the statement DATA=INSERT. DATA=INSERT causes the translator to omit the usual FORTRAN statements for reading the observed data. Some variable names that may be useful in programming appropriate READ statements are:

IN$	Input "tape number"--set to 5
ND$	Number of data points--set to 1000
YD$(I)	Value of the dependent variable for the Ith data point. This array is dimensioned by 1000.
XD$(J,I)	Value of the Jth independent variable for the Ith data point. J is determined by the variable's position in the list following IV=. This array is dimensioned XD$(20,1000).
W$(I)	Weight of the Ith data point. This array is dimensioned W$(1000).

When the DATA=INSERT option is used, only one END card follows the data for each case.

APPENDIX B

Notes on the FORTRAN Language

1. Form of FORTRAN Statements

These notes are not intended to be a complete description of the FORTRAN language but only to help a user prepare minimal input. For further information, the user is referred to the following IBM OS/360 publications:

Form C28-6515-4	Fortran IV Language
Form C28-6602	Fortran IV (H) Programmer's Guide

FORTRAN statements for the minimal input will consist of combinations of constants and the names of the dependent variables, independent variables, and parameters.

A constant consists of one through seven decimal digits. It must include a decimal point and may be followed by an exponent. The exponent is indicated by writing E, followed by a sign, followed by two digits. A constant must be zero or else must have absolute value from 16^{-63} through 16^{23}. It must not contain embedded commas.

Examples:

```
  0.0
 -5.3
  2.
562.E+03 (This, of course, is 562. x 10^3.)
-49.2E-52
```

The following operators may be used:

Operator	Definition
**	Exponentiation
*	Multiplication
/	Division
+	Addition
-	Subtraction

The following rules apply to combinations of variables, constants and operators:

a. All operations must be explicitly stated. Writing AB will not result in the product of A and B. This product is calculated only when A*B is written.

b. No two operators may appear in sequence. Thus, A*-B is not a valid combination. To obtain the product of A and -B, A*(-B) must be written.

c. If there are no parentheses or if the expression or subexpression is enclosed entirely in a pair of parentheses, operations are performed in the following order:

1) Evaluation of functions (see Part 2 of this appendix)
2) Exponentiation (**)
3) Multiplication and division
4) Addition and subtraction

Except for exponentiation, operations of the same level are performed from left to right. Thus A/B*C is evaluated by first dividing A by B, then multiplying the result by C.

Example: E+C*A/B**D is evaluated as follows:

1) Calculate B**D (call this X:E+C*A/X).
2) Calculate C*A (call this Y:E+Y/X).
3) Calculate Y/X (call this Z:E+Z).
4) Add E to Z.

Exponentiation is done from right to left. Thus A**B**C is evaluated as follows:

1) Calculate B**C (call this result X).
2) Calculate A**X.

d. Parentheses may be used to specify the order in which operations occur. An expression within parentheses is evaluated before the result is used.

2. Commonly Used Functions

The functions in Table I are supplied by IBM and may be of interest to the users of this translator. All functions are written as follows:

If FNAME is the function name and ARNAME is the name of its argument, the function is written FNAME (ARNAME). If there is more than one argument, the arguments are separated by commas. In the table, the argument is designated by "arg." If there are two arguments, they are designated arg_1 and arg_2, and the function is designated FNAME(arg_1, arg_2). Function names must appear exactly as given in the column "Function Name."

TABLE I. Function Commonly Used in FORTRAN

Function	Function Name	Definition	Number of Arguments
Exponential	DEXP	e^{arg}	1
Natural logarithm	DLOG	ln(arg)	1
Common logarithm	DLOG10	$\log_{10}(arg)$	1
Arcsine	DARSIN	arcsin(arg)	1
Arccosine	DARCOS	arccos(arg)	1
Arctangent	DATAN	arctan(arg)	1
	DATAN2	$\arctan(arg_1/arg_2)$	2
Trigonometric sine (Argument in radians)	DSIN	sin(arg)	1
Trigonometric cosine (Argument in radians)	DCOS	cos(arg)	1
Trigonometric tangent (Argument in radians)	DTAN	tan(arg)	1
Trigonometric cotangent (Argument in radians)	DCOTAN	cotan(arg)	1
Square root	DSQRT	$(arg)^{1/2}$	1
Hyperbolic tangent	DTANH	tanh(arg)	1
Largest value	DMAX1	$\max(arg_1, arg_2 \ldots)$	≥ 2
Smallest value	DMIN1	$\min(arg_1, arg_2, \ldots)$	≥ 2
Absolute value	DABS	$\lvert(arg)\rvert$	1
Hyperbolic sine	DSINH	sinh(arg)	1
Hyperbolic cosine	DCOSH	cosh(arg)	1

APPENDIX C

Programming Notes and Flow Charts

The processor consists of three main programs: VMMTR, DATAPR, and DVDN, of which the first two are in PL/I and the third is in FORTRAN IV.

1. VMMTR

VMMTR is the FORTRAN-writing code. After defining various arrays and reading the title card, VMMTR calls three routines to read and sort all the function definition cards. (The sorting was added after the code was written, when it was discovered that neither I nor any of the users could remember the order of input required by the translator.)

a. Routines Used

(1) Sorting Routines. The sorting routines are CLASS, STORE, and BUBBLE. Their functions are as follows:

(a) CLASS. This routine reads and counts the function definition statements. By using keywords (such as DATA, PARAMETERS, and IV) and insert numbers, it classifies each card, determining whether it is the list of parameter names, the list of independent variable names, or other such information. If the card contains an "=" sign but the string on the left side (with blanks omitted) is not a key word, the card is assumed to contain a formula for the dependent variable or its partial derivative with respect to one of the parameters. If it does not contain an "=" sign and is neither an insert card nor a DATA FORMAT card, it is assumed to be a continuation of the preceding card. After a card has been classified, the card image and a tag are stored in an array of character strings each 81 characters long, with the tag as the first character and the card image as the rest. The tags are arranged in collating sequence; that is, the tag for the parameter list, for instance, is represented in the computer as a smaller number than any of the other tags. Storage for this array is allocated 100 strings at a time, and the tagged card images are stored as a "push-down stack." Thus at the end of CLASS the tagged card images are arranged in blocks of 100, with the first card read in the 100th location of the block farthest down the stack. (See Flow Chart 1 in Part 4 of this appendix.)

(b) STORE. This routine allocates enough storage for all the card images in a new array, called CLST, and stores the images with the first one read in location 1. It also frees all the storage allocated in CLASS. (See Flow Chart 2 in Part 4 of this appendix.)

(c) BUBBLE. This routine uses a "bubble-through" sorting technique to arrange the tagged card images in the proper order for use by the rest of the code. The technique works as follows: Starting with the last card, it compares the tag of each card with the tag of the card immediately preceding it. Whenever a card has a tag that is less than the tag of its predecessor, the two cards are swapped so that the card with the smaller tag appears before the one with the larger tag. The array is scanned repeatedly until a scan occurs in which no swaps are made. (See Flow Chart 3.)

(2) Routines from Chemical Equation Translator. A number of the string analysis and output routines were first written for the Chemical Equation Translator, 08T7019, and are documented in the report on that program.[3] They are RDCD, PNCH, CAT, and ORDER. (ORDSP in 08T7019 was renamed ORDER.) (See also Appendix E.)

(3) Changes to Built-in Routine SUBSTR. While VMMTR was being developed, the implementation of the built-in function SUBSTR was changed so that some of the calls were no longer correct. It seemed easier to change the definition of SUBSTR than to find all the incorrect calls. Therefore a special SUBSTR was written to correct the calls and then call the built-in SUBSTR.

b. Body of VMMTR

The rest of VMMTR is concerned with generating the FORTRAN subroutine FCN$ as a set of card images and writing them on a file PROG. A fixed skeleton FCN$ contains seven sections which are stored on a disk in a file RDFILE and written on PROG by procedures BLOCK1 through BLOCK7. $ is used as a suffix on all the variable names in the skeleton FCN$ so that the user's names will be different from those in the fixed part of FCN$.

After the first fixed section of FCN$ is placed on PROG, the lists of names are analyzed by using ORDER. The first variable section of FCN$ is then constructed. In addition to an Insert 0 if present, the section consists of a DATA statement, which sets up an array of parameter names for printing, and an EQUIVALENCE statement, which assigns the user's names for independent variables, the dependent variable, parameters, and derivatives to the same storage locations as the corresponding names in the skeleton FCN$. The technique for generating these statements is used throughout VMMTR to generate lists. Usually the identifying marks (such as EQUIVALENCE) are stored in a character string CARD, and the first item in the list is added. Then for each succeeding item in the list, the procedure CAT is called to add the text ",item" to the characters in CARD and to ensure that all FORTRAN text appears in characters 1-72 of CARD, with continuation marks as needed. Thus loops for generating lists often appear with indices from 2 to the number of items. (See Flow Chart 4.)

After the nonexecutable statements, a FORMAT is constructed to print the general title, and BLOCK2 is called to put the next fixed section of FCN$ on PROG. The next variable section concerns the READ statement for the data to be fitted.

The list associated with the statement DATA= is analyzed by using ORDER, and the names are stored in an array TNAMES. If the statement is DATA=INSERT (i.e., if TNAMES(1)=INSERT) the construction is omitted. Otherwise, the next card image is checked to see whether it is a DATA FORMAT specification. If not, a READ statement is constructed, using the list-generating ideas described above. (See Flow Chart 5.)

The format chosen is 6E12.0, and the data processor will count the number of data points and rearrange the data in that format. If a DATA FORMAT card is present, counting points becomes much more complicated. The difficulty occurs because there is no easy way for the data processor to tell where one number ends and the next begins. If no format is specified, the data processer uses a GET LIST, which treats blanks as separating characters, to read all the numbers associated with one point. But here there may be blanks embedded in a number or no blanks anywhere. Short of programming a section of the PL/I code to analyze the FORTRAN format, there seemed to be no way out except to count the points in the FORTRAN code. So this is done by reading each card with a format 20A4, and testing to see whether columns 1-3 contain the word END. If they do, the reading is finished. Otherwise, the file containing these cards is backspaced one record, and the data are read using the specified format. (See Flow Chart 6 for the logic of this process.)

Note that this describes a process in the FORTRAN code, not a process for generating something in the FORTRAN code. Since the primary input stream cannot be backspaced, the card images containing these data are kept on a separate unblocked file.

Once the possible format is taken care of, a READ statement is constructed. If errors are to be read, code is generated to convert them to weights, while saving the original values. If there is an Insert 1, it appears after the READ statement. BLOCK3, Insert 2, and BLOCK4 follow in that order.

The next variable section of code occurs in the middle of the inner loop where the sum of the squared residuals and its partial derivatives are calculated. Here there are a number of card images all tagged "Insert 3": the formula for the dependent variable, formulas for its derivative with respect to each parameter, continuations of these cards, actual inserts of Level 3. Each is interpreted and inserted in the code in the order in which it occurs. Inserts are marked 3, 4, or 5 in column 2 of the tagged card image. (That is a column 1 of the card as read.)

If the card is not an insert and an equal sign appears on it, the character string on the left, with blanks removed, is compared with the dependent variable name and with the character D followed by each parameter name. If the string is one of these names, it is rearranged to occupy columns 7-72, continuation marks are added if necessary, and the statement is inserted in the code. If the string is not one of these names, it is an undefined symbol, and processing is terminated. No check is made to determine whether all names have appeared. If no equal sign appears, the card is assumed to be a continuation.

The bit string CATCALL is set "true" whenever CAT is called. This is done because CAT, after checking the length of a string, writes the first 72 characters on PROG if the total length was more than 72 characters. It writes out neither the continuation string nor initial strings of less than 72 characters. Therefore, after CAT has been called, a statement CALL PNCH must be used to put the last card image on PROG. This middle section may be terminated in one of two ways: (1) All the input cards have been analyzed, or (2) there is an Insert 4 or 5. Either occurrence signals the end of the variable section of the inner loop, and the program proceeds to the last section of the translator. (See Flow Chart 7 for the logic of this process.)

In the last section of the code, BLOCK5 and Insert 4 if present are put on PROG. MKFMT (see Flow Chart 8) is called to construct a format for printing a heading for the output, containing the user's names for the various quantities. BLOCK6, Insert 5 if present, and BLOCK7 are then put on PROG. Finally, a procedure DTPR generates card images containing information needed by the data processor: a list of the parameter names, the number of items needed to specify a data point, and numbers set to 0 or 1 depending on whether there is a data format or a DATA=INSERT specification. These card images are put on a file CDFILE and are also punched out. The rest of the data is put on CDFILE after the above card images.

2. DATAPR

DATAPR, the data processor, rearranges the data into a form acceptable to FORTRAN. The first section of the code reads information from VMMTR and the title, and deals with the parameter guesses and standard deviations. For the first case, values of these are stored in arrays GUESSF, for the guesses, and SDF, for the standard deviation. If no standard deviation is given, the value is set at -1.0; if the word CONSTANT appears, the value of the standard deviation is set to zero. (Because PL/I truncates rather than rounding, 0.000001 of each value is added to the value, so that 3. appears as 3.0, not as 2.99999.) The numbers in the arrays are stored in the order in which they appeared on the card PARAMETERS= when FCN$ was generated. After GUESSF and SDF are defined, arrays for the current case GUESS and SD are defined.

For each case, values are the same as for GUESSF and SDF, except when a card containing new guesses is present. Processing of parameter cards is finished when a card not containing an equal sign is found. This card should be blank or contain the work END. It is not interpreted as the first data card. For each I from 1 to the number of parameters, GUESS(I) and SD(I) are put on a file CDFILE in a format 2 E(12,5), X(56). This corresponds to a FORTRAN format of 2E12.5. The statement PUT FILE (CDFILE) EDIT (BL) (X(55),A(1)); allows for the 56 blanks after the last parameter guess, so that the data to be fitted starts on the next "card," not on the last half of the old one.

The data to be fitted are rearranged in one of three ways, depending on whether there is a data format, a DATA=INSERT statement, or neither of these. If there is a data format, the number of points is set to 1000. The cards for the case, through and including the first END, are read and output, without change or interpretation, on an unblocked file XTFILE. Any extra data are put on CDFILE, which becomes logical unit 5 in the FORTRAN step. The second END at the end of the case is omitted.

If reading is done with an insert, the number of data points is again set to 1000. The rest of the cards containing data for the case are read and written, exactly as they are, on CDFILE. For this option, there is only one END card to mark the end of the case. It is not written on CDFILE.

When there is neither a data format nor a DATA=INSERT statement, the number of points is actually counted. NUM, the number of items that define each data point, has been read by the data processor. A GET LIST statement is used to read the numbers from cards, NUM at a time, and convert them to floating-point decimal representation. A counter is increased by 1 each time the GET LIST is executed; 0.000001 of each value is added to the value; and the numbers are put on a temporary file TEMP in the format 6 E(12,5), X(8). The process of reading, converting, and writing continues until the word END is encountered. Attempting to convert this to floating-point decimal representation causes a conversion interrupt, which signals the end of the section. Now the number of points is written on CDFILE, followed by a blank character, in the format F(6,0), X(73), A(1). The card images are read from TEMP and written on CDFILE. Any extra cards, not including any END cards, are now written on CDFILE.

At the end of the data processor, control is transferred directly to the variable metric minimization procedure by the statement CALL DVDN. This FORTRAN code has previously been compiled as a main program, to allow the proper handling of interrupts, with a NAME=DVDN option. All information is passed through peripheral devices. The direct transfer means that the job (with a FCN$ generation) proceeds as follows:

a. Execute VMMTR to generate FCN$.

b. Compile FCN$.

c. Link edit the data processor, the variable metric procedure, and FCN$.

d. Execute DATAPR.

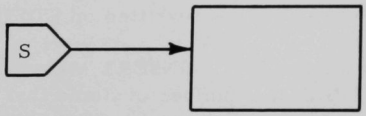
e. Execute the variable metric procedure.

3. DVDN

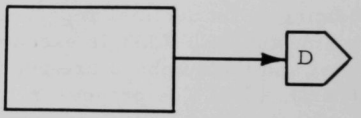
DVDN is the variable metric minimization program described in Ref. 1. Appendix B of this report gives the listing for the version currently used.

4. Flow Charts

Blocks of flow charts (Figs. 1-8) occupying more than one page are numbered on the left-hand side of each page. Where there is more than one block on a line, the blocks are numbered from left to right. Flow of logic between pages is given by an off-page connector symbol:



indicates that control comes from a block numbered S;



indicates flow of control to a block numbered D.

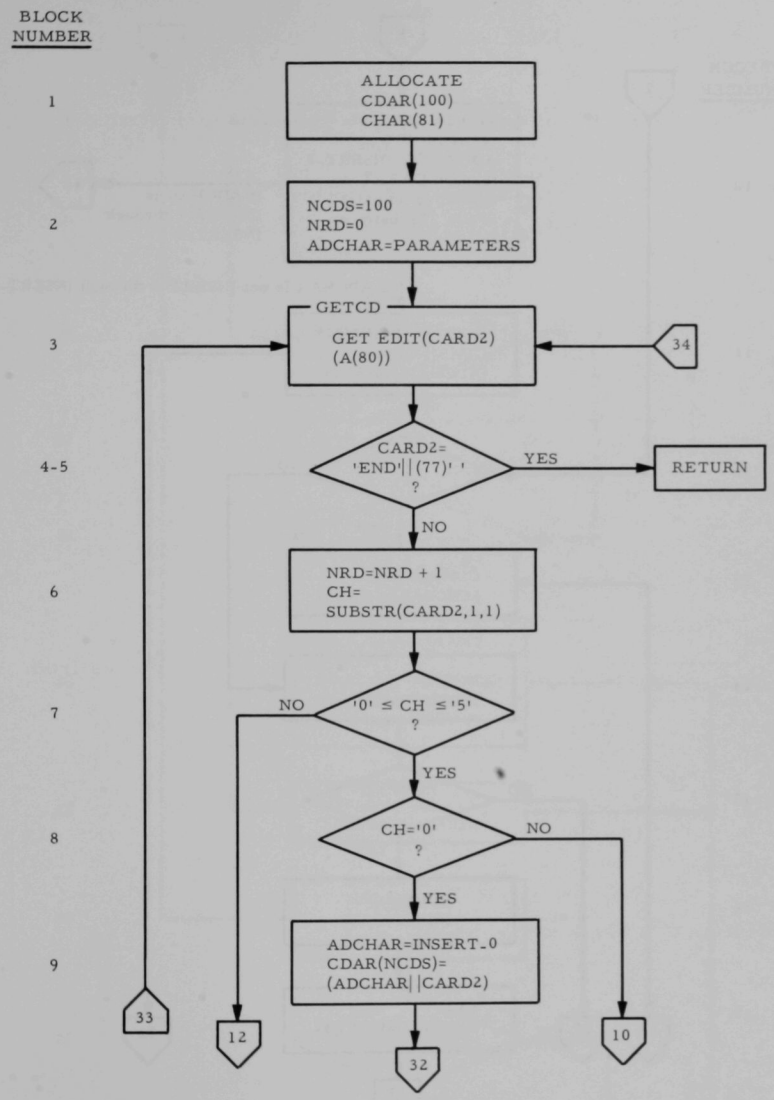


Fig. 1. Flow Chart 1, for CLASS

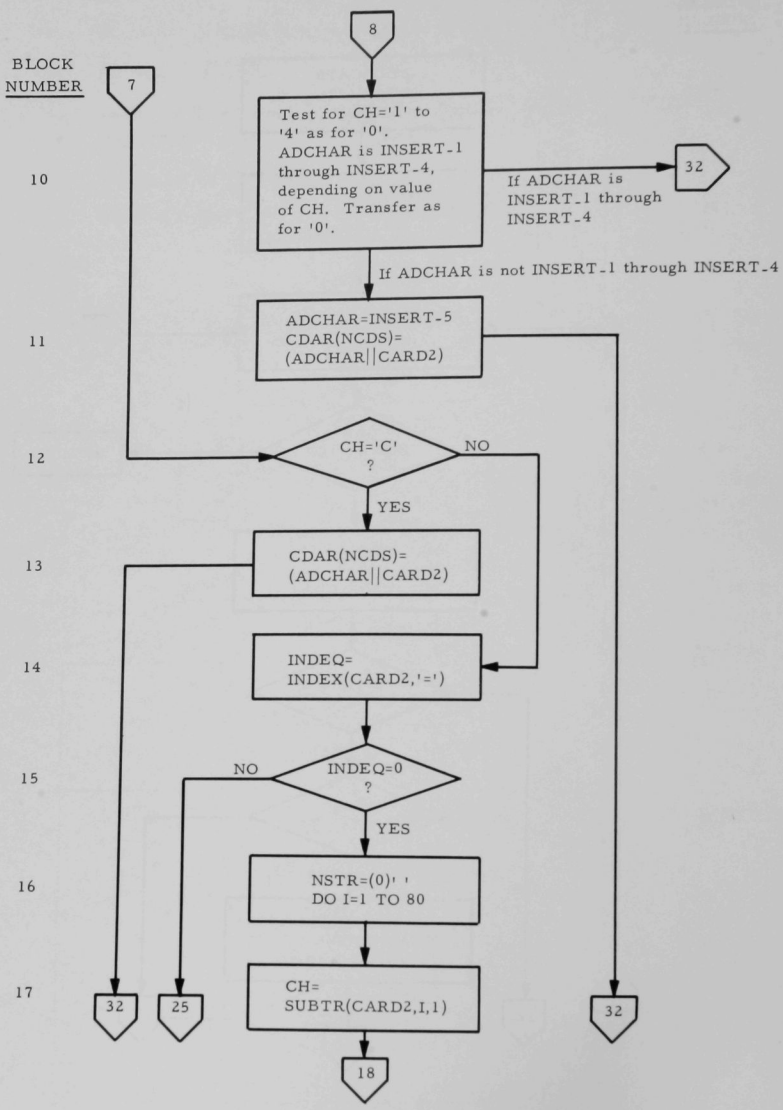

BLOCK NUMBER
7
8
10
Test for CH='1' to '4' as for '0'. ADCHAR is INSERT-1 through INSERT-4, depending on value of CH. Transfer as for '0'.
If ADCHAR is INSERT-1 through INSERT-4
32
If ADCHAR is not INSERT-1 through INSERT-4
11
ADCHAR=INSERT-5
CDAR(NCDS)=
(ADCHAR||CARD2)
12
CH='C'
?
NO
YES
13
CDAR(NCDS)=
(ADCHAR||CARD2)
14
INDEQ=
INDEX(CARD2,'=')
15
INDEQ=0
?
NO
YES
16
NSTR=(0)' '
DO I=1 TO 80
17
CH=
SUBTR(CARD2,I,1)
32
25
32
18

Fig. 1 (Contd.)

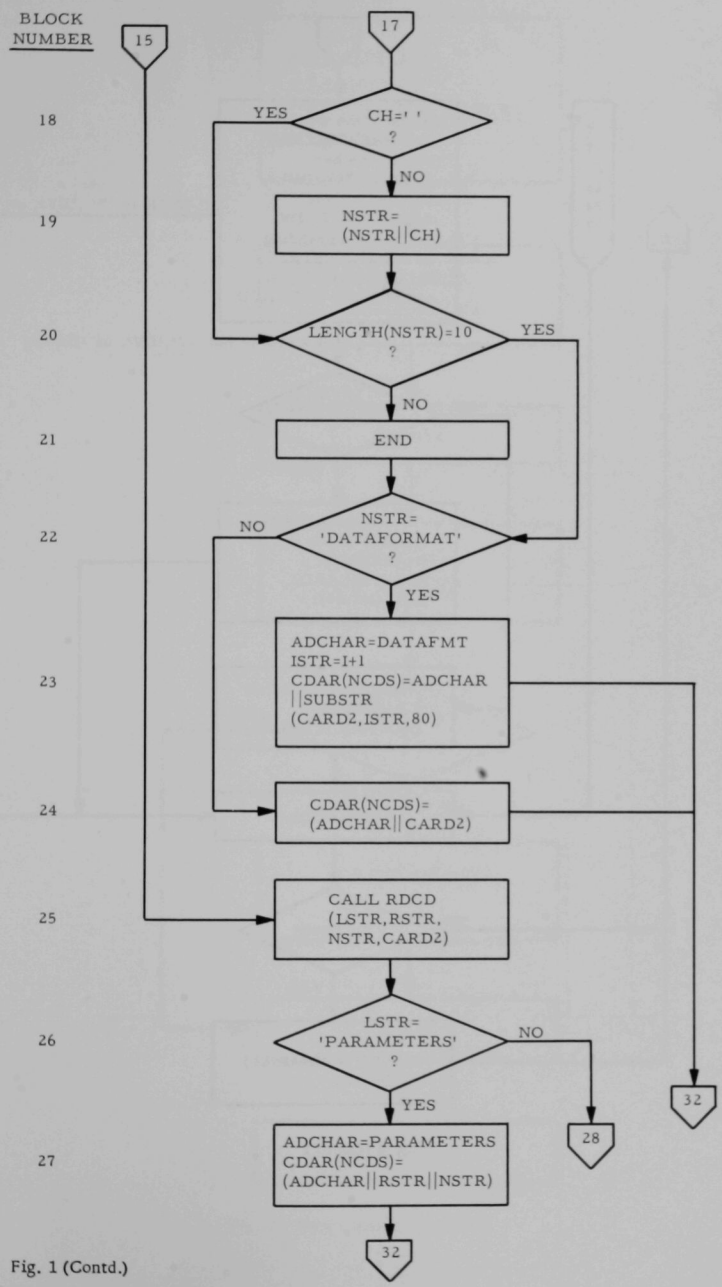

BLOCK NUMBER
15
17
18
CH=' ' ?
YES
NO
19
NSTR= (NSTR||CH)
20
LENGTH(NSTR)=10 ?
YES
NO
21
END
22
NSTR= 'DATAFORMAT' ?
NO
YES
23
ADCHAR=DATAFMT
ISTR=I+1
CDAR(NCDS)=ADCHAR ||SUBSTR (CARD2,ISTR,80)
24
CDAR(NCDS)= (ADCHAR||CARD2)
25
CALL RDCD (LSTR,RSTR, NSTR,CARD2)
26
LSTR= 'PARAMETERS' ?
NO
YES
27
ADCHAR=PARAMETERS
CDAR(NCDS)= (ADCHAR||RSTR||NSTR)
32
28
32

Fig. 1 (Contd.)

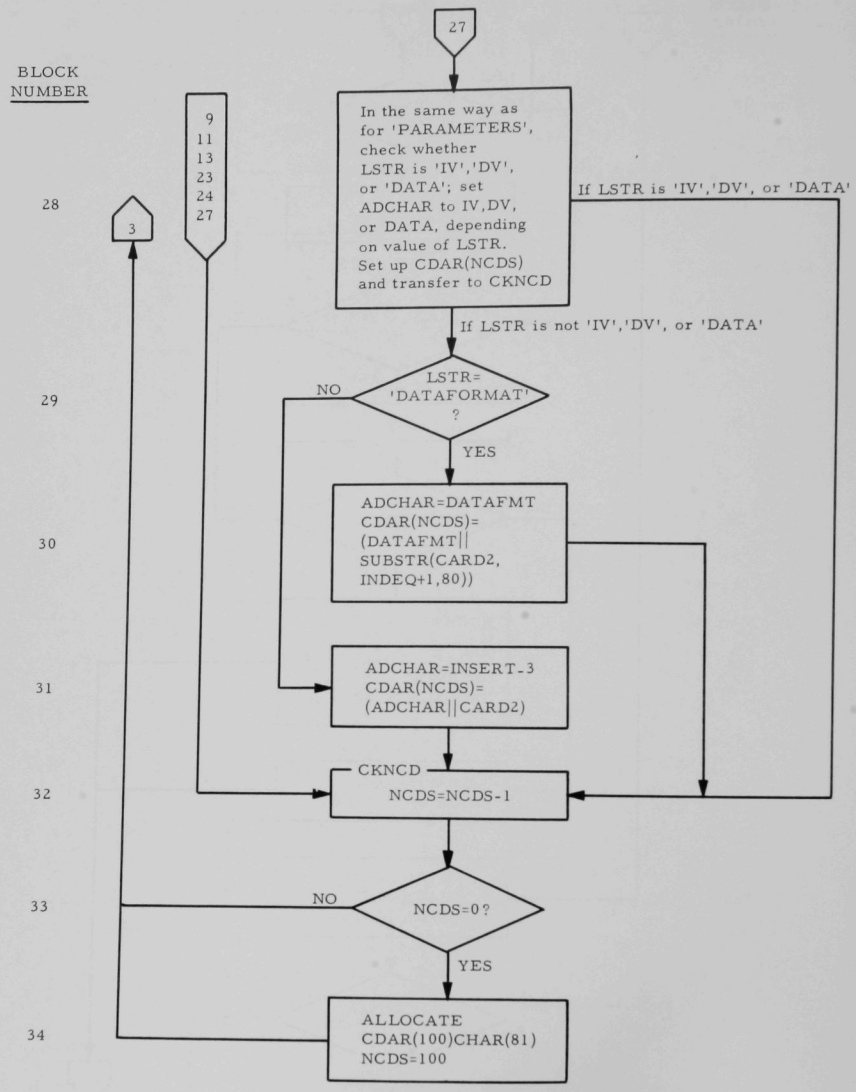

BLOCK NUMBER
27
In the same way as for 'PARAMETERS', check whether LSTR is 'IV','DV', or 'DATA'; set ADCHAR to IV,DV, or DATA, depending on value of LSTR. Set up CDAR(NCDS) and transfer to CKNCD
If LSTR is 'IV','DV', or 'DATA'
28
3
9
11
13
23
24
27
If LSTR is not 'IV','DV', or 'DATA'
29
NO
LSTR= 'DATAFORMAT' ?
YES
30
ADCHAR=DATAFMT
CDAR(NCDS)=
(DATAFMT||
SUBSTR(CARD2,
INDEQ+1,80))
31
ADCHAR=INSERT-3
CDAR(NCDS)=
(ADCHAR||CARD2)
32
CKNCD
NCDS=NCDS-1
33
NO
NCDS=0?
YES
34
ALLOCATE
CDAR(100)CHAR(81)
NCDS=100

Fig. 1 (Contd.)

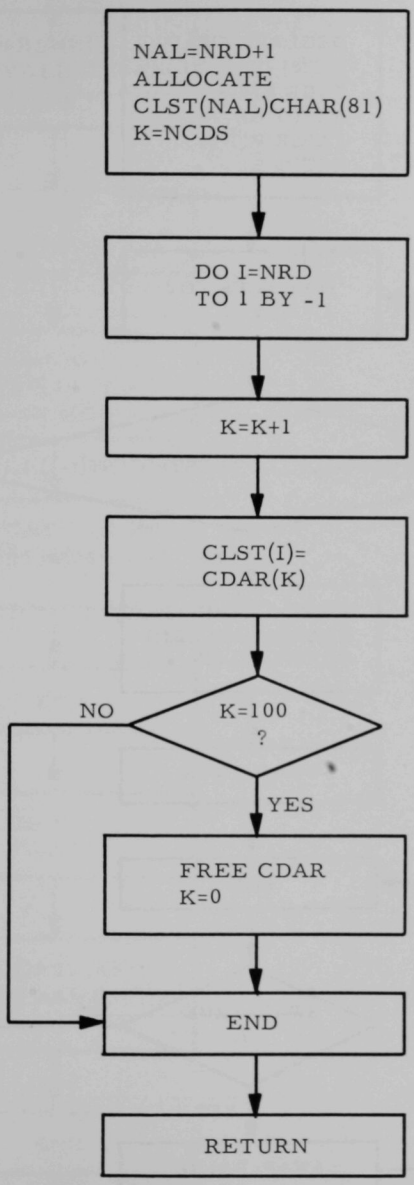


Fig. 2. Flow Chart 2, for STORE

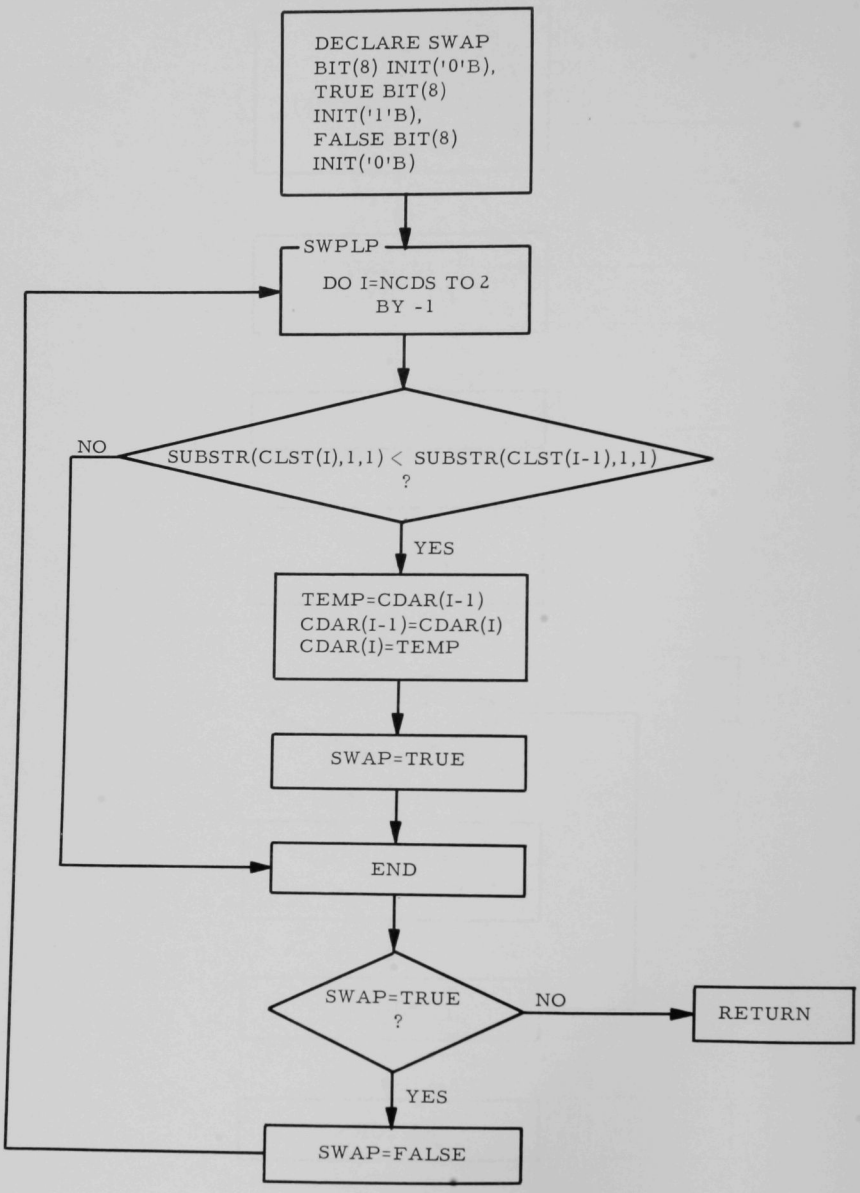


Fig. 3. Flow Chart 3, for BUBBLE

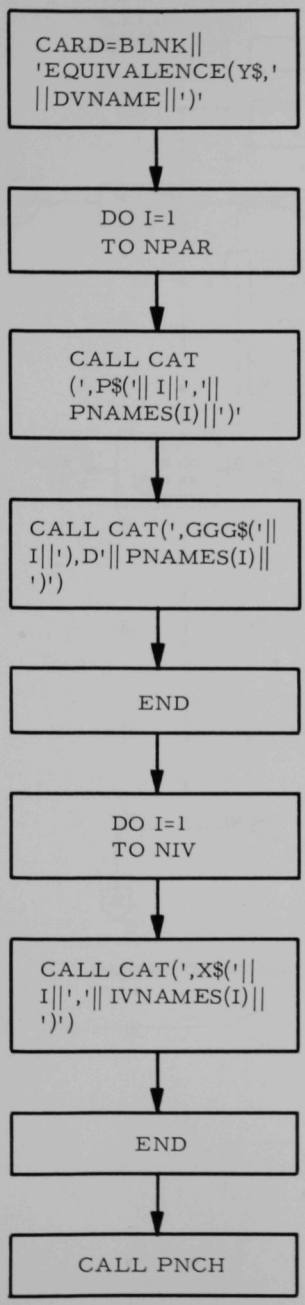


Fig. 4

Flow Chart 4, List-generating Logic:
The EQUIVALENCE Statement

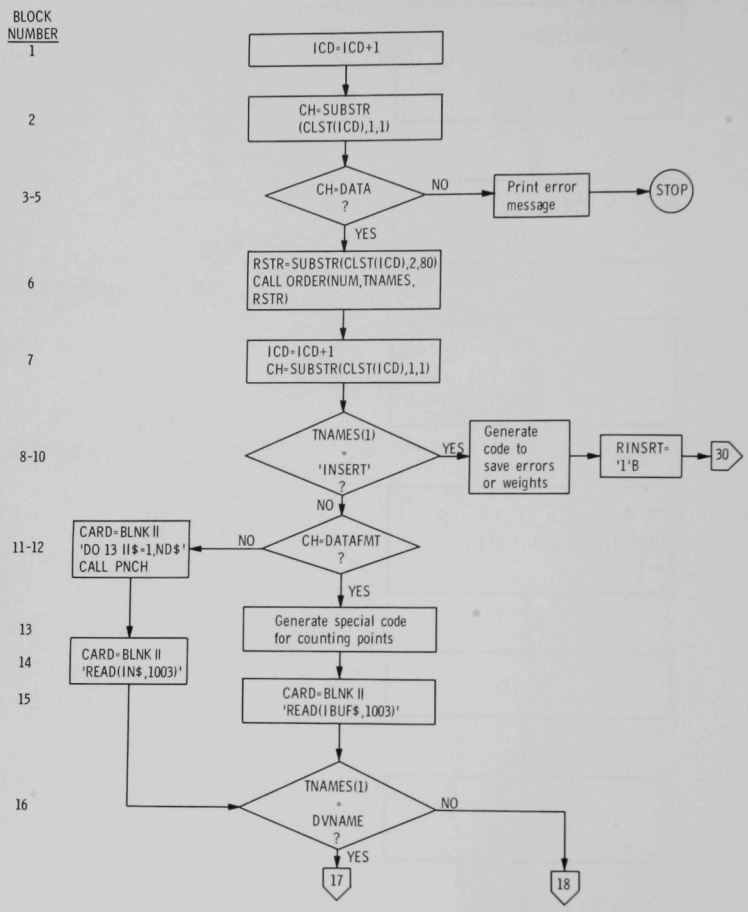


Fig. 5. Flow Chart 5, Constructing the READ Statement

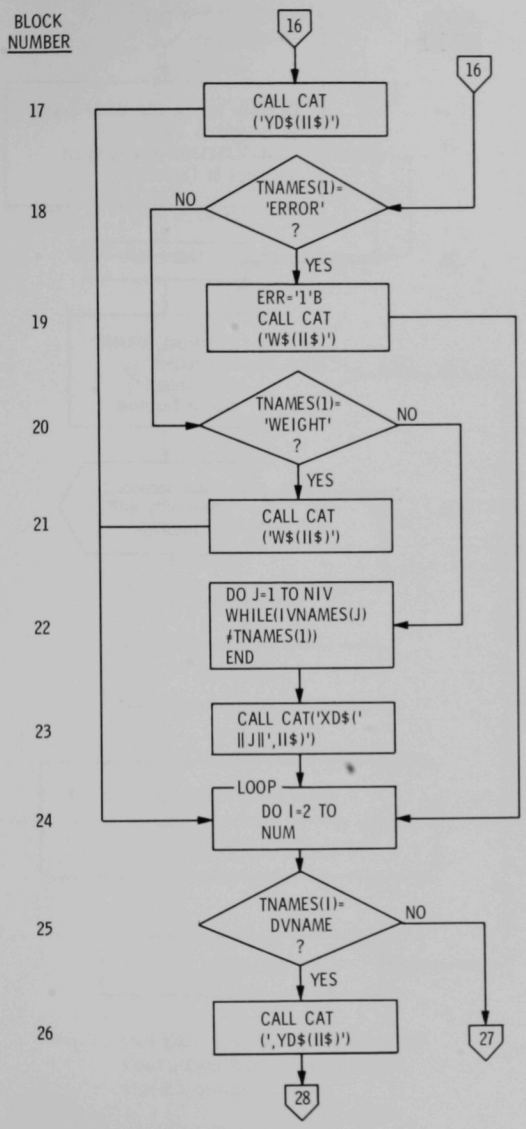

BLOCK NUMBER
16
16
17
CALL CAT ('YD$(II$)')
18
TNAMES(1)= 'ERROR' ?
NO
YES
19
ERR='1'B CALL CAT ('W$(II$)')
20
TNAMES(1)= 'WEIGHT' ?
NO
YES
21
CALL CAT ('W$(II$)')
22
DO J=1 TO NIV WHILE(IVNAMES(J) ≠TNAMES(1)) END
23
CALL CAT('XD$(' ||J||',II$)')
LOOP
24
DO I=2 TO NUM
25
TNAMES(I)= DVNAME ?
NO
YES
26
CALL CAT (',YD$(II$)')
27
28

Fig. 5 (Contd.)

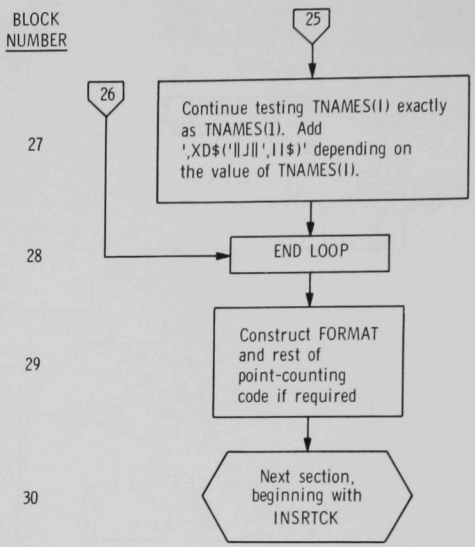

BLOCK NUMBER
25
26
27
Continue testing TNAMES(I) exactly as TNAMES(1). Add ',XD$('||J||',I1$)' depending on the value of TNAMES(I).
28
END LOOP
29
Construct FORMAT and rest of point-counting code if required
30
Next section, beginning with INSRTCK

Fig. 5 (Contd.)

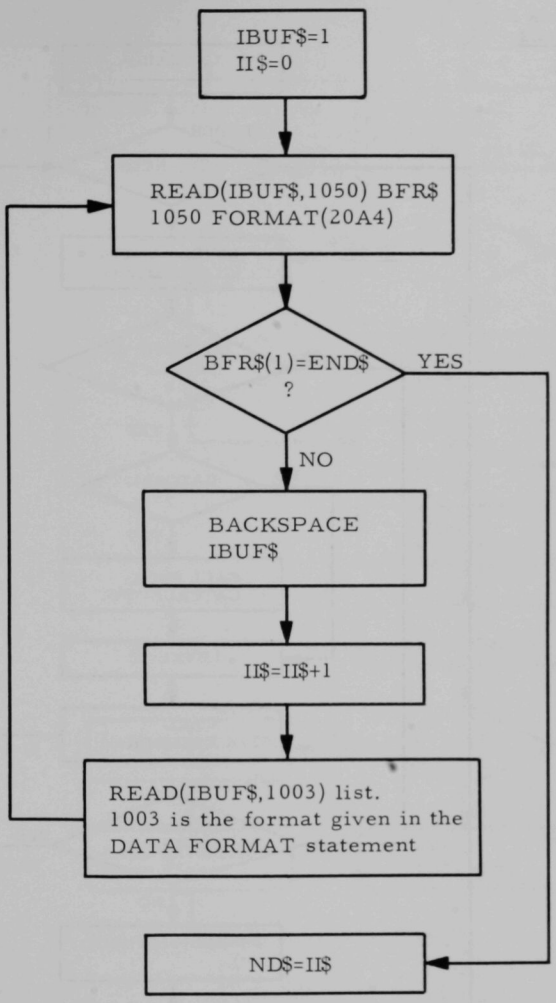


Fig. 6. Flow Chart 6, Counting Points in FCN$ When a Data Format Was Specified. This is a process in FCN$.

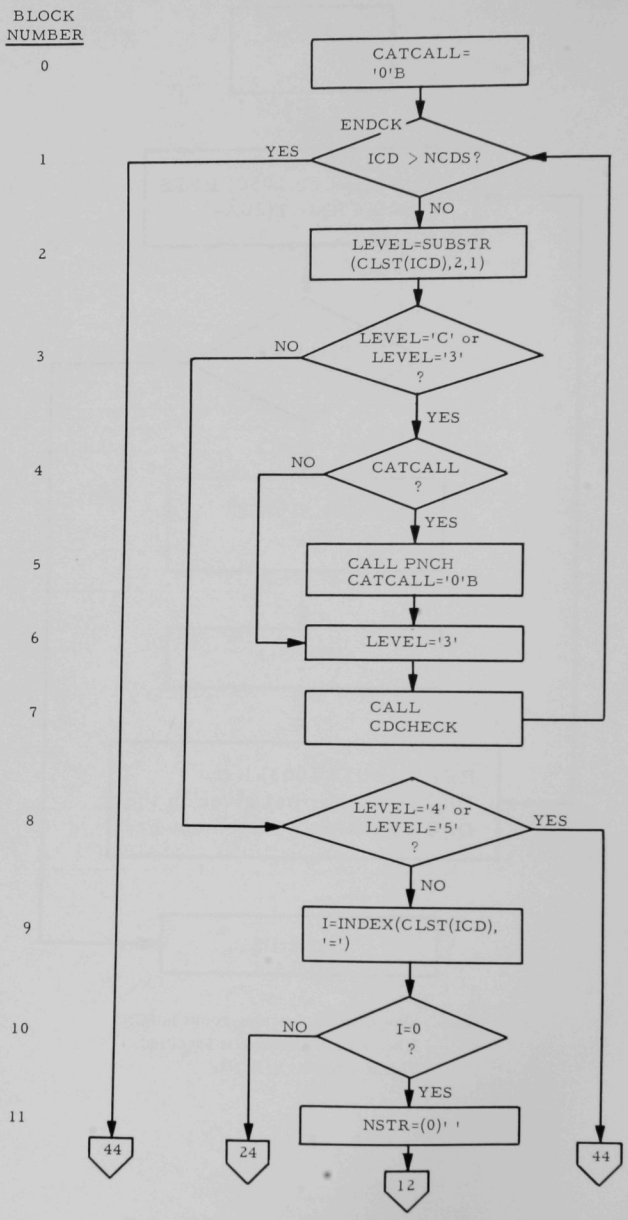


Fig. 7. Flow Chart 7, Generating the Inner Loop in FCN$

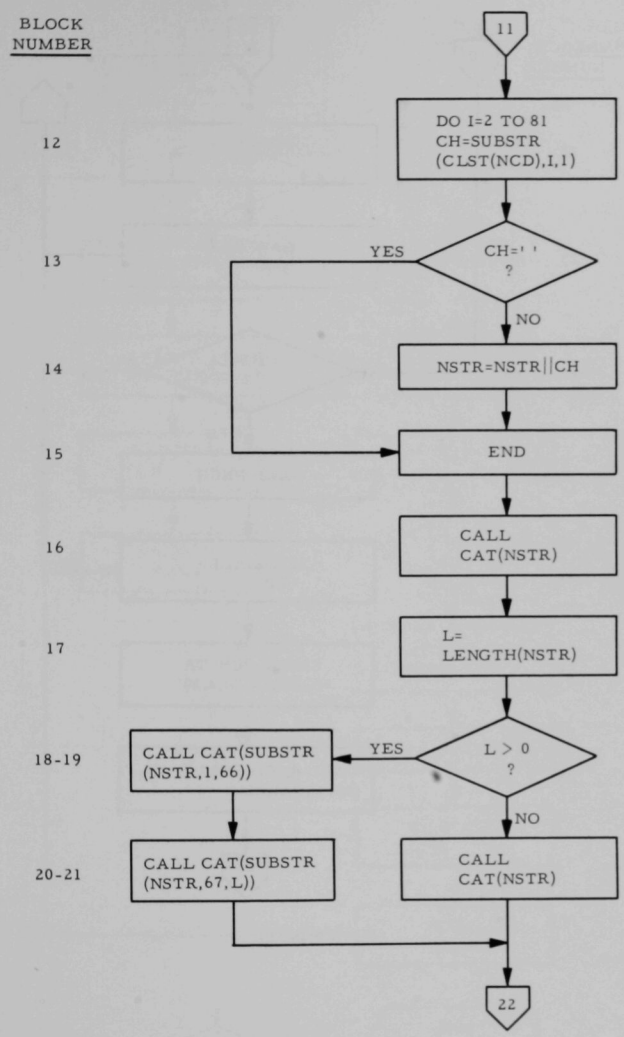

BLOCK NUMBER
11
12
DO I=2 TO 81
CH=SUBSTR
(CLST(NCD),I,1)
13
YES
CH=' '
?
NO
14
NSTR=NSTR||CH
15
END
16
CALL
CAT(NSTR)
17
L=
LENGTH(NSTR)
18-19
CALL CAT(SUBSTR
(NSTR,1,66))
YES
L > 0
?
NO
20-21
CALL CAT(SUBSTR
(NSTR,67,L))
CALL
CAT(NSTR)
22

Fig. 7 (Contd.)

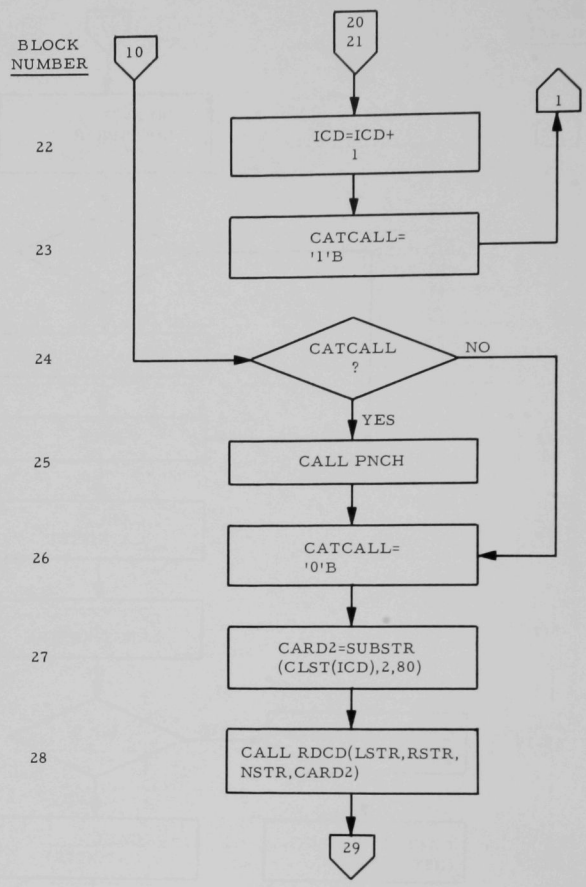

BLOCK NUMBER
10
20
21
1
22
ICD=ICD+
1
23
CATCALL=
'1'B
24
CATCALL
?
NO
YES
25
CALL PNCH
26
CATCALL=
'0'B
27
CARD2=SUBSTR
(CLST(ICD),2,80)
28
CALL RDCD(LSTR,RSTR,
NSTR,CARD2)
29

Fig. 7 (Contd.)

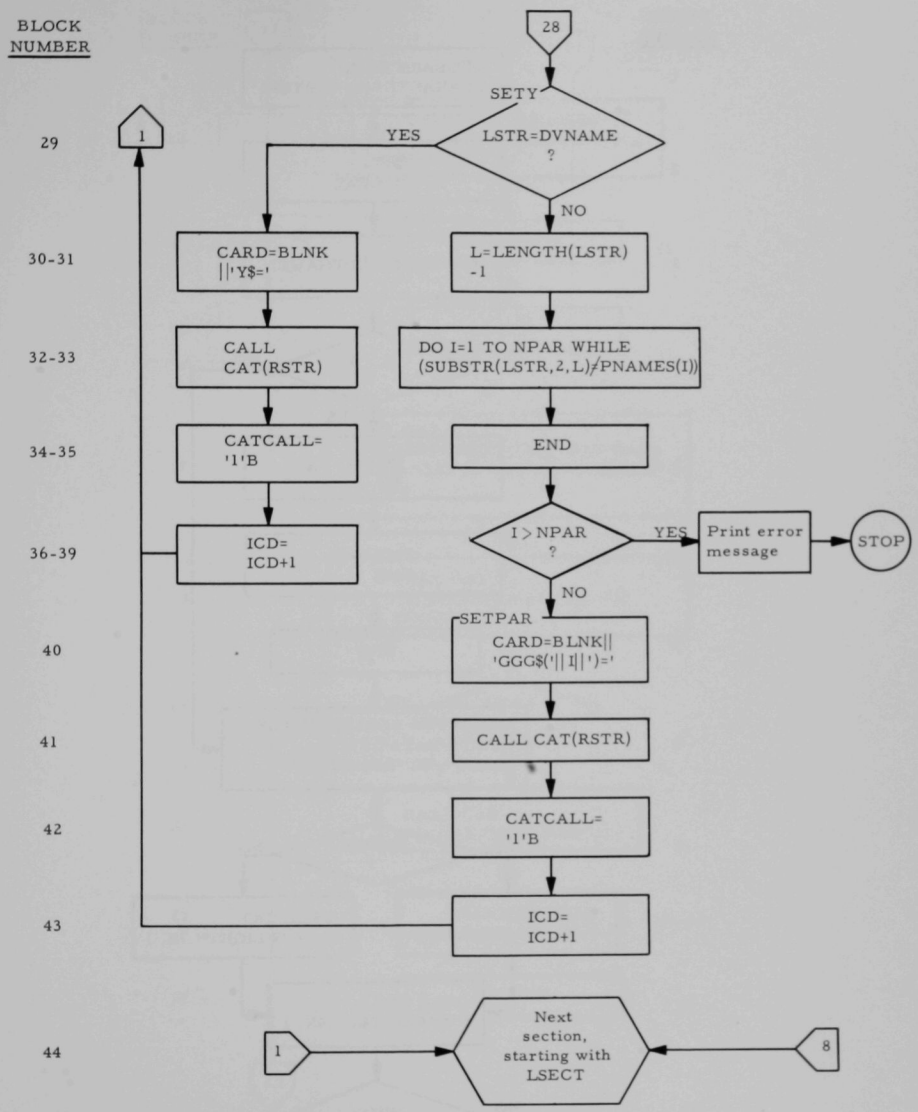

BLOCK NUMBER
28
SETY
29
1
YES
LSTR=DVNAME ?
NO
30-31
CARD=BLNK ||'Y$='
L=LENGTH(LSTR) -1
32-33
CALL CAT(RSTR)
DO I=1 TO NPAR WHILE (SUBSTR(LSTR,2,L)≠PNAMES(I))
34-35
CATCALL= '1'B
END
36-39
ICD= ICD+1
I > NPAR ?
YES
Print error message
STOP
NO
SETPAR
40
CARD=BLNK|| 'GGG$('||I||')='
41
CALL CAT(RSTR)
42
CATCALL= '1'B
43
ICD= ICD+1
44
1
Next section, starting with LSECT
8

Fig. 7 (Contd.)

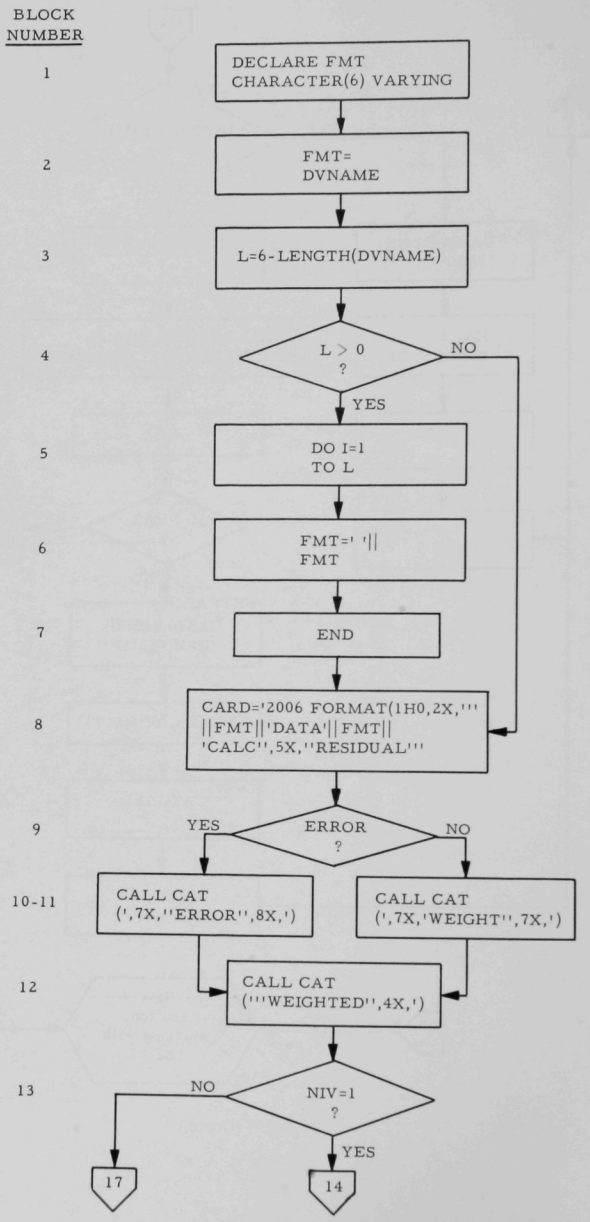


Fig. 8. Flow Chart 8, for MKFMT

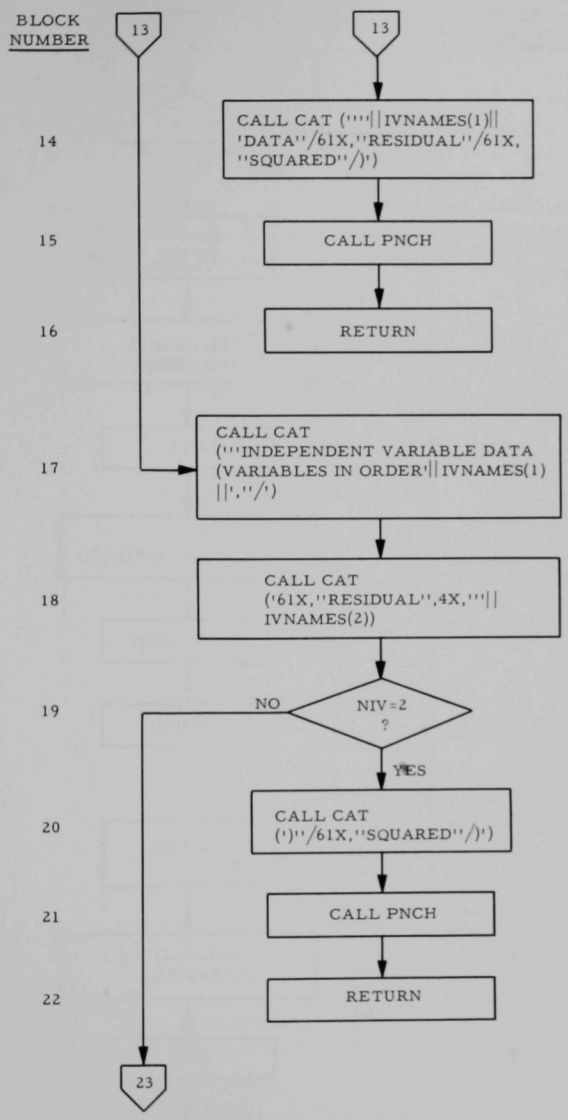

BLOCK NUMBER
13
13
14
CALL CAT (''''||IVNAMES(1)||
'DATA''/61X,''RESIDUAL''/61X,
''SQUARED''/)')
15
CALL PNCH
16
RETURN
17
CALL CAT
(''''INDEPENDENT VARIABLE DATA
(VARIABLES IN ORDER''||IVNAMES(1)
||',''/')
18
CALL CAT
('61X,''RESIDUAL'',4X,'''||
IVNAMES(2))
19
NIV=2
?
NO
YES
20
CALL CAT
(')''/61X,''SQUARED''/)')
21
CALL PNCH
22
RETURN
23

Fig. 8 (Contd.)

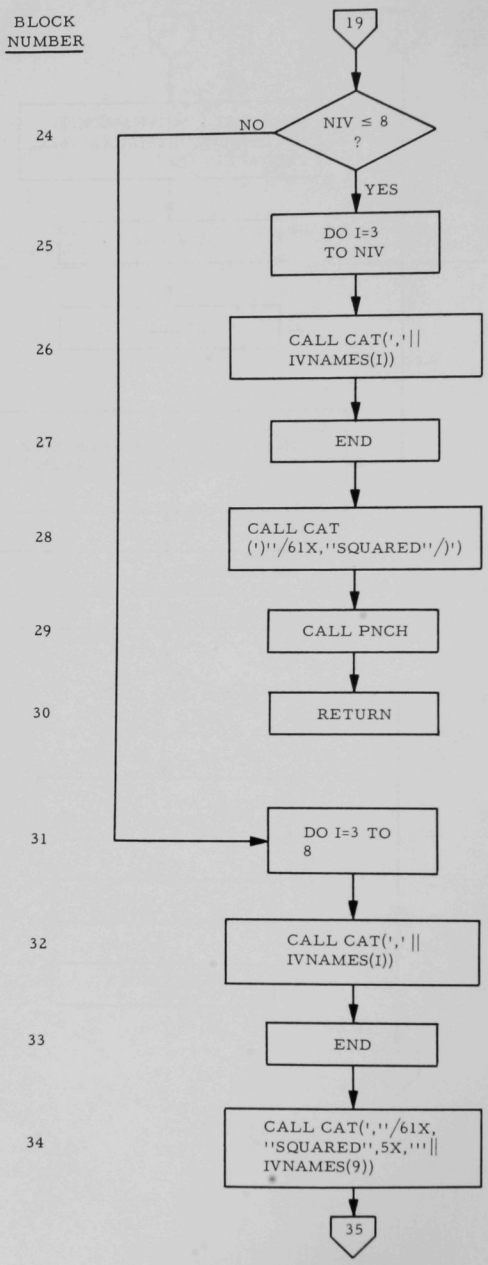

BLOCK NUMBER
19
24
NIV ≤ 8 ?
NO
YES
25
DO I=3 TO NIV
26
CALL CAT(','|| IVNAMES(I))
27
END
28
CALL CAT (')''/61X,''SQUARED''/)')
29
CALL PNCH
30
RETURN
31
DO I=3 TO 8
32
CALL CAT(',' || IVNAMES(I))
33
END
34
CALL CAT(',''/61X, ''SQUARED'',5X,'''|| IVNAMES(9))
35

Fig. 8 (Contd.)

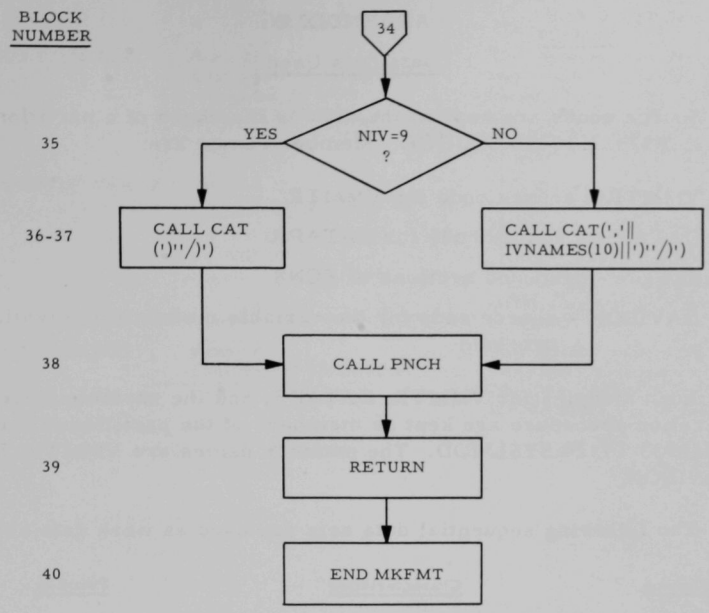

BLOCK NUMBER
34
35
YES
NIV=9 ?
NO
36-37
CALL CAT (')''/)')
CALL CAT(','|| IVNAMES(10)||')''/)')
38
CALL PNCH
39
RETURN
40
END MKFMT

Fig. 8 (Contd.)

APPENDIX D

Data Sets Used

Source codes are kept on the disk as members of a partitioned data set, C145.B12533.T7120.SOURCE. Member names are:

VMMTR - source code for VMMTR

DATAPR - source code for DATAPR

SKFCN - precoded sections of FCN$

DAVIDON - source code for the variable metric minimization program.

Load modules for VMMTR, DATAPR, and the variable metric minimization procedure are kept as members of the partitioned data set C145.B12533.T7120.SYSLMOD. The member names are VMMTR, DATAPR, and DAVIDON.

The following sequential data sets are used as work data sets:

Dsname	Characteristics	Function
C145.B12533.T7120.FORT	UNIT=2314,SPACE=(80,(50,50)), VOLUME=SER=ODDJOB, LABEL=EXPDT=68001, DCB=(RECFM=FB, LRECL=80,BLKSIZE=2000)	In VMMTR, this is the file PROG on which the subroutine FCN$ is written. As input to the FORTRAN H compiler, it has ddname SYSIN.
C145.B12533.T7120.PDAT	Same as C145.B12533.T7120.FORT	This data set is used to transfer information from VMMTR, where the file name is CDFILE, to DATAPR, where the file name is SYSIN. The data set contains the additional information, such as the number of parameters, required by the data processor, as well as card images for all the data for the cases.
C145.B12533.T7120.DATA	Same as C145.B12533.T7120.FORT	This data set contains information to be passed from the data processor to the variable metric procedure. It is the file CDFILE in DATAPR and the file FT05F001 in the variable metric program.
C145.B12533.T7120.TDAT	Same as C145.B12533.T7120.FORT	This data set is used by the data processor to store data while the points are being counted. It is used only when there is neither a DATA FORMAT card nor a DATA=INSERT option. File name is TEMP.

Dsname	Characteristics	Function
C145.B12533.T7120.XDAT	Same as C145.B12533.T7120.FORT except for the DCB: DCB=(RECFM=F,BLKSIZE=80)	This is the special unblocked file that is used for counting points and backspacing when there is a data format. In DATAPR the file name is XTFILE; in the variable metric program it is FT01F001.
C145.B12533.T7120.VMMTR	Same as C145.B12533.T7120.XDAT	This data set contains the messages written by IEHPROGM in the first job step of the procedure VMMCLG. Since the only purpose of this step is to define the program VMMTR so that the user does not need to supply a JOBLIB card, the messages are not of interest to him.
C145.B12533.T7120.EDT	Same as C145.B12533.T7120.XDAT	This data set contains the linkage editor control statements INCLUDE LIB(DATAPR,DAVIDON) ENTRY IHESAPA

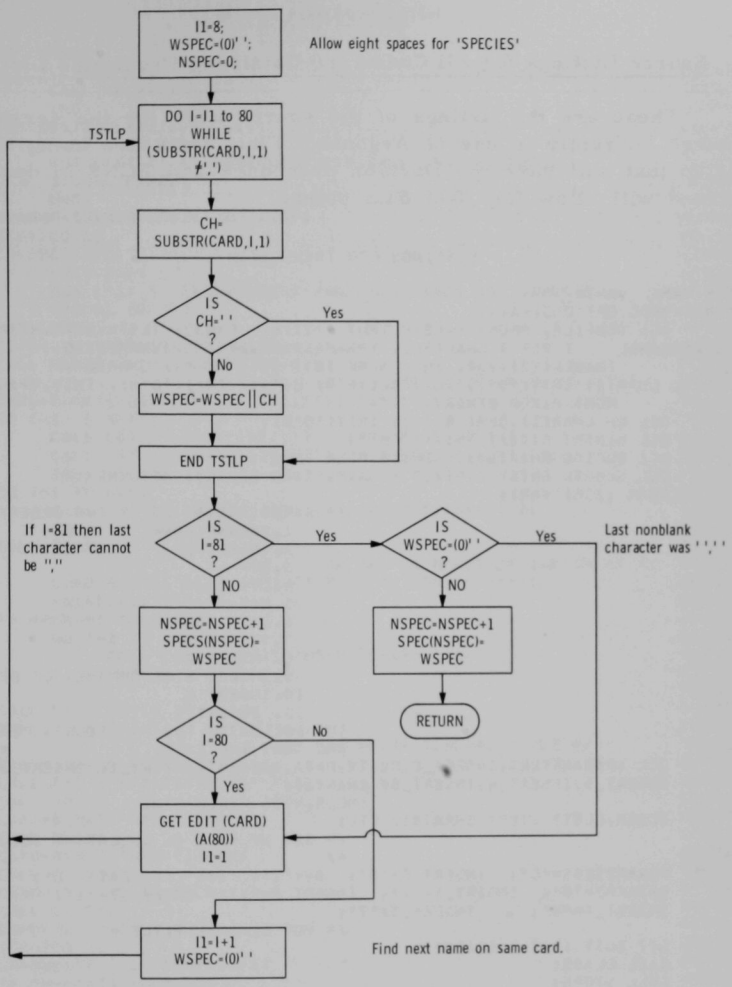
APPENDIX E

Sections from the Report on the Chemical Equation Translator

CAT and PNCH are straightforward. See the listing in Appendix F.

RDCD removes blanks from card images containing equal signs and possibly semicolons and separates the resulting string into three parts: left of equal sign, right of equal sign but left of semicolon, and right of semicolon.

Figure 9 describes the logic of ORDSP. Minor changes were made in this routine when it was included in the variable metric translator.



This reads species separated by commas, eliminating blanks. If a card ends with a null string, the program reads another card. Thus the cards look like F,
G,H

Fig. 9. Flow Chart for ORDSP

APPENDIX F

Source Listings for All Codes and Cataloged Procedure Listings

These are the listings of the source code for the version of the language currently in use at Argonne. Final tests are being made of a version that will have the Davidon procedure and FCN$ in double precision and will allow for 1000 data points.

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  VMMTR
 VMMTR  :PROC OPTIONS(MAIN);
         DCL (CDFILE, PROG) FILE OUTPUT, (TITLE, CARD2, (LSTR,RSTR,NSTR,
 CARD EXTERNAL     ) VAR ) CHAR(80),((PNAMES(40),DVNAME,IVNAMES(10),
              TNAMES(12))VAR, FMT, BLNK INIT ('       ')) CHAR(6),
    LEVEL CHAR(1) INIT('0'), (CATCALL,ERR) BIT(1) INIT('0'B), (NIV,NPAR,
              NUM) FIXED BINARY;
         DCL CH CHAR(1),DFMT BIT(1) INIT('0'B);
         DCL RINSRT BIT(1) INIT('0'B);
         DCL ENDING CHAR(80), RDFILE FILE INPUT;
         DCL SUBSTR ENTRY (,FIXED BINARY,FIXED BINARY) RETURNS
        (CHAR (256) VAR);
                                        /* CARDS ARE TO BE IN THE ORDER:
                                             1.PARAMETERS=
                                             2.INSERT 0
                                             3.DV=
                                             4.IV=
                                             5.DATA=
                                             6.DATA FORMAT
                                             7.INSERT 1
                                             8.INSERT 2
                                             9.INSERT 3 OR DVNAME= OR DPN
                                            10.INSERT 4
                                            11. INSERT 5
                                        THE FOLLOWING TWO PROCEDURES READ
                                        AND SORT THE CARDS              */
         DCL (PARAMETERS,INSERT_0,DV,IV,DATA,DATAFMT,INSERT_1, INSERT_2,
         INSERT_3,INSERT_4,INSERT_5) CHAR(1),
                                       (NCDS,NRD) FIXED BIN,
         (CDAR,CLST) (100) CHAR(81) CTL;
                                        /* SET UP TAGS IN COLLATING SEQUE
                                        */
         PARAMETERS='C';  INSERT_0='F';  DV='I';  IV='L';  DATA='O';
         DATAFMT='R';  INSERT_1='V';  INSERT_2='X';  INSERT_3='1';
         INSERT_4='4';      INSERT_5='7';
                                        /* FOR GENERAL TITLE */
         GET EDIT (TITLE)(A(80));
         CALL CLASS;
         CALL STORE;
         CALL BUBBLE;
         ICD=1;
                                        /* FOR NON-EXECUTABLE STATEMENT*/
         CALL BLOCK1;
         CARD=BLNK||
         'DIMENSION BFR$(20),WS$(500)';  CALL PNCH;
         CARD='C  IF THERE IS AN INSERT 0, IT GOES HERE';  CALL PNCH;
                                        /* FOR PARAMETER NAMES */
         CH=SUBSTR(CLST(ICD),1,1);
         IF CH¬=PARAMETERS THEN DO;
              PUT LIST (' PARAMETER NAMES NOT GIVEN');
              SIGNAL ERROR;
              END;
         RSTR=SUBSTR(CLST(ICD),2,80);
         CALL ORDER (NPAR,PNAMES,RSTR);
         ICD=ICD+1;
         CH= SUBSTR(CLST(ICD),1,1);
         IF CH= INSERT_0 THEN DO;
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  VMMTR
              LEVEL='0';                                                00000570
              CALL CDCHECK;                                             00000580
              END;                                                      00000590
         CH=SUBSTR(CLST(ICD),1,1);                                      00000600
         IF CH¬=DV THEN DO;                                             00000610
              PUT LIST (' NO DEPENDENT VARIABLE NAME GIVEN');           00000620
              SIGNAL ERROR;                                             00000630
              END;                                                      00000640
         DVNAME=SUBSTR(CLST(ICD),2,6);                                  00000650
         ICD=ICD+1;                                                     00000660
         CH=SUBSTR(CLST(ICD),1,1);                                      00000670
         IF CH¬=IV THEN DO;                                             00000680
              PUT LIST (' INDEPENDENT VARIABLE NAMES NOT GIVEN');       00000690
              SIGNAL ERROR;                                             00000700
              END;                                                      00000710
         RSTR=SUBSTR(CLST(ICD),2,80);                                   00000720
         CALL ORDER (NIV,IVNAMES,RSTR);                                 00000730
                                        /* SET UP EQUIVALENCE */        00000740
         CARD=BLNK||'EQUIVALENCE (Y$,'||DVNAME||')';                    00000750
         DO I=1 TO NPAR;                                                00000760
              CALL CAT (',(P$('||I||'),'||PNAMES(I)||')');              00000770
              CALL CAT (',(GGG$('||I||'),D'||PNAMES(I)||')');           00000780
              END;                                                      00000790
         DO I=1 TO NIV;                                                 00000800
              CALL CAT (',(X$('||I||'),'||IVNAMES(I)||')');             00000810
              END;                                                      00000820
         CALL PNCH;                                                     00000830
                                        /* SET UP NAMES OF VARIABLES */ 00000840
              CARD=BLNK||'DATA END$ /''END ''/,PNAME$/'''||             00000850
              PNAMES(1)||'''';                                          00000860
         IF NPAR¬=1 THEN DO;                                            00000870
              DO I=2 TO NPAR;                                           00000880
                   CALL CAT (','''||PNAMES(I)||'''');                   00000890
                   END;                                                 00000900
              END;                                                      00000910
         CALL CAT ('/');                                                00000920
         CALL PNCH;                                                     00000930
                                        /* PRINT GENERAL TITLE */      00000940
         CARD=' 2000 FORMAT (1H0,';                                     00000950
         CALL CAT ('''''||TITLE||''')');                                00000960
         CALL PNCH;                                                     00000970
         CARD=BLNK||'NP$='||NPAR;                                       00000980
         CALL PNCH;                                                     00000990
         CARD=BLNK||'NX$='||NIV;                                        00001000
         CALL PNCH;                                                     00001010
         CARD='C';  CALL PNCH;                                          00001020
         CALL BLOCK2;                                                   00001030
         CARD='C';  CALL PNCH;                                          00001040
         ICD=ICD+1;                                                     00001050
         CH=SUBSTR(CLST(ICD),1,1);                                      00001060
         IF CH¬=DATA THEN DO;                                           00001070
              PUT LIST (' ORDER OF DATA NOT GIVEN');                    00001080
              SIGNAL ERROR;                                             00001090
              END;                                                      00001100
                                        /* SET UP ORDER FOR READING */  00001110
         RSTR=SUBSTR(CLST(ICD),2,80);                                   00001120
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  VMMTR
        CALL ORDER (NUM,TNAMES,RSTR);                                   00001130
        ICD=ICD+1;                                                      00001140
        CH=SUBSTR (CLST(ICD),1,1);                                      00001150
        IF TNAMES(1)='INSERT' THEN DO;                                  00001160
             CARD=BLNK||' DO 11 I$=1,ND$';  CALL PNCH;                  00001170
             CARD='   11 WS$(I$)=W$(I$)'; CALL PNCH;                    00001180
             RINSRT='1'B;                                               00001190
             GO TO INSRTCK;                                             00001200
             END;                                                       00001210
        IF CH=DATAFMT THEN DO;                                          00001220
             DFMT='1'B;                                                 00001230
              CARD=BLNK||'II$=0';  CALL PNCH;                           00001240
              CARD=BLNK||                                               00001250
    'IBUF$=1';  CALL PNCH;                                              00001260
             CARD='    9 READ (IBUF$,1050) BFR$';  CALL PNCH;           00001270
              CARD=                                                     00001280
 1050 FORMAT (20A4)   ';  CALL PNCH;                                    00001290
             CARD= BLNK||'IF (BFR$(1) .EQ. END$)  GO TO 13';CALL PNCH; 00001300
              CARD=BLNK||                                               00001310
    'BACKSPACE IBUF$   ';  CALL PNCH;                                   00001320
              CARD=BLNK||                                               00001330
    'II$=II$+1   ';     CALL PNCH;                                      00001340
              CARD=BLNK||                                               00001350
    ' READ (IBUF$,1003)  '  ;                                           00001360
              END;                                                      00001370
        ELSE DO;                                                        00001380
              CARD=BLNK||'DO 13 II$=1,ND$';  CALL PNCH;                 00001390
              CARD='   13 READ (IN$, 1003)' ;                           00001400
              END;                                                      00001410
                                        /*SET UP FIRST ITEM IN DATA LIST 00001420
                                        */                              00001430
             IF TNAMES(1)=DVNAME THEN CALL CAT( 'YD$(II$)' );           00001440
                  ELSE DO;                                              00001450
                       IF TNAMES(1)='ERROR' THEN DO;                    00001460
                            ERR='1'B;                                   00001470
                            CALL CAT ('W$(II$)' )   ;                   00001480
                            GO TO LOOP;                                 00001490
                            END;                                        00001500
                     IF TNAMES(1)='WEIGHT' THEN CALL CAT('W$(II$)' );   00001510
                            ELSE DO;                                    00001520
                                 DO J=1 TO NIV WHILE (IVNAMES(J)¬=      00001530
                                      TNAMES(1));END;                   00001540
                                 CALL CAT('XD$('||J||',II$)' );         00001550
                                 END;                                   00001560
                  END;                                                  00001570
                                        /*FINISH DATA LIST*/            00001580
 LOOP:   DO I=2 TO NUM;                                                 00001590
             IF TNAMES(I)=DVNAME THEN CALL CAT(',YD$(II$)' );           00001600
                  ELSE DO;                                              00001610
                       IF TNAMES(I)='ERROR' THEN DO;                    00001620
                            ERR='1'B;                                   00001630
                            CALL CAT(',W$(II$)' )   ;                   00001640
                            GO TO ENDLP;                                00001650
                            END;                                        00001660
                     IF TNAMES(I)='WEIGHT' THEN CALL CAT( ',W$(II$)');  00001670
                            ELSE DO;                                    00001680
```

LISTINGS FOR TRANSLATOR

MEMBER NAME VMMTR

```
                                        DO J=1 TO NIV WHILE (IVNAMES(J)¬=      00001690
                                             TNAMES(I));END;                   00001700
                                        CALL CAT (',XD$('||J||',II$) ');       00001710
                                        END;                                   00001720
                    END;                                                       00001730
ENDLP:        END;                                                             00001740
        CALL PNCH;                                                             00001750
        IF DFMT THEN DO;                                                       00001760
              CARD=BLNK|| 'GO TO 9';  CALL PNCH;                               00001770
              CARD='   13 ND$=II$';  CALL PNCH     ;                           00001780
              END;                                                             00001790
             CARD=BLNK||'DO 11 I$=1,ND$';  CALL PNCH;                          00001800
        CARD=BLNK||'WS$(I$)= W$(I$)';  CALL PNCH;                              00001810
        IF ERR THEN                                                            00001820
             CARD='   11 W$(I$)=1./(W$(I$)*W$(I$))';                           00001830
        ELSE CARD='   11 CONTINUE';                                            00001840
        CALL PNCH;                                                             00001850
                                        /*CONSTRUCT FORMAT FOR READING DATA*/ 00001860
        IF DFMT THEN DO;                                                       00001870
             CARD=' 1003 FORMAT(';                                             00001880
CNTFMT:      CALL CAT(SUBSTR(CLST(ICD),2,40));                                 00001890
             CALL CAT(SUBSTR(CLST(ICD),42,40));                                00001900
                                        /* CHECK FOR CONTINUATIONS */          00001910
             ICD=ICD+1;                                                        00001920
             CH=SUBSTR(CLST(ICD),1,1);                                         00001930
             IF CH=DATAFMT THEN GO TO CNTFMT;                                  00001940
             CALL CAT (')');                                                   00001950
             CALL PNCH;                                                        00001960
             END;                                                              00001970
        ELSE DO;                                                               00001980
        CARD=' 1003 FORMAT (6E12.0)' ;  CALL PNCH;                             00001990
             END;                                                              00002000
INSRTCK:CARD='C  IF THERE IS AN INSERT 1, IT GOES HERE';  CALL PNCH;           00002010
        IF CH=INSERT_1 THEN DO;                                                00002020
             LEVEL='1';                                                        00002030
             CALL CDCHECK;                                                     00002040
             CH=SUBSTR(CLST(ICD),1,1);                                         00002050
             END;                                                              00002060
NXTBLK:  CARD='C';  CALL PNCH;                                                 00002070
        CALL BLOCK3;                                                           00002080
        CARD='C  IF THERE IS AN INSERT 2, IT GOES HERE';  CALL PNCH;           00002090
        IF CH=INSERT_2 THEN DO;                                                00002100
             LEVEL='2';                                                        00002110
             CALL CDCHECK;                                                     00002120
             CH=SUBSTR(CLST(ICD),1,1);                                         00002130
             END;                                                              00002140
        CALL BLOCK4;                                                           00002150
        CARD='C  IF THERE IS AN INSERT 3, IT GOES HERE OR AFTER FUNCTIO00002160
N OR DERIVATIVE';  CALL PNCH;                                                  00002170
        CARD='C  STATEMENTS';  CALL PNCH;                                      00002180
ENDCK:  IF ICD>NCDS THEN GO TO LSECT;                                          00002190
        LEVEL=SUBSTR(CLST(ICD),2,1);                                           00002200
        IF LEVEL='3'|LEVEL='C' THEN DO;                                        00002210
             IF CATCALL THEN DO;                                               00002220
                  CALL PNCH;                                                   00002230
                  CATCALL='0'B;                                                00002240
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  VMMTR
                    END;
               LEVEL='3';
               CALL CDCHECK;
               GO TO ENDCK;
               END;
          IF (LEVEL='4')|(LEVEL='5') THEN GO TO LSECT;
                                        /* LOOK FOR = SIGN   */
          I=INDEX(CLST(ICD),'=');
          IF I=0 THEN DO;
               NSTR=(0)' ';
               DO I=2 TO 81;
                    CH=SUBSTR(CLST(ICD),I,1);
                    IF CH¬=' ' THEN NSTR=NSTR||CH;
                    END;
               L=LENGTH(NSTR)-66;
               IF L>0 THEN DO;
                    CALL CAT(SUBSTR(NSTR,1,66));
                    CALL CAT (SUBSTR(NSTR,67,L));
                    END;
               ELSE CALL CAT(NSTR);
               CATCALL= '1'B;
               ICD=ICD+1;
               GO TO ENDCK;
               END;
                                        /*AN = SIGN WAS FOUND          */
          IF CATCALL THEN CALL PNCH;
          CATCALL='0'B;
          CARD2=SUBSTR(CLST(ICD),2,80);
          CALL RDCD(LSTR,RSTR,NSTR,CARD2);
 SETY    :IF LSTR=DVNAME THEN DO;
               CARD=BLNK||'Y$=';
               CALL CAT(RSTR);
               CATCALL='1'B;
               ICD=ICD+1;
               GO TO ENDCK;
               END;
          L=LENGTH(LSTR)-1;
                                        /* LOOK FOR PARAMETER NAME */
          DO I= 1 TO NPAR WHILE(SUBSTR(LSTR,2,L)¬=PNAMES(I));    END;
          IF I>NPAR THEN DO;
               PUT LIST (' WORD ',LSTR,' IS NEITHER A KEY WORD NOR THE NA
 ME OF THE DEPENDENT VARIABLE NOR A DPNAME FOR A PARAMETER NAME');
               SIGNAL ERROR;
               STOP;
               END;
 SETPAR :CARD=BLNK||'GGG$('||I||')=';
          CALL CAT (RSTR);
          CATCALL='1'B;
          ICD=ICD+1;
          GO TO ENDCK;
 LSECT  :      IF CATCALL THEN CALL PNCH;
          CARD = 'C END OF POSSIBLE INSERT 3';  CALL PNCH;
               CALL BLOCK5;
          CARD='C  IF THERE IS AN INSERT 4, IT GOES HERE';  CALL PNCH;
          IF LEVEL='4' THEN CALL CDCHECK;
               CALL MKFMT;
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  VMMTR
         CARD = 'C';  CALL PNCH;                                          00002810
         CALL BLOCK6;                                                     00002820
         CARD='C  IF THERE IS AN INSERT 5, IT GOES HERE';                 00002830
         CALL PNCH;                                                       00002840
         IF SUBSTR(CLST(ICD),1,1)=INSERT_5 THEN DO;                       00002850
               LEVEL='5';                                                 00002860
               CALL CDCHECK;                                              00002870
               END;                                                       00002880
         CALL BLOCK7;                                                     00002890
         CALL DTPR;                                                       00002900
 ORDER  :PROC(NSPEC,SPECS,CARD);                                          00002910
         DCL (WSPEC CHAR(6),CARD CHAR(80))VAR,SPECS(20)  CHAR(6) VAR,     00002920
         CH CHAR(1);                                                      00002930
         I1=1;                                                            00002940
         L=LENGTH(CARD);                                                  00002950
         L1=L+1;                                                          00002960
         WSPEC=(0)' ';                                                    00002970
         NSPEC=0;                                                         00002980
 TSTLP:  DO I=I1 TO L  WHILE (SUBSTR(CARD,I,1)¬=',');                     00002990
         CH=SUBSTR(CARD,I,1);                                             00003000
         IF CH¬=' ' THEN WSPEC=WSPEC||CH;                                 00003010
         END TSTLP;                                                       00003020
 /*                                  IF I=81, NONE OF THE CHARACTERS     00003030
                                     FROM I1 ON ARE ','                 */00003040
         IF I=L1 THEN DO;                                                 00003050
               IF WSPEC=(0)' ' THEN DO;                                   00003060
                   ICD=ICD+1;                                             00003070
                   CARD=SUBSTR(CLST(ICD),2,80);                           00003080
                   I1=1;                                                  00003090
                   L=80;                                                  00003100
                   L1=81;                                                 00003110
                   GO TO TSTLP; END;                                      00003120
               ELSE DO;                                                   00003130
                   NSPEC=NSPEC + 1;                                       00003140
                   SPECS(NSPEC) = WSPEC;                                  00003150
                   RETURN;                                                00003160
                   END;                                                   00003170
               END;                                                       00003180
         ELSE DO;                                                         00003190
               NSPEC = NSPEC+1;                                           00003200
               SPECS(NSPEC) = WSPEC;                                      00003210
               IF I=L  THEN DO;                                           00003220
                   ICD=ICD+1;                                             00003230
                   CARD=SUBSTR(CLST(ICD),2,80);                           00003240
                   I1=1;                                                  00003250
                   L1=81;                                                 00003260
                   L=80;                                                  00003270
                   WSPEC=(0)' ';                                          00003280
                   GO TO TSTLP;END;                                       00003290
               ELSE DO;                                                   00003300
                   I1=I+1;                                                00003310
                   WSPEC=(0)' ';                                          00003320
                   GO TO TSTLP;END;                                       00003330
         END;                                                             00003340
         RETURN    ;                                                      00003350
         END ORDER;                                                       00003360
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  VMMTR
    PNCH:PROC;
         DCL CDO CHAR(80) STATIC,CARD CHAR(80)VAR EXTERNAL;
         CDO=CARD;
         PUT FILE(PROG) EDIT (CDO) (A(80));
      /* PUT DATA (CDO) SKIP(2);    */
         RETURN;
         END PNCH;
    CAT :PROC(BCD);DCL CARD CHAR(80) VAR EXT;
         DCL(BCD CHAR(66)VAR,CDTEM CHAR(138)VAR);
         CDTEM=CARD||BCD;
         L=LENGTH(CDTEM);
         IF L<73 THEN DO;
         CARD=CDTEM;
         RETURN;END;
         ELSE;CARD=SUBSTR(CDTEM,1,72);
         CALL PNCH;L=L-72;CARD='     X'||SUBSTR(CDTEM,73,L);
         RETURN;END CAT;
 DTPR:   PROC;
         DCL CD CHAR(80) VAR;
         ON ENDFILE(SYSIN) BEGIN;
              PUT LIST (' END OF CASES' );
              CLOSE FILE(CDFILE),FILE(SYSIN),FILE(PROG),FILE(SYSPRINT),
              FILE (SYSPNCH);
              GO TO LAST;
              END;
         CD='PARAMETERS='||PNAMES(1);
         N=2;
         IF NPAR>1 THEN DO;
 PLP:         DO I=N TO NPAR;
                   CD=CD||',';
                   L=LENGTH(CD||PNAMES(I));
                   IF L>79 THEN GO TO PTCD;
                   CD=CD||PNAMES(I);
                   END;
 PTCD:        CARD2=CD;
              CD=PNAMES(I);
              PUT FILE (CDFILE) EDIT (CARD2) (A(80));
              PUT FILE  (SYSPNCH) EDIT (CARD2) (A(80));
              N=I+1;
              IF N<=NPAR THEN GO TO PLP;
              END;
                                   /*IF NPAR=1, OUTPUT THE NAME*/
         ELSE DO;
              CARD2=CD;
              PUT FILE(CDFILE) EDIT (CARD2)(A(80));
              PUT FILE (SYSPNCH)EDIT (CARD2) (A(80));
              END;
                                   /*IF 'DATA FORMAT' WAS SPECIFIED,
                                   ITST=1. IF NOT, ITST=0.  IF ALL
                                   DATA READING IS SPECIALLY
                                   PROGRAMMED WITH AN INSERT 1,
                                   INSRT=1.  OTHERWISE, INSRT=0.
                                   IN ANY CASE, OUTPUT NUM,ITST, AND
                                   INSRT  */
         IF DFMT THEN ITST= 1;
         ELSE ITST=0;
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  VMMTR
         IF RINSRT THEN INSRT=1;                                        00003930
         ELSE INSRT=0;                                                  00003940
         PUT STRING (CARD2) LIST (NUM,ITST,INSRT);                      00003950
         PUT FILE(CDFILE) EDIT (CARD2) (A(80));                         00003960
         PUT FILE  (SYSPNCH) EDIT (CARD2) (A(80));                      00003970
                                        /*PUT THE DATA,AS IT IS, ON    00003980
                                        CDFILE                        */00003990
 GETCD:  GET EDIT (CARD2) (A(80));                                      00004000
         PUT FILE (CDFILE) EDIT (CARD2) (A(80));                        00004010
         GO TO GETCD;                                                   00004020
 LAST:   RETURN;                                                        00004030
         END DTPR;                                                      00004040
 CDCHECK:PROC;                                                          00004050
         DCL CH CHAR(1);                                                00004060
 FIRST:  CARD2=SUBSTR(CLST(ICD),2,80);                                  00004070
         CH=SUBSTR(CARD2,1,1);                                          00004080
         IF CH=LEVEL THEN DO;                                           00004090
              CARD=' '||SUBSTR(CARD2,2,79);                             00004100
              CALL PNCH;                                                00004110
              ICD=ICD+1;                                                00004120
              GO TO FIRST;                                              00004130
              END;                                                      00004140
         IF CH='C' THEN DO;                                             00004150
              CARD=CARD2;                                               00004160
              CALL PNCH;                                                00004170
              ICD=ICD+1;                                                00004180
              GO TO FIRST;                                              00004190
              END;                                                      00004200
         CARD='C END OF INSERT '||LEVEL;                                00004210
         CALL PNCH;                                                     00004220
         RETURN;                                                        00004230
         END CDCHECK;                                                   00004240
  MKFMT: PROC;                                                          00004250
         DCL FMT CHAR(6) VAR;                                           00004260
         FMT=DVNAME;                                                    00004270
         L=6-LENGTH(DVNAME);                                            00004280
         IF L>0 THEN DO;                                                00004290
              DO I=1 TO L;                                              00004300
              FMT=' '||FMT;                                             00004310
              END;                                                      00004320
              END;                                                      00004330
         CARD=' 2006 FORMAT (1H0,2X,'''||FMT||' DATA   '||FMT||         00004340
        ' CALC'',5X,''RESIDUAL''';                                      00004350
         IF ERR THEN CALL CAT (',7X,''ERROR'',8X,');                    00004360
         ELSE CALL CAT (',7X,''WEIGHT'',7X,');                          00004370
         CALL CAT (''''WEIGHTED'',4X,');                                00004380
         IF NIV=1 THEN DO;                                              00004390
              CALL CAT (''''||IVNAMES(1)||' DATA''/61X,''RESIDUAL''/61X,00004400
 ''SQUARED''/)');                                                       00004410
              CALL PNCH;                                                00004420
              RETURN;                                                   00004430
              END;                                                      00004440
         CALL CAT('''INDEPENDENT VARIABLE DATA (VARIABLES IN ORDER '    00004450
         ||IVNAMES(1)||',''/');                                         00004460
         CALL CAT ('61X,''RESIDUAL'',4X,'''||IVNAMES(2));               00004470
         IF NIV=2 THEN DO;                                              00004480
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  VMMTR
                CALL CAT (')''/61X,''SQUARED''/)');
                CALL PNCH;
                RETURN;
                END;
           IF NIV<=8 THEN DO;
                DO I=3 TO NIV;
                     CALL CAT (', '||IVNAMES(I));
                     END;
                CALL CAT (')''/61X,''SQUARED''/)');
                CALL PNCH;
                RETURN;
                END;
           DO I=3 TO 8;
                CALL CAT (', '||IVNAMES(I));
                END;
           CALL CAT (',''/61X,''SQUARED'',5X,'''||IVNAMES(9));
           IF NIV=9 THEN CALL CAT (')''/)');
           ELSE CALL CAT (', '||IVNAMES(10)||')''/)');
           CALL PNCH;
           RETURN;
           END MKFMT;
  BLOCK1:  PROC;
           GET FILE (RDFILE)EDIT(CARD) (A(80));
           IF SUBSTR(CARD ,1,6) ¬='BLOCK1' THEN DO;
                PUT LIST (' BLOCK NAME 1 NOT GIVEN');
                SIGNAL ERROR;
                STOP;
                END;
  GTPRG:   GET FILE (RDFILE) EDIT (CARD) (A(80));
           IF SUBSTR(CARD,1,3) = 'END' THEN RETURN;
           CALL PNCH;
           GO TO GTPRG;
           END BLOCK1;
  BLOCK2:  PROC;
           GET FILE (RDFILE)EDIT(CARD) (A(80));
           IF SUBSTR(CARD ,1,6) ¬='BLOCK2' THEN DO;
                PUT LIST (' BLOCK NAME 2 NOT GIVEN');
                SIGNAL ERROR;
                STOP;
                END;
  GTPRG:   GET FILE (RDFILE) EDIT (CARD) (A(80));
           IF SUBSTR(CARD,1,3) = 'END' THEN RETURN;
           CALL PNCH;
           GO TO GTPRG;
           END BLOCK2;
  BLOCK3:  PROC;
           GET FILE (RDFILE)EDIT(CARD) (A(80));
           IF SUBSTR(CARD ,1,6) ¬='BLOCK3' THEN DO;
                PUT LIST (' BLOCK NAME 3 NOT GIVEN');
                SIGNAL ERROR;
                STOP;
                END;
  GTPRG:   GET FILE (RDFILE) EDIT (CARD) (A(80));
           IF SUBSTR(CARD,1,3) = 'END' THEN RETURN;
           CALL PNCH;
           GO TO GTPRG;
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  VMMTR
         END BLOCK3;                                                    00005050
 BLOCK4: PROC;                                                          00005060
         GET FILE (RDFILE)EDIT(CARD) (A(80));                           00005070
         IF SUBSTR(CARD ,1,6) ¬='BLOCK4' THEN DO;                       00005080
              PUT LIST (' BLOCK NAME 4 NOT GIVEN');                     00005090
              SIGNAL ERROR;                                             00005100
              STOP;                                                     00005110
              END;                                                      00005120
 GTPRG:  GET FILE (RDFILE) EDIT (CARD) (A(80));                         00005130
         IF SUBSTR(CARD,1,3) = 'END' THEN RETURN;                       00005140
         CALL PNCH;                                                     00005150
         GO TO GTPRG;                                                   00005160
         END BLOCK4;                                                    00005170
 BLOCK5: PROC;                                                          00005180
         GET FILE (RDFILE)EDIT(CARD) (A(80));                           00005190
         IF SUBSTR(CARD ,1,6) ¬='BLOCK5' THEN DO;                       00005200
              PUT LIST (' BLOCK NAME 5 NOT GIVEN');                     00005210
              SIGNAL ERROR;                                             00005220
              STOP;                                                     00005230
              END;                                                      00005240
 GTPRG:  GET FILE (RDFILE) EDIT (CARD) (A(80));                         00005250
         IF SUBSTR(CARD,1,3) = 'END' THEN RETURN;                       00005260
         CALL PNCH;                                                     00005270
         GO TO GTPRG;                                                   00005280
         END BLOCK5;                                                    00005290
 BLOCK6: PROC;                                                          00005300
         GET FILE (RDFILE)EDIT(CARD) (A(80));                           00005310
         IF SUBSTR(CARD ,1,6) ¬='BLOCK6' THEN DO;                       00005320
              PUT LIST (' BLOCK NAME 6 NOT GIVEN');                     00005330
              SIGNAL ERROR;                                             00005340
              STOP;                                                     00005350
              END;                                                      00005360
 GTPRG:  GET FILE (RDFILE) EDIT (CARD) (A(80));                         00005370
         IF SUBSTR(CARD,1,3) = 'END' THEN RETURN;                       00005380
         CALL PNCH;                                                     00005390
         GO TO GTPRG;                                                   00005400
         END BLOCK6    ;                                                00005410
 BLOCK7: PROC;                                                          00005420
         GET FILE (RDFILE)EDIT(CARD) (A(80));                           00005430
         IF SUBSTR(CARD ,1,6) ¬='BLOCK7' THEN DO;                       00005440
              PUT LIST (' BLOCK NAME 7 NOT GIVEN');                     00005450
              SIGNAL ERROR;                                             00005460
              STOP;                                                     00005470
              END;                                                      00005480
 GTPRG:  GET FILE (RDFILE) EDIT (CARD) (A(80));                         00005490
         IF SUBSTR(CARD,1,3) = 'END' THEN RETURN;                       00005500
         CALL PNCH;                                                     00005510
         GO TO GTPRG;                                                   00005520
         END BLOCK7    ;                                                00005530
 CLASS:  PROC;                                                          00005540
         DCL ADCHAR CHAR(1),INDEQ FIXED BIN;                            00005550
         ALLOCATE CDAR(100) CHAR(81);                                   00005560
         NCDS=100;                                                      00005570
         NRD=0;                                                         00005580
         ADCHAR=PARAMETERS;                                             00005590
 GETCD:  GET EDIT(CARD2)(A(80));                                        00005600
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  VMMTR
          PUT DATA (CARD2) SKIP;
          IF CARD2=('END'||(77)' ') THEN RETURN;
          NRD=NRD+1;
          CH=SUBSTR(CARD2,1,1);
                                        /* LOOK FOR INSERT */
          IF ('0'<=CH)&('5'>=CH) THEN DO;
               IF CH='0' THEN DO;
                    ADCHAR=INSERT_0;
                    CDAR(NCDS)=(ADCHAR ||CARD2);
                    GO TO CKNCD;
                    END;
               IF CH='1' THEN DO;
                    ADCHAR=INSERT_1;
                    CDAR(NCDS)=(ADCHAR||CARD2);
                    GO TO CKNCD;
                    END;
               IF CH='2' THEN DO;
                    ADCHAR=INSERT_2;
                    CDAR(NCDS)=(ADCHAR||CARD2);
                    GO TO CKNCD;
                    END;
               IF CH='3' THEN DO;
                    ADCHAR=INSERT_3;
                    CDAR(NCDS)=(ADCHAR||CARD2);
                    GO TO CKNCD;
                    END;
               IF CH='4' THEN DO;
                    ADCHAR=INSERT_4;
                    CDAR(NCDS)=(ADCHAR||CARD2);
                    GO TO CKNCD;
                    END;
               ADCHAR=INSERT_5;
               CDAR(NCDS)=(ADCHAR||CARD2);
               GO TO CKNCD;
               END;
          IF CH='C' THEN DO;
               CDAR(NCDS)=(ADCHAR||CARD2);
               GO TO CKNCD;
               END;
                                        /* CHECK FOR = SIGN */
          INDEQ=INDEX(CARD2,'=');
          IF INDEQ=0 THEN DO;
               NSTR=(0)' ';
               DO I=1 TO 80;
                    CH=SUBSTR(CARD2,I,1);
                    IF CH¬=' ' THEN NSTR=(NSTR||CH);
                    IF LENGTH(NSTR)=10 THEN GO TO TSTFMT;
                    END;
 TSTFMT:       IF NSTR='DATAFORMAT' THEN DO;
                    ADCHAR=DATAFMT;
                    ISTR=I+1;
                    CDAR(NCDS)=(ADCHAR||SUBSTR(CARD2,ISTR,80));
                    GO TO CKNCD;
                    END;
                                        /*IF THE CARD IS NOT A DATA
                                        FORMAT SPECIFICATION, A CONTINUAT
```

LISTINGS FOR TRANSLATOR

MEMBER NAME VMMTR

```
                                        IS ASSUMED*/
            CDAR(NCDS)=(ADCHAR||CARD2);
            GO TO CKNCD;
            END;
        CALL RDCD(LSTR,RSTR,NSTR,CARD2);
        IF LSTR='PARAMETERS' THEN DO;
            ADCHAR=PARAMETERS;
            CDAR(NCDS)=(ADCHAR||RSTR||NSTR);
            GO TO CKNCD;
            END;
        IF LSTR='DV' THEN DO;
            ADCHAR=DV;
            CDAR(NCDS)=(ADCHAR||RSTR||NSTR);
            GO TO CKNCD;
            END  ;
        IF LSTR='IV' THEN DO;
            ADCHAR=IV  ;
            CDAR(NCDS)=(ADCHAR||RSTR||NSTR);
            GO TO CKNCD;
            END;
        IF LSTR='DATA' THEN DO;
            ADCHAR=DATA;
            CDAR(NCDS)=(ADCHAR||RSTR||NSTR);
            GO TO CKNCD;
            END;
        IF LSTR='DATAFORMAT' THEN DO;
            ADCHAR=DATAFMT;
            CDAR(NCDS)=(ADCHAR||SUBSTR(CARD2,INDEQ+1,80));
            GO TO CKNCD;
            END;
                                        /* ASSUME HERE THAT ANY OTHER
                                        CARD CONTAINING = GIVES A FORMULA
                                        FOR THE DEPENDENT VARIABLE OR A
                                        DERIVATIVE                    */
        ADCHAR=INSERT_3;
        CDAR(NCDS)=(ADCHAR||CARD2);
CKNCD:  NCDS=NCDS-1;
        IF NCDS>0 THEN GO TO GETCD;
        ALLOCATE CDAR(100) CHAR(81);
        NCDS=100;
        GO TO GETCD;
        END CLASS;
STORE:  PROC;
        DCL (ICD,K,I) FIXED BINARY      ;
                                        /* THIS ROUTINE ARRANGES THE TAGG
                                        CARD IMAGES IN ORDER FROM FIRST R
                                        TO LAST READ AND STORES THEM IN C
                                        */
        NAL=NRD+1;
        ALLOCATE CLST(NAL) CHAR(81);
        CLST(NAL)=(81)' ';
        K=NCDS;
        DO I=NRD  TO 1 BY -1;
              K=K+1;
              CLST(I) = CDAR(K);
              IF K=100 THEN DO;
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  VMMTR
                    FREE CDAR;
                    K=0;
                    END;
               END;
          NCDS=NRD;
          RETURN;
          END STORE;
 SUBSTR:PROC(STR,IS,L) CHAR (256) VAR;
          DCL STR CHAR(256) VAR,(IS,L)FIXED BINARY;
          DCL S CHAR(256) VAR;
          DCL SUBSTR BUILTIN;
          M=L;
          LN=LENGTH(STR);
          IF L+IS-1>LN THEN M=LN-IS+1;
          IF IS>LN THEN S=(0)' ';
               ELSE S=SUBSTR(STR,IS,M);
          RETURN (S);
          END;
 RDCD    :PROC(LSTR,RSTR,RTSTR,CARD);
          DCL CH CHAR(1) STATIC,CARD CHAR(80);
          DCL      ( LSTR,RSTR,RTSTR)VAR  CHAR(80) ;
          LSTR,RSTR,RTSTR=(0)' ';
          DO IL=1 TO 80 WHILE (SUBSTR(CARD,IL,1)¬='=');
          CH=SUBSTR(CARD,IL,1);IF CH=' ' THEN GO TO L1001;
          ELSE LSTR=LSTR||CH;
 L1001   :END;
          IF IL=80 THEN DO;PUT LIST('NO = SIGN FOUND IN THIS CARD  ',
          CARD);GO TO ERROR;END;
          DO IR=IL+1 TO 80
                    WHILE (SUBSTR(CARD,IR,1)¬=';');
          CH=SUBSTR(CARD,IR,1);IF CH=' ' THEN GO TO L1002;
          ELSE RSTR=RSTR||CH;
 L1002   :END;
          IF IR=80 THEN RETURN;
          RTSTR=SUBSTR(CARD,IR+1,80);
          RETURN;
 ERROR   :CLOSE FILE(SYSPRINT);SIGNAL ERROR;
          END RDCD;
 BUBBLE: PROC;
          DCL TEMP CHAR(81),((SWAP,FALSE) INIT('0'B),TRUE INIT('1'B))
          BIT (8),SUBSTR BUILTIN;
 SWPLP:  DO I=NCDS TO 2 BY -1;
               IF SUBSTR(CLST(I),1,1)<SUBSTR(CLST(I-1),1,1) THEN DO;
                    TEMP=CLST(I-1);
                    CLST(I-1)=CLST(I);
                    CLST(I)=TEMP;
                    SWAP=TRUE;
                    END;
               END;
          IF SWAP=TRUE THEN DO;
               SWAP=FALSE;
               GO TO SWPLP;
               END;
          RETURN;
          END BUBBLE;
          END VMMTR;
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  DATAPR
 (SUBRG):                                                                  00000010
 DATAPR: PROC OPTIONS(MAIN);                                               00000020
         ON SUBSCRIPTRANGE BEGIN;                                          00000030
              PUT LIST (' SUBSCRIPT RANGE INTERRUPT');                     00000040
              CALL IHEDUMP;                                                00000050
              END;                                                         00000060
         DCL SUBSTR ENTRY (,FIXED BINARY,FIXED BINARY) RETURNS             00000070
        (CHAR (256) VAR);                                                  00000080
         DCL CDFILE FILE OUTPUT, TEMP FILE,(CARD2,(LSTR,RSTR,NSTR)VAR)     00000090
         CHAR(80), PNAMES(40) CHAR(6) VAR, (NPAR,NUM,DFMT) FIXED BIN,      00000100
         (DATA(12),(GUESS,GUESSF,SD,SDF)(40)) FLOAT DEC,(FRST,NXT)BIT(1)00000110
         INIT ('1'B), BL CHAR(1) INIT(' '), XTFILE FILE OUTPUT;            00000120
                                        /*CHECK FOR END OF FILE */        00000130
         ON ENDFILE(SYSIN) BEGIN;                                          00000140
              PUT LIST (' END OF CASES');                                  00000150
              NXT='0'B;                                                    00000160
              CLOSE FILE(SYSIN),FILE(CDFILE),FILE(SYSPRINT),               00000170
              FILE (XTFILE),FILE (TEMP);                                   00000180
              END;                                                         00000190
         OPEN FILE(CDFILE),FILE(TEMP) OUTPUT, FILE(SYSIN);                 00000200
         GET EDIT(CARD2) (A(80));                                          00000210
         CALL RDCD (LSTR,RSTR,NSTR,CARD2);                                 00000220
                                        /*FIND PARAMETER NAMES*/          00000230
         IF LSTR¬='PARAMETERS' THEN DO;                                    00000240
              PUT LIST(' PARAMETER NAMES NOT GIVEN');                      00000250
              SIGNAL ERROR;                                                00000260
              END;                                                         00000270
         CALL ORDER(NPAR,PNAMES,RSTR);                                     00000280
         GET EDIT (CARD2) (A(80));                                         00000290
         GET STRING(CARD2) LIST (NUM,DFMT,INSRT);                          00000300
         GET EDIT(CARD2) (A(80));                                          00000310
                                        /*THIS IS THE CASE TITLE*/        00000320
 START:  PUT FILE (CDFILE) EDIT (CARD2) (A(80));                           00000330
         IF FRST THEN DO;                                                  00000340
                                        /*SET UP GUESSES TO BE STORED*/   00000350
              FRST='0'B;                                                   00000360
              DO J=1 TO NPAR;                                              00000370
                   GET EDIT (CARD2) (A(80));                               00000380
                   CALL RDCD (LSTR,RSTR,NSTR,CARD2);                       00000390
                   DO I=1 TO NPAR WHILE(LSTR¬=PNAMES(I));  END;            00000400
                   IF I=NPAR+1 THEN DO;                                    00000410
                        PUT LIST ('THE PARAMETER NAME ',LSTR,              00000420
                        'WAS NOT FOUND IN THE LIST OF PARAMETER NAMES');00000430
                        SIGNAL ERROR;                                      00000440
                        END;                                               00000450
                   GUESSF(I)=RSTR;                                         00000460
                   GUESSF(I)=GUESSF(I)+GUESSF(I)*.000001 ;                 00000470
                   IF NSTR='CONSTANT' THEN SDF(I)=0.0;                     00000480
                   ELSE DO;                                                00000490
                        IF NSTR=(0)' ' THEN SDF(I)=-1.0;                   00000500
                        ELSE DO;                                           00000510
                             SDF(I)=NSTR;                                  00000520
                             SDF(I)=SDF(I)+SDF(I)*.000001;                 00000530
                             END;                                          00000540
                        END;                                               00000550
                   END;                                                    00000560
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  DATAPR
                END;
          DO I=1 TO NPAR;
                GUESS(I)=GUESSF(I);
                SD(I)=SDF(I);
                END;
                                        /*CHECK FOR CHANGES IN PARAMETERS
                                        */
 GTCD:    GET EDIT (CARD2) (A(80));
                                        /*LOOK FOR = SIGN*/
          DO I=1 TO 80 WHILE(SUBSTR(CARD2,I,1)¬='=');  END;
          IF I¬=81 THEN DO;
                CALL RDCD(LSTR,RSTR,NSTR,CARD2);
                DO I=1 TO 80 WHILE (LSTR¬=PNAMES(I));   END;
                IF I=NPAR+1 THEN DO;
                     PUT LIST (' THE PARAMETER NAME ',LSTR,
                     ' DOES NOT APPEAR IN THE LIST OF PARAMETERS');
                     SIGNAL ERROR;
                     END;
                GUESS(I)=RSTR;
                GUESS(I)=GUESS(I)+GUESS(I)*.000001;
                IF NSTR='CONSTANT' THEN SD(I)=0.0  ;
                ELSE DO;
                     IF NSTR=(0)' ' THEN SD(I)=-1.0;
                     ELSE DO;
                          SD(I)=NSTR;
                          SD(I)=SD(I)+SD(I)*.000001;
                          END;
                     END;
                GO TO GTCD;
                END;
          PUT FILE (CDFILE) EDIT((GUESS(I),SD(I) DO I=1 TO NPAR))
          (2 E(12,5),X(56));
          PUT FILE(CDFILE) EDIT (BL)(X(55),A(1));
          IF INSRT=1 THEN DO;
                NPTS=500;
                PUT FILE (CDFILE) EDIT (NPTS,BL) (F(6,0),X(73),A(1));
                GO TO RXTRA;
                END;
          IF DFMT¬=1 THEN DO;
                ON CONVERSION BEGIN;
                     CLOSE FILE(TEMP);
                     IF ONSOURCE¬='END' THEN DO;
                          PUT LIST (' CONVERSION ERROR ON CHARACTER ',
                          ONCHAR,' OF STRING ',ONSOURCE );
                          SIGNAL ERROR;
                          END;
                     GO TO PTFILE;
                     END;
                OPEN FILE (TEMP) OUTPUT;
                NPTS=0;
                IF NUM<=6 THEN NX=6-NUM;
                ELSE NX=12-NUM;
 GTLST:         DO J=1 TO NUM;
                     GET LIST (DATA(J));
                     DATA(J) = DATA(J) + DATA(J)*.000001;
                     END;
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  DATAPR
             PUT FILE (TEMP) EDIT ((DATA(J) DO J=1 TO NUM)) (6 E(12,5),00001130
             X(8));                                                   00001140
             PUT FILE (TEMP) EDIT ((BL DO J=1 TO NX))(X(11),A(1));    00001150
             PUT FILE (TEMP) EDIT (BL) (X(7),A(1));                   00001160
             NPTS=NPTS+1;                                             00001170
             GO TO GTLST;                                             00001180
 PTFILE:     GET EDIT (CARD2) (A(76));                                00001190
             PUT FILE (CDFILE) EDIT (NPTS,BL) (F(6,0),X(73),A(1));    00001200
             OPEN FILE(TEMP) INPUT;                                   00001210
             ON ENDFILE (TEMP) BEGIN;                                 00001220
                  CLOSE FILE(TEMP);                                   00001230
                  GO TO  RXTRA;                                       00001240
                  END;                                                00001250
 GTTMP:      GET FILE(TEMP) EDIT(CARD2) (A(80));                      00001260
             PUT FILE (CDFILE) EDIT (CARD2)(A(80));                   00001270
             GO TO GTTMP;                                             00001280
             END;                                                     00001290
                                  /*IF A DATA FORMAT WAS GIVEN, THE00001300
                                  FOLLOWING PROCESS IS USED      */00001310
         ELSE DO;                                                     00001320
             OPEN FILE (XTFILE);                                      00001330
              NPTS=500;                                               00001340
             PUT FILE (CDFILE) EDIT (NPTS,BL) (F(6,0),X(73),A(1));    00001350
 GTCD2:      GET EDIT (CARD2) (A(80));                                00001360
             IF SUBSTR(CARD2,1,3)¬='END' THEN DO;                     00001370
                  PUT FILE (XTFILE) EDIT (CARD2) (A(80));             00001380
                  GO TO GTCD2;                                        00001390
                  END;                                                00001400
             PUT FILE (XTFILE) EDIT (CARD2) (A(80));                  00001410
             END;                                                     00001420
 RXTRA:  GET EDIT (CARD2) (A(80));                                    00001430
         IF SUBSTR(CARD2,1,3) = 'END' THEN DO;                        00001440
             GET EDIT (CARD2) (A(80));                                00001450
             IF NXT THEN GO TO START;                                 00001460
             ELSE GO TO LAST;                                         00001470
             END;                                                     00001480
         PUT FILE (CDFILE) EDIT (CARD2) (A(80));                      00001490
         GO TO RXTRA;                                                 00001500
 SUBSTR:PROC(STR,IS,L) CHAR (256) VAR;                                00001510
         DCL STR CHAR(256) VAR,(IS,L)FIXED BINARY;                    00001520
         DCL S CHAR(256) VAR;                                         00001530
         DCL SUBSTR BUILTIN;                                          00001540
         M=L;                                                         00001550
         LN=LENGTH(STR);                                              00001560
         IF L+IS-1>LN THEN M=LN-IS+1;                                 00001570
         IF IS>LN THEN S=(0)' ';                                      00001580
              ELSE S=SUBSTR(STR,IS,M);                                00001590
         RETURN (S);                                                  00001600
         END;                                                         00001610
 ORDER  :PROC(NSPEC,SPECS,CARD);                                      00001620
         DCL (WSPEC CHAR(6),CARD CHAR(80))VAR,SPECS(20)  CHAR(6) VAR, 00001630
         CH CHAR(1);                                                  00001640
         I1=1;                                                        00001650
         L=LENGTH(CARD);                                              00001660
         L1=L+1;                                                      00001670
         WSPEC=(0)' ';                                                00001680
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  DATAPR
         NSPEC=0;
 TSTLP:  DO I=I1 TO L  WHILE (SUBSTR(CARD,I,1)¬=',');
         CH=SUBSTR(CARD,I,1);
         IF CH¬=' ' THEN WSPEC=WSPEC||CH;
         END TSTLP;
 /*                                   IF I=81, NONE OF THE CHARACTERS
                                      FROM I1 ON ARE ','              */
         IF I=L1 THEN DO;
               IF WSPEC=(0)' ' THEN DO;
                   GET EDIT (CARD) (A(80));
                   I1=1;
                   L=80;
                   L1=81;
                   GO TO TSTLP; END;
               ELSE DO;
                   NSPEC=NSPEC + 1;
                   SPECS(NSPEC) = WSPEC;
                   RETURN;
                   END;
               END;
         ELSE DO;
               NSPEC = NSPEC+1;
               SPECS(NSPEC) = WSPEC;
               IF I=L  THEN DO;
                   GET EDIT (CARD) (A(80));
                   I1=1;
                   L1=81;
                   L=80;
                   WSPEC=(0)' ';
                   GO TO TSTLP;END;
               ELSE DO;
                   I1=I+1;
                   WSPEC=(0)' ';
                   GO TO TSTLP;END;
         END;
         RETURN    ;
         END ORDER;
 RDCD   :PROC(LSTR,RSTR,RTSTR,CARD);
         DCL CH CHAR(1) STATIC,CARD CHAR(80);
         DCL      ( LSTR,RSTR,RTSTR)VAR  CHAR(80) ;
         LSTR,RSTR,RTSTR=(0)' ';
         DO IL=1 TO 80 WHILE (SUBSTR(CARD,IL,1)¬='=');
         CH=SUBSTR(CARD,IL,1);IF CH=' ' THEN GO TO L1001;
         ELSE LSTR=LSTR||CH;
 L1001  :END;
         IF IL=80 THEN DO;PUT LIST('NO = SIGN FOUND IN THIS CARD  ',
         CARD);GO TO ERROR;END;
         DO IR=IL+1 TO 80
                 WHILE (SUBSTR(CARD,IR,1)¬=';');
         CH=SUBSTR(CARD,IR,1);IF CH=' ' THEN GO TO L1002;
         ELSE RSTR=RSTR||CH;
 L1002  :END;
         IF IR=80 THEN RETURN;
         RTSTR=SUBSTR(CARD,IR+1,80);
         RETURN;
 ERROR  :CLOSE FILE(SYSPRINT);SIGNAL ERROR;
         END RDCD;
 LAST:   CALL DVDN    ;
         END DATAPR;
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  SKFCN
BLOCK1
      SUBROUTINE FCN$(NP$,G$,F$,R$,MM1$)
      REAL*8 PNAME$
      COMMON H$(40,40),PP$(40),GG$(40),S$(40),XP$(40),GP$(40),T$(40),
     1GB$(40),FF$,GS$,EL$,SL$,FP$,GSP$,TO$,Z$,Q$,A$,GSS$,FO$,GTP$,FB$,
     2GTT$,GSB$,FAC$,DELTA$,E$,RS$,C$(40,10),N$,K$,NC$,NSSW1$,NSSW2$,MS$
     3 ,IT$,M1$,L$
      DIMENSION HSTOR$(40,40),GINT$(40),CORR$(40)
      DIMENSION CASET$(20),ESTP$(40),SD$(40),P$(40),YD$(500),W$(500),
     1XD$(20,500),X$(20),G$(40),GGG$(40),GENT$(20),YSAVE$(500),
     2PSAVE$(40),GSAVE$(40),YC$(500),PNAME$(40),R$(40)
END
BLOCK2
      DO 4 I$=1,NP$
    4 P$(I$)=R$(I$)
      GO TO (5,15,65,45), MM1$
    5 IN$=5
      IO$=6
      IP$=7
      NC$=0
      K$=0
      NSSW1$=0
      NSSW2$=0
      RS$=0.
      E$=0
      READ(IN$,1000,END=80)(CASET$(I$),I$=1,20)
      READ(IN$,1001)(ESTP$(I$),SD$(I$),I$=1,NP$)
      DO 10 I$=1,NP$
      IF(SD$(I$) .GE. 0.) GO TO 8
      SD$(I$)=.1*ESTP$(I$)
      IF(SD$(I$) .EQ. 0.) SD$(I$)=.1
    8 P$(I$)=ESTP$(I$)
      R$(I$)=P$(I$)
      DO 10JJ$=1,NP$
      II$=I$
      H$(II$,JJ$)=0.
      IF (II$.EQ.JJ$) H$(II$,JJ$)=SD$(I$)*SD$(I$)
   10 CONTINUE
      READ(IN$,1002) ND$
      DO 12 I$=1,ND$
   12 W$(I$)=1.
END
BLOCK3
   15 F$=0
END
BLOCK4
      DO 20 I$=1,NP$
   20 G$(I$)=0.
      DO 30 II$=1,ND$
      DO 25 JJ$=1,NX$
   25 X$(JJ$)= XD$(JJ$,II$)
END
BLOCK5
      YC$(II$)=Y$
      DF$=YD$(II$)-Y$
      F$=F$+DF$*DF$*W$(II$)
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  SKFCN
      DFW$=W$(II$)*DF$*2.
      DO 30 I$=1,NP$
   30 G$(I$)=G$(I$)-GGG$(I$)*DFW$
      IF(MM1$.NE.1) RETURN
      FINT$=F$
      DELTA$=1.
      IDF$=ND$-NP$
      DO 33 I$=1,NP$
      IF ( SD$(I$) .EQ. 0. ) IDF$=IDF$+1
   33 CONTINUE
      CF$=IDF$/(2.*F$)
      DO 35 II$=1,NP$
      GINT$(II$)=G$(II$)
      DO 35 JJ$=1,NP$
   35 H$(II$,JJ$)=H$(II$,JJ$)*CF$
      IF (E$ .EQ. 0. ) E$=F$/(10.**6)
      GO TO 50
   45 IF ( RS$ .EQ. 0.) RS$=SQRT(F$)
      IF (F$.GT.FSAVE$) RETURN
   50 FSAVE$=F$
      DO 55 I$=1,NP$
      PSAVE$(I$)=P$(I$)
      GSAVE$(I$)=G$(I$)
      II$=I$
      DO 55 JJ$=1,NP$
   55 HSTOR$(II$,JJ$)=H$(II$,JJ$)
      DO 60 I$=1,ND$
   60 YSAVE$(I$)=YC$(I$)
      RETURN
   65 WRITE (IO$,2001)
      WRITE(IO$,2000)
      WRITE (IO$,2002)(CASET$(I$),I$=1,20)
      WRITE (IO$,2003)
      IDF$=ND$-NP$
      SE$=FSAVE$/IDF$
      DO 70 I$=1,NP$
      II$=I$
      SS$=SQRT(2.*HSTOR$(II$,II$)*SE$)
   70 WRITE(IO$,2004)PNAME$(I$),ESTP$(I$),PSAVE$(I$),SS$,GINT$(I$),
     1       GSAVE$(I$)
      WRITE (IO$,2009) (PNAME$(I$),I$=1,NP$)
      DO 73 II$=1,NP$
      DO 72 JJ$=1,NP$
      IF(HSTOR$(II$,II$).NE.0..AND.HSTOR$(JJ$,JJ$).NE.0.)GO TO 71
      CORR$(JJ$)=0.
      GO TO 72
   71 CORR$(JJ$)=HSTOR$(II$,JJ$)/(SQRT(HSTOR$(II$,II$))*SQRT(HSTOR$(JJ$,
     1          JJ$)))
   72 CONTINUE
   73 WRITE (IO$,2010) PNAME$(II$),(CORR$(JJ$),JJ$=1,NP$)
      WRITE(IO$,2005)FINT$
      WRITE(IO$,2007)FSAVE$
      WRITE (IO$,2011)SE$,IDF$
      WRITE(IO$,2006)
      DO 75 II$=1,ND$
      DF$=YD$(II$)-YSAVE$(II$)
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  SKFCN
      WSR$=W$(II$)* DF$*DF$                                              00001130
   75 WRITE(IO$,2008)YD$(II$),YSAVE$(II$),DF$,WS$(II$),WSR$,             00001140
     1      (XD$(JJ$,II$),JJ$=1,NX$)                                     00001150
END                                                                      00001160
BLOCK6                                                                   00001170
   80 MM1$=3                                                             00001180
END                                                                      00001190
BLOCK7                                                                   00001200
      RETURN                                                             00001210
 1000 FORMAT( 20A4)                                                      00001220
 1001 FORMAT (2E12.0)                                                    00001230
 1002 FORMAT( I6)                                                        00001240
 2001 FORMAT(29H1VARIABLE METRIC MINIMIZATION )                          00001250
 2002 FORMAT(/1H ,20A4/ )                                                00001260
 2003 FORMAT(15H PARAMETER NAME,5X,16HINITIAL ESTIMATE,5X,11HFINAL VALUE00001270
     X,5X,18HSTANDARD DEVIATION,3X,37HINITIAL DERIVATIVE   FINAL DERIVAT00001280
     1IVE  /)                                                            00001290
 2004 FORMAT (5X,A6,3X,1P3E19.5,2E21.5)                                  00001300
 2005 FORMAT(/ 46H INITIAL SUM OF THE WEIGHTED SQUARED RESIDUALS , 1PE15 00001310
     X.5)                                                                00001320
 2007 FORMAT(/ 44H FINAL SUM OF THE WEIGHTED SQUARED RESIDUALS,1PE15.5/)00001330
 2008 FORMAT (1P9E14.5/(70X,1P4E14.5))                                   00001340
 2009 FORMAT (/ 19H CORRELATION MATRIX  / (7X,9(5X,A6,3X)))              00001350
 2010 FORMAT (1H ,A6,1P9E14.5/(7X,1P9E14.5))                             00001360
 2011 FORMAT (17H CHI SQUARE VALUE,1PE15.5,5X,18HDEGREES OF FREEDOM,I10)00001370
      END                                                                00001380
END                                                                      00001390
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  DAVIDON
      COMMON       H(40,40),X(40),G(40),S(40),XP(40),GP(40),T(40),GB(40),00000010
     1              F,GS,EL,SL,FP,GSP,TO,Z,Q,A,GSS,FO,GTP,FB,GTT,GSB,     00000020
     2              FAC,DELTA,E,P,C(40,10),N,K,NC,NSSW1,NSSW2,MS,IT,M1,L 00000030
      COMMON/VMM2/IY                                                      00000040
    1 IY=1                                                                00000050
   15 M1=1                                                                00000060
      MS=0                                                                00000070
  100 IT=0                                                                00000080
      CALL FCN$(N,G,F,X,M1)                                               00000090
      IF (M1.EQ.3) GO TO 1000                                             00000100
      IF (NSSW1 .EQ. 0 ) GO TO 120                                        00000110
      WRITE (6,8)IT,MS,F                                                  00000120
      WRITE (6,10)(X(I),I=1,N)                                            00000130
      WRITE (6,13)(G(I),I=1,N)                                            00000140
  120 IF (NC.EQ.0) GO TO 151                                              00000150
      DO 150 J=1,NC                                                       00000160
      CALL MATMPY (N,N,H,C(1,J),S)                                        00000170
      CALL MATMPY(1,N,S,C(1,J),TO)                                        00000180
      IF (M1.NE. 1 .AND. TO .LE. E) GO TO 150                             00000190
  130 DO 140 II=1,N                                                       00000200
      DO 140 JJ=1,N                                                       00000210
  140 H(II,JJ)=H(II,JJ)-S(II)*S(JJ)/TO                                    00000220
  150 CONTINUE                                                            00000230
  151 CONTINUE                                                            00000240
      M1=2                                                                00000250
  200 CALL READY                                                          00000260
      IF (L-2 ) 570,300,500                                               00000270
  300 CALL AIM                                                            00000280
      IF (L.NE.1) GO TO 500                                               00000290
  400 CALL FIRE                                                           00000300
      IF (L.GT.2) GO TO 300                                               00000310
  500 CALL DRESS                                                          00000320
      GO TO 120                                                           00000330
  570 M1=4                                                                00000340
      CALL FCN$(N,G,F,X,M1)                                               00000350
      M1=2                                                                00000360
  580 CALL STUFF                                                          00000370
      IF (L.EQ.1) GO TO 100                                               00000380
  800 M1=3                                                                00000390
      IF(NSSW1.NE.0)WRITE(6,12)GS                                         00000400
  900 IF(NSSW2.NE.0) WRITE (7,3)(X(I),I=1,N)                              00000410
      CALL FCN$(N,G,F,X,M1)                                               00000420
      GO TO 15                                                            00000430
 1000 REWIND 6                                                            00000440
      RETURN                                                              00000450
    3 FORMAT(6E12.5)                                                      00000460
    8 FORMAT(4H1IT I4,7H  STEP I4,4H  F=E14.5)                            00000470
   10 FORMAT(3H0X=8E14.5/(3H0  8E14.5))                                   00000480
   12 FORMAT(5H  GS=,E14.5)                                               00000490
   13 FORMAT(3H0G=8E14.5/(3H0  8E14.5))                                   00000500
      END                                                                 00000510
      SUBROUTINE READY                                                    00000520
      COMMON       H(40,40),X(40),G(40),S(40),XP(40),GP(40),T(40),GB(40),00000530
     1              F,GS,EL,SL,FP,GSP,TO,Z,Q,A,GSS,FO,GTP,FB,GTT,GSB,     00000540
     2              FAC,DELTA,E,P,C(40,10),N,K,NC,NSSW1,NSSW2,MS,IT,M1,L 00000550
C                                                                         00000560
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  DAVIDON
  200 L=1
      CALL MATMPY(N,N,H,G,S)
      DO 205 I=1,N
  205 S(I)=-S(I)
      CALL MATMPY(1,N,S,G,GS)
      IF (GS+E) 210,240,240
C
  210 L=2
      EL=2.0
      TO=EL*F/GS
      IF (TO+EL) 213, 213, 212
  212 EL=-TO
  213 SL=-GS
      DO 215 I=1,N
  215 XP(I)=X(I)+EL*S(I)
      CALL FCN$(N,GP,FP,XP,M1)
      CALL MATMPY(1,N,S,GP,GSP)
      IF (-GSP) 240,240,220
  220 IF (F-FP) 240,240,225
C
  225 L=3
      IF(NSSW1.NE.0)PRINT  1
      FB=FP
      DO 230 I=1,N
      GB(I)=GP(I)
      T(I)=XP(I)
  230 CONTINUE
      IF (EL-2.0) 240,235,240
C
  235 L=4
      DELTA=DELTA+DELTA
      TO=1.0/SL
C
  240 RETURN
    1 FORMAT(10HOUNDERSHOT)
      END
      SUBROUTINE AIM
      COMMON       H(40,40),X(40),G(40),S(40),XP(40),GP(40),T(40),GB(40),
     1              F,GS,EL,SL,FP,GSP,TO,Z,Q,A,GSS,FO,GTP,FB,GTT,GSB,
     2              FAC,DELTA,E,P,C(40,10),N,K,NC,NSSW1,NSSW2,MS,IT,M1,L
C
  300 L=1
      Z=3.0/EL*(F-FP)+GS+GSP
      Q=ABS (Z*SQRT (1.0-(GS/Z)*(GSP/Z)))
      A=(Q-Z+GSP)/(Q+Q-GS+GSP)
      TO=EL/3.0*(Q+Q+Z+GSP)*A*A
      FO=FP-TO
      CALL MATMPY(N,N,H,GP,T)
      DO 305 I=1,N
  305 T(I)=(GSP/SL)*S(I)-T(I)
      CALL MATMPY(1,N,T,GP,GTP)
      IF (TO+TO+GTP) 315,310,310
  310 DO 312 I=1,N
  312 T(I)=XP(I)+A*(X(I)-XP(I))
      GO TO 340
  315 IF (F+F+GTP) 310,320,320
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  DAVIDON
  320 DO 322 I=1,N
  322 T(I)=T(I)+XP(I)
      CALL FCN$(N,GB,FB,T,M1)
      IF (FO-FB) 310,325,325
C
  325 L=3
      IF (NSSW1.NE.0)PRINT 1
      DO 327 I=1,N
  327 S(I)=T(I)-XP(I)
      CALL MATMPY(1,N,S,GB,GTT)
      GTT=GTT-GTP
      IF (GTT) 340,330,330
C
  330 L=2
      GSS=GTT
      SL=-GTP
      EL=1.0
C
  340 RETURN
    1 FORMAT (9HORICOCHET)
      END
      SUBROUTINE FIRE
      COMMON      H(40,40),X(40),G(40),S(40),XP(40),GP(40),T(40),GB(40),
     1             F,GS,EL,SL,FP,GSP,TO,Z,Q,A,GSS,FO,GTP,FB,GTT,GSB,
     2             FAC,DELTA,E,P,C(40,10),N,K,NC,NSSW1,NSSW2,MS,IT,M1,L
      EQUIVALENCE (TEMP,GTT)
C
  400 L=1
      TEMP=A/(1.0-A)
      CALL FCN$(N,GB,FB,T,M1)
      CALL MATMPY(1,N,S,GB,GSB)
      TO=F
      IF (TO-FP) 403, 403, 402
  402 TO=FP
  403 IF (TO-FB+E) 415, 405, 405
  405 GSS=Q+Q
      TO=GSB*(TEMP-1.0/TEMP)
      IF (ABS (TO)-Q) 430,410,410
C
  410 L=2
      GO TO 440
C
  415 L=3
      IF (FP-F) 425,420,420
C
  420 IF(NSSW1.NE.0) PRINT 1
      EL=(1.0-A)*EL
      FP=FB
      GSP=GSB
      DO 422 I=1,N
      XP(I)=T(I)
      GP(I)=GB(I)
  422 CONTINUE
      GO TO 440
C
  425 IF(NSSW1.NE.0) PRINT 2
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  DAVIDON
      EL=EL*A                                                           00001690
      F=FB                                                              00001700
      GS=GSB                                                            00001710
      DO 427 I=1,N                                                      00001720
      X(I)=T(I)                                                         00001730
      G(I)=GB(I)                                                        00001740
  427 CONTINUE                                                          00001750
      GO TO 440                                                         00001760
C                                                                       00001770
  430 GSS=GSS+TO                                                        00001780
      DO 435 I=1,N                                                      00001790
  435 G(I)=(GB(I)-G(I))*TEMP+(GP(I)-GB(I))/TEMP                         00001800
C                                                                       00001810
  440 RETURN                                                            00001820
    1 FORMAT(10HOMOVE LEFT )                                            00001830
    2 FORMAT (11HOMOVE RIGHT)                                           00001840
      END                                                               00001850
      SUBROUTINE DRESS                                                  00001860
      COMMON       H(40,40),X(40),G(40),S(40),XP(40),GP(40),T(40),GB(40),00001870
     1              F,GS,EL,SL,FP,GSP,TO,Z,Q,A,GSS,FO,GTP,FB,GTT,GSB,   00001880
     2              FAC,DELTA,E,P,C(40,10),N,K,NC,NSSW1,NSSW2,MS,IT,M1,L 00001890
C                                                                       00001900
  500 GO TO (505,520,530,525), L                                        00001910
C                                                                       00001920
  505 CALL MATMPY(N,N,H,G,X)                                            00001930
      CALL MATMPY(1,N,X,G,TO)                                           00001940
      IF (TO-GSS**2/SL-E) 515,510,510                                   00001950
  510 DO 512 II=1,N                                                     00001960
      DO 512 JJ=1,N                                                     00001970
  512 H(II,JJ)=H(II,JJ)-X(II)*X(JJ)/TO                                  00001980
      DELTA=DELTA*(EL*GSS/TO)                                           00001990
      TO=EL/GSS                                                         00002000
      GO TO 525                                                         00002010
C                                                                       00002020
  515 IF(NSSW1.NE.0)PRINT 1                                             00002030
C                                                                       00002040
  520 DELTA=DELTA*(EL*SL/GSS)                                           00002050
      TO=EL/GSS-1.0/SL                                                  00002060
C                                                                       00002070
  525 DO 527 II=1,N                                                     00002080
      DO 527 JJ=1,N                                                     00002090
  527 H(II,JJ)=H(II,JJ)+TO*S(II)*S(JJ)                                  00002100
C                                                                       00002110
  530 IT=IT+1                                                           00002120
      F=FB                                                              00002130
      IF (NSSW1.NE.0) PRINT 4, IT,MS,F,GS                               00002140
      DO 532 I=1,N                                                      00002150
      G(I)=GB(I)                                                        00002160
      X(I)=T(I)                                                         00002170
  532 CONTINUE                                                          00002180
      IF (NSSW1) 535,540,535                                            00002190
  535 PRINT 2, (X(I),I=1,N)                                             00002200
      PRINT 3, DELTA                                                    00002210
C                                                                       00002220
  540 RETURN                                                            00002230
    1 FORMAT(9HOCOLINEAR)                                               00002240
```

LISTINGS FOR TRANSLATOR

```
MEMBER NAME  DAVIDON
    2 FORMAT(3H0X=8E14.5/(3H0  8E14.5))
    3 FORMAT(7H0DELTA=E14.5/20H0- - - - - - - - - -)
    4 FORMAT(4H0IT I4,7H  STEP I4,4H  F=E14.5,5H  GS=E14.5)
      END
      SUBROUTINE STUFF
      COMMON       H(40,40),X(40),G(40),S(40),XP(40),GP(40),T(40),GB(40),
     1              F,GS,EL,SL,FP,GSP,TO,Z,Q,A,GSS,FO,GTP,FB,GTT,GSB,
     2              FAC,DELTA,E,P,C(40,10),N,K,NC,NSSW1,NSSW2,MS,IT,M1,L
      COMMON/VMM2/IY
C
  600 L=2
      K=K-1
      IF (K) 640,610,610
C
  610 L=1
      MS=MS+1
      IF (NSSW1 .NE. 0) WRITE(6,1)MS,DELTA,GS
      DO 620 I=1,N
      CALL RANDU (IY,YFL)
  620 T(I)=YFL-.5
      CALL MATMPY(N,N,H,T,S)
      CALL MATMPY(1,N,S,T,EL)
      EL=P/SQRT (EL)
      DO 630 I=1,N
  630 X(I)=X(I)+EL*S(I)
C
  640 RETURN
    1 FORMAT(13HORANDOM STEP I4,8H  DELTA=E14.5,5H  GS=E14.5)
      END
      SUBROUTINE RANDU(IX,YFL)
      IY=IX*65539
      IF (IY)5,6,6
    5 IY=IY+2147483647+1
    6 YFL=IY
      YFL=YFL*.4656613E-9
      IX=IY
      RETURN
      END
      SUBROUTINE MATMPY(M,N,H,G,S)
      DIMENSION H(40,40),G(40),S(40)
C
  700 DO 720 II=1,M
      S(II)=0.0
      DO 720 JJ=1,N
  720 S(II)=H(JJ,II)*G(JJ)+S(II)
C
  740 RETURN
      END
```

LISTINGS OF CATALOGED PROCEDURES

```
MEMBER NAME  VMMCLG
//VMMTR  EXEC  PGM=IEHPROGM                                          00000010
//PRGG DD DSNAME=C145.B12533.T7120.SYSLMOD(VMMTR),                   C00000020
//             DISP=OLD,UNIT=DISK,VOLUME=SER=ODDJOB                  00000030
//SYSPRINT DD DSNAME=C145.B12533.T7120.VMMTR,                        C00000040
//             DISP=OLD,VOLUME=REF=*.PRGG                            00000050
//SYSIN DD DUMMY                                                     00000060
//GO  EXEC  PGM=*.VMMTR.PRGG                                         00000070
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)        00000080
//SYSPNCH DD UNIT=(SYSCP,,DEFER),DCB=(RECFM=F,BLKSIZE=80)            00000090
//RDFILE    DD DSNAME=C145.B12533.T7120.SOURCE(SKFCN),               C00000100
//             DISP=OLD,VOLUME=REF=*.VMMTR.PRGG                      00000110
//CDFILE    DD DSNAME=C145.B12533.T7120.PDAT,                        C00000120
//             DISP=OLD,VOLUME=REF=*.VMMTR.PRGG                      00000130
//PROG      DD DSNAME=C145.B12533.T7120.FORT,                        C00000140
//             DISP=OLD,VOLUME=REF=*.VMMTR.PRGG                      00000150
//FTH     EXEC PGM=FORTRANH                                          00000160
//SYSPUNCH  DD UNIT=(SYSCP,,DEFER)                                   00000170
//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)         00000180
//SYSLIN DD DSNAME=SYS1.SYSLIN,DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200),  C00000190
//             DISP=(OLD,PASS)                                       00000200
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                           00000210
//SYSIN     DD DSNAME=C145.B12533.T7120.FORT,                        C00000220
//             DISP=OLD,VOLUME=REF=*.VMMTR.PRGG                      00000230
//EDT     EXEC PGM=LINKEDIT,COND=(5,LT,FTH),PARM='LIST,MAP'          00000240
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)        00000250
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                           00000260
//SYSLIN    DD DSNAME=*.FTH.SYSLIN,DISP=OLD                          00000270
//       DD DSNAME=C145.B12533.T7120.EDT,                            C00000280
//             DISP=OLD,VOLUME=REF=*.FTH.SYSIN                       00000290
//          DD DDNAME=SYSIN                                          00000300
//SYSLIB    DD DSNAME=SYS1.FORTLIB,UNIT=2314,VOLUME=SER=SYSLIB,DISP=OLD 00000310
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,4)),DISP=(,PASS),          C00000320
//             DSNAME=&&G(DVDN)                                      00000330
//LIB     DD                    DSNAME=C145.B12533.T7120.SYSLMOD,    C00000340
//             DISP=OLD,VOLUME=REF=*.FTH.SYSIN                       00000350
//RUN     EXEC PGM=*.EDT.SYSLMOD,COND=((5,LT,FTH),(5,LT,EDT))        00000360
//FT05F001    DD DSNAME=C145.B12533.T7120.DATA,                      C00000370
//             DISP=OLD,VOLUME=REF=*.FTH.SYSIN                       00000380
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)        00000390
//FT07F001  DD UNIT=(SYSCP,,DEFER)                                   00000400
//CDFILE    DD DSNAME=C145.B12533.T7120.DATA,                        C00000410
//             DISP=OLD,VOLUME=REF=*.FTH.SYSIN                       00000420
//TEMP      DD DSNAME=C145.B12533.T7120.TDAT,                        C00000430
//             DISP=OLD,VOLUME=REF=*.FTH.SYSIN                       00000440
//XTFILE    DD DSNAME=C145.B12533.T7120.XDAT,                        C00000450
//             DISP=OLD,VOLUME=REF=*.FTH.SYSIN                       00000460
//FT01F001    DD DSNAME=C145.B12533.T7120.XDAT,                      C00000470
//             DISP=OLD,VOLUME=REF=*.FTH.SYSIN                       00000480
//SYSPRINT    DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)      00000490
//SYSIN    DD DSNAME=C145.B12533.T7120.PDAT,                         C00000500
//             DISP=OLD,VOLUME=REF=*.FTH.SYSIN                       00000510
```

LISTINGS OF CATALOGED PROCEDURES

```
MEMBER NAME  VMMLG
//EDT      EXEC PGM=LINKEDIT,PARM='LIST,MAP'                                00000010
//LIB      DD                    DSNAME=C145.B12533.T7120.SYSLMOD,         C00000020
//              DISP=OLD,UNIT=DISK,VOLUME=(PRIVATE,RETAIN,SER=ODDJOB)       00000030
//SYSPRINT  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=726)               00000040
//SYSUT1    DD DSNAME=SYS1.SYSUT1,DISP=OLD                                  00000050
//SYSLIB    DD DSNAME=SYS1.FORTLIB,UNIT=2314,VOLUME=SER=SYSLIB,DISP=OLD 00000060
//SYSLMOD   DD UNIT=DISK,SPACE=(TRK,(99,5,1)),DISP=(,PASS),               C00000070
//              DSNAME=&&G(G)                                               00000080
//SYSLIN DD DSNAME=C145.B12533.T7120.EDT,                                   C00000090
//              DISP=OLD,VOLUME=(PRIVATE,RETAIN,REF=*.LIB)                  00000100
//           DD DDNAME=SYSIN                                                00000110
//GO       EXEC PGM=*.EDT.SYSLMOD,COND=(5,LT,EDT)                           00000120
//FT05F001    DD DSNAME=C145.B12533.T7120.DATA,                            C00000130
//              DISP=OLD,VOLUME=(PRIVATE,RETAIN,REF=*.EDT.LIB)              00000140
//FT06F001  DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)               00000150
//FT07F001  DD UNIT=(SYSCP,,DEFER)                                          00000160
//CDFILE  DD    DSNAME=C145.B12533.T7120.DATA,                              C00000170
//              DISP=OLD,VOLUME=(PRIVATE,RETAIN,REF=*.EDT.LIB)              00000180
//TEMP      DD DSNAME=C145.B12533.T7120.TDAT,                               C00000190
//              DISP=OLD,VOLUME=(PRIVATE,RETAIN,REF=*.EDT.LIB)              00000200
//XTFILE    DD DSNAME=C145.B12533.T7120.XDAT,                               C00000210
//              DISP=OLD,VOLUME=(PRIVATE,RETAIN,REF=*.EDT.LIB)              00000220
//FT01F001    DD DSNAME=C145.B12533.T7120.XDAT,                             C00000230
//              DISP=OLD,VOLUME=(PRIVATE,RETAIN,REF=*.EDT.LIB)              00000240
//SYSPRINT    DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=798)             00000250
```

ACKNOWLEDGMENTS

I wish to thank Janet Anderson for writing the precoded FORTRAN skeleton and making many tests of the system, and J. R. Gabriel for giving much helpful advice during the writing of this translator.

REFERENCES

1. Davidon, William C., Variable Metric Method for Minimization, ANL-5990 (Rev. 2), Feb 1966.

2. Garbow, Burton S., Variable Metric Minimization, Internal Report ANL-Z013, Aug. 9, 1965.

3. Gabriel, J. R., and Gabriel, M., Chemical Equation Translator, Program 08T7019, Sept. 25, 1967.

4. Anderson, J. H., documentation of FCN$ still in progress.

ARGONNE NATIONAL LAB WEST
3 4444 00011440 5