ORNL/CSD/TM-45

OPRODF, A Decsystem-10 Data Manipulation Program for ORELA Data Formatted Files

J. G. Craven

OAK RIDGE NATIONAL LABORATORY
OPERATED BY UNION CARBIDE CORPORATION · FOR THE DEPARTMENT OF ENERGY

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161
Price: Printed Copy $6.50; Microfiche $3.00

ORNL/CSD/TM-45

Contract No. W-7405 eng 26

COMPUTER SCIENCES DIVISION

OPRODF, A DECSYSTEM-10 DATA MANIPULATION PROGRAM FOR ORELA DATA FORMATTED FILES

J. G. Craven

Computing Applications Department

Date Published - May, 1978

UNION CARBIDE CORPORATION, NUCLEAR DIVISION
Operating the
Oak Ridge Gaseous Diffusion Plant . Oak Ridge National Laboratory
Oak Ridge Y-12 Plant . Paducah Gaseous Diffusion Plant
for the
DEPARTMENT OF ENERGY

CONTENTS

CONTENTS (Cont.)

ACKNOWLEDGMENTS

The author acknowledges with gratitude the assistance of D. K. Olsen in providing the user's point of view for much of the user's interface into the program and in providing many of the examples contained in this document displaying the use of the program. J. W. T. Dabbs, R. Gwin, and L. W. Weston contributed much in their discussions into the problems encountered in getting started with the program, and their suggestions for making the program more useful have been invaluable. Recognition for the design of the ORELA data format goes to N. A. Betz and C. L. Thompson. Thanks also go to Marie Cuthbert for her help in the preparation of the manuscript.

OPRODF, A DECSYSTEM-10 DATA MANIPULATION PROGRAM FOR ORELA DATA FORMATTED FILES

J. G. Craven

ABSTRACT

OPRODF allows the user to perform mathematical operations on his/her disk ORELA data formatted files by simply typing mathematical equations on his/her terminal. Expressions contained in the equations define the file, the starting position in the file, the ending position in the file, and the values such as content, time, energy, etc., associated with position in the file. If several equations are used, the order of execution of the equations can be controlled by FORTRAN control statements. Incorporated into OPRODF are statements for creating ODF files and listing selected data on the line printer, displaying selected data on the TEKTRONIX or PDP-15 scopes, plotting selected data on the CALCOMP, and dumping selected data onto magnetic tape.

1. INTRODUCTION

OPRODF began as a program written for D. K. Olsen of the Neutron Physics Division who wanted to operate on his ODF datasets simply by giving a mathematical equation. Since then OPRODF has developed into a system of programs allowing the user to easily operate on, list, display, plot, and dump his/her datasets.

Adding datasets, background correcting, exponential tail correcting, calculating transmission, transmission error, cross section, and cross section errors are examples of operations that can be done by a single OPRODF equation. Dead-time correcting, integral summing, and averaging are examples of operations that can be done by an OPRODF program which consists of several equations whose order of execution are determined by control statements. OPRODF operates on PDP-10 disk files written with the ORELA Data Format, ODF, and converts files to and from ODF files as needed with the MAKE and LIST statements.

Except for the monitor disk I/O routines, OPRODF is written in FORTRAN. The compiler program requires about 22K of core. The execution of the user's equation after compiling requires 3K. The execution of the user's program after compiling requires from 3 to 5K.

2. RUNNING OPRODF

The user initiates OPRODF at his/her terminal by keying:

.R OPRODF

2.1 Input parameters

(Prompt)

(Reply)

(1) FILENAME 1=

Type the filename (dsk:file.ext[p,pn]/switch) of the file to be associated with the symbol "F1" in the arithmetic expressions. If the file exists and no part is to be written over, no /switch is required. If the file exists and part of it is to be written over, the switch "WRITE" is required. If the file is to be created, the switch "NEW" IS REQUIRED.

(1) FILENAME 2=

Type the filename (dsk:file.ext[p,pn]/switch) of the file to be associated with the symbol "F2" in the arithmetic expressions.

(1) FILENAME x=

Type the filename (dsk:file.ext[p,pn]/switch) of the file to be associated with the symbol "Fx" (x will be between 1 and 15) in

the arithmetic expressions. If all desired filenames have been typed, type a <CR>. A maximum of 15 files can be used.

TYPE EQUATION

Type either an equation, an equation-like statement (LIST, TYPE, DUMP, MAKE, RESTORE, PLOT, VT or TEK), "PROGRAM" to start a program, a plot switch, or "STOP".

2.2 "/NEW" switch questions

When the "/NEW" switch is used, the file parameter questions below are asked. The answers to the questions are used to create the new file and cannot be changed at a later time. If the user does not know what the question is about, he/she may type a carriage return for the default file parameter to be used.

(Question)

(Reply)

FILENAME OF FILE CONTAINING DEFAULT FILE PARAMETERS=

Type the filename of another ODF file whose file parameters will be used for the remaining file parameter questions if they are replied to with a <CR>. If the answers to all of the questions are to be copied, include the "/COPY" switch after the default filename and the remaining questions will not be asked. If there is no default file, type a <CR>.

ODF MODE=

Type the ODF data mode (half-word integers=0, full-word integers=1, floating point=3).

YOU HAVE x DATASET TABLES TYPE THE DATASET TABLE NUMBER.

This question is asked only if the default file is a mode 0 ODF file containing more than one dataset table and the file being created is not a mode 0 file. Type the dataset table number that is to be written into the parameter section of the file being created.

SOURCE=

Type the dataset source number (SEL=0, CSISRS =1, ENDFB =2).

RUN ID=

Type the numerical run ID to be associated with the file.

NUMBER COMMENT BLOCKS=

Type the number of comment blocks to be allocated in the file. One block holds 680 characters.

NUMBER VARIABLES=

Type the number of user variables to be allocated in the file.

NUMBER OF DATASETS=

Type the number of datasets to be allocated in the file. If the default answer file is a mode 0 file and the file being created is not, the default number of datasets will be equal to the number of tagged storage sections allocated by the first dataset's crunch table(s) in the parameter section.

IS DATASET 1 ENERGY?(Y<CR> OR N<CR>)

This question will not be asked for a mode 0 file. Type a Y<CR> if the first dataset is to contain energy or an N<CR> if the first dataset is not to contain energy.

NUMBER OF CHANNELS PER DATASET=

Type the number of channels that are to be stored in each dataset. If the datasets are of different length, type the largest number of channels in a dataset. This number can be different than the number indicated by the dataset table in the parameter section. When the default answer file is a mode zero file and the file being created is not, the default number of channels is the number of channels allocated by the first dataset's crunch table(s) in the parameter section.

(1) ZAN=

Type the ENDFB designation.

(1) AWR=

Type the ratio of the nuclear mass of the material to that of the neutron.

(1) MAT=

Type the ENDFB mat number.

(1) MF=

Type the ENDFB file number.

(1) MT=

Type the ENDFB reaction number.

(1) Asked for ENDFB data only.

3. PROGRAM OPERATION

The program begins by asking the user to type the filenames of all the ODF files that are to be referenced in the OPRODF equations. The program prompts the user to type each filename by typing "FILENAME x=" where x will increase from one to a maximum of 15. As each filename is typed, the program confirms that the file can be read and/or written by typing the run ID, the number of ODF datasets and the number of channels in each dataset. The user indicates there are no more filenames to be given by typing a <CR> in place of another filename.

If the file is to be written into, the "/NEW" or "/WRITE" switch must be included with the filename. The "/NEW" switch is used when the file is to be created and if a file already exists, it will be deleted. The user will be asked to type the required information necessary to create the file before asking for the next filename. The "/WRITE" switch is used when the file already exists but part of it is to be written over.

After all the filenames have been given, the program asks the user to type an equation. If the user wants to use a program instead of an equation or an equation-like statement, the first line of the program must be the "PROGRAM" statement. The OPRODF equation is repeatedly executed, incrementing the starting channel number of each ODF file reference (F1S1SC22) by 1 until the ending channel of the output file is written. The OPRODF program starts execution with the "END" statement and is executed

until a "STOP" statement is encountered or the last statement of the program has been executed a number of times equal to the largest number of channels to be written into any one output file. Each time an ODF file reference is executed, the starting channel number is incremented by 1.

After the equation or program has been executed, OPRODF types "TYPE EQUATION" and waits for the next equation or program to be typed. The user can exit the program by using the "STOP" command or typing a control C.

4. THE OPRODF EQUATION

The OPRODF equation is a single line which consists of variables, constants, functions, file references, and operators. The equation starts in column 1 and takes the form:

<A>=<B><C><B><C><ETC.><ETC.>

where <A> is a variable (SUM) or file reference (F1S1SC1EC11=); <B> is a constant (1.23), variable (SUM), function (EXP(SUM)), or file reference (F1S1SC1EC11); and <C> is an arithmetic operator (+ - * / **).

5. THE OPRODF PROGRAM

The OPRODF program is a set of equations whose order of execution can be controlled by control statements (<GO TO 11>, <IF(A.LT.B)GO TO 11>, etc.). The first statement of the program is the "PROGRAM" statement and the last statement is the "END" statement. The format of the program equation is the same as the single equation except each equation starts in column 7. A tab may be used to skip over columns 1 through 6. Command files can and should be used to input programs.

5.1 Statement format

Columns 1-5; The statement label and number fields

Columns 1 through 5 can contain a one- to five-digit number to be referenced by another statement. Leading zeros and spaces in the statement number are ignored. Statement numbers may be assigned in any order and must be unique with respect to all other statement numbers. A tab may be used to skip over all or part of the statement number field.

Column 6; The line continuation field

Any alphanumeric character, except a blank or zero, placed in column 6 identifies the line as a continuation line. Whenever a tab is used to skip over the label field of a continuation line, the next character must be one of the digits 1 through 9 to identify the line as a continuation line.

Column 7-72; The statement field

Columns 7 through 72 contain the statement to be executed. Spaces and tabs are ignored.

Comment statements

Any statement that contains the letter "C" in column 1 is a comment statement and has no effect on the OPRODF program or equation.

5.2 Statements

(Statement)

(Meaning)

PROGRAM

A program is to be given instead of an equation.

DIMENSION ARRAY(LENGTH)

The array "ARRAY" is to have "LENGTH" elements.

DOUBLE PRECISION

All operations are to be done in double precision.

DATA VARIABLE/VALUE/

The initial value of the variable "VARIABLE" is to be "VALUE".

DO I J=K,L,M

Set the value of the variable "J" equal to the value of the variable "K" and execute the statements following the "DO"

statement up to and including the statement with a statement number of "I". After executing statement number "I", add the value of the variable "M" to the value of the variable "J". If the new value of the variable "J" is less than or equal to the value of the variable "L", execute the statements after the "DO" statement again. Otherwise continue executing the statements following statement number "I". The statement "I" must appear after the "DO" statement. The value of "K" and "M" must be positive nonzero. The value of "L" must be greater than the value of "K". "DO" loops appearing inside other "DO" loops must end inside those "DO" loops. "DO" loops may end on the same statement.

CONTINUE

The "CONTINUE" statement may be placed anywhere in the program without affecting the execution of the program. "CONTINUE" statements are generally used to end "DO" loops when no other statement can be used.

GO TO X

Continue execution at statement X where X is a statement number.

IF(A.LT.B) STATEMENT

If the value of the variable A is less than the value of the variable B, execute the STATEMENT.

IF(A.LE.B) STATEMENT

If the value of the variable A is less than or equal to the value of the variable B, execute the STATEMENT.

IF(A.EQ.B) STATEMENT

If the value of the variable A is equal to the value of the variable B, execute the STATEMENT.

IF(A.NE.B) STATEMENT

If the value of the variable A is not equal to the value of the variable B, execute the STATEMENT.

IF(A.GE.B) STATEMENT

If the value of the variable A is greater than or equal to the value of the variable B, execute the STATEMENT.

IF(A.GT.B) STATEMENT

If the value of the variable A is greater than the value of the variable B, execute the STATEMENT.

TYPE <A>,<A>

Type the results of each <A> where <A> is a variable for file reference. (See TYPE statement and note 1.)

LIST <A>,<A>

List the results of each <A> in a disk file. (See LIST statement and note 1.)

DUMP <A>,<A>

List the result of each <A> in a magnetic tape file. (See DUMP statement and note 1.)

MAKE <A>,<A>

Perform the inverse of the LIST statement. (See MAKE statement and note 1.)

RESTORE <A>,<A>

Perform the inverse of the DUMP statement. (See RESTORE statement and note 1.)

(1) PLOT <A>,<A>

Plot the result of each <A> in a disk file. (See PLOT statement and note 1.)

VT <A>,<A>

Plot the result of each <A> on the PDP-15 display. (See VT statement and note 1.)

TEK <A>,<A>

Plot the result of each <A> on the TEKTRONIX display terminal. (See the TEK statement and note 1.)

STOP

Stop the execution of the OPRODF program.

END

Last statement of the program.

Note 1: One, and only one, TYPE, LIST, DUMP, MAKE, RESTORE, PLOT, VT, or TEK statement may be used in a program. That is, one of these statements may appear only once in a program.

5.3 Variables

A variable is a datum (storage location) that is identified by a symbolic name. The symbolic name is made up of six characters or less from the letters A through Z and the numbers 0 through 9. The first character of a variable name must be a letter. Variables specify values which are assigned to them by either arithmetic statements (VAR=22) or data statements (DATA VAR/22/). Variables are initially set to zero unless specified by a DATA statement.

5.4 Arrays

An array is an ordered set of data identified by an array name. Array names are symbolic names and must conform to the rules for writing variable names. Each datum of an array is referenced by appending a subscript to the array name that describes the position of the datum. The subscript may be a given value (A(1)=22) or a variable (A(I)=22). The size of an array must be declared with the "DIMENSION" statement (DIMENSION A(10)). Array datum are initially set to zero unless specified by a DATA statement.

5.5 Arithmetic operators

(Operator)	(Operation)
+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Exponentiation

5.6 Arithmetic functions

(Function)	(Meaning)
SQRT	The square root.
DSQRT	The double precision square root.
ALOG	The natural log base e.
DLOG	The double precision natural log base e.
EXP	The exponential.
DEXP	The double precision exponential.
ABS	The absolute value.
DABS	The double precision absolute value.

6. ODF FILE CHANNEL VARIABLE REFERENCES

All ODF file channel variables may be referenced on the right side of the equal sign in the arithmetic expressions. Only references to the file channel value, that is channel contents, may be used on the left side of the equal sign.

The user references the files whose names he/she has given by using the "F" symbol followed by the filename number that was associated with the filename when the filename was typed. "F1" indicates the file whose name was typed first, "F2" indicates the file whose name was typed second, etc.

The user references the ODF file's datasets by using the "S" symbol followed by the ODF dataset number. "S1" indicates dataset 1, "S2" indicates dataset 2, etc. The "F" symbol must precede the "S" symbol.

The user indicates where to start referencing the file variables by using the "SC" symbol followed by the channel number of the first channel whose variable is to be used or by using the "SE" symbol followed by the energy of the first channel whose variable is to be used. The "S" symbol must precede the "SC" or "SE" symbol.

The default starting channel of an output file is the first channel of that file. The following two examples are equivalent.

F1S1SC1=F2S1

F1S1=F2S1

The default starting channel of an input file that appears in a statement that does not reference an output file is the first channel of that file. The following two examples are equivalent.

```
VAR=F1S1SC1+F2S1SC1

VAR=F1S1+F2S1
```

The default starting channel of an input file that appears in a statement that references an output file is the same starting channel given for the output file. The following two examples are equivalent.

```
F1S1SC4=F2S1SC4+F3S1SC4

F1S1SC4=F2S1+F3S1
```

The user indicates where to stop referencing the file by using the "EC" symbol followed by the channel number of the last channel whose variable is to be used or by using the "EE" symbol followed by the energy of the last channel whose variable is to be used. The "S" symbol must precede the "EC" or "EE" symbols.

The default ending channel of an output file is the last channel of that file. The following two examples are equivalent (the file has 1000 channels).

```
F1S1EC1000=F2S1

F1S1=F2S1
```

The default ending channel of an input file that appears in a statement that references an output file is the same ending channel given for the output file. The following two examples are equivalent.

F1S1SC4EC22=F2S1SC4EC22+F3S1SC4EC22

F1S1SC4EC22=F2S1+F3S1

If the user specifies an ending channel on an input file and during execution the starting channel is incremented beyond that channel, the value returned for that file reference will be zero. In the following example the last 10 channels will be zero.

F1S1EC20=F2S1EC10

If the user does not specify an ending channel on an input file and during execution the starting channel is incremented beyond the last channel of the file, execution is stopped and an error message is given.

The default channel variable is the channel value. The other channel variables are channel number "CN", channel time width "CTW", channel starting time "CST", channel center time "CCT", channel ending time "CET", channel starting energy "CSE", channel center energy "CCE", and channel ending energy "CEE". All channel time references are in microseconds. All calculated energies are in eV.

The channel starting/center/ending time symbols are used in two ways. If a dataset number does not precede the symbol (F1CCT), the time will be obtained from the dataset table. If a

dataset number precedes the symbol (F1S2CCT), the time will be the sum of that dataset's TZERO and the time obtained from the dataset table.

The channel starting/center/ending energy symbols are used in two ways. If a dataset number does not precede the symbol (F1CCE), the energy is obtained from dataset 1. The channel starting energy will be the average of the current channel center energy and the previous channel center energy. The channel center energy is the channel content from the energy dataset. The channel ending energy is the average of the next channel center energy and the current channel center energy. The starting energy of the first channel is obtained by subtracting the difference between the second-channel center energy and the first-channel center energy from the first-channel center energy. The last-channel ending energy is obtained by adding the difference of the last-channel center energy and the next-to-last channel center energy to the last-channel center energy.

If a dataset number precedes the channel starting/center/ending energy symbol (F1S2CCE), the energy is calculated using the TZERO and flight path from that dataset, the dataset table from the indicated file and the equation:

$$E(R)=E(N)*(1+1.596470E-9*E(N)+2.831906-18*E(N)**2)$$

where

$$E(N)=(72.29855*\text{FLIGHT PATH/TIME})**2$$

6.1 Constants

The ODF file constants TZERO and flight path may be referenced on either side of the equal sign in the arithmetic expressions. When used on the left side of the equal sign, the value of the constant at the completion of the execution of the equation or program will be written and stored in the file. During the execution of the equation or program, the constant is handled as if it were a program variable which was initialized with a data statement. The initial value of the constant is the value in the file before execution of the equation or program.

The user references the TZERO constant (starting time of the first channel in microseconds) with the "TZERO" symbol. A file and dataset number must precede the TZERO symbol (F1S2TZERO).

The user references the flight path constant (flight path length in meters) with the "FP" symbol. A file and dataset number must precede the FP symbol (F1S2FP).

6.2 Variables

ODF file variable references may be used on either side of the equal sign in the arithmetic expressions. When used on the left side of the equal sign, the value of the variable at the completion of the execution of the equation or program will be written and stored in the file. During the execution of the equation or program, the variable is handled as if it were a program variable which was initialized with a data statement. The initial value of the variable is the value in the file before execution of the equation or program.

The user references the file variables with the "V" symbol. The "V" symbol must be preceded with the file number. File variables are used to store variables for use in future equations and programs.

Files created by the SEL program 6SND have 16 file variables allocated. Any number of variables may be allocated when creating an ODF file with the "/NEW" switch.

6.3 Data acquisition counters and scalers

The data acquisition scalers and SEL counters are stored in the ODF counter section and can be referenced with the "C" symbol. Each data acquisition routine has a different storage format for the counters; therefore, the user must consult his/her routine for his/her format. ODF file counters may be referenced on either side of the equal sign in the arithmetic expressions. When used on the left side of the equal sign, the value of the counter at the completion of the execution of the equation or program will be written and stored in the file. During the execution of the equation or program, the counter is handled as if it were a program variable which was initialized with a data statement. The initial value of the counter is the value in the file before execution of the equation or program. The "C" symbol must be preceded with the file number (F1C1B16).

6.4 Reference symbols

The default starting channel of an ODF file variable reference (F1S1CN) is the starting channel number of the first file reference. The default ending channel number is the ending channel number of the first file reference. The following two examples are equivalent:

(Symbol)

(Meaning)

Fx

The file x where x is the number given to the filename in the "INPUT PARAMETERS" section.

Example: F1

Sx

The dataset x where x is the ODF dataset number.

Example: F1S1

SCx

The starting channel where x is the starting channel number.

Example: F1S1SC22

ECx

The ending channel where x is the ending channel number.

Example: F1S1SC22EC44

NCx

The ending channel where SC+x-1 is the ending channel number.

Example: F1S1SC22NC23

SEx

The starting channel where x is the energy of that channel.

Example: F1S1SE456.2

EEx

The ending channel where x is the energy of that channel.

Example: F1S1SE456.7EE567.8

CN

The channel number.

Example: F1S1=F2CN

TZERO

The TZERO value of the given dataset in the given file. To set the value of the TZERO, see the SET command.

Example: F1S1=(72.297*40.1/(F2S1TZERO+F2S1CCT))**2

FPD

The flight path value of the given dataset in the given file. To set the value of the flight path, see the SET command.

Example: F1S1=(72.297*F2S1FPD/(F2S1TZERO+F2S1CCT))**2

CTW

The channel time width in microseconds.

Example: F1S2=F2CTW

CST

The channel starting time in microseconds. (See note 1.)

Example: F1S1=F2CST OR F1S1=F2S1CST

CCT

The channel center time in microseconds. (See note 1.)

Example: F1S1=F2CCT OR F1S1=F2S1CCT

CET

The channel ending time in microseconds. (See note 1.)

Example: F1S1=F2CET OR F1S1=F2S1CET

CSE

The channel starting energy. (See note 2.)

Example: F1S1=F1S2CSE OR VAR=F1CSE

CCE

The channel center energy. (See note 2.)

Example: F1S1=F1S2CCE OR VAR=F1CCE

CEE

The channel ending energy. (See note 2.)

Example: F1S1=F1S2CEE OR VAR=F1CEE

Note 1: If a dataset number precedes the symbol (F1S1CST), the TZERO of that dataset will be added to the time. If a dataset number does not precede the symbol (F1CST), a TZERO is not added to the time.

Note 2: If a dataset number does not precede the symbol (F1CSE), the channel center energy will be the contents of the energy dataset. The starting/ending energy will be the average of the previous/next channel center energy and the current channel center energy. If a dataset number precedes the symbol (F1S2CCE), the energy is calculated using the TZERO and flight path from that dataset, the dataset table from the indicated file, and the equation:

$$E(R)=E(N)*(1+1.59651E-9*E(N)+4.531273E-18*E(N)**2)$$

where

$$E(N)=(72.2977*\text{FLIGHT PATH/TIME})**2$$

If averaging is done, the dataset table must be remade before using these symbols. To remake the dataset table, see the SET command.

7. ODF FILE SET COMMANDS

(Command)

(Meaning)

FxSyTZERO=VALUE

Set the TZERO value of dataset "y" in the file "x" to "VALUE" microseconds.

Example: F1S1TZERO=-.476

FxSyFPD=VALUE

Set the flight path value of dataset "y" in the file "x" to "VALUE" meters.

Example: F1S1FPD=40.

FxVy=VALUE

Set the "y"th variable in file "x" to "VALUE".

Example: F1V5=.1638E2

FxCyBz=VALUE

Set the "y"th SEL counter in file "x" to "VALUE". The counter is to be store as a "z" bit counter.

Example: F1C5B32=F2C5B32+F3C5B32

FxCOMMENTS=

Input the comments and store them into the comment section of file "x" until a blank comment line is read or the comment section becomes full.

Example: F1COMMENTS=

FxTABLE=

Remake the dataset table in file "x". The program will ask the user to type the number of channels and the crunch factor for those channels, one pair per line. When all pairs have been typed, type another <CR>.

Example:

```
F1TABLE=
TYPE NUMBER CHANNELS, CRUNCH FACTOR
(<CR> WHEN DONE)
100 1
200 4
300 16
```

8. ODF FILE AND VARIABLE TYPEOUT COMMANDS

(Command)

(Meaning)

FwSxSCyECz

Type out the channel contents of channels "y" through "z" from dataset "x" in file "w". If "ECz" is not given, the channel contents of channel "SCy" will be typed.

Example: F1S1SC1EC10 OR F1S1SC1

FWsXSEyEEz

Type out the channel contents of the channels between the energies "y" and "z" from dataset "x" in file "w". If "EEz" is not given, the channel contents of the channel that has an energy of "y" will be typed.

Example: F1S2SE4.75EE5.75 OR F1S2SE4.75

VARIABLE

Type out the value of the OPRODF program variable "VARIABLE" from the last program executed.

Example: VAR

FxSyTZERO

Type out the TZERO value of dataset "y" in file "x".

Example: F1S1TZERO

FxSyFPD

Type out the flight path value of dataset "y" in file "x".

Example: F1S1FPD

FxTABLE

Type out the dataset table in file "x".

Example: F1TABLE

FxCOMMENTS

Typeout the comments in file "x".

Example: F1COMMENTS

9. THE LIST COMMAND

The LIST statement is used to create a readable file containing information from ODF files. The format of the LIST statement is:

```
LIST <A>,<A>,<ETC>
```

where LIST is in columns 1-4 and <A> is an ODF file variable (F1S1CN), arithmetic expression (F1S1+200), variable (SUM), array element (SUM(1) or SUM(I)), or implied DO loop (SUM(I),I=J,K,M). When used in a program, the LIST statement starts in column 7. All LIST statements in a program will use the same line format.

Each line will have one number for each <A>. The total number of lines listed, not including the page header, is equal to the largest difference between the ending and starting channel numbers of any ODF file variable reference plus 1. The example:

```
LIST F1S1SC10EC20,F2S1SC10EC30
```

will create 21 lines with two numbers per line.

```
LIST F1S1SC10EC20,F2S1SC10EC20
LIST F1S1SC10EC20,F2S1
```

9.1 LIST statement questions

When the LIST statement is used, the following questions are asked after input of the equation or program and prior to the execution of the equation or program.

(Question)

(Answer)

FILENAME OF FILE TO STORE LISTING INTO=

Type the filename (dev:file.ext[p,pn]) of the file for the listing to be stored into.

TYPE THE PAGE HEADER (END WITH /<CR>)

If no page header is desired, type a <CR>. If a page header is desired, type the header followed by /<CR>. An example would be:

```
ENERGY   TRANSMISSION   ERROR
/
```

The user should be careful with the use of the ', ", and characters in the header as they are specific command file characters which cause special action to be taken. The header may contain up to 630 characters making up as many lines as needed.

NUMBER OF LINES PER PAGE=

Type the number of lines per page of output. This does not include the lines of the page header, but only the lines generated by the line format statement. The default number of lines per page is 999999.

LINE FORMAT=

Type the format statement each line is to be listed with. Only the I, E, F, and X format descriptors may be used. An example would be:

I8,2X,F10.3,2(2X,E14.7)

When the I format descriptor is used, the number will be fixed before being listed (all numbers used in OPRODF are in floating point). Each format descriptor is separated by commas. If several descriptors are to be repeated, they may be enclosed with parentheses with the repeat count placed in front of the starting parenthesis. The following two examples are equivalent:

X,I5,X,I5

2(X,I5)

Format descriptors

(Descriptor)

(Meaning)

rIw

The next "r" numbers are to be read/written as integers (12) occupying "w" columns.

rFw.d

The next "r" numbers are to be read/written as floating point numbers (1.2) occupying "w" columns with "d" numbers after the decimal point.

rEw.d

The next "r" numbers are to be read/written as exponential numbers (.12E01) occupying "w" columns with "d" numbers after the decimal point. "w" must be greater than "d" plus 5.

rX

"r" spaces are to be skipped/written.

10. THE TYPE STATEMENT

The TYPE statement is identical to the LIST statement except no filename is required. The listing is typed on the user's terminal instead of being stored in a disk file.

11. THE DUMP STATEMENT

To use the DUMP statement, the user must mount a magnetic tape with the PDP-10 system MOUNT command before running OPRODF. An example would be:

```
.MOUNT MTA:/REELID:X1234/WE
```

where "X1234" is replaced with the user's tape label.

The DUMP statement is used to create a magnetic tape file that can be read by either the PDP-10 system program "PIP" or by any IBM computer. The format of the "DUMP" statement is:

```
DUMP/FILEx <A>,<A>,<ETC.>
```

where "DUMP" is in columns 1-4 and "<A>" is the same as in the "LIST" statement. When used in a program, the "DUMP" statement starts in column 7. All "DUMP" statements in a program will use the same line format. "FILEx" is used to rewind the tape and position it in front of file "x" (1 for the first file, 2 for the second file, etc.) before writing the new file. "/NEW" may be used in place of "FILE1" if desired. If the "/FILEx" switch is not used (DUMP F1S1,F1S2,F1S3), the tape is positioned at the end of the last file written on the tape before writing the new file. Each "DUMP" statement writes one tape file. Note that if a tape has six files and the user gives the "DUMP" statement:

```
DUMP/FILE4 F1S1,F1S2
```

files 5 and 6 will no longer exist after the new file 4 is written.

The questions asked by the "DUMP" statement are the same as the questions asked by the "LIST" statement except the first "LIST" question is replaced with the question:

PDP10 OR IBM (I<CR> OR P<CR>)

If the user replies with P<CR>, the tape will be written in a DIGITAL-COMPATIBLE format readable by the PDP-10 system program PIP. If the user replies with I<CR>, the tape will be written in an INDUSTRY-COMPATIBLE format readable by IBM computers (EBCDIC, 800BPI, ODD PARITY). Each page of output is written into one magnetic tape record. The blocksize is equal to the number of characters in a page rounded up to the nearest multiple of 4. A page size of 399 characters will have a blocksize of 400. The record size will be the number of characters created by the line format. The format I8,2F7.1 creates a record size of 22.

12. THE MAKE STATEMENT

The MAKE statement is used to make an ODF file from a readable file. The "MAKE" statement is the inverse of the "LIST" statement and follows the same format, rules, and procedures.

12.1 MAKE statement questions

When the MAKE statement is used, the following questions are asked after input and prior to execution of the equation or program.

(Question)

(Answer)

FILENAME OF FILE TO READ LISTING FROM=

Type the filename (dev:file.ext[p,pn]) for the listing to be read from.

NUMBER OF PAGE HEADER LINES TO BE SKIPPED=

Type the number of page header lines that appear before the lines containing the data to be read.

NUMBER OF LINES PER PAGE=

Type the number of data lines between the page headers.

LINE FORMAT=

Type the format statement each line is to be read with. The rules for the format statement are the same as for the "LIST" statement.

13. THE RESTORE STATEMENT

To use the "RESTORE" statement, the user must mount, before running OPRODF, the magnetic tape with the PDP-10 system MOUNT command in the same manner as for the DUMP statement. The "RESTORE" statement is the same as the "MAKE" statement except no filename is required. The listing is read from magnetic tape instead of a disk file. The "RESTORE" statement uses the same tape positioning switches as the DUMP statement. Each "RESTORE" equation or program leaves the tape position at the start of the next tape file. The questions asked by the "RESTORE" statement are the same as those asked for by the "MAKE" statement, with the exception that the first "MAKE" question is replaced with the question:

PDP10 OR IBM (I<CR> OR P<CR>)

The reply should be the same as that for the DUMP statement.

14. PLOT, VT, AND TEK STATEMENTS

The PLOT, VT, and TEK statements are used to create plots in disk files or on PDP-15 VT04 and TEKTRONIX display terminals. Scaling and labeling are automatic unless specified; the user must give the file and range to be plotted. The program produces a 5x7-inch linear plot with labeling on the left and bottom axes. The format of the statements is:

PLOT/SWITCHvalue/SWITCHvalue <A>,/SWITCHvalue <A>

where "PLOT" is in columns 1-4 and <A> is a file reference (F1S2). Each <A> is completely plotted before going on to the next <A>. The example:

PLOT F1S2SE2.4EE4.7,F2S2

will plot all points between the energies 2.4 and 4.7 from file 1, then all points between the energies 2.4 and 4.7 from file 2.

Each plot statement or program with plot statements produces a single plot in the disk file FOR29.DAT. Before creating another plot, the user should rename (REN FILE.EXT=FOR29.DAT) the file to some other name or queue (QSPLT/D OR QPLT/D) the file to some CALCOMP with the /D switch. Otherwise, the next plot will replace the previous plot. An example would be:

```
PLOT F1S2SE2.4EE4.7
QSPLT/D
```

```
PLOT F2S2SE2.4EE4.7
QSPLT/D
```

The VT and TEK statements operate in the same manner as the PLOT statement except the user must type a carriage return for the program to continue after each plot. The screen is cleared when the carriage return is typed.

Plot switches are used to alter the type of plot (log, linear, symbol, etc.), alter the labeling, or add titles. Switches used to create a plot remain in effect for future plots until changed by the user. Switches are always given for the next file reference. In the example:

```
PLOT/SYM1 F1S1,F2S1,/SYM2 F3S1
```

the request for symbol 1 is used for F1S1 and F2S1. The request for symbol 2 is used for F3S1 and the next plot file reference. As each plot is plotted, the switches used for that plot are written into the file OPRPLT.TMP. When OPRODF is initially run, the previous plotting switches are read from the file OPRPLT.TMP. A "RESET" switch restores the plot switches to their default values.

Any plot statement containing only switches does nothing but set those switches. Any statement with a slash in column 1 is assumed to be a plot statement containing only switches. The following two statements do the same thing:

```
PLOT/SYM0/ERR3
/SYM0/ERR3
```

If file 1 has energy in dataset 1, data in dataset 2, errors in dataset 3, and fitted curve in dataset 4, and the user wants a symbol plot of the data with error bars and fitted curve, he/she would type:

```
PLOT/SYM0/ERR3 F1S2SE3.4EE7.2,/NOSY/NOER F1S4
```

or

```
/SYM0/ERR3
PLOT F1S2SE3.4EE7.2,/NOER/NOSY F1S4
```

or

```
PROGRAM
/SYM0/ERR3
PLOT F1S2SE3.4EE7.2
/NOSYM/NOER
PLOT F1S4
END
```

To replot the first example with titles, type:

```
/LTIT $|BARNS$,/BTIT $|ENERGY |E|V$
PLOT/SYM0/ERR3 F1S2SE3.4EE7.2,/NOSY/NOER F1S4
```

All future plots will retain the above axis titles until changed by the user.

When the user plots on the VT04, the plot is scaled down if needed to fit the plotting area.

14.1 Plotting switches

The "?" in the switches below must be replaced with "R" when referring to the right axis, "T" when referring to the top axis, "L" when referring to the left axis, "B" when referring to the bottom axis, "X" when referring to both the top and bottom axis, and "Y" when referring to both the left and right axis. Only enough characters to distinguish the switch need to be typed (/RESET /RE).

(Switch)

(Meaning)

RESET

Reset all switches, switch values, and switch contents to their default values.

WHAT

Type all switches, switch values, and switch contents currently in effect.

XOFFzzz

The minimum x-coordinate of the plotting area is to be zzz inches. The default is 1 inch.

YOFFzzz

The minimum y-coordinate of the plotting area is to be zzz inches. The default is 1 inch.

XALzzz

The x-axis length is to be zzz inches. The default is 7 inches.

YALzzz

The y-axis length is to be zzz inches. The default is 5 inches.

XMINzzz

The value of the minimum x-coordinate of the plotting area is to be zzz. The default will be the smallest starting channel/energy used in the file reference.

XMAXzzz

The value of the maximum x-coordinate of the plotting area is to be zzz. The default will be the largest channel/energy used in the file reference.

YMINzzz

The value of the minimum y-coordinate of the plotting area is to be zzz.

YMAXzzz

The value of the maximum y-coordinate of the plotting area is to be zzz.

The default YMAX will be the power of 10 above the largest value to be plotted, that power being scaled down until the range of the plotted data is greater than half the range between YMAX and YMIN. The default YMIN will be the power of 10 below the smallest value to be plotted, that power being scaled up. Default YMIN and YMAX values will not be used unless both YMAX and YMIN are zero.

XSNzzz

Values for the x-axis are to be read from dataset zzz. This switch is used to obtain an energy plot when specifying starting and ending channels.

XBASEzzz

The channel offset is to be zzz. A zzz of 4096 is used to plot channels 4097 through 8192 as channels 1 through 4096. (PLOT/XBASE4096 F1S1SC4097EC8192)

NOXBASE

Resets the XBASE switch if set.

XNORMzzz

The x-axis normalization factor is to be zzz. Each x-axis label is to be multiplied by zzz before being plotted. If the energy dataset is in eV and the XNORM is .001, then XMIN and XMAX must be given in KeV.

NOXNORM

Resets the XNORM switch if set.

XENDzzz

The plot is to be ended zzz inches beyond XMAX. The default is 6 inches.

XLOG

The plot is to be log base 10 on the x-axis.

NOXLOG

Resets the XLOG switch if set.

YLOG

The plot is to be log base 10 on the y-axis.

NOYLOG

Resets the YLOG switch if set.

LINE

A line is to be drawn between each point plotted. If SYMBOL, ERROR, or HISTOGRAM is not requested, LINE is assumed.

NOLINE

Resets the LINE switch if set.

HISTOGRAM

The plot is to be a histogram plot.

NOHISTOGRAM

Resets the HISTOGRAM switch if set.

SYMBOLzzz

Symbol number zzz is to be plotted at each point.

NOSYMBOL

Resets the SYMBOL switch if set.

SHEIGHTzzz

The symbol height used in the SYMBOL switch is to be zzz.

AVERAGEzzz

Each point plotted will be the average of zzz points from the input file.

NOAVERAGE

Resets the AVERAGE switch if set.

SUMzzz

Each point plotted will be the sum of zzz points from the input file.

NOSUM

Resets the SUM switch if set.

SKIPzzz

Every zzz'th point from the input file is to be plotted.

NOSKIP

Resets the SKIP switch if set.

YERRORzzz

Vertical error bars are to be plotted at each point. The errors are to be read from dataset zzz of the file read for the points.

NOYERROR

Resets the YERROR switch if set.

XERRORzzz

Horizontal error bars are to be plotted at each point. The errors are to be read from dataset zzz of the file read for the points.

NOXERROR

Resets the ERROR switch if set.

?NTzzz

The ?-axis is to have zzz tic marks, that is, divided into zzz+1 intervals. The default is a tic every .75 to 1.5 inches, whatever yields the best label.

?Tzzz

The ?-axis tic mark length is to be zzz inches. The default is .105. A negative length will produce tic marks on both sides of the axis. A length greater than the axis will produce tic marks the length of the axis.

?NSTzzz

The ?-axis is to have zzz sub-tic marks between each tic mark, that is, divided into zzz+1 intervals. The default for linear is zero; the default for log is 8.

?STzzz

The ?-axis sub-tic mark length is to be zzz inches. The default is half that of the tic. A negative length will produce tic marks on both sides of the axis.

?LSTzzz

Label the ?-axis log sub tic mark zzz. Sub tic marks will be put

at 2 through 9, tic marks will be put at 1 and 10. The default is LLST2, LLST5, BLST2, and BLST5.

?FTLzzz

The first ?-axis tic mark label is to be zzz.

?ILzzz

The ?-axis tic label difference is to be zzz. Default is a value such that tic marks will appear between every .75 to 1.5 inches, whatever yields the best label.

?LTLzzz

The last ?-axis tic mark label is to be zzz.

?LR

The ?-axis is to be labeled. The default is LLR and BLR.

NO?LR

Resets the ?-axis label requested switch if set.

?CLR

The ?-axis corners are to be labeled.

NO?CLR

Resets the ?-axis corner label requested switch if set.

?CLDXzzz

The ?-axis corner axis delta-x is to be zzz inches.

?CLDYzzz

The ?-axis corner axis delta-y is to be zzz inches.

The delta-x and delta-y values for the corner labeling are the values to be added to the X and Y coordinates of the axis's corners to obtain the X and Y coordinates of the lower left corner of the label.

?CLH

The ?-axis corner label character height is to be zzz inches. The default is .105.

?CLAzzz

The ?-axis corner label angle is to be zzz.

?CLF<sp>zzz

The ?-axis corner label format is to be zzz. Only the I, F, and E formats may be used.

?LDXzzz

The ?-axis label delta-x is to be zzz inches.

?LDYzzz

The ?-axis label delta-y is to be zzz inches.

The delta-x and delta-y values for the tic labeling are the values to be added to the X and Y coordinates of the tic base to obtain the X and Y coordinates of the lower left corner of the tic label. Default is such that the distance between the axis and the nearest edge of the rotated label is half the character height.

?LHzzz

The ?-axis label character height is to be zzz inches. The default is .105.

?LAzzz

The ?-axis label angle is to be zzz.

?LF<sp>zzz

The ?-axis label format is to be zzz. The default format will be Ixxx if the tic label can be expressed as an integer with less than eight digits. xxx will be the number of digits in the label. If the tic label cannot be expressed as an integer but can be expressed as a floating point number with less than eight digits, then the default format will be the F format. If the label is too large for the F format, the E format will be used. When plotting a log plot, the labels will always be the number 10 raised to some power.

?LSzzz

The ?-axis label scale factor is to be zzz. The tic mark label value is multiplied by the label scale factor before the tic mark label is plotted.

?LCFzzz

The ?-axis label centering factor is to be zzz. The label centering factor value ranges from 0 to 1 and is used to center the label around the tic base. A value of .25 will put the tic at the end of the first quarter of the label. A value of .5 will put the tic in the center of the label, etc. The default is BLCF.25.

NO?LSTzzz

Resets the ?LSTzzz switch if set.

?TDXzzz

The ?-axis title delta-x is to be zzz inches.

?TDYzzz

The ?-axis title delta-y is to be zzz inches.

The delta-x and delta-y values for the axis titles are the values to be added to some origin to obtain the coordinates of the lower left corner of the title. The origin for the bottom and left axis title addition is (XMIN,YMIN). The origin for the right axis title addition is (XMAX,YMIN). The origin for the top

axis title addition is (XMIN,YMAX). The default delta-x and delta-y will be such that the rotated titles will not be on the label or grid.

?THzzz

The ?-axis title character height is to be zzz inches. The default is .105 inches.

?TAzzz

The ?-axis title angle is to be zzz degrees. The default is XTA0 and YTA90.

?TCFzzz

The ?-axis title centering factor is to be zzz. The title centering factor is used to center the title around a given point on the axis. The centering point is obtained by adding the product of the centering factor and axis length to XMIN for the x-axis titles and to YMIN for the y-axis titles. If the factor is zero, the title will be put at the x-y coordinates determined by the TDX and TDY switches.

?TITLE<sp>$text$

The ?-axis title is to be the characters between the "$" characters. Any character may be used in place of the "$" character. The characters of the axis and main titles are lower case until an up-arrow is encountered in the title text. The up-arrow is used to shift the case to the other level (lower case

to upper case, upper to lower) and is not included in the title when plotted. The title may be broken into several lines by ending each line with two consecutive up-arrows (line1||line2). To use centered symbols in the title, the symbol's number must be preceded with a back-arrow and terminated with a space (_19<sp>). The title can contain a maximum of 200 characters.

SLC<SP>z

The character "z" is to be used in place of the character "|" to indicate character case shift lock when keying in titles.

MTDXzzz

The main title x-coordinate is to be zzz inches.

MTDYzzz

The main title y-coordinate is to be zzz inches.

MTHzzz

The main title character height is to be zzz inches. The default is .105.

MTAzzz

The main title angle is to be zzz.

MTITLE<sp>$text$

Same as for ?TITLE.

15. QUEUE AND FILE COMMANDS

(Command)

(Meaning)

(1) QSLPT<SP>DEV:FILE.EXT[P,PN]/SWITCH

Enter the file "dev:file.ext[p,pn]" in the SEL's line printer queue of PDP-10 files to be printed.

(1) QSPLT<SP>DEV:FILE.EXT[P,PN]/SWITCH

Enter the file "dev:file.ext[p,pn]" in the SEL's CALCOMP queue of PDP-10 files to be plotted.

(1) QLPT<SP>DEV:FILE.EXT[P,PN]/SWITCH

Enter the file "dev:file.ext[p,pn]" in the PDP-10's line printer queue of files to be printed.

(1) QPLT<SP>DEV:FILE.EXT[P,PN]/SWITCH

Enter the file "dev:file.ext[p,pn]" in the PDP-10's CALCOMP queue of files to be plotted.

(1) RENAME DEV:FILE.EXT[P,PN]=DEV:FILE.EXT[P,PN]

Change the filename of the file on the right side of the equal sign to the filename on the left side.

(1) DELETE DEV:FILE.EXT[P,PN]

Delete the indicated file.

(1) DIRECT DEV:FILE.EXT[P,PN]

Type the creation date and time, access date, and block size.

(1) DSK is default for DEV. FOR29.DAT is default filename and extension for plot queues. The user's P,PN are default for P,PN. The /D switch will remove the file from the user's area and then plot or print it.

16. OPRODF COMMAND FILES

Command files are used to input many lines of text from a disk file instead of typing the text on the terminal. The user commands the program to read a command file by typing an "at" (@) sign followed by the file name. An example would be:

@FILE

Each line of the command file is used to build an input line to the program. Command files cannot be used in the execution of a program, only in the setup of the program.

16.1 Typeout from a command file

All characters between single quotes will be typed on the user's terminal and will not be included in the input line to the program. If the end of the command file line is reached before the matching single quote is found, a carriage return line feed is typed and the end of line is treated as if it were the matching single quote. If all of the input file line is typed, no input line is generated. An example would be:

'TYPE THE STARTING ENERGY'

or

'STARTING ENERGY='

16.2 Input to the command file

If a double quote is found in the command file line, it will be replaced by the line of text typed by the user on his/her terminal. An example would be:

'STARTING ENERGY='<TAB>SE="

or

'TYPE THE STARTING ENERGY

<TAB>SE="

If the user types "12.3<CR>", the input line will be:

<TAB>SE=12.3

If two consecutive double quotes are found in the command file line, they will be replaced by the text typed by the user on his/her terminal until a space or carriage return is typed. The terminating character will not be part of the input line. An example would be:

SUM=F1S""+F1S""+F1S""

If the user types "1<sp>2<sp>3<CR>", the input line will be:

SUM=F1S1+F1S2+F1S3

If the user types a carriage return instead of a space while typing in text being inserted into the input line due to two double quotes in the command file line, the remainder of the command file line will be ignored. An example would be:

SUM=F1S""+F1S""+F1S""+F1S""

If the user types "1<sp>2<CR>", the input line will be:

SUM=F1S1+F1S2

16.3 Command file variables

When constants must appear in several command file lines, they may be read in from the user's terminal at the beginning of the command file and then inserted by the program at the appropriate places. An example would be:

```
'STARTING ENERGY="SE'
```

The text "STARTING ENERGY=" will be typed on the user's terminal and a line of text read from the user's terminal. The line of text is assigned to the command file variable "SE". No input lines are generated by command file lines that assign text to command file variables.

Command file variables are enclosed with pound signs (#) and are replaced in the input line with the text assigned to them. If in the previous example the user had typed "247.3<CR>" and a command file line contained:

```
<TAB>SE=#SE#+.5
```

the input line to the program would be:

```
<TAB>SE=247.3+.5
```

All text assigned to command file variables are deassigned when the program is executed.

16.4 Looping command files

Whenever a null command file variable (##) is encountered, the program will start reading from the front of the command file.

17. GENERAL NOTES

All operations are performed in floating point. Decimal points do not have to be included in numbers used in the OPRODF program. Operations between parentheses are performed first, multiplication and divisions second (from left to right), and addition and subtraction last.

When using an OPRODF program, one cannot reference the file being written into on the right side of the equal sign in an equation. Data arrays or simple variables must be used to hold data for later use in the program.

18. EXAMPLES

In the examples below the underlined text is typed by OPRODF.

Example 1

Set channels 1 through 54 of dataset 1 in the file DATA1 to zero, and add 65536 to the contents of channel 146.

```
.R OPRODF

FILENAME 1=DATA1/WRITE
RUN 1000 1 DATASET 543400 CHANNELS/DATASET MODE 0
FILENAME 2=
TYPE EQUATION
F1S1EC54=0
TYPE EQUATION
F1S1SC146EC146=F1S1+65536
TYPE EQUATION
STOP
```

Example 2

Create the file DATA3 containing the sum of the files DATA1 and DATA2.

```
.R OPRODF

FILENAME 1=DATA3/NEW
FILENAME OF FILE CONTAINING DEFAULT PARAMETERS=DATA1/COPY
FILENAME 2=DATA1
RUN 1000 1 DATASET 543400 CHANNELS/DATASET MODE 0
FILENAME 3=DATA2
RUN 1000 1 DATASET 543400 CHANNELS/DATASET MODE 0
FILENAME 3=
TYPE EQUATION
F1S1=F2S1+F3S1
TYPE EQUATION
STOP
```

Example 3

Create the dead-time corrected file DATA4 and the dead-time factor file DATA44 from the raw data in the file DATA3.

```
.R OPRODF

FILENAME 1=DATA4/NEW
FILENAME OF FILE CONTAINING DEFAULT PARAMETERS=DATA3
ODF MODE=3
SOURCE=
RUN ID=
NUMBER COMMENT BLOCKS=
NUMBER VARIABLES=
NUMBER DATASETS=
IS DATASET 1 ENERGY(Y<CR> OR N<CR>)
NUMBER CHANNELS PER DATASET=
FILENAME 2=DATA44/NEW
FILENAME OF FILE CONTAINING DEFAULT PARAMETERS=DATA4/COPY
FILENAME 3=DATA3
RUN 1000 1 DATASETS 543400 CHANNELS/DATASET MODE 0
FILENAME 4=
TYPE EQUATION
      PROGRAM
C SET THE INITIAL CHANNEL WIDTH TO 1
C THE INITIAL VALUE OF LCHAN AND TCHAN IS ZERO
      DATA/CW/1/
C ADD THE LAST LEADING CHANNEL CONTENT TO THE SUM AGAIN
      SUM=SUM+LCHAN
C GET THE NEXT LEADING CHANNEL CONTENT
      LCHAN=F3S1
C ADD NEXT LEADING CHANNEL CONTENT TO SUM
      SUM=SUM+LCHAN
C SET LT TO THE CENTER TIME CURRENT CHANNEL
      LT=F3CCT
C IF CURRENT TIME SPAN IS LESS THAN DEAD-TIME GO TO 2
1     IF(LT-TT.GT.4.168)GO TO 2
C SET CW TO THE CURRENT TRAILING CHANNEL WIDTH
      CW=F3CTW
C SET TT TO THE CENTER TIME CURRENT TRAILING CHANNEL
      TT=F3CCT
C TAKE LAST TRAILING CHANNEL CONTENT FROM SUM
      SUM=SUM-TCHAN
C GET NEXT TRAILING CHANNEL CONTENT
      TCHAN=F3S1
C SUBTRACT NEXT TRAILING CHANNEL CONTENT FROM SUM
      SUM=SUM-TCHAN
C GO SEE IF WINDOW IS LESS THAN DEAD TIME
      GO TO 1
C DEAD TIME FACTOR=1/(1-(SUM/TRIGGERS))
2     V=1./(1.-(SUM/2+(4.168-LT+TT)/CW*CHAN)/108477000)
```

```
C STORE THE CORRECTED CHANNEL CONTENT
      F1S1=V*F3S1
C STORE THE DEAD TIME FACTOR
      F2S1=V
      END
TYPE EQUATION
C THE ABOVE IS REPEATED FOR EACH DATASET
TYPE EQUATION
STOP
```

Example 4

Create the background corrected file DATA5 from the dead-time corrected DATA in the file DATA4.

```
.R OPRODF

FILENAME 1=DATA5.DAT/NEW
FILENAME OF FILE CONTAINING DEFAULT PARAMETERS=DATA4/COPY
FILENAME 2=DATA4
RUN 1000 10 DATASET 54340 CHANNELS/DATASET MODE 3
FILENAME 3=
TYPE EQUATION
      PROGRAM
      V1=EXP(-.0377*F2CCT)
      V2=EXP(-.00866*F2CCT)
      F1S1=F2S1-F2CTW*(1771+131148*V1+13881*V2)
      F1S2=F2S2-F2CTW*(515.4+31605*V1+3394*V2)
      F1S3=F2S3-F2CTW*(1860+122246*V1+14016*V2)
      F1S4=F2S4-F2CTW*(547.7+29571*V1+3439*V2)
      F1S5=F2S5-F2CTW*(2055+75261*V1+12720*V2)
      F1S6=F2S6-F2CTW*(734.7+21371*V1+3664*V2)
      F1S7=F2S7-F2CTW*(2378+11731*V1+6368*V2)
      F1S8=F2S8-F2CTW*(751.1+2711*V1+1493*V2)
      F1S9=F2S9-F2CTW*(1732+134796*V1+13822*V2)
      F1S10=F2S10-F2CTW*(499.2+32288*V1+3359*V2)
      END
TYPE EQUATION
STOP
```

Example 5

Subtract an exponential tail from the background corrected data in the file DATA5.

```
.R OPRODF

FILENAME 1=DATA5/WRITE
RUN 1000 10 DATASETS 54340 CHANNELS/DATASET
FILENAME 2=
TYPE EQUATION
F1S2SC4EC590=F1S2SC4-.087*F1S2SC3-.021*F1S2SC2-.006*F1S2SC1
TYPE EQUATION
F1S4SC4EC590=F1S4SC4-.087*F1S4SC3-.021*F1S4SC2-.006*F1S4SC1
TYPE EQUATION
F1S6SC4EC590=F1S6SC4-.087*F1S6SC3-.021*F1S6SC2-.006*F1S6SC1
TYPE EQUATION
F1S8SC4EC590=F1S8SC4-.087*F1S8SC3-.021*F1S8SC2-.006*F1S8SC1
TYPE EQUATION
STOP
```

Example 6

Create the file DATA6 containing energy, transmission, and transmission error for four of the samples using the raw data in the file DATA3, the dead-time corrected data in the file DATA4, and the background corrected data in the file DATA5.

```
.R OPRODF

FILENAME 1=DATA6/NEW
FILENAME OF FILE CONTAINING DEFAULT PARAMETERS=DATA5
ODF MODE=3
SOURCE=
RUN ID=
NUMBER COMMENT BLOCKS=
NUMBER VARIABLES=
NUMBER DATASETS=9
IS DATASET 1 ENERGY(Y<CR> OR N<CR>)
NUMBER CHANNELS PER DATASET=
FILENAME 2=DATA5
RUN 1000 10 DATASETS 54340 CHANNELS/DATASET MODE 3
^FILENAME 3=DATA4
RUN 1000 10 DATASETS 54340 CHANNELS/DATASET MODE 3
FILENAME 4=DATA3
RUN 1000 1 DATASET 543400 CHANNELS/DATASET MODE 0
FILENAME 5=
TYPE EQUATION
      PROGRAM
      V9=F4S1SC434721*(F3S9/F2S9)**2
      F1S1=F1S2CCE
      V2=.9559*F2S1/F2S9*1.0025
      F1S2=V2
      F1S3=V2*SQRT(F4S1SC1*(F3S1/F2S1)**2+V9)
      V4=.8678*F2S3/F2S9*1.0025
      F1S4=V4
      F1S5=V4*SQRT(F4S1SC108681*(F3S3/F2S3)**2+V9)
      V6=.6519*F2S5/F2S9*1.0075
      F1S6=V6
      V7=V6*SQRT(F4S1SC217361*(F3S5/F2S5)**2+V9)
      V8=.3907*F2S7/F2S9*1.01
      F1S8=V8
      F1S9=V8*SQRT(F4S1SC326041*(F3S7/F2S7)**2+V9)
      END
TYPE EQUATION
STOP
```

Example 7

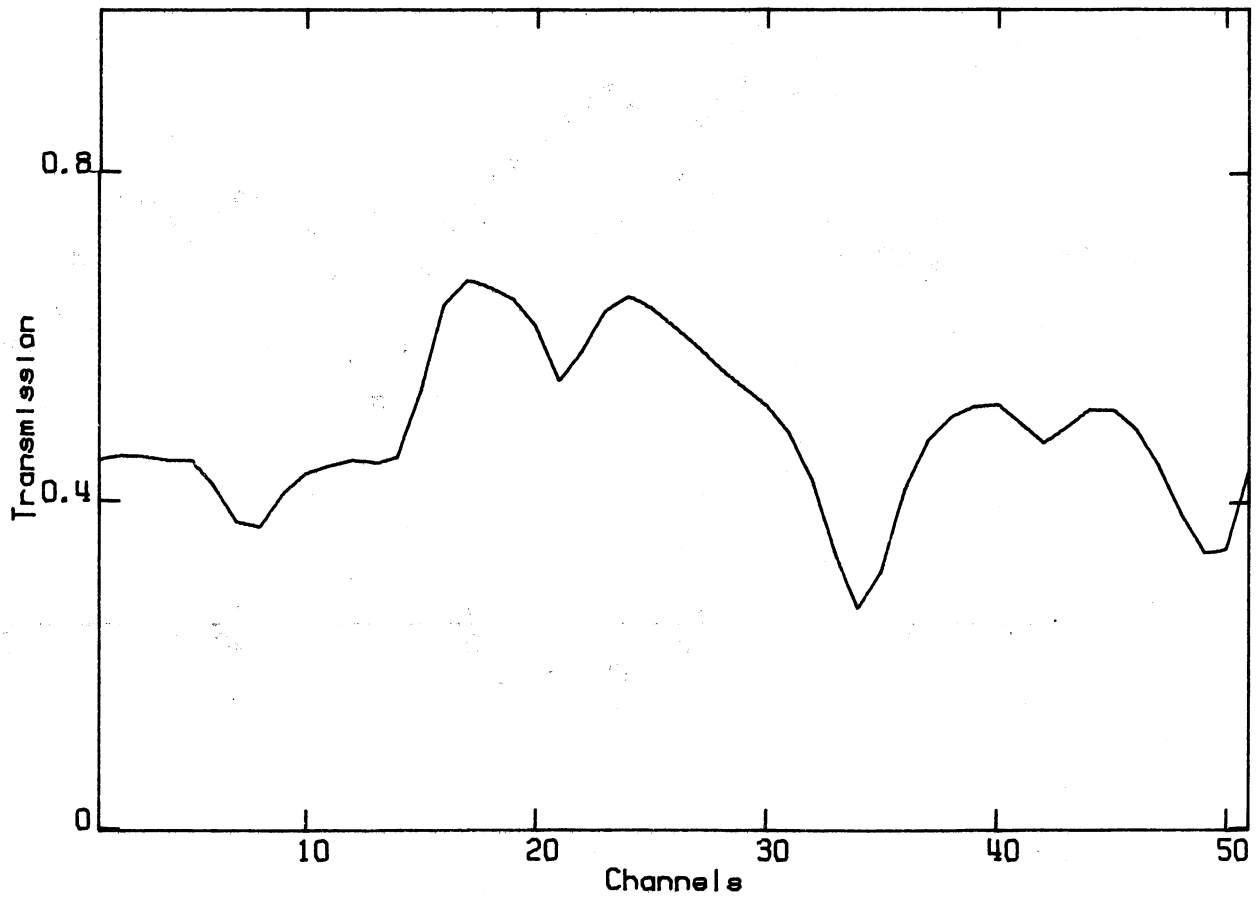
Create the file DATA7 containing energy, cross section, and cross section error for four samples using the file DATA6 which contains energy, transmission, and transmission error.

```
.R OPRODF

FILENAME 1=DATA7.DAT/NEW
FILENAME OF FILE CONTAINING DEFAULT PARAMETERS=DATA6/COPY
FILENAME 2=DATA6
RUN 1000 9 DATASETS 54340 CHANNELS/DATASET MODE 3
FILENAME 3=
TYPE EQUATION
F1S1=F2S1
TYPE EQUATION
F1S2=-265.9*ALOG(F2S2)
TYPE EQUATION
F1S3=265.9*F2S3/F2S2
TYPE EQUATION
F1S4=-80.71*ALOG(F2S4)
TYPE EQUATION
F1S5=80.71*F2S5/F2S4
TYPE EQUATION
F1S6=-19.2*ALOG(F2S6)
TYPE EQUATION
F1S7=19.2*F2S7/F2S6
TYPE EQUATION
F1S8=-5.703*ALOG(F2S8)
TYPE EQUATION
F1S9=5.703*F2S9/F2S8
TYPE EQUATION
STOP
```

Example 8

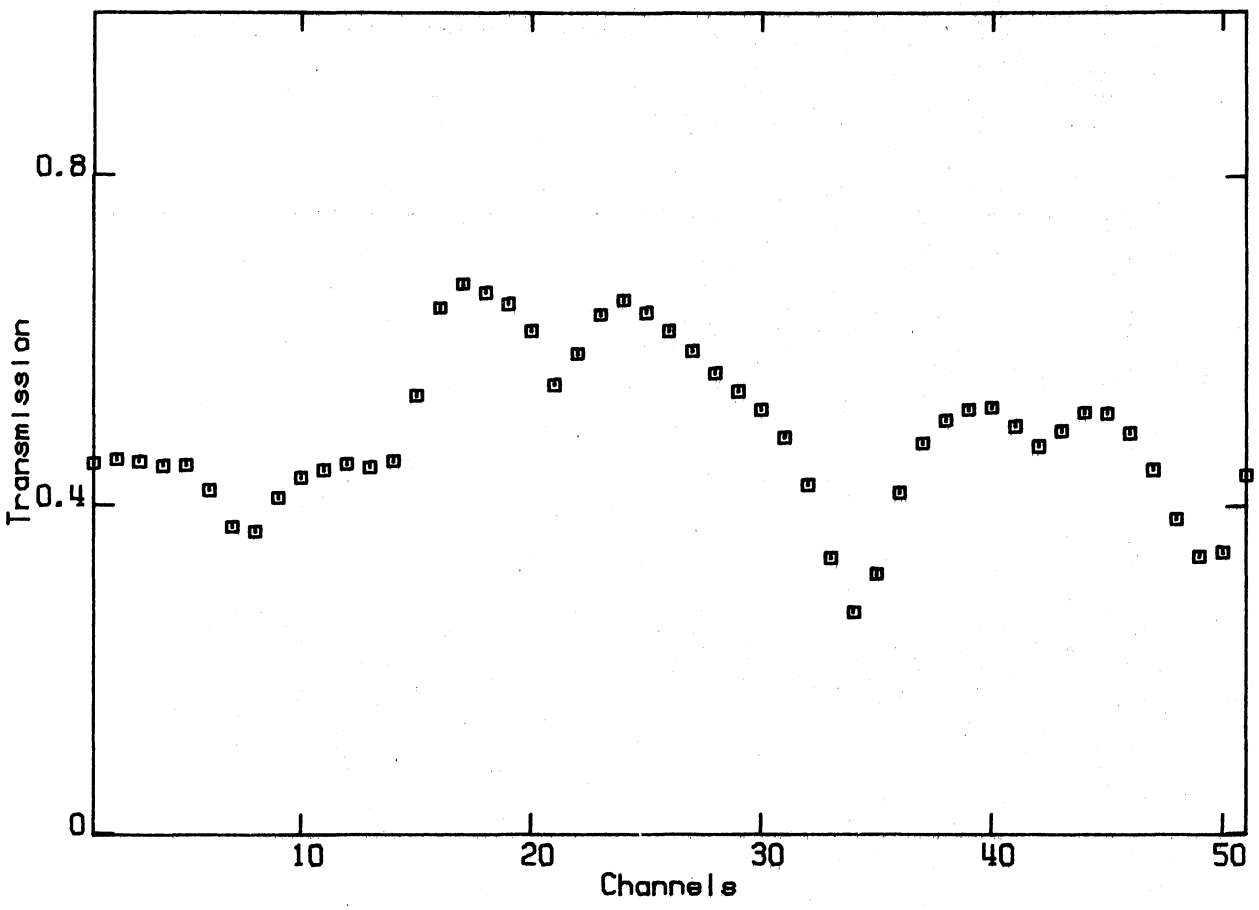
The file NA10 contains energy in dataset 1, data in dataset 2, errors in dataset 3, and fitted curve in dataset 4. Create a line plot of channels 1 through 51 of dataset 2.

```
.R OPRODF

FILENAME 1=NA10
RUN 1000 4 DATASET 2800 CHANNELS/DATASET MODE 3
FILENAME 2=
TYPE EQUATION
PLOT/BTI $|C|HANNELS$/LTI $|T|RANSMISSION$ F1S2SC1EC51
TYPE EQUATION
QSPLT/D
TYPE EQUATION
```



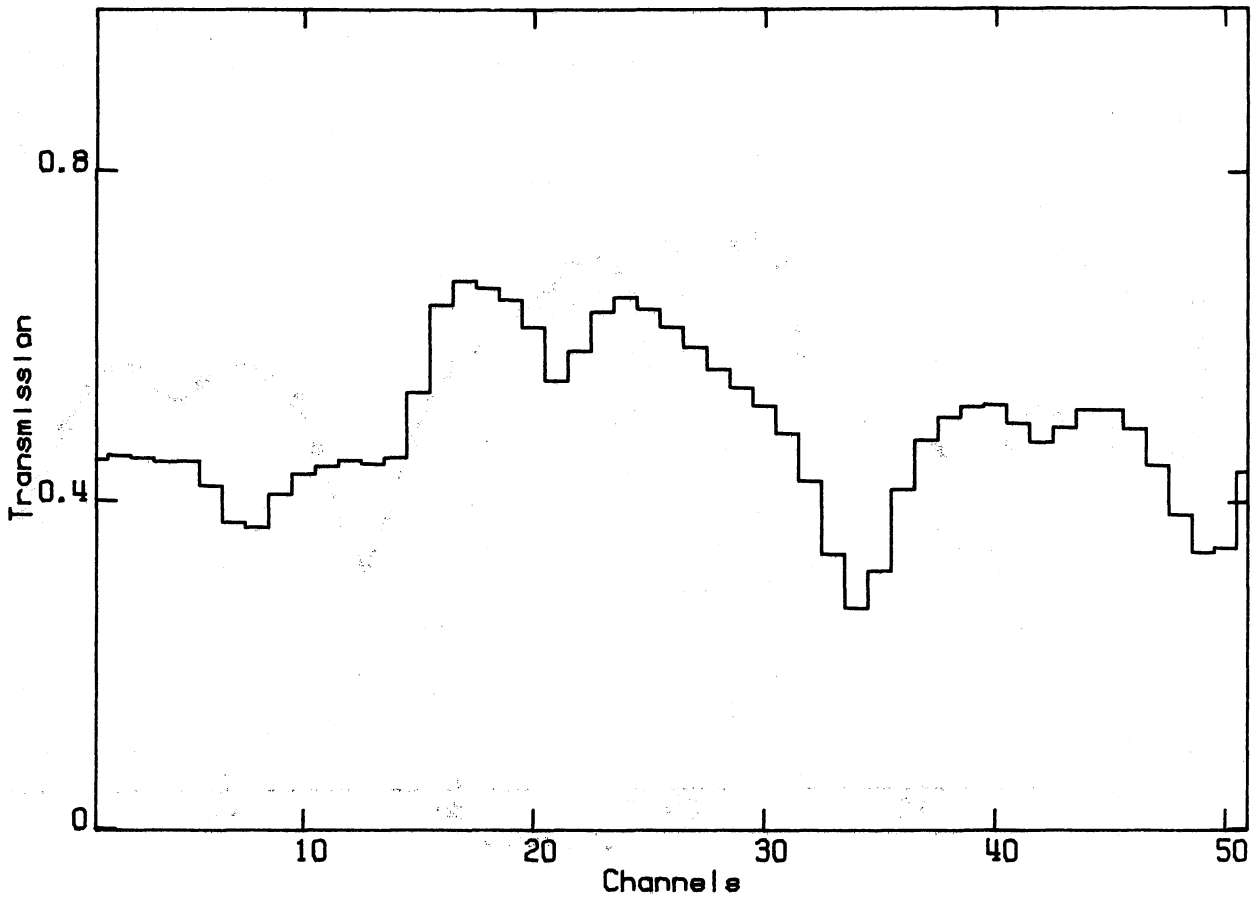
Continuing example 8, create a center symbol plot of channels 1 through 51 of dataset 2.

```
PLOT/SYM0 F1S2SC1EC51
TYPE EQUATION
QSPLT/D
TYPE EQUATION
```



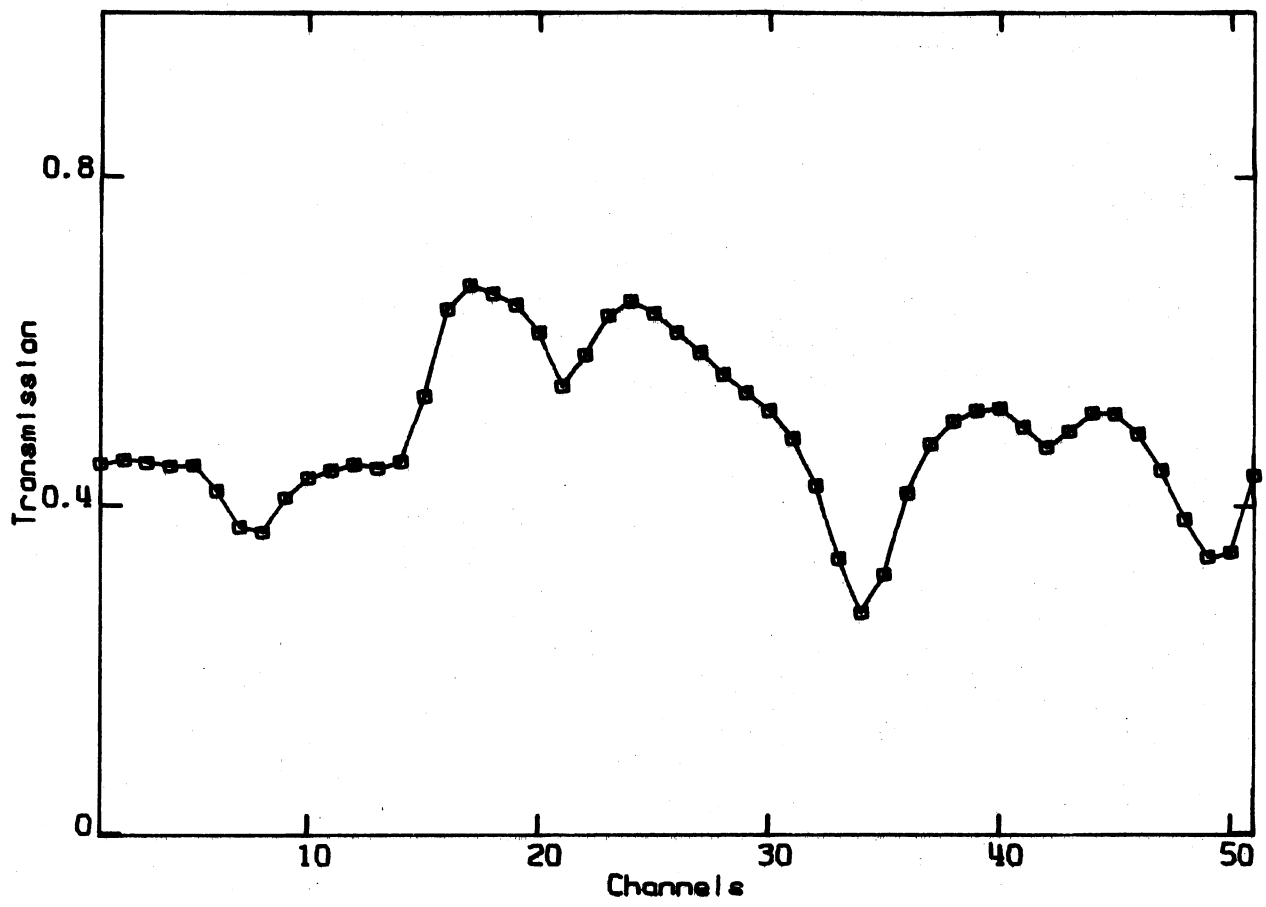
Continuing example 8, create a histogram plot of channels 1 through 51 of dataset 2. Notice the symbol plot switch from the previous plot is reset.

```
PLOT/NOSYM/HIST F1S2SC1EC51
TYPE EQUATION
QSPLT/D
TYPE EQUATION
```



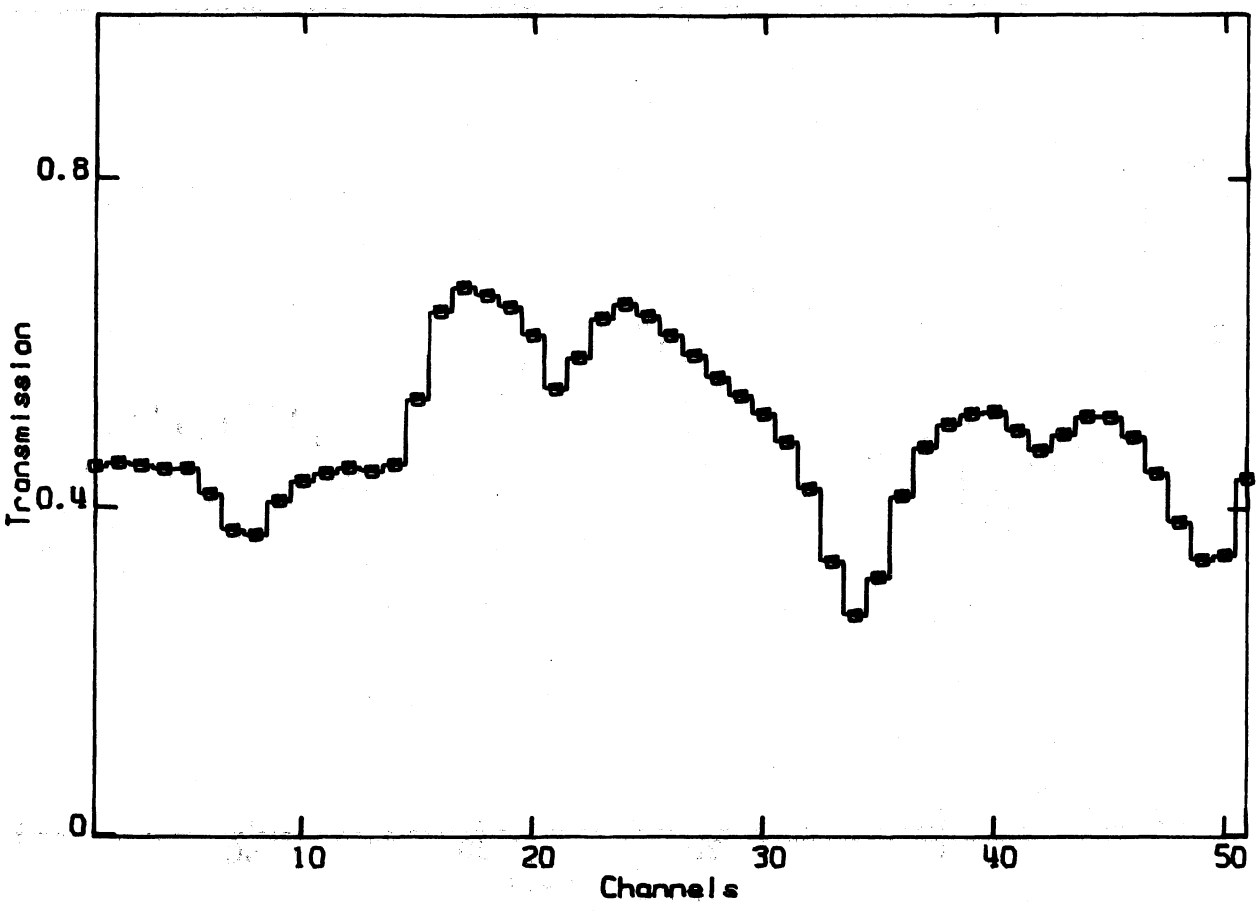
Continuing example 8, create a line center symbol plot of channels 1 through 51 of dataset 2.

```
PLOT/NOHIS/SYM0/LINE F1S2SC1EC51
TYPE EQUATION
QSPLT/D
TYPE EQUATION
```



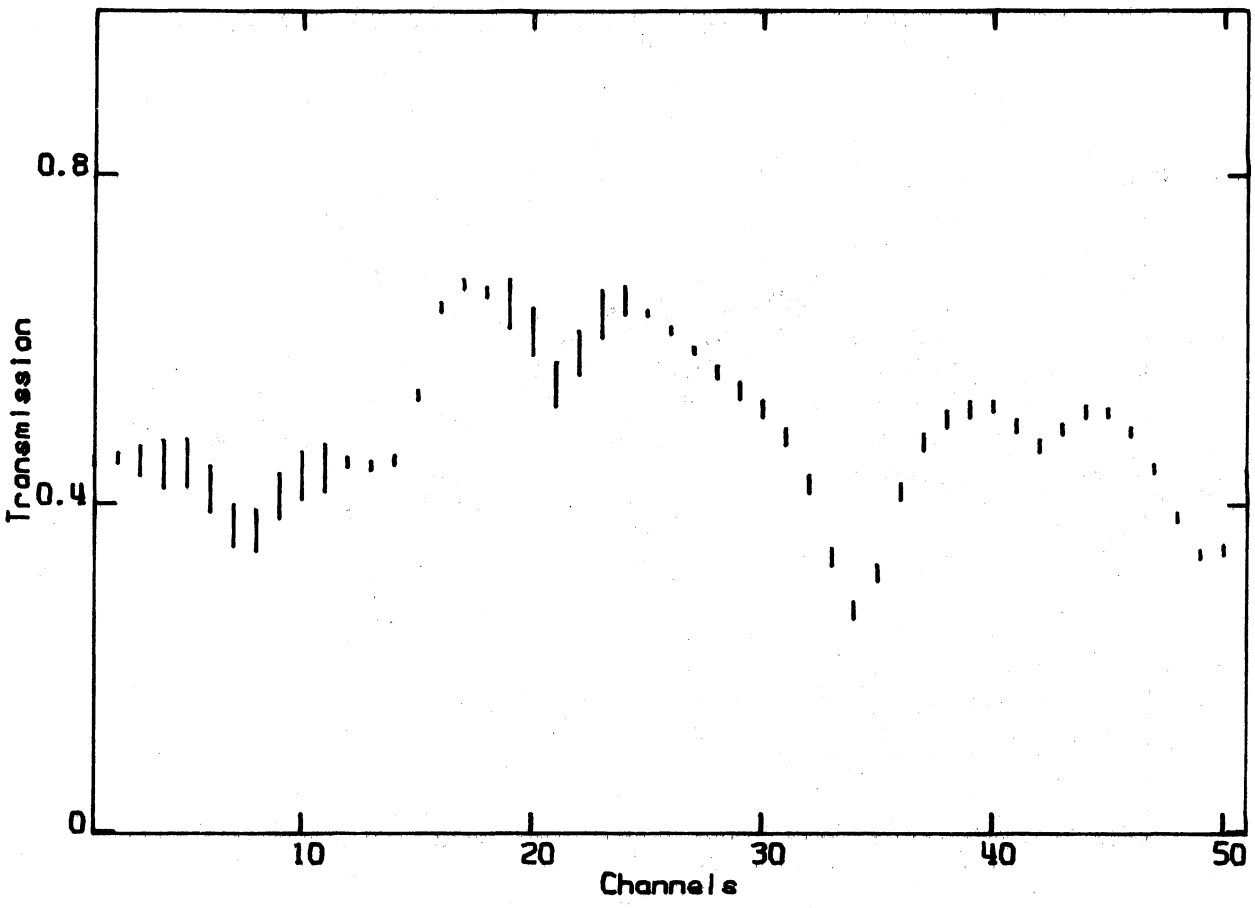
Continuing example 8, create a histogram center symbol plot of channels 1 through 51 of dataset 2.

```
PLOT/SYM0/HIST F1S2SC1EC51
TYPE EQUATION
QSPLT/D
TYPE EQUATION
```



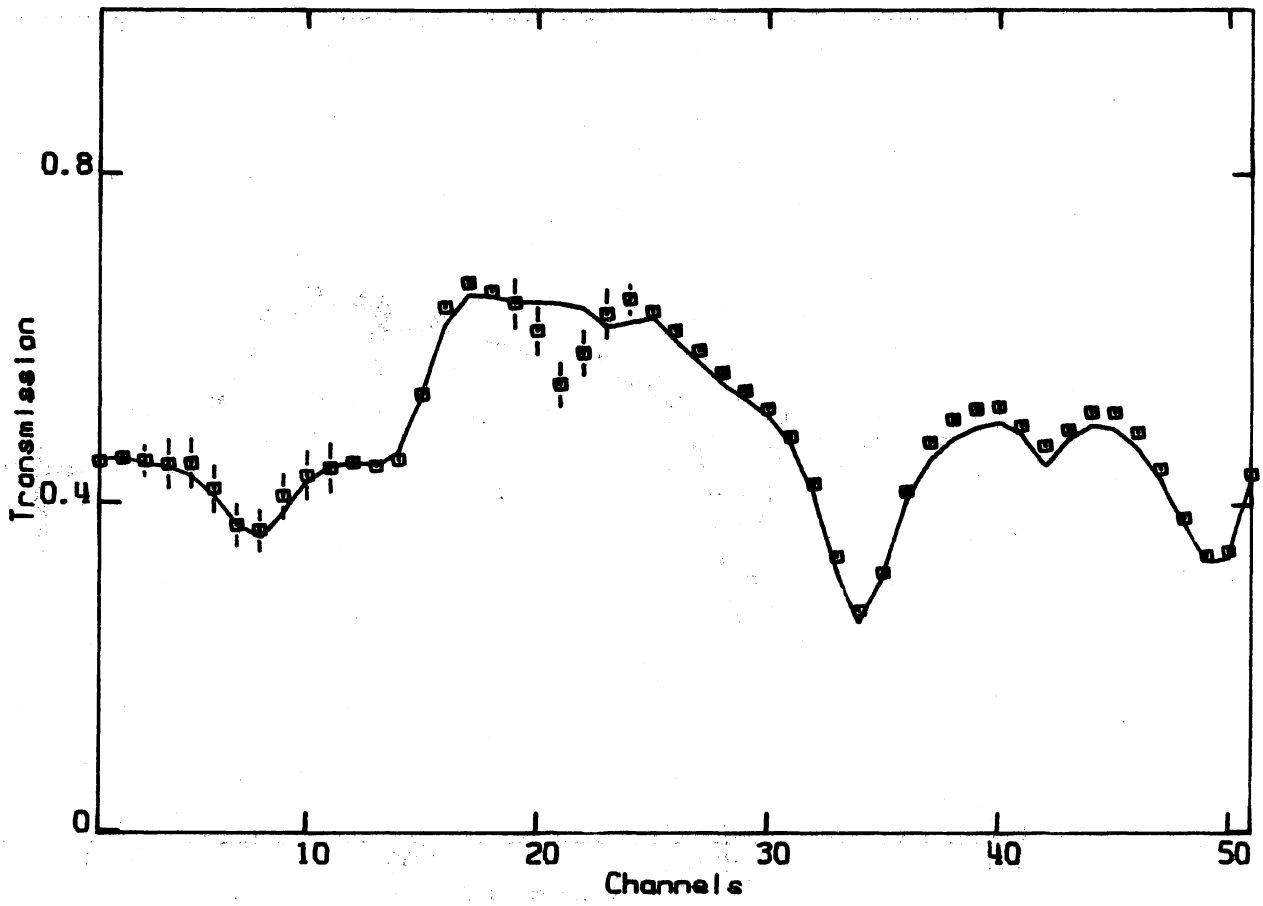
Continuing example 8, create an error bar plot of channels 1 through 51 of dataset 2.

```
PLOT/NOSYM/NOHIS/ERR3/NOLINE F1S2SC1EC51
TYPE EQUATION
QSPLT/D
TYPE EQUATION
```



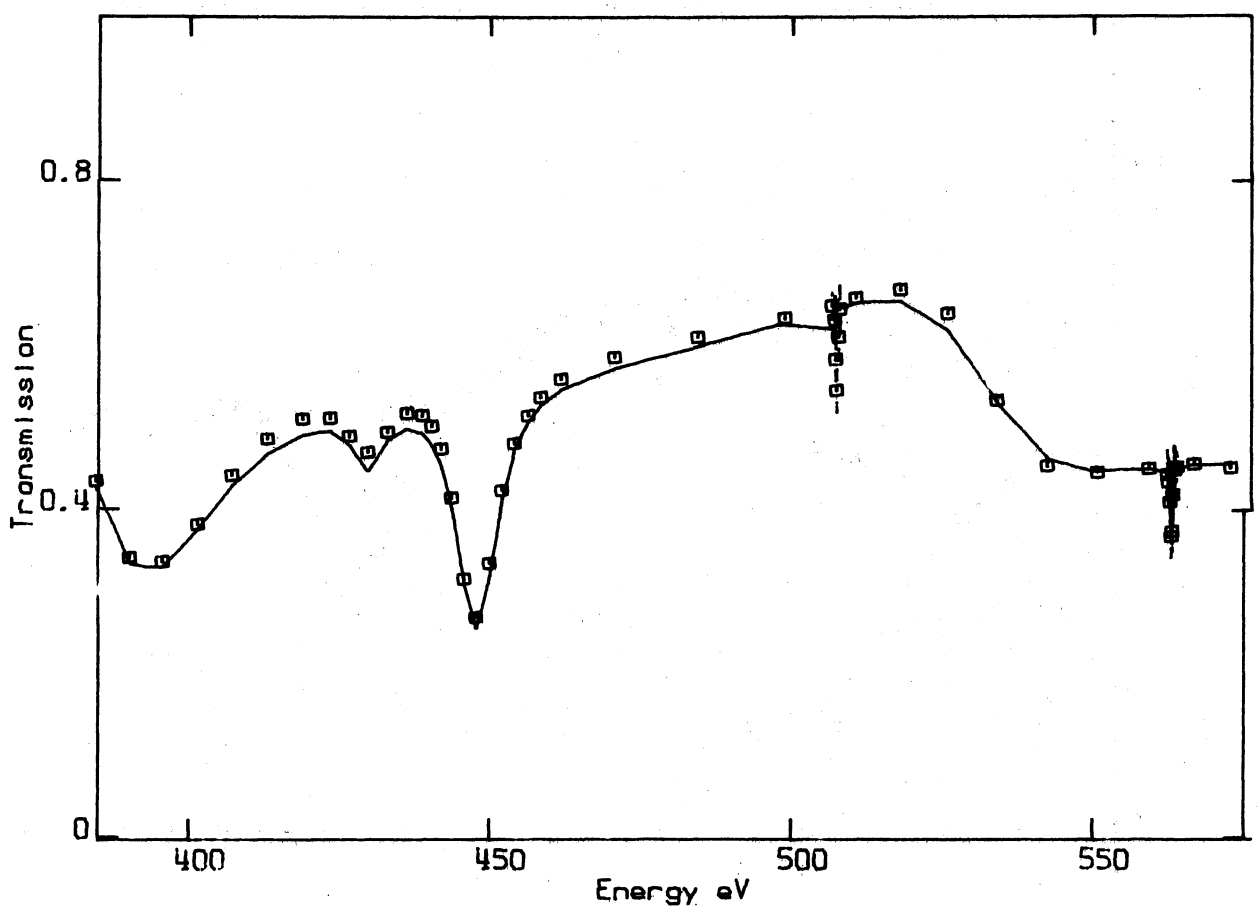
Continuing example 8, create a center symbol plot of channels 1 through 51 of dataset 2, an error plot of dataset 3, and a line plot of dataset 4.

```
PLOT/SYM0/YER3 F1S2SC1EC51,/NOSYM/NOERR F1S4
TYPE EQUATION
QSPLT/D
TYPE EQUATION
```



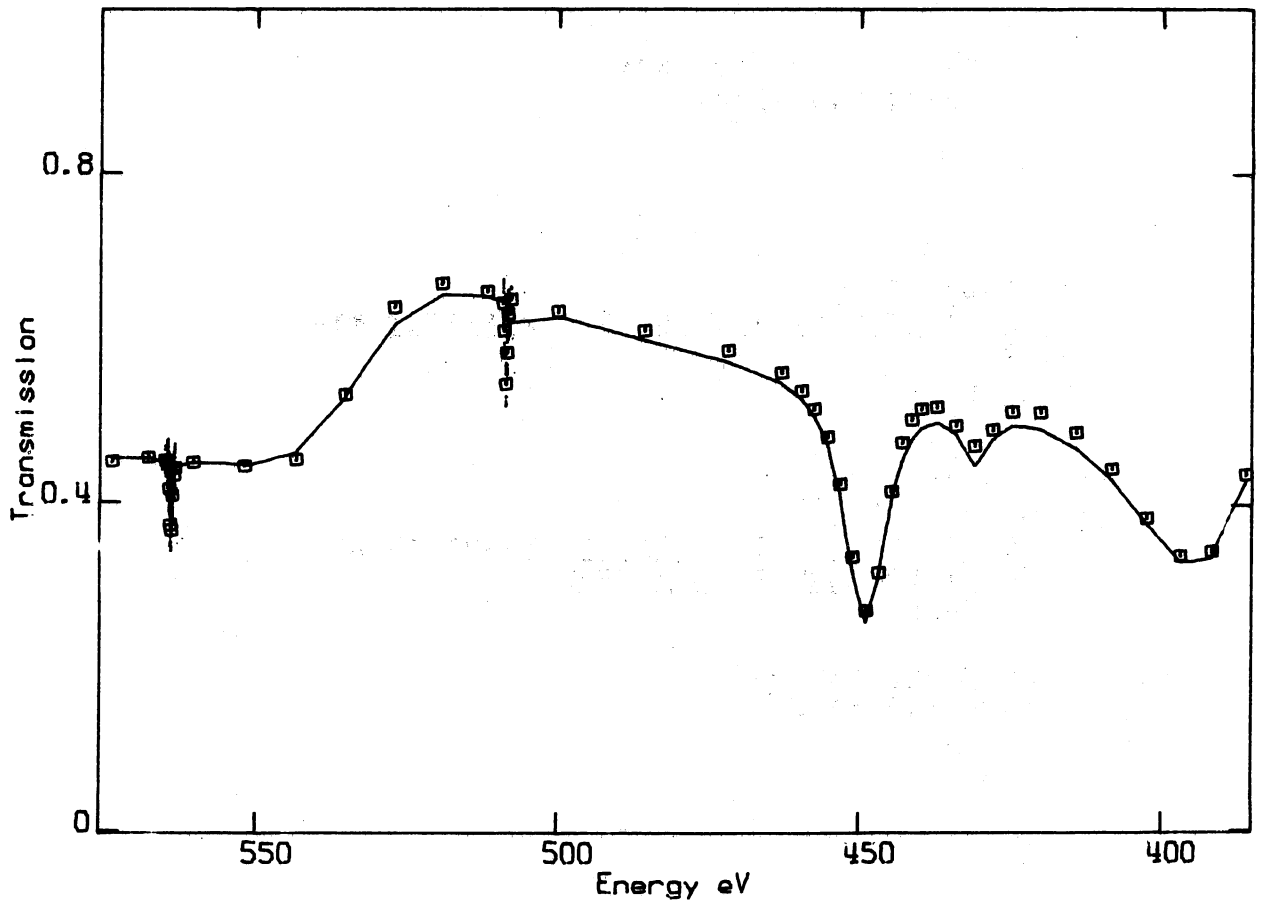
Continuing example 8, create the previous plot using starting energy 385 and ending energy 576.

```
/BTI |E|NERGY E|V$
TYPR EQUATION
PLOT/SYM0/YER3 F1S2SE385EE576,/NOSYM/NOERR F1S4
TYPE EQUATION
QSPLT/D
TYPE EQUATION
```



Continuing example 8, create the previous plot using starting energy 576 and ending energy 385.

```
PLOT/SYM0/YER3 F1S2SE576EE385,/NOSYM/NOERR F1S4
TYPE EQUATION
QSPLT/D
TYPE EQUATION
```



Example 9

Using a command file, create the readable file DATA3.LST containing the channel contents from file DATA3 between two channel numbers which are to be input from the user's terminal along with the dataset number. The command file LIST contains:

```
'DATASET NUMBER="SN'
'STARTING CHANNEL="SC'
'ENDING CHANNEL="EC'
      PROGRAM
      DIMENSION V(10)
      DO 2 I=#SC#,#EC#,10
      DO 1 J=1,10
1     V(J)=F1S#SN#SC#SC#EC#EC#
2     LIST I,(V(J),J=1,10)
      STOP
      END

.R OPRODF

FILENAME 1=DATA3
RUN 1000 1 DATASET 543400 CHANNELS/DATASET MODE 0
FILENAME 2=
TYPE EQUATION
@LIST
DATASET NUMBER=1
STARTING CHANNEL=32
ENDING CHANNEL=75
FILENAME OF FILE TO STORE LISTING IN=DATA3.LST
TYPE PAGE HEADER (END WITH /<CR>)
FILE DATA3 CHANNELS 32-75
/
NUMBER LINES PER PAGE=50
LINE FORMAT=I6,10F10.2
TYPE EQUATION
STOP
```

Example 10

Using a command file, create an ODF dataset from the readable file DATA3.LST created in the previous example. The command file MAKE contains:

```
'DATASET NUMBER="SN'
'STARTING CHANNEL="SC'
'ENDING CHANNEL="EC'
      PROGRAM
      DIMENSION V(10)
      DO 2 I=#SC#,#EC#,10
      MAKE (V(J),J=1,10)
      DO 1 J=1,10
1     F1S#SN#SC#SC#EC#EC#=V(J)
2     CONTINUE
      STOP
      END

.R OPRODF

FILENAME 1=DATA3
RUN 1000 1 DATASET 543400 CHANNELS/DATASET MODE 0
FILENAME 2=
TYPE EQUATION
@MAKE
DATASET NUMBER=1
STARTING CHANNEL=32
ENDING CHANNEL=75
FILENAME OF FILE TO READ LISTING FROM=DATA3.LST
NUMBER OF PAGE HEADER LINE TO SKIP=1
NUMBER LINES PER PAGE=50
LINE FORMAT=I6,10F10.2
TYPE EQUATION
STOP
```

Example 11

Create the readable file DATA7.LST containing the channel contents of the file DATA7.

```
.R OPRODF

FILENAME 1=DATA7
RUN 1000 9 DATASET 54340 CHANNELS/DATASET MODE 3
FILENAME 2=
TYPE EQUATION
LIST F1S1SC35EC75,F1S2,F1S3
TYPE THE PAGE HEADER (END WITH /<CR>)
ENERGY SIGMA ERROR
/
NUMBER LINES PER PAGE=50
LINE FORMAT=I6,2F6.2
TYPE EQUATION
STOP
```

Example 12

Using a command file, type the channel sum between two energies that are to be input from the user's terminal. The command file SUM contains:

```
'STARTING ENERGY="SE'
'ENDING ENERGY="EE'
'DATASET NUMBER="SN'
      PROGRAM
      SE=F1SE#SE#CSE
      EE=F1SE#SE#CEE
      COUNT=F1S#SN#SE#SE#
      SUM=((#SE#-EE)/(SE-EE))*COUNT-COUNT
1     COUNT=F1S#SN#SE#SE#
      SUM=SUM+COUNT
      EE=F1SE#SE#CEE
      IF(EE.GT.#EE#)GO TO 1
      SE=F1SE#EE#CSE
      SUM=SUM-((#EE#-EE)/(SE-EE))*COUNT
      END
'SUM='SUM


.R OPRODF

FILENAME 1=DATA5.DAT
RUN 1000 10 DATASETS 54340 CHANNELS/DATASET
FILENAME 2=
TYPE EQUATION
@SUM
STARTING ENERGY=100000
ENDING ENERGY=150000
DATASET NUMBER=2
SUM=  0.1543766E+9
TYPE EQUATION
STOP
```

Example 13

Using a command file, type the integral sum between two energies that are to be input from the user's terminal. The command file INTGRL contains:

```
'STARTING ENERGY="SE'
'ENDING ENERGY="EE'
'DATASET NUMBER="SN'
      PROGRAM
      SE=F1SE#SE#CSE
      EE=F1SE#SE#CEE
      SUM=((SE-#SE#)/(SE-EE))*
     XF1S#SN#SE#SE#*(SE-EE)*-1
1     SE=F1SE#SE#CSE
      EE=F1SE#SE#CEE
      SUM=SUM+F1S#SN#SE#SE#*(SE-EE)
      IF(EE.GT.#EE#)GO TO 1
      SUM=SUM-(((#EE#-EE)/(SE-EE))*
     XF1S#SN#SE#EE#*(SE-EE))
      STOP
      END
'SUM='SUM


.R OPRODF

FILENAME 1=DATA5.DAT
RUN 1000 10 DATASETS 54340 CHANNELS/DATASET
FILENAME 2=
TYPE EQUATION
@INTGRL
STARTING ENERGY=100000
ENDING ENERGY=150000
DATASET NUMBER=2
SUM=  0.1543766E+9
TYPE EQUATION
STOP
```

Example 14

Using a command file average dataset 2 of file DATAX by .02 eV storing the result in dataset 7 beginning at channel 10. The command file ENGAVR contains:

```
'STARTING ENERGY="SE'
'ENDING ENERGY="EE'
'ENERGY SPAN="EI'
'SECTION NUMBER="SN'
'OUTPUT SECTION="OS'
'STARTING CHAN="SC'
C
      PROGRAM
      CSE=F1SE SE CSE
      CEE=F1SE SE CEE
      CC=F1S SN SE SE
      CE= SE
      HEI= EI /2
1     SUM=((CSE-CE) / (CSE-CEE))* (CC * (CSE-CEE))*-1
      CE=CE- EI
      IF(CE.LT. EE )STOP
2     CSE=F1SE SE CSE
      CEE=F1SE SE CEE
      CC=F1S SN SE SE
      SUM=SUM+CC*(CSE-CEE)
      IF(CEE.GT.CE)GO TO 2
      SUM=SUM-(((CE-CEE) / (CSE-CEE)) * CC*(CSE-CEE))
      F2S1SC SC =CE+HEI
      F2S OS SC SC =SUM
      CCN=CCN+1
      GO TO 1
      END

.R OPRODF

FILENAME 1=DATAX
RUN 1000 9 DATASET 54340 CHANNELS/DATASET MODE 3
FILENAME 2=
TYPE EQUATION
@ENGAVR
STARTING ENERGY=.5
ENDING ENERGY=.9
ENERGY SPAN=..02
SECTION NUMBER=2
OUTPUT SECTION=7
STARTING CHAN=10
TYPE EQUATION
STOP
```

Example 15

Set the TZERO and flight path distance of dataset 2 in the file DATA7 to -2.695 microseconds and 155.442 meters.

```
.R OPRODF

FILENAME 1=DATA7
RUN 1000 9 DATASET 54340 CHANNELS/DATASET MODE 3
FILENAME 2=
TYPE EQUATION
F1S2TZERO=-2.695
TYPE EQUATION
F1S2FPD=155.442
TYPE EQUATION
F1S2TZERO
 -0.2695000E+01
TYPE EQUATION
F1S2FPD
  0.1554420E+03
TYPE EQUATION
STOP
```

Example 16

Type out the dataset table in the file DATA7.

```
.R OPRODF

FILENAME =DATA7
RUN 1000 9 DATASET 54340 CHANNELS/DATASET MODE 3
FILENAME 2=
TYPE EQUATION
F1TABLE

STARTING    STARTING CRUNCH
  TIME      CHANNEL  FACTOR
    0.0000        1      40
   30.0000      751       4
  120.0000    23251       8
  200.0000    33251      16
  300.0000    39501      32
  500.0000    45751      64
  756.0000    49751     128
 1076.0000    52251    1000
 1106.0000    52281     100
 1196.0000    53181     200
 1276.0000    53581     400
 1376.0000    53831     800
 1576.0000    54081    1600
 1832.0000    54241    3200
 2152.0000    54341       0

TYPE EQUATION
STOP
```

Example 17

Insert some comments into the file DATA7.

```
.R OPRODF

FILENAME 1=DATA7
RUN 1000 1 DATASET 543400 CHANNELS/DATASETMODE 3
FILENAME 2=
TYPE EQUATION
F1COMMENTS=
FILE CREATED FROM DATA6 7-22-76

TYPE EQUATION
F1COMMENTS
FILE CREATED FROM DATA6 7-22-76
TYPE EQUATION
STOP
```

19. THE ORELA DATA FORMATTED FILE

The ORELA Data Formatted, ODF, file provides a storage format for up to 63 common datasets, either half-word integers, full-word integers, or floating point, provides the user with the ability to store information about the data sets with the datasets such as comments, program variables, and data acquistion constants; provides the fastest simultaneous access to multiple datasets; provides the ability to easily change part of a dataset; and provides a standard storage format for use by all data acquisition, display and analysis programs. An ODF file may consist only of the datasets. ODF information (scalers, comments, parameters, etc.) need not exist.

Programs and subroutines written to use the ODF file structure are stored in the area OREL:[101,1010]. The subroutines are stored in the library file OREL:ORELIB[101,1010] in this area. Each program and subroutine in this area has a documentation file of the same name with the extension "ORL". The source files for these programs and subroutines are in the area OR05:[101,1010].

HEADER BLOCK

(WORD)	(SYMBOL) (BITS)
1	
2	...(0-11), MODE(12-14), SOURCE(15-17), ID(18-35)
3	NCBLKS(0-17), NCWRDS(18-35)
4	NSBLKS(0-17), NSWRDS(18-35)
5	NCSTRT(0-17), NCNTRS(18-35)
6	NXSTRT(0-17), NXWRDS(18-35)
7	NPBLKS(0-17), NPWRDS(18-35)
8	NDTYPE(0), NDVARS(1-6), NDBLKS(7-14), NDWRDS(15-35)
9	NDBEND(0-17), NDWEND(18-35)
10	ZAN(0-15)
11	AWR(0-35)
12	MAT(0-35)
13	MF(0-35)
14	MT(0-35)
15	VARSWT(0-35)
16	DTCSWT(0-35)
17	INSTRT(0-35)
18	ORELNK(0-35)
19-26	Not used.
27-126	Energy index table.

COMMENT SECTION; pointed to by NCBLKS.

SCALER SECTION; pointed to by NSBLKS.

PARAMETER SECTION; pointed to by NPBLKS.

DATA SECTION; pointed to by NDBLKS.

19.1 Header block

(WORD)	(SYMBOL)	(MEANING OR CONTENT)
1		Not used.
2	MODE	=0; The data section contains 18-bit integer values stored two per word.
		=1; The data section contains 36-bit integer values.
		=3; The data section contains floating point values.
	SOURCE	=0; The file was created with SEL data.
		=1; The file was created with CSISRS data.
		=2; The file was created with ENDF/B data.
	ID	The numerical ID given to the file by the user.
3	NCBLKS	The starting block number of the comment section.
	NCWRDS	The number of words written in the comment section.
4	NSBLKS	The starting block number of the scaler section.
	NSWRDS	The number of words written in the scaler section.
5	NCSTRT	The starting word number of the data acquisition scaler and counter section in the scaler section.
	NCNTRS	The number of words written in the data acquisition scaler and counter section.
6	NXSTRT	The starting word number of the data acquisition variable section in the scaler section.
	NXWRDS	The number of words written in the data acquisition variable section.

7	NPBLKS	The starting block number of the parameter section.
	NPWRDS	The number of words written in the parameter section.
8	NDTYPE	=0; The storage of the data in the data section is described by the table in the parameter section.
		=1; The storage of the data in the data section is dependent on the data stored in dataset 1. If dataset 1 contains energy, dataset 2 transmission, dataset 3 transmission error, dataset 4 cross section, dataset 5 cross section error, etc., then the content of each channel in dataset 2 through x corresponds to the energy from that channel in dataset 1.
	NDVARS	The number of datasets in the data section.
	NDBLKS	The starting block number of the data section.
	NDWRDS	The number of words in each dataset.
9	NDBEND	The last block number written between block NPBLKS and NDBLKS.
	NDWEND	The ending word number of the last word written in block NDBEND.
10	ZAN	The ENDF/B designation (charge,mass).
11	AWR	The ENDF/B ratio of the nuclear mass of the material (isotope, element, molecule, or mixture) to that of the neutron.

12 MAT	The ENDF/B mat number assigned by the NATIONAL NEUTRON CROSS SECTION CENTER.
13 MF	The ENDF/B file number (class of data).
14 MT	The ENDF/B reaction type number.
15 VARSWT	VARSWT is ignored if NDTYPE is zero. =0; Dataset 1 decreases in value. =1; Dataset 1 increases in value.
16 DTCSWT	=0; The data has not been dead-time corrected. =1; The data has been dead-time corrected.
17 INSTRT	The starting word number of each dataset in the data section (mode 0 only).
18-26	Not used.
27-126	The energy index table. Each word contains the largest energy for each n blocks in the data set. N is equal to (NDWRDS/125)+1.

19.2 Comment block

The starting block number of the comment section (NCBLKS) can be found in block 1, word 3, bits 0-17. The number of words in the comment section (NCWRDS) can be found in block 1, word 3, bits 18-35. The total number of words that can be written into the comment section is the difference between the starting block of the scaler section (NSBLKS) and the starting block of the comment section (NCBLKS) times 126 ((NSBLKS-NCBLKS)*126). The comment section contains seven-bit ASCII characters stored five per word.

19.3 Scaler block

The starting block number of the scaler section (NSBLKS) can be found in block 1, word 4, bits 0-17. The number of words in the scaler section (NSWRDS) can be found in block 1, word 4, bits 18-35. The scaler section is made up of ODF file variables: data acquisition counters, scalers, and variables.

File variables

The starting word number of the file variables in the scaler section is 1. The number of file variables in the scaler section is one less than the starting word number of the data acquistion counters (NCSTRT). File variables are used by the user's programs to store program variables for future use. The program ORELNK allocates 16 variables when converting from an SEL file to an ODF file.

Data acquisition counters and scalers

The starting word number of the data acquisition counters and scalers (NCSTRT) in the scaler section can be found in block 1, word 5, bits 0-17. The number of data acquisition counters and scalers (NCNTRS) in the scaler section can be found in block 1, word 5, bits 18-35. The data acquisition counters and scalers are readings from the data acquisition scalers and the software counters from the data acquisition computers. The counters and scalers are stored right-adjusted (bits 4-35), two per word.

Data acquisition variables

The starting word number of the data acquisition variables (NXSTRT) can be found in block 1, word 6, bits 0-17. The number of data acquisition variables (NXWRDS) in the scaler section can be found in block 1, word 6, bits 18-35. The data acquisition variables are those used by the data acquisition program. The variables are stored right-adjusted (bits 4-35) two per word.

19.4 Parameter section

The starting block number of the parameter section (NPBLKS) can be found in block 1, word 7, bits 0-17. The number of words in the parameter section (NPWRDS) can be found in block 1, word 7, bits 18-35. The parameter section contains the dataset TZEROs, flight paths, and data acquisition parameters.

TZERO values

The starting word number of the TZERO values in the parameter section is 2. The number of TZEROs values in the parameter section is equal to the number of datasets in the data section. The TZERO value is the starting time in microseconds of the first channel in each dataset. The first TZERO goes with the first dataset, the second TZERO goes with the second dataset, etc.

Flight path values

The starting word number of the flight path values in the parameter section is equal to one-half of the content of the first word of the parameter section minus 2 ((I-2)/2). The number of flight path values is equal to the number of datasets in the data section. The flight path value is the flight path length in meters that was used during data acquisition. The first flight path goes with the first dataset, the second flight path goes with the second dataset, etc.

Data acquisition parameter section

The starting word number of the data acquisition parameter section in the parameter section is the content of the first word of the parameter section. The last word number of the data acquisition parameter section in the parameter section is the last word number of the parameter section. The data acquisition parameter section contains the parameters used to create the dataset during data acquisition which are:

WORD	CONTENT
1	CLOCK TIC TIME
2	NUMBER ENTRIES IN TWO PARAMETER TOF TABLE
3	NUMBER ENTRIES IN ONE PARAMETER PH TABLE
4	NUMBER ENTRIES IN TWO PARAMETER TOF TABLE
5	NUMBER ENTRIES IN ONE PARAMETER TOF TABLE
6	NUMBER ENTRIES IN ONE PARAMETER PH TABLE
7	STARTING TIME THIS CRUNCH
8	STARTING CHANNEL THIS CRUNCH
9	CRUNCH FACTOR
10	STARTING TIME THIS CRUNCH
11	STARTING CHANNEL THIS CRUNCH
12	CRUNCH FACTOR
13	ETC.
A	ENDING TIME PLUS ONE PREVIOUS CRUNCH
B	ENDING CHANNEL PLUS ONE PREVIOUS CRUNCH
C	ZERO
D	TOTAL NUMBER CHANNELS USED BY THIS TABLE
E	NUMBER OF FOLLOWING TAG BIT SECTION NUMBERS
F	STORAGE SECTION NUMBER FOR TAG BIT VALUE 0
G	STORAGE SECTION NUMBER FOR TAG BIT VALUE 1
H	ETC.

Word 1 contains the clock tic time in nanoseconds. Originally the left half (bits 0-17) of the clock tic time contained the integer portion and the right half (bits 18-35) contained the fractional portion times 10000. Some programs change the clock tic time in the dataset table to a PDP-10 floating point number. To obtain the time from either format use the coding:

```
ITIME=IPARAM(IPARAM(1))
IF(ITIME.GT."777777777) GO TO XX
TIME=ITIME/"1000000+((ITIME.AND."777777)/1ed0000.)
```

where TIME and ITIME have been equivalenced.

Words 2 through 6 contain the number of channel group entries in each table. In general the tables are time of flight (TOF), and pulse height (PH). However, the tables may be used for any type of data. Only two of the two parameter tables may be used (TOF vs PH /or/ PH vs TOF) making a maximum of three types of datasets per set of tables. If the number of table entries is less than zero, no table was generated for that entry. If the number of table entries is zero, then the table consists only of words A through H. If the number of table entries is negative, then the positive value of that entry is the total number of words used by the table(s). If word H of the last table is not the last word used by the tables as indicated by the negative number of entries, then another table follows word H.

Word 7 contains the starting time of a group of channels whose starting channel number is in word 8 and whose crunch factor is in word 9. The table starting time has no units. To obtain the starting time, multiply the table starting time by the clock tic time. The crunch factor is the number of input channel that went into each output channel. The time width of each channel is equal to the channel crunch factor times the clock tic time. Words 7 through 9 are repeated for each table entry.

Words F through words F plus the content of word E contain the tag bit section numbers. If the content of F is zero, words F through H do not exist. The tag bit section numbers are valid only with mode zero data. When the file is changed from mode 0 to mode 3, the datasets described by each table will be put in a separate file and each tag bit section for that table will be a dataset in the data section. The first section number (F) is the section where the data was stored for a tag value of zero, the second section number (G) is the section where the data was stored for a tag value of 1, etc. Each tag section corresponds to the dataset table. Section number 0 starts at the channel number in word 7, section number 1 follows section 0, etc.

19.5 Data section

The datasets stored in the data section are blocked into 126 word blocks. Each block consists of 1 control word and 125 data words. A block of words from each dataset is stored in the data section one after the other in a round-robin style. If there are

two datasets in the data section, then every other block in the data section goes to make up a dataset. If there are three datasets in the data section, then every third block goes to make up a dataset. Mode 0 ODF files are treated as having only one dataset.

20. READING THE ODF HEADER BLOCK

The following code reads in the header block (block 1) and extracts the pointers and counters of the file.

```
      DOUBLE PRECISION FILE
      DIMENSION BUF(126),IBUF(126),EBUF(100)
      EQUIVALENCE( BUF(1), IBUF(1) )
      EQUIVALENCE( BUF(10), ZAN)
      EQUIVALENCE( BUF(11), AWR)
      EQUIVALENCE( BUF(12), MAT)
      EQUIVALENCE( BUF(13), MF)
      EQUIVALENCE( BUF(15), VARSWT)
      EQUIVALENCE( BUF(16), DTCSWT)
      EQUIVALENCE( BUF(17), INSTRT)
      EQUIVALENCE( BUF(18), ORELNK)
      EQUIVALENCE( BUF(27),EBUF(1) )
      DATA IU/20/
      FILE='DATA.DAT'
      OPEN(UNIT=IU,FILE=FILE,ACCESS='RANDOM',RECORD=126)
      READ(IU 1)BUF
      MODE=(IBUF(2).AND."70000000)/"10000000
      ISRC=(IBUF(2).AND."7000000)/"1000000
      ID=IBUF(2).AND."777777
      NCBLKS=IBUF(3)/"1000000
      NCWRDS=IBUF(3).AND."777777
      NSBLKS=IBUF(4)/"1000000
      NSWRDS=IBUF(4).AND."777777
      NCSTRT=IBUF(5)/"1000000
      NCNTRS=IBUF(5).AND."777777
      NXSTRT=IBUF(6)/"1000000
      NXWRDS=IBUF(6).AND."777777
      NPBLKS=IBUF(7)/"1000000
      NPWRDS=IBUF(7).AND."777777
      NDTYPE=IBUF(8)
      NDVARS=(IBUF(8).AND."374000000000)/"4000000000
      NDBLKS=(IBUF(8).AND."370000000)/"10000000
      NDWRDS=IBUF(8).AND."777777
      NDBEND=IBUF(9)/"1000000
      NDWEND=IBUF(9).AND."777777
```

21. READING THE ODF DATA SECTION

The following code reads channels ISC through IEC from the mode 0 file open on channel IU into the array WBUF.

```
      IBLK=NDBLKS+(ISC-1+INSTRT*2)/250
      READ(IU#IBLK)ITBUF
      ISIDE=ISC-1-((ISC-1+INSTRT*2)/250)*250
      IPNT=ISIDE/2+1
      IF(ISIDE.EQ.0)ISIDE=-1
      IF(ISIDE.GT.0)IPNT=IPNT+1
      DO 3 I=1,IEC-ISC+1
      ISIDE=-ISIDE
      IF(ISIDE.LT.0)GO TO 2
      IPNT=IPNT+1
      IF(IPNT.LE.126)GO TO 1
      IPNT=2
      IBLK=IBLK+1
      READ(IU#IBLK)ITBUF
1     WBUF(I)=FLOAT(ITBUF(IPNT)/"1000000).AND."777777)
      GO TO 3
2     WBUF(I)=FLOAT(ITBUF(IPNT).AND."777777)
3     CONTINUE
```

The following code reads from section ISN channels ISC through channel IEC from the mode 3 file opened on unit IU into the array WBUF.

```
      IBLK=NDBLKS+(ISC-1/125)*NDVARS+ISN-1
      IPNT=ISC-((ISC-1)/125)*125+1
      READ(IU IBLK)TBUF
      DO 1 I=1,IEC-ISC+1
      IPNT=IPNT+1
      IF(IPNT.LE.126)GO TO 1
      IPNT=2
      IBLK=IBLK+NDVARS
      READ(IU IBLK)TBUF
1     WBUF(I)=TBUF(IPNT)
```

DISTRIBUTION

1. L. L. Anthony
2. N. A. Betz
3. A. A. Brooks
4. R. D. Burris
5. H. P. Carter/ CSD X-10 Library
6. G. T. Chapman
7. J. T. Ching
8-27. J. G. Craven
28. J. S. Crowell
29. J. W. T. Dabbs
30. J. K. Dickens
31. J. D. Drischler
32. J. L. Fowler
33. C. Y. Fu
34. C. A. Giles
35. R. Gwin
36. J. Halperin
37. J. A. Harvey
38. R. W. Henderson
39. R. F. Hibbs
40. R. W. Ingle
41. C. H. Johnson
42. C. O. Kemper
43. D. C. Larson
44. R. L. Macklin
45. F. C. Maienschein
46. R. D. McCulloch
47. J. W. McConnell
48. G. L. Morgan
49-58. ORELA Library
59. D. K. Olsen
60. R. W. Peelle
61. C. M. Perey
62. F. G. Perey
63. R. B. Perez
64. H. Postma
65. C. E. Price
66. G. B. Raine
67. S. Raman
68. G. de Saussure
69. R. J. Sentell
70. G. G. Slaughter
71. R. R. Spencer
72. C. R. Stewart
73. C. L. Thompson
74. J. H. Todd
75. P. R. Vanstrum
76. G. W. Westley
77. L. W. Weston
78. G. E. Whitesides
79-80. Cen. Res. Library
81. Doc. Ref. Sec., Y-12
82-84. Laboratory Records
85. Laboratory Records-RC
86. ORNL Patent Office

87-113. Tech. Info. Center, DOE

114. Research & Tech. Support Div., ORO

115. Chief, Mathematics and Geoscience Branch, DOE, Washington, DC 20545

116. J. N. Rogers, Division 8324, Sandia Laboratories, Livermore, CA 94550