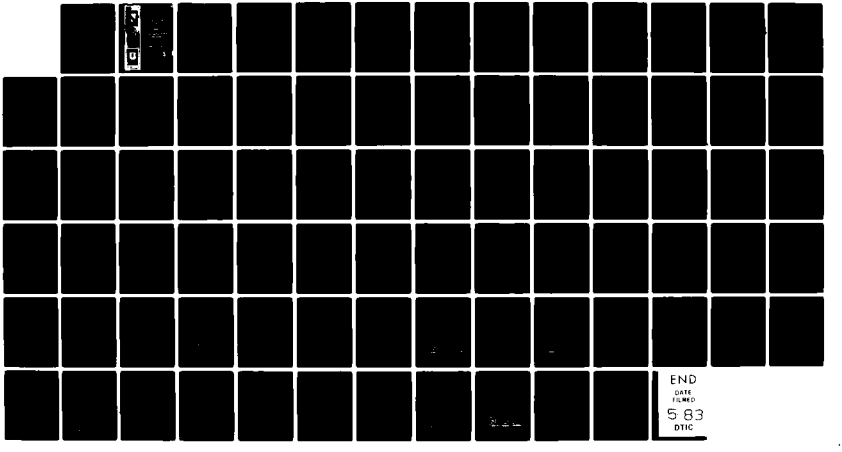
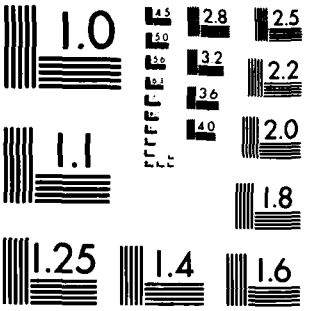
AD-A126 674 PHD: A DATA HANDLING PROGRAM IN DEC BASIC(U) FRANK J 1/1
SEILER RESEARCH LAB UNITED STATES AIR FORCE ACADEMY CO
A A FANNIN FEB 83 FJSRL-TR-83-0002

UNCLASSIFIED F/G 9/2 NL



END
DATE FILMED
5 83
DTIC



FRANK J. SEILER RESEARCH LABORATORY

FJSRL-TR-83-0002

FEBRUARY 1983

PHD:

A Data Handling Program in DEC BASIC

Armand A. Fannin, Jr.

PROJECT 2303

AIR FORCE SYSTEMS COMMAND

UNITED STATES AIR FORCE

FJSRL-TR-83-0002

This document was prepared by the Electrochemistry Division, Directorate of Chemical Sciences, Frank J. Seiler Research Laboratory, United States Air Force Academy, CO. The research was conducted under Project Work Unit number 2303-F2-10. Lt Colonel Armand A. Fannin, Jr. was the project scientist.

When U.S. Government drawings, specifications or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the government may have formulated, furnished or in any way supplied the said drawings, specifications or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation or conveying any rights or permission to manufacture, use or sell any patented invention that may in any way be related thereto.

Inquiries concerning the technical content of this document should be addressed to the Frank J. Seiler Research Laboratory (AFSC), FJSRL/NC, USAF Academy, CO 80840. Phone AC 303 472-2655.

This report has been reviewed by the Commander and is releasable to the National Technical Information Service (NTIS). At NTIS it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

Armand A. Fannin, Jr, Lt Col
Project Scientist

Armand A. Fannin, Jr, Lt Col
Director, Chemical Sciences

William D. Siuru, Jr., Colonel
Commander

Copies of this report should not be returned unless return is required by security considerations, contractural obligations, or notice on a specific document.

Printed in the United States of America. Qualified requestors may obtain additional copies from the Defense Documentation Center. All others should apply to:

National Technical Information Service
6285 Port Royal Road
Springfield, Virginia 22161

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER FJSRL-TR-83-0002	2. GOVT ACCESSION NO. ADA 126 67	3. RECIPIENT'S CATALOG NUMBER 4
4. TITLE (and Subtitle) PHD: A Data Handling Program in DEC Basic		5. TYPE OF REPORT & PERIOD COVERED
		6. PERFORMING ORG. REPORT NUMBER FJSRL-TR-83-0002
7. AUTHOR(s) ARMAND A. FANNIN, JR.		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2303-F2-10
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE February 1983
		13. NUMBER OF PAGES 71
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release, Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

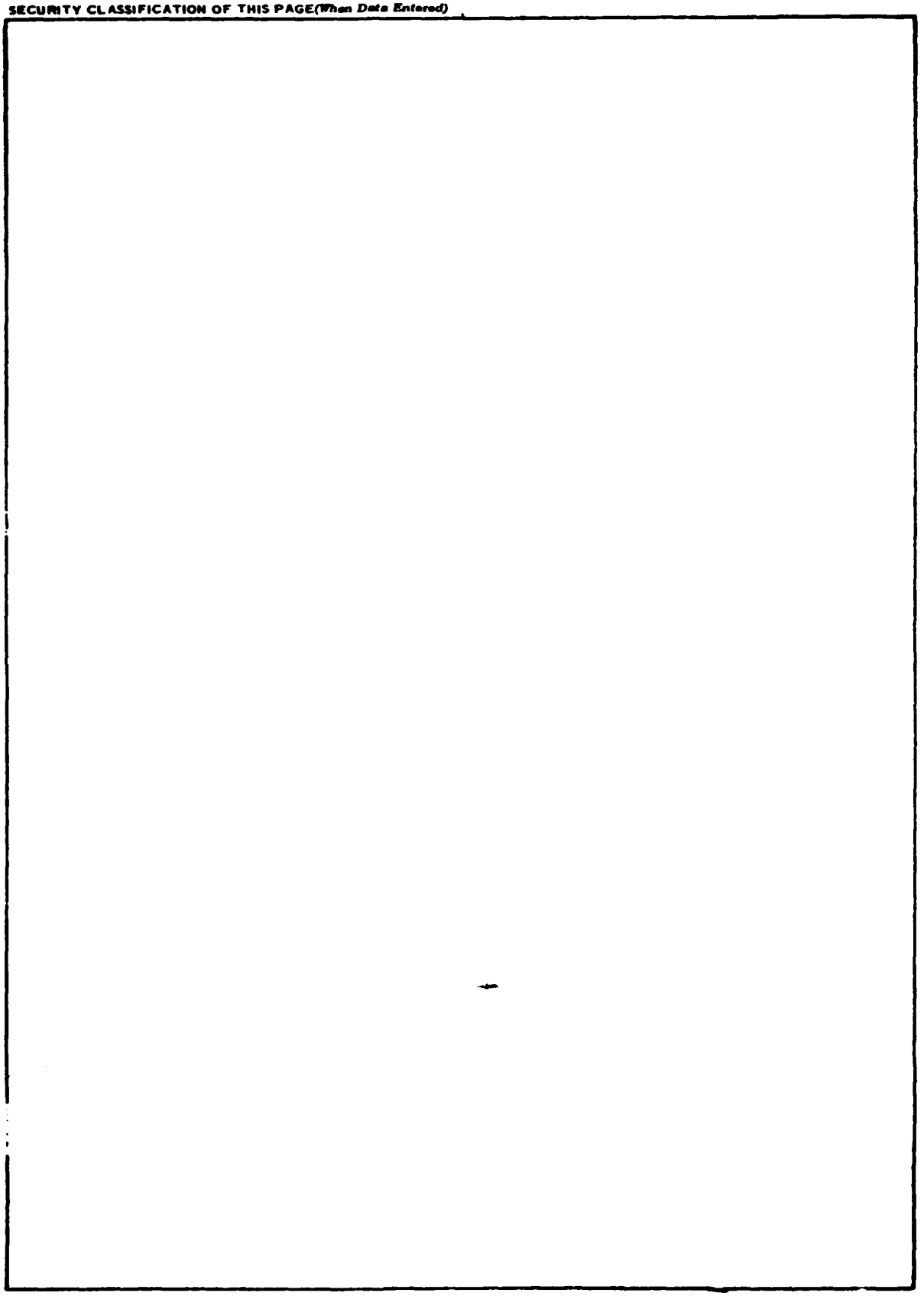
Data base
Least Squares
Data Reduction

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

PHD is a program for storage, display, testing and manipulation of small numeric data collections (2 to 500 observations). The implementation is on a VAX 11/780 in BASIC. Modules allow entry, correction, storage, retrieval, sorting, least squares fitting and plotting. One may also add user written modules to manipulate sets of data. The graphics module given requires a Tektronix 4014 or compatible terminal. Program listing included.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED



FJSRL-TR-83-0002

PHD: A Data Handling Program in DEC BASIC

Armand A. Fannin, Jr.

February 1983

Directorate of Chemical Sciences
The Frank J. Seiler Research Laboratory
Air Force Systems Command
United States Air Force Academy
Colorado Springs, Colorado 80840

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont'd)

INTRODUCTION

Background

In the course of many physical measurements, a relatively small collection of raw data is used in a series of calculations, transformations, graphs and data reduction routines until the final result consisting of a parameterized equation, a graph or a table is produced. In the course of the calculations it is often very desireable to use the same operation, or series of operations, on each set of observations and similar calculations are often performed on unrelated types of data. A careful investigator often wants to list and check intermediate data, or perhaps examine it using some testing algorithm, perform some systematic corrections, and keep the data in some safe and convenient form until his results have been confirmed by the scrutiny of others.

Before the advent of the computer, many of the calculations involving minor corrections or repetitive measurements were simply not done. A minimal amount of data was taken and processed. There were frequently questions as to whether all observations had been treated consistently during the reduction process, and much time and effort was often spent in investigating discrepancies that resulted from calculational error during data reduction. Even with the aid of the computer, calculations were often done piecemeal in a series of programs, or a special program to perform a series of calculations was laboriously built for each individual experiment.

The concept of a database management system has allowed the investigators in this laboratory to collect and process data systematically, and with greater confidence in calculational consistency, since early 1977. This

database management system, called PHD (Program for Handling Data), has been developed to support the collection of physical measurement data. This type of data usually consists of a small number of observations (2-100), each observation consisting of a set of measured variables and their uncertainties in some specified set of units. For convenience, an observation identifier may be included. PHD was developed to support this type of data collection.

PHD consists of a series of programs written primarily in BASIC which allow the manipulation of data mathematically. In addition, it provides visual aids such as listings and graphics as well as the facility to add, delete, correct, sort, merge, rearrange, store, retrieve and perform multivariate non-linear regression on collections of data stored as files under the host computer operating system (RSX11-M on the PDP 11/45 or 11/10, MINC BASIC on the MINC, and VAX/VMS on the VAX 11/780). The system is designed to allow users to add their own manipulation programs to those available in the system. The following document describes the capabilities of the software available in the system and how to use them; it also provides sufficient detail to allow users to write routines for additional processing of data while preserving the advantages of the facilities available to manage data sets within the system. While some of the programs are peculiar to the type of hardware being used--the graphics routines, for example--the algorithms used are adaptable to many systems with, as the saying goes, only minor changes in the coding.

PHD is a modular system; the main program defines the internal data storage system and then passes control to a selector or menu program, MENU. MENU calls each module requested, including user written programs, and each of the modules it calls normally exits back to the MENU. The description of the

data contained within the database (called the BASE) is maintained in main memory along with some ancillary storage used in data manipulation. The data in the database are maintained in either main memory or a virtual array which is stored primarily on a random access mass storage device (e.g., a disk). When necessary the MENU cleans up the disk by deleting the virtual array or the temporary transfer files from the mass storage device when they are no longer needed.

Because of the inability to trap errors in some versions of BASIC, occurrence of errors often results in what can easily be a catastrophic loss of the data currently being used. The system has been designed to compensate for this by exposing only the data in the current WORKSET to loss from an untrappable error, and generally recovery of data after such an error is a simple process.

Acknowledgements

A number of people have contributed to the conception and testing of this program and its various modules. I would like to express my thanks to all. Deserving special thanks for their involvement in this process are Dr Lowell A. King and Dr John S. Wilkes. Lt Robert E. Wheelock concieved the algorithm and coded the initial version of the plotting module in 1978. The bulk of this module remains intact in the current version. I also would like to thank Lt Col Chester J. Dymek for very helpful discussions and suggestions on the documentation.

Concept

Each set of data (DATASET) in the PHD system may be thought of as a two-dimensional array of data with each observation occupying a row. Therefore the number of rows in the array is equal to the number of observations. Associated with each observation (or with each row in the array) is an observation identifier, composed of two parts, a plot symbol number (PSN) and an alphanumeric identifier (ID). The columns of the array correspond to an ordered set of variables measured for each observation.

Each column has associated with it a NAME for the variable, the UNIT in which the data is measured, and a value related to the uncertainty with which this variable is measured, the METRIC. In addition each DATASET has a one-line COMMENT stored with it. The first part of this COMMENT is generated by the system and consists of the date and time at which the DATASET was SAVEd (i.e., made a permanent file on the mass storage device). After it is SAVEd, a DATASET may not itself be modified. As a safeguard against accidental loss of data, it resides unmodified on the mass storage device until explicitly deleted. Only the current or working dataset (WORKSET), which exists only during the active operation of the PHD program, may be manipulated and changed. If a new DATASET is created with the same name as one already on mass storage, it is stored as a new version of the file in systems where the operating system allows this (caution must be exercised when using multiple versions as a purge of the data files may delete all but the last version of a DATASET). In operating systems where multiple versions are not allowed, it replaces the old version.

Most of the modules of PHD are used to define the dataset structure (the BASE), or manipulate the data in it. In figure 1, on the following page, is a typical DATASET. The underlined text is not part of the dataset but explanatory material.

DATASET NAME: PRESUR (The file name would be PRESUR.DAT;n--n is the version)

COMMENT(one-line): 7-JAN-81/14:02:03!VAPOR PRESSURE FOR PERWAXY CHLORIDE

Data matrix:

PSN:ID	Pressure [Torr]	Temperature [Kelvin]	← Variable NAME ← UNITs of variable
	0.1	.001	← METRIC of variable
2:Pnt 1	34.52	279.333	← Observations follow
2:Pnt 2	16.6	265.24	.
.	.	.	.
.	.	.	.
.	.	.	.
1:Pnt 99	295.33	456.899	← Last observation

Figure 1. Sample Data Set

In the example DATASET given above there are two variables stored for each observation. The operation of PHD will be explained in terms of entering the DATASET given above and performing various manipulations on it. A summary of the various modules and the information (arguments) they may require or optionally take is given at the end of this report.

Hardware requirements

A computer capable of running DEC's BASIC-11 under RSX-11M or VAX/VMS native mode BASIC under VMS is necessary to run PHD without modification. A mass storage device capable of storing the DATASET produced and sufficient main memory or virtual array capability as defined for BASIC-11 is also required. The Tektronix 4014 terminal must be used to take advantage of all the features of PHD. If a different terminal is used, spurious output of 8's,

9's, :'s or ;'s will occur whenever a character size change is requested by PHD. The .PLT files generated by PHD under RSX-11M or the plots produced on the VAX may only be plotted on the Tektronix 4014 or a terminal compatible with it.

OPERATION

Initiating PHD

Calling the PHD program varies with operating system of the host computer. Under DEC's RSX11-M or on the MINC, PHD is called as any other BASIC program. On the VAX it is most efficiently called as a program under the DCL monitor. Familiarity with the computer system sufficient to sign on and, if necessary, get into BASIC is assumed.

On the PDP 11/10 or 11/45 a one statement BASIC program written exactly as follows is necessary to initiate PHD. Assuming the programs for PHD are stored and accessible under the [142,1] user code the program would be entered as follows:

```
NEW PHD□                          (See note below)
10 CHAIN "[142,1]PHD" LINE 10□
SAVE□
```

Once such a program is written and saved, PHD may be initiated at any time while in BASIC by entering the following:

```
RUN PHD□
```

On the VAX, the PHD program is used in a compiled form normally. The program used in the Chemistry Directorate (NC) is stored in the directory [SEILERV.PHD] as PHD.EXE. It can be run from any other directory by using the full file specification in the command:

```
RUN [SEILERV.PHD]PHD
```

Note: Here and in the rest of this report, the symbol □ will be used to represent a carriage return.

By using the DCL global definition,

PHD:==RUN [SEILERV.PHD]PHD

the program may be initiated simply by typing in PHD↵ (in all NC directories the global definition is made in the login command file).

From the initiation of the PHD program until the user exits from it, the operations appear the same to the user on any system. To continue the illustration of using PHD, let us assume the user proceeds by one of the above methods to initiate PHD.

The response from PHD will be to go to a new page (clear the screen) and present the MENU. When the MENU is presented, at least two letters of one of the options must be typed in, followed by any argument the option requires or will accept (at least one blank must separate the option from the argument). For example, if the dataset of figure 1 is to be created, the option specified would be BASE, and the argument to indicate a new dataset is NEW. The command line would be as follows:

BASE NEW↵

PHD responds by deleting the current WORKSET and its BASE definition. It then requests the NAME for the new WORKSET and a new COMMENT line. New VARIABLE definitions may then be inserted into the BASE, which is initially null (i.e., no variables are defined). A VARIABLE is added by specifying its NAME,UNIT,METRIC with commas between. A null string may be used for any of the three fields in the entry. The null string is interpreted to mean do not change from the current value. For new VARIABLEs the initial values are the null string for NAME and UNIT and a zero for the METRIC. A zero METRIC means the measurements are to be considered exact. For the example DATASET the following entry would be typical: (Computer responses are underlined.)

```
WORKSET?  PRESUR□

COMMENT?  VAPOR PRESSURE FOR PERWAXY CHLORIDE□

THE CURRENT BASE HAS 0 VARIABLES PER OBSERVATION.

NAME,UNIT,METRIC FOR ADDITIONAL VARIABLE ? PRESSURE,TORR,.1□

THE CURRENT BASE HAS 1 VARIABLES PER OBSERVATION.

PRESSURE[TORR]+/-.1

NAME,UNIT,METRIC FOR ADDITIONAL VARIABLE ? TEMPERATURE,K,.001□

THE CURRENT BASE HAS 2 VARIABLES PER OBSERVATION.
PRESSURE[TORR]+/-.1          TEMPERATURE[K]+/-.001

NAME,UNIT,METRIC FOR ADDITIONAL VARIABLE ? □
```

When the BASE has been defined, one may enter the observations using the option, ADD. The following entry provides an example:

```
ADD□

ENTERING 'ERROR' WILL CAUSE REENTRY OF LAST OBSERVATION STARTED.
NEXT OBSERVATION WILL BE NUMBER 1

OBSERVATION ID OR 'END'?2:PNT1□
PRESSURE[TORR]? 34.52□
TEMPERATURE[K]? 279.333□

NEXT OBSERVATION WILL BE NUMBER 2

OBSERVATION ID OR 'END'?
```

Succeeding observations would be entered in the same manner. When all observations have been entered, the response END□ is given in response to the next observation ID request.

```
NEXT OBSERVATION WILL BE NUMBER n

OBSERVATION ID OR 'END'? END□
```

This entry will terminate the ADD module and return control to the MENU. Usually at this point it is prudent to safeguard the data by SAVEing it, or creating a DATASET on the disk. The response to the MENU would be as follows:

OPTION ? SAVE□

CURRENT WORKSET NAME: <workset name>
NEW DATASET NAME (cr TO USE WORKSET NAME)? <datasetname>

The WORKSET would then be stored on mass storage under the <datasetname>.

As can be seen from the above detailed examples, the operation of the various modules involves a conversational interaction with the user. For example, the response required by the SAVE module is for a DATASET name to store the data under. As the program takes some action, it responds with informational replies as to what it is doing. When the DATASET has been created on the disk, control returns to the MENU.

After entering and storing the data in a DATASET, a listing could be made to check for errors. The option LIST will present the DATASET name, COMMENT, and data matrix listing on the terminal much as it is given in figure 1. If errors are present, the CORRECT module may be used to make changes. The data may then be sorted using the SORT option, PRINTed on the line printer, FITted, PLOTted or otherwise manipulated using the USE option to call programs which have been written by the user in BASIC.

On the following pages are details on the use of each of the options available. Two lists of the BASIC variables used internally in PHD follow, one for the RSX-11M version, the other for the user accessible BASIC variables in the VAX/VMS version. Normally the BASIC variables used by PHD should be treated as read only variables. If it is necessary to change the values of some of these variables, it should be done with extreme care to avoid disastrous results in processing data in a user module.

Two short descriptions of the files associated with particularly useful PHD options, USE and FIT are then given. The first treats the writing of a

user option program module and the second the parameter description files used in fitting data to a generalized polynomial.

Finally, listings of PHD, the USER envelope program and an example of its use are included. A summary of the PHD options, with their arguments completes the report.

Detailed Operation Descriptions

Option: ADD

Syntax: ADD
where no arguments are allowed.

Description of results:

Upon entering this module a check is made to see if a WORKSET BASE has been defined (i.e., if the number of variables defined, N%, is greater than zero). If it has not, the user is advised to use the modules BASE or GET to define the BASE and an exit is made to the MENU. If a data base has been defined, the following advisory is given:

ENTERING 'ERROR' WILL CAUSE REENTRY OF LAST OBSERVATION STARTED.
NEXT OBSERVATION WILL BE NUMBER n

OBSERVATION ID OR 'END' ?

One of the following valid entries is made at this point:

□	which is the same as entering 1:□
n:□	where n is a valid PSN (plot symbol number), see PLOT option for valid numbers. No check is made of the validity at this point.
:s□	where s is any string of six or fewer characters (comma is not an allowed character).
n:s□	where s and n are as defined above and the additional restriction that the total number of characters in the identifier cannot exceed eight, including the colon.
ERROR□	this entry will cause the previous observation to be deleted and re-entry of that observation begun.
END□	this entry will terminate the ADD option and return to the MENU.

After the identifier has been entered, the module will request each of the variables defined in the data base, by NAME, in the appropriate UNIT, as follows:

<name>[<unit>]?

Only three possible entries are valid here:

END□	this entry will cause an EXIT from the ADD module and the partially entered data observation will not be included in the dataset.
ERROR□	this entry will cause the current observation being entered to be discarded and re-entry begun.

Option: ADD (cont'd)

<value>□	where <value> is a real number acceptable to BASIC, a positive or negative number in the range 10^{-38} to 10^{38}. Any number of significant digits may be entered, but only 6 are retained accurately.

Each variable will be requested in the same manner until all are obtained. No more than 1000 items (number of variables times number of observations) may be entered without modification of PHD.

Possible errors and recovery options:

An erroneous value may be re-entered by using the ERROR entry or by changing it later with the CORRECT option. Any non-numeric entry for a <value>, other than ERROR or END, may in some versions result in an abnormal exit from PHD and the loss of the current WORKSET.

Date of last module update: 23 Mar 82

Option: BASE

Syntax 1) BASE
2) BASE NEW
3) BASE <dataset>
where <dataset> is the name of a valid PHD dataset.

Description of results:

For the first syntax, the module BASE will be called and the current WORKSET BASE will be displayed as follows:

THE CURRENT BASE HAS n VARIABLES PER OBSERVATION.
<name 1>[<unit 1>]+/- <metric 1> <name 2>[<unit 2>]+/-<metric 2>
ADD|DELETE|CHANGE NAME|UNITS|METRIC OF A VARIABLE ?

where <name i>, <unit i> and <metric i> are the NAME, UNIT, and METRIC of VARIABLE i. There are four valid entries at this point:

- □ this entry will cause a return to the MENU with no further changes in the BASE.
- A□ this entry will result in a request for a new VARIABLE definition of NAME,UNIT,METRIC. A null entry is valid for any of these fields (successive commas indicate a leading null string). If there is more than one observation in the workset, PHD will repack the data at this point giving the advisory message: REPACKING DATA (STRETCH).
- D□ this entry will cause the request of the NAME of the variable to delete from the workset. This name must be in the current BASE definition. If there is more than one observation in the workset, PHD will repack the data at this point giving the advisory message: REPACKING DATA (SQUEEZE).
- C□ this entry will locate the VARIABLE definition in the current BASE and result in a request for a new variable definition of NAME,UNIT,METRIC. A null entry is valid for any of these fields (successive commas indicate a leading null string), and will leave that value unchanged.

The second syntax results in a discarding of the current WORKSET (equivalent to a DELETE BASE option) and a request for new VARIABLE definitions of NAME,UNIT,METRIC (See the A response in syntax 1). Additional VARIABLEs may be defined up to sixteen. A null response will cause return to the MENU.

Option: BASE (cont'd)

The third syntax results in the opening of the DATASET stored under the name given and the construction of a BASE identical to the one specified in that DATASET.

Possible errors and recovery options:

Most errors may be corrected by using the C response of syntax 1 above.

Date of last module update: 23 Mar 82

Option: CM (COMMENT Modification)

Syntax: CM <optional editing character><comment>
where <comment> is any character string which does not include a carriage return, and <optional editing character> and its possible meanings are as follows:

1) > meaning replace the rightmost portion of the current COMMENT with the following string. The replacement is made on a character for character basis.
2) < meaning replace the leftmost portion of the current COMMENT with the following string.
3) - meaning add the following string preceding the current COMMENT.
4) + meaning add the following string to the end of the current COMMENT.

Description of results:

The first character of the argument is examined, if it is an editing character the current COMMENT is modified as specified by that editing character. If no editing character is found, the <comment> becomes the new COMMENT and the old COMMENT is discarded.

Possible errors and recovery options: None

Date of last module update: 27 Dec 82

Option: CORRECT

Syntax: CORRECT
where no arguments are allowed.

Description of results:

PHD will ask for the sequence number of the observation to be corrected:

SEQUENCE NUMBER OF OBSERVATION TO CORRECT ? n□

where n is the sequence numbe. (obtained from the LIST or PRINT options). If a null entry (carriage return only) is given here, an exit is made to the main MENU.

Having received the sequence number of an observation, PHD checks it to make sure it is in range (see possible errors). The observation is then presented on the terminal and a request for corrections made.

'PSN:ID' OR 'VARIABLE NAME,VALUE' ?

There are several possible entries at this point:

□	Return to get new sequence number of observation to be corrected.
<psn>:□	This will result in only the plot symbol number being changed.
:<id>□	This results in only the identifier being changed.
<psn>:<id>□	This changes both the plot symbol number and the identifier. Any of the options containing a colon (:) will result in the message NEW ID IS <psn>:<id> and a request for further corrections for the same observation.
<variable name>,<value>□	This results in the value of the variable named being changed to <value> for the selected observation. After the change has been made, requests for further changes to this observation will be made.

Possible errors and recovery options:

The presence of the colon (:) in the input indicates that entry is a change to either the plot symbol number or identifier, depending on the location of the colon. No check is made on the validity of the plot symbol number or the identifier. The new plot symbol number and identifier are displayed after the change. Note that the composite string <psn>:<id> is truncated after eight characters with no warning.

Option: CORRECT (Cont'd)

If the input does not correspond to any of the above syntaxes, then UNACCEPTABLE INPUT is printed and the program returns to the request for a sequence number.

In the last syntax, if a variable name is referenced that does not exist in the current BASE, a warning is printed, NO VARIABLE <variable name> FOUND, no changes are made to the observation, and the next correction for that observation is requested.

Date of last module update: 27 Dec 82

Option: DELETE

Syntax: 1) DELETE
2) DELETE <sequence set>
where <sequence set> is a set of <sequence range>s separated by commas (,).
<sequence range> is <sequence number> or <start sequence number>- or <start sequence number>-<end sequence number> or -<end sequence number>

Description of results:

If no <sequence set> is specified, as in syntax 1, then PHD will request one with the message:

SEQUENCE NUMBER(S) OF OBSERVATION(S) TO DELETE ?

PHD will scan the <sequence set> entered and print the message:

MARKED FOR DELETION
ID variablename 1[unit] variablename 2[unit] etc.
(A listing of the observations marked will follow this header message.)

The request for a <sequence set> given above, will also occur each time after all observations in a <sequence set> have been marked for deletion. Another sequence set may then be entered or a null entry may be made in order to return to the MENU. When a return to the menu is made, PHD deletes all the observations that have been marked for deletion and resequences the observations that are left, issuing the message n OBSERVATION(S) DELETED; WORKSET RESEQUENCED. on exit to the MENU.

Possible errors and recovery options:

1) If a <sequence range> is illegal, (i.e., one in which <starting sequence number> is larger than <end sequence number>) then the message ILLEGAL SEQUENCE SET is printed, no observations are marked for deletion and the scan of the <sequence set> continues.

2) If a <sequence range> extends beyond the current set of data sequence number, i.e., less than 1 or greater than the number of observations, then the message SEQUENCE RANGE IS FROM 1 TO n where n is the sequence number of the last observation.

3) If a observation already marked for deletion occurs in an another <sequence range> then the message OBSERVATION # n ALREADY MARKED FOR DELETION will be issued. The observation will remain marked for deletion.

Date of last module update: 27 Dec 82

Option: ENTER

Syntax: ENTER
where no arguments are allowed.

Description of results:

The current WORKSET BASE description will be printed in the format of the following example:

SEQ NR	NAME	UNIT	METRIC
0	ID		
1	TEMP	DEG C	0
2	SP COND	MHO/CM	.0002
.			
.			
.			

A request for which VARIABLE is to be entered into the WORKSET will be made. In the following example the VARIABLE "SP COND" is to be entered.

```
NUMBER OF VARIABLE TO ENTER ? 2□
```

An advisory message and request for change for the specified variable in each observation will be made as in the following example:

```
ENTER NEW VALUES FOR SP COND
1        2:pt 1    .59687    NEW VALUE (CR FOR NO CHANGE)> ? .59684□
2        2:pt 2    .47859    NEW VALUE (CR FOR NO CHANGE)> ? □
.
.
.
```

After the last value of the VARIABLE is entered, a return will be made to the main MENU.

A different format is used if the ID is to be changed. The initial display after entering the value 0 in response to the number of the variable to enter is as follows:

```
ENTER NEW ID'S
IF ALL NEW ID'S ARE TO BE ALIKE, ENTER NEW ID HERE (OTHERWISE CR)> ?
```

If any of the ID's are to be different, then all of them must be entered individually (or left as they are), unless the user wishes to change them all and then make selective changes by using ENTER again. If a non-null entry is made at this point, then all ID's are changed to that value and the advisory message DONE is printed and an exit is made to the main MENU. If a null entry is made here then the changes made are much the same as for numerical values. The display of the individual observation ID's for change appears as:

Option: ENTER (Cont'd)

OBSERVATION n: PRESENT PSN:ID IS m:iiiiii; ENTER NEW PSN:ID (CR FOR NO CHANGE)? 8:NEWID□

where n is the observation sequence number and m:iiiiii is the current <psn>:<id> for observation n. In the example, the new PSN and ID would be 8 and NEWID, respectively.

Possible errors and recovery options:

Please note in changing ID's, that both the PSN and the ID must be entered separated by a colon (:). If the changed PSN:ID string does not have a number in the range 1-12 followed by a colon as its initial characters, erroneous results may be obtained in SAVEing the data and in the PLOT module as well as other modules which depend on this specific format for the PSN:ID string. Note also that the total length of the PSN:ID string including the colon must be less than 9 characters. If the string is changed to one that is longer, it will be truncated after the first eight characters with no warning message.

If a VARIABLE sequence number that is out of the range for the current BASE, no warning message is given; instead, the BASE description is redisplayed and another request for the number of the VARIABLE to enter will be made.

If an attempt is made to replace an observation value with a character string that is not a valid BASIC number, ILLEGAL NUMBER will be displayed, and a new request for a change in that observation made.

Date of last module update: 27 Dec 82

Option: EXIT

Syntax: 1)EXIT
2) EXIT <program file name>
where <program file name> is the name of a valid .EXE file to which control is to be transferred on exit from PHD.

Description of results:

PHD will exit to the program file if specified (syntax 2) or back to the operating system or BASIC, whichever it was called from (syntax 1). **Note that on leaving PHD all data currently in the WORKSET is lost. If the data has not previously been SAVEd in a permanent DATASET, it should be saved before exiting by using the SAVE option.**

Possible errors and recovery options: None.

Date of last module update: 27 Dec 82

Option: FIT

Syntax: 1) FIT
2) FIT <parameter file name>
where <parameter file name> is a name for a file that provides equation and parameter information to fit to the observations in the current WORKSET. The file extension must be .PAR (see Parameter Files for Data Fitting, p 38)

Description of results:

If syntax 1 is used, PHD will ask for the parameter file name by requesting:

```
NAME OF PARAMETER FILE ? .
```

After the parameter file is entered results are the same as for syntax 2.

With syntax 2, PHD will read the parameter file specified. The banner (fit title) is printed as it is read and an indication of end-of-file is given when the end of the file is reached. The parameters specified are then sorted based on their DECODE symbol. Diagnostics are printed as the parameters are checked. A typical output stream follows:

```
OPTION? FIT IRATE
LINEAR FIT OF % YIELD TO TIME (INIT. RATE)
**EOF**

3 PARAMETERS.  2 ADJUSTABLE.
SORTING...

PARAMETER TABLE
```

NR	NAME	UNITS	EQ DECODE	INIT VALUE	ADJ
1	INT	[%]	<+00>	0	Y
2	IRATE	[%PROD/MIN]	<+10>	0	Y
3	%PROD	[%]	<-01>	-1	N

After the parameters are checked, two inputs are requested: A new banner and a criterion for fit convergence. Default values appear in brackets in the prompts.

```
BEGIN FIT:
TITLE:  [LINEAR FIT OF % YIELD TO TIME (INIT. RATE)]? %YIELD FOR ACETONE□
DELTA SIGMA FOR CONVERGENCE [1E-7]?□
```

The current banner is displayed (taken from the parameter file) and if a different one is desired, it should be entered after the question mark. A null entry will retain the current banner. The delta sigma requested is the upper bound for the fractional change in standard deviation which will terminate the fitting process. Any fractional change in the standard

Option: FIT (Cont'd)

deviation less than the specified delta sigma will satisfy the criterion for convergence. During the fitting process changes in the adjustable parameters, sigma and delta sigma are printed after each iteration.

```
ITERATION 1 SIGMA = 131.789
7  OBSERVATIONS USED.  DELTA SIGMA = 0

ADJUSTABLE PARAMETERS CORRECTED VALUE
INT = 132.385     IRATE -2.31018
ITERATION 2 SIGMA = .125667
7  OBSERVATIONS USED.  DELTA SIGMA = .983

ADJUSTABLE PARAMETERS CORRECTED VALUE
INT = 132.385     IRATE -2.31
ITERATION 2 CONVERGENCE OCCURRED.
HIT CARRIAGE RETURN TO CONTINUE.      ?
```

After the indication that the fitting process has converged, a pause is made to await a carriage return from the user. On recieving this carriage return, the screen will be cleared and the name of the dataset fitted, the banner, sigma (the standard deviation) and delta sigma and a summary of the parameters and their uncertainty is given. Continuing with the above example:

```
CPY26BF
%YIELD FOR ACETONE

SIGMA = .125667      DELTA SIGMA = -1.99E-8
```

PARAMETER	UNITS	VALUE AND UNCERTAINTY
INT	[%]	(+/-) 8.5E-02
IRATE	[%PROD/MIN]	(+/-) 3.2E-01
%PROD	[%]	NOT ADJUSTABLE

```
FIT TERMINATION--ENTER cr FOR MAIN MENU
?
```

On display of the FIT TERMINATION message the program will wait for a carriage return before exiting to the main MENU.

Possible errors and recovery options:

If the specified parameter file does not exist the message "NO SUCH FILE" will be given and a the module reinitiated.

There is a maximum of 16 parameters (adjustable and non-adjustable combined). If an attempt is made to enter more than that, the message MAXIMUM PARAMETERS EXCEEDED will be printed and an attempt to fit will be made with the parameters already specified.

Option: FIT (Cont'd)

During the parameter checking phase a number of different messages may be given:

If no decode string is given for a parameter, HAS NULL DECODE is printed and a DECODE of <-0> is assumed (ASSUMING DECODE <-0>).

If the DECODE does not specify the exact number of VARIABLEs in the current WORKSET, then one of two messages is given:

1) if it specifies fewer than the number of VARIABLEs in the WORKSET then ACCESSES VARIABLES n-m is printed.

2) if it specifies more than the number of VARIABLEs in the WORKSET, then DECODE SPECIFIES NON-EXISTENT VARIABLES j-k
DECODE CHANGED FROM <current decode> TO
<truncated decode>.

If an illegal character is present in the DECODE, it is replaced by a 0 and the messages USES ILLEGAL DECODE ELEMENT '<illegal char>'
and ILLEGAL ELEMENT REPLACED BY '0', are given.

Date of last module update: 27 Dec 82

Option: GET

Syntax: 1)GET
2)GET <dataset>
where <dataset> is the name of an existing DATASET on the mass storage device (the file name is <dataset>.DAT). The format of the DATASET file on mass storage must be compatible with PHD.

Description of results:

If syntax 1 is used, PHD will repeatedly request <dataset>s and add the observations from the specified DATASET to the current WORKSET until a null entry is made for the <dataset>. The BASE definition of the first DATASET placed in the WORKSET will be retained when additional DATASETs are added to the WORKSET. When a null entry is given an exit to the main MENU is made. After the initial DATASET is read in , each DATASET is checked for the number of VARIABLEs per observation, (N%). If the number of VARIABLEs per observation is not the same as in the WORKSET, the DATASET is not added to the WORKSET and an advisory message is printed (MISMATCH IN NUMBER OF VARIABLES PER OBSERVATION). The COMMENT and variable NAMEs, UNITs, and METRICs are ignored for any DATASET if a WORKSET BASE already exists and the original value of these items is retained.

When syntax 2 is used, only the DATASET specified is added to the WORKSET, then an exit is made back to the MENU.

In both syntaxes, a number of advisory messages are printed while the DATASET is being processed:

DATASET: <dataset name> n VARIABLES PER OBSERVATION.
COMMENT line
(A summary of the BASE definition is printed here)

On completion of the DATASET processing, the following messages are printed:

NOW nn OBSERVATION(S) IN CURRENT WORKSET
DATASET <dataset name> CLOSED.

Possible errors and recovery options:

If a <dataset> that does not exist on mass storage is specified, the message NO SUCH DATASET is given. Badly formatted data may produce the error messages NO DATA IN OBSERVATION> psn:id, or ERROR nn IN LINE mmm IN GET. In the former case undefined (and probably erroneous) data will exist in the observation specified. The second error is more serious and will terminate the GET operation at the point where the error occurred with a return to the main MENU. In this case the state of the WORKSET will be unknown.

Date of last module update: 27 Dec 82

Option: LIST

Syntax: LIST
where no arguments are allowed.

Description of results:

The contents of the current WORKSET are listed on the terminal in a tabular form along with the WORKSET name and COMMENT. If more than 50 items are in the WORKSET, the listing is halted every 50 entries for viewing, copying, etc. Entering a carriage return will resume the listing process.

Possible errors and recovery options: None.

Date of last module update: 27 Dec 82

Option: MFP (Indirect Option File Processing)

Syntax: MFP
where no arguments are allowed.

Description of results:

PHD will request the name of a command file, <@file>. This command file must contain a sequence of PHD MENU options. The file name on mass storage must be <@file>.PHD. This type of file must have been previously built using one of the host system editors (e.g., the MCR EDI editor, or EDT under DCL). When in the MFP mode, instead of returning to the main MENU for instructions, PHD will process the options from the command file specified instead. Note that sub-commands (i.e. those given in response to requests within each module) must still be input from the terminal during execution of that module. The occurence of an EXIT command in the command file will cause a termination of the MFP mode and a return to the main MENU at the terminal.

Possible errors and recovery options:

Illegal commands and arguments are treated the same as if they had been submitted in response to the main MENU. **Failure to include an EXIT at the end of the command file will result in the abnormal termination of PHD and loss of the current WORKSET.**

Date of last module update: 27 Dec 82

Option: PLOT

Syntax: PLOT
where no arguments are allowed.

Description of results:

This module is designed to plot all or portions of the current WORKSET. When called the PLOT module will request information on what observations to plot, VARIABLEs and labels for the axes and information necessary for "nice" scaling and plotting. All plots are made in the same format: A frame with tic marks at the specified intervals, with labels on the left of the ordinate and below the abcissa. A title is placed across the bottom of the plot, centered below the abcissa label.

The plot can be made only on a TEKTRONIX 40xx series terminal or one compatible with these terminals. An example of the dialog occurring in a call to PLOT follows. In the example below, all values were taken as the default by simply entering a carriage return. If desired, changes could be made by entering the desired value (character string or number). :

```
PLOT□
        WELCOME TO PhD PLOT II
        n VARIABLES/OBSERVATION
        m OBSERVATIONS
        VARIABLES AVAILABLE:
             VARIABLE #      NAME          UNITS

                 1           name1         unit1
                 2           name2         unit2
                 3           name3         unit3

VARIABLE AS ABCISSA[1] ?□
VARIABLE AS ORDINATE[2] ?□
FIRST OBSERVATION TO PLOT[1] ?□
LAST OBSERVATION TO PLOT[m] ?□
PLOTTING name1 IN unit1
  VERSUS name2 IN unit2
OBSERVATIONS 1-m


TO LEAVE UNCHANGED, PRESS <CR>

        PLOT TITLE:  COMMENT for current WORKSET□

        X LABEL:  name1? □

          UNITS:  unit1? □

        Y LABEL:  name1? □

          UNITS:  unit1? □
```

Option: PLOT (Cont'd)

AXIS LIMITS (TO LEAVE UNCHANGED, PRESS <CR>)

X MINIMUM = xmin? □

X MAXIMUM = xmax? □

Y MINIMUM = ymin? □

Y MAXIMUM = ymax? □

TIC MARK INTERVALS (TO LEAVE UNCHANGED, PRESS <CR>)

X: xtic? □
Y: ytic? □

MIN X = xdatamin MAX X = xdatamax
MIN Y = ydatamin MAX Y = ydatamax
SYMBOL SIZE (1-10) [5] ?□

After the entry for the symbol size, the screen will be cleared and the plot produced. PHD then awaits a carriage return to again blank the screen and return to the main MENU.

The legal plot symbol numbers (PSN in the PSN:ID string for each observation) and the symbols they produce are as follows:

1	+
2	X
3	*
4	□
5	diamond
6	Δ
7	∇
8	Move to location but plot no symbol (move with pen up).
9	Solid line from current location to specified location (move with pen down).
10	Same as 9 but dotted line.
11	Same as 9 but dash-dot line.
12	Same as 9 but dashed line.

Possible errors and recovery options:

In general, the PLOT module will recover with an error message ILLEGAL NUMBER or the system error message for "number required", whenever a non-numeric is entered when a number is required. The labels have length limitations which depend on the length of the UNIT name and scaling factors for the particular plot. Generally no warning message is given when a label is truncated, except its truncated form on the plot itself.

Date of last module update: 27 Dec 82

Option: PRINT

Syntax: 1)PRINT
2)PRINT <number of copies>
where <number of copies> is an integer between 1 and 10.

Description of results:

A listing file (PHD.LIS) containing the number of copies of the current WORKSET specified in <number of copies> is produced on the mass storage device. The file may be spooled to the printer using the DCL command PRINT PHD in response to the $ after an EXIT has been made from PHD.

Possible errors and recovery options: None.

Date of last module update: 27 Dec 82

Option: RENAME

Syntax: 1) RENAME
2) RENAME <new workset name>
where <new workset name> is a file name acceptable to VMS.

Description of results:

For both syntaxes, PHD will display the current WORKSET name immediately on entry:

CURRENT WORKSET NAME: <workset name>

If syntax 1 has been used, a <new workset name> will then be requested:

WORKSET NAME ?

The WORKSET name will be changed to the one specified. If a null entry is given, the WORKSET name will not be changed and an immediate exit to the main MENU will be made.

After the WORKSET name change is made, the current COMMENT line will be displayed:

CURRENT COMMENT: <current comment>

Changes may then be made to the COMMENT line by using a <comment> optionally preceded by an editing symbol (see CM option), when the request,

COMMENT ?

is made. If a null entry is made at this point, the COMMENT will remain unchanged. A return is made to the main MENU after the request for COMMENT editing has been processed.

Possible errors and recovery options:

A WORKSET name longer than 9 characters may cause problems in SAVEing the WORKSET. COMMENTs longer than 80 characters will be truncated to 80 characters on transfer to any user written module (USE option). COMMENT lines longer than 40 characters may be truncated in the title of any plot (PLOT option).

Date of last module update: 27 Dec 82

Option: SAVE

Syntax: 1) SAVE
2) SAVE <dataset name>
where <dataset name> is a valid VMS file name.

Description of results:

If syntax 1 is used, PHD will request a <dataset name>, prompting the current WORKSET name as a default.

```
CURRENT WORKSET:  <workset name>
NEW DATASET NAME (cr IF NO CHANGE) ? <dataset name>□
```

A null return at this point will result in the WORKSET being written as a DATASET with the same name as the current WORKSET. The WORKSET will then be written to mass storage under the <dataset name> specified (if a DATASET with that name already exists on mass storage then a new version will be created), with the following advisory messages:

```
CREATING NEW DATASET: <dataset name>
n VARIABLES PER OBSERVATION.  m OBSERVATIONS.
(a display of the BASE being written out will appear here)
DATASET <dataset name> CLOSED.
```

An exit to the main MENU then occurs.

Possible errors and recovery options: None.

Date of last module update: 27 Dec 82

Option: SORT

Syntax: SORT
where no arguments are allowed.

Description of results:

The sorting routine is designed to implement a hierarchy of sorts. The latest sort made is always the major sort, the earliest sort the most minor. The observations will be in the order specified by the major sort VARIABLE first. When values of the major VARIABLE are identical, the order will be determined by the value of the VARIABLE just minor to the major VARIABLE, etc. to the most minor sort specified.

On entry into this module, a sequenced list of VARIABLEs in the *WORKSET* BASE will be given along with an advisory message:

THERE ARE n VARIABLES PER OBSERVATION.
(sequenced list of VARIABLEs appears here)
A POSITIVE SEQUENCE NUMBER IMPLIES ASCENDING SORT
A NEGATIVE SEQUENCE NUMBER IMPLIES DESCENDING SORT
'CR' WILL IMPLEMENT THE SORT VECTORS THAT HAVE BEEN BUILT
SEQUENCE NUMBER OF VARIABLE TO SORT ON?

In addition to the sequence number for a VARIABLE, ID or -ID may be entered to build a sort vector, alphabetically or in reverse alphabetical order, respectively. In these cases the sort will be made on the value of the <psn>:<id> for each observation.

As each VARIABLE sequence number entry is made, the advisory BUILDING SORT VECTOR ON KEY <variable name>, will be given and the sort vector modified to reflect this sort. On completion of the vector, VECTOR BUILT will be given and a request for another VARIABLE to sort on made. When the final vector has been built, the actual sort (physical rearrangement of data in memory) is made by giving a null entry (carriage return only).

The sort vector built to that point is then implemented. The advisory SORT IN PROGRESS is given and a period (.) is printed for every vector segment processed during the sort. When the sort is done the message SORT COMPLETED. is given and an exit to the main MENU made.

Possible errors and recovery options:

If a VARIABLE sequence number that is out of range is given, the entry advisories will be repeated and a new request for sequence number made. **An alphanumeric entry that is not either ID or -ID or null will result in abnormal termination of PHD and the loss of the current WORKSET.**

Date of last module update: 27 Dec 82

Option: SWAP

Syntax: SWAP
where no arguments are allowed.

Description of results:

This option is useful where a particular ordering of the VARIABLEs in the WORKSET is required. Options such as the FIT option or user written options may depend on having the VARIABLEs in a particular order within an observation. When two different options require different ordering of VARIABLEs, it becomes necessary to either create two DATASETs or reorder the VARIABLEs. To avoid this difficulty the SWAP option allows the exhange of two different VARIABLEs within all observations. VARIABLEs may be placed in any order desired by a suitable sequence of SWAP operations.

On entry into this module, a sequenced listing of the VARIABLEs in the BASE will be given and a request for the sequence numbers of the VARIABLEs to be swapped made.

VARIABLES IN FILE
(sequenced listing of variables in the WORKSET BASE appears here)
NUMBERS OF VARIABLES TO EXCHANGE ?

Valid sequence numbers for two VARIABLEs to be exchanged must be given at this point. The values of the VARIABLEs for all observations in the WORKSET, their NAMEs, UNITs, and METRICs will be exchanged. A return to the main MENU is then made.

Possible errors and recovery options:

If invalid numbers are entered as VARIABLE sequence numbers, a system error will result. If an out-of-range sequence number is given, no action will be taken and the SWAP module re-initiated.

Date of last module update: 27 Dec 82

Option: USE

Syntax: 1) USE
2) USE <user file name>

where <user file name> must be the name of an .EXE type file built using the USER.BAS module as an envelope program (See How to Write a User Program Module, p 35). The user written program lines which may modify the WORKSET must be sequenced between 1000 and 29998 in order not to overlap with the statements of the envelope program..

Description of results:

If syntax 1 is used, PHD will request the <user file name>:

PROGRAM NAME ?

If a null entry is made at this point then a return to the main MENU will be made; otherwise, the action proceeds as for syntax 2.

When syntax 2 is used, a request is made for arguments to be passed to the user program via the string variable Y$. The WORKSET is stored temporarily in the file USER.TMP and the user program called. When the user program returns to PHD, a check is made for the existence of the file USER.TMP. If this file is found an advisory, USER PROGRAM COMPLETE...RECOVERING WORKSET, is made and the contents of the file USER.TMP are read into the WORKSET. The retrieval and temporary storage of the WORKSET within the user program module is done by the instructions in the user program envelope maintained in the file [SEILERV.PHD]USER.BAS (see How to Write a User Program Module, p 35).

Possible errors and recovery options:

If an error is made in the <user file name>, or no such file exists in the user's directory, an error message will be printed:

ERROR (n) IN LINE m
error explanation
ABORTING USER PROGRAM CALL FOR <user file name>

A return to the main MENU will be made at this point.

If a fatal error occurs during the execution of the user program, then the exit to return to PHD may not be made or not be made properly. In such a case the temporary file USER.TMP will continue to exist on mass storage. The next time that PHD is initiated, it will find this file and attempt a recovery, resulting in a spurious recovery message and the retrieval of a WORKSET unexpectedly.

If for some reason the temporary file does not exist when the user program is executed, the message NO DATA BASE TRANSFERRED FROM PHD--EXITING TO PHD is given, and an attempt to return to PHD is made.

Date of last module update: 27 Dec 82

BASIC VARIABLE MAPS OF PHD

List of PHD variables accessible from user written modules*: **VAX/VMS**

Real

M(9)	VARIABLE METRICs are stored in this array.
X(1000)	Observations are stored in this array. The values for the ith observation are stored in the same order as the VARIABLEs in the BASE, beginning with the first in the (i-1)*N%th element of X().

Integer

L%(500)	This array is a scratch pad for sorting, linking, etc.
N%	Contains the current number of VARIABLEs per observation, i.e., the number of VARIABLEs in the WORKSET.
N9%	Contains the number of observations in the WORKSET.

String

A$(500)	Contains the PSN:ID for the observations. The PSN:ID for the ith observation is in the (i-1)th element.
F$	Contains the WORKSET name. 40 characters maximum length transferred from/to PHD.
C$	Contains the current COMMENT. 80 characters maximum length transferred from/to PHD.
U$(9)	Contains the UNIT names for the VARIABLEs in the WORKSET. 16 characters maximum transferred from/to PHD.
U9$	Contains the directory of the version of PHD being used by the user written module (normally [SEILERV.PHD]).
X$(9)	Contains the NAMEs for the VARIABLEs in the WORKSET. 16 characters maximum transferred from/to PHD.
Y$	Argument string passed to/from PHD on exit/entry to the user written module respectively. 80 character maximum length transferred to/from PHD.

*PHD depends on the values of these variables being preserved. They should be used by the user in his modules only with caution.

List of variables used by the PHD modules*: RSX11-M Version

Real:

M(18)*	Variable METRICs are stored in this array.
X(1000)	Observations are stored in this array
Z	This variable is used as a pseudo variable to set the linewidth of the terminal in PHD.

Integer:

C%	
F%	
I%	
J%	
K%	
L%	
L%(500)	
M%	
N%*	Number of VARIABLEs in current BASE.
N9%*	Number of observations in WORKSET.
Q%	
P9%	
S%	
T%	

String:

A$	
A$(500)=8*	Observation PSN:IDs stored in this virtual array.

* PHD depends on having the values of these variables preserved. They should be used in a USER written program only with caution.

List of variables used by the PHD modules (Cont'd)*: **RSX11-M Version**

C$*	This string contains the COMMENT for the WORKSET.
D$	
E$*	This variable is used to store the ESCape character used to control the terminal functions.
F$*	This string contains the name of the WORKSET.
F1$	
F2$	
F3$	
F4$	
O$*	This string contains the option list used by the MENU module to display and check the validity of option selections.
Q$	
T$*	This string contains the banner (Program for Handling Data) used by the MENU.
X$	
X$(18)*	VARIABLE NAMEs are stored in this array.
U$(18)*	VARIABLE UNITs are stored in this array.
U9$*	This string is used to contain the mass storage device and directory where the PHD modules are stored. Used for transfer to all modules except the USER programs.
Y$	Argument list carried from module to module in this string.

* PHD depends on having the values of these variables preserved. They should be used in a USER written program only with caution.

HOW TO WRITE A USER PROGRAM MODULE

Before beginning a user program you must understand the structure of the database. That is, the BASIC names of the variables where the database is stored (i.e., the BASE definitions, NAMEs, UNITs, METRICs, WORKSET NAME, COMMENT, data PSN:IDs, and the data itself). The table of PHD variables accessible from user written modules is a good place to start. When the structure is understood, the algorithm for the program must be designed.

In the design of this algorithm the assumption is made that the WORKSET of PHD is available, and the corresponding BASIC variable names must be used in the user program. In virtually any user program it is necessary to calculate the address of some VARIABLE in an observation. The algorithm must be designed to use information from the data base to find this number. For instance, N% is an integer number that gives the number of VARIABLEs in the current BASE.

Using this value the user can calculate the address of any particular VARIABLE in an observation. All data numeric data for the observations are stored in the one-dimensional array X(). Given a BASE that contains 4 VARIABLEs per observation, the element of X() that contains the third VARIABLE in the 12th observation would be (12-1)*4+3-1 [using the formula from the table of variables accessible to user modules]. In general then xth VARIABLE of the yth observation would be at (y-1)*N%+x-1. Since the number of observations is stored in the BASIC variable N9%, y could be checked to see if it lies in the range 1 to N9% inclusive. The other elements of the data base that may be necessary are the BASE definition items: NAME, UNIT, METRIC of each variable, the WORKSET name, COMMENT and the PSN:ID of the observation.

HOW TO WRITE A USER PROGRAM MODULE (Cont'd)

The NAMEs, UNITs and METRICs for the NZ VARIABLEs in the BASE are stored in the elements of the one dimensional string arrays X$(),U$(), and the one dimensional real array M(), repectively, in the order in which they were defined (unless SWAP operations have been done). The first VARIABLE defined has its defining NAME, UNIT and METRIC in element 1 of X$(), U$(), and M(), respectively. Element 0 for these arrays is undefined. The WORKSET name is stored in the string variable F$ and the COMMENT in the string variable C$. The PSN:ID for observation i is stored in the i-1th element of the string array A$(). A$() is a non-dynamic string array (i.e., all elements of the array are of a fixed length, 8 characters in this case). If a PSN:ID string is shorter than 8 characters, it is padded on the right with blanks. If an attempt is made to store a string longer than 8 characters into any element of A$(), only the first (leftmost) eight characters are stored. The various string functions available in BASIC may be used to isolate, change or examine the PSN or ID parts of this string by using the colon which separates them as a marker.

A typical program might progress through the data using a FOR/NEXT loop and modify a specified VARIABLE. As an example, let us write a program to transform any VARIABLE to the log(base e) of that VARIABLE. One program to do this would be as follows:
(the unnumbered lines are comments explaining what the next line or lines of code do)

HOW TO WRITE A USER PROGRAM MODULE (Cont'd)

Sample User Program Body

Get the sequence number of the VARIABLE that is to be transformed.

```
1000 INPUT "NUMBER OF VARIABLE TO TRANSFORM TO LOG";V%
```

Check to make sure the VARIABLE entered is in the range of the number of VARIABLEs in the BASE.

```
1010 IF V%<1% OR V% > N% THEN PRINT "NO SUCH VARIABLE"\GOTO 1000
```

Define a FOR/NEXT loop to run through all the observations from 1 to N9% (note 0 to N9%-1 is used here to make calculations easier)

```
1020 FOR I%=0% TO N9%-1%
```

From the value of the observation sequence number I% calculate the location of the V%th VARIABLE in each observation.

```
1030 K%=I%*N%+V%-1%
```

Take the log of the specified VARIABLE and place it back in the same element of X().

```
1040 X(K%)=LOG(X(K%))
```

Continue this process as long as there is a NEXT value.

```
1050 NEXT I%
```

Change the NAME of the V%th VARIABLE by prefixing "ln " to its current name.

```
1060 X$(V%)="ln "+X$(V%)
```

Change the units of the V%th variable to null.

```
1070 U$(V%)=""
```

HOW TO WRITE A USER PROGRAM MODULE (Cont'd)

Notify the user that the transformation has been done by printing a message on the terminal.

```
1080 PRINT "TRANSFORMATION COMPLETE"
```

When the program has been written, a check must be made to make sure that the program lines all fall in the range 1000-29998. Then the envelope program, USER.BAS, is added to the program. If the above program were saved on mass storage under the name XFORM.BAS, the executable user program file, XFORM.EXE, would be built using the following DCL and BASIC commands (all user input ends with a □):

```
$ BAS□
VAX-11 BASIC V1.4

Ready
OLD XFORM□

Ready
APPEND [SEILERV.PHD]USER□

Ready
REPLACE XFORM□

Ready

EXIT□

$ BAS XFORM□
$ LINK XFORM□
```

The executable file, XFORM.EXE, would now exist in the directory that XFORM.BAS was stored in. The program may be run by replying USE XFORM to the OPTION? request in the main MENU.

PARAMETER FILES FOR DATA FITTING

Data fitting in PHD is done on the current WORKSET using either a user written program with the USE option, or with the FIT option if the functional form of the equation to fit is (or can be made to be) compatible with that required by this option.

The functional form required by the FIT option is the generalized polynomial in the VARIABLEs, x_j, linear in the parameters, θ_i.

$$0 = \sum_i \left(\theta_i \prod_j x_j^{m_j}\right)$$

where the sum is over all i parameters specified by the parameter file, θ_i ($i<17$), and the product is over all j variables, x_j, in the WORKSET to some power m_j. The $\ln(x_j)$ may also be used in place of $x_j^{m_j}$

A function is linear in its parameters if the partial derivatives with respect to each of the parameters gives a set of equations that are not functions of any of the parameters. Two additional constraints are imposed by the current implementation of the FIT option:

1) The VARIABLEs in the polynomial must be taken to an integer power (or the natural logarithm may be used) and,

2) The exponents must be in the range from -9 to +9 inclusive.

To describe the particular polynomial desired, a file called a parameter file is required by the FIT option. The file may have any name, but the extention must be .PAR, in order to be accessed by the option. The file must contain the following:

1) A banner (or default title for any fit to be made using this parameter file) The banner is any character string not containing a carriage return.

PARAMETER FILES FOR DATA FITTING (Cont'd)

2) A set of parameter description records, one for each parameter. The parameter description records contain four elements separated by commas.

a) The name of the parameter. Up to 16 characters (no commas or carriage returns)

b) The unit of the parameter. Up to 16 characters (no commas or carriage returns)

c) The initial value of the parameter. Any number in a format acceptable to DEC BASIC.

d) The parameter DECODE string. This is a string of characters which uses positional notation to define the function of each VARIABLE in the WORKSET to be used in the term containing the parameter. The position of the character and its associated meaning are as follows:

The first position--Must be either a + or a -. The + indicates the parameter is to be adjusted by the fitting program, - specifies that the initial value of the parameter is be retained during the fit. If the first position is neither a + nor a - then the program inserts a -.

All following positions (2 through the number of VARIABLEs in the WORKSET--The ith position corresponds to the ith-1 VARIABLE in the WORKSET and the character itself specifies the function of the VARIABLE to be used: either a power of the VARIABLE or its logarithm to the base e.

PARAMETER FILES FOR DATA FITTING (Cont'd)

The characters 0 through 9 specify the zeroth through ninth power of the VARIABLE, respectively; the letters A-I specify the reciprocal of the zeroth through ninth power of the VARIABLE, respectively; an L specifies the natural logarithm of the VARIABLE.

For a valid description of the equation, the number of positions in the DECODE must be one greater than the number of VARIABLEs in the WORKSET. If it is desired not to include a VARIABLE in a term, the character 0 should be used in the DECODE (since $x^0 = 1$, this effectively removes the VARIABLE from the term). As a concrete example, a parameter file is constructed to fit the quadratic equation

$$\text{Resistance} = A + B*\text{Temperature} + C*\text{Temperature}^2$$

to a DATASET containing the variables TEMPERATURE and RESISTANCE.

Letting R be resistance and T be temperature, the equation is rearranged to standard form:

$0 = A + B*T + C*T^2 + D*R$ where the value of D must be -1.

If the DATASET has the VARIABLEs stored with T as the first VARIABLE and R as the second VARIABLE, the DECODEs for A, B, C and D will be +00,+10,+20, and -01, respectively. The parameter file must be created before fitting using a system editor (e.g., EDI or EDT) and would appear as

```
FIT OF RESISTANCE TO QUADRATIC IN TEMPERATURE
A,OHM,0,+00
B,OHM/K,0,+10
C,OHM/K2,0,+20
D,OHM,-1,-01
```

PARAMETER FILES FOR DATA FITTING (Cont'd)

If the file in which these images are stored is called RQUAD.PAR then the fit of the DATASET name RVST would be made using PHD by GETting the DATASET RVST and specifying FIT RQUAD in response to the next OPTION? request in the main MENU.

Linking Programs for the VAX

```
PHD           3-MAR-1983 08:40

10 REM ### THIS IS VAX PHD ###
20 DIM X(1000),XS(16),US(16),DS(16),M(16),L%(500)
30 COM (PHDID) A$(500)=8
40 DIM PHS(16),PDS(16),PUS(16),PV(16)
50 DECLARE REAL M(16,16),MI(16,16),R(16),D(16),C(16)
60 TERM$='4014' \ ES=CHR$(27)\PRINT ES;CHR$(140) \ SLEEP 1 \ PRINT \PRINT
70 PRINT 'PHD VERSION 8.8 VAX BASIC 24-FEB-83'
80 IF TERM$='4014' THEN 90 ELSE T$='Program for Handling Data' \ GOTO 230
90 T$=ES+'2P'+ES+';ROGRAM FOR '+ES+'2H'+ES+';ANDLING '+ES+'2D'+ES+'!ATA'
100 MARGIN 80,132 \ ON ERROR GO TO 230
110 OPEN 'USER.TMP' FOR INPUT AS FILE #2, SEQUENTIAL VARIABLE
120 PRINT 'USER PROGRAM COMPLETE...RECOVERING WORKSET'
130 GET #2 \ MOVE FROM #2,Y$=80 \ GET #2 \ MOVE FROM #2,C$=80
140 GET #2 \ MOVE FROM #2,M%,M9%,F$=40,U9$=16
150 FOR I%=1% TO M%
160 GET #2 \ MOVE FROM #2,XS(I%)=16,US(I%)=16,M(I%) \ NEXT I%
170 FOR I%=0% TO M9%-1% STEP 5% \ GET #2
180 MOVE FROM #2,A$(I%)=8,A$(I%+1%)=8,A$(I%+2%)=8,A$(I%+3%)=8,A$(I%+4%)=8
190 NEXT I%
200 FOR I%=0% TO M9%+M%-1% STEP 10% \ GET #2
210 MOVE FROM #2,X(I%),X(I%+1%),X(I%+2%),X(I%+3%),X(I%+4%),X(I%+5%),X(I%+6%),X(I%+7%),X(I%+8%),X(I%+9%)
220 NEXT I% \ CLOSE #2 \KILL 'USER.TMP'\ GOTO 250
230 M%=0% \ M9%=0% \ F$='' \ U9$='[SEILERV.PHD]' \ RESUME 240
240 ON ERROR GO TO 0
250 GOSUB 270 \ GO TO 250
260 REM ### THIS IS VAX CHOO ###
270 O$='!ADD !BASE new/<dataset>!CORRECT!CM comment!DELETE base/data!'
272 O$=O$+'ENTER!EXIT program!FIT parmfile!GET dataset!LIST!MFP!PLOT!PRINT!'
274 O$=O$+'RENAME name!SAVE dataset!SORT!SWAP!USE userprog!'
280 I%=120%
290 IF SEG$(O$,I%,I%)='!' THEN 300 ELSE I%=I%-1% \ GOTO 290
300 PRINT \ PRINT T$ \ PRINT  \ PRINT SEG$(O$,1%,I%) \ PRINT '!';SEG$(O$,I%+1%,255%) \ LINPUT 'OPTION';O$
310 O$=EDIT$(O$,156) \ Y$='' \ IF O$='' THEN 270
320 O%=POS(O$,' ',1) \ IF O%>0% THEN Y$=SEG$(O$,O%+1%,255%) \ O$=SEG$(O$,1%,O%-1%)
330 I%=POS(O$,'!'+SEG$(O$,1,4),1) \ IF I%>0% THEN 350
340 PRINT 'OPTION INCOMPLETE OR NOT AVAILABLE' \ GO TO 410
350 O$='!'+SEG$(O$,I%+1%,I%+4%) \ IF Y$='' THEN 380 ELSE IF POS('!DELE',O$,1)<>1 THEN 380
360 IF POS('DATA',TRM$(Y$),1)=1 THEN M9%=0% \ GO TO 410
370 IF POS('BASE',TRM$(Y$),1)=1 THEN M9%=0% \ M%=0% \ C$='' \ F$='' \ GO TO 410
380 O1$='!AD!BA!CO!DE!EN!EX!FI!GE!LI!MF!PL!PR!SA!SO!SW!US!RE!CM'
390 Z%=POS(O1$,SEG$(O$,1,3),1)/3+1\IF Z%<1% THEN 340
400 ON Z% GOTO 1730,2120,2600,2810,4265,1580,430,3110,3450,4266,4270,3550,3690,3890,4267,1930,1600,420
410 RETURN
420 IF Y$='' THEN 1620 ELSE 1630
430 P9%=16%
440 ON ERROR GOTO 1240
450 DECODE$='ABCDEFGHI0123456789L'
460 P%=0% \ P$=EDIT$(Y$,156)\IF P$='' THEN LINPUT 'PARAMETER FILE';P$
470 P$=EDIT$(P$,156)
480 IF P$='' THEN 610 ELSE IF POS(P$,'.',1)<1 THEN P$=P$+'.PAR'
490 OPEN P$ FOR INPUT AS FILE #2
500 LINPUT #2,TITLE$

Ready
```

PWD   3-MAR-1983 08:40

```
510 PRINT TITLE$
520 IF P%>=P9% THEN 1230
530 INPUT #2,PN$(P%+1%),PU$(P%+1%),PU(P%+1%),PD$(P%+1%) \ P%=P%+1%
540 GO TO 520
550 PRINT\PRINT 'PARAMETER TABLE' \ PRINT
560 PRINT 'NR  NAME';TAB(21);'UNITS';TAB(39);'EQ DECODE';TAB(56);'INIT VALUE';TAB(76);'ADJ'
570 PRINT \ RETURN
580 PRINT I%;TAB(4);PN$(I%);TAB(20);'E';PU$(I%);'3';TAB(38);'<';PD$(I%);'>';TAB(56);PU(I%);
590 IF SEG$(PD$(I%),1,1)='-' THEN PRINT TAB(76);' N' ELSE PRINT TAB(76);' Y'
600 RETURN
610 P8%=0 \ FOR I%=1% TO P% \ C$=SEG$(PD$(I%),1%,1%)
620 IF C$='-' THEN 630 ELSE IF C$<>'+' THEN PD$(I%)='+'+PD$(I%)
630 IF SEG$(PD$(I%),1%,1%)<>'-' THEN P8%=P8%+1%
640 NEXT I%
650 PRINT [illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
800 [illegible]
810 [illegible]
820 PRINT [illegible] ILLEGAL DECODE ELEMENT [illegible];SEG$(PD$, [illegible]
830 PRINT 'ILLEGAL ELEMENT REPLACED BY [illegible]'
840 PD$(I%)=SEG$(PD$(I%),1%,J%-1%)+'0'+SEG$(PD$(I%),J%+1%,255%)
850 NEXT J%
860 NEXT I%
870 PRINT 'BEGIN FIT'
880 TITLE$=FNA$('TITLE:  ',TITLE$)
890 TOL=FNA('DELTA SIGMA FOR FIT CONVERGENCE','1E-7')
900 MAT H=ZER(P8%,P8%) \ MAT HI=ZER(P8%,P8%)
910 OLDSIGMA=0 \ ITN%=0%
920 MAT C=ZER(P8%) \ MAT R=ZER(P8%) \ SSQ=0
930 POINTS%=0%\FOR I%=0% TO M9%-1% \ K%=I%*N%-1%
940 MAT D=CON(P%)
950 FOR J%=1% TO P%
960 A$=SEG$(PD$(J%),2%,255%) \ IF LEN(A$)<N% THEN L9%=LEN(A$) ELSE L9%=N%
970 FOR L%=1% TO L9% \D(J%)=D(J%)*FND(SEG$(A$,L%,L%),L%)\NEXT L%
980 NEXT J%
990 IF ITN%>0% THEN 1040
1000 FOR J%=1% TO P8% \ FOR L%=J% TO P8% \ H(J%,L%)=H(J%,L%)+D(J%)*D(L%)
```

Ready

```
PHD          3-MAR-1983 08:41

1010 NEXT L% \ NEXT J%
1020 FOR L%=1% TO P8% \ FOR J%=L%+1% TO P8% \ H(J%,L%)=H(L%,J%)
1030 NEXT J% \ NEXT L%
1040 D(0%)=0 \ FOR J%=1% TO P%\D(0%)=D(0%)+PV(J%)*D(J%)\ NEXT J%
1050 $$0=$$0+D(0%)*D(0%)
1060 FOR J%=1% TO P8% \ R(J%)=R(J%)-D(0%)*D(J%) \ NEXT J%
1070 POINT$%=POINT$%+1%
1080 NEXT I%
1090 IF ITN%=0% THEN MAT HI=INV(H)
1100 FOR J%=1% TO P8% \ C(J%)=0 \ FOR L%=1% TO P8% \C(J%)=C(J%)+HI(J%,L%)*R(L%)
1110 NEXT L% \ NEXT J%
1120 FOR J%=1% TO P8% \ PV(J%)=PV(J%)+C(J%)\ NEXT J%
1130 OLDSIGMA=SIGMA \SIGMA=SQR($$0/POINT$%)
1140 IF ITN%=0% THEN OLDSIGMA=-SIGMA
1150 IF 2*(SIGMA-OLDSIGMA)<=(OLDSIGMA+SIGMA)*TOL THEN 1450
1160 GOSUB 1170 \ GO TO 920
1170 ITN%=ITN%+1%\ PRINT 'ITERATION ';ITN%;' SIGMA = ';SIGMA
1180 IF (OLDSIGMA+SIGMA)>0 THEN DELSIG=2*(SIGMA-OLDSIGMA)/(SIGMA+OLDSIGMA) ELSE DELSIG=99
1190 PRINT POINT$%;' POINTS USED.  DELTA SIGMA = ';DELSIGMA \ PRINT
1200 PRINT 'ADJUSTABLE PARAMETERS CORRECTED VALUE'\ FOR I%=1% TO P8%
1210 PRINT PH$(I%);' = ';PV(I%);'    ';\ IF MAR(0)-CCPOS%(0)<40 THEN PRINT
1220 NEXT I% \ PRINT \ RETURN
1230 PRINT 'MAXIMUM PARAMETERS EXCEEDED ';P9% \ GOTO 610
1240 IF ERR=11 AND ERL=530 THEN PRINT '**EOF**'\RESUME 610
1250 IF ERR=5 AND ERL=490 THEN PRINT 'NO SUCH PARAMETER FILE!  ';P$ \ Y$=''\ RESUME 440
1260 IF ERL>=930 AND ERL<1080 THEN PRINT 'POINT ';I%;' NOT USED'\RESUME 1080
1270 PRINT 'ERROR';ERR;'IN LINE';ERL;' OF MODULE ';ERN$
1280 PRINT ERT$(ERR) \ RESUME 1560
1290 DEF FND(C$,X%)
1300 ON ERROR GO BACK
1310 Z%=POS(DECODE$,C$,1)
1320 IF Z%>19% THEN 1350
1330 IF Z%>9% THEN X=X(K%+X%)^(Z%-10%) ELSE X=1/X(K%+X%)^Z%
1340 FND=X \ FNEXIT
1350 IF Z%=20% THEN X=LOG(X(K%+X%)) \ GO TO 1340
1360 FNEND
1370 DEF FNA(MSG$,DEF$)
1380 PRINT MSG$;' [';DEF$;']'; \LINPUT X$ \ IF X$='' THEN X$=DEF$
1390 ON ERROR GOTO 1400 \ FNA=VAL(X$) \ FNEXIT
1400 FNA=VAL(DEF$) \ RESUME 1410
1410 FNEND
1420 DEF FNA$(MSG$,DEF$)
1430 PRINT MSG$;' [';DEF$;']';\LINPUT X$\IF X$='' THEN X$=DEF$
1440 FNA$=X$ \ FNEND
1450 GOSUB 1170
1460 PRINT 'ITERATION ';ITN%;' CONVERGENCE OCCURRED.'
1470 LINPUT 'HIT CARRIAGE RETURN TO CONTINUE',X$
1480 PRINT CHR$(27);CHR$(140);CHR$(27);'9' \ SLEEP 1\ PRINT \ PRINT  \PRINT\PRINT F$\PRINT TITLE$\PRINT
1490 PRINT 'SIGMA = ';SIGMA;'  DELTA SIGMA = ';DELSIG \ PRINT
1500 PRINT 'PARAMETER';TAB(17);'UNITS';TAB(35);'VALUE AND UNCERTAINTY' \ PRINT

Ready
```

```
1510 FOR I%=1% TO P%
1520 IF I%>P8% THEN X$=' NOT ADJUSTABLE' ELSE X$='(+/-) '+FORMAT$(SIGMA*SQR(ABS(HI(I%,I%))),'#.#^^^^')
1530 PRINT PN$(I%);TAB(17);'E';PU$(I%);'J';TAB(35);PU(I%);TAB(55);X$
1540 NEXT I% \ PRINT
1550 PRINT 'FIT TERMINATION--ENTER cr FOR MAIN MENU' \ LINPUT Y$
1560 PRINT CHR$(27);'1'
1570 RETURN
1580 IF Y$<>'' THEN CHAIN Y$
1590 PRINT 'Exiting PhD' \ GO TO 6699
1600 PRINT 'CURRENT WORKSET NAME:  ';F$ \ IF Y$='' THEN LINPUT 'WORKSET NAME';Y$
1610 IF Y$='' THEN RETURN ELSE F$=Y$
1620 PRINT 'CURRENT COMMENT:  ';C$ \ Y$='' \ LINPUT 'COMMENT';Y$
1630 Y$=EDIT$(Y$,136) \ Q$=SEG$(C$,POS(C$,'!',1)+1,80)
1640 IF Y$='' THEN RETURN ELSE ON POS([illegible]',SEG$(Y$,1,1)),[illegible] GO TO 1690,1660,[illegible],1670 [illegible]556
1650 IF LEN(Y$)[illegible]LEN(C$) THEN Q$=SEG$(Q$,1,LEN(Q$)[illegible])+SEG$(Y$,2,[illegible]
1660 IF LEN(Y$)[illegible]LEN(Q$) THEN Q$=SEG$([illegible]+1,[illegible] GO TO 1700
1670 Q$=SEG$(Y$,2,80)+Q$ \ GO TO 1700
1680 Q$=Q$+SEG$(Y$,2,80) \ GO TO 1700
1690 Q$=Y$
1700 C$=TIME$(0)+[illegible]+DATE$(0)[illegible]+Q$
1710 PRINT 'DATASET: [illegible] PRINT [illegible]
1720 REM [illegible] THIS IS VAX [illegible]
1730 IF N%>0% THEN 1770
1740 PRINT [illegible] NOT DEFINED. USE [illegible]
1750 GO TO 1920
1760 PRINT 'ENTERING [illegible] LAST [illegible]
1770 K%=50%
1780 IF K%<0% THEN 1790
1790 I%=N9%[illegible] WILL BE NUMBER [illegible]+1%
1800 PRINT [illegible] LINPUT [illegible];Q$
1810 IF Q$='ERROR' THEN N9%[illegible] THEN 1[illegible] ELSE [illegible] GO TO [illegible]
1820 IF Q$='END' THEN 1920 ELSE IF POS(Q$,'[illegible]',2)=0 THEN Q$=[illegible]+Q$
1830 IF LEN(Q$)<? THEN 1850
1840 PRINT 'IDC'+Q$;'J IS TOO LONG' \ GO TO 1800
1850 A$(N9%)=Q$
1860 N9%=N9%+1% \ FOR J%=1% TO N%
1870 PRINT X$(J%);'E';U$(J%);'J'; \ LINPUT '';Q$
1880 IF Q$='ERROR' THEN N9%=N9%-1% \ J%=N% \ GO TO 1900
1890 X(I%)=VAL(Q$) \ I%=I%+1%
1900 NEXT J%
1910 K%=K%-N%-2% \ GO TO 1780
1920 RETURN
1930 IF Y$='' THEN LINPUT 'PROGRAM NAME';Y$ \ IF Y$='' THEN RETURN
1940 Q$=Y$ \PRINT TRM$(Q$);' '; \ Y$='' \ LINPUT 'ARGUMENTS';Y$
1950 OPEN 'USER.TMP' FOR OUTPUT AS FILE #2,RECORDSIZE 80%, SEQUENTIAL VARIABLE
1960 MOVE TO #2,Y$=80 \ PUT #2,COUNT 80% \ MOVE TO #2,C$=80 \ PUT #2,COUNT 80%
1970 MOVE TO #2,N%,N9%,F$=40,U9$=16\ PUT #2, COUNT 64%
1980 FOR I%=1% TO N%
1990 MOVE TO #2,X$(I%)=16,U$(I%)=16,M(I%) \ PUT #2,COUNT 40%\ NEXT I%
2000 FOR I%=0% TO N9%-1% STEP 5%

Ready
```

```
PHD            3-MAR-1983 08:42

2010 MOVE TO #2,A$(I%)=#,A$(I%+1%)=#,A$(I%+2%)=#,A$(I%+3%)=#,A$(I%+4%)=#
2020 PUT #2,COUNT 40% \ NEXT I%
2030 FOR I%=0% TO N9%*N%-1% STEP 10%
2040 MOVE TO #2,X(I%),X(I%+1%),X(I%+2%),X(I%+3%),X(I%+4%),X(I%+5%),X(I%+6%),X(I%+7%),X(I%+8%),X(I%+9%)
2050 PUT #2,COUNT 80% \ NEXT I% \ CLOSE #2
2060 ON ERROR GO TO 2070 \ CHAIN O$
2070 PRINT "ERROR (";ERR;") IN LINE ";ERL\ PRINT ERT$(ERR)
2080 PRINT "ABORTING USER PROGRAM CALL FOR ";O$
2090 KILL "USER.TMP"\ON ERROR GO TO 0 \ RESUME 2100
2100 RETURN
2110 REM ### THIS IS VAX BASE ###
2120 IF Y$="NEW" THEN D$=Y$ \Y$="" \N%=0% \N9%=0% \GOSUB 1600 \ GO TO 2160
2130 IF Y$="" THEN 2170 ELSE IF POS(Y$,".",1)<1 THEN Y$=Y$+".DAT"
2140 OPEN Y$ FOR INPUT AS FILE #2 \ INPUT #2,N% \ FOR I%=1% TO N%
2150 INPUT #2,X$(I%),U$(I%),M(I%) \ NEXT I% \ CLOSE #2
2160 GOSUB 2430 \ IF D$="NEW" THEN 2260
2170 LINPUT "A(DD)|D(ELETE)|C(HANGE) NAME|UNIT|METRIC OF A VARIABLE ";Y$
2180 O$=SEG$(Y$,1,1) \ IF O$="A" THEN 2210 ELSE IF O$="D" THEN 2270 ELSE IF O$="C" THEN 2350
2190 IF O$="" THEN 2570 ELSE IF O$="N" THEN 2570 ELSE IF O$="Y" THEN 2560
2200 PRINT Y$;"??? TYPE 'CR' TO EXIT" \ GO TO 2160
2210 IF N9%<1% THEN 2260 ELSE PRINT "REPACKING DATA (STRETCH)"
2220 FOR I%=N9%-1% TO 0% STEP -1% \ J%=I%*(N%+1%)+N% \ K%=I%*N%+N%-1%
2230 X(J%)=0 \ J%=J%-1%
2240 FOR L%=N% TO 1% STEP -1% \ X(J%)=X(K%) \ J%=J%-1% \ K%=K%-1% \ NEXT L%
2250 NEXT I%
2260 K%=N%+1% \ Y$="ADDITIONAL" \ GOSUB 2460 \ IF Y$="" THEN 2580 ELSE N%=K% \ GO TO 2160
2270 PRINT "VARIABLE TO DELETE"; \ GOSUB 2380
2280 IF K%<1% THEN 2160
2290 FOR I%=K%+1% TO N% \ X$(I%-1%)=X$(I%) \ U$(I%-1%)=U$(I%) \ M(I%-1%)=M(I%) \ NEXT I%
2300 X$(N%)="" \ U$(N%)="" \ M(N%)=0
2310 N%=N%-1 \ IF N9%<1% THEN 2160 ELSE PRINT "REPACKING DATA (SQUEEZE)"
2320 J%=0% \ L%=0% \ FOR I%=0% TO N9%-1%
2330 FOR M%=1% TO N%+1% \ IF M%=K% THEN 2340 ELSE X(J%)=X(L%) \ J%=J%+1%
2340 L%=L%+1% \ NEXT M% \ NEXT I% \ GO TO 2160
2350 PRINT "VARIABLE TO CHANGE";
2360 GOSUB 2380 \ IF K%<1% THEN 2160
2370 Y$="CHANGED" \ GOSUB 2460 \ GO TO 2160
2380 INPUT $0,Y$ \ K%=0% \ IF Y$="" THEN RETURN
2390 FOR J%=1% TO N% \ IF Y$<>X$(J%) THEN 2410
2400 K%=J% \ J%=N%
2410 NEXT J% \ IF K%>0% THEN RETURN
2420 PRINT "THERE IS NO VARIABLE NAMED '";Y$;"'" \ RETURN
2430 PRINT\ PRINT\ PRINT "THE CURRENT BASE HAS";N%;" VARIABLES PER OBSERVATION."
2440 FOR I%=1% TO N%\ PRINT X$(I%);"(";U$(I%);")=";STR$(M(I%));"  ";
2450 NEXT I% \ PRINT  \ PRINT  \ RETURN
2460 PRINT "ENTER NAME,UNIT,METRIC FOR ";Y$;" VARIABLE"; \ LINPUT $0,O$
2470 IF O$="" THEN Y$="" \ RETURN
2480 O%=POS(O$,",",1) \ IF O%<>0% THEN 2490 ELSE O%=LEN(O$)+1%
2490 IF O%=1% THEN 2500 ELSE X$(K%)=SEG$(O$,1%,O%-1%)
2500 O$=SEG$(O$,O%+1%,255%)

Ready
```

```
PHD          3-MAR-1983 08:42

2510 Q%=POS(Q$,',',1) \ IF Q%)0% THEN 2520 ELSE Q%=LEN(Q$)+1%
2520 IF Q%=1% THEN 2530 ELSE U$(K%)=SEG$(Q$,1%,Q%-1%)
2530 Q$=SEG$(Q$,Q%+1%,255%)
2540 IF Q$<>'' THEN M(K%)=VAL(Q$)
2550 RETURN
2560 PRINT 'WHICH ONE, SMARTY?' \ GO TO 2160
2570 IF M9%<1% THEN 1730
2580 RETURN
2590 REM ### THIS IS VAX CORR ###
2600 INPUT 'SEQUENCE NUMBER OF OBSERVATION TO CORRECT';Y$
2610 IF Y$='' THEN 2790 ELSE L%=VAL(Y$) \ I%=L%*N%-N% \ IF I%<0% THEN 2740
2620 IF L%>M9% THEN 2740 ELSE GO TO 2710
2630 PRINT  \ LINPUT 'PSN!ID' OF 'VARIABLE NAME,VALUE' ';Q$
2640 IF Q$='' THEN 2600 ELSE Q$=TRM$(Q$) \ K%=0%
2650 J%=POS(Q$,'!',1) \ IF Q%>0% THEN 2750 ELSE Q%=POS(Q$,',',1) \ IF Q%<1% THEN 2710
2660 Y$=SEG$(Q$,Q%+1%,255%) \ Q$=SEG$(Q$,1%,Q%-1%)
2670 FOR J% 1% TO N% \ IF A$(J%)<>Q$ THEN 2680 ELSE K%=J% \ J%=N%
2680 NEXT J% \ IF K%=0% THEN 2690 ELSE K%=I%+K% \ M(K%)=VAL(Y$) \ GO TO 2710
2690 PRINT 'NO VARIABLE ';Q$;' FOUND.' \ GO TO 2630
2700 LINPUT ##.## \ GO TO 2640
2710 PRINT  ID    '.  FOR J%=1% TO N% \ PRINT A$(J%);E$;M(I%+J%);'.', \ NEXT J% \ PRINT
2720 PRINT #$(Q%-1%), \ FOR J%=1% TO N% \ PRINT #(I%+J%-1%), \ NEXT J% \ PRINT
2730 GO TO 2630
2740 PRINT 'UNACCEPTABLE INPUT' \ GO TO 2600
2750 IF Q%<1% THEN 2770 ELSE Q%=POS(A$(L%+1%),',',1) \ IF Q%<1% THEN 2780
2760 Q$=SEG$(A$(L%+1%),1%,Q%)+SEG$(Q$,2,255) \ GO TO 2770
2770 IF Q%<LEN(Q$) THEN 2780 ELSE A$(L%+1%)=SEG$(A$(L%+1%),PUT #8 I%,',',I%+1,255)
2780 A$(L%-1%)=Q$ \ PRINT 'NEW RECORD ID  ';Q$(L%-1%);',' \ GO TO 2600
2790 RETURN
2800 REM ### THIS IS VAX DELE ###
2810 IF Y$<>'' THEN 2840
2820 LINPUT 'SEQUENCE NUMBER(S) OF OBSERVATION(S) TO DELETE';Y$
2830 IF Y$='' THEN 3020
2840 PRINT 'MARKED FOR DELETION:'
2850 PRINT '  ID    ' \ FOR J%=1% TO N%  PRINT TAB(J%*24%);X$(J%);'[';U$(J%);']', \ NEXT J% \ PRINT
2860 IF Y$='' THEN 2820 ELSE Q%=POS(Y$,',',1) \ IF Q%=0% THEN Q%=LEN(Y$)+1%
2870 Q$=SEG$(Y$,1%,Q%-1%) \ Y$=SEG$(Y$,Q%+1%,255)
2880 Q%=POS(Q$,'-',1) \ IF Q%=0% THEN L%=VAL(Q$) \ K%=L% \ GOSUB 2940 \ GO TO 2860
2890 K%=I% \ IF Q%<1% THEN K%=VAL(SEG$(Q$,1%,Q%-1%))
2900 M%=M9% \ IF Q%<LEN(Q$) THEN M%=VAL(SEG$(Q$,Q%+1%,255))
2910 IF K%<1% THEN 3000 ELSE IF M%>N9% THEN 3000 ELSE IF K%>M% THEN 2930
2920 PRINT 'ILLEGAL SEQUENCE SET' \ GO TO 2860
2930 FOR L%=K% TO M% \ GOSUB 2940 \ NEXT L% \ GO TO 2860
2940 IF L%>N9% THEN 2990 ELSE IF L%<1% THEN 2990 ELSE I%=L%-1%
2950 IF A$(I%)=CHR$(1) THEN 3010
2960 IF L%<K% THEN 2970
2970 PRINT L%;') ';A$(I%), \ FOR J%=1% TO N% \ PRINT TAB(J%*24%);X(I%*N%+J%-1%), \ NEXT J% \ PRINT
2980 A$(I%)=CHR$(1) \ RETURN
2990 PRINT 'SEQUENCE RANGE IS FROM 1 TO';N9% \ RETURN
3000 GOSUB 2990 \ GO TO 2860

Ready
```

```
PHD          3-MAR-1983 08:48

3010 PRINT 'OBSERVATION #';L%;' ALREADY MARKED FOR DELETION' \ RETURN
3020 K%=0% \ M%=0% \ J%=0% \ FOR I%=0% TO N9%-1%
3030 IF K%=I% THEN 3060
3040 A$(K%)=A$(I%) \ FOR M1%=0% TO M%-1% \ X(J%+M1%)=X(M%+M1%) \ NEXT M1%
3050 IF A$(I%)=CHR$(1) THEN 3070
3060 K%=K%+1% \ J%=J%+M%
3070 M%=M%+M% \ NEXT I%
3080 PRINT N9%-K%;' OBSERVATION(S) DELETED; DATA RESEQUENCED.' \ N9%=K%
3090 RETURN
3100 REM ### THIS IS VAX GET ###
3110 X$=Y$ \ IF X$="" THEN PRINT 'DATASET '; \ LINPUT #0,X$ \ IF X$="" THEN 3430
3120 IF F$="" THEN F$=X$ \ GO TO 3140
3130 PRINT 'MERGING DATASET ';X$;' WITH WORKSET'
3140 ON ERROR GOTO 3350\IF POS(X$,'.',2)<2 THEN X$=TRM$(X$)+'.DAT'
3150 OPEN X$ FOR INPUT AS FILE #2 \ LINPUT #2,Q$ \ Q%=VAL(SEG$(Q$,1,POS(Q$,',',1)-1%))
3160 ON ERROR GO TO 3360
3170 Q$=SEG$(Q$,POS(Q$,',',1)+1,255) \ PRINT 'DATASET: ';X$,Q%;' VARIABLES PER OBSERVATION.' \ PRINT Q$
3180 IF M%=0% THEN M%=Q% \ C$=Q$ \ GO TO 3210
3190 IF M%<>Q% THEN 3200 ELSE FOR I%=1% TO M% \ LINPUT #2,Q$ \ NEXT I% \ GO TO 3230
3200 PRINT 'MISMATCH IN NUMBER OF VARIABLES PER OBSERVATION.' \ GO TO 3360
3210 FOR I%=1% TO M% \ INPUT #2,X$(I%),U$(I%),M(I%)
3220 PRINT X$(I%),U$(I%),M(I%) \ NEXT I%
3230 I%=N9%:N%
3240 ON ERROR GO TO 3360
3250 LINPUT #2,Q$ \ IF Q$="###EOF###" THEN 3390 ELSE N9%=N9%+1%\Q$=TRM$(Q$)
3260 IF SEG$(Q$,1,1)=" " THEN Q$=SEG$(Q$,2,255) \ GO TO 3260
3270 Q%=POS(Q$,',',1) \ IF Q%=0 THEN 3410 ELSE IF Q%=1% THEN Q$="1:"+Q$ \ Q%=3% \ GO TO 3300
3280 IF POS(SEG$(Q$,1%,Q%-1%),':',1)>0 THEN 3300
3290 Q$=SEG$(Q$,1%,Q%-1%)+":"+SEG$(Q$,Q%+1%,255) \ GO TO 3270
3300 A$(N9%-1%)=SEG$(Q$,1%,Q%-1%)
3310 FOR K%=I% TO I%+M%-1% \ Q$=SEG$(Q$,Q%+1%,255) \ Q%=POS(Q$,',',1%)
3320 IF Q%<>0% THEN 3330 ELSE Q%=LEN(Q$)+1%
3330 X(K%)=VAL(SEG$(Q$,1%,Q%-1%)) \ NEXT K%
3340 GO TO 3230
3350 IF ERR=5 THEN PRINT "NO SUCH DATASET ON FILE":X$="":GO TO 3420
3360 IF ERR=11 THEN RESUME 3390
3370 PRINT 'ERROR ';ERR;' IN LINE ';ERL;' IN GET' \ RESUME 3430
3380 CLOSE #2\RETURN
3390 PRINT "NOW";N9%;' OBSERVATION(S) IN WORKSET'
3400 CLOSE #2 \ PRINT 'FILE ';X$;' CLOSED.' \ IF Y$<>"" THEN 3430 ELSE GO TO 3110
3410 PRINT 'NO DATA IN OBSERVATION WITH ID:  ';Q$ \ GO TO 3250
3420 RESUME 3430
3430 ON ERROR GO TO 0\RETURN
3440 REM ### THIS IS VAX LIST ###
3450 L%=100%
3460 PRINT 'WORKSET:  ';F$;' AS OF ';DATE$(0);' AT ';TIME$(0) \ PRINT C$
3470 FOR I%=0% TO N9%-1%
3480 IF L%>99% THEN L%=0% \ GO TO 3500
3490 IF L%<50% THEN 3520 ELSE INPUT #0,Y$ \ L%=0%
3500 PRINT '    IDENT'; \ FOR J%=1% TO M% \ PRINT TAB(J%*16%+1);TRM$(X$(J%)); \ NEXT J% \ PRINT

Ready
```

```
PHD            3-MAR-1983 08:43

3610 PRINT '   [UNIT]>'; \ FOR J%=1% TO M% \ PRINT TAB(J%*16%);'[';TRM$(U$(J%));']'; \ NEXT J% \ PRINT
3620 PRINT I%+1%;'  ';A$(I%); \ FOR J%=1% TO M% \ PRINT TAB(J%*16%);X(I%*M%+J%-1%); \ NEXT J% \ PRINT
3630 L%=L%+1% \ NEXT I% \ RETURN
3640 REM *** THIS IS VAX PRIN ***
3650 C%=1% \ IF Y$<>'' THEN C%=ABS(VAL(Y$)) \ IF C%>10% THEN C%=10%
3660 L%=100%
3670 P9%=4% \ OPEN 'PHD.LIS' FOR OUTPUT AS FILE #P9% \ MARGIN #P9%,132
3680 PRINT #P9%,CHR$(12);C$;'   FILE:  ';F$;' AS OF ';DATE$(0);' AT ';TIME$(0)
3690 L%=100%
3700 PRINT #P9% \ FOR I%=0% TO N9%-1%
3710 IF L%<50% THEN 3740 ELSE L%=0%
3720 PRINT #P9%,'    ID    '; \ FOR J%=1% TO M% \ PRINT #P9%,TAB(J%*16%);TRM$(X$(J%)); \ NEXT J% \ PRINT #P9%
3730 PRINT #P9%,'  [UNIT]>'; \ FOR J%=1% TO M% \ PRINT #P9%,TAB(J%*16%);'[';TRM$(U$(J%));']'; \ NEXT J% \ PRINT #P9%
3740 PRINT #P9%,I%+1%;'  ';A$(I%); \ FOR J%=1% TO M% \ PRINT #P9%,TAB(J%*16%);X(I%*M%+J%-1%); \ NEXT J% \ PRINT #P9%
3750 L%=L%+1% \ NEXT I% \ L%=C%-1% \ IF L%>0% THEN 3580
3760 FOR I%=1% TO 50% \ PRINT #P9% \ NEXT I% \ CLOSE #P9%
3770 RETURN
3780 REM *** THIS IS VAX SAVE ***
3790 F1$='##'+'E'
3800 F3$='#.#####^^^^'
3810 F2$='RRRRRR'
3820 F4$='RRRRRR'+'RRRRRR'+'#.#####^^^^'
3830 IF Y$<>'' THEN 3860 ELSE PRINT 'WORKSET NAME:  ';F$;''
3840 LINPUT 'NEW DATASET NAME (or TO USE WORKSET NAME) ';Y$
3850 IF Y$='' THEN Y$=F$
3860 IF POS(Y$,'.',2)<2 THEN Y$=TRM$(Y$)+'.DAT'
3870 OPEN Y$ FOR OUTPUT AS FILE #2%,NOMARGIN #2
3880 PRINT 'CREATING NEW DATASET: ';Y$
3890 PRINT M%;' VARIABLES PER OBSERVATION.  ';N9%;' OBSERVATIONS.'
3900 FOR I%=1% TO M% \ PRINT USING F4$ X$(I%),',',U$(I%),',',M(I%) \ NEXT I%
3910 Q$=TIME$(0)+'"'+DATE$(0)+'"'+SEG$(C$,POS(C$,':',1)+1,255)
3920 PRINT #2 USING F1$,M%,',',Q$
3930 FOR I%=1% TO M% \ PRINT #2 USING F4$,X$(I%),',',U$(I%),',',M(I%) \ NEXT I%
3940 FOR I%=0% TO N9%-1% \ PRINT #2 USING F2$,A$(I%), \ J%=J%+M%
3950 FOR J%=J% TO J%-1%+M% \ PRINT #2,',';STR$(X(J%)); \ NEXT J% \ PRINT #2 \ NEXT I%
3960 PRINT #2,'***EOF***'\CLOSE #2 \ PRINT 'DATASET ';Y$;' SAVED.'
3970 RETURN
3980 REM *** THIS IS VAX SORT ***
3990 FOR I%=0% TO N9%-1% \ L%(I%)=I% \ NEXT I%
4000 SORTU$=''
4010 PRINT 'THERE ARE';M%;' VARIABLES PER OBSERVATION.'
4020 FOR J%=1% TO M% \ PRINT J%,X$(J%) \ NEXT J%
4030 PRINT
4040 PRINT 'YOU MAY SORT BASED ON THE VALUE OF ANY VARIABLE OR THE PSN:ID'
4050 PRINT 'USE 'ID' FOR THE SEQUENCE NUMBER OF THE PSN:ID'
4060 PRINT 'A POSITIVE SEQUENCE NUMBER IMPLIES ASCENDING SORT'
4070 PRINT 'A NEGATIVE SEQUENCE NUMBER IMPLIES DESCENDING SORT'
4080 PRINT ''CR' WILL IMPLEMENT THE SORT VECTORS THAT HAVE BEEN BUILT'
4090 PRINT  \ INPUT 'SEQUENCE NUMBER OF VARIABLE TO SORT ON';Y$
4000 IF POS(Y$,'ID',1)<1 THEN 4020 ELSE M%=1% \ IF POS(Y$,'-',1)>0 THEN M%=-1%

Ready
```

```
PHD          3-MAR-1983 08:43

4010 Y$='ID' \ GO TO 4060
4020 IF Y$='' THEN 4170 ELSE T%=VAL(Y$) \ S%=ABS(T%)
4030 IF S%<1% THEN 3910 ELSE IF S%>M% THEN 3910 ELSE M%=SGN(T%) \ L%=S%-1%
4040 Y$=X$(S%)
4050 IF M%>0% THEN SORTV$=SORTV$+'+' ELSE SORTV$=SORTV$+'-'
4060 SORTV$=SORTV$+Y$
4070 PRINT 'BUILDING SORT VECTOR ON KEY VARIABLE ';Y$
4080 J%=(3%+M%)/2%-1% \ T%=N9%+(M%-1%)/2%-1%
4090 F%=-1% \ FOR I%=J% TO T% \ K%=I%-M%
4100 IF Y$<>'ID' THEN 4130 ELSE X$=SEG$('         ',1,8-LEN(A$(L%(K%))))+A$(L%(K%))
4110 G$=SEG$('         ',1,8-LEN(A$(L%(I%))))+A$(L%(I%))
4120 IF X$<=G$ THEN 4150 ELSE 4140
4130 IF X(L%(K%)*N%+L%)<=X(L%(I%)*N%+L%) THEN 4150
4140 F%=L%(K%) \ L%(K%)=L%(I%) \ L%(I%)=F%
4150 NEXT I% \ IF F%>=0% THEN 4090
4160 PRINT 'VECTOR BUILT.' \ GO TO 3990
4170 PRINT 'SORT IN PROGRESS.' \ S%=0%
4180 PRINT 'COMPOSITE SORT VECTOR'\PRINT SORTV$
4190 PRINT '.' \ FOR I%=S% TO N9%-1% \ IF L%(I%)=I% THEN 4250
4200 S%=I% \ F%=I% \ T%=N9% \ GO TO 4220
4210 F%=L%(T%) \ IF F%=S% THEN F%=N9%
4220 L%=T%*N%-1% \ M%=F%*N%-1%
4230 A$(T%)=A$(F%) \ FOR J%=1% TO N% \ X(L%+J%)=X(M%+J%) \ NEXT J%
4240 L%(T%)=T% \ IF F%=N9% THEN 4190 ELSE T%=F% \ GO TO 4210
4250 NEXT I%
4260 PRINT 'SORT COMPLETED.' \ RETURN
4265 GOTO 6390\REM ENTER
4266 GOTO 6530\ REM MFP
4267 GOTO 6200\ REM SWAP
4270 GOSUB 4660 \ PRINT
4280 IF TERM$<>'4014' THEN PRINT 'MUST BE ON TEKTRONIX 4014' \ GO TO 4310
4290 IF N9%<2% THEN PRINT 'NEED AT LEAST TWO POINTS TO PLOT' \ GO TO 4310
4300 GO TO 4320
4310 SLEEP 1 \ RETURN
4320 PRINT \ PRINT CHR$(27);'!W E L C O M E   T O   P A D   P L O T   II'
4330 GOSUB 4450
4340 PRINT  \ PRINT 'TO LEAVE UNCHANGED, PRESS <CR>'
4350 X$=SEG$(C$,POS(C$,'!',1)+1,255) \ A$='PLOT TITLE' \ GOSUB 4420 \ T1$=X$
4360 IF LEN(T1$)>50 THEN T1$=SEG$(T1$,1,50) \ PRINT 'TITLE TO LONG.  SHORTENED TO :' \ PRINT T1$
4370 X$=L2$ \ A$='X LABEL' \ GOSUB 4420 \ L2$=X$
4380 X$=U2$ \ A$='UNIT' \ GOSUB 4420 \ U2$=X$
4390 X$=L1$ \ A$='Y LABEL' \ GOSUB 4420 \ L1$=X$
4400 X$=U1$ \ A$='UNIT' \ GOSUB 4420 \ U1$=X$
4410 GO TO 4670
4420 IF X$<>'' THEN 4430 ELSE X$='(NONE)'
4430 PRINT  \ PRINT TAB(16-LEN(A$));A$;':  ';X$; \ LINPUT B$ \ IF B$='' THEN RETURN
4440 X$=TRM$(B$) \ RETURN
4450 PRINT USING '   ### VARIABLES/OBSERVATION',M%
4460 PRINT USING '   #### OBSERVATIONS.',N9%
4470 PRINT  \ PRINT 'VARIABLES AVAILABLE:' \ PRINT  \ PRINT 'VARIABLE$          NAME                UNIT' \ PRINT
4480 FOR I%=1% TO N%
4490 PRINT USING '  ##        'CCCCCCCCC          'CCCCCCCCC',I%,X$(I%),U$(I%)
4500 NEXT I%

Ready
```

```
PHD           3-MAR-1983 08:44

4510 X%=1%
4520 A$="VARIABLE AS ABCISSA" \ D$=STR$(X%)\GOSUB 5210\IF E%>0% THEN 4520 ELSE X%=X
4530 IF X%<1% THEN 4520 ELSE IF X%>N% THEN 4520
4540 Y%=2%
4550 A$="VARIABLE AS ORDINATE"\D$=STR$(Y%)\GOSUB 5210\IF E%>0 THEN 4550 ELSE Y%=X
4560 IF Y%<1 THEN 4550 ELSE IF Y%>N% THEN 4550
4570 P1%=1%\A$="FIRST OBSERVATION TO PLOT"\D$=STR$(P1%)\GOSUB 5210\IF E%>0% THEN 4570 ELSE P1%=X
4580 IF P1%<1% THEN 4570 ELSE IF P1%>N9% THEN 4570
4590 P2%=N9% \ A$="LAST OBSERVATION TO PLOT"\D$=STR$(P2%) \ GOSUB 5210 \ IF E%>0% THEN 4590 ELSE P2%=X
4600 IF P2%>N9% THEN 4590
4610 IF P2%-P1%>1 THEN 4620 ELSE  PRINT \ PRINT "ILLEGAL POINT RANGE" \ GO TO 4570
4620 L1$=TRM$(X$(Y%)) \ L2$=TRM$(X$(X%)) \ U1$=TRM$(U$(Y%)) \ U2$=TRM$(U$(X%))
4630 PRINT "PLOTTING ";L1$;" IN ";U1$\PRINT "  VERSUS ";L2$;" IN ";U2$
4640 PRINT "OBSERVATIONS ";P1%;"-";P2%\PRINT\PRINT
4650 RETURN
4660 PRINT CHR$(27);CHR$(140) \ SLEEP 1 \ RETURN
4670 L%=(P1%-1%)*N%-1% \ X0=X(L%+X%) \ X9=X0 \ Y0=X(L%+Y%) \ Y9=Y0
4680 FOR L%=P1%-1% TO P2%-1% \ I%=L%*N%-1%
4690 IF X(I%+X%)>X9 THEN X9=X(I%+X%)
4700 IF X(I%+X%)<X0 THEN X0=X(I%+X%)
4710 IF X(I%+Y%)>Y9 THEN Y9=X(I%+Y%)
4720 IF X(I%+Y%)<Y0 THEN Y0=X(I%+Y%)
4730 L%(L%)=VAL(SEG$(A$(L%),1%,POS(A$(L%),"1",1)-1%))
4740 IF L%(L%)>0% THEN IF L%(L%)<13% THEN 4750 ELSE L%(L%)=9%
4750 NEXT L% \ DATA .1,.2,.5,.01, .02,.05,1,2,5
4760 B7%=0%
4770 IF ABS(X9)>ABS(X0) THEN 4780 ELSE IF X0=0 THEN 4790 ELSE K%=INT LOG10(ABS(X0))) \ GO TO 4800
4780 IF X9=0 THEN 4790 ELSE K%=INT(LOG10(ABS(X9))) \ GO TO 4800
4790 K%=0
4800 IF ABS(Y9)>ABS(Y0) THEN 4810 ELSE IF Y0=0 THEN 4820 ELSE K1%=INT(LOG10(ABS(Y0))) \ GO TO 4830
4810 IF Y9=0 THEN 4820 ELSE K1%=INT(LOG10(ABS(Y9))) \ GO TO 4830
4820 K1%=0
4830 K1=(Y9-Y0)/10^K1%/7 \ K=(X9-X0)/10^K%/9
4840 Y6=2 \ X6=2 \ RESTORE  \ FOR I%=0% TO 8%
4850 READ U \ IF ABS(K1-U)<ABS(K1-Y6) THEN Y6=U
4860 IF ABS(K-U)<ABS(K-X6) THEN X6=U
4870 NEXT I% \ RESTORE  \ T1=Y6*10^K1% \ T2=X6*10^K%
4880 IF B7%=1 THEN 5070
4890 IF Y0<0 THEN 4900 ELSE Y0=(INT(Y0/T1))*T1 \ GO TO 4910
4900 Y0=(INT(Y0/T1))*T1
4910 IF Y9<0 THEN 4920 ELSE Y9=(INT(Y9/T1)+1)*T1 \ GO TO 4930
4920 Y9=(INT(Y9/T1)+1)*T1
4930 IF X0<0 THEN 4940 ELSE X0=(INT(X0/T2))*T2 \ GO TO 4950
4940 X0=(INT(X0/T2))*T2
4950 IF X9<0 THEN 4960 ELSE X9=(INT(X9/T2)+1)*T2 \ GO TO 4970
4960 X9=(INT(X9/T2)+1)*T2
4970 PRINT  \ PRINT "AXIS LIMITS (TO LEAVE UNCHANGED, PRESS (CR))" \ B$="  .  'LLLLIMUM =  $.$$$^^^^"
4980 A$="X MIN" \ X=X0 \ GOSUB 5160 \ X0=X
4990 A$="X MAX" \ X=X9 \ GOSUB 5160 \ X9=X
5000 A$="Y MIN" \ X=Y0 \ GOSUB 5160 \ Y0=X

Ready
```

```
PHD           3-MAR-1983 08:44

5010 A$="Y MAX" \ X=Y9 \ GOSUB 5160 \ Y9=X
5020 B7%=1 \ GO TO 4770
5030 Y8=2 \ X8=2 \ FOR I%=0% TO 7%
5040 READ U \ IF ABS(K1-U)<ABS(K1-Y8) THEN Y8=U
5050 IF ABS(K-U)<ABS(K-X8) THEN X8=U
5060 NEXT I% \ RESTORE  \ T1=Y8/10^K1% \ T2=X8/10^K%
5070 PRINT  \ PRINT  \ PRINT "TIC MARK INTERVALS (TO LEAVE UNCHANGED, PRESS <CR>)" \ PRINT
5080 X=T2 \ A$="X" \ GOSUB 5180 \ T2=X \ X=T1 \ A$="Y" \ GOSUB 5180 \ T1=X
5090 IF (X9-X0)>2.01*T2 THEN 5100 ELSE T2=T2/10
5100 IF (Y9-Y0)>2.01*T1 THEN 5110 ELSE T1=T1/10
5110 X1=(4095-2*372)/(X9-X0)
5120 Y1=(3119-2*372)/(Y9-Y0)
5130 PRINT "MIN X = ";X0;TAB(40);"MAX X = ";X9
5140 PRINT "MIN Y = ";Y0;TAB(40);"MAX Y = ";Y9
5150 GO TO 5260
5160 PRINT  \ PRINT USING B$,A$,X; \ INPUT X$ \ IF X$<>"" THEN GOSUB 5230
5170 RETURN
5180 PRINT USING "      ":  #.###^^^^",A$,X; \ INPUT X$ \ IF X$<>"" THEN GOSUB 5230
5190 IF X<>0 THEN 5180
5200 RETURN
5210 PRINT A$;" [";D$;"] ";\LINPUT X$\X$=X$
5220 IF X$="" THEN X$=D$
5230 ON ERROR GO TO 5250 \ E%=0% \ X=VAL(X$)
5240 ON ERROR GO TO 0 \ RETURN
5250 PRINT "UNACCEPTABLE NUMERIC" \ X=0 \ E%=1% \ RESUME 5240
5260 D$="5" \ A$="SYMBOL SIZE (1-10)" \ GOSUB 5210 \ S=X
5270 IF S<1 THEN 5260 ELSE IF S>10 THEN 5260 ELSE X1%=X% \ Y1%=Y%
5280 GOSUB 4660
5290 C0$="" \ C1$="" \ C1$=CHR$(27)+CHR$(104)+CHR$(29) \ GOSUB 5640
5300 NOMARGIN $0 \ X%=372 \ Y%=558 \ GOSUB 5520
5310 X%=3723 \ GOSUB 5520 \ Y%=2933 \ GOSUB 5520
5320 X%=372 \ GOSUB 5520 \ Y%=558 \ GOSUB 5520
5330 C1$=CHR$(27)+CHR$(96)+CHR$(29) \ GOSUB 5640
5340 N=3351/T2/X1 \ T3=T2*X1 \ T4=T1*Y1 \ N=N+.01
5350 FOR I=0 TO INT(N) \ X%=372+I*T3 \ Y%=558 \ GOSUB 5520
5360 Y%=558+74 \ GOSUB 5520 \ C1$=CHR$(29) \ GOSUB 5640 \ Y%=2859 \ GOSUB 5520 \ Y%=2933 \ GOSUB 5520
5370 C1$=CHR$(29) \GOSUB 5640
5380 Y%=558 \ X%=X%-10 \ GOSUB 5520 \ C1$=CHR$(31)+CHR$(27)+CHR$(59)+CHR$(10)+CHR$(10)
5390 GOSUB 5640 \ Z=X0+I*T2 \ IF ABS(Z)<=.01*T2 THEN Z=0
5400 I$=STR$(Z/10^K%) \ IF LEN(I$)<=7 THEN 5410 ELSE I$=SEG$(I$,1,7)
5410 FOR J%=1 TO INT(LEN(I$)/2) \ C1$=CHR$(8) \ GOSUB 5640 \ NEXT J%
5420 C1$=I$+CHR$(29) \ GOSUB 5640 \ NEXT I
5430 N=2375/T1/Y1 \ C1$=CHR$(29) \ GOSUB 5640 \ N=N+.01
5440 FOR I=0 TO INT(N) \ Y%=558+I*T4 \ X%=372 \ GOSUB 5520
5450 X%=372+74 \ GOSUB 5520 \ C1$=CHR$(29) \ GOSUB 5640 \ X%=3649 \ GOSUB 5520 \ X%=3723 \ GOSUB 5520
5460 C1$=CHR$(29) \ GOSUB 5640 \ X%=372 \ Y%=Y%-15 \ GOSUB 5520
5470 C1$=CHR$(31)+CHR$(27)+CHR$(59) \ GOSUB 5640 \ Z=Y0+I*T1 \ IF ABS(Z)<=.01*T1 THEN Z=0
5480 I$=STR$(Z/10^K1%) \ IF LEN(I$)<=7 THEN 5490 ELSE I$=SEG$(I$,1,7)
5490 FOR J%=0 TO INT(LEN(I$)/2)+2 \ C1$=CHR$(8) \ GOSUB 5640 \ NEXT J%
5500 C1$=I$+CHR$(29) \ GOSUB 5640 \ NEXT I

Ready
```

```
PMD          3-MAR-1983 08:45

5610 GO TO 5670
5620 C1%=INT(Y%/128)+32
5630 C2%=(Y%-INT(Y%/4)*4)*4+X%-INT(X%/4)*4+96
5640 C3%=INT((Y%-INT(Y%/128)*128)/4)+96
5650 C4%=INT(X%/128)+32
5660 C5%=INT((X%-INT(X%/128)*128)/4)+64
5670 IF C1%=B1% THEN 5680 ELSE C1$=C1$+CHR$(C1%)
5680 IF C2%=B2% THEN 5690 ELSE C1$=C1$+CHR$(C2%)
5690 IF C3%=B3% THEN IF C2%=B2% THEN IF C4%=B4% THEN 5610
5800 C1$=C1$+CHR$(C3%)
5610 IF C4%=B4% THEN 5620 ELSE C1$=C1$+CHR$(C4%)
5620 C1$=C1$+CHR$(C5%)
5630 B1%=C1% \ B2%=C2% \ B3%=C3% \ B4%=C4%
5640 IF LEN(C0$)+LEN(C1$)>255 THEN 5660
5650 C0$=C0$+C1$ \ C1$="" \ RETURN
5660 C0$=CHR$(5)+C0$+CHR$(27) \ PRINT C0$ \ C0$=C1$ \ C1$="" \ RETURN
5670 FOR I1%=P1%-1% TO P2%-1% \ I%=I1%*M%-1%
5680 X=X(I%+X1%) \ Y=X(I%+Y1%) \ A%=L%(I1%)
5690 IF X>X9 THEN A%=0 \ X=X9
5700 IF X<X0 THEN A%=0 \ X=X0
5710 IF Y>Y9 THEN A%=0 \ Y=Y9
5720 IF Y<Y0 THEN A%=0 \ Y=Y0
5730 ON A%+1% GOSUB 6160,5950,5970,6000,6010,6040,6080,6110,6140,6150,6150,6150,6150,6150
5740 NEXT I1% \ Y$="" \ IF L1$<>"(NONE)" THEN Y$=Y$+L1$+" / "
5750 IF K1%<>0% THEN Y$=Y$+"10*"+STR$(K1%)+" "
5760 IF U1$="(NONE)" THEN 5770 ELSE Y$=Y$+U1$
5770 IF Y$=" " THEN 5820 ELSE IF SEG$(Y$,LEN(Y$)-2,LEN(Y$))=" / " THEN Y$=SEG$(Y$,1,LEN(Y$)-3)
5780 J%=LEN(Y$)/2 \ C1$=CHR$(29) \ GOSUB 5640 \ X%=15 \ Y%=1746 \ GOSUB 5520 \ C1$=CHR$(31)+CHR$(27)+CHR$(57)
5790 GOSUB 5640
5800 FOR I%=1 TO J% \ C1$=CHR$(139) \ GOSUB 5640 \ NEXT I% \ Y$=Y$
5810 FOR I%=1 TO LEN(Y$) \ C1$=SEG$(Y$,I%,I%)+CHR$(10)+CHR$(8) \ GOSUB 5640 \ NEXT I%
5820 C1$=CHR$(29) \ GOSUB 5640 \ X%=2047 \ Y%=200 \ GOSUB 5520 \ C1$=CHR$(31)+CHR$(27)+CHR$(57) \ GOSUB 5640
5830 X$="" \ IF L2$<>"(NONE)" THEN X$=X$+L2$+" / "
5840 IF K%<>0% THEN  X$=X$+"10*"+STR$(K%)+" "
5850 IF U2$="(NONE)" THEN 5860 ELSE X$=X$+U2$
5860 IF X$=" " THEN 5880 ELSE IF SEG$(X$,LEN(X$)-2,LEN(X$))=" / " THEN X$=SEG$(X$,1,LEN(X$)-3)
5870 X$=X$+J%+LEN(X$)/2 \ FOR I%=1 TO J% \ C1$=CHR$(8) \ GOSUB 5640 \ NEXT I% \ C1$=X$ \ GOSUB 5640
5880 C1$=CHR$(29) \ GOSUB 5640 \ X%=2048 \ Y%=15 \ GOSUB 5520 \ J%=LEN(T1$)/2 \ C1$=CHR$(31)+CHR$(27)+CHR$(56)
5890 GOSUB 5640 \ FOR I%=1 TO J% \ C1$=CHR$(8) \ GOSUB 5640 \ NEXT I%
5900 GOSUB 5640 \ C1$=T1$ \ GOSUB 5640 \ C1$=CHR$(27)+CHR$(59)
5910 GOSUB 5640 \ IF LEN(C0$)>0 THEN PRINT CHR$(15);C0$;CHR$(27)
5920 INPUT C1$ \MARGIN #0,132\PRINT CHR$(27);"I"\GOSUB 4660\ RETURN
5930 X%=X%+C1$ \ Y%=Y%+D1$ \ GOSUB 5520 \ C=0 \ D=0 \ RETURN
5940 X%=X1%(X-X0)+372 \ Y%=Y1%(Y-Y0)+558\ GOSUB 5520 \ RETURN
5950 GOSUB 6190 \ GOSUB 5940 \ C=5.29 \ GOSUB 5930 \ C=-10.57 \ GOSUB 5930 \ C=5.29
5960 GOSUB 5930 \ D=5.29 \ GOSUB 5930 \ D=-10.57 \ GOSUB 5930 \ D=5.29 \ GOSUB 5930 \ RETURN
5970 GOSUB 6190 \ GOSUB 5940 \ C=5.29 \ D=5.29 \ GOSUB 5930 \ C=-10.57 \ D=-10.57
5980 GOSUB 5930 \ GOSUB 6190 \ C=10.57 \ GOSUB 5930 \ C=-10.57
5990 D=10.57 \ GOSUB 5930 \ C=5.29 \ D=-5.29 \ GOSUB 5930 \ RETURN
6000 GOSUB 5960 \ GOSUB 5970 \ RETURN

Ready
```

```
PWD           3-MAR-1983 08:45

6010 GOSUB 6190 \ GOSUB 5940 \ GOSUB 5930 \ C=5.29 \ GOSUB 6190 \ GOSUB 5930 \ D=5.29
6020 GOSUB 5930 \ C=-10.57 \ GOSUB 5930 \ D=-10.57 \ GOSUB 5930 \ C=10.57 \ GOSUB 5930
6030 D=5.29 \ GOSUB 5930 \ C=-5.29 \ GOSUB 6190 \ GOSUB 5930 \ RETURN
6040 GOSUB 6190 \ GOSUB 5940 \ GOSUB 5930 \ GOSUB 6190 \ C=7.48 \ GOSUB 5930 \ C=-7.48
6050 D=7.48 \ GOSUB 5930 \ C=-7.48 \ D=-7.48 \ GOSUB 5930 \ C=7.48
6060 D=-7.48 \ GOSUB 5930 \ C=7.48 \ D=7.48 \ GOSUB 5930 \ C=-7.48
6070 GOSUB 6190 \ GOSUB 5930 \ RETURN
6080 GOSUB 6190 \ GOSUB 5940 \ GOSUB 5930 \ GOSUB 6190 \ D=7 \ GOSUB 5930 \ C=-6.14
6090 D=-10.57 \ GOSUB 5930 \ C=12.29 \ GOSUB 5930 \ C=-6.14 \ D=10.57 \ GOSUB 5930
6100 D=-7 \ GOSUB 6190 \ GOSUB 5930 \ RETURN
6110 GOSUB 6190 \ GOSUB 5940 \ GOSUB 5930 \ GOSUB 6190 \ D=-7 \ GOSUD 5930 \ C=6.14
6120 D=10.57 \ GOSUB 5930 \ C=-12.29 \ GOSUB 5930 \ C=6.14 \ D=-10.57 \ GOSUB 5930
6130 GOSUB 6190 \ D=7 \ GOSUB 5930 \ RETURN
6140 GOSUB 6190 \ GOSUB 5940 \ RETURN
6150 C1$=CHR$(27)+CHR$(L%(I1%)+95) \ GOSUB 5640 \ GOSUB 5940 \ C1$=CHR$(27)+CHR$(9C) \ GOSUB 5640 \ RETURN
6160 GOSUB 6190 \ GOSUB 5940
6170 C=5.29 \ D=5.29 \ GOSUB 5930 \ C=-10.57 \ GOSUB 5930 \ C=10.57
6180 D=-10.57 \ GOSUB 5930 \ C=-10.57 \ GOSUB 5930 \ C=5.29 \ D=5.29 \ GOSUB 5930 \ RETURN
6190 C1$=CHR$(29)+CHR$(27)+CHR$(104)\ GOSUB 5640 \ RETURN
6200 REM !!! THIS IS VAX SWAP !!!
6210 GOSUB 6330
6220 INPUT 'NUMBERS OF THE VARIABLES TO EXCHANGE',X%,Y%
6230 IF X%)N% THEN 6210 ELSE IF X%<1% THEN 6210
6240 IF Y%)N% THEN 6210 ELSE IF Y%<1% THEN 6210
6250 FOR I%=0% TO N9%-1% \ K%=I%*N%-1%
6260 X=X(K%+X%)\X(K%+X%)=X(K%+Y%)\X(K%+Y%)=X
6270 NEXT I%
6280 X$=X$(X%)\X$(X%)=X$(Y%)\X$(Y%)=X$
6290 X$=U$(X%)\U$(X%)=U$(Y%)\U$(Y%)=X$
6300 X=M(X%)\M(X%)=M(Y%)\M(Y%)=X
6310 GOSUB 6330
6320 RETURN
6330 PRINT 'VARIABLES IN BASE ';F$
6340 PRINT @%,'ID'
6350 FOR I%=1% TO N%
6360 PRINT I%,X$(I%),U$(I%),M(I%)
6370 NEXT I%
6380 RETURN
6390 REM !!! THIS IS VAX ENTER !!!
6400 GOSUB 6330
6410 INPUT 'NUMBER OF VARIABLE TO ENTER) ',X%
6420 IF X%)N% THEN 6400 ELSE IF X%<0% THEN 6400
6430 IF X%=0% THEN 6600
6440 PRINT 'ENTER NEW VALUES FOR ';X$(X%)
6450 ON ERROR GOTO 6520
6460 FOR I%=0% TO N9%-1% \ K%=I%*N%-1%
6470 PRINT I%+1%;'  ';A$(I%);'  ';X(K%+X%);
6480 INPUT 'NEW VALUE (CR FOR NO CHANGE)) ',X$
6490 IF X$<>'' THEN X(K%+X%)=VAL(X$)
6500 NEXT I% \ ON ERROR GO TO 0

Ready
```

```
PWD           3-MAR-1983 08:45

6510 RETURN
6520 PRINT 'ILLEGAL NUMBER ' \ RESUME 6470
6530 REM VAX MULTI FILE PROCESSOR
6540 INPUT 'COMMAND FILE NAME',FILE$
6550 IF POS(FILE$,'.',1)<1 THEN FILE$=FILE$+'.PWD'
6560 OPEN FILE$ FOR INPUT AS FILE #1
6570 LINPUT #1,Q$ \PRINT FILE$;') ';Q$\ IF Q$='EXIT' THEN CLOSE #1\RETURN
6580 GOSUB 310
6590 GO TO 6570
6600 REM ### ADDENDUM TO VAX ENTER (LINE 5170) ###
6610 PRINT 'ENTER NEW PSN:ID'S'
6620 LINPUT 'IF ALL NEW PSN:ID'S ARE TO BE ALIKE, ENTER NEW PSN:ID HERE (OTHERWISE CR)';A1$
6630 IF A1$<>'' THEN FOR I%=0% TO N9%-1% \ A$(I%)=A1$ \ NEXT I% \ PRINT 'DONE' \ GOTO 6510
6640 FOR I%=0% TO N9%-1%
6650 PRINT 'OBSERVATION ';I%+1%;': PRESENT PSN:ID IS ';A$(I%);'; ENTER NEW PSN:ID (CR FOR NO CHANGE)';
6660 INPUT A1$ \ IF A1$<>'' THEN A$(I%)=A1$
6670 NEXT I% \ GOTO 6510
6680 REM ### END OF ADDENDUM TO FILL
6690 END

Ready
```

Listing of the USER Envelope Program for the VAX

```
USER          3-MAR-1983 08:40

10 COM (PHDSVS) X(1000),AS(500)=8,XS(9),US(9),M(9),L%(500),N%,N9%,CS=80,FS=40,U9S=16,YS=80
20 PRINT 'USER PROGRAM: ';TRM$(YS) \ ON ERROR GO TO 140
30 OPEN 'USER.TMP' FOR INPUT AS FILE #2, SEQUENTIAL VARIABLE
40 GET #2 \ MOVE FROM #2,YS=80 \ GET #2 \ MOVE FROM #2,CS=80
50 GET #2 \ MOVE FROM #2,N%,N9%,FS=40,U9S=16\PRINT N%;'VARIABLES'
55 PRINT N9%;'POINTS' \ PRINT 'FILE IS ';FS \ PRINT CS \ PRINT 'USER ARG: ';YS
60 FOR I%=1% TO N%
70 GET #2 \ MOVE FROM #2,XS(I%)=16,US(I%)=16,M(I%) \ NEXT I%
80 FOR I%=0% TO N9%-1% STEP 5% \ GET #2
90 MOVE FROM #2,AS(I%)=8,AS(I%+1%)=8,AS(I%+2%)=8,AS(I%+3%)=8,AS(I%+4%)=8
100 NEXT I%
110 FOR I%=0% TO N9%&N%-1% STEP 10% \ GET #2
120 MOVE FROM #2,X(I%),X(I%+1%),X(I%+2%),X(I%+3%),X(I%+4%),X(I%+5%),X(I%+6%),X(I%+7%),X(I%+8%),X(I%+9%)
130 NEXT I% \ CLOSE #2 \KILL 'USER.TMP'\ GOTO 1000
140 IF ERR<>5 THEN 170
150 PRINT 'NO DATA BASE TRANSFERRED FROM PHD--EXITING TO PHD'
160 RESUME 190
170 PRINT 'ERROR (';ERR;') IN LINE ';ERL;' IN MODULE ';ERN$
175 PRINT ERT$(ERR)
180 RESUME 190
190 SLEEP 1 \ U8$=TRM$(U9$)+'PHD' \ CHAIN U8$
999 REM !!!!!!!!!!!!!!!!!!!!S T A R T  U S E R  P R O G R A M !!!!!!!!!!!!!!!!!
29999 REM !!!!!!!!!!!!!!!!!!!!!!!!!E N D  U S E R  P R O G R A M !!!!!!!!!!!!!!!
30000 OPEN 'USER.TMP' FOR OUTPUT AS FILE #2,RECORDSIZE 80%, SEQUENTIAL VARIABLE
30010 MOVE TO #2,YS=80 \ PUT #2,COUNT 80% \ MOVE TO #2,CS=80 \ PUT #2,COUNT 80%
30020 MOVE TO #2,N%,N9%,FS=40,U9S=16\ PUT #2, COUNT 64%
30030 FOR I%=1% TO N%
30040 MOVE TO #2,XS(I%)=16,US(I%)=16,M(I%) \ PUT #2,COUNT 40%\ NEXT I%
30050 FOR I%=0% TO N9%-1% STEP 5%
30060 MOVE TO #2,AS(I%)=8,AS(I%+1%)=8,AS(I%+2%)=8,AS(I%+3%)=8,AS(I%+4%)=8
30070 PUT #2,COUNT 40% \ NEXT I%
30080 FOR I%=0% TO N9%&N%-1% STEP 10%
30090 MOVE TO #2,X(I%),X(I%+1%),X(I%+2%),X(I%+3%),X(I%+4%),X(I%+5%),X(I%+6%),X(I%+7%),X(I%+8%),X(I%+9%)
30100 PUT #2,COUNT 80% \ NEXT I% \ CLOSE #2
30110 U8$=TRM$(U9$)+'PHD' \ CHAIN U8$
30120 END

Ready
```

Sample PHD USER Program for the VAX

The FIT Option written as a USER program

```
FIT          3-MAR-1983 08:38

10 COM (PHDSYS) X(1000),A$(500)*8,X$(9),U$(9),M(9),L%(500),N%,N9%,C$*80,F$*40,U9$*16,Y$*80
20 PRINT 'USER PROGRAM: ';TRM$(Y$) \ ON ERROR GO TO 140
30 OPEN 'USER.TMP' FOR INPUT AS FILE #2, SEQUENTIAL VARIABLE
40 GET #2 \ MOVE FROM #2,Y$*80 \ GET #2 \ MOVE FROM #2,C$*80
50 GET #2 \ MOVE FROM #2,N%,N9%,F$*40,U9$*16\PRINT N%;'VARIABLES'
55 PRINT N9%;'POINTS' \ PRINT 'FILE IS ';F$ \ PRINT C$ \ PRINT 'USER ARG: ';Y$
60 FOR I%*1% TO N%
70 GET #2 \ MOVE FROM #2,X$(I%)*16,U$(I%)*16,M(I%) \ NEXT I%
80 FOR I%*0% TO N9%-1% STEP 5% \ GET #2
90 MOVE FROM #2,A$(I%)*8,A$(I%+1%)*8,A$(I%+2%)*8,A$(I%+3%)*8,A$(I%+4%)*8
100 NEXT I%
110 FOR I%*0% TO N9%\N%-1% STEP 10% \ GET #2
120 MOVE FROM #2,X(I%),X(I%+1%),X(I%+2%),X(I%+3%),X(I%+4%),X(I%+5%),X(I%+6%),X(I%+7%),X(I%+8%),X(I%+9%)
130 NEXT I% \ CLOSE #2 \KILL 'USER.TMP'\ GOTO 1000
140 IF ERR<>5 THEN 170
150 PRINT 'NO DATA BASE TRANSFERRED FROM PHD--EXITING TO PHD'
160 RESUME 190
170 PRINT 'ERROR (';ERR;') IN LINE ';ERL;' IN MODULE ';ERN$
175 PRINT ERT$(ERR)
180 RESUME 190
190 SLEEP 1 \ U8$*TRM$(U9$)+'PHD' \ CHAIN U8$
999 REM %%%%%%%%%%%%%%%%%%S T A R T  U S E R   P R O G R A M %%%%%%%%%%%%%%%%%
1000 DIM PN$(16),PD$(16),PU$(16),PV(16)
1010 DECLARE REAL H(16,16),HI(16,16),R(16),D(16),C(16)
1020 P9%*16%
1030 ON ERROR GOTO 1830
1040 DECODE$*'ABCDEFGHI0123456789L'
1050 P%*0% \ P$*EDIT$(Y$,156)\IF P$*'' THEN LINPUT 'PARAMETER FILE';P$
1060 P$*EDIT$(P$,156)
1070 IF P$*'' THEN 1200 ELSE IF POS(P$,'.',1)<1 THEN P$*P$+'.PAR'
1080 OPEN P$ FOR INPUT AS FILE #2
1090 LINPUT #2,TITLE$
1100 PRINT TITLE$
1110 IF P%>*P9% THEN 1820
1120 INPUT #2,PN$(P%+1%),PU$(P%+1%),PV(P%+1%),PD$(P%+1%) \ P%*P%+1%
1130 GO TO 1110
1140 PRINT\PRINT 'PARAMETER TABLE' \ PRINT
1150 PRINT 'NR  NAME';TAB(21);'UNITS';TAB(39);'EQ DECODE';TAB(56);'INIT VALUE';TAB(76);'ADJ'
1160 PRINT \ RETURN
1170 PRINT I%;TAB(4);PN$(I%);TAB(20);'[';PU$(I%);']';TAB(38);'<';PD$(I%);'>';TAB(56);PV(I%);
1180 IF SEG$(PD$(I%),1,1)*'-' THEN PRINT TAB(76);' N' ELSE PRINT TAB(76);' Y'
1190 RETURN
1200 P8%*0 \ FOR I%*1% TO P% \ C$*SEG$(PD$(I%),1%,1%)
1210 IF C$*'-' THEN 1220 ELSE IF C$<>'+'THEN PD$(I%)*'-'+PD$(I%)
1220 IF SEG$(PD$(I%),1%,1%)<>'-' THEN P8%*P8%+1%
1230 NEXT I%
1240 PRINT\PRINT P%;'PARAMETERS.  ';P8%;' ADJUSTABLE.'\PRINT 'SORTING...'
1250 J%*-1%\FOR I%*2% TO P% \ IF ASCII(PD$(I%))*ASCII(PD$(I%-1%)) THEN 1300

Ready
```

```
FIT          3-MAR-1983 09:39                                Ready

1260 XS=PNS(I%)\PNS(I%)=PNS(I%-1%)\PNS(I%-1%)=XS
1270 XS=PUS(I%)\PUS(I%)=PUS(I%-1%)\PUS(I%-1%)=XS
1280 XS=PDS(I%)\PDS(I%)=PDS(I%-1%)\PDS(I%-1%)=XS
1290 X=PU(I%)\PU(I%)=PU(I%-1%)\PU(I%-1%)=X\J%=I%
1300 NEXT I% \ IF J%>-1% THEN 1250
1310 GOSUB 1140\FOR I%=1% TO P%\GOSUB 1170
1320 X%=LEN(PDS(I%))-1% \ IF X%>0% THEN 1340 ELSE PRINT 'HAS NULL DECODE'
1330 PDS(I%)='-'+STRINGS(N%,ASCII('0'))\ PRINT 'ASSUMING DECODE ';PDS(I%)
1340 IF X%<N% THEN PRINT 'ACCESSES VARIABLES 1';STR$(-X%) \ GO TO 1390
1350 IF X%=N% THEN 1390
1360 PRINT 'DECODE SPECIFIES NON-EXISTENT VARIABLES ';STR$(N%+1%);STR$(-X%)
1370 PRINT 'DECODE CHANGED FROM (';PDS(I%);') TO (';
1380 PDS(I%)=SEGS(PDS(I%),1%,N%+1%) \ PRINT PD(I%);')'
1390 FOR J%=2% TO LEN(PDS(I%))
1400 IF POS(DECODES,SEGS(PDS(I%),J%,J%),1)>0% THEN 1440
1410 PRINT 'USES ILLEGAL DECODE ELEMENT '';SEGS(PDS,J%,J%);''
1420 PRINT 'ILLEGAL ELEMENT REPLACED BY '0''
1430 PDS(I%)=SEGS(PDS(I%),1%,J%-1%)+'0'+SEGS(PDS(I%),J%+1%,255%)
1440 NEXT J%
1450 NEXT I%
1460 PRINT 'BEGIN FIT:'
1470 TITLES=FNAS('TITLE:  ',TITLES)
1480 TOL=FNA('DELTA SIGMA FOR FIT CONVERGENCE','1E-7')
1490 MAT H=ZER(P%,P%) \ MAT HI=ZER(P%,P%)
1500 OLDSIGMA=0 \ ITN%=0%
1510 MAT C=ZER(P%) \ MAT R=ZER(P%) \ SSQ=0
1520 POINTS%=0%\FOR I%=0% TO M9%-1% \ K%=I%+N%-1%
1530 MAT D=CON(P%)
1540 FOR J%=1% TO P%
1550 AS=SEGS(PDS(J%),2%,255%) \ IF LEN(AS)<N% THEN L9%=LEN(AS) ELSE L9%=N%
1560 FOR L%=1% TO L9% \D(J%)=D(J%)*FND(SEGS(AS,L%,L%),L%)\NEXT L%
1570 NEXT J%
1580 IF ITN%>0% THEN 1630
1590 FOR J%=1% TO P% \ FOR L%=J% TO P% \ H(J%,L%)=H(J%,L%)+D(J%)*D(L%)
1600 NEXT L% \ NEXT J%
1610 FOR L%=1% TO P% \ FOR J%=L%+1% TO P% \ H(J%,L%)=H(L%,J%)
1620 NEXT J% \ NEXT L%
1630 D(0%)=0 \ FOR J%=1% TO P%\D(0%)=D(0%)+PU(J%)*D(J%)\ NEXT J%
1640 SSQ=SSQ+D(0%)*D(0%)
1650 FOR J%=1% TO P% \ R(J%)=R(J%)-D(0%)*D(J%) \ NEXT J%
1660 POINTS%=POINTS%+1%
1670 NEXT I%
1680 IF ITN%=0% THEN MAT HI=INV(H)
1690 FOR J%=1% TO P% \ C(J%)=0 \ FOR L%=1% TO P% \C(J%)=C(J%)+HI(J%,L%)*R(L%)
1700 NEXT L% \ NEXT J%
1710 FOR J%=1% TO P% \ PU(J%)=PU(J%)+C(J%)\ NEXT J%
1720 OLDSIGMA=SIGMA \SIGMA=SQR(SSQ/POINTS%)
1730 IF ITN%=0% THEN OLDSIGMA=-SIGMA
1740 IF 2*(SIGMA-OLDSIGMA)<=(OLDSIGMA+SIGMA)*TOL THEN 2040
1750 GOSUB 1760 \ GO TO 1510
1760 ITN%=ITN%+1%\ PRINT 'ITERATION ';ITN%;' SIGMA = ';SIGMA
1770 IF (OLDSIGMA+SIGMA)>0 THEN DELSIG=2*(SIGMA-OLDSIGMA)/(SIGMA+OLDSIGMA) ELSE DELSIG=99
1780 PRINT POINTS%;' POINTS USED.  DELTA SIGMA = ';DELSIG \ PRINT
1790 PRINT 'ADJUSTABLE PARAMETERS CORRECTED VALUE'\ FOR I%=1% TO P%
1800 PRINT PNS(I%);' = ';PU(I%);'    ';\ IF MAR(0)-CCPOS%(0)<40 THEN PRINT
```

```
FIT          3-MAR-1983 08:38

1810 NEXT I% \ PRINT \ RETURN
1820 PRINT 'MAXIMUM PARAMETERS EXCEEDED ';P9% \ GOTO 1200
1830 IF ERR=11 AND ERL=1120 THEN PRINT 'SIZEOF##'\RESUME 1200
1840 IF ERR=5 AND ERL=1080 THEN PRINT 'NO SUCH FILE> ';P$ \ RESUME 1030
1850 IF ERL>=1580 AND ERL<1670 THEN PRINT 'OBSERVATION ';I%;' NOT USED'\RESUME 1670
1860 PRINT 'ERROR';ERR;'IN LINE';ERL;' OF MODULE ';ERN$
1870 PRINT ERT$(ERR) \ RESUME 2150
1880 DEF FND(C$,X%)
1890 ON ERROR GO BACK
1900 Z%=POS(DECODE$,C$,1)
1910 IF Z%>19% THEN 1940
1920 IF Z%>9% THEN X=X(K%+X%)^(Z%-10%) ELSE X=1/X(K%+X%)^Z%
1930 FND=X \ FNEXIT
1940 IF Z%=20% THEN X=LOG(X(K%+X%)) \ GO TO 1930
1950 FNEND
1960 DEF FNA(MSG$,DEF$)
1970 PRINT MSG$;' [';DEF$;']'; \LINPUT X$ \ IF X$='' THEN X$=DEF$
1980 ON ERROR GOTO 1990 \ FNA=VAL(X$) \ FNEXIT
1990 FNA=VAL(DEF$) \ RESUME 2000
2000 FNEND
2010 DEF FNA$(MSG$,DEF$)
2020 PRINT MSG$;' [';DEF$;']';\LINPUT X$\IF X$='' THEN X$=DEF$
2030 FNA$=X$ \ FNEND
2040 GOSUB 1760
2050 PRINT 'ITERATION ';ITN%;' CONVERGENCE OCCURRED.'
2060 LINPUT 'HIT CARRIAGE RETURN TO CONTINUE',X$
2070 PRINT CHR$(27);CHR$(140);CHR$(27);'9' \ SLEEP 1\ PRINT \ PRINT  \PRINT\PRINT F$\PRINT TITLE$\PRINT
2080 PRINT 'SIGMA = ';SIGMA;'  DELTA SIGMA = ';DELSIG \ PRINT
2090 PRINT 'PARAMETER';TAB(17);'UNITS';TAB(35);'VALUE AND UNCERTAINTY' \ PRINT
2100 FOR I%=1% TO P%
2110 IF I%>P8% THEN X$=' NOT ADJUSTABLE' ELSE X$='(+/-) '+FORMAT$(SIGMA*SQR(ABS(HI(I%,I%))),'8.8^^^^')
2120 PRINT PN$(I%);TAB(17);'[';PU$(I%);']';TAB(35);PV(I%);TAB(55);X$
2130 NEXT I% \ PRINT
2140 PRINT 'FIT PROGRAM TERMINATION--EXIT TO PHD WITH A cr'\LINPUT Y$
2150 PRINT CHR$(27);':FIT PROGRAM TERMINATING' \ SLEEP 1
29999 REM ##########################HD  USER  PROGRAM##################
30000 OPEN 'USER.TMP' FOR OUTPUT AS FILE #2,RECORDSIZE 80%, SEQUENTIAL VARIABLE
30010 MOVE TO #2,Y8=80 \ PUT #2,COUNT 80% \ MOVE TO #2,C8=80 \ PUT #2,COUNT 80%
30020 MOVE TO #2,N%,N9%,F8=40,U9$=16\ PUT #2, COUNT 64%
30030 FOR I%=1% TO N%
30040 MOVE TO #2,X8(I%)=16,U8(I%)=16,M(I%) \ PUT #2,COUNT 40%\ NEXT I%
30050 FOR I%=0% TO N9%-1% STEP 5%
30060 MOVE TO #2,A8(I%)=8,A8(I%+1%)=8,A8(I%+2%)=8,A8(I%+3%)=8,A8(I%+4%)=8
30070 PUT #2,COUNT 40% \ NEXT I%
30080 FOR I%=0% TO N9%*N%-1% STEP 10%
30090 MOVE TO #2,X(I%),X(I%+1%),X(I%+2%),X(I%+3%),X(I%+4%),X(I%+5%),X(I%+6%),X(I%+7%),X(I%+8%),X(I%+9%)
30100 PUT #2,COUNT 80% \ NEXT I% \ CLOSE #2
30110 U8$=TRM$(U9$)+'PHD' \ CHAIN U8$
30120 END

Ready
```

SUMMARY SHEET FOR PHD OPTION COMMANDS

ADD
no arguments are allowed.

BASE
BASE NEW
BASE <dataset>
<dataset> is the name of a valid PHD dataset.

CM <optional editing character><comment>
<comment> is any character string which does not include a carriage return.
Possible <optional editing character>s and their meanings are
1) > replace the rightmost portion of the COMMENT with this string. The replacement is made on a character for character basis.
2) < replace the leftmost portion of the COMMENT.
3) - add this string preceding the COMMENT.
4) + add this string to the end of the COMMENT.

CORRECT
no arguments are allowed.

DELETE
DELETE <sequence set>
<sequence set> is a set of <sequence range>s separated by commas (,).
<sequence range> is <sequence number> or <start sequence number>- or <start sequence number>-<end sequence number> or -<end sequence number>

ENTER
no arguments are allowed.

EXIT
EXIT <program file name>
<program file name> is the name of a valid .EXE file to which control is to be transferred on exit from PHD.

FIT
FIT <parameter file name>
<parameter file name> is a name for an descriptor file that provides parameter information for the fit. The file extension must be .PAR

GET
GET <dataset name>
<dataset name> is the name of an existing dataset on the mass storage device (the file name is <dataset name>.DAT). The format of the dataset must be compatible with PHD.

SUMMARY SHEET FOR PHD COMMANDS (Cont'd)

LIST
no arguments are allowed.

MFP
no arguments are allowed.

PLOT
no arguments are allowed.

PRINT
PRINT <number of copies>
<number of copies> is an integer between 1 and 10.

RENAME
RENAME <new workset name>
<new workset name> is a filename acceptable to VMS.

SAVE
SAVE <dataset name>
<dataset name> is a valid VMS file name.

SORT
no arguments are allowed.

SWAP
no arguments are allowed.

USE
USE <user file name>
<user file name> must be the name of an .EXE type file built using the USER.BAS module as an envelope. The user written program lines which may modify the WORKSET must be sequenced between 1000 and 29999.


FILME
5-8