```
ASCII
;     11      32  ASCII          999
  3He-Rich Event Sources
#0 Rate
I      -1      5000    *3He Flux (x10**-3)
I2      0      5000    *3He/4He (%)
I       0      100     *3He/4He Pct. Err.
I      -1      50      X-ray Duration (min)
I      -1      10000   Soft X-ray Peak Int. (/10)
F      -1      35.0    X-Ray Temp (x10**6 K)
I      -1      4       M III Intensity
F      -1     13.0     Km III Intensity
I      -1     5000     SMM Peak (C/S)
I      -1      600     HXR Spect. Index (x100)
I       0    10000     Time? (2000Y+100M+D)
#End
```

defines verse type # 0

verse type #

```
0      20   36      10     28       8    918      16.3     3      11.9      -1      -1  1108    1978 Nov  8  1751
0      20   2600    1000   38       5     50      12.6     3       8.6      -1      -1  1127         Nov 27  2055
0     200   310     60     19       4      5       8.7     0       9.2      -1      -1  1226         Dec 26
0     100   80      20     25      -1     -1      -1       0      8.32      -1      -1  1226         Dec 26  2104
0     150   68      10     15       0     -1      -1       0       9.7      -1      -1  2210    1979 Feb 10  1817
0     300   1200    200    17      -1     -1       9.6     2       9.3      -1      -1  2517         May 17  0551
0      30   12      2      17       8    546      14.2     2      11.3      -1      -1  2814         Aug 14  2049
0      10   200     140    70      10     20      12.9     2      10.0      -1      -1  2906         Sep  6  0906
0      10   240     170    70       6     20      17.5     3      11.1      -1      -1  2906         Sep  6  1139
0      40   26      10     38       5     -1      14.9     3      10.9      -1      -1  2906         Sep  6  1332
0      70   44      10     23       0     11      -1       2      10.0      -1      -1  2906         Sep  6  1850
0    2000   150     10     7        9   1330      16.7     2       9.4      -1      -1  3214         Dec 14  1553
0     400   290     20     7       -1      5      -1       0      10.1      -1      -1  3223         Dec 23
0     300   90      20     22       5    518      11.7     3       9.8      90      -1  5109    1980 Nov  9  1621
0    1300   370     50     14       5     91      12.0     2      10.4      63      -1  5109         Nov  9  2028
0     500   400     200    50      -1     24      12.6     2      10.0      61      -1  5110         Nov 10  0446
0     500   400     400    100     -1     -1      -1       2       8.4      -1      -1  5110         Nov 10  0744
0     100   30      10     33      -1     -1      -1       2      -1        -1      -1  5111         Nov 11  1515
0     600   160     160    100     -1     -1      -1       0       8.3      -1      -1  5115         Nov 15  0953
0      30   50      10     20       9   7830      30.4     3      12.5    3958     367  5216         Dec 16  1427
0     250   120     20     17      -1     -1      -1       3      11.5      70      -1  6915    1981 Sep 15
0     300   20      10     50      -1     -1      -1       2      10.2      -1      -1  7120         Nov 20  ambi
0     600   220     220    100     -1     -1      -1       3      -1        -1      -1  8310    1982 Mar 10  1205
0     300   70      20     28      -1     -1      -1       3      -1        -1      -1  8310         Mar 10  1845
0     700   24      4      17      15    403      15.1     3       9.8     187     460  8625         Jun 25  0528
0    1000   36      8      22       7   5120      24.6     3      11.0    1276     468  8625         Jun 25  1941
0     150   140     140    100     -1     -1      -1       0       8.1      -1      -1  8630         Jun 30
0     300   190     30     16      -1     -1      -1       2      -1        186     430  8813         Aug 13
0     900   80      10     12       7   4020      22.1     3      11.3     446     357  8813         Aug 13  2257
0     900   120     10     8       13   3700      18.5     2       9.8     610     494  8814         Aug 14  0237
0      -1   24      5      21       5   9060      31.8     3      12.1     535     371  8814         Aug 14  0507
;0     -1   3       3      100     -1    146      15.2     3      11.6      -1      -1  2814    1979 Aug 14  1728
;
;    3He    He     Ratio  He pct    Soft   Xray     Xray     III     Km        SMM    SMM    Mo     Date       Flare
;    Flux   Ratio  Error  Error     Dur.   Int       Temp    Int     Int       Peak   Indx   day                Time
```

post the defined record length & here ignored.

In ASCII format a verse can occupy at most one line

Blank Spacer Page

LOW-ENERGY COSMIC-RAY GROUP

PC DATA BASE

D. V. Reames
11/05/86

INTRODUCTION

For the past 3 years our group has been moving toward the use of (IBM-PC standard) personal computers as a primary tool for solving our data-acquisition, data-analysis and graphic-presentation needs. Such machines, based on the 8086/80286/80386/...-microprocessor architecture, can be used as on-board flight computers, lab computers, GSE computers, data analysis systems and graphic workstations. For the first time it will be possible to use a common set of software modules in all aspects of our business.

A set of graphic software is being written that will access data sets ranging from small hand-typed files to full satellite data bases to real-time pulse heights. This software must be fast and responsive providing displays and plots in seconds or minutes rather than weeks required with present mainframe-based systems. To provide adequate performance, all software for this system will be written in the C programming language with minimal use Assembly language for unique hardware interfacing or small high-speed routines.

New 80386-based systems will be available within a year that provide performance of 5x a PC/AT (2x a VAX 11/780 or 25% of an IBM 3081). Significant advances will also occur in the next few years in graphics, in Winchester and optical disks and in array processors. This

new high-performancee lower-cost technology will appear in the PC market years before it is seen on minicomputers or mainframes. We intend to take full advantage of this new technology.

DATA BASES

A general format has been defined for particle data bases (both flight and laboratory data). This format, based on the Lal 'encyclopedia' tape format on the 3081, is used for files containing either pulse height data or 'flux' data of fairly general form. Each file contains an ASCII text header that describes the file contents to both man and machine, i.e. the header can be examined with a standard text editor making the files self documenting. These data files are time-ordered sequential structures with variable-length fields; direct access to data at a specific time is provided via directory files described in a later section.

In the ASCII headers, defined fields (text or numeric) should be non-blank and each line should be terminated with CR LF (hex 0d 0a). Comments, preceeded by ';', may be included in the header. Software will ignore all characters after the ';' in any line and multiple lines of comments may be included at any location in the header (some restrictions apply).

Data in the body of the files is in binary form. Multi-byte integers are always stored with the least-significant byte (LSB) first. Floating-point quantities are stored in IEEE standard 754 form, also LSB first. Time is represented as a 4-byte signed integer containing the

number of seconds since 0000 UT on Jan 1, 1970. This form encodes time as a single variable (from 1902 thru 2038), permits easy differencing, and conforms to the UNIX standard base-year.

The logical structure of the Data files consists of the header followed by 'chapters' each containing one or more 'Verses'. Each chapter will normally contain data from a specific time interval (or accelerator run) and the chapter heading defines the begin time, end time, and a count of the verses it contains. Chapters are in time-sequential order but they need not be present during data gaps and the time periods they cover need not be the same length. Each verse within a chapter contains data from a different telescope, coincidence mode or gain, etc. The verse number, contained in the verse, is an index that refers to a verse descriptor in the header. The format and content of each fields within the verse are described by corresponding entries in the header. Time may be included in the first verse of a chapter, if that verse will always be present, otherwise it should occur in a verse by itself. (how would it then be related to other verse type data?)

Three explicit verse types presently defined are PHA, Flux and Rate verses. PHA verses are of variable length, thus they contain a length field after the verse number. The other two verse types have fixed lengths defined in their header. Flux verses contain rate and box count information; their fields must be combined to produce actual fluxes. Rate verses are a catch-all verse in which all fields are explicitly defined in the header.

Presently defined field types are as follows:

T	Time (long, 32 bit, integer) - sec. since 1/1/70.
T2	Begin and end time.
I	Integer (16 bit)- meaning depends upon context.
I2	Integer (16 bit) value and error.
B	Bit field (16 bit)- used for tag bits.
F	Single float value (32 bit).
F2	Float value and error (2*32 bit).
R2	Ratio (2*32 bit)- count and live time.
Sn	A byte string of n characters.
Cn	A command/status field of n bytes.

Avenues of probable expansion include I3 and F3 for values with asymmetric errors and F8 for 8 sectored-rate counts.

L Long 32 bit integer

E2 upper + lower value, floating point (as in energies) can plot an energy error bar

Table 1. PC Data File - Sample Header
(The three verse types would not normally be on the same file.)

```
Binary   created 1999 Jan 3 0005:22     ; or ASCII(?) or Directory
ISEE-3  15 min Mixed Data File          ; Sat. name(s) & file title

1978 Aug 15 00:00:00          ; Time of first Chapt. in file
1989 Sep  3 00:00:00          ; Time of last  Chapt. in file

#0 Rate
  T2                              Time

#1 PHA  ICH1ASTHI  HET 1   A-Stop   High Gain
  R2      1     1.0E5               Ast Rate
;Type  Min      Max    FS MeV    Name
  I       0     4096      188.     A1
  I       0     4096      188.     A2
  I       0     4096     3523.     C1+C2+C3
  B       0     ffff               Tags
;
#2 Flux ICVMev0  Mean VLET 1,2   High Z
  R2      1     1.0E5               VLET  Rate
  L       0     32767               Count on VLET matrix
;Type E Min   E Max    Geom.     Name
  I     2.2     2.8     .27       C12
  I     2.8     3.6     .27       C12
  I     3.6     4.4     .27       C12
  I     2.2     2.8     .27       O16
  I     2.8     3.6     .27       O16
  I     3.6     4.4     .27       O16
  I     2.2     2.8     .27       Fe56
  I     2.8     3.6     .27       Fe56
  I     3.6     4.4     .27       Fe56
;
#3 Rate I3
  R2      1     1.0E5               Ast  I3  HET 2
  R2      1     1.0E5               A1   I3  HET 2
  R2      1     1.0E5               Z1   I3  VLET 2
  R2      1     1.0E5               Z2   I3  VLET 2
  R2      1     1.0E5               Z3   I3  VLET 2
  F2      1     1.0E5               0.2  2.2 Electron I3 HET 2
; This verse type may contain any mix of defined field types
; such as T, I, B, F, R, S, and C.
#End
```

Table 2. PC Data File - Data Fields

Chapter:

Time Verse (Rate type):

Offset	Length	Description
0	2	Verse-Type No.
2	4	Chapter Begin Time
6	4	Chapter End Time

PHA Verse:

Offset	Length	Description
0	2	Verse-Type No.
2	4	No. bytes in rest of verse (=8*NPHA+8)
6	4	Rate Count (not trend checked)
10	4	Rate Live Time (")
14 ...{		
	2	PHA 1
	2	PHA 2
	2	PHA 3
		...
	2	Tag Bit Field
}		

Flux Verse:

Offset	Length	Description
0	2	Verse-Type No.
2	4	Rate Count (~~not trend checked~~)
6	4	Rate Live Time (")
10	4	No. PHA Events on the Matrix
14	2	No. counts Box 1
16	2	No. counts Box 2
18	2	No. counts Box 3
...	2	... (Each defined box)

Rate Verse:

Offset	Length	Description
0	2	Verse-Type No.
2	4	Rate Count
6	4	Rate Live Time
10	4	Rate Count
14	4	Rate Live Time
...		
42	4	Flux
46	4	Error

DIRECTORY FILES

Data files are unstructured serial files with variable length records. Random access into these files is provided by a separate Directory file. Directory files provide a two-level hierarchy of pointers to each chapter (time period) in the data base. That data base may consist of a single file or many files, each of which contains a different (sequential) block of time.

A Directory file contains a header, a directory table, and a chapter table. The header describes the data contained in each verse and is identical in form to the headers on the data files. The directory table has time-ordered entries in the following form:

1. Begin time.
2. End time.
3. Offset to first entry in chapter table.
4. Number of chapters.
5. Data file name (e.g. C:\dir\file.dat)

Entries in the chapter table have only two components:

1. Begin time of chapter.
2. Offset of chapter in data file.

A Directory file is produced by a directory program. This program accepts a sequential list of data files as input and traverses each file, building the two tables described above as it goes. Directory files will normally be kept on fast Winchester disk drives (or RAM disk), while the data files they describe may be on any drive available to the system.

An analysis program (such as a plot program) accesses a data base via the named Directory file. Initially it need only read the two time fields of each entry in the directory table. When seeking a specific time it would look in this time table, read the remainder of the desired directory field, read the block of chapter table entries defined and locate the desired time, open the named data file (if it differs), and finally, read the chapter from the data file.

Note that the directory scheme will allow time periods that were not included in the original data base to be 'logically' added later. Thus it provides for updating data that are kept on write-once optical disks.

Blank Spacer Page

THE LOW-ENERGY-PARTICLE GROUP
GRAPHIC ANALYSIS PACKAGE

3 basic commands

cross CRP1OFA.dat /p enables print to printer

alt-O to get options

esc return to plot screen

esc enter to replot if changes were made btwn plots

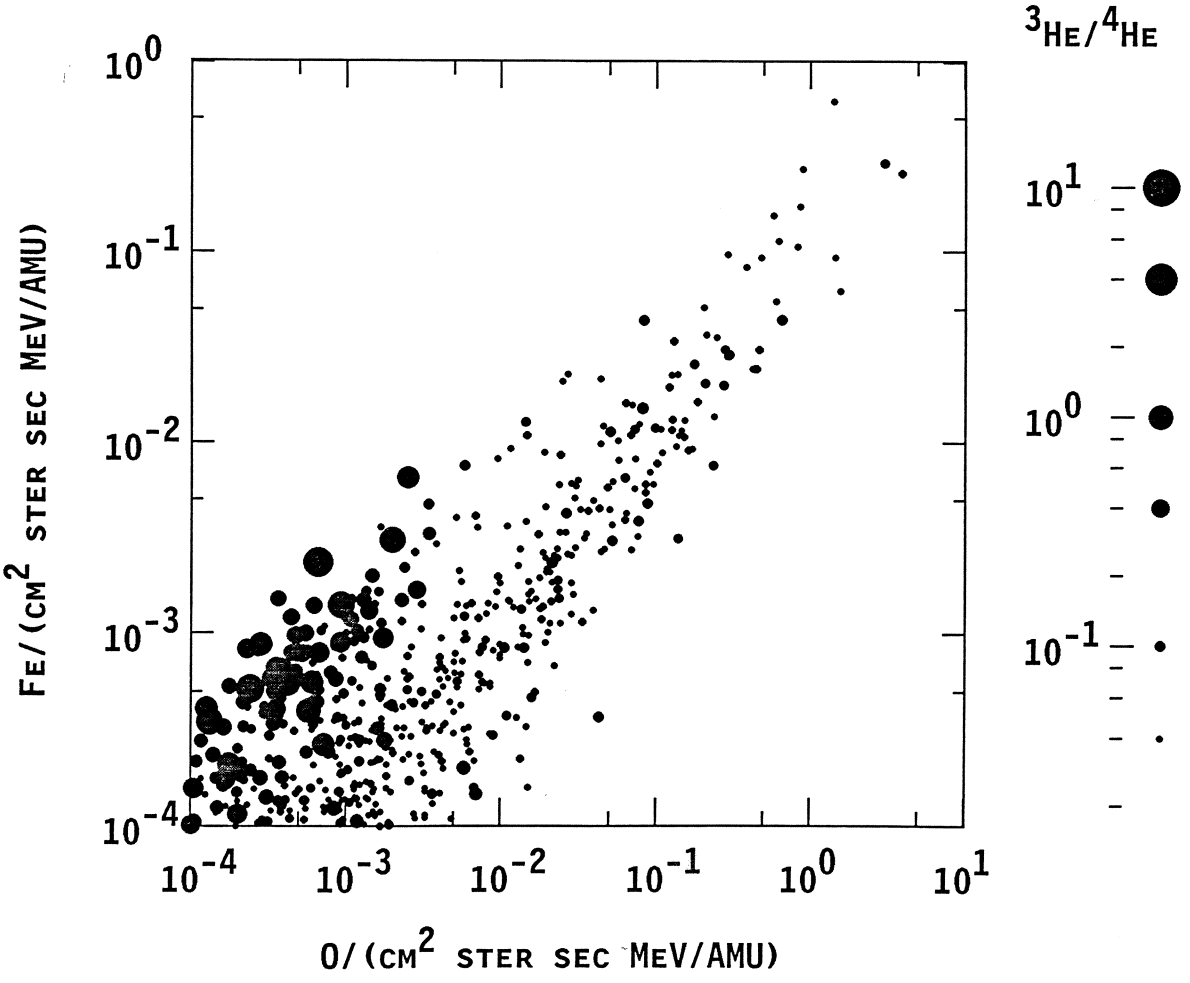
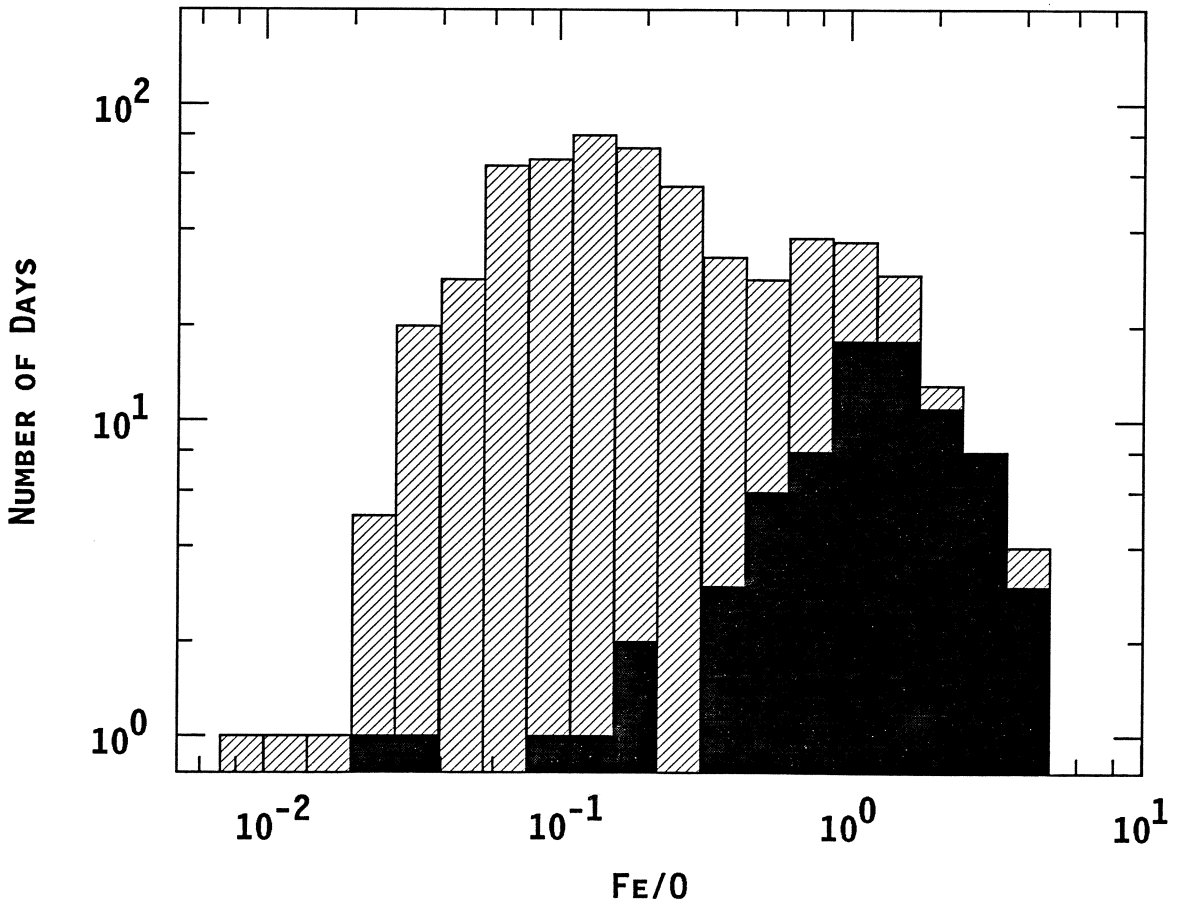


TABLE OF CONTENTS



The Low-Energy-Particle-Group Graphic Analysis Package

D. V. Reames
NASA/ Goddard Space Flight Center, Greenbelt, MD
4/14/88

I. INTRODUCTION

A set of graphic analysis programs has been written that provide a flexible means for studying multi-dimensional scientific data sets and for the preparation of publication-quality figures. A typical data set consists of a number of "events" for which several parameters have been measured, such as fluxes of several particle species as a function of time or a variety of particle, X-ray and radio parameters for many solar particle events. Individual programs allow the data to be selected and displayed in different ways while using a consistent set of menus and commands.

The programs run on an IBM-PC /XT or /AT (or equivalent) equipped with a Professional Graphics controller and display (640 x 480 pixels at 256 colors out of 4096). Black-and-white hard-copy plots may be made on the Canon LPB-8 A2 or equivalent laser printer that supports the ISO standard graphic command language. The programs are normally configured to use a numeric coprocessor (8087/80287/80387) and require about 128KB of RAM. A hard disk is recommended, but is not absolutely required. The programs run under the MS-DOS (or PC-DOS) operating system, version 2.0 or later. Typical systems currently in use by our group are 80386-based machines equipped with >60MB Winchester disk drives and 200MB/side (Optotech) optical disk drives as well as the display and printer.

The programs accept input data files in the LECR-group standard data format that is described herein. The files consist of an ASCII header that describes the format of data fields followed by the data itself that may be in either ASCII or binary form. The format is designed to support extremely large multi-spacecraft particle data bases that are stored on optical disk as well as small event files that can be typed in with a standard text editor.

Individual programs in the set are "Cross", "2d", "Stripe", "Matrix" and "Direct". Cross displays data in a 3-axis space consisting of x and y axes in the display plane and a z axis represented by color and/or the size or type of plotted symbol. The program 2d displays data in a 2- dimensional space but it allows the display of multiple functions of a single variable. Stripe is similar to 2d but displays in a non-overlapped strip-chart mode. Matrix is similar to Cross but is specifically designed for pulse-height data from particle telescopes. Finally, Direct is a utility program that makes directories of time-ordered data files for more efficient access and allows grouping of files covering different time periods into a single directory.

In any of the plotting programs, data may be selected subject to bounds on any of the parameters including those that are not plotted. Any plotted axis may be linear, logarithmic or time (for temporal data).

The programs also allow new data quantities to be generated that are algebraic combinations of the original ones, and these new quantities can be selected for plotting along any axis.

During the last year, plots produced by these programs have been included in about a dozen articles in Ap. J., Ap. J. (Letters), J.G.R. and several conference proceedings. The plots are easily converted to viewgraphs for oral presentations. The use of these programs has reduced our group's need for manual drafting support and has sharply reduced the time required for the preparation of scientific papers.

The ability to manipulate large quantities of data with relative ease is a necessity of modern science. We intend to continue to expand the power and convenience of these analysis programs.

II. OVERVIEW

This section describes the operation of the programs in sufficient detail that the reader can begin to use them on existing data files. Some familiarity with the MS-DOS operating system is assumed.

For simplicity, it is convenient to think of a data file as a table of numbers. Each row in the table corresponds to a particular event or time interval, each column gives the values of a particular parameter (e.g. 10 MeV proton flux, D1 pulse height, time, type II radio intensity) in each event. Any of the columns may be selected as a plot axis in the plot programs. Information in the file header describes the data format of each column, gives initial upper and lower bounds of the axis and provides a title.

Programs are invoked from the DOS command line as follows:

```
C>cross datafile.1 datafile.2 /p
```

Only the data files listed on the command line can be examined from within the programs; you must exit and restart the program to select new files. The programs are designed to accommodate either one or two files, however, provisions for a third file have been included recently with only minor inconveniences (the third file header is partly obscured in the menu display). If multiple files are chosen, they must all be directories or all be non-directory files. Using multiple directory files allows one to plot data from multiple spacecraft on the same plot, for example (for non-directory files, events in the each file must correspond one for one).

directories
multiple spacecraft
limitations of this are not clear here

The optional parameter /p on the command line will send plots directly to an on-line laser printer, otherwise they will go into the file plot.las that can be printed later. The option should only be used if the correct printer is actually on line.

In addition to plot.las, the programs open two other output files, point.lst and note.pad, that can be used for listing selected data points and noting cursor positions with comments. Since these files are automatically written on the default drive and subdirectory, **the programs should not be run with the optical disk selected (usually drive E:) since these files can not be erased.** Run the programs from C:.

The programs present the user with two primary data screens. The menu screen is in text mode and is used for selection and editing of the plot axes and labels and plot criteria. The plot screen is in high-resolution graphic mode and shows the actual data plot and allows additional axis scaling, cursoring, pan, zoom, color selection and printing. A third screen, the options screen, is accessible from the menu screen by pressing <Enter> on the first line or by pressing alt-O elsewhere; it allows selection and editing of several plot and print options. The <Esc> key toggles between the primary screens and exits from the options screen. From any screen, the program may be terminated (returned to DOS) using ^C (^C means the <Ctrl>-C combination).

To change an item on the menu screens, position the cursor to the left of the item and press <Enter>. This will select a new value, a new menu, or allow the line to be edited. <Enter> is also pressed when editing is complete or the new menu item is chosen. The most important keys to remember are:

Key	Screen	Function
<Enter>	Plot	Begin plotting.
	Menu	Select the current line for change ... enter the new values.
<Esc>	Plot	Stop plotting! Exit any secondary mode. Switch to menu screen.
	Menu	Exit any secondary menu (abandon changes). Exit edit mode (abandon changes). Switch to plot screen.
^C	anywhere	Exit the program (return to DOS).

On some machines, the <Enter> and <Esc> activities have been programmed onto the buttons of a (Microsoft) mouse so the the entire selection process can be controlled with the mouse.

In data files with many selectable axes, additional items may be brought into view for selection or editing by using the <Pg Up> and <Pg Dn> keys.

In Cross and Matrix, the last column, "Histogram", may be selected as the second or third axis. This will cause the points to be binned in 1- or 2-dimensional space, respectively. The bin width for a given axis is specified in parenthesis after the bounds on the editable axis description line.

An important function on the high-resolution plot screen is:

^P Copy the current plot to the printer (or to plot.las).

Descriptions of more advanced functions and tables of the complete list of control keys are found later in this document.

The banner line, printed when a program is invoked, shows a version number and date. The programs have evolved rapidly during the last two years and that evolution is likely to continue. For best results the date on this document should correspond with that on the banner line.

III. DATA FILES

Data files begin with an ASCII header that describes each of the data fields (columns) in the file. The actual data follows the header and may be either ASCII or binary as specified on the first line of the header. The columns are grouped into records called verses in order to group data of a similar kind i.e. data from a given particle telescope. For a given time period or event, not all verses need be present, thus, only the verses containing active data take up room in the data file. The group of verses that make up an event or time period is called a chapter.

The four types of verses presently defined are Rate, PHA, Flux, and Algebraic. Rate verses are an extremely general type (despite the cryptic name) since they can include a wide variety of fixed-length data formats. PHA and Flux verses are rather specific to particle data. Algebraic verses are defined in the file header but have no corresponding data in the file. The fields in the an Algebraic verse represent algebraic combinations of the other data fields.

Each field in a Rate verse is described by a format as listed in Table I. A field containing a begin and end time is format T2, a real (floating point) value and error has format F2, and a single integer is I (or I1) format, for example. The number and mixture of fields in a verse is arbitrary (within limits given below) but the fields in the data must correspond one-for-one with the descriptors in the header. Figure 1 shows an example of a small file with ASCII data. The description of each Rate verse in the header begins with "#n Rate". The verse number, n, appears again at the beginning of each verse of data. The line describing each field in the verse contains the format, the initial axis limits and the title.

PHA verses are designed for pulse-height data from multi-element particle telescopes. The verses are of variable length and each verse must begin with a byte count and a normalizing count rate. These are followed by the group of pulse height fields for each particle. The pulse height group is described only once in the header. The number of particles represented in a given verse is inferred from the byte count. A portion of a file header describing PHA verses is shown in Figure 2. Only the Matrix program can plot pulse-height data, however, the other programs can read the files and access the byte count and rate information.

Table I. Format Descriptors for Data Fields. All format types may be used in Rate verses; other verse types have a restricted set of formats. The sizes of the data fields refer to binary data only.

T	Time (long, 32 bit, integer) - sec. since 1/1/70.
T2	Begin and end time.
I	Single Integer (16 bit).
I2	Integer value and error (2*16 bit).
I3	Integer value, + error, - error (3*16 bit).
B	Bit field (16 bit)- used for tag bits.
E2	Float begin and end values (2*32 bit).
F	Single float value (32 bit).
F2	Float value and error (2*32 bit).
F3	Float value, + error, and - error (3*32 bit).
R2	Ratio- Float count and live time (2*32 bit).
Sn	A byte string of n characters (partially implemented).
Cn	A command/status field of n bytes (partially implemented).

Flux verses are an efficient form for storing large numbers of particle fluxes derived from a single pulse-height matrix. The normalizing rates and counts are stored at the beginning of the verse and the count of a given species and energy (box count) is stored as an integer. This allows each flux to be stored in a 16-bit integer instead of a 64-bit real value and error. The energy limits and geometry factor needed to normalize each flux are listed as fields in the line of the header describing that value. The programs calculate the flux and its error from these various input values.

Algebraic verses are an extension of the general algebraic capability in the programs that is described more fully elsewhere in this document. In the programs, each data field is assigned a letter-number combination where the letter represents the sequential verse (beginning with A) and the number is the field number in the verse (beginning with 0). Within a program it is easy to define a new data quantity that is "2.567*A5/B7-1.3*(C0-C1);", for example, that combines field descriptors and constants. These quantities may also be predefined in an Algebraic verse in the file header if one is willing to figure out the appropriate column numbers. Note that the expressions can only refer to previously-defined non-algebraic fields above it in the header. Note also that the letter assignments continue to increase in the second file read in.

Figure 3 shows a portion of the header for a file with Flux, Rate and Algebraic verses.

```
ASCII
 Ne/O and Fe/O vs Temp in 3He-Rich Events
;
1978 Aug 15 0000
1982 Aug 15 0000
;
#0 Rate
T1          Time
F1          9       17 X-ray Temperature (x10^6 K)
F2    5.00E-2       4.9  Ne/O
F2    0.99E-1       10.1 Fe/O
I3          5     5000 3He/4He (%)
#End
;
;    Time                Temp      Ne/O      error       Fe/O      error    He-ratio  +err     -err
;
0   1978 Nov  8 1751    14.4       1.4237    1.3321     2.4576    2.0766     36       10       10
0   1979 Aug 14 2049    11.3    4.1311E-1 2.8061E-1  4.1803E-1 2.8296E-1     12        2        2
0   1979 Sep  6 1336    13.6    7.2291E-1 4.3217E-1     2.7265    1.2124     26       10       10
0   1979 Dec 14 1553    14.7    3.6593E-1 7.5234E-2     1.5485 2.0800E-1    150       10       10
0   1980 Nov  9 2028      12    9.9448E-1 8.1228E-1  3.3149E-1 3.8277E-1    370      180       50
;  Comment lines, beginning with ';', may be used anywhere in the header or the ASCII data.
;  The entire contents of this page (including caption) will run as a data file.
0   1980 Nov 10 0446    15.7    1.6266E-1 1.7568E-1  8.1047E-1 4.9459E-1    400      300      200
0   1980 Dec 16 1452    10.7            0 1.4888E-1  5.5831E-1 3.5150E-1     50       10       10
0   1982 Jun 25 0531    12.2    5.5423E-1 2.1864E-1  9.3938E-1 3.1767E-1     24        4        4
0   1982 Jun 25 1944    10.7    5.5551E-1 1.4076E-1     1.3381 2.6600E-1     36        8        8
0   1982 Aug 13 2259    14.3    3.6697E-1 1.6230E-1     3.4251 8.9514E-1     80       10       10
0   1982 Aug 14 0242    13.9       1.1235 4.7002E-1     3.3748    1.1623    120       10       10
0   1982 Aug 14 0507    12.4    5.9420E-1 1.9279E-1     1.4909 3.7746E-1     24        5        5
;
;
;       Figure 1.  A example of a amall ASCII data file.  The first line must contain
;                  the data type.  The second line contains the file title.  Begin and
;                  end times are required only if the file contains time.  A single
;                  Rate verse is defined with 5 axes.  Note that each event in the
;                  data begins with the verse number.  Every field must contain a value.
;
;
;
```

```
 BINARY CREATED                    1988 MAR 04 14:00:42
 ISEE-3 PHA DATA FILE
82/ 6/25  0: 0: 0 ; TIME OF THE FIRST CHAPTER
82/ 6/27  0: 0: 0 ; TIME OF THE LAST CHAPTER
 0   0    0:15:00   ; AVER. INTERVAL Y D HH:MM:SS
#0 RATE VERSE
  T2                                TIME
#5 PHA   ICH1ASTHI  HET 1  A-STOP  HIGH GAIN
  L      0    200000            BYTE COUNT
  R2  1.0E-6  1.0E5              RATE  ICH1ASTHI
  I      0    4096     200      A1    ICH1ASTHI
  I      0    4096     197.1    A2    ICH1ASTHI
  I      0    4096    3533.     C123  ICH1ASTHI
  B      0    FFFF              TAGS  ICH1ASTHI
#6 PHA   ICH1ASTLO  HET 1  A-STOP  LOW GAIN
  L      0    200000            BYTE COUNT
  R2 1.0E-6   1.0E5              RATE  ICH1ASTLO
  I      0    4096     953.     A1    ICH1ASTLO
  I      0    4096     931.     A2    ICH1ASTLO
  I      0    4096   17833.     C123  ICH1ASTLO
  B      0    FFFF              TAGS  ICH1ASTLO
#7 PHA   ICH1BSTP   HET 1  B-STOP  PROTONS
  L      0    200000            BYTE COUNT
  R2 1.0E-6   1.0E5              RATE  ICH1BSTP
  I      0    4096     614.     B1    ICH1BSTP
  I      0    4096     671.     B2    ICH1BSTP
  I      0    4096    3646.     C432  ICH1BSTP
  B      0    FFFF              TAGS  ICH1BSTP
#8 PHA   ICH1BSTE   HET 1  B-STOP  ELECTRONS
  L      0    200000            BYTE COUNT
  R2 1.0E-6   1.0E5              RATE  ICH1BSTE
  I      0    4096     614.     B1    ICH1BSTE
  I      0    4096     671.     B2    ICH1BSTE
  I      0    4096    3646.     C432  ICH1BSTE
  B      0    FFFF              TAGS  ICH1BSTE
#9 PHA   ICH1BSTLO  HET 1  B-STOP  LOW GAIN
  L      0    200000            BYTE COUNT
  R2  1.0E-6  1.0E5              RATE  ICH1BSTLO
  I      0    4096    2245.     B1    ICH1BSTLO
  I      0    4096    4517.     B2    ICH1BSTLO
  I      0    4096   17984.     C432  ICH1BSTLO
  B      0    FFFF              TAGS  ICH1BSTLO
#10 PHA ICH1PENHI   HET 1  PEN     HIGH GAIN
  L      0    200000            BYTE COUNT
  R2  1.0E-6  1.0E5               RATE  ICH1PENHI
  I      0    4096     614.      B1    ICH1PENHI
  I      0    4096    1016.      C1    ICH1PENHI
  I      0    4096    3646.      C432  ICH1PENHI
  B      0    FFFF               TAGS  ICH1PENHI
```

Figure 2. A portion of the header for a data file containing PHA verses. Each particle measurement consists of three integer pulse heights and a tag field for the verses shown here. The third numeric field is the calibration in MeV full scale (not yet fully supported).

```
 BINARY CREATED  1987 NOV 20 13.17.53
ISEE-3   24 Hr  Composition Data (trend checked)
78/ 8/15   0: 0: 0   ; START TIME OF FILE
87/ 2/ 7   0: 0: 0   ; STOP  TIME OF FILE
0    1    0000
#0  RATE
  T2                                  TIME;
; Verses 1-4 omitted from figure
# 5   FLUX   19-VLET I, EVENT TYPE 0
  R2    1.0E-05    1.0E+05           MEAN VLET  ET 0    RATE
  L     0            32767           COUNT ON VLET Mean  ET 0    MATRIX
;TYPE   E MIN     E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  I       1.90      2.82     0.259   C12      ;   MD2
  I       3.96      7.08     0.287   C12      ;   MD3
  I       7.08     12.67     0.279   C12      ;   MD3
  I       1.90      2.81     0.287   N14      ;   MD2
  I       3.97      7.02     0.287   N14      ;   MD3
  I       7.02     12.60     0.287   N14      ;   MD3
  I       1.90      2.80     0.287   O/(cm2 ster sec MeV/AMU)
  I       3.94      7.01     0.287   O16      ;   MD3
  I       7.01     12.53     0.287   O16      ;   MD3
  I       1.90      2.80     0.287   NE20     ;   MD2
  I       4.00      7.16     0.287   NE20     ;   MD3
  I       7.16     12.58     0.287   NE20     ;   MD3
  I       1.90      2.80     0.287   MG24     ;   MD2
  I       3.97      7.10     0.287   MG24     ;   MD3
  I       7.10     12.60     0.287   MG24     ;   MD3
  I       1.90      2.80     0.285   SI28     ;   MD2
  I       3.98      7.01     0.287   SI28     ;   MD3
  I       7.01     12.53     0.287   SI28     ;   MD3
  I       1.90      2.81     0.275   S32      ;   MD2
  I       4.02      7.10     0.287   S32      ;   MD3
  I       7.10     12.53     0.287   S32      ;   MD3
  I       1.91      2.80     0.269   CA40     ;   MD2
  I       3.96      7.13     0.287   CA40     ;   MD3
  I       7.13     12.61     0.287   CA40     ;   MD3
  I       1.90      2.80     0.287   Fe/(cm2 ster sec MeV/AMU)
# 7   FLUX   22-VLET II, EVENT TYPE 1
  R2    1.0E-05    1.0E+05           VLET II ET 1    RATE
  L     0            32767           COUNT ON VLET II  ET 1    MATRIX
  I       1.10      1.34     0.252   HE4      ;   IID2        22
  I       1.34      1.63     0.272   HE4      ;   IID2        22
  I       1.35      1.65     0.187   HE3      ;   IID2        22
  I       1.98      2.82     0.214   HE4      ;   IID3        22
  I       2.29      3.24     0.221   HE4      ;   IID3        22
# 8   RATE
  R2    1.0E-05    1.0E+05           IILDI-(U)
  R2    1.0E-05    1.0E+05           IIL
  R2    1.0E-05    1.0E+05           IILZ2
  R2    1.0E-05    1.0E+05           IILZ3
#9 Alg
F2       .01        20              G4/G3-.05; 1.3 3He/4He
F2       .005       10.1            F26/F8; 2- 3  Fe/O
F2       .99        1010            H1/H2-1.; >1.1  H/He
F2       .02       500              D2/D4;  p/e
#END
```

Figure 3. An abridged version of the header for a file containing Flux, Rate and Algebraic verses.

Verse numbers must increase monotonically from verse to verse throughout a chapter, but they do not have to be sequential. In processing each chapter (in non-directoried files), the programs will try to load each verse described in the header until an equal or smaller verse number is found. Subsequent data will be treated as beginning the next chapter.

Files containing ASCII data have some limits that do not apply to to those with binary data. An ASCII verse is considered to be a single line terminated by CR, LF. This line length is presently limited to 200 characters (multiple verses may be used to to overcome this limitation). Files containing PHA verses can not now be ASCII. Note that ASCII files are completely ASCII, they can not be partially binary. Data fields in ASCII files may be entered in free format separated by one or more blanks with no particular column alignment required, however, a consequence of this is that a data field may not be left blank. Missing data may be identified by using a negative error field or by omitting the entire verse containing the data.

Data files do not have to be time ordered and, in fact, do not have to contain time. When included on a file, however, time plays a special role and it is treated somewhat differently from other variables. In binary form, it must be encoded as a 32-bit integer representing the number of seconds since 1970 Jan 1 0000:00 (valid from 1902 through 2038). ASCII times may be entered in any of the 3 standard formats (1978 aug 15 1500, 82/12/21 0101, 1987 102 1200) interchangeably, and some latitude is allowed such as day of month > 31 or hours > 24. The axis boundaries for time are taken from the file begin and end times at the top of the header; they are omitted from the data field definition.

In binary files integers in the data are stored with least-significant byte (LSB) first. Real numbers are in IEEE 754 standard form. These are the standard IBM-PC & 8087 coprocessor conventions.

In the header and in ASCII data, lines beginning with a ';' will be treated as comments and ignored.

The programs are currently limited to a total of 150 columns (axes), 24 verses and 2 files (or directories).

Is this the sum of all opened files constraint on database creations?

IV. DIRECTORIES

When the programs access normal data files, they do not assume any particular ordering of the chapters in the file, therefore, the entire file must be read sequentially each time a plot is produced. This is tolerable for small files and it is necessary for general-purpose plotting of unordered data, however, it becomes woefully unacceptable when plotting a 10-year spacecraft data base. In order to cope with these large time-ordered data bases and to take advantage of the random-access optical disks on which they can be stored, support for a data-directory system has been included in the software.

A directory file, containing pointers to the beginning of each chapter in one or more time-ordered data files, can be created by the Direct program. Direct takes its input from a small "make" file that contains the name of the output directory and a sequential list of input data files and the time interval to be included from each. Data

files in the list should be identical, except for the time intervals that they contain. The first field in each chapter of the data files *must* be the time.

When a directory is accessed by a plotting program, only the header on the first file is loaded. Furthermore, Direct allows its first input data file to have the same begin and end time. This allows the first file to be an editable header-only file so that axis names and limits can be changed and algebraic verses can be added even though the rest of the data files are stored in permanent form on a write-once optical disk. An example of a directory make file with such a header is shown in Figure 4. To create this directory, Direct is invoked with the name of this file as a command-line argument. By convention, make-directory files have the extent .md and the resulting directories are given the extent .dir but the programs make no assumption about naming.

```
ic_30m.dir
;
c:\data\ic_30m.hdr   78 Aug 15 0000   78 Aug 15 0000
e:\isee\30m_a.flx    78 Aug 15 0000   82 jan  1 0000
e:\isee\30m_b.flx    82 jan  1 0000   85 jan  1 0000
e:\isee\30m_c.flx    85 jan  1 0000   87 feb  7 0000
```

Figure 4. Example of a make-directory file for input to Direct. A header and three data files are combined to form the directory ic_30m.dir.

Note that the directory facility allows new data to be added and blocks of old data to be replaced simply by referencing the files and time periods in the desired order and creating a new directory. There is no separate limit to the number of times a given data file may be referenced in a .md file. A total of 20 separate time blocks are presently allowed in a directory so that the directory can access data from as many as 20 different files.

Directories may be made for ASCII files as well as binary files (not a mixture in a single directory). This is convenient for making a directory of a file containing a list of event times. Running a plot program such as 2d with this directory and a main data-base directory allows the user to step from event to event while examining any features in the data base.

V. THE MENU AND OPTIONS SCREENS

When any of the plot programs is invoked, the menu screen comes into view. Quantities on the menu screen are initially derived mainly from the file header. The quantities on a given line may be altered by positioning the cursor on the line (using <up-arrow> and <down-arrow> keys) and pressing the <Enter> key. This "selected" quantity will either 1) toggle to the next value, 2) display a new list and cursor for further selection or 3) highlight the line or quantity with red or blue background for alteration with the editor. Quantities on the options screen are altered in the same way.

A. Axis Selection

Typical menu screens are shown in Figures 5 through 7. The top of Figure 5 shows a menu screen from Cross with the cursor (not shown) in the leftmost column. In the screen at the bottom of Figure 5, the y-axis parameter has been selected and the cursor is on the sub-menu to the right; it may be moved up and down and a new axis may be selected (<Enter>) or the old axis retained (<Esc>).

If there are too many possible axes to show on the screen at once, others may be seen using <Pg up> and <Pg dn>. The list is moved 10 lines at a time. Both the edit list below and the axis sub-menu begin with the same variable and move together.

The linear/log fields may be toggled with <Enter>. These fields should be selected after the axis is selected since they are "sticky", i.e. this property stays with each axis as the axis is selected and deselected. If the time axis is changed to linear or log, it will be displayed in sec since the selected begin time (1978 Aug 15 0000 in the example in Figure 5). This is useful in showing time since an event onset, etc.

Menu screens for 2d and Matrix are shown Figures 6 and 7, respectively. The format for 2d allows multiple y values to be selected; limits for the first y-value in this list will be used for the axis bounds. For Matrix, a verse should be selected before choosing axes since the choice of axes changes with the verse selection.

In Cross and Matrix, the y or z axes may be selected as histogram. Selecting the y axis as histogram produces a one-dimensional histogram in which the events are binned based upon their x-value. The bin width is derived from the number in parenthesis on the edit line for that axis. For linear axes the bin width has the same data type and units as the axis; for log axes the bin widths are multiplying factors for floating-point variables and units of 256 $\log_2(dx/x)$ for integers (sorry about that!). Further information on histograms is given on page 24.

```
Cross plot 3.2  (all)                                              3-22-88
oxy.lst              ISEE-3  Daily Avg. O >= 1.e-4                 02:41:07 PM

x (log) :
    1.90  2.80 O/(cm2 ster sec MeV/AMU)
y (log) :
    1.90  2.80 Fe/(cm2 ster sec MeV/AMU)
z (log) :
    1.3 3He/4He
A0  78 Aug 15 0000  87 Feb  7 0000 (86400)|TIME                        T2
A1  -1.00E-4     1000 (    1)|1.90  2.82 C12                          F2
A2   1.00E-4    10.10 (    1)|1.90  2.80 O/(cm2 ster sec MeV/AMU)      F2
A3  -1.00E-4     1000 (    1)|1.90  2.80 NE20                         F2
A4  -1.00E-4     1000 (    1)|1.90  2.80 MG24                         F2
A5  -1.00E-4     1000 (    1)|1.90  2.80 SI28                         F2
A6   1.00E-4     1.01 (    1)|1.90  2.80 Fe/(cm2 ster sec MeV/AMU)    F2
A7   1.00E-1       20 (    1)|1.3 3He/4He                             F2
B0  -5.00E-2    10.10 (    1)|A1/A2;   C/O                            F2
B1  -2.50E-2     5.05 (    1)|A3/A2;   Ne/O                           F2
B2  -2.50E-2     5.05 (    1)|A4/A2;   Mg/O                           F2
B3  -1.50E-2     3.03 (    1)|A5/A2;   Si/O                           F2
B4  -5.00E-3    10.10 (    1)|A6/A2;   Fe/O                           F2
Z0         0      100 (    0)|Histogram                               I1
```

```
Cross plot 3.2  (all)                                              8
oxy.lst              ISEE-3  Daily Avg                              02:45:41 PM

x (log) :
    1.90  2.80 O/(cm2 ster sec MeV/A
y (log) :
--> 1.90  2.80 Fe/(cm2 ster sec MeV/
z (log) :
    1.3 3He/4He
A0  78 Aug 15 0000  87 Feb  7 0000 (86
A1  -1.00E-4     1000 (    1)|1.90  2.
A2   1.00E-4    10.10 (    1)|1.90  2.
A3  -1.00E-4     1000 (    1)|1.90  2.
A4  -1.00E-4     1000 (    1)|1.90  2.
A5  -1.00E-4     1000 (    1)|1.90  2.80 SI28                         F2
A6   1.00E-4     1.01 (    1)|1.90  2.80 Fe/(cm2 ster sec MeV/AMU)    F2
A7   1.00E-1       20 (    1)|1.3 3He/4He                             F2
B0  -5.00E-2    10.10 (    1)|A1/A2;   C/O                            F2
B1  -2.50E-2     5.05 (    1)|A3/A2;   Ne/O                           F2
B2  -2.50E-2     5.05 (    1)|A4/A2;   Mg/O                           F2
B3  -1.50E-2     3.03 (    1)|A5/A2;   Si/O                           F2
B4  -5.00E-3    10.10 (    1)|A6/A2;   Fe/O                           F2
Z0         0      100 (    0)|Histogram                               I1

    A0 : TIME
    A1 : 1.90  2.82 C12
    A2 : 1.90  2.80 O/(cm2 ster sec MeV
    A3 : 1.90  2.80 NE20
    A4 : 1.90  2.80 MG24
    A5 : 1.90  2.80 SI28
    A6 : 1.90  2.80 Fe/(cm2 ster sec Me
    A7 : 1.3 3He/4He
    B0 : A1/A2;   C/O
    B1 : A3/A2;   Ne/O
    B2 : A4/A2;   Mg/O
    B3 : A5/A2;   Si/O
    B4 : A6/A2;   Fe/O
    Z0 : Histogram
```

Figure 5. A typical menu screen for cross is shown at the top of the figure. Below, the y-axis has been selected so that the axis-selection sub-menu appears at the right.

```
2-D  V. 3.2  (xy)                                                    3-22-88
e:ic_15m.dir        ISEE-3    15 MIN  SUM. DATA FILE               02:53:55 PM

x (time) :
    Time
y (log) :
     0.22  2.00 Elec   ICIIA2
    30.24 45.20 Proton ICIB3
    ---------
    ---------
    ---------
A0  78 Aug 15 0000  87 Feb  7 0000 (  900)|Time                        T2
B0   1.00E-5   100000 (    1)|HET I BSTP,HI RATE                      R2
B1         0  1000000 (    1)|COUNT ON HET I BSTP,HI MATRIX           L1
B2   1.00E-6     1000 (    1)|22.14 27.17 Proton ICIB2                fI1
B3   1.00E-6     1000 (    1)|30.24 45.20 Proton ICIB3                fI1
B4   1.00E-6     1000 (    1)|45.20 57.39 Proton ICIB3                fI1
C0   1.00E-5   100000 (    1)|HET I BSTP,LO RATE                      R2
C1         0  1000000 (    1)|COUNT ON HET I BSTP,LO MATRIX           L1
C2   1.00E-6     1000 (    1)|30.01 45.09 He4    ICIL3               fI1
C3   1.00E-6     1000 (    1)|45.09 57.15 He4    ICIL3               fI1
D0   1.00E-5   100000 (    1)|HET I PEN,HI RATE                       R2
D1         0  1000000 (    1)|COUNT ON HET I PEN,HI MATRIX            L1
D2   1.00E-6     1000 (    1)|132.30 241.23 Proton ICIPH              fI1
E0   1.00E-5   100000 (    1)|HET II AST,HI RATE                      R2
```

```
2-D  V. 3.2  (xy)                                  A0 : Time                                 8
e:ic_15m.dir        ISEE-3    15 MIN               B0 : HET I BSTP,HI RATE      02:56:48 PM
                                                   B1 : COUNT ON HET I BSTP,HI MATRIX
x (time) :                                         B2 : 22.14 27.17 Proton ICIB2
    Time                                           B3 : 30.24 45.20 Proton ICIB3
y (log) :                                          B4 : 45.20 57.39 Proton ICIB3
     0.22  2.00 Elec   ICIIA2                      C0 : HET I BSTP,LO RATE
    30.24 45.20 Proton ICIB3                       C1 : COUNT ON HET I BSTP,LO MATRIX
-->  ---------                                     C2 : 30.01 45.09 He4    ICIL3
    ---------                                      C3 : 45.09 57.15 He4    ICIL3
    ---------                                      D0 : HET I PEN,HI RATE
A0  78 Aug 15 0000  87 Feb  7 0000 (               D1 : COUNT ON HET I PEN,HI MATRIX
B0   1.00E-5   100000 (    1)|HET I BS             D2 : 132.30 241.23 Proton ICIPH
B1         0  1000000 (    1)|COUNT ON             E0 : HET II AST,HI RATE
B2   1.00E-6     1000 (    1)|22.14 27              E1 : COUNT ON HET II AST,HI MATRIX
B3   1.00E-6     1000 (    1)|30.24 45              E2 :  0.22  2.00 Elec   ICIIA2
B4   1.00E-6     1000 (    1)|45.20 57              E3 :  4.45  6.41 Proton ICIIA2
C0   1.00E-5   100000 (    1)|HET I BS              E4 :  7.08 12.83 Proton ICIIA3
C1         0  1000000 (    1)|COUNT ON              E5 : 12.83 22.66 Proton ICIIA3
C2   1.00E-6     1000 (    1)|30.01 45              E6 :  4.40  6.42 He4    ICIIA2
C3   1.00E-6     1000 (    1)|45.09 57              E7 :  7.08 12.64 He4    ICIIA3
D0   1.00E-5   100000 (    1)|HET I PE              E8 : 12.64 22.08 He4    ICIIA3
D1         0  1000000 (    1)|COUNT ON              F0 : VLET II ET 0  RATE
D2   1.00E-6     1000 (    1)|132.30 2              F1 : COUNT ON VLET II ET 0 MATRIX
E0   1.00E-5   100000 (    1)|HET II A              F2 :  2.09  2.81 C12    ICIID2
```

Figure 6. A typical menu screen for 2d is shown at the top of the figure. Below, the third y-axis has been selected so that the axis-selection sub-menu appears at the right.

```
Matrix  V. 3.2  (all)                                              3-22-88
82jun25.pha          ISEE-3 PHA DATA FILE                        03:15:49 PM
  PHA verse B: ICH1ASTHI  HET 1  A-STOP  HIGH GAIN
x (linear) :
    A2    ICH1ASTHI
y (linear) :
    A1    ICH1ASTHI
z (linear) :
    C123  ICH1ASTHI
A0  82 Jun 25 0000  82 Jun 27 0000 (  900)|TIME                    T2
B2         0     4096 (    1)|A1    ICH1ASTHI                    I1
B3         0     4096 (    1)|A2    ICH1ASTHI                    I1
B4         0     4096 (    1)|C123  ICH1ASTHI                    I1
B5         0     FFFF  M=FFFF|TAGS  ICH1ASTHI                    B1
Z0         0      100 (    0)|Histogram                          I1
```

```
Matrix  V. 3.2  (all)                                  A: VERSE
82jun25.pha          ISEE-3 PHA DATA F                 B: ICH1ASTHI  HET 1  A-STOP  03:17:02 PM
-->PHA verse S: ICV2HE      VLET 2  EVE                C: ICH1ASTLO  HET 1  A-STOP  LOW GAIN
x (linear) :                                           D: ICH1BSTP   HET 1  B-STOP  PROTONS
    D2    ICV2HE                                       E: ICH1BSTE   HET 1  B-STOP  ELECTRONS
y (linear) :                                           F: ICH1BSTLO  HET 1  B-STOP  LOW GAIN
    D1    ICV2HE                                       G: ICH1PENHI   HET 1  PEN     HIGH GAI
z (linear) :                                           H: ICH1PENLO   HET 1  PEN     LOW GAIN
    E     ICV2HE                                       I: ICH2ASTHI   HET 2  A-STOP  HIGH GAI
A0  82 Jun 25 0000  82 Jun 27 0000 (                   J: ICH2ASTLO   HET 2  A-STOP  LOW GAIN
S2         0     2048 (    1)|D1    IC                K: ICH2BSTP    HET 2  B-STOP  PROTONS
S3         0     2048 (    1)|D2    IC                L: ICH2BSTE    HET 2  B-STOP  ELECTRON
S4         0     1024 (    1)|E     IC                M: ICH2BSTLO   HET 2  B-STOP  LOW GAIN
S5         0     FFFF  M=FFFF|TAGS  IC                N: ICH2PENHI   HET 2  PEN     HIGH GAI
Z0         0      100 (    0)|Histogra                O: ICH2PENLO   HET 2  PEN     LOW GAIN
                                                       P: ICV1HIZ     VLET 1  EVENT TYPE 0
                                                       Q: ICV1HE      VLET 1  EVENT TYPE 1
                                                       R: ICV2HIZ     VLET 2  EVENT TYPE 0
                                                       S: ICV2HE      VLET 2  EVENT TYPE 1
```

Figure 7. A menu screen from the Matrix program is shown at the top. The bottom panel shows a similar screen with the verse sub-menu selcted. The verse should be selected before selecting individual axes in a verse.

B. Editing Axis Data

Lines of axis data containing limits, bin widths and labels appear at the bottom of the menu screens. Lines begin with a letter that changes when the verse or the file changes, and a number that sequences within a verse. This composite axis number is used to identify axes in algebraic expressions. This number is followed by the lower and upper bounds, the bin width, the axis label and the data-type descriptor.

It is possible to edit axis lines, to delete them, and to insert new lines. To delete a line, move the cursor to the line (not in edit mode) and press <^Backspace>. The line will disappear for the remainder of the session and will not be involved in any bounds checking. To add a new axis see the section on algebraic axes.

To edit data on an axis line, position to the beginning of the line and press <Enter>. When editing is complete pressing <Enter> will cause the new data to be read; <Esc> will select the old data. A recognized value error on a new line will cause the PC to beep once.

Within edit mode the arrow, tab, home and end keys position the cursor, <Ins> toggles insert (blue)/ replace (red), <Del> deletes the cursored character, <backspace> deletes to the left, and the alphanumeric keys type normally. It is also possible to enter some printer control characters in the text fields using the standard keyboard feature that allows entering their decimal equivalents. Two characters of special interest are partial-line-down (alt-139) and partial-line-up (alt-140) that allow embedded subscripts and superscripts.

In edit mode, the vertical arrows have the effect of accepting the new data on the current line then moving to the current cursor location on the next line if the target line is editable. If the target line is not editable, the effect is the same as <Enter>.

Fields containing time may be entered in a variety of formats. Dates may be entered in year/month/day formats like 1978 Apr 10 or 78/04/10. The 3-character month may be any mixture of upper and lower case. The year- day of year format may also be used, e.g. 1978 100. The 4-digit years may be replaced by their last 2 digits. UT time may be written as a 4-digit value with or without a colon.

C. Algebraic Axes

New axes involving combinations of the original ones may be entered by typing ^<Enter>. A new edit line will be shown at the end of the axis list (ahead of the histogram line in Cross and Matrix). The first line entered will define a new algebraic verse. An expression is typed in the field that usually contains the axis label. The expression may contain constants (e.g. 5 or 3.14159 or 6.23e-4), variables (e.g. A5 or C15) that refer to previously-defined non-algebraic fields and the operators +, -, *, /, unary -, and ( ) at any nesting level. The expression must end with ; and may be followed by a text label if space permits.

If the expression involves only integers (I1), the result will be integral, otherwise it will be float (F1 or F2). Expressions resulting

in errors or overflow will not pass bounds tests; those with missing values will.

In algebraic expressions involving quantities with errors (e.g. F2 format), the errors are propagated to the result using the usual rules for derivatives. Asymmetric errors (F3) are converted to a symmetric form for propagation.

D. The Options Screen

The creation of a plot from a data set can involve a large number of selection and formatting parameters. The values of some of these parameters are accessible on the options screen that is available for each of the plotting programs.

Typical options menus for Cross and 2d are shown with default settings in Figure 8. An options menu is obtained from the menu screen by pressing <Enter> with the cursor on the top line or by pressing alt-O on any other line. Fields in the options screen are selected or edited using <Enter> in a manner similar to those on the menu screen.

Referring to Figure 8, we consider each line of the options menu in order in the following sections.

1. Axis Bounds.

Axis bounds check specifies whether all axes or just the x, y, and z axes are required to be within bounds for the point to be included. Fields with data absent for a given event are not considered to be out of bounds unless these fields are selected for plotting. Missing data means that either the verse containing the data is absent or the quantity has a negative error value.

2. Cross Correlation

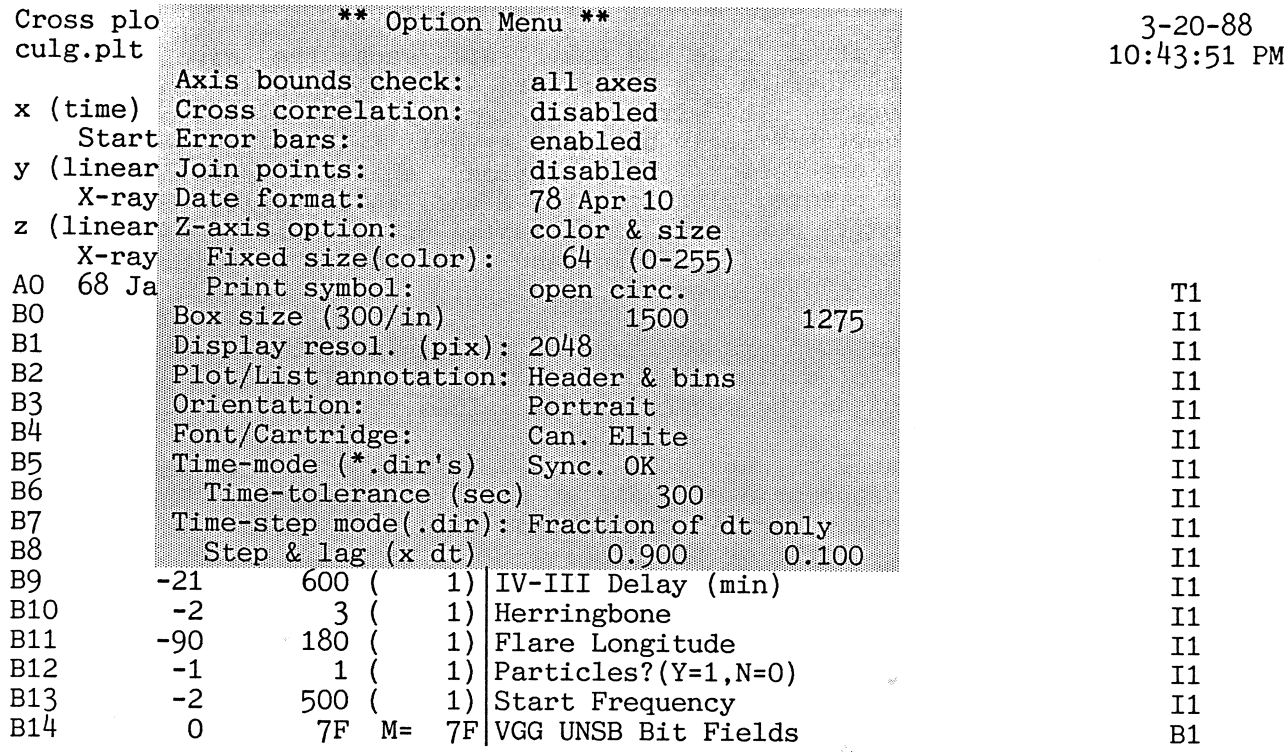
An x-y cross-correlation coefficient (unweighted) may be calculated for direct (non-histogram) plots. If this calculation is enabled, the coefficient will be shown on the plot.

3. Formatting, symbols, colors and labels.

Plotting of error bars may be enabled or disabled for points with defined errors. Successive points may be joined with a fine line in the order they are encountered if desired. Out-of-bounds points are treated as if they do not exist in the join sequence. At present the point-joining algorithm does not consider the the averaging interval on time-ordered .dir files as it obviously should.

The date format output by the program may be selected. The format for date input to the program is always optional.

The Z-axis option or curve option allows a choice of the color and size of points on the display screen and the symbol and size of the points on the printed plot. With the exception of one-dimensional histograms in Cross or Matrix, the symbol on the PGA screen is always an open circle. The printed version of a plot may contain any of 9 pre-defined symbols, however, in lieu of color.



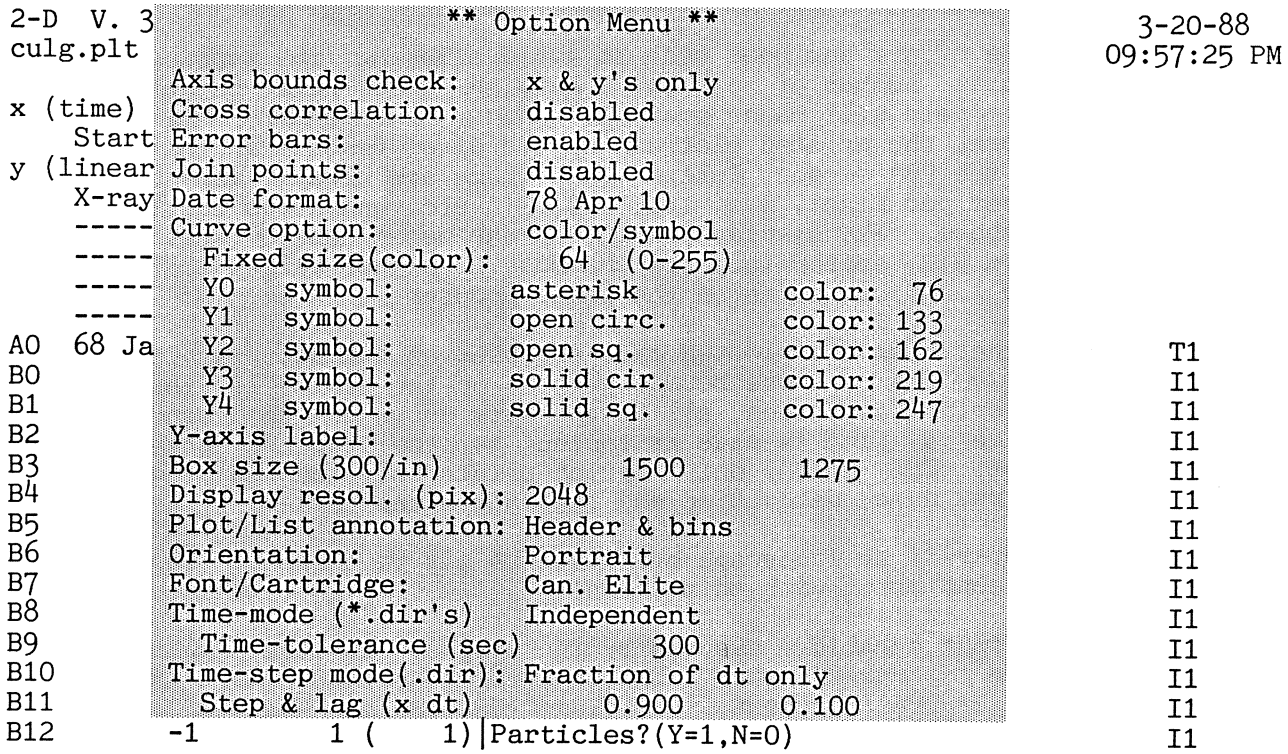


Figure 8. The options screens selected for Cross (top) and for 2d (bottom). The option screen is actually a submenu of the main menu screen that is selected from the first line or by using alt-0.

In Cross the Z-axis option "size and color" will cause the circle size and color to vary together on the PGA screen and displayed in the z-axis scale if it is selected. The size of the corresponding symbol on the printed plot will vary similarly but the symbol itself can be selected separately. In the "color/symbol" option, the PGA display remains the same, but the printed plot maps the z-axis to a series of symbols of increasing density but fixed (selectable) size. The "none" option allows overrides the z-axis variation and plots the all points with the same symbol of the chosen size and type.

In 2d, the curve number plays the role of a z-axis value. The mapping of each curve to a given symbol, color and size can be independently selected among a the define set of values.

A separate y-axis label may be typed in for 2d since none of the individual labels may be appropriate to all of the plotted curves. The labels of the individual curves are shown together with the corresponding symbol to the right of the plot.

The absolute size of the printed x-y box (excluding labels) may be set in each of the programs. Restraint should be used in shrinking the plots since character sizes do not scale. If the box is made too large, wrapping and other undesirable phenomena may occur. Note that the maximum reasonable size of each axis will depend upon the orientation of the plot on the page and the presence or absence of a bin list (see below).

A considerable effort has been made to avoid aliasing in mapping data onto a discrete number of pixels. Mapping 1000 pulse-height channels onto 900 pixels will cause every 10^{th} value to disappear, for example. Furthermore, the printed plot can have a pixel resolution that is 4 times that of the PGA screen. Integer data items are always mapped at an integral number per pixel, i.e. 1:1, 2:1, 3:1 etc. and never 1.1:1 as in the example above. The display resolution parameter can be varied in factors of 2 from 512, appropriate for the PGA, to higher values that are appropriate for large box sizes on the laser printer.

When axes other than the plotted ones are used as bounds, it is useful to have the values of those bounds printed on the plot. If the "Header and bins" option is chosen, all of the data shown on the menu screen at the time a plot is made is copied to the top of page containing the plot. The "Header only" option prints only the top few lines with the program name, date, time and file names and titles.

Either the portrait or landscape orientations may be chosen. The box size should be consistent with this choice.

The default font cartridge assumed by the programs is the Canon Elite 12 cartridge A. For finished plots or viewgraphs the 16 or 18 point fonts on the HP Presentations 1 cartridge (92286R) may be used. The plot on the cover of this document is produced with Presentation Bold 16, other plots are labeled in Elite 12 bold (Pres. Bold 18 is overkill). The programs cannot tell which cartridge is in place. The correct cartridge must be in place *before* running a program in /p mode and it can not be changed during a single run.

4. Time Synchronism and stepping.

When two directory files are examined together, they may be joined (in the data-base sense) along the time axis. This allows the programs to plot the 10 MeV proton intensity on IMP against the corresponding intensity on ISEE "at the same time". The treatment of this synchronism is controlled by the time-mode fields in the options menu. An independent time mode is appropriate for plotting as a function of time, so that 15-min and 2-hr averaged data can be displayed correctly on the same plot, for example. The "Sync. OK" setting is appropriate for joining the files as in the ISEE *vs.* IMP example, and the time tolerance may be set as desired.

When multiple directories are accessed, the primary time axis is always that on the first directory file. The first time axis should be selected when time is displayed. The bounds on the this axis then determine the limits of the plot and it is these bounds that must be set to plot a 2-day interval out of a 10-year data base, for example. Time bounds on the time axis associated with the second directory file should be set to allow access to the entire file.

The time step fields on the options menu control the behavior of the alt-T time-step command that is available when the high-resolution graphic screen is displayed. The step and lag parameters are expressed as a fraction of the time interval defined by the bounds on the primary time axis. The width of this interval is not changed in a time step but its position is displaced in time. For the example shown in Figure 8, a step of 0.9 and a lag of 0.1 will cause the time that occurs at the 90% position in the old interval to be moved to the 10% position in the newly displayed interval. The time step parameters may be set as desired; negative values are also permitted.

Other modes involve the actual times on the data files in the stepping process. If the mode labeled "Fract. + next t on file 0" is chosen with the previous values of step and lag, the time at 90% of the interval, t_{90}, will be found, the first file (file 0) will be advanced to the next time after t_{90}, and this new time will appear at the 10% position of the new interval. It is this mode that allows a data base to be easily scanned as file 1 with an event list as file 0. Each event time can be located at a convenient place in the field of view while stepping from event to event. Note that both files must be directories.

The final mode will advance to the next time occurring on either directory file in the stepping process as above.

It is not necessary for time to be selected as a display axis for stepping to occur.

E. Control Keys in the Menu/Options Mode.

A complete key listing for the menu and options modes is shown in table II.

Table II -Key Definitions in Menu/Options Mode

Escape	Exit the current mode
F1	" " " " (obsolete)
^C	Exit to DOS
up/down arrows	Move cursor one row
Pg up/Pg down	Move axis list up/down 10 lines.
^Backspace	Delete the axis descriptor on the current line if it is not currently selected as x, y, or z. (Menu screen only)
Enter	Select or change current row (see below)

Axis select: (Menu screen only)

Escape	Return to menu mode -no changes
F1	" " "
up/down arrows	Move cursor one row
Pg up/Pg down	Move list up/down 10 lines.
Enter	Select current row for menu
^C	Exit to DOS.

Edit parameters:

Escape	Exit edit field -no changes
Insert	Toggle insert(blue)/replace(red).
left/rt arrows	Move cursor one column
up/down arrows	Accept changes on current line and move to next editable line if any.
Tab left/rt	Move to next tab stop (8 col).
Home	Move to left end of line
End	Move to right end of line
Del	Delete char under cursor
Backspace	Delete char to left of cursor
ASCII char	Insert/repl. the char. (>=20H)
Enter	Accept current line, ret. to menu.
^C	Exit the program

alt-O Display options screen:.
Parameters in the options screen may be selected or edited just as those in the menu screen. Escape returns to menu mode.

VI. THE GRAPHICS SCREEN

The selected plot is shown on the graphics screen. When axes or scaling has been changed, only the axis box will appear. This allows the scaling to be checked without taking the time to plot the points. When all changes are complete, press <Enter> to cause points to be plotted. <Esc> toggles between the menu and graphic screens.

Point plotting may be terminated immediately by pressing <Esc>. This is useful when mistakes are made that could result in a very long plotting time (e.g. 10 years of a 15-min data base).

The z-axis scale may be displayed to the right of the plot using ^Z. The presentation toggles from color bar, to circle size and color, to off using this key.

A box cursor or tracking cross may be selected with ^X that toggles between the small cursor, the cross and the undisplayed states. The step size of the cursor is determined by the bin widths (values in parenthesis on the menu screen) associated with the displayed axes. The tab keys will move the cursor along the x-axis 8 steps at a time.

A. Pan and Zoom Control.

The initial size of the displayed plot is controlled by the axis boundaries selected on the menu screen. The plot is scaled so that the full range within these bounds may be viewed. The plot may be expanded and examined within the initial bounds using the pan and zoom capabilities of the graphic screen.

Adjacent pairs of function keys are used to expand or contract (zoom) each axis: F7/F8 for x, F5/F6 for y and F3/F4 for z. Expansion occurs after values are mapped to the integer plot space so that rounding errors can occur on highly expanded axes. The program will beep when the expansion causes a resolution of less than 50 pixels/display (2%). Resolution can be retained or restored by changing the bounds for that axis on the menu screen (rather than the function keys).

Panning the x- and y- window in an expanded space is basically a cursor-controlled function. The cursor is active even when not displayed. The cursor moves with the arrow keys and moves the window over the plot (slowly) at the edge of the plot until it reaches axis bounds. The plot may also be moved in half-page increments with <page> and ^<arrow> keys. Further details on cursor control and all other keys are given in Table III. Note particularly that ^S moves the current cursor point to the lower left corner of the plot window.

When a histogram is selected but the points are not yet plotted, a box the size of the histogram bin appears in the lower left corner of the plot. The alt-function keys may be used to adjust the size of this box. The initial box widths are set from the box-width parameters in parenthesis on the axis listing in the menu screen.

B. Printing Plots and Listing Points

The ^P key sends a plot to the laser printer or to plot.las. The appearance of this plot is strongly controlled by printer options that have been selected on the options menu as described previously.

The ^L key makes a listing of each valid point on the listing file point.lst. For Cross and Matrix, the listing includes the x, y and z values of each point that is plotted or the x and y value if the y-axis is a histogram count. For 2d and Stripe, each line of output contains the x value and all selected y values. If one y-value meets the criteria for display, all y values will be listed (unless the bounds check variable is set to "all", in which case the point is neither listed or

plotted). Histogram counts or algebraic axes may be listed just as any other variables.

The output listing file is formatted in a manner similar to a input data file. To use this capability, only a single file should be written or the individual files may be stripped apart with an editor. *It is essential to rename point.lst (or any other output file) after it is created if you wish to save it.* A new point.lst is created (destroying the old) whenever one of the plotting programs is invoked.

The listing capability is an extremely powerful tool since it allows the creation of new selected data files. It is possible to selectively list all time periods with Fe/O > 0.5, for example, beginning with a large data base. The ability to list the values of complex algebraic expressions is also useful. Careful thought should be given to the selection criteria involved in the output listing, however.

If alpha-numeric keys are typed in graphics mode, they appear on the screen at the cursor position as expected. This feature is not fully implemented, however, since this text is not copied to the printer when the screen is replotted after ^P. Similar comments apply for the rubber-band line toggled on and off with ^R.

C. Color-Select Mode

A sub-mode of graphic screen is used to select and adjust the color palette. It is selected with alt-C and ended with <Esc>. Since normal graphic keys are disabled during color select, the note "set color!" appears on the screen to flag the new mode. It is wise to display a color-bar scale with ^Z before entering color-select mode so that you are not flying blind.

Several color palettes may be selected in this mode as seen in Table III. The normal "rainbow" palette repeats its color sequence over a z-axis distance set by the gradient adjustment using the left-right arrows. The other palettes vary from black at low z to white at high z over a width determined by the gradient setting. The phase or location of the mid-point of the variation is changed with the vertical arrow keys. The text color may be changed with the + and - keys.

The palette is useful in emphasizing or suppressing various regions of the z-axis. The algorithms were developed for studying star images and the C code was provided by R. Kaipa.

Table III -Key Definitions in Graphic Mode

F3/F4	Expand/Contract z (color) axis
F5/F6	Expand/Contract y axis
F7/F8	Expand/Contract x axis
Escape	Stop a data plot or Select Menu mode
Enter	Replot data
^C	Exit to DOS
^L	Add list points (x,y,z) to point.lst
^P	Add plot to printer file (plot.las) or print it
^S	Shift current cursor coord to origin
^X	Toggle cursor - Box/Tracking cross/Off
^Z	Toggle z(color)-axis display
Arrow keys	Step cursor left, right, up, down
^ Right arrow	Move right 1/2 page (+x)
^ Left arrow	Move left 1/2 page (-x)
Page up	Move up 1/2 page (+y)
Page down	Move down 1/2 page (-y)
^ Page up	Step z (color) scale up
^ Page down	Step z (color) scale down
Home	Move curs. to origin (lower left) on screen
End	Move curs. to upper right corner
^Home	Move plot to origin (lower left) of plot space
^End	Move plot to upper right corner of plot space
alt-F3/alt-F4	Shrink/expand z-axis histogram bin (no effect)
alt-F5/alt-F6	Shrink/expand y-axis histogram bin
alt-F7/alt-F8	Shrink/expand x-axis histogram bin
alt-C	Enter color-set mode:
Esc	Leave color-set mode
a	Select "apple" color set.
g	Select "gray" color set.
i	Select "inferno" color set.
r	Select "rainbow" color set.
Rt/left arrow	Incr/decr color gradient
^Rt/^left	Fast " " "
Up/down arrow	Raise/lower color axis
^Up/^down	Fast " " "
+/-	Move color of cursor
alt-H	Save current 1-D histogram
alt-N	Enter notepad.
	All edit keys are active for the current line.
Enter	Save line, return to graphic screen
Esc	Abandon line and return
alt-T	Step forward in time.
(Left-over features, not fully implemented)	
ASCII char	Type char on screen
^R	Rubber-band-line toggle
^W	Re-initialize windows
F1	Select Menu mode

D. Histograms

The histogram capability has been provided in the Cross and Matrix programs to determine the distribution of events in one or two dimensional space with or without constraints on other parameters in the data base. Histograms may be formed in log or linear space. Binning is accomplished using sparse matrix techniques (binary trees) and dynamic memory allocation.

The histogram axis behaves somewhat differently from other axes. It can not be used as a parameter in an algebraic expression, for example. In printing plots and listing points, the behavior is different from direct plots. <Enter> causes the binning to occur and the plot to be produced, ^P and ^L assume that the binning has occurred and generate their output rapidly from the current histogram. For direct plots, ^P and ^L can be used in lieu of <Enter> since they rescan the data to produce their output.

<Esc> may be used to interrupt the histogram binning and produce a partial result as with a direct plot.

A special facility has been installed for one-dimensional histograms to allow a subset histogram to overlay a previously-generated histogram. The procedure for using the histogram-save facility, ~~^H~~, is as follows: 1) Generate the larger less-constrained histogram as usual. 2) Press ~~^H~~ to save it. 3) Return to the menu screen and set the additional axis constraints. 4) Return to the graphics screen and press <Enter>; both histograms will appear. 5) Make hardcopies with ^P and ^L as desired. Both histograms will be erased together under the usual conditions such as a scale change, axis change or option change. The event count associated with the plot is for the last histogram only although the bin count includes both. An example of a single histogram is shown in Figure 9 and an overlaid histogram in Figure 10.

alt-H
alt-H

An example of a 2-D histogram is shown in Figure 11. Note that a large number of points may be binned, but the number of bins should be kept reasonable. The programs are not presently protected against memory overflow which probably occurs at about 2000 bins. If this becomes a problem, inform the author.

The histogram bin width is stored in the same format as the axis variable being binned. Little attention has been lavished on beautifying these formats. In linear space, the bin widths are entered in the appropriate units of the variable, integer bins for integers and float bins for float variables (types F, E, R, fI, etc.). Time bins are entered in sec in long integer format (no exponents).

In log space bin widths are interpreted as multiplicative factors e.g. 2.0 makes factor-of-2 bins and 1.414 makes 2 bins per factor of 2. For illegal values the default value of 2 is used. For binning integers in log space the value 256 $\log_2$(factor) is used, e.g. 256 for factor-of-2 bins, 128 for two bins per factor of 2. This strange mapping occurs for historical reasons because it allowed all integer operations to take place at high speed with no floating-point operations. Integers are still mapped to log space by a table look up.

E. The Notepad.

A capability has been added to the programs recently to facilitate the laborious process of scanning through large quantities of data to determine the times and intensities of particle increases. The notepad capability allows the user to capture the current cursor coordinates on a line of text, add edited comments to that line, and then append the line to the output file, note.pad.

To use the notepad, position the cursor as desired and press alt-N. The coordinates will appear on a text line in edit mode. After making any changes to the line, it may be accepted using <Enter> or rejected using <Esc>. The alt-N, <Esc> combination may also be used simply to view the cursor coordinates.

The screen containing the notepad line will accumulate these lines as long as the menu screen is not accessed. These earlier lines may not be edited, however, and they do not necessarily show the complete contents of note.pad. Like plot.las and point.lst, the file note.pad should be renamed, to save its contents, when the program is exited.

VII. SAMPLE PLOTS.

A selection of printed plots that illustrate different plot modes are shown in Figures 9- 17. Figures 9- 11 show histograms as described previously. Fig. 9 is printed with the standard Elite cartridge and the axis list in the header is included to show the criteria involved in the point selection on the plot. Fig. 10 was produced in a more finished form with the axis list omitted and using the Prestige 16 font. The primary histogram in Fig. 10 is the same as that in Fig. 9 and the secondary one was produced by enabling the lower bound (removing the minus sign) on the $^{3}He/^{4}He$ axis.

Figure 11 shows an example of 2-D binning from Matrix. Another example from Matrix with three separate pulse heights enabled and log axes is shown in Fig. 12. Note that the algebraic capability can be used plot pulse-height sums that are often of interest.

Figure 13 shows a more traditional time history plot combining data from two spacecraft. The plot is produced in landscape mode with the 16-point Prestige font. Figure 14 combines data from two files with different averaging periods. Figure 15 shows a more compressed view of energetic particle data with solar flare particles rising from the galactic cosmic-ray background.

Figures 16 and 17 demonstrate the use of the programs for utility plotting. The former shows the look-angle distributions of different facets of a telescope being designed and the latter was used to study an FFT algorithm. In both cases, the output of the programs that made these calculations was modified slightly to produce plotable files.

VIII. SOFTWARE DISTRIBUTION, PHILOSOPHY AND EVOLUTION

Copies of the software and of this document will be made available to users in the scientific and technical community upon request. Please feel free to make additional copies of both.

The source code for this software comprises over 11,000 lines of C code (and a small amount of assembly code). No commercial subroutine packages (e. g. graphics subroutine packages) were used. Code development has been driven by the requirements of specific scientific projects and we did not shrink from adding an important capability because the interface was a bit inelegant. Nevertheless, speed, convenience and ease of use were considered to be extremely important since they strongly affect productivity. Some of the keystroke sequences especially the Cntl- and Alt- sequences may be difficult for beginners to remember but are faster than menus or commands for experienced users.

A standard (but *not* proprietary) computer, display and printer were chosen by our group expressly to avoid having to waste software resources on supporting many different devices. An extremely high performance system with a display, printer and optical disk can be purchased for under $10K. Several scientists can be equipped with these systems for the price of one man-year of an experienced programmer.

Many of the limitations of the programs have been mentioned above. There are some additional limits on the dynamic range of plotted variables, log axes should not range over more than about 12 orders of magnitude, for example, and numbers along the axes that are forced into e-format (e. g. 2.345e-12) do not fit within the borders of the plot screen since most of the pixels have been reserved for data rather than for labels. This type of problem be overcome by rescaling the data onto a more benign range with the algebraic facility.

The programs are expected to evolve as new capabilities are added. Suggestions and comments on this evolution are welcome, however, the programming resources that have produced this software and are available for changes consist of one physicist available part time. Suggestions that are accompanied by offers of assistance will be given priority. Source code for the package will not normally be distributed in order to avoid the creation of many incompatible versions; if you are interested in making changes, I would prefer to work with you so that useful changes are made available to all users.

Cross plot 3.2 (all) 3-26-88
\isee\oxy.lst ISEE-3 Daily Avg. O >= 1.e-4 06:35:27 PM

x (log, bin= 34.4%) :
A6/A2; Fe/O
y (linear) :
No. of Days

A0	78 Aug 15 0000	87 Feb 7	0000	(86400)	TIME T2
A1	-1.00E-4	1000	(1)	1.90 2.82	C12 F2
A2	1.00E-4	1000	(1)	1.90 2.80	O/(cm^2 ster sec MeV/AMU) F2
A3	-1.00E-4	1000	(1)	1.90 2.80	NE20 F2
A4	-1.00E-4	1000	(1)	1.90 2.80	MG24 F2
A5	-1.00E-4	1000	(1)	1.90 2.80	SI28 F2
A6	1.00E-4	1000	(1)	1.90 2.80	Fe/(cm^2 ster sec MeV/AMU) F2
A7	-1.50E-1	20	(1)	1.3	^{3}He/^{4}He (-0.05) F2
B0	-5.00E-2	10.10	(1)		A1/A2; C/O F2
B1	-2.50E-2	5.05	(1)		A3/A2; Ne/O F2
B2	-2.50E-2	5.05	(1)		A4/A2; Mg/O F2
B3	-1.50E-2	3.03	(1)		A5/A2; Si/O F2
B4	5.00E-3	10.10	(1.41)		A6/A2; Fe/O F2
Z0	0	100	(0)		No. of Days I1

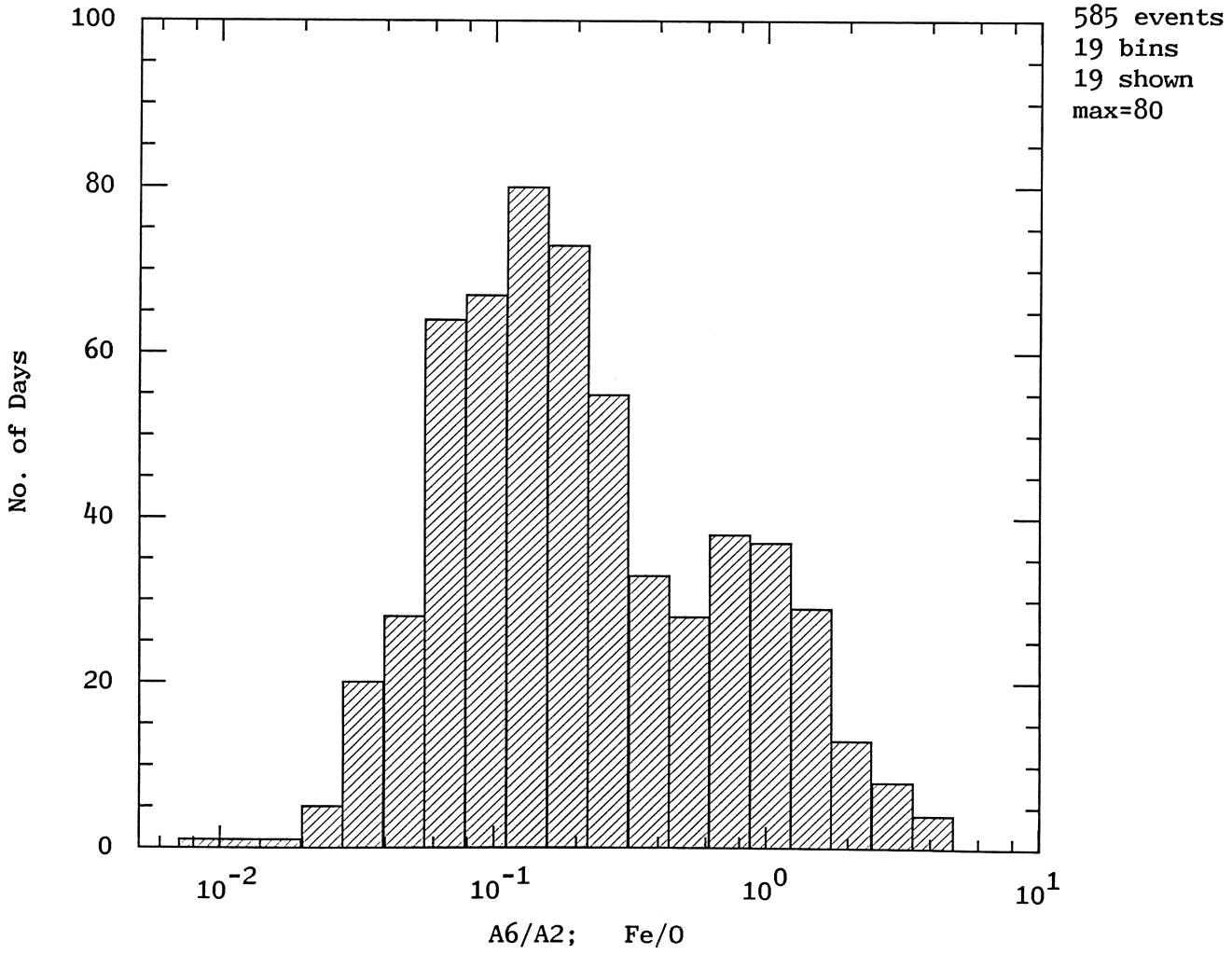


Figure 9. Single Histogram in Elite font and axes in header

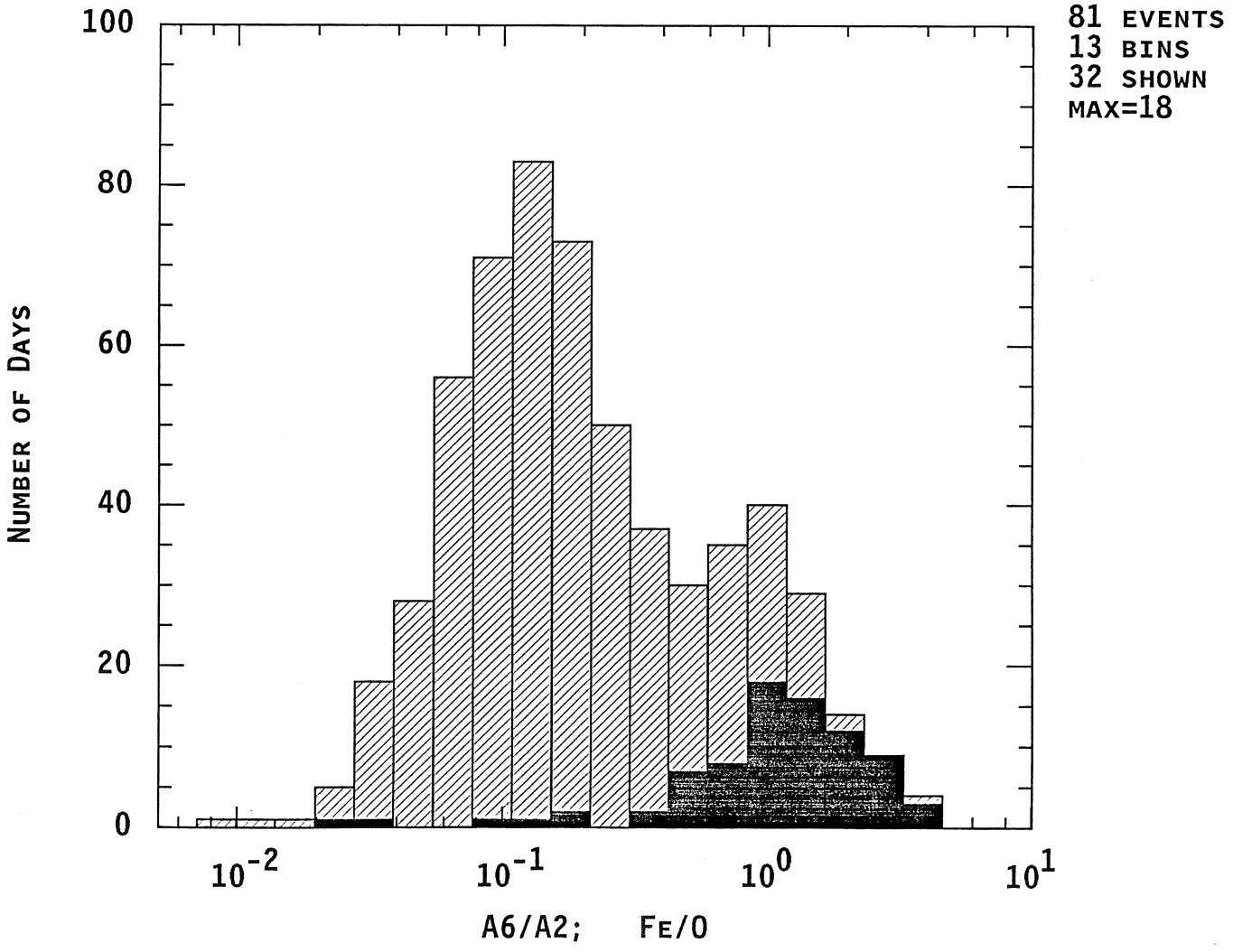


FIGURE 10. DUAL HISTOGRAM IN PRESTIGE 16 FONT.

```
MATRIX  V. 3.2  (ALL)                                          3-26-88
82JUN25.PHA          ISEE-3 PHA DATA FILE                      07:43:47 PM
  PHA VERSE Q: ICV1HE      VLET 1  EVENT TYPE 1
X (LINEAR, BIN=1) :
    D1    ICV1HE
Y (LINEAR, BIN=1) :
    D1    ICV1HE
Z (LOG) :                  BINNING DATA!
    HISTOGRAM
A0  82 JUN 25 0000  82 JUN 27 0000 (  900)|TIME                    T2
Q2          0       60 (    1)|D1    ICV1HE                      I1
Q3          0       60 (    1)|D1    ICV1HE                      I1
Q4          0        1 (    1)|E     ICV1HE                      I1
Q5          0     FFFF  M=FFFF|TAGS  ICV1HE                      B1
Z0          0      300 (    0)|HISTOGRAM                        I1
```

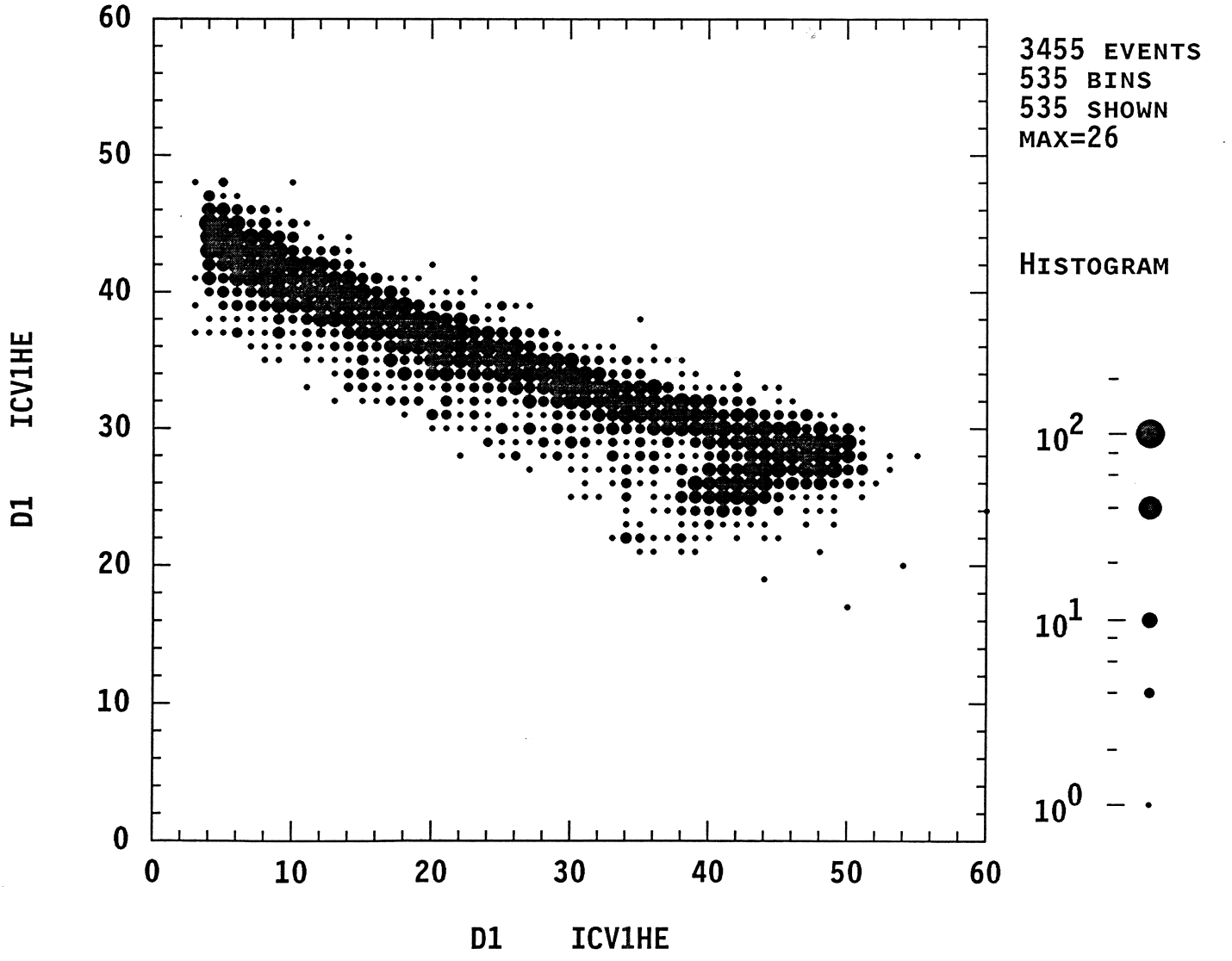


FIGURE 11. A PULSE-HEIGHT HISTOGRAM (MORE VIVID IN COLOR)

PHA VERSE I: ICH2ASTHI HET 2 A-STOP HIGH GAIN

X (LOG) :
A2 ICH2ASTHI
Y (LOG) :
A1 ICH2ASTHI
Z (LOG) :
C123 ICH2ASTHI

A0	82 JUN 25 0000	82 JUN 25 0600 (900)	TIME		T2
I2	0	4096 (1)	A1	ICH2ASTHI	I1
I3	0	4096 (1)	A2	ICH2ASTHI	I1
I4	0	4096 (1)	C123	ICH2ASTHI	I1
I5	0	FFFF M=FFFF	TAGS	ICH2ASTHI	B1
Z0	0	300 (0)	HISTOGRAM		I1

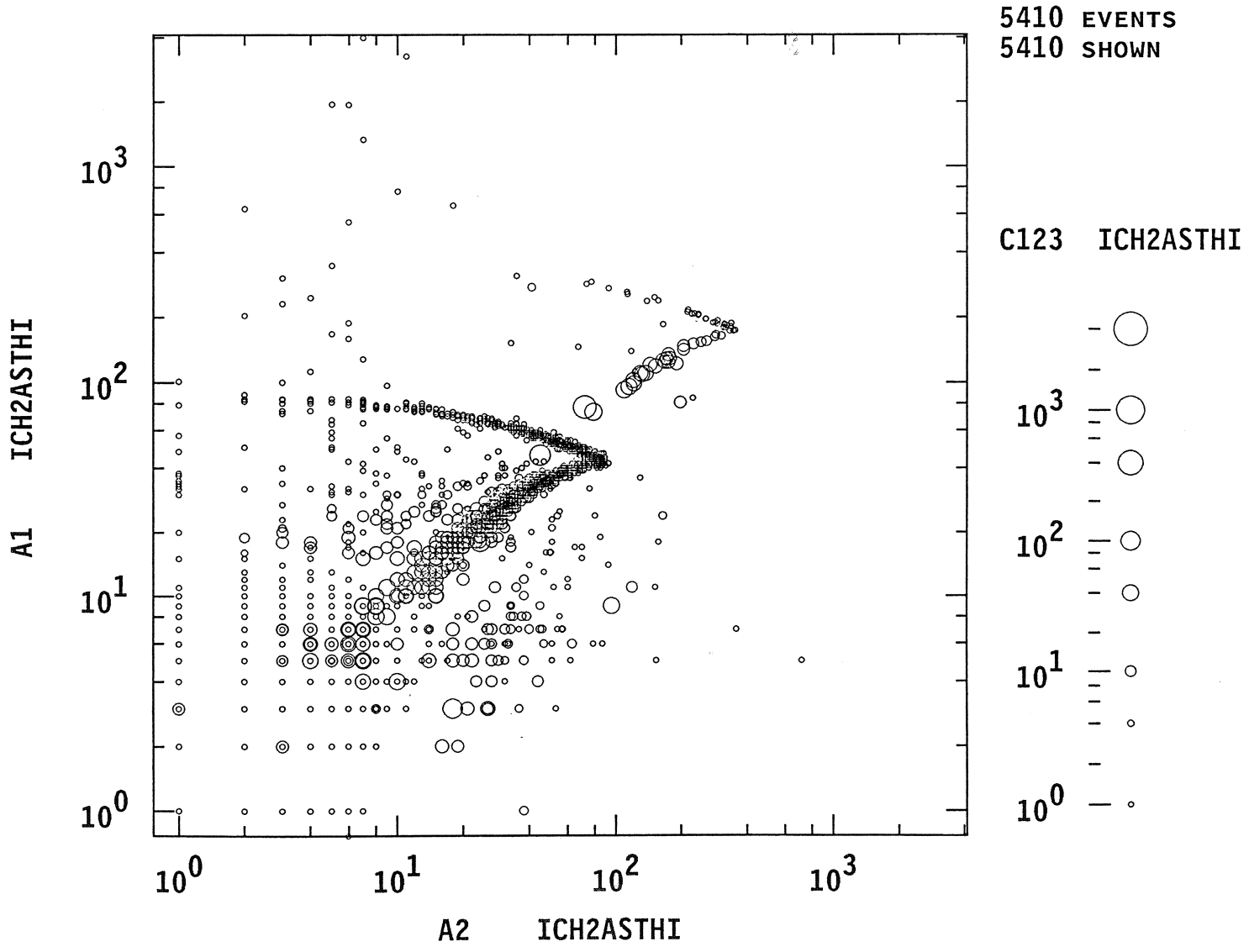


FIGURE 12. A 3-AXIS PHA DISTRIBUTION

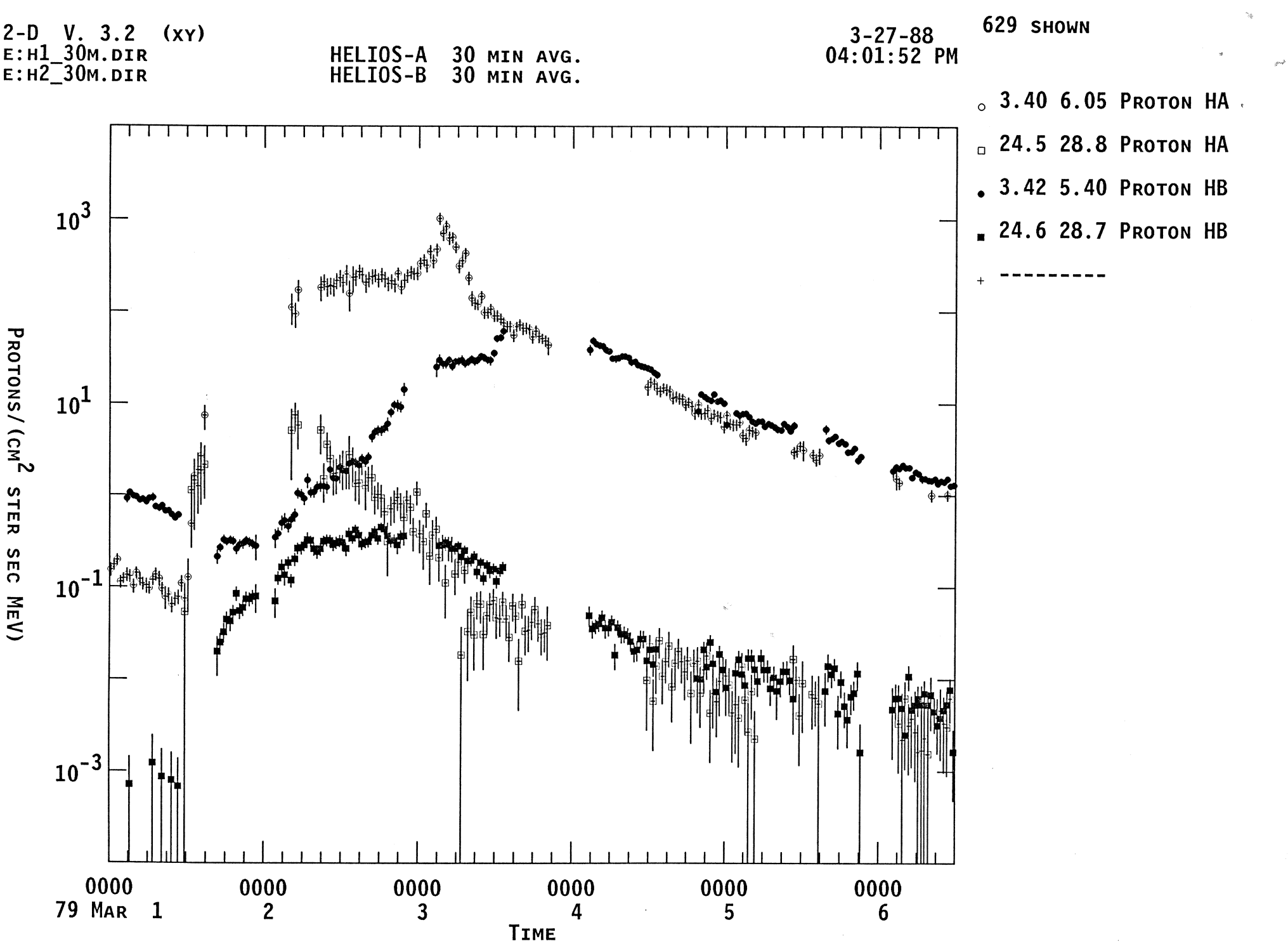


FIGURE 13. TIME HISTORIES OF PROTONS FROM TWO SPACECRAFT

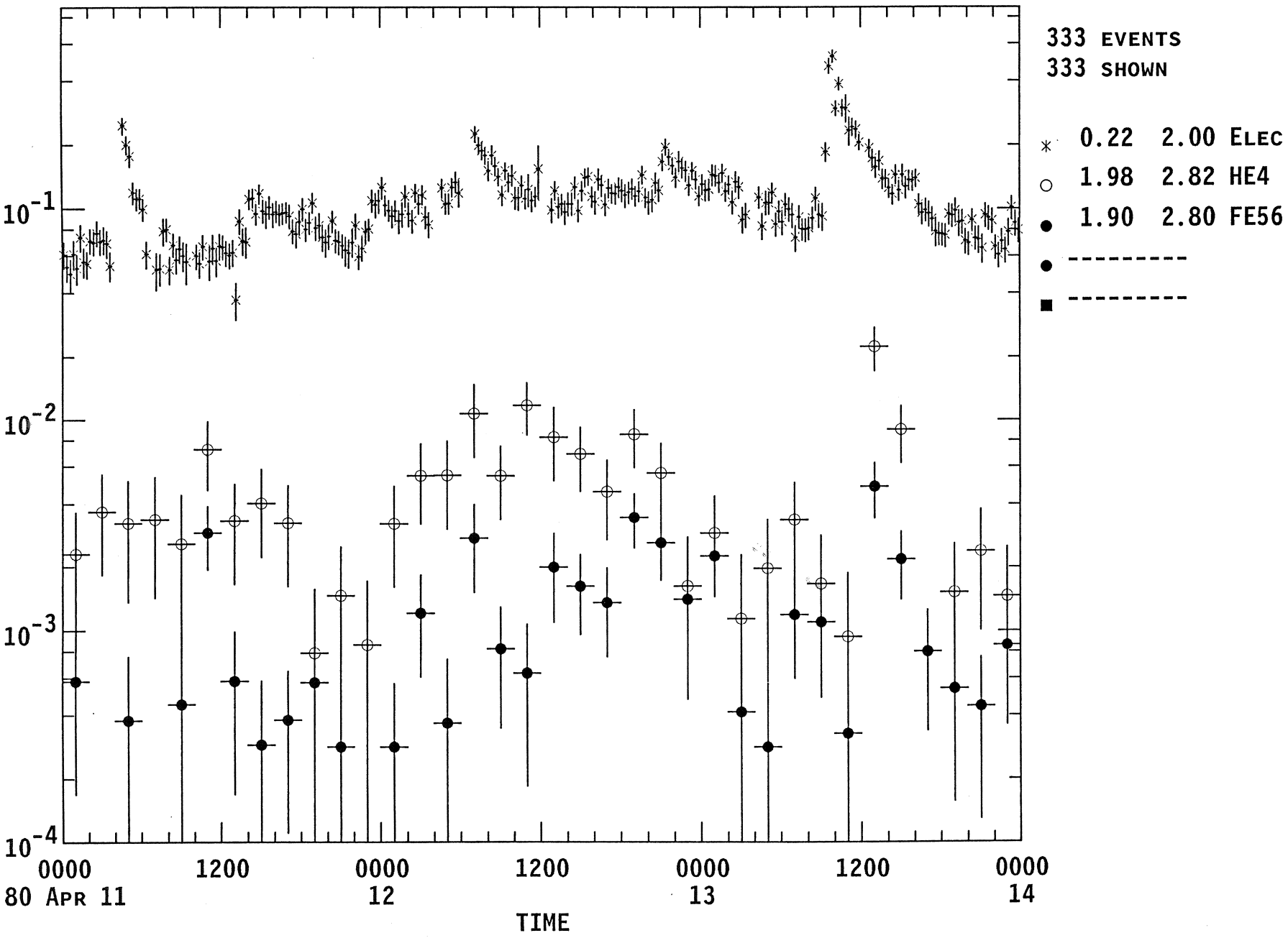


FIGURE 14. DATA FROM FILES WITH DIFFERENT AVERAGING INTERVALS

7658 EVENTS

7658 SHOWN

● 121 230 PROTON

○ ---------

□ ---------

• ---------

■ ---------

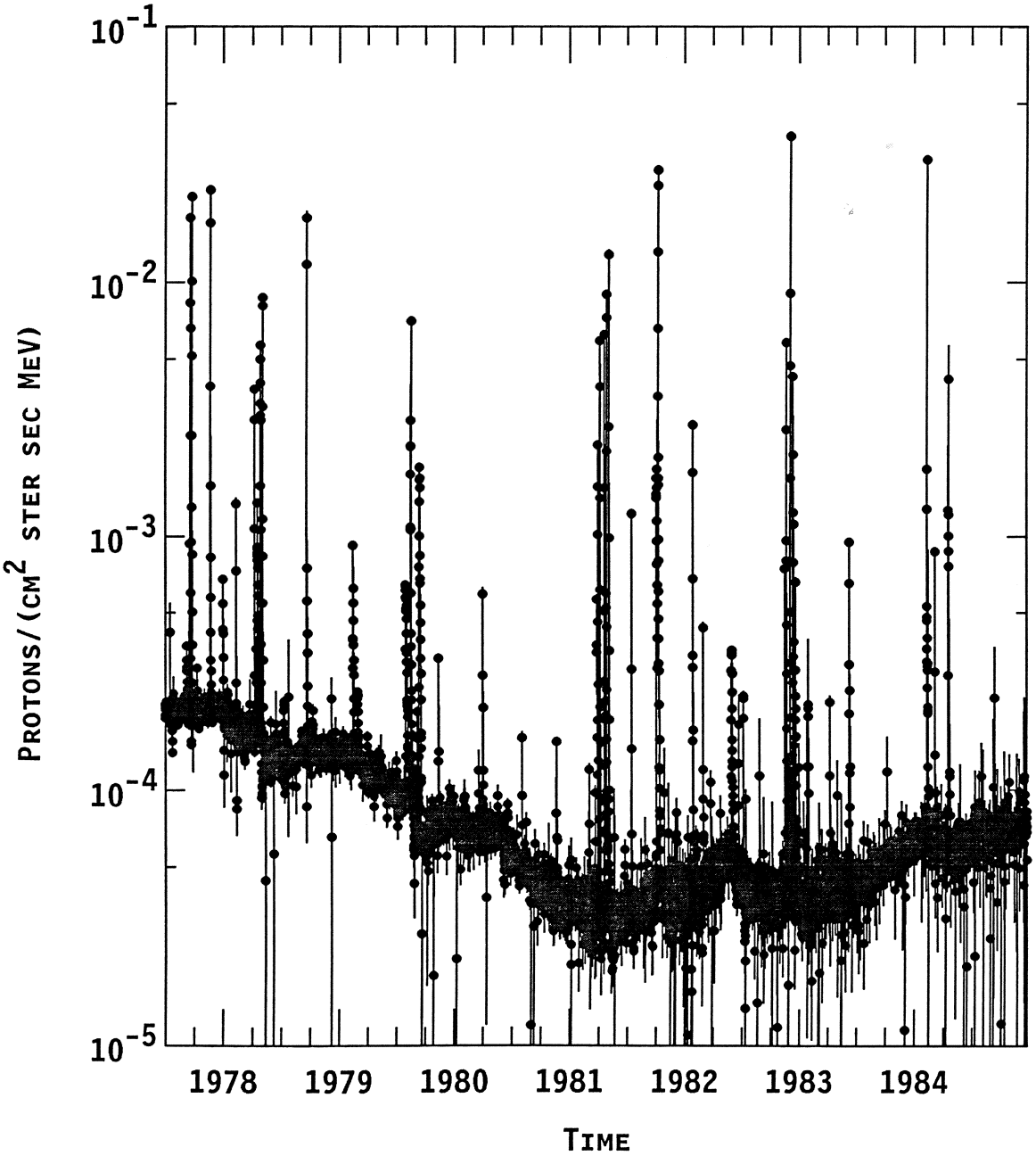


FIGURE 15. SOLAR CYCLE VIEW SHOWING SOLAR AND GALACTIC PROTONS

CROSS PLOT 3.2 (ALL) 3-27-88.
TABL.ASC TABLE - AVG. LAT., LONG., THICK. 06:17:33 PM

X (LINEAR) :
AVG. LONGITUDE
Y (LINEAR) :
AVG. LATITUDE
Z (LINEAR) :
DOME DET. NO.

A0	0	20	(1)	DOME DET. NO.		I1
A1	0	5	(1)	X INDEX		I1
A2	0	5	(1)	Y INDEX		I1
A3	-90	90	(1)	AVG. LATITUDE		F1
A4	0	200	(1)	AVG. LONGITUDE		F1
A5	0	30	(1)	AVG. THICKNESS (U)		F1
A6	0	10	(1)	THICKNESS OFFSET (U)		F1
Z0	0	100	(0)	HISTOGRAM		I1

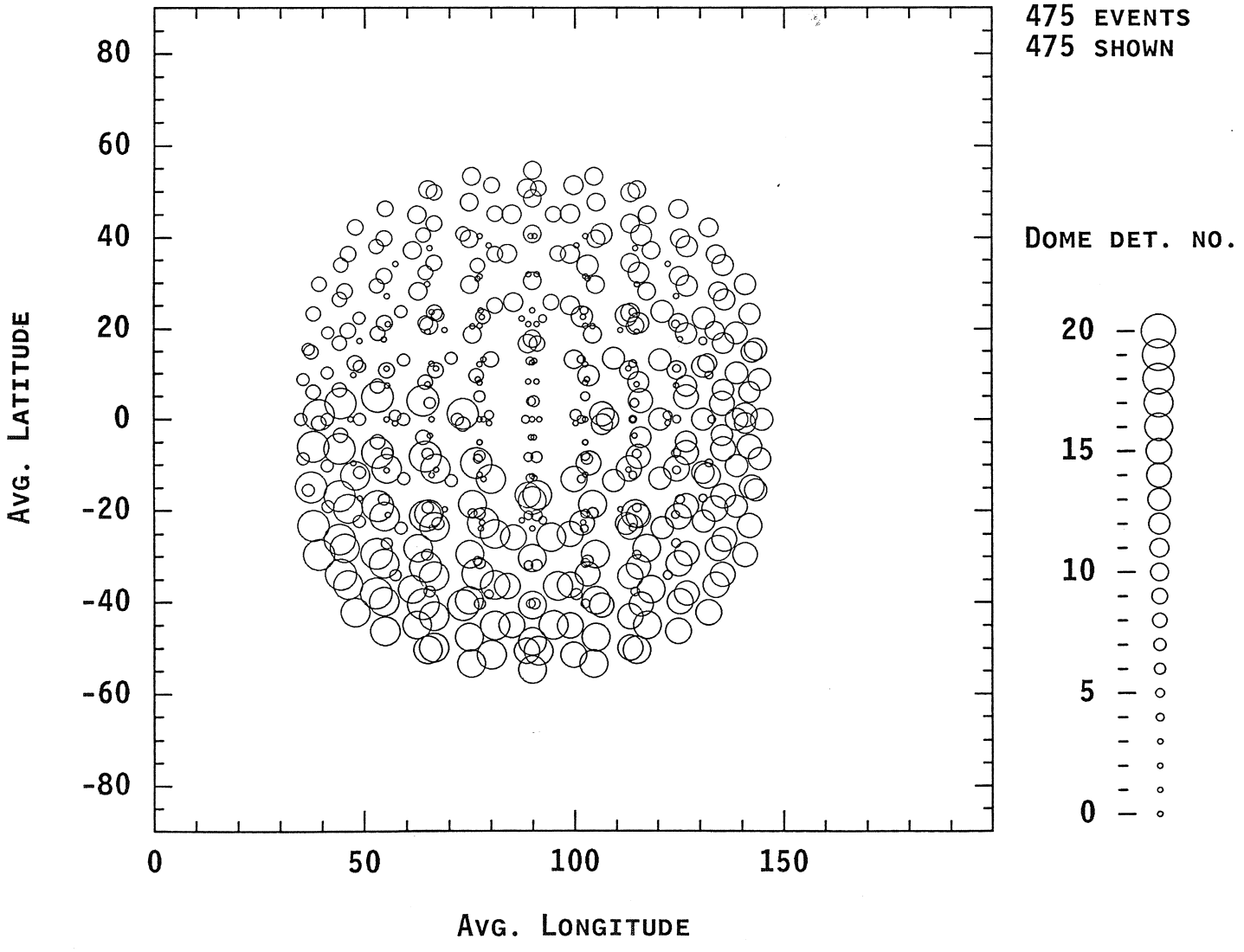


FIGURE 16. SCATTER PLOT OF TELESCOPE FACET LOOK ANGLES.

2-D V. 3.2 (XY) 3-27-88
FOUR.LST FFT OF SQUARE WAVES 06:35:16 PM

X (LINEAR) :
X
Y (LINEAR) :
INPUT*.08
T MAGNITUDE
T REAL
T IMAG

A0	0	63	(1)	X		F1
A1	0	360	(1)	T PHASE		F1
A2	-5.00E-2	1.00E-1	(1)	T MAGNITUDE		F1
A3	-1.00E-1	1.00E-1	(1)	T REAL		F1
A4	-1.00E-1	1.00E-1	(1)	T IMAG		F1
A5	-2.00E-2	1.00E-1	(1)	INPUT*.08		F1

256 EVENTS
256 SHOWN

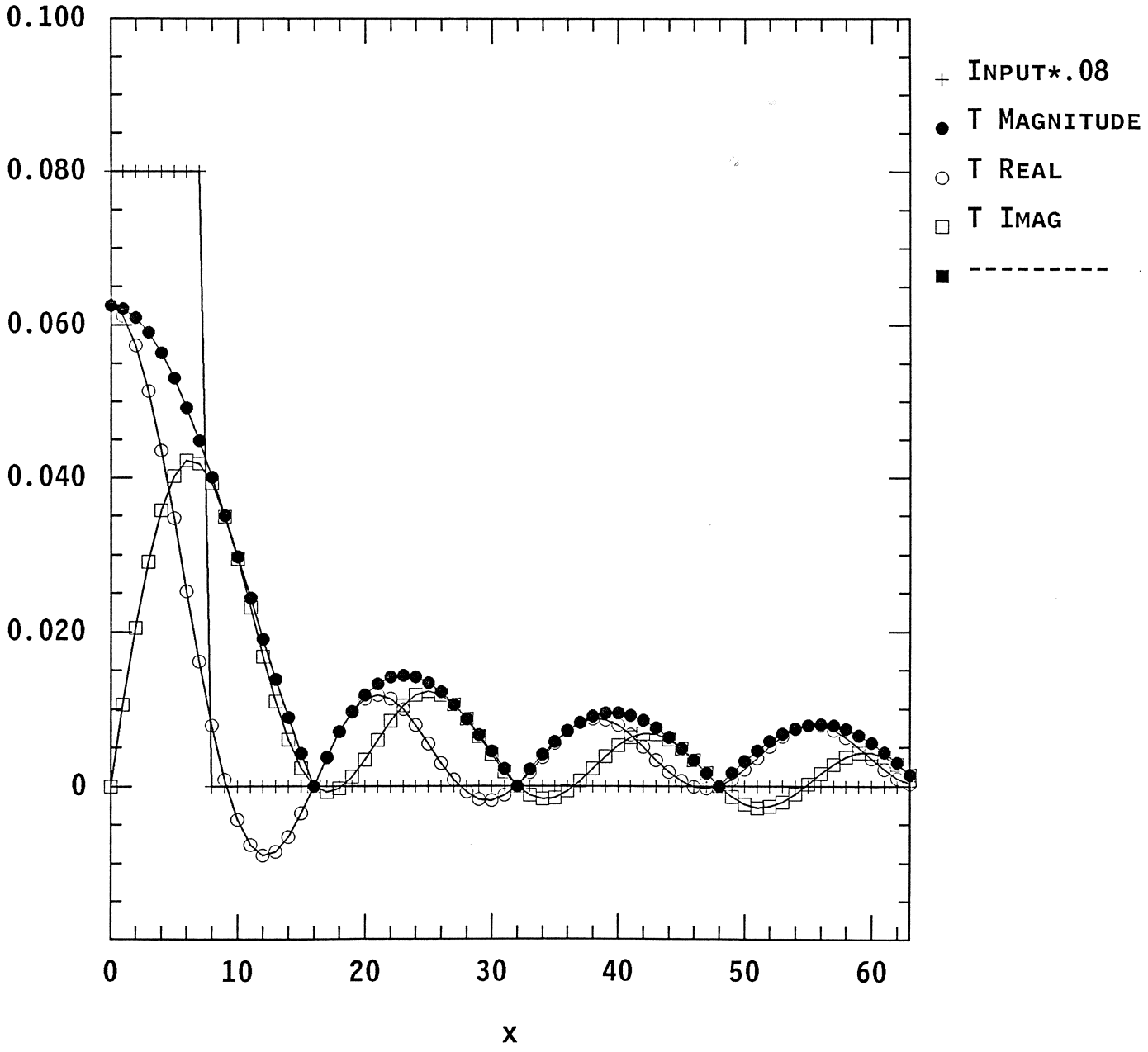


FIGURE 17. UTILITY PLOTTING; OUTPUT FROM AN FFT ALGORITHM

Blank Spacer Page

Table III -Key Definitions in Graphic Mode

F3/F4	Expand/Contract z (color) axis
F5/F6	Expand/Contract y axis
F7/F8	Expand/Contract x axis
Escape	Stop a data plot or Select Menu mode
Enter	Replot data
^C	Exit to DOS
^L	Add list points (x,y,z) to point.lst
^P	Add plot to printer file (plot.las) or print it
^S	Shift current cursor coord to origin
^X	Toggle cursor - Box/Tracking cross/Off
^Z	Toggle z(color)-axis display
Arrow keys	Step cursor left, right, up, down
^ Right arrow	Move right 1/2 page (+x)
^ Left arrow	Move left 1/2 page (-x)
Page up	Move up 1/2 page (+y)
Page down	Move down 1/2 page (-y)
^ Page up	Step z (color) scale up
^ Page down	Step z (color) scale down
Home	Move curs. to origin (lower left) on screen
End	Move curs. to upper right corner
^Home	Move plot to origin (lower left) of plot space
^End	Move plot to upper right corner of plot space
alt-F3/alt-F4	Shrink/expand z-axis histogram bin (no effect)
alt-F5/alt-F6	Shrink/expand y-axis histogram bin
alt-F7/alt-F8	Shrink/expand x-axis histogram bin
alt-C	Enter color-set mode:
Esc	Leave color-set mode
a	Select "apple" color set.
g	Select "gray" color set.
i	Select "inferno" color set.
r	Select "rainbow" color set.
Rt/left arrow	Incr/decr color gradient
^Rt/^left	Fast " " "
Up/down arrow	Raise/lower color axis
^Up/^down	Fast " " "
+/-	Move color of cursor
alt-H	Save current 1-D histogram
alt-N	Enter notepad.
	All edit keys are active for the current line.
Enter	Save line, return to graphic screen
Esc	Abandon line and return
alt-T	Step forward in time.

(Left-over features, not fully implemented)

ASCII char	Type char on screen
^R	Rubber-band-line toggle
^W	Re-initialize windows
F1	Select Menu mode

algebraic

Cntl enter after page down so that A60 is only one

A60
BP

A9*.002;1A+01*.002

D. Histograms

The histogram capability has been provided in the Cross and Matrix programs to determine the distribution of events in one or two dimensional space with or without constraints on other parameters in the data base. Histograms may be formed in log or linear space. Binning is accomplished using sparse matrix techniques (binary trees) and dynamic memory allocation.

The histogram axis behaves somewhat differently from other axes. It can not be used as a parameter in an algebraic expression, for example. In printing plots and listing points, the behavior is different from direct plots. <Enter> causes the binning to occur and the plot to be produced, ^P and ^L assume that the binning has occurred and generate their output rapidly from the current histogram. For direct plots, ^P and ^L can be used in lieu of <Enter> since they rescan the data to produce their output.

<Esc> may be used to interrupt the histogram binning and produce a partial result as with a direct plot.

A special facility has been installed for one-dimensional histograms to allow a subset histogram to overlay a previously-generated histogram. The procedure for using the histogram-save facility, ^H, is as follows: 1) Generate the larger less-constrained histogram as usual. 2) Press ^H to save it. 3) Return to the menu screen and set the additional axis constraints. 4) Return to the graphics screen and press <Enter>; both histograms will appear. 5) Make hardcopies with ^P and ^L as desired. Both histograms will be erased together under the usual conditions such as a scale change, axis change or option change. The event count associated with the plot is for the last histogram only although the bin count includes both. An example of a single histogram is shown in Figure 9 and an overlaid histogram in Figure 10.

An example of a 2-D histogram is shown in Figure 11. Note that a large number of points may be binned, but the number of bins should be kept reasonable. The programs are not presently protected against memory overflow which probably occurs at about 2000 bins. If this becomes a problem, inform the author.

The histogram bin width is stored in the same format as the axis variable being binned. Little attention has been lavished on beautifying these formats. In linear space, the bin widths are entered in the appropriate units of the variable, integer bins for integers and float bins for float variables (types F, E, R, fI, etc.). Time bins are entered in sec in long integer format (no exponents).

In log space bin widths are interpreted as multiplicative factors e.g. 2.0 makes factor-of-2 bins and 1.414 makes 2 bins per factor of 2. For illegal values the default value of 2 is used. For binning integers in log space the value 256 $\log_2$(factor) is used, e.g. 256 for factor-of-2 bins, 128 for two bins per factor of 2. This strange mapping occurs for historical reasons because it allowed all integer operations to take place at high speed with no floating-point operations. Integers are still mapped to log space by a table look up.

Blank Spacer Page

Add to Cross Documentation

Table I.-Key Definitions in Menu/Options Mode

F1	Select graphics mode
Escape	" " "
F10	Enable/Disable the "append command" option to output command file
^C	Exit to DOS
up/down arrows	Move cursor one row
Pg up/Pg down	Move axis list up/down 10 lines.
^Backspace	Delete the axis descriptor on the current line (if it is not currently selected as x, y, or z).
Enter	Select or change current row (see below)

Axis select:

Escape	Return to menu mode -no changes
F1	" " "
up/down arrows	Move cursor one row
Pg up/Pg down	Move list up/down 10 lines.
Enter	Select current row for menu
^C	Exit to DOS.

Edit axis parameters:

Escape	Return to menu mode -no changes
Insert	Toggle insert(blue)/replace(red).
left/rt arrows	Move cursor one column
up/down arrows	Accept changes on current line and move to next editable line.
Tab left/rt	Move to next tab stop (8 col).
Home	Move to left end of line
End	Move to right end of line
Del	Delete char under cursor
Backspace	Delete char to left of cursor
ASCII char	Insert/repl. the char. (>=20H)
Enter	Accept current line, ret. to menu.
^C	Exit the program

alt-O Display options screen:.
Parameters in the options screen may be selected or edited just as those in the menu screen. F1 returns to menu mode.

Table II -Key Definitions in Graphic Mode

F1	Select Menu mode
F2	Toggle active panel
F10	Enable/Disable the "append command" option to output command file
F3/F4	Expand/Contract z (color) axis
F5/F6	Expand/Contract y axis
F7/F8	Expand/Contract x axis
Escape	Stop a data plot or start it
Enter	Replot data
^C	Exit to DOS
^L	Add list points (x,y,z) to point.lst
^P	Add plot to printer file (plot.las) or print it
^S	Shift current cursor coord to origin
^X	Toggle cursor - Box/Tracking cross/Off
^Z	Toggle z(color)-axis display

```
Arrow keys        Step cursor left, right, up, down
^ Right arrow     Move right 1/2 page (+x)
^ Left arrow      Move left 1/2 page (-x)
Page up           Move up 1/2 page (+y)
Page down         Move down 1/2 page (-y)
^ Page up         Step z (color) scale up
^ Page down       Step z (color) scale down

Home              Move curs. to origin (lower left) on screen
End               Move curs. to upper right corner
^Home             Move plot to origin (lower left) of plot space
^End              Move plot to upper right corner of plot space

alt-F3/alt-F4   Shrink/expand z-axis histogram bin (no effect)
alt-F5/alt-F6   Shrink/expand y-axis histogram bin
alt-F7/alt-F8   Shrink/expand x-axis histogram bin

alt_L/alt_R     Move Z-axis left\right ( after right shift\left shift)

sft_F2/sft_F3   Increment/Decrement  panel movement step size (<<1/>>1)
sft_F4/sft_F9   Move panel diagonally downwards/upwards
sft_F5/sft_F8   Move panel  left/right
sft_F6/sft_F7   Move panel  upwards/downwards

alt-C             Enter color-set mode:
        Esc               Leave color-set mode
        a                 Select "apple" color set.
        g                 Select "grey" color set.
        i                 Select "inferno" color set.
        r                 Select "rainbow" color set.
        Rt/left arrow     Incr/decr color gradient
        ^Rt/^left         Fast "      "      "
        Up/down arrow     Raise/lower color axis
        ^Up/^down         Fast "        "    "
        +/-               Move color of cursor

(Left-over features, not fully implemented)
  ASCII char      Type char on screen
  ^R              Rubber-band-line toggle
  ^W              Refresh the PANELS and all ATRRIBUTES
```

DOCUMENTATION FOR *MACRO*

Version **1**_ 89.12

Tipparaju S.

MACRO - An Integrated Graphic Analysis Package for MATRIX & CROSS

A unified version of MATRIX and CROSS - ***MACRO*** (previously *COMBINE*) is developed which enables multiple analysis to be performed in a particular session. The software has an optional front-end Command Language Interface (CLI), and also the capability to perform realtime analysis.

Features:

The salient feature of Macro is the ability to open as many panels as the user requires and perform distinct analysis in each panel. An optional Command File can be created to interface with the software; this minimizes the need to input repetitively the information required to perform analysis in related sessions. The command file name is an optional first command argument for Macro. The session summary, which itself forms a command file for subsequent session(s), is the second command argument for Macro. For example:

C>
C> Macro <Input command file> < Output command file>

The input command file name is optional;
C> Macro .. O.K.

But, in conjunction with output command file, input command file name is required.
C> Macro <output command file> .. ERROR !

Again, the output command file name is optional.
C> Macro <input command file> .. O.K.

In other words, the first argument is always the input, and second argument is always the output. Default input and output files are built into the software.
Default input command file name : i.cmd
Default output command file name : o.cmd

The possible commands in a command file are listed below. The number or the index on the left of the command indicates the precise order of the command in the command file. Any other possible combination could result in errors.

1)	Command Type	
2)	File	Culg.plt
2)	Plotfile	Plot.las
2)	Listfile	List.lst
2)	Notefile	Note.pad
3)	Npan	1
	Cpan	1
4)	Ispha	0
5)	Verse	B

```
Axis       X  A0
Delete     B2
Set        Z0  1    50  (    1) | Number of Events
Cursor     0
Binlist    1
Zmode      2
Color      0
Symbol     4
Errbar     0
Join       0
Datefmt    2
Checkall   1
Syncmode   0
Toler      300
Toffset    0
Portrait   1
Tstep      .8
Stepmode   0
Tlag       0.2
Font       0
Dispen     1
Xlabel     1
Ylabel     1
Zlabel     1
Display    1
List       1
Print      1
Main       1
Auxlabel   1
Axismode   X  1
Boxsize    1500   1800
Dor        0  0
Dtop       1024    840
Curloc     512    420
Dispx      0  1024
Dispy      0    1600
Zaxis      2
Zax_step   2
Newcol     0
```

Help on commands:

Command

This is a header command to indicate that a command input file is effective. This is **required** as the **first** command in any input command file. Absence of this command in a command file overrides all the commands that follow in the input file.

Type

Not currently used.

File *<input data file name>*

The name of the input **data** file. At most three (3) data files can be currently opened. Each data file has to be command input seperately.As was previously, directories or files only. No mixed mode. Default is culg.plt.

Plotfile *<plot file name>*

The name of any plot file. The last of the input plot file names is effective. Default is plot.las.

Listfile *< list file name>*

The name of any listing file. The last of the input list file names is effective. Default is list.lst.

Notefile *<note file name>*

The name of any note pad file. The last of the input note file names is effective. Default is note.pad.

Npan *#*

The number of panels required. Default # is 1.

Cpan *#*

Current panel in context. Used to change the context of the panel. Default # is 1. The current panel is effective until a new one is specified.

Axis *Axis Column*

This command is used to select the *Axis* with a particular *Column* in the current panel. Default *Axis* is X ; default *Column* is A0.

Delete *Column*

Command used to delete *Column* in the current panel. Default *Column* B2.

Ispha *#*

Command used to prescribe the current panel for MATRIX (# = 1) or for CROSS (# = 0). Must be specified for each panel else default is CROSS.

Verse *Verse*

The *Verse* is selected as the active verse in Matrix. Default *Verse* is B.

Set *Column Lower bound Higher bound (Bin width) /Label*

To change/edit a particular *Column* . Default is B2 2 3 (1)|New

Cursor #

Enable the cursor if # = 1; disable the cursor if # = 0.

Binlist #

Plot/List Annotation.
Header only if # = 0; Header and all bins if # = 1.

Zmode #

Z_Axis option value specifying the plotted/printed variation of the symbol with Z-axis value.

= 0, none;
= 1, color/symbol;
= 2, color & size;
= 3, size/symbol;

Color #

Color; default # = 64.

Symbol #

Print Symbol

# = 0, symbol is +	# = 5, symbol is ∞
# = 1, symbol is x	# = 6, symbol is ▲
# = 2, symbol is *	# = 7, symbol is ●
# = 3, symbol is Δ	# = 8, symbol is ■
# = 4, symbol is O	# = 9, text value.

Errbar #

Enable error bars if # = 1; else disable error bars if # = 0;
Plot point if value < 1.5 error

Join #

Join sequential points if # = 1; else display points if # = 0.

Datefmt #

Date output format type

= 0, format type is 78 Apr 10
= 1, format type is 78 / 04/ 10
= 2, format type is 78 100

Checkall #

Check axis bounds on all columns if # = 1; else check on x, y, z only.

Syncmode #

Time synchronization of multiple files; independent or sync. if times within tolerance.

Toler #

= Time sync. tolerance.

Toffset #

= Time sync. offset

Portrait #

Laser printer plot orientation.
if # = 0, orientation is Landscape; else if # = 1, Portrait

Stepmode #

Time step mode.

= 0: Fraction of dt only;
= 1: Fraction + next time on file 0;
= 2: Fract. + next time on any file.

Tstep #

Time step value = #; fraction of current time interval.
Default = 0.9

Tlag #

Time lag value = #; fraction of current time interval.
Default = 0.1

Font #

Font number

= 0, Elite;
= 1, H.PPrestige 14;
= 2, H.P. Prestige 16.

Dispen #

if # = 0 disable display in the current panel; if # = 1, enable.

Xlabel #

if # = 1, label X axis in the plot.

Ylabel #

if # = 1, label Y axis in the plot

Zlabel #

if # = 1, label Z axis in the plot

Display #

if # = 1, display the plot on the screen; else not.

List #

if # = 1, List the coordinates of all the points on the current panel to list file, else not.

Print #

if # = 1, plot the plot on plot file (laser printer), else not.

Main #

if # =1, return to text window for further analysis.

Auxlabel #

Write auxilliary details like event count, number plotted etc.

= 0, none;
= 1, realtime plot
= 2, end of plot
= 3, realtime and end of plot

Axismode Axis #

Set axis mode of *Axis* # = 0, linear
= 1, log
= 2, time.

Boxsize # #

Laser printer box size (300/inch) of panel 1; all other panels are scaled proportionally. Default 1500 1500.

Dor # #

Origin of the current panel. Default 0, 0 (Npan = 1)

Dtop # #

Diagonal upper bounds of the current panel. Default 1024, 840 for panel # 1

Curloc # #

X, Y coordinates of Cursor location; default (512,420)

Dispx # #

X axis range with lower and upper bounds for the current panel; default
0, 1024 (Cpan = 1)
1048, 2048 (Cpan = 2)

Dispy # #

Y axis range with lower and upper bounds for the current panel;default
0, 840 (Cpan = 1)
0, 840 (Cpan = 2)

Zaxis #

Z axis scale display type

= 0, none;
= 1, color bar whose width indicates symbol size
= 2, selected points in scale

Zax_step #

Step size for movement of the Z axis location

Newcol #

Create a new algebraic column if # = 1.
Enters a new algebraic column that can be set as desired using the **SET** command.

Note(s)

1) The commands in the COMMAND LIST without any index, can occur any number of times.

2) Focus on the command CPAN: this command is required when ever the context of the panel changes.

3) Command(s) which refer to unindentifiable quantities are not effected. For example,

Axis X C2.

If Coulmn C2 is not defined at the time the above command is sequenced in the command input file, the axis X is not set to C2. This error is possible if C2 is an algebraic column, not yet defined.

Blank Spacer Page

CROSS/PC

Study.

7530-00-286-6952
FEDERAL SUPPLY SERVICE

Steps to get spectra

	data files needed	action
① needed: data - fluxes	: comp8at.flx	Create comp8.dir
header for ↑	: comp8.hdr	
MD file	: comp8.MD	

② time pds : int1a.lis : Spectrum time periods for analysis (contained within comp8at.flx file)
md file : int1a.md
Create int1a.dir

③ run the FSUM program

ex. Fsum 7 int1a.dir comp8.dir templ.dat

See header file comp8.hdr:
Fsum selects all verse 7 from comp8at.flx which fall into the spectrum time period lists.
(~~The several time periods of int1a.lis are evidently summarized as 11 adapters.~~)

④ make templ.dat hdr file : using comp8.hdr, make a header for templ.dat
Change I to L in verse description because of Long time summary

make templ.MD + create directory for templ.dat (templ.dir)

⑤ run the VERT program

VERT templ.dir
Select up to 8 bins
Press Cntl-L → data goes to point.lst

⑥ use point.lst as spectrum input to 2D or cross
point.lst may need header edit.
set options menu time step

over →

make directory for spect.dat

Intia.dir already exists for event list

use Intia as file 0
Spect as file 1

2D Intia.dir Spect.dir (see pg 19 of ROSS guide)

tried: time A0 set to end Sep 27 0000
step + lag default
this doesn't do it right - close the

temp1.dir
d:\spectrum\temp1.hdr 1978 aug 15 0000 1978 aug 15 0000
d:\spectrum\temp1.dat 1978 aug 15 0000 1982 jan 1 0000

temp1.MD

```
 BINARY CREATED  1987 NOV 21 04:54
ISEE-3   8 Hr  Composition Data (trend checked)
;
78/ 8/15   0: 0: 0    ; START TIME OF FILE
82/ 1/ 1   0: 0: 0    ; STOP  TIME OF FILE
0    0    0800
;
; TIME OF FIRST CHAPT. IN FILE  78/ 8/15   0: 0: 0
;
#0  RATE
  T2                                  TIME ;
# 7   FLUX    22-VLET II, EVENT TYPE 1
  R2     1.0E-05     1.0E+05          VLET II ET 1    RATE
  L      0             32767          COUNT ON VLET II  ET 1    MATRIX
;TYPE    E MIN     E MAX      GEOM     PARTICLE     MODENAME  EVENT TYPE
  L       1.10      1.34      0.252    HE4      ;   IID2         22
  L       1.34      1.63      0.272    HE4      ;   IID2         22
  L       1.35      1.65      0.187    HE3      ;   IID2         22
  L       1.98      2.82      0.214    HE4      ;   IID3         22
  L       2.29      3.24      0.221    HE4      ;   IID3         22
  L       3.72      5.42      0.227    HE4      ;   IID3         22
  L       2.26      3.41      0.206    HE3      ;   IID3         22
#END
```

tampl.hdr

```
D:\SPECTRUM>print temp1.hdr

    D:\SPECTRUM\TEMP1.HDR is currently being printed

D:\SPECTRUM>type temp1.dat
Binary  created 09/10/89
 Short intervals
78 aug 15 0000
82 jan  1 0000
#0  Rate
T2              Time
# 7   FLUX    22-VLET II, EVENT TYPE 1
  R2    1.0E-05    1.0E+05          VLET II ET 1    RATE
  L     0            32767          COUNT ON VLET II  ET 1    MATRIX
;TYPE   E MIN    E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  L       1.10     1.34     0.252   HE4     ;   IID2        22
  L       1.34     1.63     0.272   HE4     ;   IID2        22
  L       1.35     1.65     0.187   HE3     ;   IID2        22
  L       1.98     2.82     0.214   HE4     ;   IID3        22
  L       2.29     3.24     0.221   HE4     ;   IID3        22
  L       3.72     5.42     0.227   HE4     ;   IID3        22
  L       2.26     3.41     0.206   HE3     ;   IID3        22
;
#End
  `aj mk p
```

```
comp8.dir
c:\isee\comp8.hdr        1978 aug 15 0000        1978 aug 15 0000
c:\isee\comp8at.flx      1978 aug 15 0000        1983 jan  1 0000
```

comp8.MD

To make spectra s

View flx file (from PCFLUX pgm) header
edit example header to agree (can lines be edited out as a block from flux? no - not save to edit mixed data)
make a pgm to do it!

direct comp8.MD
(makes comp8.dir)

Made INT1a.LIS spectral periods 78-82 from int1.LIS example
rename INT1a.LIS → INT1a.hdr
make int1a.MD
direct int1a.MD

run fsum 7 int1a.dir comp8.dir temp1.dat
creates a PC flux like format file of verse 7 only from the main database. The several time periods given by INT1a.hdr are apparently summarized as "chapters" (?)

made temp1.hdr - edit of examp.hdr to look like temp1.dat header
made temp1.MD
direct temp1.MD →
vert temp1.dir

Change I to L in verse description because of long time summing

```
TEMP1     HDR        996    9-10-89    1:41a
TEMP1     BAK        152    9-06-89    7:52p
TEMP1     DIR        469    9-10-89    1:42a
PLOT      LAS         10    9-10-89    4:43a
POINT     LST          0    9-10-89    1:48a
NOTE      PAD         62    9-10-89    1:48a
FLUXREAD  C         1193    4-20-90    8:44p
SPECT     DAT      23603    9-10-89    1:48a
TEST      DIR        139    7-05-90    3:45p
       28 File(s)    251904 bytes free

D:\SPECTRUM>dir /w

 Volume in drive D has no label
 Directory of  D:\SPECTRUM

.                  ..                 COMP8AT   FLX      COMP8     HDR      COMP8     BAK
INT1      LIS      BIGABUND BAT       VERT      EXE      FSUM      EXE      DIRECT    EXE
COMP8     MD       EXAMP    HDR       COMP8     DIR      INT1A     MD       INT1A     LIS
INT1A     BAK      INT1A    DIR       TEMP1     DAT      TEMP1     MD       TEMP1     HDR
TEMP1     BAK      TEMP1    DIR       PLOT      LAS      POINT     LST      NOTE      PAD
FLUXREAD  C        SPECT    DAT       TEST      DIR
       28 File(s)    251904 bytes free

D:\SPECTRUM>
```

```
      #include <stdio.h>

/* change struct and print formats according to the number of flux verses  */

      struct flux_ver {
       int ver;
       long start;
       long stop;
       int req;
       float rate;
       float live;
       long cnt;
       int pha1;
       int pha2;
       int pha3;
       int pha4;
       int pha5;
       int pha6;
       }
      struct flux_ver rec;

      main(argc,argv)

      int argc;
      char *argv[];
      {
      FILE *fp;
      char ch;
      char buffer[30];

      if ((fp = fopen(argv[1],"rb")) == NULL)
       printf("\n FILE CANNOT BE OPENED");

      fscanf(fp,"%s",buffer);
      while ((strcmp(buffer,"#END") != 0) && (!feof(fp)))
       fscanf(fp,"%s",buffer);
      printf("\n Buffer = %s",buffer);

      ch = fgetc(fp);
      ch = fgetc(fp);
      ch = fgetc(fp);

      while (!feof(fp))   {
       fread(&rec,sizeof(struct flux_ver),1,fp);
       printf("\n %d %ld %ld %d %f %f %ld %d %d %d %d %d %d",rec.ver,rec.start,r
ec.stop,
            rec.req,rec.rate,rec.live,rec.cnt,rec.pha1,rec.pha2,rec.pha3,
            rec.pha4,rec.pha5,rec.pha6);
                        }
      fclose(fp);
        }   /* end program */
```

output is ascii file, user must add labels

Program VERT makes ascii file from a file which was made from make database start-stop time

fsum only does 1 verse at a time

prog. 2. no pgm to get longer averaging from flux verses

pc flux pgm should be able to get this from PHA database

Std "FLUX" DB has 4 directories, isee IMP-8 2 helios

Cross "DIRECTORY"

verse# of I-data input

fsum 5 input \isee\comp2.dir temp1.dat

direct int1.~~md~~ int1.dir

direct temp1.dat

fsum 7, temp1.dir \isee\comp2.dir temp2.dat

bagabund ← eventually get file ~~[illegible]~~

vert bagabund.dir

select 8 pts (max)

point.lst may need editing in header

≫ make into directory - make md file then do direct

~~[illegible]~~

Cross plots junk.dir

int1a.dir
d:\spectrum\int1a.lis 1978 aug 15 0000 1982 jan 1 0000

int1a.md

```
ASCII
 Short intervals
78 aug 15 0000
82 jan  1 0000
#0  Rate
T2                Time
#End
0 1978 Sep 23 1400   1978 Sep 24 1808
0 1978 Sep 25 2332   1978 Sep 27 1348
0 1979 Apr  3 1440   1979 Apr  4 0728
0 1979 Apr  5 0812   1979 Apr  6 1844
0 1979 Jun  6 1542   1979 Jun  6 2146
0 1979 Jun  7 2230   1979 Jun  9 1630
0 1979 Jul  6 0108   1979 Jul  6 2044
0 1979 Jul  7 0508   1979 Jul  9 0636
0 1979 Aug 19 0646   1979 Aug 20 0922
0 1979 Aug 20 1814   1979 Aug 23 0118
0 1979 Sep 14 2344   1979 Sep 18 0800
0 1979 Nov 16 0244   1979 Nov 18 1812
0 1980 Apr  4 1945   1980 Apr  7 1457
0 1980 Jul 18 1440   1980 Jul 18 1920
0 1980 Jul 19 1304   1980 Jul 21 1720
0 1980 Oct 15 1756   1980 Oct 19 1132
0 1980 Nov 23 1723   1980 Nov 26 2347
0 1981 Apr  1 0813   1981 Apr  3 1929
0 1981 Apr 10 2347   1981 Apr 14 0035
0 1981 Apr 24 1652   1981 Apr 26 0228
0 1981 Apr 26 0900   1981 Apr 27 0752
0 1981 May  9 0510   1981 May 10 1858
0 1981 May 11 2258   1981 May 13 0038
0 1981 May 16 1527   1981 May 17 1639
0 1981 May 18 0707   1981 May 19 1903
0 1981 Jul 20 1854   1981 Jul 23 0322
0 1981 Aug  9 1019   1981 Aug 12 0915
0 1981 Oct  9 0239   1981 Oct 12 0835
0 1981 Oct 12 0932   1981 Oct 13 1044
0 1981 Oct 14 1856   1981 Oct 16 1516
0 1981 Dec 11 1701   1981 Dec 13 0841
```

intia.lis

```
comp8.dir
d:\spectrum\comp8.hdr        1978 aug 15 0000        1978 aug 15 0000
d:\spectrum\comp8at.flx      1978 aug 15 0000        1982 jan  1 0000
```

Comp8.md

```
 BINARY CREATED  1987 NOV 21 04:54
ISEE-3   8 Hr  Composition Data (trend checked)
;
78/ 8/15   0: 0: 0    ; START TIME OF FILE
82/ 1/ 1   0: 0: 0    ; STOP  TIME OF FILE
0    0    0800
;
; TIME OF FIRST CHAPT. IN FILE  78/ 8/15   0: 0: 0
;
#0  RATE
  T2                                  TIME ;
# 1   FLUX     7-HET I BSTP, HIGH GAIN
  R2    1.0E-05    1.0E+05           HET I   BSTP,HI RATE
  L     0            32767           COUNT ON HET I    BSTP,HI MATRIX
;TYPE   E MIN    E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  I      22.14    27.17     1.705   PROTON  ;   IB2          7
;
# 2   FLUX     9-HET I BST, LOW GAIN
  R2    1.0E-05    1.0E+05           HET I   BSTP,LO RATE
  L     0            32767           COUNT ON HET I    BSTP,LO MATRIX
;TYPE   E MIN    E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  I      30.01    45.09     1.598   HE4     ;   IL3          9
;
# 3   FLUX    12-HET II AST, HIGH GAIN
  R2    1.0E-05    1.0E+05           HET II  AST,HI  RATE
  L     0            32767           COUNT ON HET II   AST,HI  MATRIX
;TYPE   E MIN    E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  I       4.45     6.41     1.107   PROTON  ;   IIA2        12
  I       7.08    12.83     1.283   PROTON  ;   IIA3        12
  I       0.22     2.00     1.280   ELEC    ;   IIA2        12
  I       7.08    12.64     1.217   HE4     ;   IIA3        12
;
# 4   FLUX    13-HET II AST, LOW GAIN
  R2    1.0E-05    1.0E+05           HET II  AST,LO  RATE
  L     0            32767           COUNT ON HET II   AST,LO  MATRIX
;TYPE   E MIN    E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  I      24.44    47.86     1.270   C12     ;   IIA3        13
  I      12.71    24.76     1.283   N14     ;   IIA3        13
  I      24.76    45.52     1.279   N14     ;   IIA3        13
  I       9.50    13.01     1.273   O16     ;   IIA2        13
  I      14.11    24.27     1.283   O16     ;   IIA3        13
  I      24.27    45.08     1.282   O16     ;   IIA3        13
  I      25.33    46.07     1.283   SI28    ;   IIA3        13
  I      15.07    23.00     1.255   FE56    ;   IIA2        13
  I      24.99    45.56     1.283   FE56    ;   IIA3        13
;
# 5   FLUX    19-VLET I, EVENT TYPE 0
  R2    1.0E-05    1.0E+05           MEAN VLET  ET 0    RATE
  L     0            32767           COUNT ON VLET Mean  ET 0    MATRIX
;TYPE   E MIN    E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  I       1.90     2.82     0.259   C12     ;   MD2
  I       3.96     7.08     0.287   C12     ;   MD3
  I       7.08    12.67     0.279   C12     ;   MD3
  I       1.90     2.81     0.287   N14     ;   MD2
  I       3.97     7.02     0.287   N14     ;   MD3
  I       7.02    12.60     0.287   N14     ;   MD3
  I       1.90     2.80     0.287   O16     ;   MD2
  I       3.94     7.01     0.287   O16     ;   MD3
  I       7.01    12.53     0.287   O16     ;   MD3
  I       1.90     2.80     0.287   NE20    ;   MD2
  I       4.00     7.16     0.287   NE20    ;   MD3
  I       7.16    12.58     0.287   NE20    ;   MD3
```

Comp8.dat v1/2

comp8.hdr 2/2

```
  I       1.90      2.80      0.287    MG24     ;    MD2
  I       3.97      7.10      0.287    MG24     ;    MD3
  I       7.10     12.60      0.287    MG24     ;    MD3
  I       1.90      2.80      0.285    SI28     ;    MD2
  I       3.98      7.01      0.287    SI28     ;    MD3
  I       7.01     12.53      0.287    SI28     ;    MD3
  I       1.90      2.81      0.275    S32      ;    MD2
  I       4.02      7.10      0.287    S32      ;    MD3
  I       7.10     12.53      0.287    S32      ;    MD3
  I       1.91      2.80      0.269    CA40     ;    MD2
  I       3.96      7.13      0.287    CA40     ;    MD3
  I       7.13     12.61      0.287    CA40     ;    MD3
  I       1.90      2.80      0.287    FE56     ;    MD2
  I       4.01      7.11      0.287    FE56     ;    MD3
  I       7.11     12.55      0.287    FE56     ;    MD3
  I      12.55     24.03      0.287    FE56     ;    MD3
;
;# 6   FLUX    21-VLET II, EVENT TYPE 0
;
# 7   FLUX    22-VLET II, EVENT TYPE 1
  R2    1.0E-05    1.0E+05          VLET II ET 1    RATE
  L     0            32767          COUNT ON VLET II  ET 1    MATRIX
;TYPE   E MIN     E MAX      GEOM     PARTICLE     MODENAME  EVENT TYPE
  I       1.10      1.34      0.252    HE4      ;    IID2        22
  I       1.34      1.63      0.272    HE4      ;    IID2        22
  I       1.35      1.65      0.187    HE3      ;    IID2        22
  I       1.98      2.82      0.214    HE4      ;    IID3        22
  I       2.29      3.24      0.221    HE4      ;    IID3        22
  I       3.72      5.42      0.227    HE4      ;    IID3        22
  I       2.26      3.41      0.206    HE3      ;    IID3        22
;
# 8   RATE
  R2    1.0E-05    1.0E+05          IILDI-(U)
  R2    1.0E-05    1.0E+05          IIL
  R2    1.0E-05    1.0E+05          IILZ2
  R2    1.0E-05    1.0E+05          IILZ3
  R2    1.0E-05    1.0E+05          IIA1H-(U)
  R2    1.0E-05    1.0E+05          IIAS
#END
```

```
ASCII
 2-D  V. 4.0  (xy)                                                    09/10/89
; temp1.dir            ISEE-3   8 Hr  Composition Data (trend checked)
;
1978 Aug 15 0000
1982 Jan  1 0000
0 0 0800
#0 Rate
T2          TIME ;
F2  .01  1010   Intensity
E2   1   100    Energy
S8   0   8      Name
;F2    1e-006     1000  1.10  1.34 HE4     ;   IID2        22
;F2    1e-006     1000  1.34  1.63 HE4     ;   IID2        22
;F2    1e-006     1000  1.35  1.65 HE3     ;   IID2        22
;F2    1e-006     1000  1.98  2.82 HE4     ;   IID3        22
;F2    1e-006     1000  2.29  3.24 HE4     ;   IID3        22
;F2    1e-006     1000  3.72  5.42 HE4     ;   IID3        22
;F2    1e-006     1000  2.26  3.41 HE3     ;   IID3        22
#End

0   1978 Sep 23 1400  1978 Sep 24 1808        39.6      1.38      1.10  1.34 HE4
     ;   IID2         22
0   1978 Sep 23 1400  1978 Sep 24 1808        41.5      1.24      1.34  1.63 HE4
     ;   IID2         22
0   1978 Sep 23 1400  1978 Sep 24 1808        3.12     0.402      1.35  1.65 HE3
     ;   IID2         22
0   1978 Sep 23 1400  1978 Sep 24 1808        14.2      0.48      1.98  2.82 HE4
     ;   IID3         22
0   1978 Sep 23 1400  1978 Sep 24 1808        37.2     0.718      2.29  3.24 HE4
     ;   IID3         22
0   1978 Sep 23 1400  1978 Sep 24 1808        28.6     0.464      3.72  5.42 HE4
     ;   IID3         22
0   1978 Sep 23 1400  1978 Sep 24 1808        1.25     0.124      2.26  3.41 HE3
     ;   IID3         22

0   1978 Sep 25 2332  1978 Sep 27 1348        36.2     0.418      1.10  1.34 HE4
     ;   IID2         22
0   1978 Sep 25 2332  1978 Sep 27 1348        24.4     0.301      1.34  1.63 HE4
     ;   IID2         22
0   1978 Sep 25 2332  1978 Sep 27 1348        2.03     0.103      1.35  1.65 HE3
     ;   IID2         22
0   1978 Sep 25 2332  1978 Sep 27 1348        5.44     0.094      1.98  2.82 HE4
     ;   IID3         22
0   1978 Sep 25 2332  1978 Sep 27 1348        7.98     0.105      2.29  3.24 HE4
     ;   IID3         22
0   1978 Sep 25 2332  1978 Sep 27 1348        2.46    0.0432      3.72  5.42 HE4
     ;   IID3         22
0   1978 Sep 25 2332  1978 Sep 27 1348       0.233     0.017      2.26  3.41 HE3
     ;   IID3         22

0   1979 Apr  3 1440  1979 Apr  4 0728        82.1      1.65      1.10  1.34 HE4
     ;   IID2         22
0   1979 Apr  3 1440  1979 Apr  4 0728        56.2       1.2      1.34  1.63 HE4
     ;   IID2         22
0   1979 Apr  3 1440  1979 Apr  4 0728        4.36     0.395      1.35  1.65 HE3
     ;   IID2         22
0   1979 Apr  3 1440  1979 Apr  4 0728        12.5     0.373      1.98  2.82 HE4
     ;   IID3         22
0   1979 Apr  3 1440  1979 Apr  4 0728        16.3     0.395      2.29  3.24 HE4
     ;   IID3         22
0   1979 Apr  3 1440  1979 Apr  4 0728        4.18     0.147      3.72  5.42 HE4
```

spect.dat

intensity I(err) E low Enp part

```
     ;   IID3          22
0   1979 Apr  3 1440  1979 Apr  4 0728        0.652     0.0743      2.26   3.41 HE3
     ;   IID3          22

0   1979 Apr  5 0812  1979 Apr  6 1844         2.28     0.0235      1.10   1.34 HE4
     ;   IID2          22
0   1979 Apr  5 0812  1979 Apr  6 1844         1.23     0.0151      1.34   1.63 HE4
     ;   IID2          22
0   1979 Apr  5 0812  1979 Apr  6 1844        0.111    0.00537      1.35   1.65 HE3
     ;   IID2          22
0   1979 Apr  5 0812  1979 Apr  6 1844         0.19    0.00393      1.98   2.82 HE4
     ;   IID3          22
0   1979 Apr  5 0812  1979 Apr  6 1844         0.19    0.00363      2.29   3.24 HE4
     ;   IID3          22
0   1979 Apr  5 0812  1979 Apr  6 1844       0.0264   0.000999      3.72   5.42 HE4
     ;   IID3          22
0   1979 Apr  5 0812  1979 Apr  6 1844      0.00585     0.0006      2.26   3.41 HE3
     ;   IID3          22

0   1979 Jun  6 1542  1979 Jun  6 2146          174       4.41      1.10   1.34 HE4
     ;   IID2          22
0   1979 Jun  6 1542  1979 Jun  6 2146          105          3      1.34   1.63 HE4
     ;   IID2          22
0   1979 Jun  6 1542  1979 Jun  6 2146         13.2       1.26      1.35   1.65 HE3
     ;   IID2          22
0   1979 Jun  6 1542  1979 Jun  6 2146         18.3       0.83      1.98   2.82 HE4
     ;   IID3          22
0   1979 Jun  6 1542  1979 Jun  6 2146         23.9      0.877      2.29   3.24 HE4
     ;   IID3          22
0   1979 Jun  6 1542  1979 Jun  6 2146         7.18      0.355      3.72   5.42 HE4
     ;   IID3          22
0   1979 Jun  6 1542  1979 Jun  6 2146         1.63      0.216      2.26   3.41 HE3
     ;   IID3          22

0   1979 Jun  7 2230  1979 Jun  9 1630         40.9       0.36      1.10   1.34 HE4
     ;   IID2          22
0   1979 Jun  7 2230  1979 Jun  9 1630           26      0.251      1.34   1.63 HE4
     ;   IID2          22
0   1979 Jun  7 2230  1979 Jun  9 1630         2.24     0.0873      1.35   1.65 HE3
     ;   IID2          22
0   1979 Jun  7 2230  1979 Jun  9 1630         4.87      0.072      1.98   2.82 HE4
     ;   IID3          22
0   1979 Jun  7 2230  1979 Jun  9 1630          5.8     0.0726      2.29   3.24 HE4
     ;   IID3          22
0   1979 Jun  7 2230  1979 Jun  9 1630         1.26      0.025      3.72   5.42 HE4
     ;   IID3          22
0   1979 Jun  7 2230  1979 Jun  9 1630        0.236     0.0138      2.26   3.41 HE3
     ;   IID3          22

0   1979 Jul  6 0108  1979 Jul  6 2044         25.7       0.33      1.10   1.34 HE4
     ;   IID2          22
0   1979 Jul  6 0108  1979 Jul  6 2044         15.5      0.224      1.34   1.63 HE4
     ;   IID2          22
0   1979 Jul  6 0108  1979 Jul  6 2044         1.32     0.0779      1.35   1.65 HE3
     ;   IID2          22
0   1979 Jul  6 0108  1979 Jul  6 2044         2.52       0.06      1.98   2.82 HE4
     ;   IID3          22
0   1979 Jul  6 0108  1979 Jul  6 2044         2.57     0.0561      2.29   3.24 HE4
     ;   IID3          22
0   1979 Jul  6 0108  1979 Jul  6 2044        0.385      0.016      3.72   5.42 HE4
     ;   IID3          22
0   1979 Jul  6 0108  1979 Jul  6 2044        0.115     0.0112      2.26   3.41 HE3
```

```
     ;   IID3         22

0   1979 Jul  7 0508  1979 Jul  9 0636        14.2      0.115      1.10  1.34 HE4
     ;   IID2         22
0   1979 Jul  7 0508  1979 Jul  9 0636        8.82      0.079      1.34  1.63 HE4
     ;   IID2         22
0   1979 Jul  7 0508  1979 Jul  9 0636       0.761     0.0275      1.35  1.65 HE3
     ;   IID2         22
0   1979 Jul  7 0508  1979 Jul  9 0636        1.71      0.023      1.98  2.82 HE4
     ;   IID3         22
0   1979 Jul  7 0508  1979 Jul  9 0636        2.07     0.0234      2.29  3.24 HE4
     ;   IID3         22
0   1979 Jul  7 0508  1979 Jul  9 0636       0.481    0.00833      3.72  5.42 HE4
     ;   IID3         22
0   1979 Jul  7 0508  1979 Jul  9 0636      0.0807    0.00436      2.26  3.41 HE3
     ;   IID3         22

0   1979 Aug 19 0646  1979 Aug 20 0922        23.1      0.345      1.10  1.34 HE4
     ;   IID2         22
0   1979 Aug 19 0646  1979 Aug 20 0922        16.3      0.253      1.34  1.63 HE4
     ;   IID2         22
0   1979 Aug 19 0646  1979 Aug 20 0922        1.19     0.0811      1.35  1.65 HE3
     ;   IID2         22
0   1979 Aug 19 0646  1979 Aug 20 0922           4     0.0832      1.98  2.82 HE4
     ;   IID3         22
0   1979 Aug 19 0646  1979 Aug 20 0922         6.9      0.101      2.29  3.24 HE4
     ;   IID3         22
0   1979 Aug 19 0646  1979 Aug 20 0922        3.66     0.0543      3.72  5.42 HE4
     ;   IID3         22
0   1979 Aug 19 0646  1979 Aug 20 0922       0.256     0.0183      2.26  3.41 HE3
     ;   IID3         22

0   1979 Aug 20 1814  1979 Aug 23 0118        27.6       0.28      1.10  1.34 HE4
     ;   IID2         22
0   1979 Aug 20 1814  1979 Aug 23 0118        18.8      0.202      1.34  1.63 HE4
     ;   IID2         22
0   1979 Aug 20 1814  1979 Aug 23 0118        1.56     0.0691      1.35  1.65 HE3
     ;   IID2         22
0   1979 Aug 20 1814  1979 Aug 23 0118        4.13     0.0628      1.98  2.82 HE4
     ;   IID3         22
0   1979 Aug 20 1814  1979 Aug 23 0118        6.13     0.0707      2.29  3.24 HE4
     ;   IID3         22
0   1979 Aug 20 1814  1979 Aug 23 0118        2.18     0.0311      3.72  5.42 HE4
     ;   IID3         22
0   1979 Aug 20 1814  1979 Aug 23 0118       0.255     0.0136      2.26  3.41 HE3
     ;   IID3         22

0   1979 Sep 14 2344  1979 Sep 18 0800        2.16     0.0188      1.10  1.34 HE4
     ;   IID2         22
0   1979 Sep 14 2344  1979 Sep 18 0800        1.55      0.014      1.34  1.63 HE4
     ;   IID2         22
0   1979 Sep 14 2344  1979 Sep 18 0800       0.101    0.00422      1.35  1.65 HE3
     ;   IID2         22
0   1979 Sep 14 2344  1979 Sep 18 0800        0.37    0.00452      1.98  2.82 HE4
     ;   IID3         22
0   1979 Sep 14 2344  1979 Sep 18 0800       0.621    0.00542      2.29  3.24 HE4
     ;   IID3         22
0   1979 Sep 14 2344  1979 Sep 18 0800       0.308    0.00282      3.72  5.42 HE4
     ;   IID3         22
0   1979 Sep 14 2344  1979 Sep 18 0800      0.0186   0.000882      2.26  3.41 HE3
     ;   IID3         22
```

```
0   1979 Nov 16 0244  1979 Nov 18 1812        6.98    0.0607     1.10  1.34 HE4
     ;   IID2         22
0   1979 Nov 16 0244  1979 Nov 18 1812        4.38    0.0421     1.34  1.63 HE4
     ;   IID2         22
0   1979 Nov 16 0244  1979 Nov 18 1812       0.386    0.0148     1.35  1.65 HE3
     ;   IID2         22
0   1979 Nov 16 0244  1979 Nov 18 1812       0.882    0.0125     1.98  2.82 HE4
     ;   IID3         22
0   1979 Nov 16 0244  1979 Nov 18 1812        1.12     0.013     2.29  3.24 HE4
     ;   IID3         22
0   1979 Nov 16 0244  1979 Nov 18 1812       0.264   0.00468     3.72  5.42 HE4
     ;   IID3         22
0   1979 Nov 16 0244  1979 Nov 18 1812      0.0403   0.00233     2.26  3.41 HE3
     ;   IID3         22

0   1980 Apr  4 1945  1980 Apr  7 1457        7.92    0.0478     1.10  1.34 HE4
     ;   IID2         22
0   1980 Apr  4 1945  1980 Apr  7 1457        3.45    0.0276     1.34  1.63 HE4
     ;   IID2         22
0   1980 Apr  4 1945  1980 Apr  7 1457       0.521    0.0127     1.35  1.65 HE3
     ;   IID2         22
0   1980 Apr  4 1945  1980 Apr  7 1457       0.387   0.00613     1.98  2.82 HE4
     ;   IID3         22
0   1980 Apr  4 1945  1980 Apr  7 1457       0.262   0.00466     2.29  3.24 HE4
     ;   IID3         22
0   1980 Apr  4 1945  1980 Apr  7 1457      0.0224   0.00101     3.72  5.42 HE4
     ;   IID3         22
0   1980 Apr  4 1945  1980 Apr  7 1457      0.0158   0.00108     2.26  3.41 HE3
     ;   IID3         22

0   1980 Jul 18 1440  1980 Jul 18 1920        44.8      1.51     1.10  1.34 HE4
     ;   IID2         22
0   1980 Jul 18 1440  1980 Jul 18 1920        28.4      1.05     1.34  1.63 HE4
     ;   IID2         22
0   1980 Jul 18 1440  1980 Jul 18 1920        4.45     0.495     1.35  1.65 HE3
     ;   IID2         22
0   1980 Jul 18 1440  1980 Jul 18 1920         5.9     0.318     1.98  2.82 HE4
     ;   IID3         22
0   1980 Jul 18 1440  1980 Jul 18 1920         9.4     0.372     2.29  3.24 HE4
     ;   IID3         22
0   1980 Jul 18 1440  1980 Jul 18 1920        3.66     0.171     3.72  5.42 HE4
     ;   IID3         22
0   1980 Jul 18 1440  1980 Jul 18 1920       0.547    0.0844     2.26  3.41 HE3
     ;   IID3         22

0   1980 Jul 19 1304  1980 Jul 21 1720        23.7     0.219     1.10  1.34 HE4
     ;   IID2         22
0   1980 Jul 19 1304  1980 Jul 21 1720        14.9     0.152     1.34  1.63 HE4
     ;   IID2         22
0   1980 Jul 19 1304  1980 Jul 21 1720        1.39    0.0552     1.35  1.65 HE3
     ;   IID2         22
0   1980 Jul 19 1304  1980 Jul 21 1720        2.89    0.0444     1.98  2.82 HE4
     ;   IID3         22
0   1980 Jul 19 1304  1980 Jul 21 1720        3.41    0.0447     2.29  3.24 HE4
     ;   IID3         22
0   1980 Jul 19 1304  1980 Jul 21 1720       0.721    0.0151     3.72  5.42 HE4
     ;   IID3         22
0   1980 Jul 19 1304  1980 Jul 21 1720       0.168   0.00932     2.26  3.41 HE3
     ;   IID3         22

0   1980 Oct 15 1756  1980 Oct 19 1132        4.08    0.0289     1.10  1.34 HE4
     ;   IID2         22
```

```
0   1980 Oct 15 1756  1980 Oct 19 1132        2.61    0.0202     1.34  1.63 HE4
     ;   IID2         22
0   1980 Oct 15 1756  1980 Oct 19 1132       0.215   0.00688     1.35  1.65 HE3
     ;   IID2         22
0   1980 Oct 15 1756  1980 Oct 19 1132       0.544   0.00611     1.98  2.82 HE4
     ;   IID3         22
0   1980 Oct 15 1756  1980 Oct 19 1132        0.71   0.00646     2.29  3.24 HE4
     ;   IID3         22
0   1980 Oct 15 1756  1980 Oct 19 1132       0.247   0.00281     3.72  5.42 HE4
     ;   IID3         22
0   1980 Oct 15 1756  1980 Oct 19 1132      0.0305   0.00126     2.26  3.41 HE3
     ;   IID3         22

0   1980 Nov 23 1723  1980 Nov 26 2347        8.52    0.0586     1.10  1.34 HE4
     ;   IID2         22
0   1980 Nov 23 1723  1980 Nov 26 2347        4.29    0.0364     1.34  1.63 HE4
     ;   IID2         22
0   1980 Nov 23 1723  1980 Nov 26 2347       0.517     0.015     1.35  1.65 HE3
     ;   IID2         22
0   1980 Nov 23 1723  1980 Nov 26 2347       0.623   0.00919     1.98  2.82 HE4
     ;   IID3         22
0   1980 Nov 23 1723  1980 Nov 26 2347        0.57   0.00814     2.29  3.24 HE4
     ;   IID3         22
0   1980 Nov 23 1723  1980 Nov 26 2347      0.0754   0.00218     3.72  5.42 HE4
     ;   IID3         22
0   1980 Nov 23 1723  1980 Nov 26 2347      0.0247   0.00159     2.26  3.41 HE3
     ;   IID3         22

0   1981 Apr  1 0813  1981 Apr  3 1929        12.7    0.0851     1.10  1.34 HE4
     ;   IID2         22
0   1981 Apr  1 0813  1981 Apr  3 1929        6.82    0.0547     1.34  1.63 HE4
     ;   IID2         22
0   1981 Apr  1 0813  1981 Apr  3 1929       0.734    0.0213     1.35  1.65 HE3
     ;   IID2         22
0   1981 Apr  1 0813  1981 Apr  3 1929        1.05    0.0142     1.98  2.82 HE4
     ;   IID3         22
0   1981 Apr  1 0813  1981 Apr  3 1929       0.947    0.0125     2.29  3.24 HE4
     ;   IID3         22
0   1981 Apr  1 0813  1981 Apr  3 1929        0.13   0.00341     3.72  5.42 HE4
     ;   IID3         22
0   1981 Apr  1 0813  1981 Apr  3 1929       0.049   0.00268     2.26  3.41 HE3
     ;   IID3         22

0   1981 Apr 10 2347  1981 Apr 14 0035        24.4     0.199     1.10  1.34 HE4
     ;   IID2         22
0   1981 Apr 10 2347  1981 Apr 14 0035        13.9     0.131     1.34  1.63 HE4
     ;   IID2         22
0   1981 Apr 10 2347  1981 Apr 14 0035        1.52    0.0516     1.35  1.65 HE3
     ;   IID2         22
0   1981 Apr 10 2347  1981 Apr 14 0035        2.49    0.0368     1.98  2.82 HE4
     ;   IID3         22
0   1981 Apr 10 2347  1981 Apr 14 0035        2.84    0.0364     2.29  3.24 HE4
     ;   IID3         22
0   1981 Apr 10 2347  1981 Apr 14 0035       0.765    0.0139     3.72  5.42 HE4
     ;   IID3         22
0   1981 Apr 10 2347  1981 Apr 14 0035       0.137   0.00754     2.26  3.41 HE3
     ;   IID3         22

0   1981 Apr 24 1652  1981 Apr 26 0228         140      2.12     1.10  1.34 HE4
     ;   IID2         22
0   1981 Apr 24 1652  1981 Apr 26 0228        95.4      1.53     1.34  1.63 HE4
     ;   IID2         22
```

```
0   1981 Apr 24 1652  1981 Apr 26 0228        11.3      0.625      1.35  1.65 HE3
     ;   IID2         22
0   1981 Apr 24 1652  1981 Apr 26 0228        19.2      0.456      1.98  2.82 HE4
     ;   IID3         22
0   1981 Apr 24 1652  1981 Apr 26 0228        24.8      0.478      2.29  3.24 HE4
     ;   IID3         22
0   1981 Apr 24 1652  1981 Apr 26 0228        7.09      0.189      3.72  5.42 HE4
     ;   IID3         22
0   1981 Apr 24 1652  1981 Apr 26 0228        1.28      0.102      2.26  3.41 HE3
     ;   IID3         22

0   1981 Apr 26 0900  1981 Apr 27 0752        57.1       1.26      1.10  1.34 HE4
     ;   IID2         22
0   1981 Apr 26 0900  1981 Apr 27 0752        42.7      0.951      1.34  1.63 HE4
     ;   IID2         22
0   1981 Apr 26 0900  1981 Apr 27 0752        4.17      0.352      1.35  1.65 HE3
     ;   IID2         22
0   1981 Apr 26 0900  1981 Apr 27 0752        10.3      0.309      1.98  2.82 HE4
     ;   IID3         22
0   1981 Apr 26 0900  1981 Apr 27 0752        15.3      0.349      2.29  3.24 HE4
     ;   IID3         22
0   1981 Apr 26 0900  1981 Apr 27 0752        5.03      0.148      3.72  5.42 HE4
     ;   IID3         22
0   1981 Apr 26 0900  1981 Apr 27 0752       0.719     0.0712      2.26  3.41 HE3
     ;   IID3         22

0   1981 May  9 0510  1981 May 10 1858        30.9      0.319      1.10  1.34 HE4
     ;   IID2         22
0   1981 May  9 0510  1981 May 10 1858        20.3      0.227      1.34  1.63 HE4
     ;   IID2         22
0   1981 May  9 0510  1981 May 10 1858        2.13      0.087      1.35  1.65 HE3
     ;   IID2         22
0   1981 May  9 0510  1981 May 10 1858        4.48     0.0705      1.98  2.82 HE4
     ;   IID3         22
0   1981 May  9 0510  1981 May 10 1858        6.48     0.0785      2.29  3.24 HE4
     ;   IID3         22
0   1981 May  9 0510  1981 May 10 1858        2.25     0.0341      3.72  5.42 HE4
     ;   IID3         22
0   1981 May  9 0510  1981 May 10 1858       0.311     0.0162      2.26  3.41 HE3
     ;   IID3         22

0   1981 May 11 2258  1981 May 13 0038        38.4      0.558      1.10  1.34 HE4
     ;   IID2         22
0   1981 May 11 2258  1981 May 13 0038        24.9      0.394      1.34  1.63 HE4
     ;   IID2         22
0   1981 May 11 2258  1981 May 13 0038        2.66      0.153      1.35  1.65 HE3
     ;   IID2         22
0   1981 May 11 2258  1981 May 13 0038        5.05      0.117      1.98  2.82 HE4
     ;   IID3         22
0   1981 May 11 2258  1981 May 13 0038        6.25      0.121      2.29  3.24 HE4
     ;   IID3         22
0   1981 May 11 2258  1981 May 13 0038        1.44     0.0428      3.72  5.42 HE4
     ;   IID3         22
0   1981 May 11 2258  1981 May 13 0038       0.321     0.0258      2.26  3.41 HE3
     ;   IID3         22

0   1981 May 16 1527  1981 May 17 1639        49.2       1.01      1.10  1.34 HE4
     ;   IID2         22
0   1981 May 16 1527  1981 May 17 1639        40.3      0.804      1.34  1.63 HE4
     ;   IID2         22
0   1981 May 16 1527  1981 May 17 1639        2.73      0.248      1.35  1.65 HE3
     ;   IID2         22
```

```
0   1981 May 16 1527  1981 May 17 1639        11.5     0.285      1.98  2.82 HE4
     ;   IID3         22
0   1981 May 16 1527  1981 May 17 1639        19.6     0.344      2.29  3.24 HE4
     ;   IID3         22
0   1981 May 16 1527  1981 May 17 1639        8.27     0.165      3.72  5.42 HE4
     ;   IID3         22
0   1981 May 16 1527  1981 May 17 1639       0.956    0.0715      2.26  3.41 HE3
     ;   IID3         22

0   1981 May 18 0707  1981 May 19 1903        9.97    0.0867      1.10  1.34 HE4
     ;   IID2         22
0   1981 May 18 0707  1981 May 19 1903        6.55    0.0615      1.34  1.63 HE4
     ;   IID2         22
0   1981 May 18 0707  1981 May 19 1903       0.552    0.0212      1.35  1.65 HE3
     ;   IID2         22
0   1981 May 18 0707  1981 May 19 1903        1.34    0.0184      1.98  2.82 HE4
     ;   IID3         22
0   1981 May 18 0707  1981 May 19 1903        1.68    0.0191      2.29  3.24 HE4
     ;   IID3         22
0   1981 May 18 0707  1981 May 19 1903       0.464    0.0074      3.72  5.42 HE4
     ;   IID3         22
0   1981 May 18 0707  1981 May 19 1903      0.0741   0.00378      2.26  3.41 HE3
     ;   IID3         22

0   1981 Jul 20 1854  1981 Jul 23 0322        7.71    0.0637      1.10  1.34 HE4
     ;   IID2         22
0   1981 Jul 20 1854  1981 Jul 23 0322        5.01     0.045      1.34  1.63 HE4
     ;   IID2         22
0   1981 Jul 20 1854  1981 Jul 23 0322       0.422    0.0155      1.35  1.65 HE3
     ;   IID2         22
0   1981 Jul 20 1854  1981 Jul 23 0322        1.02    0.0134      1.98  2.82 HE4
     ;   IID3         22
0   1981 Jul 20 1854  1981 Jul 23 0322        1.27    0.0139      2.29  3.24 HE4
     ;   IID3         22
0   1981 Jul 20 1854  1981 Jul 23 0322       0.312   0.00507      3.72  5.42 HE4
     ;   IID3         22
0   1981 Jul 20 1854  1981 Jul 23 0322      0.0534   0.00268      2.26  3.41 HE3
     ;   IID3         22

0   1981 Aug  9 1019  1981 Aug 12 0915          30     0.225      1.10  1.34 HE4
     ;   IID2         22
0   1981 Aug  9 1019  1981 Aug 12 0915        17.6     0.151      1.34  1.63 HE4
     ;   IID2         22
0   1981 Aug  9 1019  1981 Aug 12 0915        1.88    0.0585      1.35  1.65 HE3
     ;   IID2         22
0   1981 Aug  9 1019  1981 Aug 12 0915        3.19    0.0427      1.98  2.82 HE4
     ;   IID3         22
0   1981 Aug  9 1019  1981 Aug 12 0915        3.58    0.0418      2.29  3.24 HE4
     ;   IID3         22
0   1981 Aug  9 1019  1981 Aug 12 0915       0.793    0.0145      3.72  5.42 HE4
     ;   IID3         22
0   1981 Aug  9 1019  1981 Aug 12 0915       0.161   0.00836      2.26  3.41 HE3
     ;   IID3         22

0   1981 Oct  9 0239  1981 Oct 12 0835        15.4     0.119      1.10  1.34 HE4
     ;   IID2         22
0   1981 Oct  9 0239  1981 Oct 12 0835        10.4    0.0859      1.34  1.63 HE4
     ;   IID2         22
0   1981 Oct  9 0239  1981 Oct 12 0835       0.934    0.0305      1.35  1.65 HE3
     ;   IID2         22
0   1981 Oct  9 0239  1981 Oct 12 0835        2.35     0.027      1.98  2.82 HE4
     ;   IID3         22
```

```
0   1981 Oct  9 0239  1981 Oct 12 0835        3.29    0.0296      2.29  3.24 HE4
     ;   IID3         22
0   1981 Oct  9 0239  1981 Oct 12 0835        1.16    0.0129      3.72  5.42 HE4
     ;   IID3         22
0   1981 Oct  9 0239  1981 Oct 12 0835       0.134   0.00562      2.26  3.41 HE3
     ;   IID3         22

0   1981 Oct 12 0932  1981 Oct 13 1044        62.2      1.13      1.10  1.34 HE4
     ;   IID2         22
0   1981 Oct 12 0932  1981 Oct 13 1044        44.1      0.83      1.34  1.63 HE4
     ;   IID2         22
0   1981 Oct 12 0932  1981 Oct 13 1044         4.9     0.328      1.35  1.65 HE3
     ;   IID2         22
0   1981 Oct 12 0932  1981 Oct 13 1044        9.75     0.259      1.98  2.82 HE4
     ;   IID3         22
0   1981 Oct 12 0932  1981 Oct 13 1044        13.7     0.283      2.29  3.24 HE4
     ;   IID3         22
0   1981 Oct 12 0932  1981 Oct 13 1044        5.37     0.131      3.72  5.42 HE4
     ;   IID3         22
0   1981 Oct 12 0932  1981 Oct 13 1044       0.823    0.0655      2.26  3.41 HE3
     ;   IID3         22

0   1981 Oct 14 1856  1981 Oct 16 1516        25.3     0.312      1.10  1.34 HE4
     ;   IID2         22
0   1981 Oct 14 1856  1981 Oct 16 1516        18.2     0.231      1.34  1.63 HE4
     ;   IID2         22
0   1981 Oct 14 1856  1981 Oct 16 1516        1.41    0.0765      1.35  1.65 HE3
     ;   IID2         22
0   1981 Oct 14 1856  1981 Oct 16 1516        4.29    0.0744      1.98  2.82 HE4
     ;   IID3         22
0   1981 Oct 14 1856  1981 Oct 16 1516        6.33    0.0837      2.29  3.24 HE4
     ;   IID3         22
0   1981 Oct 14 1856  1981 Oct 16 1516        2.41    0.0381      3.72  5.42 HE4
     ;   IID3         22
0   1981 Oct 14 1856  1981 Oct 16 1516       0.268    0.0162      2.26  3.41 HE3
     ;   IID3         22

0   1981 Dec 11 1701  1981 Dec 13 0841        6.77    0.0614      1.10  1.34 HE4
     ;   IID2         22
0   1981 Dec 11 1701  1981 Dec 13 0841        3.47    0.0385      1.34  1.63 HE4
     ;   IID2         22
0   1981 Dec 11 1701  1981 Dec 13 0841       0.369    0.0149      1.35  1.65 HE3
     ;   IID2         22
0   1981 Dec 11 1701  1981 Dec 13 0841       0.513    0.0098      1.98  2.82 HE4
     ;   IID3         22
0   1981 Dec 11 1701  1981 Dec 13 0841        0.47   0.00868      2.29  3.24 HE4
     ;   IID3         22
0   1981 Dec 11 1701  1981 Dec 13 0841      0.0655   0.00239      3.72  5.42 HE4
     ;   IID3         22
0   1981 Dec 11 1701  1981 Dec 13 0841      0.0203    0.0017      2.26  3.41 HE3
     ;   IID3         22
```

```
 BINARY CREATED  1987 NOV 21 04:54
ISEE-3   8 Hr  Composition Data (trend checked)
;          82 1 1
78/ 8/15    0: 0: 0    ; START TIME OF FILE
87/ 2/ 7    0: 0: 0    ; STOP  TIME OF FILE
0    0    0800
;
; TIME OF FIRST CHAPT. IN FILE  78/ 8/15   0: 0: 0
;
#0  RATE
  T2                                  TIME ;
# 1   FLUX     7-HET I BSTP, HIGH GAIN
  R2    1.0E-05    1.0E+05           HET I   BSTP,HI RATE
  L     0            32767           COUNT ON HET I    BSTP,HI MATRIX
;TYPE   E MIN    E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  I      22.14    27.17     1.705   PROTON  ;   IB2          7
;
# 2   FLUX     9-HET I BST, LOW GAIN
  R2    1.0E-05    1.0E+05           HET I   BSTP,LO RATE
  L     0            32767           COUNT ON HET I    BSTP,LO MATRIX
;TYPE   E MIN    E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  I      30.01    45.09     1.598   HE4     ;   IL3          9
;
# 3   FLUX    12-HET II AST, HIGH GAIN
  R2    1.0E-05    1.0E+05           HET II  AST,HI  RATE
  L     0            32767           COUNT ON HET II   AST,HI  MATRIX
;TYPE   E MIN    E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  I       4.45     6.41     1.107   PROTON  ;   IIA2        12
  I       7.08    12.83     1.283   PROTON  ;   IIA3        12
  I       0.22     2.00     1.280   ELEC    ;   IIA2        12
  I       7.08    12.64     1.217   HE4     ;   IIA3        12
;
# 4   FLUX    13-HET II AST, LOW GAIN
  R2    1.0E-05    1.0E+05           HET II  AST,LO  RATE
  L     0            32767           COUNT ON HET II   AST,LO  MATRIX
;TYPE   E MIN    E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  I      24.44    47.86     1.270   C12     ;   IIA3        13
  I      12.71    24.76     1.283   N14     ;   IIA3        13
  I      24.76    45.52     1.279   N14     ;   IIA3        13
  I       9.50    13.01     1.273   O16     ;   IIA2        13
  I      14.11    24.27     1.283   O16     ;   IIA3        13
  I      24.27    45.08     1.282   O16     ;   IIA3        13
  I      25.33    46.07     1.283   SI28    ;   IIA3        13
  I      15.07    23.00     1.255   FE56    ;   IIA2        13
  I      24.99    45.56     1.283   FE56    ;   IIA3        13
;
# 5   FLUX    19-VLET I, EVENT TYPE 0
  R2    1.0E-05    1.0E+05           MEAN VLET  ET 0    RATE
  L     0            32767           COUNT ON VLET Mean  ET 0    MATRIX
;TYPE   E MIN    E MAX     GEOM     PARTICLE    MODENAME  EVENT TYPE
  I       1.90     2.82     0.259   C12     ;   MD2
  I       3.96     7.08     0.287   C12     ;   MD3
  I       7.08    12.67     0.279   C12     ;   MD3
  I       1.90     2.81     0.287   N14     ;   MD2
  I       3.97     7.02     0.287   N14     ;   MD3
  I       7.02    12.60     0.287   N14     ;   MD3
  I       1.90     2.80     0.287   O16     ;   MD2
  I       3.94     7.01     0.287   O16     ;   MD3
  I       7.01    12.53     0.287   O16     ;   MD3
  I       1.90     2.80     0.287   NE20    ;   MD2
  I       4.00     7.16     0.287   NE20    ;   MD3
  I       7.16    12.58     0.287   NE20    ;   MD3
```

comp8. hdr
example hdr
for attaching to
fsum or vert?
output file

```
  I        1.90      2.80      0.287    MG24     ;    MD2
  I        3.97      7.10      0.287    MG24     ;    MD3
  I        7.10     12.60      0.287    MG24     ;    MD3
  I        1.90      2.80      0.285    SI28     ;    MD2
  I        3.98      7.01      0.287    SI28     ;    MD3
  I        7.01     12.53      0.287    SI28     ;    MD3
  I        1.90      2.81      0.275    S32      ;    MD2
  I        4.02      7.10      0.287    S32      ;    MD3
  I        7.10     12.53      0.287    S32      ;    MD3
  I        1.91      2.80      0.269    CA40     ;    MD2
  I        3.96      7.13      0.287    CA40     ;    MD3
  I        7.13     12.61      0.287    CA40     ;    MD3
  I        1.90      2.80      0.287    FE56     ;    MD2
  I        4.01      7.11      0.287    FE56     ;    MD3
  I        7.11     12.55      0.287    FE56     ;    MD3
  I       12.55     24.03      0.287    FE56     ;    MD3
;
;# 6   FLUX    21-VLET II, EVENT TYPE 0
;
# 7   FLUX    22-VLET II, EVENT TYPE 1
  R2    1.0E-05     1.0E+05          VLET II ET 1    RATE
  L     0             32767          COUNT ON VLET II  ET 1    MATRIX
;TYPE   E MIN     E MAX      GEOM     PARTICLE     MODENAME  EVENT TYPE
  I       1.10      1.34      0.252    HE4      ;    IID2        22
  I       1.34      1.63      0.272    HE4      ;    IID2        22
  I       1.35      1.65      0.187    HE3      ;    IID2        22
  I       1.98      2.82      0.214    HE4      ;    IID3        22
  I       2.29      3.24      0.221    HE4      ;    IID3        22
  I       3.72      5.42      0.227    HE4      ;    IID3        22
  I       2.26      3.41      0.206    HE3      ;    IID3        22
;
# 8   RATE
  R2    1.0E-05     1.0E+05          IILDI-(U)
  R2    1.0E-05     1.0E+05          IIL
  R2    1.0E-05     1.0E+05          IILZ2
  R2    1.0E-05     1.0E+05          IILZ3
  R2    1.0E-05     1.0E+05          IIA1H-(U)
  R2    1.0E-05     1.0E+05          IIAS
;
#9 Alg
F2      .01       20              G4/G3-.05; 1.3 3He/4He
F2      .99       500             G5/F8;  2- 3  4He/O
F2      .099      6               F26/F8; 2- 3  Fe/O
F2      .99       1010            H1/H2-1.; >1.1  H/He
F2      .99       500             F8/F9;  O Spectrum
#END
```

absent from comp & at. flg header

INT1.LIS

```
ASCII
 Short intervals
78 aug 15 0000
85 jan  1 0000
#0  Rate
T2                Time
#End
0 1978 Sep 23 1400   1978 Sep 24 1808
0 1978 Sep 25 2332   1978 Sep 27 1348
0 1979 Apr  3 1440   1979 Apr  4 0728
0 1979 Apr  5 0812   1979 Apr  6 1844
0 1979 Jun  6 1542   1979 Jun  6 2146
0 1979 Jun  7 2230   1979 Jun  9 1630
0 1979 Jul  6 0108   1979 Jul  6 2044
0 1979 Jul  7 0508   1979 Jul  9 0636
0 1979 Aug 19 0646   1979 Aug 20 0922
0 1979 Aug 20 1814   1979 Aug 23 0118
0 1979 Sep 14 2344   1979 Sep 18 0800
0 1979 Nov 16 0244   1979 Nov 18 1812
0 1980 Apr  4 1945   1980 Apr  7 1457
0 1980 Jul 18 1440   1980 Jul 18 1920
0 1980 Jul 19 1304   1980 Jul 21 1720
0 1980 Oct 15 1756   1980 Oct 19 1132
0 1980 Nov 23 1723   1980 Nov 26 2347
0 1981 Apr  1 0813   1981 Apr  3 1929
0 1981 Apr 10 2347   1981 Apr 14 0035
0 1981 Apr 24 1652   1981 Apr 26 0228
0 1981 Apr 26 0900   1981 Apr 27 0752
0 1981 May  9 0510   1981 May 10 1858
0 1981 May 11 2258   1981 May 13 0038
0 1981 May 16 1527   1981 May 17 1639
0 1981 May 18 0707   1981 May 19 1903
0 1981 Jul 20 1854   1981 Jul 23 0322
0 1981 Aug  9 1019   1981 Aug 12 0915
0 1981 Oct  9 0239   1981 Oct 12 0835
0 1981 Oct 12 0932   1981 Oct 13 1044
0 1981 Oct 14 1856   1981 Oct 16 1516
0 1981 Dec 11 1701   1981 Dec 13 0841
0 1982 Jan 31 0008   1982 Feb  1 0244
0 1982 Feb  1 2300   1982 Feb  4 2156
0 1982 Mar  7 0630   1982 Mar  9 2158
0 1982 Jun  7 0545   1982 Jun 10 0441
0 1982 Jul 10 0241   1982 Jul 12 1600
0 1982 Jul 12 1752   1982 Jul 13 0848
0 1982 Jul 14 0644   1982 Jul 15 2156
0 1982 Jul 22 2047   1982 Jul 25 1215
0 1982 Nov 22 1813   1982 Nov 25 1517
0 1982 Nov 26 0648   1982 Nov 27 2324
0 1982 Dec  8 0455   1982 Dec 11 0351
0 1982 Dec 17 1057   1982 Dec 19 0429
0 1982 Dec 26 0348   1982 Dec 29 1356
0 1983 Feb  3 1456   1983 Feb  4 0908
0 1983 Feb  5 0320   1983 Feb  7 0736
0 1984 Feb 19 0349   1984 Feb 22 1357
0 1984 Mar 14 0512   1984 Mar 16 1312
0 1984 Apr 25 2057   1984 Apr 26 1249
0 1984 Apr 27 0249   1984 Apr 29 1049
```

NO EXCLUDED TIMES EXAMPLE; assume spectrum bin db's exist on PC

make = use editor

① - make 4 lists as intia.lis example of Don time lists for spectra

- make	V1Times.lis V1times.MD	V2times.lis V2times.MD	P10times.lis P10times.MD	P11times.lis P11times.MD
- direct creates	V1times.dir	V2times.dir	P10times.dir	P11times.dir

② -
- Fsum 4 V1times.dir V1db.dir V1templ.dat
  - make V1templ.hdr; use a template e.g. for V1db.dir edit I to L in verse description
  - make V1templ.MD
  - direct V1templ.MD → V1templ.dir
- Fsum 7 V1templ.dir V1db.dir V1temp2.dat (contains ≤ 4, 7 verses)
  - make V1temp2.hdr
  - make V1temp2.MD
  - direct V1temp2.MD → V1temp2.dir
- Fsum 18 V1temp2.dir V1db.dir V1temp3.dat (contains ≤ 4, 7, 18
- Fsum 20
- Fsum 22
- Fsum 24 - - - - - - V1temp6.dat (contains ≤ 4, 7, 18, 20, 22, 24 verse types)
- direct → V1temp6.dir directory for fluxes of V1

- [Repeat procedure for V2
- [Analagous procedure for P10
- [" " P11
- [

Header file edits could be set up at the same time time-lists are edited

MD file edits could also be set up ahead of time

All output file names systematically set up in advance.

Create 1 batch file to do all processing, assuming all HDR + MD files are available at start.

At conclusion of Fsum step, 4 "sub-databases" exist, 1 for each satellite. The individual db's contain verses needed in the spectra request. V1, for e.g., contains 6 verses:

Het I IAS, IBSP, Let A, B, C, D event types, plus a time verse indicating the spectrum start + stop time (?)

? What does the final FSUMMED Header contain for time (it makes proper header)

? Each spectrum requested time interval (for ex. 3 days) is fully contained/summed=averaged into 1 chapter of this "sub-database".
yes

Create a directory for each of these 4 sub-db's – (last step in FSUM segment for each satellite)

(3) run the VERT Mod to get actual spectra.

The VERT mod must handle ~ 60 bins (greater is desired)

If both Hets would be examined + Lets for one particle type.

Hets + Lets should go into different verses still. (be able to pick Het I only + Het or HET II only + Let, or perhaps Het I, II. with different symbols to check calibs.)

Yes this can be done if multi-satellite compares are needed

? Assign an interval number to each time period
? Where (at what processing step) is this done?

? Where is satellite name and time period preserved

VERT

V1temp6.dir → V1spect.dat
V2temp_.dir → V2spect.dat
P10temp_.dir → P10spect.dat
P11temp_.dir → P11spect.dat

existing software, if only redimensioned to handle more bins, would do this!

Edit the point.lis outputs of VERT as needed and rename them to

(4) Write software to do any background correction that may be needed, + run it

Make separate directory for the final spectral data files, one for each satellite

Use Tselect (to get quiet "selected" database)
Fsam (to get summing needed [for the software to generate 26 day avg times for ex.] → can be used to "exclude" if needed
vert (new version needed)
existing format databases
convert 3081 plot pgms to PC and/or Sun

dos bat system limitations:
2nd level command.com (not nice)

dos 3.3 + later
call clist %% no keywords ↑ can store into envir (like global)

extended batch (shareware one)
handles
do more than normal (may also cause software interaction glitches (as we experienced at encounter)
"dos"

PC Keyboard macro capabilities processors
"deskview" he also can multitask several things going on at once

quarterdeck $150 DESQVIEW → 386 is like a VM ← paging etc runs the machines in protected mode can access large amt of memory (he has 4M)
↙ a facility like windows
| Don has a copy

Frank Jones – prefers desqview

PcFLUX doesn't do means

(1)

IF "no excluded times"

(A) make up 4 lists of include times V1, V2, P10, P11

(B) assume 4 databases of (at least) standard Het & Let (V only) spectrum bins H, He [~19p, 17α Het] [4 Let p 4 Let α]

Het I, II double

P10, 11 18α, 14p (4 Let p special case (α not done))

A mod of Fsum to put out more than 1 verse type is needed

run Fsum w V1 list on V1 database → V1f
run " V2 V2 → V2f
run " P10 P10 → P10f
run P11 P11 → P11f

(C) 4 databases of summed bin data

— V1 Fsum 5 (A stp) int1.dir (→ times) comp8.dir (→ data) temp1.dat (→ output verse 5)
Het make temp1.MD (→ temp1.hdr file + direct temp file) →temp1.dir comp8.dir temp2.dat ≤ 5,7
Fsum 7 (B stp)

Let separately, for plotting? Let A, Let B, Let C, Let D all different et.

— Repeat for V2
— P10
— P11 — create a directory for each satellites data

(C)

VERT MOD is needed

Vert sat i. dir
select bins for spectrum
press CNTRL → data goes to point.lst

point.lst may need header edit

Vert mod would work something like FSum used in succession.
All V1 bins would get written to 1 verse = V1
2nd run could add V2. Etc.

Use multi panel overlay option of MACRO to make plots of 4 verses 1 verse (= 1 sat's spectrum) per panel -

Current prog. looses time-tags on plot

Voyagers
each Het std bins

4,11	IA 2	1	20 bins +20 Het I
	IA 3	5	
6,13	IL 3	3	
7,14	IB 3	3	
10,17	IP ~~4~~	2	
	IP ~~4~~	6	(9,16 Z omitted) (IPEAL omitted) (+6 more)
18	LA 2	~~4~~ 3	16 bins
~~19~~ 20, 23	LA 3		
24,	LD		

~ 56 bins p
~ 56 bins α

V1 & V2 data could be directioned together because the bins match - (what y some are missing relative to the other? (p10 in CROSS doc)

use 2D to get
list of quiet times
as demod by dow

Tselect works on PAT dbs

(Vert merge would be on a single pass

verse 0
verse 1 V1
verse 0
verse 2 V2
verse 0
verse 3 P 10 etc
P 10

Int 1

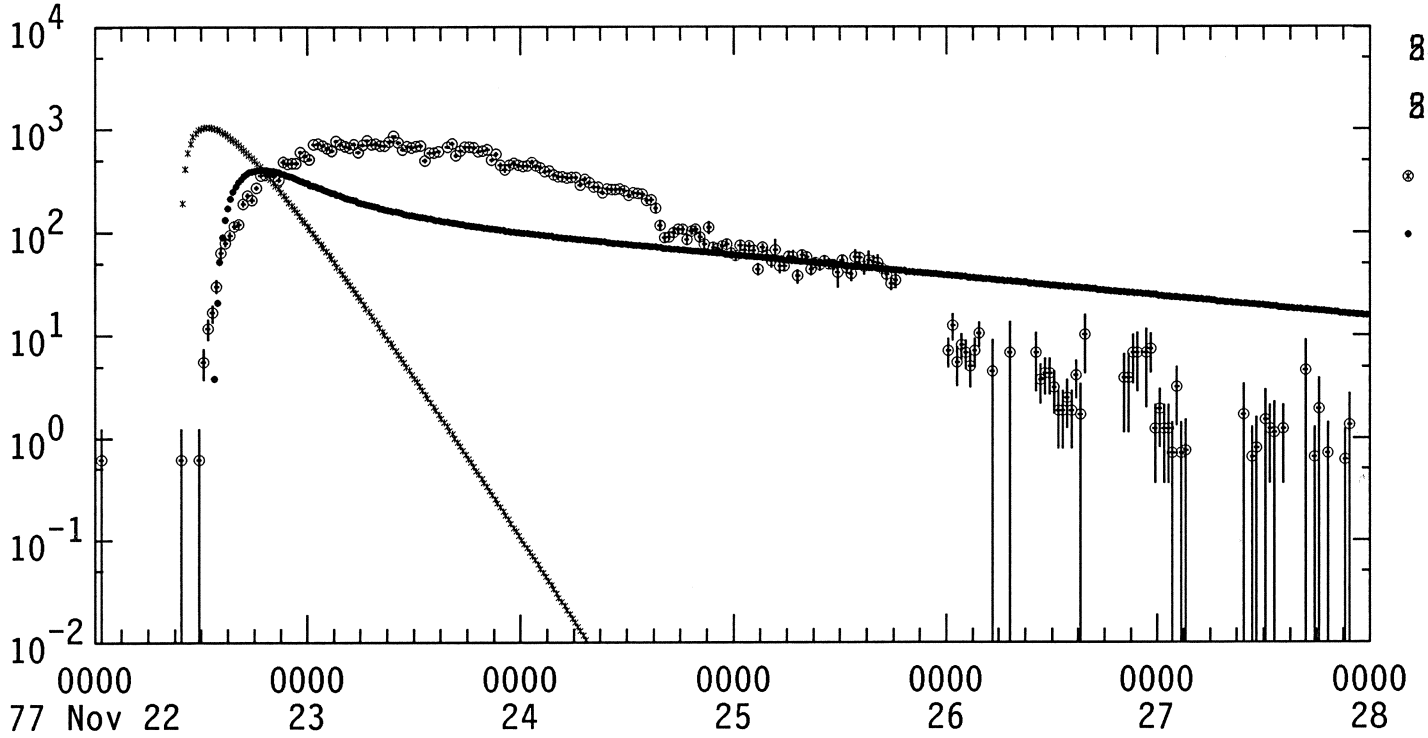
verse 0

Int 2

I8.DAT
Nov05.d2

IMP8 30-min Anisotropy Data
WHAT YOU WANT FOR 22/11/77 EVENT

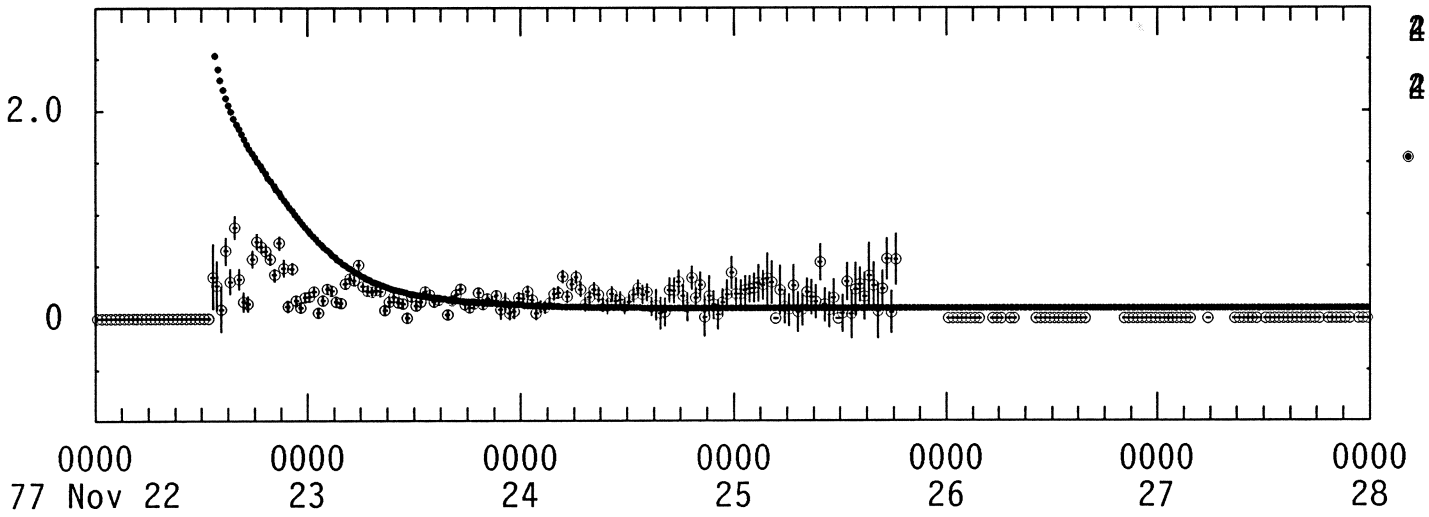
01/31/91
11:44:03 AM



888 events
888 shown
8000cm A1;
. Intensity at r=1.0 A

IMP8 He >1.7 MeV/n

2 panels overlaid



250 events
250 shown
A1 >1.7 p I8

He

2 panels overlaid

Vert mod (first guess)

Verse
T2 time
I Int #
E2 Energy

Verse 1
F2 Flux V1

Verse 2
F2 Flux V2

etc
3

4

5 Imp (Imp treated separately + added)

#End
N1 0 t1 t2 int energy1 energy2
1 flux1 err1 Σ spectrum Intervals 1 -> i

no way to put times into output of Cross in 2D

```
direct int1.md
direct int2.md
fsum      5  int1.dir  \isee\comp2.dir  temp1.dat
direct temp1.md
fsum      7  temp1.dir  \isee\comp2.dir  temp2.dat
direct temp2.md
fsum      5  int2.dir  temp2.dir  temp1.dat
direct temp1.md
fsum      7  temp1.dir  temp2.dir  bigabund.dat
direct bigabund.md
```

flare lists

verse 5 data for int1 times

temp1.dat

Σ of 5 + 7 data?

→ a longer time(s) which encompass int1 times (actually 2 larger s.t. intervals)

a longer timeSum was made from the time-pieces segregated by int1 timelist

bigabund.bat

~~comp8.dir~~ named file comp8.MD

int1a.md

direct int1a.md
" comp8.md

fsum int1a.dir comp8.dir temp1.dat

invert can select quant and ask for plot

invert select 8 bins

Cntl L gives listing file
Cntl C

data to point.lis → spect.dat

after edit

19Ø
19D
19E

---

do we have nands catalog. data hisths2
routine sources (catman)

get new backup of HISTHS2
ref. newflex stuff

Electron Responses to Voyager

= (FR80 → FT41 IMP RATEPLOT needs to be done)

= (never finished getting 26 day listing datasets for Tycho
for D6, 7, 8 (need to redate)

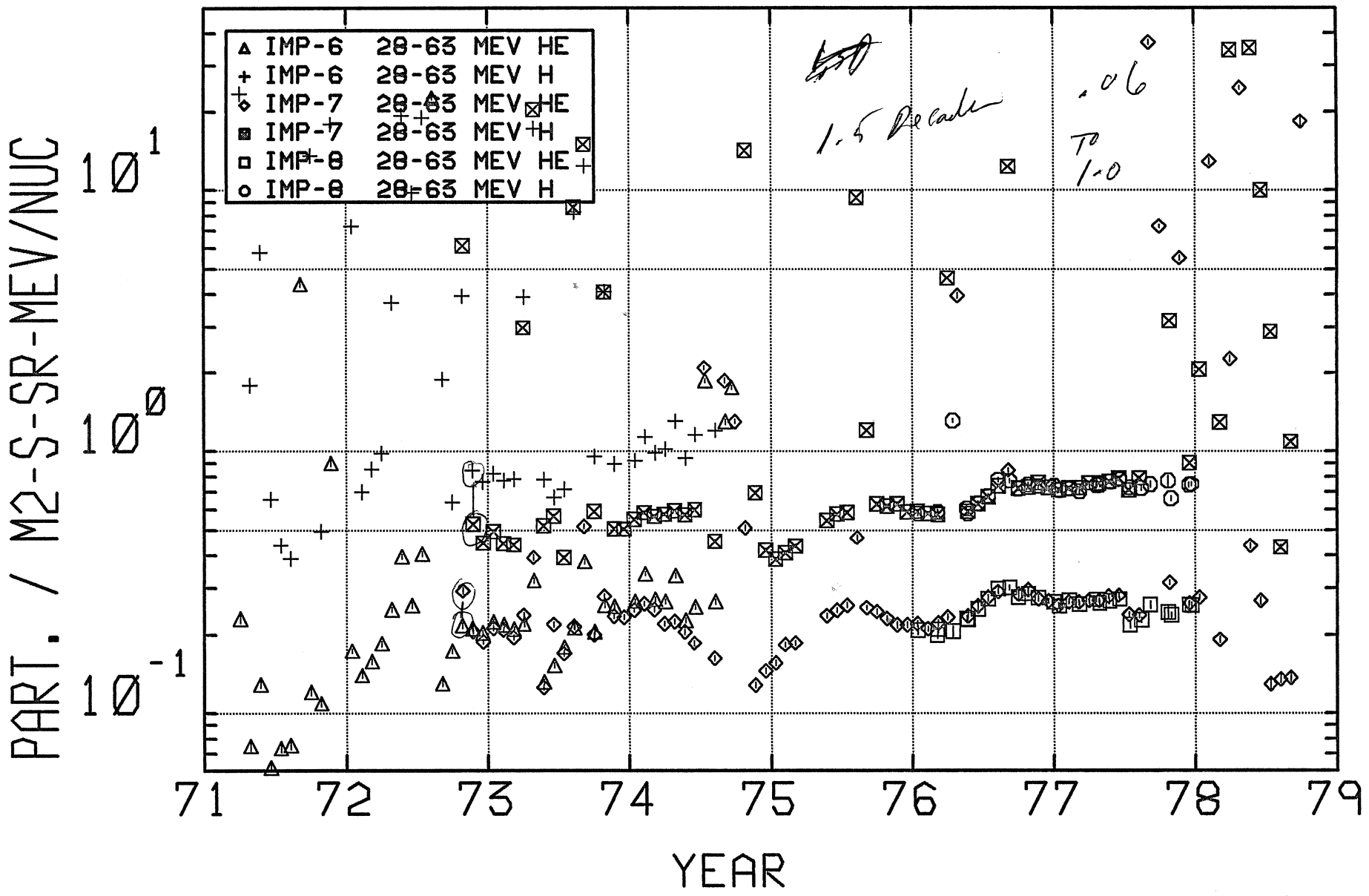
Voyager survey work FLUXTWOP version made
Plots made to plot
Voyager data periods where rates are weird for several hours
June - Dec 90
- Edrlist not run yet -

Fmcd requests + + ray. info PIUV plots in Heap John + me

Plots for Don to look at + 1 conversation -

IMP checkout group 1411
BMWRK for Isee didn't work → SHow changes

- FLXDBG compares ~ 1/3 done P-10

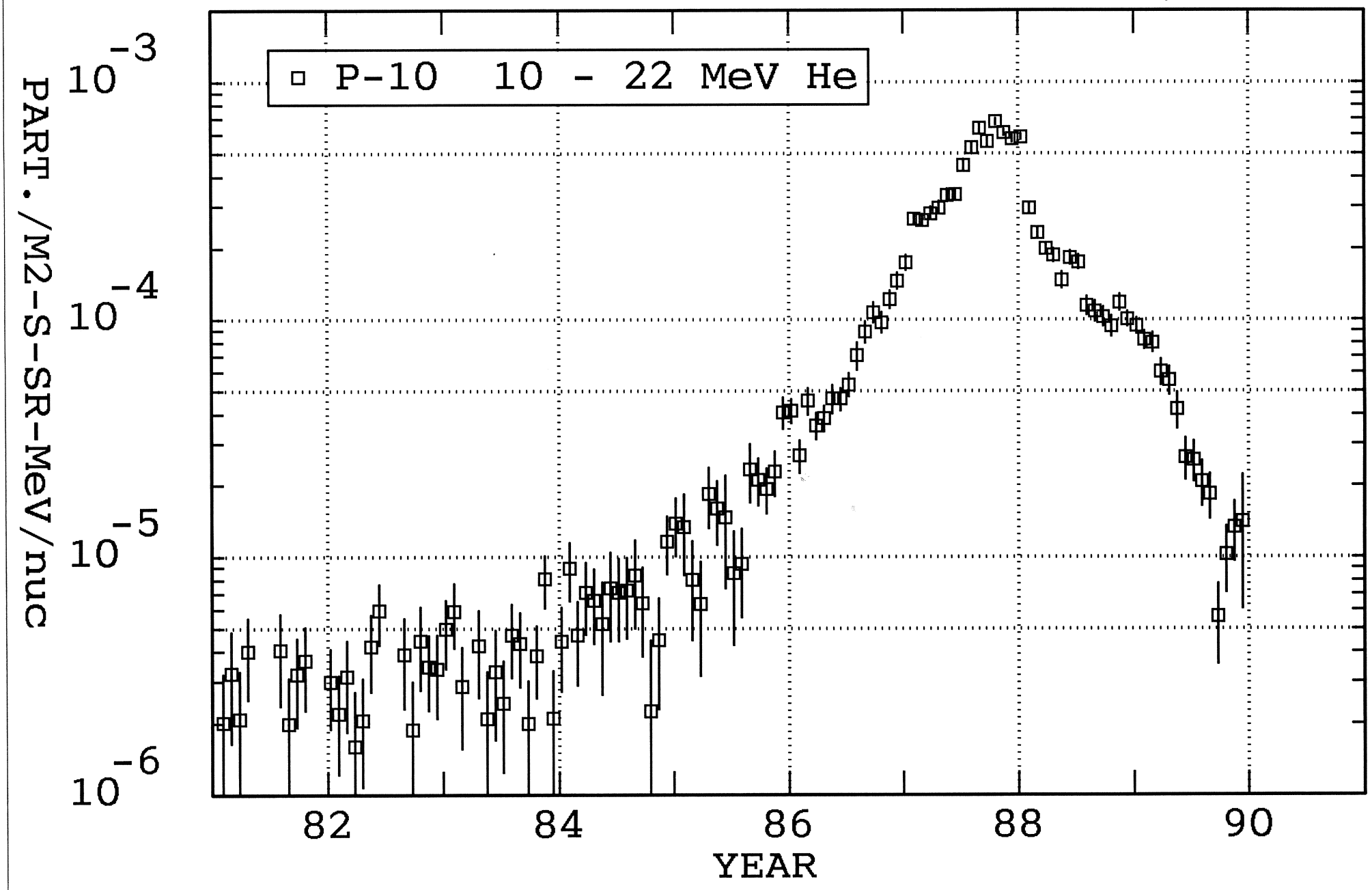

IMP-6 28-63 MEV HE
IMP-6 28-63 MEV H
IMP-7 28-63 MEV HE
IMP-7 28-63 MEV H
IMP-8 28-63 MEV HE
IMP-8 28-63 MEV H
1.5 Decade
.06
TO
1.0
PART. / M2-S-SR-MEV/NUC
10^1
10^0
10^-1
71
72
73
74
75
76
77
78
79
YEAR

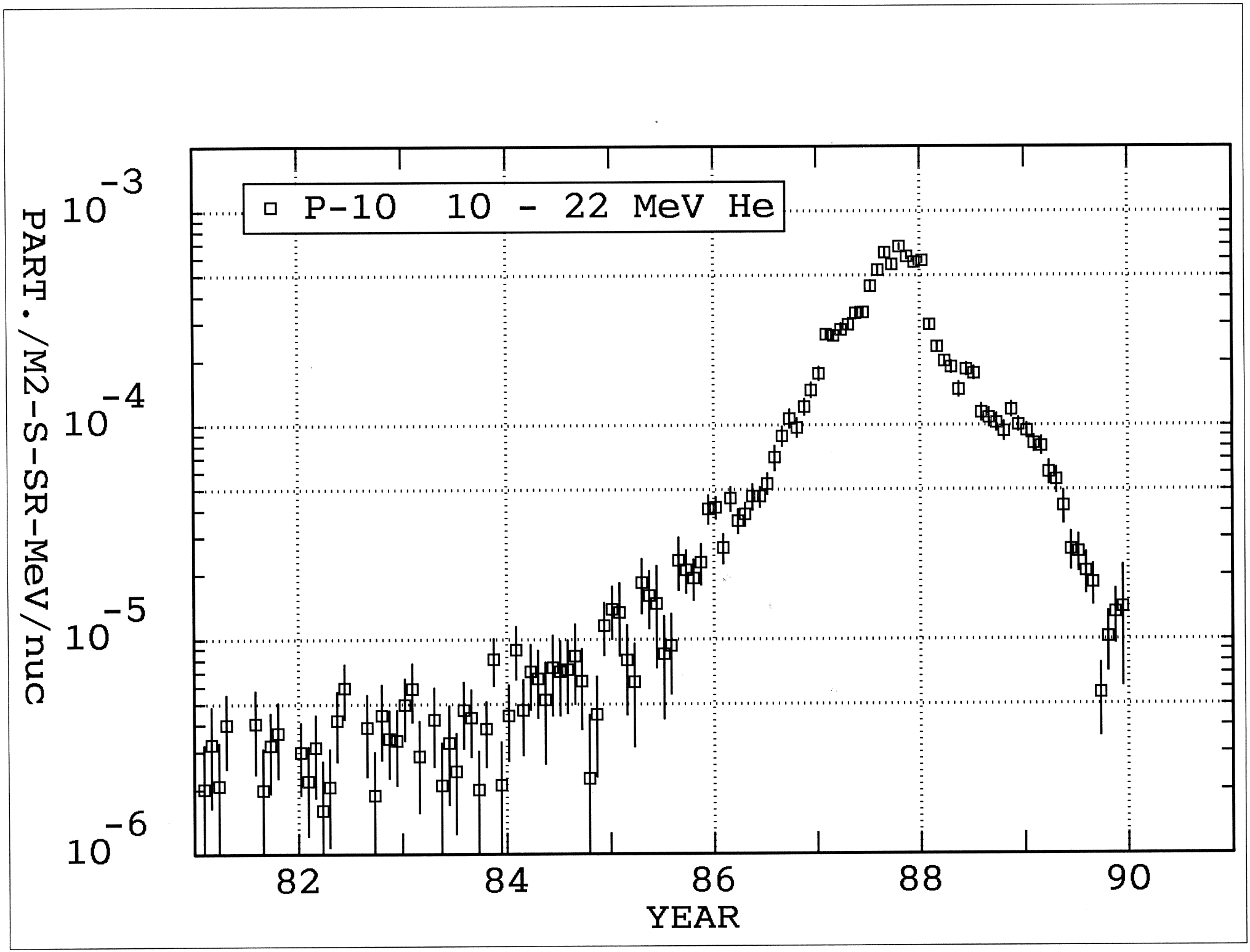
PCPHA Viet database 1973 IMPS

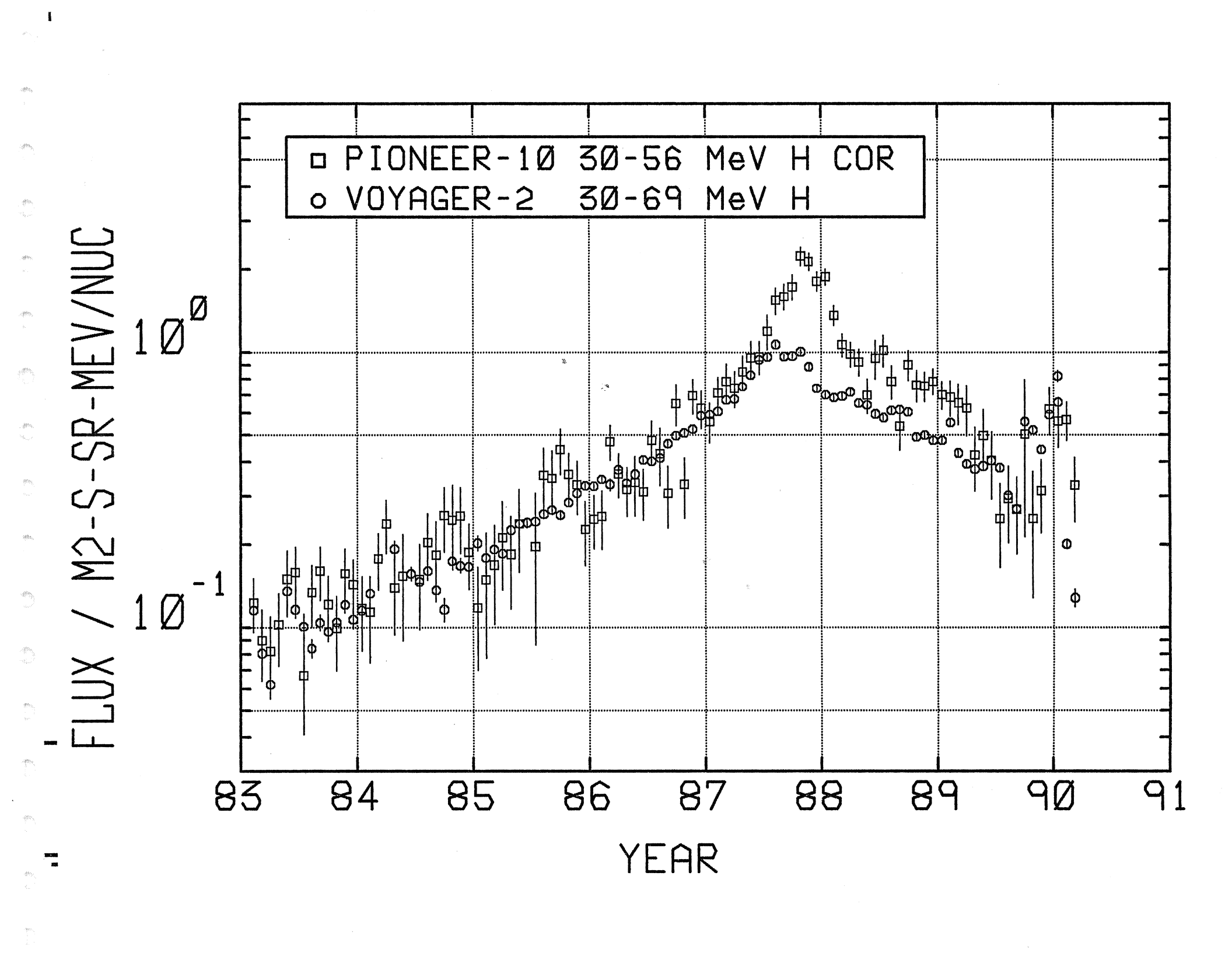
1974 180002 f 5 E 02763 5/16/74 - 6/4/74
RSeg 31

spb writes too many lines

Shift Prt Scrn made this from display at wait




P-10 10 - 22 MeV He
PART·/M2-S-SR-MeV/nuc
10-3
10-4
10-5
10-6
82
84
86
88
90
YEAR


PIONEER-10 30-56 MeV H COR
VOYAGER-2 30-69 MeV H
FLUX / M2-S-SR-MEV/NUC
10^0
10^-1
83
84
85
86
87
88
89
90
91
YEAR

textedit
modeless editor (comp. to vi)
insert between characters
delete char – "select" & push del
delete char to left del
to right shift del
word to left cntl w
to right shift cntl w
delete previous line cntl u
delete rest of line shift cntl u

reinsert last deleted (Get) or cntl-G

move "select" - del - (Get) at new "select" point
copy to shelf w/o delete (Put)

Bldg 2 grnd floor 3101 62666
S202 64887

root menu

/usr/lib/.rootmenu

icons /usr/lib/images

EDRs thrown out 1987 + frame →

ZL2589	0	
ZL4417	0	
ZK1905	3	
ZL7112	0	
DG8032	5	
DE9310	5	
DE8584	5	
DE3570	5	
DE0647	5	(CLOS 6 f 94)
ZT7376	5	56 f 60
ZT4316	5	56 f 61
ZU1196	5	56 f 62
YZ9284	3	56 f 15
ZJ7621	3	56 f 13
ZM1890	0	
ZL7144	3	
ZL6473	0	
DG7795	5	56 f 101
DG4123	5	
DH3620	5	56 f 105
DG8032	5	56 f 102
DG4222	5	" f 98
DG8741	5	" f 100
DH5331	5	" 104
DE6834	5	" f 96
~~DG8741~~		" f 100

To do

Swamy task Fsum/Vert
MCDROST member
PL3810
Cabinet checking for storage move
large box review

FT31 ← FT30 download to Pc → plot files → plot pgm
↓ ?
FT33 ←?

directory of spectra – [still needs to list for user to pick]
8 BINS max spectrum

FMCD May 3

[distN – ? trajectory 3 months / or ~~2~~ ends + middle for older style

[no boxes on plots

try
check code (+/f option) for call to box routines

---

diskettes

add fill segments

fool pgm by Setting for 3 array positions '3 years' and moding the contents of those 'year slots' with the trajectory data

(yes)
!

5 grid points

From tray listings

		P-10	P-11	U2	U1
1989	Jan 1	45.4	27.35	28.0	36.3
	~~April~~ 1	46.1	28.0	28.9	37.2
	July 1	46.7	28.6	29.7	38.1
	Oct 1	47.4	29.2	30.4	39.0
1990	Jan 1	48.1	29.8	31.06	39.9
	~~Apr~~ 1	48.7	30.4	31.8	40.8

P-10 1/31 45.6
6/2 46.5

P-11 1/31 27.56
6/2 28.39
6/11 = 37.9

FNCD:
Clust Fluxplot stabs

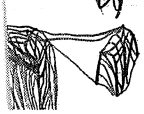
driver clust with day or date arguments (depends on SE cards whether can do for all)
(such as RZA job) (see encgen type too
day → dates

YDMD month & day from year + doy
(IYear, IDOY, IMONTH, IDAY)

YMDD (IYear, IMonth, Iday, IDOY)

for stop time, add 1 day before call
write single record out to dataset
IYear = 1989
DO I = 1, 999
Read (5, 5, END = 1000) INday, Iend] (or hard code)
5 Format (I3, 1X, I3)
Iend = Iend + 1
Call YDMD (
Call YDMD (
10 Format Write (6, 10) IYear, I1mn, I1day, I2mn, I2day INday, Iend
999 OD (10X, I2, 1X, 2(I2, 1X, I2), 1X, I3, 1X, I3
1000 Continue
Continue
STOP

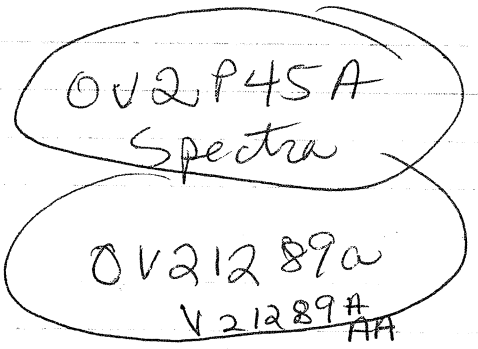
47.4
46.7



- TRAJ with SQ factor { finalize and run 6 month segments

- K? [can run 2 P-10 spectra (or ask K to) but need 1 V2 for comparison
  K could also do low energy daily ~~K~~ Spectra? of V2 1989
- ready CLIST submit for day spectra V1, V2
  - type in day ranges for conversion
  - make date conversion pgm
  - must include FT31,32 name pairs for FLUX run + write SI input list SET SE1 SE2 ? into driver clist provide for
  - make clist to edit prototype and submit job

~~V2P45A~~ V265A



$PLTRATO
↘ $RATIOS
↘ $$RATIOS

plot vs doyspec

see $DOYSPEC for template plot commandsetup

doygetts make plot datasets (edited) from FT31,32

set 52 } really values? will these equal, ?

SB#VG.LIB CNTL(ENXXXX)

~~How could~~ you have a last record with all entries full

intime needs to be set (set in getent?)

DTUPK(DMSEC, HUR, HMNTH, HDAY, HOUR, HMIN, HSEC

10, 11, 15, 6

104
4
416
52

Monday 2-8 PM

OWLTRead clust + mod
ENXXXX read over + clust
trajectory try 1
plot nobox mod x try 1
new FMCD request, failed - PENC14 missing
- tell John about preserving exclude times

10/1/87 - 12/5/87 1.40×10^{-5} (4.2×10^{-6}) 30-45

7/28/87 - 12/5/87 1.324×10^{-5} (4.3×10^{-6})

45-56

1.51×10^{-5} (4×10^{-6})

1.44×10^{-5} (4.5×10^{-6})

5/8

finished 5/7 req of FMCD P40 spectra

ran endxxx w Nand / need to add more points for V2

NuBox CLIST had wrong spelled parm.

couldn't find obvious error in trag. more study needed

cp termenal ch off (oo can use @)

mail al36@umaril.umd.edu (file x rpas opf787N

FM27

et12 #A3 0X16 V1

et11 #2 alpha V1

et13 #L3 0X16 V2

=8

i

2

2 (send)

Schuster

~~no pass~~

network name IBMNET

MPE

call NSSdca set 2 bit 2 goes down

set host NSSdca

Schuster ~~RECPL~~ PTLPLT

$ dr

$ sh quo

$ laser npg0012.dat

print " LCB ∅

laser_wide → 132 chars

ex to exit

ec edit Teco TPU FDL EDT

set dy [80 CGUIRE]

program input old values
output new values

↓

use 2 dataset derivatives
(doymrg input)

Read until eof (input)
Read " (bck)

write another 2 like derivative dataset
with background corrections

---

rerun doymrg / create "V" or recreate "u"

clust doybckgr driver to do correction
pioplot-like form

doymrgI2 → merge bckcor "2" type with
p datasets
to create "V" datasets

Zero arrays
Do I = 1, 4
Read (15, 10, END=999) elow(I) eup(I) partyp(I)
flux(I) err(I) num(I)
Read (16, 11) bck(I) bcker(I)

20 Continue
Do 30 I = 1, 4
oflux(I) = flux(I) - bck(I)
oerr(I) = SQRT(BKERR**2 + error**2)

---

link / bckcors

doygobck

doybckSp driver for background
bckcorsp

VAX edit

ereas eos E exit

2 or 3 interrupt

4 cancel

6 Show time

8 Softkey EDT key def now in effect?

$ Help newuser user

Can I do QED/CUST functions on PC (with edlin for e.g.)

Voyager edit/SORT

Use BOUNDS

SORT 20 25 1 10 2 sort fields (max=5)

↑ cols ↑

SORT 4 10 15 sort cols 10-15 in descending order

4289 recs after sort 1 field DESC M216

In 1 36 bit word

4 - 9 bit ascii chars
or 6 6 bit chars

Field data is 6 bit chars

4 chars in 3 bytes
1 char = 6 bits

"TRACK tape read with TRACK
took the 6 bit # & put into
the low order bits of an IBM
8 bit ~~byte~~.

Then TOEBCD worked

We need a pgm which takes
3 bytes & makes the 4 chars
logical*1 FIELD(3) L4(4), I4(4) integer*2 H1,
logical*1 inrec(8064)

```
100  Continue                                   eof
     Call FREAD (inrec, 10, len, &900, &910)
                                  IF len is not 8064 go to error
     nByte = 0
     DO I = 1, 2688, 3

          Call Kmove ( FIELD(i), INrec(I), 3 )
```

C have the 4 chars in 3 bytes at FIELD as so:

C: |CCCCCCDD|DDDDEEEE|EEFFFFFF| equivalence these HW, with field array
HW1 HW2

equivalence (I1, L1(4)) L4(1), L)

```
     I1 = 0
     I1(4) = FIELD(1)
     L = ISHFT (I1, -2)                    C into L
     nByte = nByte+1
     Call Fmove (OUT(nByte), 1, L(4) )
     I1 = 0
     I1 = HW1
     L = ISHFT (I1, -4)                    shift out E
     K = ISHFT (L, 26)                     shift out C
|0000CCCCCCDDDDDD|
     m = ISHFT (K, -26)                    D in lower byte of m
     nByte = nbyte+1
     Call Fmove (out(nByte), 1, m4(4) )
```

Stopped at 3584

OldVIOWL.read listing of VIOWLT before update

89/3/6 5:5:92 has zero OWLT VI
Others (so this a? pgm pblm

~~USET('HATCH')~~
~~USET('HTIMES', 5.)~~
USET('CROSSHATCH')

UPSET ('COLOR', 0.)
('GREY', 0.)

try setdash 18. → MHORIZ
VERTIC
↳ DOTLIN

~~UPSET~~ USET('WIDE')

Test T2 ('CROSSHATH') only + stddash 18.
T3 ('GREY', 0.) only

PG excludes June 14, 90 FMCD

30-56 rg

9/28/73
11/3/73 - 11/8/73
11/14/73
11/22/73

RIOD set

~~9/22/73~~
~~10/24/73~~
11/23/73 — 11/26

3-5 set

10/24/73 - 10/27/73
11/3/73 - 11/8/73

74 Jan

1/3/74
2/19 - 2/20
3/1
3/14
3/21
3/26
4/10
4/27
5/21
6/1

1/15
2/18 - 2/22
3/17
4/13
6/7 - 6/9
6/15
6/17

USET

~~USET~~ ('DIM') = 100%

USET ('BRIGHTNESS', 20.)

x, y coord of 1st vertex

U3BGFC	(x, y, z)	UBEGFC	(x, y)
iset fill	USET('SOLI')	UPSET('FCOL', 0.)	
UCRCLE	(x, y, radius) of center		
URECT	2 1 2	call URECT (x, y)	↗ diagonally opposite corner from current position

UENDFC

UPLYGN (x, y, sides, radius)
centered at ↑

[option to use polygons or old symbols or perhaps mix

(A) routine to establish marker size in DESCR
to replace USET SZmarker
USET 'WSYMBOL'

(X,Y) move the current position to X, Y

Umove
UPEN(X,Y) draw a line from current position to specified position current position is updated to (X,Y)

this ⇒ change symbol terminator to line terminator? (also consider connec option)

use Umove
then do polygon draw

○
□
△
◇ orientation angle

PLYDAT

Call PUTSYM (SYM(JJ), SFILL(J), XOUT, YOUT)

Call UBEGFC (X1, Y1)
USET ('SOLI')
UPSET ('FCOL', 2)
UCRCLE (X, Y, rad)
UPLYGN (X, Y, Sides, rad1)
UENDFC

try rad at ½ symsize for 1st

IF (connec) upen would need to be called to draw line but Symbol SZmarker must not be set
LSYMBOL, WSYMBOL not set

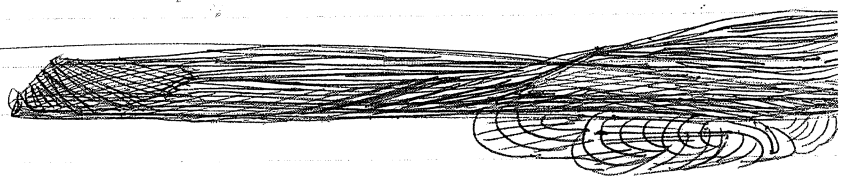
UPEN

linestyle default is 'LNULL' ← solid line
angular units 'DEGR'

'NNULL' absence of a terminator
blank is on display after call

UPSET('SYMBOL', 0.) = point line terminator

USET ('NNUL')



DESCR
PLYDAT

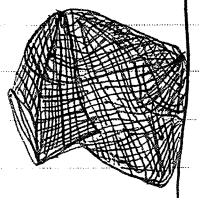
1/1/89 PLOT2 (PFLOIODN)
(PFLTP3
(PFLTP30
1/1/84 → end89
1/1/82 - end89

Common /SYMBL/ SYMSIZ, SYM(10), SYMBOX(10),
POLY(10) PFILL(10)

RATsym — logical switch array

IF POLY(I) is F use sizmarker method
is T use polygon method

Read SCAN record



turn ERT and SCET to DP mode
run timcheck code

IF time would be replaced by >tol

→ get OWLT

IF RE

IF PB

F = th in days

```
IF ( L_time == 'F' ) {
    read_inform();
    fscanf ( p_file, "%d", &iyear );
    printf ("%d \n", Lyear );

}
```

340
416
616 } statement missing
618

hard copy – no change – doesn't work – but display works, tho chars need work and chars in title box

Lad request
FMCD thru June 1 [one week breather] June 11 new FMCORR 2 matrices, [spectra, excl dates P10, P11

Add TRAJ to title Box in spectra also NOBOX
Bck cor to P10, P11 spectra (small pgm)
Spectra Chart submits
Made summary dataset MCD90AGU.data

Set up Sun
MCDREQST
↓
10/Voyager SORTS on SCANS / timechk
OWLT program TOEBPRE
Fill plot symbols / grid for dark on reduction
polygons

DMP TAPECART
EO5594 pblm
helped K w FLUXPLOT runs
data base time integrity
ON PC PL-3800 CGA plot
Ltime for day type plots

Don w FSann/VERT

ASK Nand about CIT tapes to fill gaps of n volumes.
Run Lselect ?

---

deliv — 7 mixed spectra final
— 4 He++ P times spectra
— K's 6 plots (see doyspc4)

work table — 3 plots 1979 data V1, P11, P10 full H spectra +AU
3 times
— runs for gt 2 new runs, 2 existing
— 9 plots 1/1/89 → to FMCD 5DMOV P10 R2A+R3A
Penn H V June 11
— distrop (2) plots ! spectra V2 comparing 2 long time
matrices P10, P11 2 + 2 June 11 Spectra
— spectra (3 plots 3 times) P10, P11 comparison H, He on same plot
P. listings excludes needed

Reader RDCCSEG @ of current segment + next word is its length
invoke RDSEGPRO R2 gets RDCCSEG and SEGHDR dsect (from FREAD)
invoke RDSEGSC1

Call CR5A here

Invoke RDIDMODE @ of mode attribute block from Reader
Call MARKUP (2) (3)

MATTRIB – MATDPI (=1 for CR5)
MATPRDQ (=# of DQSW)
MATDQ (=# DQSW)
MATMF (#MF segments)
TABLEHDR
SEGELMNT
SGDPIOFF
SGDQOFF
MFDIDPI

1st 7 basic entries in mode attribute

(A) LA 3,0(3,2) offset of first record @
Move DPI ~~DQE~~ DQSW into holding areas + input zero record areas
(B) Do for all DPI in this datamode (or temporary output area for debug)
put DPI into 8 bit format
~~Do for all~~
get offset for "old" location for this DPI
put into that location in output record buffer
OD

(C) Do for all DQSW in this datamode

COMMON/TABLEHDR/
COMMON/SEGELMNT/

determine segment as markup does
from segment map, get displacement + @ from mode attr. table
first DQI or DPI
then use SEGELMNT as dsect

table address will be in ENC@ mode datamode in ENCDMODE (or SEGDMODE after RIDIDMODE

byte word 1 bits 7-4 = 1 CRS Science
= 11 Engr (DQSW = zeros)

common /Header/ CR5A(

common /old way/ (or dsect an current encgen)

c New
c Record type byte 7 HW2 dec 24, 29
c 14 15 hex 18 1D
c DQSW word 15 31-16 for 8 mf
c disp = 15-0 "
c 54
c word 19 31-16 (HW1)
c for a total of 75 mf (?) gregn doc
c byte(high) = DQSW byte(low) = DQI
c DPI word 19 HW 2
c disp = 74
c
c W 19 HW 2 by 4 bits each MF 1 - 75
c goes thru word 28 bits 7-4
c spares WORD 28 bits 3-0
spares word 29 -> 59

Convert DQSW to 6 mf segments from 8 mf segments

CRSATTR says 60 mf / segment Segment size = 12 m 60
Seg = 4

DPI

DO For each HW
m = HW
K = 0 L = 0
LL = 0
m = HW
K = ISHFT (m, 4)
check shift direction L = ISHFT (m, -4)
LL = ISHFT (L, 4)

K = upper of HW
LL = lower of HW
move lower byte of these into output buffer
of correct address

one (DPI # * n + offset) for 1 byte

retrieve I # of DQSW/MF in old format 1/6 mf 10 bytes

DQ for N/MF DQSW of new FORMAT 1/8 mf 7 1/2

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
~~2~~	~~3~~	~~4~~	~~5~~	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

19	20	21	22	23	24	25	26	27	28	29	30
19	20	21	22	23	24						

4 old → 3 New

48	49	50	51	52	53	54	55	56	57	58	59	6
48												

old group # → new group # determined by

old	new
1	1
2	1+2
3	2+3
4	~~3~~
5	~~5~~ 4
6	~~5~~ ~~6~~ 4+5
7	~~6~~ ~~7~~ 5+6
8	~~8~~ 6
9	~~9~~ 7
10	7+8 (1st 8 part)

fake worst DQSW as DQSW

60 MF only (not 75)

DPI 4 bits = 0 if present 4 bits = 1 if missing

DQSW in upper byte

DQI in lower byte 7 - 5 unused

4 3 2 1 0

spare

54 DQSW HW 27 FW
74 DPI HW 37

?dump
12 words / line = 48

2nd half of word 14 = DQSW

13
4
52

New - 10 entries 8 mf / DQSW

0000 FFFF
00000F0F 0F0F

Reset record to zero. Ie make sure upper bytes are set to zero

1

75

24
6

30 F's mf 1. 30 times

LPG001
PLCV2032

~~m2834B~~
H2766C → m2834B → H2766C
face label

H13-24 FT20

H2766C should be on 23477
m2834B " " " 23476
+ tell John

~~F2476FRE 66 of~~ 4

RE B2712B 15036 66 of 5

timecheck
do not check Decoms?

record 215 FDSC was consistent scet was low but maybe OK

? 241, 3 should not be flagged ENGR

? 258 SDR " devFDSC was 313 LC

? 646, 8 "

702-710

766

1459

Record 241 Decom FDSC was consistent w ert, but scet was earlier by 114000 secs
& 243

Δscet was -365730 Δ FDSC was 613.18

47 & 316 258 SDR FDSC dev 313 LC Scet OK

646 SDR ΔScet = -7705758., FDSC 128433
~2½ hrs

this is probably PB

647 SDR same
648 ENGR

702 SDR ΔScet -92969.6 ΔLC = 1553.14

** note $\frac{\Delta scet}{\Delta FDSC}$ ~ 60 at 646 + 702 + 241

record 705 Decm ΔScet -1171168806 ΔLC 1553.14
record 710 Should be flagged Scet > ert

$\frac{\Delta scet}{\Delta LC}$ ~ 60 here, tho

IF (times(3) - scet) < 0 possibly PB

766 ΔLC 6906
ΔSCET 414629.55
note, time is greater, but not a lot greater

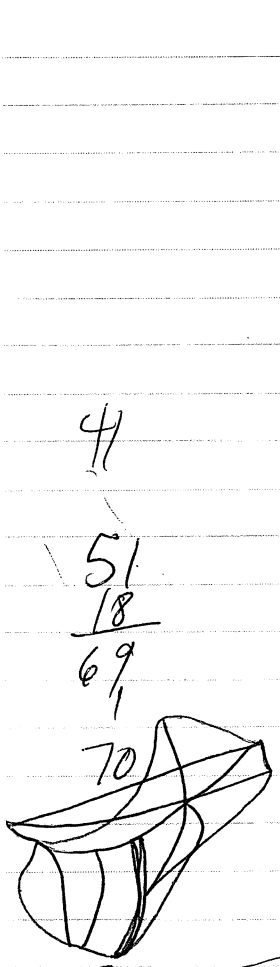
1459 ΔLC 116513
ΔSCET −6990,558.4

573 ΔSC 635,428 (10 min) ΔLC −10586.8
779 ΔSC −24569 ΔLC 413
1191 ΔLC 213.13
1340 ΔSC 36631 ΔLC 606.9
1817 ΔSC +97831.7 ΔLC 1626.9

Table of # between Δdev for gauge of Tol criterion

Add to EDITSCAN to see if it can get same ΔSC + ΔLC
then can do tables

? → |ΔLC| / 200 let it truncate
bins ± ½ hr segs for values up to 5 hours 100,000 LC or in ms
4 ± 15 min segs for 1 hr
19 ± 2 hr segs for values up to 1 day
1 ± 1 bin for values > 1 day



4-6 V background
matrix V
June 89

CRUV
Edutsan
FmCD 48+24 Indiv of (Let means) 6 times

P10U109
DBTime time handling pblm

TIMCHK handling of 44032 19 291 from Edetscan Test

CFDSC 7238654905.20

DTimes(1) 8405007491.00

175104
48000
1400832000
700416
8404992000

175104.32
48000 | 8405007491.00
48000
360500
336000
245007
240000
58074
48000
207491
192000
154910
144800
19100
96000
51000
48000
35000

~~DMOD = FDSC - FDSC MOD ( )~~

~~7238654905.20~~

8405007491.00 − 8404992000 =
8404992000
000015491

result of MOD of Times(1) = 15491
This is the LC value to < 1 m216

→ CFDSC = CFDSC − 15491
Temp = CFDSC / 48000
$Temp = 150804
.Dev = FDSC − (iTemp × 48000)
150804
48000
1206432000
603216
7238592000

7238654905.2
− 15491.0
7238639414.2

150804.98 → 150805

48000 | 7238639414.200
48000
243863
240000
386394
384000
239414
192000
474142
432000
421420
384000
374200

7238639414.2
7238654905.20
19785.8

15491 above
→ CFDSC = CFDSC + DMOD (Times(1), 4.8004)

7238639414.2
+ 15491
7238654905.2 agrees with substituted FDSC

but dev is quite different, being 47414.2

Solar wind
Liz Kennedy
-6695

Schardts het geom factors

P-10, -11 encounter data at NSSDC

P10JUF	~ Nov - Dec 73
P10JUG	~ Nov - Dec 74
P10SAG	~ Aug - Sept 79

John found

FILL ~~'EMPT'~~ ~~'SOLI'~~
hatch

Filled
used 'SOLI'

not filled
'EMPT'

upset
'FCOL'

	type	default		Upset?
'orientation'	P	0.	... looks possible	'ORIE'
'Xrotation'	P	0.		

yess, will rotate 4PLYGAN URECT, NCONIC
lower left corner rotation

rotate rectangle ~~45~~ 90°
triangle 180°

QEVCHK

SYM
SYMSIZ
CONNEC

Crudes dimension (12)
— unreachable stmt

8 Continue ?

203 not defined

Test 1 markers

4 F0
3 F0
1 F0

2

40 excps T10 T10 T10 } job FIL

2 F0
3 T10
1 F0

2 FØ
3 FØ
1 FØ } job NF1 20 excps

3

2 T1
3 T2
1 T3

template error 236
Integer exponent .LE 0

[uendFC needs args.

FT09 19,543 excps

Idena 6
HATCH

---

full acc 3016 template 254

T1
T1
T1 } 3022 F22

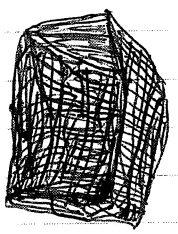
2
2 3023
2

3
3
3 3025

M ~~X~~ cards as are

looks like this worked

worked



doydr-ppp (add He)
doybWPFB has α & p
GB

make doybWVB
VZB ~~ptd~~

special let for α? P10/11

VINWbins
2

Aug 9 FMCD

modtime doybWPFC
GC
VIC
VZC
} α only stopping

modtime

doydrWB → ddsnabdoy
doydrWC → ddsnabdoyT

~ let α ranges
2D 1.9 - 2.8 5/3 1.7
2.8 - 8.1

naming convention for α let data will be: instead of LB, X, Y, Z, U
W L, ~~X~~, ~~Y~~, ~~Z~~, VL

doybWLBA
02LBA
bM1

doybm1LA
2LA

doyb1LBA
1LCA
2LBA 2LDA
2LCA

doydrive dist
is doydr2LA

made clists + submitted
He Stopping HeT for P10, 11, V1, V2 ~ 28 jobs
He het indw + mean for V1, V2 (7+6) x 4 jobs
V1 het proton redo 24 jobs
V2 revised
made ~~He~~ het data for P: have het in 2nd title card
+ reran clist doyget for aug3A branch
doygeb2L het only
* need to do aug3 + copy into MCD90AGU

doygeALP for alpha het only spectrum datasets
V1, V2

doy get α a for Het alpha
ran

Part 1 of FMCD: plotting 3 Lets + Het
$PLOTHET does p, α both

Todo:
make P10,11 He dataset clist

Voyager		Pioneer	
lets, α,	QWL&dsn&doy	α PPGA&doy	→ u?
let, p,	QLW &dsn&doy	p UPG&doy	
Het, α,	P&dsn. a&doy		
Het, p,	P&dsn&doy		

may have Voyager plot problems — check carefully.
Needed to regenerate V1 proton let for new geom
doydraw2L

Editscan
Bydel / engr ; summary tables / talk w Rand / count mf thrown out
SCAN - add replace values to p.01

FMCD V1+2 Matty runs I#2
revised Lat geom factors 32 tables
ReadFT41 program + 8 plots I8 6hr avgs 1988→
V2 new Het runs, p data only made + plotted [24 V2 + 24 V2/P11]
Aug 9 request begun 80 (28+52) FLUXPLOT runs +24 more (V1 p redos)
rigidity added to plot pgm

polygons are going "fill" doesn't work yet overlap isn't hatched

small work on 3081→PC planning

deliv 12 p only, p+α P10-1.
1a p only, α only Lat mea +Het V1, V
6 p only V2 indiv Lat
" V1 "
6 He V2 "
6 He V1 "
26 listings
12 Rigidity plots

Retrowrite part of CRRV5A
Aug 22 request 28 Fluxplot jobs
I8 QT spectrum 1989 p+α

plot pioneer data
p, α, ox spectra P10,11, V1-2
3 times × 4 sats
1 time V2, P10
4 ox plots
4 p only plots
4 α only plots
14 listings

~~$PLOTLEP~~ two datasets NO
$$SPEC38D $PLOTLEP
(multiple particles)
Aug 9A of doyspc4
8088
8092 test plots time#1

test plots also for #1
for Aug9 of doyspc4

F T
.5 0

Char*1 PATERN
Char*11 SATELT
equiv (SATELT, PATERN(1))

redone Lets: U-1
LA V102ST 1 count 3-5 MeV other 3es
LZ "

LD
1.8-3.1
3.4 .5.4
5.4-7.7
(7.7-8.1)

LC 3.4-5.5
5.5-7.5
(7.5-8.1)

V1, V2 similar energy range in upper bin 320 cts in LZV2032T

U-2 reality bins:
LC 1.8-3.2
3.3-5.6
5.6-7.4
7.4-8.6

L1 vs L2
L1 vs L3
L2 vs L3

L4 threshold

LD 1.8-3.3
3.3-5.6
5.6-7.4
7.4-8.04

Provide a Clist to users which will allow option of specifying delete HSM backups. Then have the Clist go to HSM and delete them.

Provide a Utility which will dust the dataset names in HSM in such a way that they could be a driver for a version of HBdelete Clist which would allow the user to delete all unwanted HSM backups

---

Rigidity Plots

P/He
3/3/89 – 3/9/89 P-11 set has 2 straight lines as the points with huge error -

P/He
5/1/89 – 5/11/89 P-11 set has 1 line like above

Monthly, Aug 90

Edtscan "scan" p.o. changes Nond requested

V2 Matrices (7) #stp
V1 " "
Het geom factors 32 hot tables revised

Talk w Bob

~ 300 Mbytes 2 sides read only one at a time

- present program capability
- functions that neither PC or 3081 support
- fractional charge plots in multiparameter environ
  3 parameter analysis –
  (higher priority than conversion of 3081 stuff)
- can cross do excludes
- use TMSUM to get at finegains maybe not
- gain tables to PC ? maybe not

~~can don't handle excludes~~

FLUXTNN uses DDATE + BASEYR → RMJDD
lines 156 – 163

looks like mod/new of RMJDD is needed

IMP-7 run

```
Do J=1,35
   Do I=1,12
   HD(I,J), = HD(I,J) + HD71(I)
```

expand HD to include 1 more year

.set password/Generate
answag
@KF$ COPELAND

72 3 4 5 6
365

(2192)

times lists of times

2 pass use of FSUM
1st pass set up like we do IMP now
2nd pass uses standard data 26 day time boundaries for time inputs

TSelect ___ does FSum like function currently for all verses

Blank Spacer Page

AN INTRODUCTION TO TIMPLT

D. V. REAMES
6/1/80

The TIMPLT program is an interative graphics program for the study of time-ordered data via those Vector General (VG) graphics unit on the PDP 11/70. The program will handle any properly formated data files in either of the two circa-1980 formats. Programs exist to generate properly formated data files from output of the Voyager, ISEE and IMP flux programs on the IBM 360 and from the ISEE data-pool tapes.

TIMPLT is a second-generation version of the FLXPLT program written by T. Conlon. In comparison it has been enlarged to permit multiple data files and the simultaneous presentation of multiple time histories (e.g. several particle species) on either of two arbitrarily-scaled plot panels and also to present multiple energy spectra at a given time on a separate panel. Many of the FLXPLT features are retained however the user interface is drastically altered.

In this next section we define the constructs recognized by TIMPLT and give examples of the commands that manipulate them. Details of the command syntax are presented later. The following descriptions assume some familiarity with the 11/70 and VG that can be acquired by demonstrations.

A. Concepts

The TIMPLT program recognizes and manipulates three different entities; files, plot panels and data elements as follows:

1) Files

Files are referenced by capital letters A and B (and possibly more). These letters are associated with the standard predefined data files (named by the 11/70 conventions) by typed command on the VG keyboard.

FILE A = RD1: [200, 122] V112HRAV.FLU

or

FILE B = [200, 105] POOL.DAT

If a valid file exists, a numerically ordered list of quantities on the file will appear (see Fig. 1). The lists may be reaccessed at any time by typing

File A

or

File B

A file may be redefined at any time. The program requires that at least one file be defined before any other commands are allowed.

2) Panels

A panel consists of a plotable box with tic marks, vertical and horizontal scaling and coarse and fine grid lines. Three (or more) panels are defined, two t histories and one (or more) spectrum. The command

PLOT 1

will display the first panel (similarly for 2 and 3).

PLOT BOTH

will display panel 1 over panel 2.

Default scaling of the time axes is taken from the first file entered and attempts to span the whole file.(or the first 240 points, whichever is less).

The scaling parameters are displayed by typing

SCALE 2

(see Fig. 2). This display may be cursor edited using

EDIT 2

When the fields have been changed as desired carriage-return will enter the new data or pressing button 2 will abort the process, leaving the old scaling unchanged. (Standard VG features use FS and BS to move within a field and → to change from field to field.).

The physical size of the plot X SIZE and Y SIZE should not exceed 4000. If the Y SIZE of plots 1 & 2 are too large, the plots will overlap.

The row of buttons available when plots are displayed are toggles that display or omit:

button	function
8, 16	box and tic marks
9, 17	coarse grid
10, 18	fine grid
11, 19	vertical scale
12, 20	horizontal scale
13, 21	title
14, 22	elements (plotted points)
15, 23	elements titles
7	Panic button - aborts program

When both panels are displayed buttons 8 - 15 affect panel 1 and 16 - 23 affect panel 2.

3) Elements

An element consists of a single entity or ratio of two entities available as a function of time (or a function of energy). Referring to Fig. 1, an element could be defined as A45/A26; the Fe^{56}/O^{16} ratio in the 2.0 to 2.9 MeV/nucl. interval as a function of time. A spectral element would be defined as all energy intervals with the same species name at a specific time. It would be distinguished from a time history element by being assigned to panel 3.

Elements are referenced either by E1, E2, E3 . . . or by lower case letters a, b, c. An element for 3 properties; 1) the panel on which it will appear, 2) the file data it contains and, 3) the symbol to be plotted.

An example of element definition might be

a = P1, A45/26, '*'

The first and last property may be omitted - they will be defaulted to P1 and 'a'. Once defined any of the above properties may be changed, e.g.

a = 'R'

will change the symbol (the symbol 'blank' evokes a histrogram plot). If more than 1 property is specified they must be in the order given above. The delimiters may be either commas or blanks (or both). The current definition of an element may be found by typing the elements name.

When a ratio is specified, both quantitites must come from the same file. For spectral ratios, the file is scanned for pairs of entries with 8 character names the same as those specified (e.g. A45 is 'FE56' and A26 is '016') and with upper and lower energies that are equal within tolerance (presently 5% and 20% respectively).

Presently, 8 elements may be defined.

B. Operation

The program is evoked by typing

RUN [200, 105] TIMPLT$

where $ is the escape key. All subsequent operations take place at the VG console and keyboard.

Commands are typed on the bottom cursored line on the VG and entered, when complete, using the carriage return key. A prompt appears initially to show that a file name must be entered, also appearing is a brief list of commands that can be reaccessed with the HELP command--this list will always be current.

After a file name has been entered, commands may entered in any order. The input syntax is designed to be very flexible. Generally a command may be abbreviated to as few as 2 letters and may consist of any combination of upper and lower case. Blanks may be inserted liberally between fields but not between digits of a number or letter of a command.

Generally, letters may be typed in either upper or lower case with the exception of:

1. File letters must be upper case (e.g. File B or A45)
2. Element symbols must be lower case letters a, b, ... or the or the upper case alternative E1, E2, ...

Once plots have been created you can advance by one plotted time interval using the NEXT command and back up using the BACK command. If panels 1 and 2 have the same time interval, both will be moved if either is.

The object field of the PLOT, SCALE, EDIT, NEXT or BACK will default to the current panel or scale (if one is displayed) or to panel 1. The latter two commands will not operate unless panels are displayed.

The commands NEXT and BACK return their own names rather than a blank line after successful execution. This permits a rapid sequence of foreward and backward moves simply by pressing carriage-return and is especially useful for spectral plots.

The program may be terminated via the STOP command or button 7.

The system will permit plotting many points on a few elements or a few points on many elements. Restraint should be exercised on the number of panels and plotted points since the VG system does not always fail gracefully.

Appendix A. A Terse Guide to the Programming

For those who wish to alter the code, all Fortran files are specified in [200, 105]FORTIM.CMD. Task building is controlled by TIMPLT.CMD in the same library, and overlay control by TIMPLT.ODL.

The program subroutines intercommunicate primarily by 3 COMMON blocks: PLOTCM (FLXCOM.FTN) contains the current status of all files, panels and elements; CONTR (FLCMD.FTN) is used in command and control language processing; PBUF(BUFR.FTN) is a buffer used during element plotting.

The program nucleus, TIMPLT, only monitors buttons and awaits commands, major sections of the program are:

1. COMMAND - interprets typed commands
   READEL - interprets element specifications

2. REPLOT (with RESTAT, REPLEL, RESTEL, ALLOFF and BOTH) supervise the status, creation and destruction of plots.

3. NEWPAN - supervises panel plotting
   BOX - plots the Box and grids
   FSCALE - labels time histories
   ESCALE - labels spectra
   TIC - automatic selection of tic mark scaling

4. DRAWEL - draws time history elements
   DRAW3 - draws spectral elements

5. NEWFIL (& LISTFI) - open files and create file display

6. SCALE (& SCEDIT) - supervise display and edit of plot scale parameters

Other subroutines are essentially utilities except CINIT and PINIT which provide command file and panel initialization (and default conditions).

FILE A REAMES1.DAT

0 6: 0: 0 AVERAGES FROM 1979 41 0: 0: 0 TO 1979 43 12: 0: 0

```
A  1  = 4.404E 00 - 6.408E 00 MeV PROTON   FLUX        (MEAN -(IA2 ),(IIA2))
A  2  = 6.496E 00 - 1.335E 01 MeV PROTON   FLUX        (MEAN -(IA3 ),(IIA3))
A  3  = 1.527E 01 - 2.259E 01 MeV PROTON   FLUX        (MEAN -(IA3 ),(IIA3))
A  4  = 2.343E 01 - 5.534E 01 MeV PROTON   FLUX        (MEAN -(IA3 ),(IIA3))
A  5  = 4.456E 00 - 2.223E 01 MeV DEUTRON  FLUX        (MEAN -(IA3 ),(IIA3))
A  6  = 1.345E 00 - 1.693E 00 MeV HE3      FLUX        (MEAN -(ID2 ),(IID2))
A  7  = 4.523E 00 - 6.562E 00 MeV HE3      FLUX        (MEAN -(ID3 ),(IID3))
A  8  = 1.515E 01 - 2.213E 01 MeV HE3      FLUX        (MEAN -(IA3 ),(IIA3))
A  9  = 2.308E 01 - 5.513E 01 MeV HE3      FLUX        (MEAN -(IA3 ),(IIA3))
A 10  = 1.346E 00 - 1.689E 00 MeV HE4      FLUX        (MEAN -(ID2 ),(IID2))
A 11  = 2.056E 00 - 2.978E 00 MeV HE4      FLUX        (MEAN -(ID3 ),(IID3))
A 12  = 4.475E 00 - 6.446E 00 MeV HE4      FLUX        (MEAN -(IA2 ),(IIA2),(ID3 ),(I
A 13  = 6.410E 00 - 1.308E 01 MeV HE4      FLUX        (MEAN -(IA3 ),(IIA3))
A 14  = 1.502E 01 - 2.206E 01 MeV HE4      FLUX        (MEAN -(IA3 ),(IIA3))
A 15  = 2.311E 01 - 5.514E 01 MeV HE4      FLUX        (MEAN -(IA3 ),(IIA3))
A 16  = 2.000E 00 - 2.903E 00 MeV C12      FLUX        (MEAN -(ID2 ),(IID2))
A 17  = 4.444E 00 - 6.437E 00 MeV C12      FLUX        (MEAN -(ID3 ),(IID3))
A 18  = 6.437E 00 - 1.305E 01 MeV C12      FLUX        (MEAN -(ID3 ),(IID3))
A 19  = 1.523E 01 - 2.240E 01 MeV C12      FLUX        (MEAN -(IA3 ),(IIA3))
A 20  = 2.333E 01 - 5.540E 01 MeV C12      FLUX        (MEAN -(IA3 ),(IIA3))
A 21  = 2.002E 00 - 2.903E 00 MeV N14      FLUX        (MEAN -(ID2 ),(IID2))
A 22  = 4.440E 00 - 6.431E 00 MeV N14      FLUX        (MEAN -(ID3 ),(IID3))
A 23  = 6.431E 00 - 1.301E 01 MeV N14      FLUX        (MEAN -(ID3 ),(IID3))
A 24  = 1.507E 01 - 2.226E 01 MeV N14      FLUX        (MEAN -(IA3 ),(IIA3))
A 25  = 2.329E 01 - 5.518E 01 MeV N14      FLUX        (MEAN -(IA3 ),(IIA3))
A 26  = 2.000E 00 - 2.900E 00 MeV O16      FLUX        (MEAN -(ID2 ),(IID2))
A 27  = 4.459E 00 - 6.443E 00 MeV O16      FLUX        (MEAN -(ID3 ),(IID3))
A 28  = 6.443E 00 - 1.304E 01 MeV O16      FLUX        (MEAN -(ID3 ),(IID3))
A 29  = 1.506E 01 - 2.215E 01 MeV O16      FLUX        (MEAN -(IA3 ),(IIA3))
A 30  = 2.323E 01 - 5.594E 01 MeV O16      FLUX        (MEAN -(IA3 ),(IIA3))
A 31  = 2.009E 00 - 2.901E 00 MeV NE20     FLUX        (MEAN -(ID2 ),(IID2))
A 32  = 4.423E 00 - 6.455E 00 MeV NE20     FLUX        (MEAN -(ID3 ),(IID3))
A 33  = 6.455E 00 - 1.305E 01 MeV NE20     FLUX        (MEAN -(ID3 ),(IID3))
A 34  = 2.324E 01 - 5.549E 01 MeV NE20     FLUX        (MEAN -(IA3 ),(IIA3))
A 35  = 2.011E 00 - 2.901E 00 MeV MG24     FLUX        (MEAN -(ID2 ),(IID2))
A 36  = 4.425E 00 - 6.440E 00 MeV MG24     FLUX        (MEAN -(ID3 ),(IID3))
A 37  = 6.440E 00 - 1.306E 01 MeV MG24     FLUX        (MEAN -(ID3 ),(IID3))
A 38  = 2.321E 01 - 5.556E 01 MeV MG24     FLUX        (MEAN -(IA3 ),(IIA3))
A 39  = 2.015E 00 - 2.901E 00 MeV SI28     FLUX        (MEAN -(ID2 ),(IID2))
A 40  = 4.446E 00 - 6.426E 00 MeV SI28     FLUX        (MEAN -(ID3 ),(IID3))
A 41  = 6.426E 00 - 1.305E 01 MeV SI28     FLUX        (MEAN -(ID3 ),(IID3))
A 42  = 2.321E 01 - 5.524E 01 MeV SI28     FLUX        (MEAN -(IA3 ),(IIA3))
A 43  = 2.007E 00 - 2.901E 00 MeV CA40     FLUX        (MEAN -(ID2 ),(IID2))
A 44  = 4.431E 00 - 6.436E 00 MeV CA40     FLUX        (MEAN -(ID3 ),(IID3))
A 45  = 2.002E 00 - 2.902E 00 MeV FE56     FLUX        (MEAN -(ID2 ),(IID2))
A 46  = 4.450E 00 - 6.422E 00 MeV FE56     FLUX        (MEAN -(ID3 ),(IID3))
A 47  = 6.422E 00 - 1.305E 01 MeV FE56     FLUX        (MEAN -(ID3 ),(IID3))
A 48  = 1.501E 01 - 2.203E 01 MeV FE56     FLUX        (MEAN -(IA2 ),(IIA2))
A 49  = 2.321E 01 - 5.551E 01 MeV FE56     FLUX        (MEAN -(IA3 ),(IIA3))
```

FILE A=REAMES1.DAT

∞

FIG 1

PLOT 1 ISEE-3 FLUX FOR THE PERIOD 2/10/79 0: 0: 0 TO

BEGIN 1979: 41: 0: 0

END 1979: 43: 18: 0

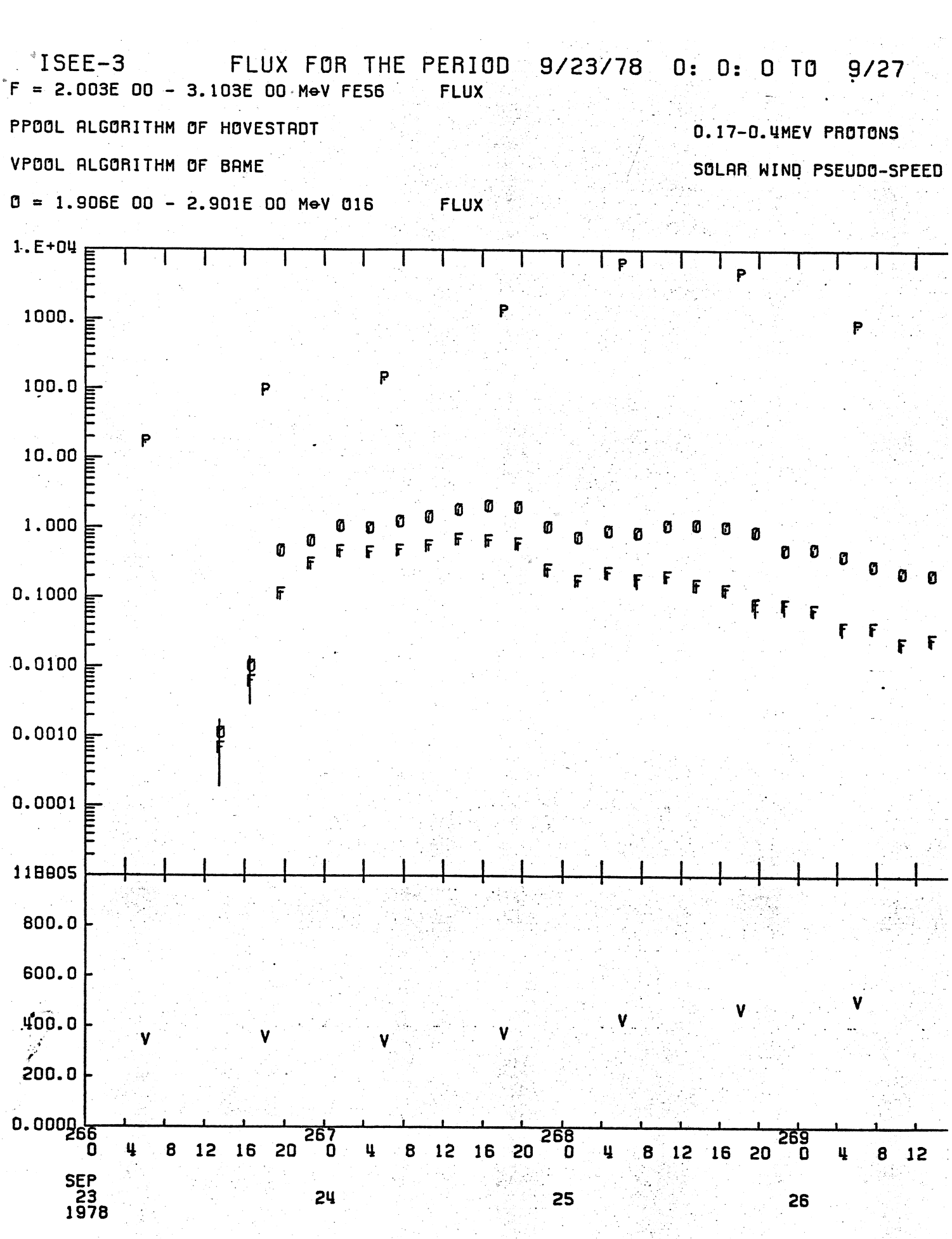
YMIN= 0.100E-02 YMAX= 0.100

XSIZE= 3753 YSIZE= 1500

COMMAND (EG.)	ACTION
HELP	-DISPLAY THIS LIST
FILE A= NAME.DAT	-SELECT NEW FILE A
FILE B	-DISPLAY FILE LIST FOR FILE B
SCALE 1	-DISPLAY SCALE DATA FOR PANEL 1
EDIT 2	-CURSOR EDIT SCALE DATA FOR PANEL 2 (BUTTON 1 RESETS WITHOUT CHANGE)
PLOT 1	-PLOT PANEL 1
PLOT BOTH	-PLOT PANELS 1 AND 2 (BUTTONS 8-23 SELECT GRIDS & LABELS
NEXT	-ADVANCE PANEL(S) TO NEXT TIME INTER
BACK	-BACK UP TO PREVIOUS TIME INTERVAL
E1= P2, A5/A6, '*'	-DEFINE ELEMENT(CURVE) 1 ON PANEL 2 AS THE RATIO OF THE OBJECTS 5 & 6 ON FILE A. USE SYMBOL '*'.
E3	-DISPLAY CURRENT DEFINITION OF EL. 3
STOP	-CLOSE FILES, GET OFF VG, STOP. (BUTTON 7 HAS SAME EFFECT)

ENTER FILE NAME:
FILE A=∞

Fig 3



$\Sigma \epsilon_{\Delta}$ MAG

∞

Fig 4

Blank Spacer Page

THIS DOCUMENT DESCRIBES THE FLUX FILE FORMAT AND FLXMA2 UTILITY PROGRAM.

****

AN ALTERNATE STRUCTURE FOR THE FLUX DATA FILES USED BY TOM CONLN'S 'FLXPLT' PROGRAM HAS NOW BEEN DEVELOPED.

IN THE NEW FORMAT, THE "T-FILES" ARE MERGED WITH THE MAIN DATA FILE, THUS ELIMINATING THE NEED FOR A SEPARATE "T-FILE". THE STRUCTURE OF THE NEW FILE IS AS FOLLOWS:

RECORDS 1 TO 2+NOBINS : SAME AS THE ORIGINAL FORMAT OF THE CONLON DIRECTORY OR "T-FILE". THE PARAMETER LCOUNT IS NOW THE TOTAL NUMBER OF RECORDS IN THE FILE (=2+NOBINS + NUMBER OF TIME DATA RECORDS).

RECORDS NOBINS+3 TO LCOUNT : DATA RECORDS IN THE SAME FORMAT AS THE ORIGINAL CONLON DATA RECORD FORMAT.

THE 'RECORDSIZE' OF THE NEW FILE IS COMPUTED AS MAX(2+2*NOBINS,33). NOTE THAT THE 'RECORDSIZE' PARAMETER OF THE 'OPEN' STATEMENT ONLY NEEDS TO BE SPECIFIED WHEN A FILE IS OPENED FOR WRITING.

THE PROGRAM FLXPL2 (IN [200,102]) IS A MODIFIED VERSION OF THE ORIGINAL FLXPLT PROGRAM. FLXPL2 READS THE NEW FILE FORMAT AND PRESERVES ALL OPTIONS OF THE ORIGINAL FLXPLT.

THE PROGRAM FLXMA2 (IN [200,102]) IS AN EXPANDED VERSION OF THE PROGRAM FLXMAN AND IS A GENERAL PURPOSE UTILITY FOR MANIPULATING (MERGING, REDUCING, LISTING, AVERAGING) FILES IN THE FLUX FILE FORMAT. THE PROGRAM HAS FIVE (5) BASIC OPTIONS.

1 = "HORIZONTAL MERGING" = CONCATENATION OF FILES WITH IDENTICAL AVERAGING PERIODS, NUMBERS AND IDENTIFICATIONS OF BINS, AND NO TIME OVERLAPS. THE FINAL NUMBER OF BINS ON THE MERGED FILES = INITIAL NUMBER OF BINS ON EITHER OF THE TWO INPUT FILES. THE BASE YEAR AND OUTPUT FILE NAME ARE SPECIFIED BY THE USER (DEFAULT BASE YEAR IS THE SMALLER OF THE BASE-YEARS IN THE TWO INPUT FILES OR 1977).

2 = "VERTICAL MERGING" = MERGING OF FILES WITH IDENTICAL AVERAGING PERIODS BUT OTHERWISE NOT CONSTRAINED. THE FINAL NUMBER OF BINS ON THE MERGED FILE = SUM OF THE NUMBER OF BINS IN THE TWO INPUT FILES. BASE YEAR AND OUTPUT FILE NAME ARE HANDLED AS IN OPTION 1 ABOVE.

3 = "STRIPPING",LISTING = THIS OPTION ALLOWS THE SPECIFICATION OF A SUBSET OF BINS AND TIMES FROM A GIVEN FILE TO BE LISTED ON THE LINE PRINTER AND/OR WRITTEN TO A NEW OUTPUT FILE. "STRIPPING" IS OFTEN USEFUL WHEN COMBINED WITH "VERTICAL MERGING" TO CREATE A COMBINED DATA FILE FROM TWO OR MORE LARGE INPUT FILES SO THAT E.G. SPECIFIC BINS OF DIFFERENT SATELLITES CAN BE DIRECTLY COMPARED. BASE YEAR AND OUTPUT FILE (IF DESIRED) ARE SPECIFIED AS IN OPTIONS 1,2 ABOVE. THE INPUT FILE IS SPECIFIED WITH TWO TRUE,FALSE OPTIONS TO CONTROL WHETHER A PRINTER LISTING AND/OR DISK FILE OUTPUT ARE REQUIRED. THE PROGRAM HAS OPTIONS TO ALLOW A LISTING OF INDIVIDUAL BIN LABELS AT THE TERMINAL FOR SELECTION OR DIRECT SPECIFICAION OF RANGES OF BIN NUMBERS TO BE TRANSFERRED. THE ANALYSIS OF NON-CONTIGUOUS TIME INTERVALS IS ALLOWED.

4 = AVERAGING = ALLOWS THE AVERAGING OF DATA OVER TIME WITH OPTIONS TO SELECT THE METHOD OF AVERAGING, CONSTRUCT EITHER PERIODIC OR NON-CONTIGUOUS TIME INTERVALS AND TO LIST AND/OR WRITE THESE RESULTS TO LINE PRINTER/DISK AS IN OPTION 3 ABOVE. IN THE PERIODIC MODE, AN AVERAGING PERIOD IS SPECIFIED (NOT NECESSARILY AN EVEN MULTIPLE OF THE INPUT FILE AVERAGING PERIOD) AND THE PROGRAM DOES THE AVERAGING. FOR AVERAGING TOGETHER NON-CONTIGUOUS INTERVALS, NO AVERAGING PERIOD IS SPECIFIED BUT THE LIMIT OF THE GROUP OF TIMES IS INDICATED BY SETTING ISYR = -2.

THREE(3) AVERAGING PROCEDURES ARE CURRENTLY ALLOWED BY THE PROGRAM: (1) 1/SIGMA SQUARE (WITH ZERO FLUX,ERROR TIMES WEIGHTED BY THE AVERAGE WEIGHT OF ALL NON-ZERO POINTS); (2) NO WEIGHTING (DIPECT ADDITION OF FLUXES); (3) ADDITION OF COUNTS COMPUTED FROM EACH FLUX BY THE EQUATION (FLUX/ERROR)**2 WITH DETECTOR ACTIVE TIMES PROPORTIONAL TO COUNTS/FLUX.

5 = "T-FILE" REFORMATING = AUTOMATIC PROCEDURE TO CONVERT FILES IN THE OLD CONLON TWO FILE STRUCTURE (DIRECTORY PLUS DATA FILE) TO THE NEW ONE FILE STRUCTURE. THE INPUT FILES CURRENTLY MUST RESIDE ON THE DEFAULT DISK; THE OUTPUT FILE IS ASSIGNED THE NAME OF THE INPUT FILE WITH A VERSION NUMBER INCREMENTED BY 1. THE PRINT AND WRITE OPTIONS ARE DUMMY PARAMETERS IN THIS OPTION. NOTE THAT NORMAL EXECUTION OF THE PROCEDURE SHOULD RETURN 'STAT' VALUES IN THE COMPLETION MESSAGE OF ZERO(0).

ALL OPTIONS (EXCEPT 5) ALLOW ACCESS TO FILES ON ANY DISK DEVICE (DB0,RD1, DX0,DX1) AND ANY LIBRARY. OPTIONS 1,2,5 GENERATE LITTLE OR NO PRINTER OUTPUT. THE BASE YEAR OF THE OUTPUT FILE CAN BE ALTERED EXCEPT IN OPTION 5.

THIS DOCUMENT IS STORED IN [200,102]FLXFILE.DOC .

Blank Spacer Page

```
RESUME WSP  [1]                    20 23
PAM    LIB  [380]                  20 24  L W
PAM1   LIB  [100]                  20 25  L

Files=16,Blocks=861,Free=18675,Largest=18574

#dir dk3
        pam.lib


  Directory Structure of DK3:PICS    17-May-78
  Directory Blocks: Available=  4, Used=  1

  TEKLOG PDB  [4]      30-Aug-77  15:35
  GFM    PDB  [23]      4-May-77   9:20
  TIGER  PDB  [330]    26-Nov-77  16:15
  VIRTUA PDB  [3]       5-Dec-77   8:14
  POPEYE PDB  [14]

  Files=5,Blocks=374,Free=2,Largest=2

#dir dk3:pam1.lib


  Directory Structure of DK3:SYM     21-Apr-81
  Directory Blocks: Available=  4, Used=  2

  CIRCL  PDB  [1]       7-Aug-76  19:51  W
  SHIELD PDB  [1]      10-Aug-76  22:48  W
  GEAR   PDB  [2]       6-May-77  11:33  W
  DOT    PDB  [1]       7-Aug-76  19:46  W
  4081   PDB  [4]      11-Apr-77         W
  MONITO PDB  [2]      11-Apr-77  16:17  W
  OVST   PDB  [2]      11-Apr-77         W
  STRSHP PDB  [7]      11-Apr-77         W
  WIZARD PDB  [19]      8-Sep-77  15:46  W
  ARROW  PDB  [1]       2-Aug-76         W
  BOX    PDB  [1]      10-Aug-76  20:08  W
  CIRCLE PDB  [1]      15-Aug-76   0:15  W
  ARROB  PDB  [1]       5-Aug-76   0:12  W
  FLAG   PDB  [2]      21-Apr-81  15:54  W
  OFLAG  PDB  [1]      21-Apr-81  15:54  W
  PARALL PDB  [1]      11-May-81  13:56
  LIST   PDB  [1]      11-May-81  13:56
  HEBE   PDB  [2]      11-May-81  14:19
  CATLG  PDB  [1]      11-May-81  14:27
  PROGRM PDB  [1]       8-Jun-81  11:17
  DIAMND PDB  [1]      12-Jun-81  12:11
  SEMCIR PDB  [1]       2-Jul-81   9:48
  TAPE   PDB  [1]      21-May-82  10:12
  RECT   PDB  [1]      21-Sep-82  11:09
  DISK   PDB  [1]      21-Sep-82  11:20
```

```
  Files=25,Blocks=57,Free=39,Largest=38

#copy ■■■
      ■
      dk3 pam■wsp.lib=dk0■
                          :wsp.lib
-Files Copied-
DK0 WSP    LIB
#dir ■
     dk3:pam■
            wsp.lib


  Directory Structure of DK3:WSP    25-Jun-81
  Directory Blocks: Available=  4, Used=  1

  BLOCK   WSP  [2]      17-Mar-78   8:41
  PART    WSP  [2]       5-Oct-77  20:55
  CIRCLE  WSP  [3]      23-Mar-78   7:32
  FACT    WSP  [1]       9-Mar-77   9:15
  FLEET   WSP  [2]       9-Mar-77   9:20
  GFM     WSP  [4]      25-Jun-81  14:15

  Files=6,Blocks=14,Free=0,Largest=0

#
```

Directory Structure of DK2:[GMH] on 25-Nov-76
Directory Blocks Available= 13, Used= 0

CENTER	HLP	[1]	30-Nov-76	8	31
CLEAR	HLP	[1]	30-Nov-76	8	44
COS	HLP	[2]	30-Nov-76	9	12
CREATE	HLP	[2]	30-Nov-76	9	20
CCL	HLP	[1]	30-Nov-76	16	56
DIFFER	HLP	[1]	30-Nov-76	17	26
DISPLA	HLP	[2]	30-Nov-76	17	34
ERASE	HLP	[1]	30-Nov-76	18	18
FIX	HLP	[1]	30-Nov-76	18	24
FRACTI	HLP	[2]	30-Nov-76	18	28
GREATE	HLP	[1]	30-Nov-76	18	33
BOOLEA	HLP	[1]	30-Nov-76	18	39
GRID	HLP	[2]	30-Nov-76	19	06
HDCOPY	HLP	[1]	30-Nov-76	19	14
IF	HLP	[3]	30-Nov-76	19	17
INTENS	HLP	[2]	1-Dec-76	7	03
LESS	HLP	[1]	1-Dec-76	7	21
LPRODU	HLP	[2]	1-Dec-76	7	23
LSUM	HLP	[1]	1-Dec-76	7	27
MAXIMU	HLP	[1]	1-Dec-76	7	29
MINIMU	HLP	[1]	1-Dec-76	7	33
MOD	HLP	[1]	1-Dec-76	7	35
NOT	HLP	[2]	1-Dec-76	8	30
PATTER	HLP	[3]	1-Dec-76	8	34
PRINT	HLP	[2]	1-Dec-76	8	52
PRODUC	HLP	[1]	1-Dec-76	9	00
QUOTIE	HLP	[1]	1-Dec-76	9	04
HELP	HLP	[2]	21-Mar-77	12	38
PICK	HLP	[3]	22-Mar-77	9	02
MENUSH	HLP	[5]	15-Apr-77	8	25
SAVE	HLP	[3]	1-Dec-76	10	26
SHOW	HLP	[2]	1-Dec-76	10	42
SIN	HLP	[2]	1-Dec-76	10	47
SQRT	HLP	[2]	1-Dec-76	10	53
SUM	HLP	[1]	1-Dec-76	11	02
VIEWPO	HLP	[2]	1-Dec-76	11	38
WINDOW	HLP	[2]	1-Dec-76	11	49
ZOOM	HLP	[2]	1-Dec-76	12	44
LIST	HLP	[5]	1-Dec-76	13	12
DIRECT	HLP	[1]	29-Mar-77	10	21
END	HLP	[2]	6-Dec-76	11	13
DASH	HLP	[3]	4-Mar-77	8	13
VECTOR	HLP	[6]	4-Mar-77	9	20
DOUBLE	HLP	[2]	4-Mar-77	9	39
INSTAN	HLP	[3]	4-Mar-77	9	56
MOVE	HLP	[2]	4-Mar-77	10	08
POINT	HLP	[1]	4-Mar-77	10	12
REMOVE	HLP	[6]	4-Mar-77	10	16
ROTATE	HLP	[4]	4-Mar-77	11	18
SCALE	HLP	[4]	4-Mar-77	11	31
TRANSL	HLP	[3]	4-Mar-77	11	49
RESET	HLP	[3]	21-Mar-77	12	32

```
CONCAT HLP  [4]      25-Mar-77  12:20
SAME   HLP  [1]       4-Mar-77  12:10
FUNKEY HLP  [2]      17-Mar-77   9:33
LINE   HLP  [2]      21-Mar-77  11:11
LENGTH HLP  [2]      31-Mar-77   9:15
TEXT   HLP  [3]       9-Mar-77  12:22
LABEL  HLP  [7]       9-Mar-77  12:24
CANCEL HLP  [4]       9-Mar-77  12:28
LINERE HLP  [2]      10-Mar-77   8:20
DASHRE HLP  [2]      10-Mar-77   8:20
POINRE HLP  [2]      10-Mar-77   8:21
MOVERE HLP  [2]      10-Mar-77   8:22
ABSOLU HLP  [1]      16-Mar-77  14:04
SLEEP  HLP  [1]      16-Mar-77  14:27
DEFINE HLP  [5]      16-Mar-77  14:36
EXTENT HLP  [4]      12-Apr-77  19:56
UNTIL  HLP  [1]      21-Apr-77
LITERA HLP  [4]      27-Apr-77  15:37
XYPOIN HLP  [2]      16-May-77  15:40
CLOSE  HLP  [4]       6-Jun-77   8:17
SETPOI HLP  [3]       7-Jun-77  11:01
DELETE HLP  [2]      29-Jun-77  14:23
EQUAL  HLP  [2]      29-Jun-77  14:26
WORKSP HLP  [3]      29-Jun-77  14:37
SELECT HLP  [6]      29-Jun-77  14:38
NAME   HLP  [10]     29-Jun-77  14:39
FRAME  HLP  [3]      29-Jun-77  14:51
ASPECT HLP  [4]      29-Jun-77  14:53
REPEAT HLP  [3]       7-Jul-77  17:03
USE    HLP  [9]      11-Jul-77  12:55
SUBPIC HLP  [3]      19-Jul-77  10:49
GIN    HLP  [2]       8-Sep-77  14:53
LDIFFE HLP  [2]      30-Oct-77  13:51
CURREN HLP  [6]       9-Nov-77   8:20
GFM    HLP  [8]      18-Jan-79   1:20
PART   HLP  [5]      18-Jan-79   1:41

Files-88,Blocks-237,Free-0,Largest-0

#
```

```
ENTER THE DATE IN THE FOLLOWING FORMAT
DD MMM YY
*?? 010-82
Invalid date specified
ENTER THE TIME IN THE FOLLOWING FORMAT
HH MM SS
*20 20 00

System Status of GOS Version 4 Level 02 (SYSGN3)

January 0, 1964      20  0  2

Devices  SYS*,KB0,DC ,JOY ,DK0,DK1 ,DK2 ,DK3 ,CT0 ,FD0,FD1 ,MT0 ,TB0(Lg),PL0
         NUL ,SYS(DK0),USR(DK3),GIN(JOY)

         Character size 3, 64KB memory, GOSTOP = 7768


** REMOVE THE GOS IPL CARTRIDGE ENTIRELY FROM THE DRIVE !!!  **

A NEWS HELP FILE FOR CURRENT DEVELOPMENTS  #HELP NEWS    LAST ENTRY WAS ON 9/16/82
->End of Batch Stream
#GFM
Plot 80/GFM Version 4 Level 02
TEST1   .PDB      3
TEST2   .PDB      3
F1      .PDB     36
F2      .PDB     36
F3      .PDB     39
F4      .PDB     55
F5      .PDB     31
F6      .PDB     31
F7      .PDB     45
F8      .PDB     45
V1      .WSP      1
TEST3   .PDB     43
TRYGAR  .PDB     12
RESUME  .WSP      1
```

Blank Spacer Page

IMP FLUX PLOT PROGRAM --- OUTPUT TAPE FORMAT 5/01/79

****

HEADER RECORDS:

RECORD	WORD	FORMAT	DESCRIPTION
1	1-20	80A1	USER TITLE
	21-27	2HA4	RUN TIME OF FLUX PLOT PROGRAM
	28	I*4	AVERAGING INTERVAL (MIN)
	29	I*4	NUMBER OF BINS
	30	I*4	PLOT POINT INTERVAL (MIN)
	31-60		SPARE (ZEROS)
2	1-2	8A1	PARTICLE LABEL FOR 1ST BIN
	3	I*4	IMP NUMBER (6,7, OR 8) FOR 1ST BIN
	4	R*4	LOWER ENERGY (MEV) FOR 1ST BIN
	5	R*4	UPPER ENERGY (MEV) FOR 1ST BIN
	6	R*4	AVERAGE ENERGY (MEV) FOR 1ST BIN
	7	R*4	NORMALIZATION FACTOR FOR 1ST BIN
	8-12	10(I*2) →	10 BOX NUMBERS FOR 1ST BIN
	13-24	SAME AS WORDS 1-12 FOR 2ND BIN	
	25-36	SAME AS WORDS 1-12 FOR 3RD BIN	
	37-48	SAME AS WORDS 1-12 FOR 4TH BIN	
	49-60	SAME AS WORDS 1-12 FOR 5TH BIN	
3	1-60	SAME AS RECORD 2 FOR BINS 6-10	
4	1-60	SAME AS RECORD 2 FOR BINS 11-15	
5	1-60	SAME AS RECORD 2 FOR BINS 16-20	
6	1-60	SAME AS RECORD 2 FOR BINS 21-25	

****

DATA RECORDS:

WORD	HW	FORMAT	DESCRIPTION
1	1	I*2	START TIME: MONTH
	2	I*2	DAY OF MONTH
2	1	I*2	YEAR - 1900
	2	I*2	HOUR
3	1	I*2	MINUTE
	2	I*2	SECOND
4		I*4	SECONDS OF THE START YEAR
5-7		6(I*2)	STOP TIME: SAME AS WORDS 1-6
8		I*4	SECONDS OF THE STOP YEAR
9		R*4	FLUX FOR 1ST BIN (COUNTS/(SEC CM2 STER MEV/N))
10		R*4	ERROR IN FLUX FOR 1ST BIN (COUNTS/(SEC CM2 STER MEV/N))
11-12		2(R*4)	FLUX AND ERROR FOR 2ND BIN
13-14		2(R*4)	FLUX AND ERROR FOR 3RD BIN
• 15-16	4	•	• •
• 17-18	5	•	• •
• 19-20	6	•	• •
57-58		2(R*4)	FLUX AND ERROR FOR 25TH BIN
59-60			SPARE (ZEROS)

****

*** END OF MEMBER *** 60 RECORDS PROCESSED *****

rates plot
1-6 rates per deck
multiple decks may be run
ZTime
keep
keep
use for label
change to A1 format

Blank Spacer Page

PDP-11/70 FLXPLT User's Guide

The FLXPLT Program generates a time history or a spectral (Flux vs. energy) plot on the PDP-11/70 Vector General. The program runs from two data sets which are described in the appendix. The data set format can be checked by running [200, 102]FLXMAN to generate a listing. To run FLXPLT the user enters:

RUN [200,102]FLXPLT$ where $ = the ESC key

then enter the time option after a prompt as for example:

2$

then enter the data set name after a prompt as for example:

DPT7879.DAT$

the prompt will return the number of records in the data set.

The user then goes to the Vector General (VG) LIST mode. displays bins
The VG is in list (L) mode with lights 0, 2, 12 on.

LIST Mode:

0: changes from ratio to non ratio or v/v

In ratio light 3;13 are also lit up

2: moves N up on list

12: moves N down on list

3: moves D up on list

13: moves D down on list

Ratio mode calculates and plots ratio between N bins and D bin

CR (carriage return) plots bin from PLOT mode

PLOT mode:

Press on the keyboard the following keys for the following functions;

E: exit FLXPLT

CR: plot next bin

T: advance time

V: change scale

curser appears under upper limit

→ to move curser

lights: 1: return to plot

2: change type from Linear to Log or v/v

type in new limits in E-format

L: to get back to List mode

B: display beginning time with curser to alter lights on: 1 press to get back to plot

G: display grid lines, press again for more

C: time plotting mode (blinking cross appears and lights form cross. Press lights to move cross.) s: halfs increment

press center to display flux and time at cross

CR: erases cross return to plot

S: spectrum appears

T: moves forward in time

CR: time history plot with cross again

F: restore increment

M: change from histogram to point plot with error bars or v/v

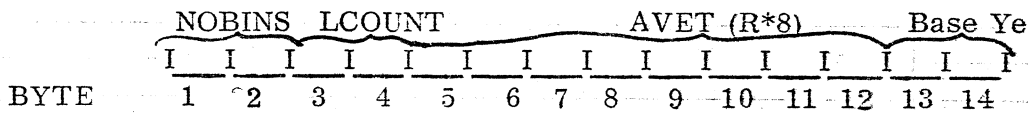
hardcopy by pressing Spec BS together then type CPY$ on main console

APPENDIX

For each data set, there exists two files - a directory file and a data file. Presently, the directory file has a 'T' prefix to the data file. Both files are direct access.

The directory file: 132 byte records

1st record -



the rest of the bytes are not looked at.

NOBINS = the number of bins (rates, boxes)

LCOUNT = the number of records in the data file (equal to or less then the number of averaging intervals in the time period covered.)

AVET = averaging interval (in seconds)

Base year = form of 1977

2nd record -

ASCII spacecraft label (e.g., IMP-7, ISEE-3, VOYAGER-1, etc,)

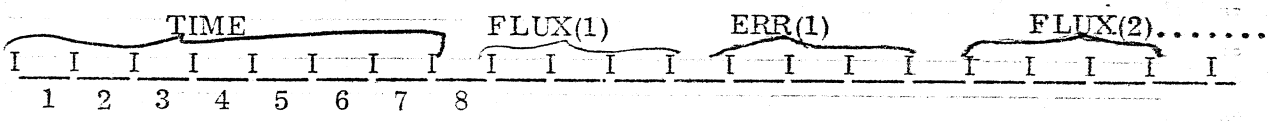
3rd thru NOBINS + 2

records contain bin labels in the order they appear on the data records. The lower and upper energies should have the format (7x, 1PE10.3, 2x, 1PE10.3, 6x, 3A4) The particle label should start in byte 35. The data file:

records size in bytes = 8*nobins+8

Data file:

Record size = 4*NOBINS+4 in halfwords



ALL binary floating point
TIME in seconds of some base year.

[200,102] Tapesn

Trygarb.FLX

[200,102] FLXMAN

option 3 read data

[200,102] NEWTIM

p3 spectrum panel

edit 3

spec fs to get versateck

spec gs to get file for tektronix

[2,76] PDBTRAN

Change labels

[200,102] LABROB alter labels

[200,102] FLXFR3 put file back together

needs send to end prog

---

after tape transfer

type:

GFPPA

at *ER DISP DK3:TRYGARB

Blank Spacer Page

S-214

Document #: 82-22-3

Transfer from VG to TEKTRONIX

Jenny Susan Jacques
Code 664
Goddard Space Flight Center

August 1982

I. Background

Many analysis programs exist on the PDP 11/70 which use the Vector General to display data. A hardcopy of this can be made on the Versatek machine. However, these plots could not be edited, overlayed, etc., and the Versatek quality sometimes is insufficient. This transfer system will create a TEKTRONIX 4081 picture file from the Vector General which can then be edited and overlayed, with either hardcopy or plotter as the output device. The plotter is versatile in its plot sizing and plotting media, using any paper or plastic film (for viewgraphs). Moreover, the user may wish to add text explanations to the picture or delete extraneous lines, symbols, etc., all of which are easily accomplished using the GFM program on the TEKTRONIX. Moreover, sections of a picture may be blown up for analysis and plotting.

II. Applicable Documents

1. Document #82-22-2
   "PDBTRAN: Transfers PDB files from disk to tape
2. TEKTRONIX 4081 GFM Users' Guide
3. TEKTRONIX 4081 Operator's Reference Manual

III. Vector and Character Conversion

1. All vector line types (solid, dash, dotted, and dash-dot-dash) are preserved.
2. Vector and character intensities are converted as follows:

VG Intensity Value	TEKTRONIX Intensity
-16 → +12	normal
13	bright
14	normal broad-line
15	bright broad-line

3. The character sets match up to ASCII character del (octal 177). All characters greater than octal 177 are changed to a space character (octal 40).

IV. Method

The transfer of files is a three step process:

1. While the picture is on the Vector General, press the SPEC and GS keys at the same time. Wait for TEKCPY--STOP message to be written on the Decwriter. There now exists a file called [1,1]TEKCPY.TEK where the TEKTRONIX picture file format has been created. Successive SPEC-GS commands produce different version numbers of [1,1]TEKCPY.TEK.

2. Transfer of these files to tape is done by running [2,76]PDBTRAN as shown on the next page. (Use the Decwriter to avoid "privilage" problems.)

   Each .TEK file is deleteted after being transferred.

   NOTE: The tapes used must be the small 8 1/2" reels. The density is 800 BPI.

Sample Transfer

(**) beside a line indicates a user response

```
                SCI>>
(Response from  TAKING TEKTRONIX HARDCOPY
  SPEC-GS)      TEKCPY  --  STOP

(**)            SCI>> RUN [2,76]PDBTRAN
                15:08:01
                ENTER DESIRED START FILE
(**)            1
                MOUNT-**VOLUME INFORMATION**
                        DEVICE  =MM0
                        CLASS   =FOREIGN.
                        UIC     =[1,1]
                        ACCESS  =[RWED,RWED,RWED,RWED]
                        CHARAC  =[FOR,DCF]
                MOUNT --  MM0 ** MOUNT COMPLETE **
                **** JOB215 - CONTINUED
                MOUNT PDBTAP ON DRIVE 0****TYPE CONT OR X TO ABORT

(**)            CONT
                ENTER THE NUMBER OF FILES TO TRANSFER OR -1 FOR ALL
(**)            2


                ** END OF TRANSFER OF       2 PDB FILES
                 FILE TRANSFER BEGAN WITH FILE NUMBER       1


                DMO -- MM0: ** DISMOUNT COMPLETE **
                **** JOB215 - CONTINUED
                **TAPE PDBTAP DISMOUNTED FROM DRIVE 0**

                JOB215  --  STOP

                15:08:30  Size: 18K  CPU: 0.28  Status: SUCCESS

                SCI>>
```

3. On the TEKTRONIX 4081, place the tape on the TEKTRONIX tape drive and boot up the system as described on the displayed instruction sheet. Run the program DK1:TAPEIO. (The # is the system prompt) See the next page.

   NOTE: Make sure you specify a .PDB file name extension.

The transfer is now complete. To manipulate, display, and plot these files, run GFM as follows:

#GFM

The refresh scroll line prompt * appears in the lower left corner. To display your file, type DISPLA filename.PDB. See the GFM manual for further explanations of the GFM capabilities.

Sample TEKTRONIX 4081 Tapeio Job

(**) indicates user response

```
(**)  *RUN DK1:TAPEIO
      ENTER 1 FOR DISK TO TAPE, 2 FOR TAPE TO DISK
(**)  2
      ENTER DATA TYPE: 1-ASCII,2-BINARY,3-FITS
(**)  2
      ENTER TRANSFER MODE: 1-AUTOMATIC , 2-INTERACTIVE
(**)  2
      ENTER TAPE READ/WRITE ERROR HANDLING OPTION:
      1 - NO ERRORS ACCEPTED
      2 - READ ERROR SPACING FORWARD TO DATA OK
      3 - ANY ERRORS ACCEPTED
(**)  1

      MOUNT TAPE ON DRIVE.ENTER CON TO CONTINUE OR ABO TO ABORT

(**)  CON
      ENTER DESIRED DISK FILE NAME:
(**)  DK3:TESTS.PDB
      ENTER TAPE START FILE AND RECORD RANGE  (START,END RECORD) FOR THIS DISK FILE.
(**)  1,1,999
      END OF FILE        1  ENCOUNTERED.
                2 TAPE LOGICAL RECORDS TRANSFERRED TO DISK FILE DK3:TESTS.PDB
      WANT TO PROCESS ANOTHER DISK FILE? (Y OR N)
(**)  N

      DISMOUNT IS SUCCESSFUL    ,STATUS=    0

      T A P E I O   R E P O R T

                        1 TAPE FILES PROCESSED
                        2 TAPE RECORDS PROCESSED
                        1 DISK FILES PROCESSED
                        2 DISK RECORDS PROCESSED
                        0 TOTAL I/O ERRORS ENCOUNTERED

      Exit
      *
```

Blank Spacer Page

Pam

CURRENT ADDITIONS AND PLANS FOR LECR AT PROGRAM DEVELOPMENT
Bob McGuire - 9/22/86

An Outline of the Improvements Made Thus Far.

The thrust of what I have been doing (when time has been available) has been to open up the capabilities of the CROSS program to files originally produced in the 11/70 format. To this end, I have developed a capability to:

1) Transfer 11/70 files directly from the 3081 disks into the AT (using the terminal emulation program YTERM and the transfer program PCTRANS. This has specifically required writing a very small 3081 VSFORTRAN program (REBLKU) to convert the very large RECFM=U blocks used in FLXWRT into smaller (<512 byte) blocks of the sort PCTRANS can handle.

2) I have written an AT program (in DeSmet C) to decode these transferred files, convert the word formats where necessary (particularly R*4 and R*8 words), write the times in I*4 as integer seconds of 1970 and generally imitate the structure defined by Don in his initial implementation of the CROSS program. The converting program is IN11. Now that these formats are "opened up", it should be quite simple to further modify IN11 to rewrite the files in the newest chapter/verse structure, with the 11/70 data in general appearing in the form of a "rate verse".

   At the moment, IN11 is set to decipher only the newest 11/70 output format (FLXWRTH2) suggested by Nand which is fully packed. The 3081 XTAPE program actually generates a slightly different format which is fixed-block with an integral number of logical records/block and 0 fill to the end of the block. Again, the modification to accomodate this (FLXWRTHD) format should be essentially trivial.

3) A significant number of additions have been made to the CROSS program itself.

   a) A new file type (Real) has been defined. The header information is in the same format as the Integer form. The data records (by definition right now) consist (for each record) of a start time (seconds of 1970 as I*4 or a long int), end time (also I*4) and a series of quantities, each consisting of a R*4 (float) real value and a R*4 error. As on the 11/70, error <0 implies missing data.

   error = 0.0

   b) CROSS will now process and plot real numbers with error bars. Presently the error bars are plotted as the normal horizontal and vertical lines with checking to keep the plotted lines within the boundaries of the plot frame. CROSS also processes the first data quantity as two I*4 words (the start and end times) and does check limits and plot versus time (but only so far as seconds of 1970, not with more useful units such as year/month/day/hours/minutes/sec).

   c) All limits processed by CROSS have been recast as real numbers for interaction with the screen display. For an Integer type file, these real limits are changed to integers before comparison with the data. Alternate plot routines have been written to Integer and Real

meeting was Thurs / Sept 25 / 1 pm

data (i.e. direct_plot vs directr_plot, tree_verse vs treer_verse). Since the plot limits are now real numbers for type Real, non-integral limits are supported (e.g. log plot scale .001 to 0.1, etc.). There remains a residual problem with round-off errors in plot labels which I plan to resolve shortly.

d) CROSS (and IN11) have been taught to handle large files, where large is meant both in the sense of many bins and/or many data points. Subroutines to control blocked access to the disk have been written where the calling program only needs to ask for the next data point and the subroutines worry about deblocking, reading in a new block, etc. IN11 uses the most recent version of this set of subroutines to allow blocked writes and well as reads and I will probably retrofit these into CROSS when there is time. So far I am only working with test files however. One of the next steps will probably be to convert a real file to the new format.

e) CROSS has been taught to handle large numbers of bins in its menu screen displays, i.e. it now supports scrolling functions in both the primary bin limit list and the secondary bin selection list. An unresolved issue is how to write the limit criteria to hardcopy output (i.e. the CANON), given that a complete list of the bins for each plot will consume a number of pages per plot. The CANON interface for the plots has not yet been tested, but Don's subroutines appear to be structured at a level at which there should be no problem once the screen display is OK.

An Outline of the Improvements Yet to be Made.

I anticipate two major thrusts in my AT program development efforts over the next several months. I need some further improvements to CROSS in order to handle the compositional analysis/statistical studies associated with finishing up the ISEE/IMP work for the ICRC manuscript and for publication. These will clearly be some priority. The second area I think it most logical to work in is the creation of our first large optical-disk databases.

In terms of CROSS improvements:

1) Round-off problems in making type float scales need to be fixed.

2) Once the chapter/verse flux format is more or less finalized, CROSS needs to be retrofitted to read and process the 11/70 data as rate type verses. The program IN11 also needs to be modified to make the new format from the input 11/70 data file.

3) The scheme used in CROSS for displaying and using bin limits needs to be changed. I suspect that a major cause for slow data display is the need to check many limits for each data point in files with many bins. I would probably propose to add some (automated) flag and perhaps a list of pointers so that only those limits that have been changed will be checked against the data. Also, then only these bin limits would need to be printed on the hardcopy output, which would help to resolve that problem.

4) In addition to this change/check flag, CROSS should support multiple "exclude" limits per bin (i.e., the limits LOW and HIGH are interpreted

as allowing points where VALUE<LOW or VALUE>HIGH if the exclude flag is set). It should also be possible to set limits on % errors allowed for accepted points (i.e., a data set with some valid points with large relative errors is difficult to display now in a way that any pattern in the more precisely determined points can be seen). These capabilities now exist in the SCATER program on the 11/70.

5) Right now I have chosen to plot errors in the traditional way with vertical and horizontal lines while Don is treating errors simply as another bin that can be encoded in color, etc. Both ways have application and I would like to modify CROSS to give the user the interactive choice of how errors should be plotted, limited etc. The actual changes are probably straight-forward but I need to think carefully about the easiest way to display the choice to the user.

6) I will probably not make any immediate effort to add more sophisticated time displays, etc. to CROSS at present. We really will need a true TIMPLT kind of program which will be separate from CROSS and my immediate thrusts are likely to be in CROSS-type scatter plot analysis and then in ISEE rare-element matrix display and analysis.

Given the things I have done with multiple satellites on the 3081 and 11/70 plus the creation of IN11 on the AT for the decoding of 11/70 file formats. I anticipate the following steps being part of that effort.

1) Revise IN11 to read the XTAPE format as well as the new native 11/70 tape file format and put the output in the new chapter/verse AT flux format when that is fully defined. The 11/70 data would undoubtedly be cast as rate verses.

2) I have just made a large series of Helios 1/2 runs including the creation of an output tape (15 minute resolution) although in the old FLUXPLOT format. I have asked Pamela to get perhaps Kristin or Pal to write a 3081 program to convert this FT30 format to FT31, FT32 form from which XTAPE can make an FT33 11/70 format. I do expect it may be some time before the various flux programs can all directly make the new AT flux file format, although that clearly should be our goal since then the actual box counts will be available to support true averaging.

3) I think the first database we should try to create will be the 15 minute ISEE database. I would propose to rerun (perhaps as early as the weekend of the 27-28th) the ISEE 15 minute averages, fixing a few of the problems that we had in earlier runs and probably adding some more bins for some additional elemental resolution. Note that we also have 15 minute data pool tapes available (which could be updated). Composition data analysis will still be severely limited however until we have access to box counts and will probably need to proceed by series of special event-average FLUXPLOT runs as before.

4) When the 3081 program to transform the old format Helios data is available, it seems reasonable to create a 15 minute Helios 1/2 database.

5) When the modifications to the IMP RATEPLOT and FLEXPLOT programs to handle larger numbers of bins are completed, we should probably rerun these IMP programs to create a 15 minute IMP database.

The above plans leave unaddressed at this time a variety of database issues (potential supplementary databases) that might be very useful. There is a tape at hourly resolution of IMP field and plasma data that, while not at the resolution we might like, could still be very useful. Anisotropy and sectored data have not been decided yet. We need online orbital data for all of Helios, ISEE and IMP (for heliospheric or magnetospheric positions). I have data files from NOAA of flare and radio data that would be useful. Online access to GOES X-ray data and magnetic storm data might be helpful in our future analyses. And I suspect this list is not complete.

Details in IN11 and the CROSS modifications.

[not yet done].

Blank Spacer Page

```
1   /*
2           Cross.c - Cross plot columns from input data table
3           Test revisions - R. McGuire - to add "real" capability
4   */
5
6   #define MXBUF  20000
7   #define BLK      120
8   #define MCOL     300    /* Max. no. data quantities (cols) */
9   #define L_LABL    40    /* Max. length of axis label */
10
11  #define M       0x10    /* Move */
12  #define D       0x28    /* Draw */
13  #define DR      0x29    /* Draw Relative */
14  #define C       0x06    /* Color */
15  #define PT      0x08    /* Point */
16  #define T       0x80    /* Text  */
17  #define TJ      0x85    /* Text Justification */
18
19  #define ISO_MODE   "\033;"              /* C48 (Diablo) */
20  #define SOFT_RESET "\033<"              /* C 1 (ISO) */
21
22  #include <keydef.h>
23  extern char scr_attr;
24
25  #include "windo.h"
26  extern struct window *w;
27
28  /* File-specific Data */
29  int infi;
30  char defi[30]="in11\\out11.plt              \0";
31  char *ptr;
32  long Ldata;
33  int Ncols, Nevents, Int_incr;
34  int Ax_type[MCOL+1];
35  /* Eventually define types for times and mixed ints, floats */
36
37  /* File- & window- specific Info */
38  char Header[60];
39  char Ax_labl[MCOL+2][L_LABL];
40  int Ax_lim[MCOL+1][2];
41  int Ax_bin[MCOL+1];
42  float Axr_lim[MCOL+1][2];
43  float Axr_bin[MCOL+1];
44
45  int Ishist=0;
46  int off[3]={0, 1, 2};
47  float psf[3]={1., 1., 1.};
48
49  char timebuf[]="mm/dd/yy hh:mm";
50  char Zbox=3;     /* Plot points as circle        */
51
52  /* Idtype=0 for Ascii and Integer input files, Idtype=1 for Real with errors */
53  int Idtype=0;
54  char Dtype[9]="        \0";
55
56  int Data[MXBUF];
```

```
57      int Bincr,No_rec,Reclen_int,Nrec_pblk,Brec_no,Ints_perblk ;
58
59      int Sdm_row=0;
60      int Scrn_rows=25;
61
62      extern int nodes, phas, maxn;
63      extern int Dif_d[3];
64      extern int Axtop[3];
65
66      char tbuf[BLK];
67      char *linlog[]={"linear","log"};
68      char date[9];
69
70      int Laser=0;    /* File output to Laser printer */
71      char *p_las={"plot.las"}; /* Laser printer file name */
72      char *prn="prn:";
73
74      main(argc,argv)
75      int argc;
76      char *argv[];
77      {
78              int lasr;
79
80              ptr=defi;
81              while (--argc>0)
82                      if (!strcmp(argv[argc],"/p"))
83                              p_las=prn;
84                      else
85                              ptr=argv[argc];
86              if ((lasr=creat(p_las))<=0)
87              {
88                      puts("Cannot open file ");
89                      puts(p_las);
90                      exit(1);
91              }
92
93              printf("This is the McGuire modified test version");
94              fputs(ISO_MODE  ,lasr);
95              fputs(SOFT_RESET,lasr);
96              close(lasr);
97              send_asc(" CA RF WI -46 593 -40 439 ");
98              send_asc("L 0  0  0  0 ");
99              send_asc("L 1  1  1  1 ");
100             send_asc("L 2  2  2  2 ");
101             send_asc("L 3  3  3  3 ");
102             send_asc("L 4  5  5  5 ");
103             send_asc("L 5  7  7  7 ");
104             send_asc("L 6  9  9  9 ");
105             send_asc("L 7 15 15 15 CX ");
106             palette('r');
107
108             winit();
109             clear_tree();
110             new_file();
111             printf("Size=%d\n",&infi-&Zbox+1);
112
```

```
113             while(1)
114             {
115                     if (Ishist)
116                     {
117                             clear_tree();
118                             if (Idtype==1)
119                                     treer_vers();
120                             else
121                                     tree_vers();
122
123                     }
124                     windo();
125             }
126             send_asc(" CA DI 1 ");
127     }
128
129     new_file()      /* Read in a new file */
130     {
131             int i, col[MCOL], *iptr;
132
133             if (!(infi=fopen(ptr,"r")))
134             {
135                     puts("Cannot open file ");
136                     puts(ptr);
137                     exit(1);
138             }
139             next_line();
140             sscanf(tbuf,"%8d%8d%8s%10ld",&Ncols,&Nevents,Dtype,&Ldata);
141             printf("%8d%8d %s%10ld\n",Ncols,Nevents,Dtype,Ldata);
142             if (Ncols<1 || Ncols>MCOL  )
143             {
144                     printf("Too many (few) columns=%d",Ncols);
145                     exit(1);
146             }
147             for (i=0; i<=Ncols; i++)
148                     Ax_bin[i]=1;
149             next_line();
150             strncpy(Header,tbuf,50);
151             for (i=0; i<Ncols; i++)
152             {
153                     if (!next_line())
154                     {
155                             puts("Error in header on file");
156                             puts(ptr);
157                             exit(1);
158                     }
159
160                     sscanf(tbuf,"%8d%8d%8e%8e",col+i,&Ax_type[i],
161                       &Axr_lim[i][0],&Axr_lim[i][1]);
162                     Ax_lim[i][0]=Axr_lim[i][0];
163                     Ax_lim[i][1]=Axr_lim[i][1];
164                     strncpy(Ax_labl[i],tbuf+32,L_LABL);
165                     if (Ax_type[i]==1)
166                     {
167                             sscanf(tbuf,"%8d%8d%8d%8d",col+i,&Ax_type[i],
168                               &Ax_lim[i][0],&Ax_lim[i][1]);
```

```
169                             Ax_bin[i]=Ax_lim[i][1];
170                             Axr_lim[i][0]=Ax_lim[i][0];
171                             Axr_lim[i][1]=Ax_lim[i][1];
172                     }
173                     clip_str(Ax_labl[i],L_LABL-2);
174             }
175             strcpy(Ax_labl[Ncols],"Histogram");
176             Axr_lim[Ncols][0]=0;
177             Axr_lim[Ncols][1]=20;
178             Ax_lim[Ncols][0]=Axr_lim[Ncols][0];
179             Ax_lim[Ncols][1]=Axr_lim[Ncols][1];
180
181             Ax_labl[Ncols+1][0]=0;
182             iptr=Data;
183             if (!strncmp(Dtype,"ASCII",5))
184             {
185                     Idtype=0;
186                     Int_incr=Ncols;
187             }
188             else if (!strncmp(Dtype,"Integer",7))
189             {
190                     Idtype=0;
191                     Int_incr=Ncols;
192             }
193             else if (!strncmp(Dtype,"Real",4))
194             {
195                     Idtype=1;
196                     Int_incr=Ncols*4;
197             }
198             else
199             {
200                     puts("Unknown file type\n");
201                     exit(2);
202             }
203
204             printf(" %d events\n",Nevents);
205             set_ax(0,0);
206             set_ax(1,1);
207             set_ax(2,min(Ncols,2));
208     }
209
210     int *get_data(phloc)
211     int *phloc;
212     {
213             int *read_data();
214             if(Brec_no<0)
215             {
216                     phloc=read_data();
217                     Bincr=0;
218             }
219             else
220             {
221                     phloc+=Reclen_int;
222                     Bincr+=Reclen_int;
223             }
224
```

```
225             if(Bincr>=Ints_perblk)
226             {
227                     phloc=read_data();
228                     Bincr=0;
229             }
230             return(phloc);
231     }
232
233     int *read_data()              /*read next data block, no selection-search*/
234     {
235             int nevs, n,i, col[MCOL], err;
236             int *phloc;
237             char tbuf[BLK];
238             phloc=Data;
239             if(Brec_no<0)
240             {
241                     nevs=min(Nrec_pblk,No_rec);
242                     Brec_no=0;
243             }
244             else
245             {
246                     nevs=min(Nrec_pblk,No_rec-(Brec_no+Nrec_pblk));
247                     Brec_no=Brec_no+nevs;
248             }
249
250             if(Idtype==0)
251             {
252                     n=0;
253                     while (next_line() && n++<=nevs && err>0)
254                             for(i=0; i<Ncols; i++)
255                                     err=sscanf(tbuf+col[i],"%8d",phloc++);
256             }
257             else
258                     err=fread(phloc,sizeof(*phloc),nevs*Reclen_int,infi);
259
260             if(err<=0)
261                     printf("read error: err,Phloc,nevs= %d %d %d \n",
262                       err,phloc,nevs);
263             return(phloc);
264
265     }
266
267     data_rewind(lposi,n1,n2)
268     int n1,n2;
269     long lposi;
270     {
271             int err;
272             err=fseek(infi,lposi,0);
273             if(err< 0)
274                     printf("fseek error: Ldata= %d \n",Ldata);
275             Brec_no=-1;
276
277             No_rec=n1;
278             Reclen_int=n2;
279             Nrec_pblk=MXBUF/Reclen_int;
280             Ints_perblk=Nrec_pblk*Reclen_int;
```

```
281     }
282
283     next_line()     /* Get next line, expand tabs */
284     {
285             char buf[BLK], *s=buf, *d=tbuf;
286             int good;
287             while (good=fgets(buf,BLK,infi))
288             {
289                     if (*s=='\n' || *s=='#')
290                             continue;
291                     _setmem(d,BLK,' ');
292                     while (*s && *s!='\n')
293                     {
294                             if (*s=='\t')
295                             {
296                                     d=tbuf+(d-tbuf+8&0x0ff8);
297                                     s++;
298                             }
299                             else
300                                     *d++=*s++;
301                     }
302                     d[BLK-1]='\0';
303                     break;
304             }
305             return good;
306     }
307     clip_str(str,last)  /* Clip blanks from end of string */
308     char *str;
309     int last;
310     {
311             int j;
312             char c;
313
314             for (j=last; j>0 ; j--) /* Remove trailing blanks */
315             {
316                     c=str[j];
317                     if(c!=' ' && c!='\n' && c!=0x0d && c!=0)
318                             break;
319             }
320             str[j+1]=0;
321     }
322
323     /*      Tree & Plot Subroutines
324      *      ------------------------
325      */
326
327     tree_vers()          /* Add this verse to tree */
328     {
329             int i, j, ok, *ph, x, xy[2];
330             int *get_data();
331
332             data_rewind(Ldata,Nevents,Int_incr);
333             for (i=0; i<Nevents; i++)
334             {
335                     ph=get_data(ph);
336                     ok=1;
```

```
337                     for (j=0; j<Ncols; j++)
338                     {
339                             x=ph[j];
340                             if (Ax_type[j]==1)
341                                     x&=Ax_bin[j];
342                             if (x<Ax_lim[j][0] ||
343                                  x>Ax_lim[j][1]   )
344                                     ok=0;
345                     }
346                     if (ok)
347                     {
348                             if (Ishist==0)
349                             {
350                                     xy[0]=lg_int(ph[off[0]],0);
351                                     xy[1]=lg_int(ph[off[1]],1);
352                             }
353                             else  if (Ishist==1)
354                             {
355                                     xy[0]=to_bin(lg_int(ph[off[0]],0),0);
356                                     xy[1]=1;
357                             }
358                             else
359                             {
360                                     xy[0]=to_bin(lg_int(ph[off[0]],0),0);
361                                     xy[1]=to_bin(lg_int(ph[off[1]],1),1);
362                             }
363                             add_ph(xy);
364                     }
365             }
366     }
367
368     treer_vers()            /* Add this verse to tree */
369                             /* Version for array of real nos. and errors */
370     {
371             int i, j, jax, ok,  xy[2],erx[2],ery[2],erz[2];
372             int *get_data();
373             float x, *ph;
374             long *time, lowtim,hightim;
375
376             lowtim=Axr_lim[0][0];
377             hightim=Axr_lim[0][1];
378
379             data_rewind(Ldata,Nevents,Int_incr);
380             for (i=0; i<Nevents; i++)
381             {
382                     ph=get_data(ph);
383                     ok=1;
384                     time=ph;
385                     if (time[0]< lowtim || time[1] >hightim)
386                             ok=0;
387
388                     for (j=2,jax=1; j<Ncols*2; j+=2,jax++)
389                     {
390                             x=ph[j];
391                             if(ph[j+1]<0)
392                                     continue;
```

```
393                             if (x<Axr_lim[jax][0] || x>Axr_lim[jax][1] )
394                                     ok=0;
395                     }
396                     for (j=0;j<3;j++)
397                             if(off[j]!=0 && ph[off[j]*2+1]<0)
398                                             ok=0;
399
400                     if (ok)
401                     {
402                             if (Ishist==0)
403                             {
404                                     xy[0]=lg_float(&ph[off[0]*2],erx,0,off[0]);
405                                     xy[1]=lg_float(&ph[off[1]*2],ery,1,off[1]);
406                             }
407                             else  if (Ishist==1)
408                             {
409                                     xy[0]=to_bin(lg_float(&ph[off[0]*2],
410                                       erx,0,off[0]),0);
411                                     xy[1]=1;
412                             }
413                             else
414                             {
415                                     xy[0]=to_bin(lg_float(&ph[off[0]*2],
416                                       erx,0,off[0]),0);
417                                     xy[1]=to_bin(lg_float(&ph[off[1]*2],
418                                       ery,1,off[1]),1);
419                             }
420                             add_ph(xy);
421                     }
422             }
423     }
424
425     direct_plot()           /* Plot all points directly */
426     {
427             int i, j, ok, *ph, x, xy[3];
428             int *get_data();
429             Dif_d[0]=0;
430             Dif_d[1]=0;
431             Dif_d[2]=0;
432             phas=0;
433
434             data_rewind(Ldata,Nevents,Int_incr);
435             for (i=0; i<Nevents; i++)
436             {
437                     ph=get_data(ph);
438                     ok=1;
439                     for (j=0; j<Ncols; j++)
440                     {
441                             x=ph[j];
442                             if (Ax_type[j]==1)
443                                     x&=Ax_bin[j];
444                             if (x<Ax_lim[j][0] ||
445                                  x>Ax_lim[j][1]   )
446                                     ok=0;
447                     }
448                     if (ok)
```

```
449                     {
450                             phas++;
451                             xy[0]=lg_int(ph[off[0]],0);
452                             xy[1]=lg_int(ph[off[1]],1);
453                             xy[2]=lg_int(ph[off[2]],2);
454     /*
455                             printf("in dt_p:xy=%d %d %d \n",xy[0],xy[1],xy[2]);
456     */
457                             point(xy[0],xy[1],xy[2],1,1);
458                     }
459             }
460     }
461
462     directr_plot()          /* Plot all points directly */
463                             /* Version for real nos. and errors */
464     {
465             int i, j, jax, ok,  xy[3], erx[2],ery[2],erz[2];
466             int *get_data();
467             float x, *ph, tdiff;
468             long *time, lowtim,hightim;
469
470             Dif_d[0]=0;
471             Dif_d[1]=0;
472             Dif_d[2]=0;
473             phas=0;
474             lowtim=Axr_lim[0][0];
475             hightim=Axr_lim[0][1];
476
477             data_rewind(Ldata,Nevents,Int_incr);
478             for (i=0; i<Nevents; i++)
479             {
480                     ph=get_data(ph);
481                     ok=1;
482                     time=ph;
483                     if (time[0]< lowtim || time[1] >hightim)
484                             ok=0;
485
486                     tdiff=(float)(time[1]-time[0])/2.;
487                     ph[0]=(float)(time[1]+time[0])/2.;
488                     ph[1]=tdiff;
489
490                     for (j=2,jax=1; j<Ncols*2; j+=2,jax++)
491                     {
492                             x=ph[j];
493                             if(ph[j+1]<0)
494                                     continue;
495                             if (x<Axr_lim[jax][0] || x>Axr_lim[jax][1] )
496                                     ok=0;
497                     }
498                     for (j=0;j<3;j++)
499                             if(off[j]!=0 && ph[off[j]*2+1]<0)
500                                     ok=0;
501
502                     if (ok)
503                     {
504                             phas++;
```

```
505                             xy[0]=lg_float(&ph[off[0]*2],erx,0,off[0]);
506                             xy[1]=lg_float(&ph[off[1]*2],ery,1,off[1]);
507                             xy[2]=lg_float(&ph[off[2]*2],erz,2,off[2]);
508                             pointf(xy[0],erx,xy[1],ery,xy[2],erz,1,1);
509
510                     }
511             }
512     }
513     plot_matrix()           /* Plot the matrix */
514     {
515             int savlas, rt;
516             int c, k;
517             send_asc(" CA F 0 DI 0 ");
518             send_asc(" C 6 ");
519             send_asc(" CX ");
520
521             rt=Laser==0?500:520;
522             rt=520; /* always make room for box */
523             savlas=Laser;
524             Laser=0;
525
526             draw_pg(M,rt,400);
527             text_pg(Ax_labl[off[0]]);
528
529             draw_pg(M,rt,385);
530             text_pg(Ax_labl[off[1]]);
531
532             draw_pg(M,rt,370);
533             text_pg(Ax_labl[off[2]]);
534
535             Laser=savlas;
536             if (Ishist)
537             {
538                     sprintf(tbuf,"%d bins",nodes);
539                     draw_pg(M,rt,340);
540                     text_pg(tbuf);
541
542                     sprintf(tbuf,"max=%d",maxn);
543                     draw_pg(M,rt,325);
544                     text_pg(tbuf);
545             }
546
547             if (!Laser)
548                     scale();
549             if (Ishist)
550                     print_ph();     /* This version of "treeprint" traverses
551                                        the tree and plots each point      */
552             else
553                     if(Idtype==1)
554                             directr_plot();
555                     else
556                             direct_plot();
557
558             color_pg(6);
559             if (Laser)
560                     scale();
```

```
561             draw_pg(M,-30,Axtop[1]+10);
562             text_pg(Ax_labl[off[1]]);
563             draw_pg(M,(Axtop[0]>>1)-(strlen(Ax_labl[off[0]])<<2),-36);
564             text_pg(Ax_labl[off[0]]);
565             draw_pg(M,rt,355);
566             sprintf(tbuf,"%d events",phas);
567             text_pg(tbuf);
568     }
569
570     /*
571      *      Menu Selection Subroutines
572      *      ____________________________
573      */
574
575     matr_sel(kw,line,sdm_row,edm_row)       /* Matrix selection menu */
576     int kw, line, sdm_row,edm_row;
577     {
578             int i, k, kt, row=0, left=1;
579             scr_clr();
580     /*
581             printf("in msel:kw,l,sd,ed=%d %d %d %d \n",
582               kw,line,sdm_row,edm_row);
583     */
584             for (i=0; i<25; i++)
585                     scr_str(i,0,"  ",15);
586             scr_attr=0x0b;
587             sprintf(tbuf,"Window %d\0",kw);
588             scr_str(row,left,tbuf,scr_attr);
589             dates(date);
590             scr_str(0,70,date,scr_attr);
591             scr_str(1,left,ptr,scr_attr);
592             scr_str(1,left+20,Header,scr_attr);
593             row=3;
594             for (i=0; i<3; i++)
595             {
596                     if (i==2 && Ishist==1)
597                             break;
598                     sprintf(tbuf,"%c (%s",'x'+i,linlog[w->lf[i]]);
599                     if (i<Ishist)
600                             sprbin(tbuf,i);
601                     strcat(tbuf,") :");
602                     scr_str(row++,left,tbuf,scr_attr);
603                     sprintf(tbuf,"    %s",Ax_labl[off[i]]);
604                     scr_str(row++,left,tbuf,scr_attr);
605             }
606             row=9;
607             for (i=sdm_row; i<edm_row; i++)
608             {
609                     if (Ax_type[i]!=1)
610                     {
611                             sprintf(tbuf,"%9.2e to %9.2e  (%3d)  :%s",Axr_lim[i][0]
    ,
612                                     Axr_lim[i][1],Ax_bin[i],Ax_labl[i]);
613                     }
614                     else
615                     {
```

```
616                             sprintf(tbuf,"%9x to %9x M=%4x  :%s",Ax_lim[i][0],
617                                     Ax_lim[i][1],Ax_bin[i],Ax_labl[i]);
618                     }
619                     scr_str(row++,left,tbuf,scr_attr);
620             }
621             scr_rowcol(line,left);
622     }
623     sprbin(str,ax)   /* Add bin-width descr to string */
624     char *str;
625     int ax;
626     {
627             int x;
628             double y;
629             x=w->zf[ax+3];
630             str+=strlen(str);
631             if (w->lf[ax])
632             {
633                     y=x*0.2708;
634                     sprintf(str,", bin=%6.1lf%%",y);
635             }
636             else
637                     sprintf(str,", bin=%d",x);
638     }
639     list_ax(ax,arrow_row,sd_row,ed_row)       /* List axis options  */
640     int ax, arrow_row,sd_row,ed_row;
641     {
642             int k, row=0, left=1;
643
644             row=0; left=40;
645             scr_attr=12;
646     /*
647             printf("in l_ax:ax,sr,er=%d %d %d \n",ax,sd_row,ed_row);
648     */
649             scr_str(arrow_row,0,"-->",14);
650             for (k=sd_row; k<min(Ncols,ed_row); k++)
651             {
652                     scr_str(row,left-1,"  ",15);
653                     sprintf(tbuf,"%s",Ax_labl[k]);
654                     scr_str(row++,left,tbuf,scr_attr);
655             }
656             if (ax>=1 && ed_row>Ncols)
657             {
658                     scr_str(row,left-1,"  ",15);
659                     scr_str(row++,left,"Histogram",scr_attr);
660             }
661     }
662     set_ax(ax,k)       /* Set axes for selected matrix */
663     int ax, k;
664     {
665             int x1, x2, dx, dd=512;
666             float fx1,fx2;
667             float lgf_plt();
668             float expl0(),logl0(),floor();
669
670             off[ax]=k;
671
```

```
672             if (k==Ncols)
673             {
674                     Ishist=ax;
675                     if (ax==1)
676                             off[2]=Ncols+1;
677             }
678             else if (ax>=Ishist)
679                     Ishist=0;
680
681             psf[ax]=1.;
682             fx1=lgf_plt(Axr_lim[k][0],ax,1);  /* arg=1 is dummy */
683             fx2=lgf_plt(Axr_lim[k][1],ax,1);
684     /*
685             printf("in setax:ax,k,f1,f2,p,x1,2,axr= %d %d %e %e %e %d %d %e %e \n",
686               ax,k, fx1,fx2,psf[ax],x1,x2,Axr_lim[k][0],Axr_lim[k][1]);
687     */
688             if(w->lf[ax]!=1)
689                     psf[ax]=exp10(floor(log10((double)((fx2-fx1)/1000.))));
690             x1=fx1/psf[ax];
691             x2=fx2/psf[ax];
692     /*
693             printf("in setax:ax,k,f1,f2,p,x1,2,axr= %d %d %e %e %e %d %d %e %e \n",
694               ax,k, fx1,fx2,psf[ax],x1,x2,Axr_lim[k][0],Axr_lim[k][1]);
695     */
696             w->p1[ax]=x1;
697             w->p2[ax]=x2;
698             w->po[ax]=x1;
699             dx=x2-x1;
700             if (ax==2)
701                     dd>>=1;
702             if (dx>dd-1)
703                     w->zf[ax]=dx/dd+1;
704             else
705                     w->zf[ax]=4-dd/dx;
706
707             if (Ishist!=1 && off[2]>Ncols)  /* Insure that axis 2 is */
708                     set_ax(2,Ncols);        /*   valid when 1 is no  */
709                                             /*   longer a hist.      */
710             if (Ax_type[k]!=1)
711                     w->zf[ax+3]=Ax_bin[k];
712             else
713                     w->zf[ax+3]=1;
714     }
715
716     plot_stat(kwind)  /* Plot and edit current window status */
717     int kwind;
718     {
719             int k, kt, displ=1, i, j;
720             int sd_row,ed_row, edm_row, sc_off=9;
721             int matr_sel(), list_ax();
722
723     /*
724       sd_row=start bin no. of display (range 0-n)
725       ed_row=end bin no. of display+1 (display range 0-n)
726     */
727
```

```
728             send_asc(" CA DI 1 ");
729             scr_setmode(3);
730             scr_clr();
731
732             if(Sdm_row+sc_off+Ncols+1>Scrn_rows)
733                     edm_row=Scrn_rows - sc_off + Sdm_row;
734             else
735                     edm_row=Ncols+1;
736
737             matr_sel(kwind,0,Sdm_row,edm_row);
738             k=0;
739             while ((k=roll(k,sc_off,1,0, kwind,k, &Sdm_row,&edm_row,matr_sel,
740               0,0,0,0,0,list_ax))>=0)
741             {
742                     if (k==0)       /* Window number */
743                     {
744                             kwind=(kwind&3)+1;
745                             set_wind(kwind);
746                             matr_sel(kwind,k,Sdm_row,edm_row);
747                             for (i=0; i<3; i++)
748                                     set_ax(i,off[i]);
749                             displ=3;
750                     }
751                     else if (k==1)
752                     {
753                             strncpy(tbuf,ptr,20);
754                             strncpy(tbuf+20,Header,48);
755                             tbuf[69]=0;
756                             ledit(tbuf,1,1,68,0);
757                             strncpy(ptr,tbuf,20);
758                             strncpy(Header,tbuf+20,48);
759                             matr_sel(kwind,k,Sdm_row,edm_row);
760                     }
761                     else if (k>=3 && k<=(Ishist==1?6:8))  /* Axes  */
762                     {
763                             i=k-3>>1;
764                             if (k-3&1)      /* Axis name */
765                             {
766                                     if(off[i]-4<0)
767                                             sd_row=0;
768                                     else
769                                             sd_row=off[i]-4;
770                                     if(sd_row+Ncols+(i?1:0)>Scrn_rows)
771                                             ed_row=Scrn_rows+sd_row;
772                                     else
773                                             ed_row=Ncols+(i?1:0);
774
775                                     list_ax(i,k,sd_row,ed_row);
776                                     kt=roll(off[i]-sd_row,0,i?1:0,39,
777                                       i,k,&sd_row,&ed_row,list_ax,
778                                       1,kwind,k,&Sdm_row,&edm_row,matr_sel);
779                                     if ((kt+=sd_row)>=0 && kt!=off[i])
780                                     {
781                                             set_ax(i,kt);
782                                             displ=3;
783                                     }
```

```
784                             }
785                             else            /* Lin/log */
786                             {
787                                     w->lf[i]^=1;
788                                     set_ax(i,off[i]);
789                                     displ=3;
790                             }
791                             matr_sel(kwind,k,Sdm_row,edm_row);
792                     }
793                     else if (k>=sc_off && k+Sdm_row<Ncols+sc_off+1)      /* Axis li
        mits */
794                     {
795                             i=k-sc_off+Sdm_row;
796                             if (Ax_type[i]!=1)
797                             {
798                                     sprintf(tbuf,"%9.2e to %9.2e  (%3d)  :%-39s",
799                                       Axr_lim[i][0],Axr_lim[i][1],
800                                       Ax_bin[i],Ax_labl[i]);
801                                     if (ledit(tbuf,k,1,58,16))
802                                     {
803                                             sscanf(tbuf,"%9e",&Axr_lim[i][0]);
804                                             sscanf(tbuf+13,"%9e",&Axr_lim[i][1]);
805                                             Ax_lim[i][0]=Axr_lim[i][0];
806                                             Ax_lim[i][1]=Axr_lim[i][1];
807                                             sscanf(tbuf+24,"%3d",&Ax_bin[i]);
808                                             Ax_bin[i]=max(1,Ax_bin[i]);
809                                             strcpy(Ax_labl[i],tbuf+32);
810                                             clip_str(Ax_labl[i],L_LABL-2);
811                                             displ=3;
812                                             for (j=0; j<3; j++)
813                                                     if (i==off[j])
814                                                             set_ax(j,i);
815                                     }
816                             }
817                             else
818                             {
819                                     sprintf(tbuf,"%9x to %9x M=%4x  :%-39s",Ax_lim[
        i][0],
820                                             Ax_lim[i][1],Ax_bin[i],Ax_labl[i]);
821                                     if (ledit(tbuf,k,1,58,0))
822                                     {
823                                             sscanf(tbuf,"%9x",Ax_lim[i]);
824                                             sscanf(tbuf+13,"%8x",&Ax_lim[i][1]);
825                                             sscanf(tbuf+25,"%4x",&Ax_bin[i]);
826                                             strcpy(Ax_labl[i],tbuf+32);
827                                             clip_str(Ax_labl[i],L_LABL-2);
828                                             displ=3;
829                                             for (j=0; j<3; j++)
830                                                     if (i==off[j])
831                                                             set_ax(j,i);
832                                     }
833                             }
834                             matr_sel(kwind,k,Sdm_row,edm_row);
835                     }
836             }
837             scr_attr=0x1e;
```

```
838             send_asc(" DI 0 CX ");
839             return(displ);
840     }
841
842     roll(row,sc_off,hist_add,left, arg1,arg2,sd_row,ed_row,d_func,
843       iflag,arg3,arg4,asd,aed,a_func)
844     int row,sc_off,hist_add,left;
845     int arg1,arg2, *sd_row,*ed_row;
846     int iflag,arg3,arg4, *asd,*aed;
847     int (*d_func)(), (*a_func)();
848     /*
849        The arguments to roll are:
850             row             current screen display row
851             sc_off          scroll offset
852                              (offset of scrolling lines from top of screen)
853             hist_add        =0 if no histogram line, =1 if histogram line
854             left             left margin of current display
855             arg1,arg2        arguments to window redisplay subroutine *d_func
856             *sd_row          sequence no. of start quantity (column) displayed (0-n)
857             *ed_row          sequence no.+1 of end quantity (column) displayed (0-n)
858             *d_func          subroutine to redisplay window
859
860        Roll also uses the global variables Ncols and Scrn_rows.
861
862        Roll defines the variable(s):
863             col_no           quantity no. associated with row (0-n)
864             scroll_sp        no. of data quantities displayable in scrolled display
865     */
866     {
867             int c, col_no,scroll_sp;
868
869             scroll_sp=min(Scrn_rows-sc_off,Ncols+hist_add);
870             scr_rowcol(row,left);
871             while ( (c=key())!=0x0d )       /* Return on  CR */
872             {
873                     if (c==K_DN)                /* Step down */
874                     {
875                             col_no=++row+*sd_row-sc_off;
876                             if (col_no>=Ncols+hist_add)
877                             {
878                                     --row;
879                             }
880                             else if(col_no>=*ed_row)
881                             {
882                                     scr_clr();
883                                     if (iflag)
884                                             (*a_func)(arg3,arg4, *asd,*aed);
885                                     (*d_func)(arg1,arg2, ++*sd_row,++*ed_row);
886                                     --row;
887                             }
888                             scr_rowcol(row,left);
889                     }
890                     else if (c==K_UP)            /* Step up */
891                     {
892                             --row;
893                             if(sc_off>0)
```

```
894                             {
895                                     if(row<0)
896                                     {
897                                             row=0;
898                                     }
899                             }
900                             else if(row<0)
901                             {
902                                     if(*sd_row>0)
903                                     {
904                                             scr_clr();
905                                             if (iflag)
906                                                     (*a_func)(arg3,arg4, *asd,*aed)
        ;
907                                             (*d_func)(arg1,arg2, --*sd_row,--*ed_ro
        w);
908                                     }
909                                     row=0;
910                             }
911                             scr_rowcol(row,left);
912                     }
913                     else if (c==K_PGDN)             /* Block down, axis limits */
914                     {
915                             *ed_row=min(*ed_row+scroll_sp-2,Ncols+hist_add);
916                             *sd_row=*ed_row-scroll_sp;
917                             scr_clr();
918
919                             if(iflag)
920                             {
921                                     (*a_func)(arg3,arg4, *asd,*aed);
922                             }
923                             (*d_func)(arg1,arg2, *sd_row,*ed_row);
924
925                             if(row-sc_off+*sd_row>*ed_row && row>=sc_off)
926                             {
927                                     row=sc_off+(*ed_row-*sd_row)-1;
928                             }
929                             scr_rowcol(row,left);
930                     }
931                     else if (c==K_PGUP)             /* Block up, axis limits */
932                     {
933                             *sd_row=max(0,*sd_row-scroll_sp+1);
934                             *ed_row=*sd_row+scroll_sp;
935                             scr_clr();
936                             if(iflag)
937                                     (*a_func)(arg3,arg4, *asd,*aed);
938                             (*d_func)(arg1,arg2, *sd_row,*ed_row);
939                             scr_rowcol(row,left);
940                     }
941                     else if (c==3)                /* ^C - Exit  */
942                     {
943                             send_asc("CA DI 1 ");
944                             exit(1);
945                     }
946                     else if (c==0x1b || c==K_F1) /* Esc or F1 return */
947                     {
```

```
948                             return(-1);
949                     }
950
951             }
952             return(row);
953     }
954     ledit(str,row,c1,c2,ca)  /* Line edit a field on screen */
955     char *str;
956     int row, c1, c2, ca;
957     {
958             int c, col;
959             col=c1;
960             scr_attr=14;
961             scr_str(row,col,str,scr_attr);
962             scr_rowcol(row,col);
963             while ( (c=key())!=K_CR )       /* Return on  CR */
964             {
965                     if (c==K_RT)
966                     {
967                             scr_rowcol(row,col=col==c2?c1:col+1);
968                     }
969                     else if (c==K_LEFT)
970                     {
971                             scr_rowcol(row,col=col==c1?c2:col-1);
972                     }
973                     else if (c==K_TBL)
974                     {
975                             scr_rowcol(row,col=max(c1,col-1&0xfff8));
976                     }
977                     else if (c==K_TBR)
978                     {
979                             scr_rowcol(row,col=min(c2,col+8&0xfff8));
980                     }
981                     else if (isdigit(c) || c==' '|| c=='-' || c=='.'
982                       || c=='e' || c=='E')
983                     {
984                             str[col-c1]=c;
985                             scr_str(row,c1,str,scr_attr);
986                             scr_rowcol(row,col=col==c2?c2:col+1);
987                     }
988                     else if (col>ca && isprint(c))
989                     {
990                             str[col-c1]=c;
991                             scr_str(row,c1,str,scr_attr);
992                             scr_rowcol(row,col=col==c2?c2:col+1);
993                     }
994                     else if (c==3)             /* ^C - Exit  */
995                     {
996                             send_asc("CA DI 1 ");
997                             exit(1);
998                     }
999                     else if (c==0x1b)            /* Esc return */
1000                    {
1001                            return(0);
1002                    }
1003            }
```

```
1004            return(1);
1005    }
1006
1007
1008    scr_str(row,col,str,colr)       /* Put string on screen */
1009    int row, col, colr;
1010    char *str;
1011    {
1012            static int offs, len;
1013            offs=(80*row+col)<<1;
1014            len=max(1,min(strlen(str),79-col));
1015    #asm
1016
1017    SSEG    equ     0b800h                  ; screen seg
1018
1019            mov     cx, word scr_str_len_
1020            mov     di, word scr_str_offs_
1021            mov     si, [bp+8]                      ; str
1022            mov     ax, SSEG
1023            mov     es, ax
1024            mov     ah, [bp+10]                     ; colr
1025            cld
1026    scr_1:
1027            lodsb
1028            stosw
1029            loop    scr_1
1030    #
1031    }
1032
1033    scr_get(row,col,str,len)        /* Get string from screen */
1034    int row, col, len;
1035    char *str;
1036    {
1037            static int offs, len;
1038            offs=(80*row+col)<<1;
1039            len=max(1,len);
1040    #asm
1041
1042
1043            mov     cx, [bp+10]                     ; len
1044            mov     si, word scr_get_offs_
1045            mov     di, [bp+8]                      ; str
1046            push    ds
1047            push    ds
1048            pop     es
1049            mov     ax, SSEG
1050            mov     ds, ax
1051            cld
1052    scr_2:
1053            lodsw
1054            stosb
1055            loop    scr_2
1056            xor     ax,ax
1057            stosb                   ; null terminator
1058            pop     ds
1059    #
```

```
1060    }
1061
1062    /*
1063    printr(iptr,i)
1064    float *iptr;
1065    int i;
1066    {
1067            int j,k;
1068            float *iiptr;
1069
1070            printf("listing of file of real nos at printr as reals.  \n");
1071            printf("  %x %x %x %x \n",*(iptr),*(iptr+1),*(iptr+2),*(iptr+3));
1072            printf("  %x %x %x %x \n",*(iptr+4),*(iptr+5),*(iptr+6),*(iptr+7));
1073            printf("  %x %x %x %x \n",*(iptr+8),*(iptr+9),*(iptr+10),*(iptr+11));
1074            printf("  %x %x %x %x \n",*(iptr+12),*(iptr+13),*(iptr+14),*(iptr+15));
1075            printf("  %x %x %x %x \n",*(iptr+16),*(iptr+17),*(iptr+18),*(iptr+19));
1076
1077            iiptr=iptr;
1078            for (j=0;j<Nevents;j++,iiptr+=Ncols*2)
1079            {
1080                    for (k=0;k<Ncols*2;k+=2)
1081                            printf("%f +- %f, ",*(iiptr+k),*(iiptr+k+1));
1082                    puts("\n");
1083            }
1084
1085    }
1086    */
1087
1088
1089
```

Blank Spacer Page

```
/*
 *      Map.c - Module defines the mapping from plot space
 *              to display space and draws into the PGA.
 *
 *              Version with added(modified) routines to handle
 *              real nos. by R. McGuire.
 */


#define M       0x10    /* Move */
#define MR      0x11    /* Move Relative */
#define D       0x28    /* Draw */
#define DR      0x29    /* Draw Relative */
#define RR      0x35    /* Rectangle Relative */
#define CI      0x38    /* Circle */
#define C       0x06    /* Color */
#define PT      0x08    /* Point */
#define T       0x80    /* Text  */
#define TJ      0x85    /* Text Justification */

#define NWIND      5            /* Number of windows            */

#include "windo.h"
extern struct window *w;

extern char Zbox;
extern int Ishist;
extern float psf[3];

extern int Laser;
extern char *p_las; /* Laser printer file name */

extern int Dif_d[3]; /* Old-new displ coords (panning) */

char txt_jst[5][3]={TJ,2,3, TJ,3,2, TJ,1,1,
                    TJ,3,3, TJ,1,2}; /* text justification */
char *dbuf[32];

#include <tabl.h>       /* tabl[i]=256*log2(i+256) for 0<=i<=255 */

int flip;       /* Control axis orientation */
int tic[14]={1,2,5,10,20,50,100,200,500,1000,2000,5000,10000,20000};

int lten[25]=  {-10205,-9355,-8504,-7654,-6803,
                -5953,-5102,-4252,-3402,-2551,-1701,-850,
                0,850,1701,2551,3402,4252,5102,5953,6803,7654,
                8504,9355,10205};
int lsub[11]={0,150,256,406,512,594,662,719,768,812,850};

int Ltic=10, Mtic=5, Stic=3;
int Axtop[3];

scale()         /* Select scales via window */
{
        int ax;
        if (w->lf[0])
```

```
57                      log_scale(0,0);
58              else
59                      lin_scale(0,0);
60              if (w->lf[1])
61                      log_scale(1,0);
62              else
63                      lin_scale(1,0);
64              if (1 /*Laser*/)   /* Draw full box */
65              {
66                      if (w->lf[0])
67                              log_scale(0,Axtop[1]);
68                      else
69                              lin_scale(0,Axtop[1]);
70                      if (w->lf[1])
71                              log_scale(1,Axtop[0]);
72                      else
73                              lin_scale(1,Axtop[0]);
74              }
75              if (w->cse)
76              {
77                      if (w->lf[2])
78                              log_scale(2,0);
79                      else
80                              lin_scale(2,0);
81              }
82      }
83
84      /*
85       *      Log coordinates are mapped to 'plot' coord. space via:
86       *              plt=256*log2(x)
87       *      This scaling allows x to range over 1.0E-12<x<1.0E12
88       *      with about 0.2% pixel resolution.  The plot coords can
89       *      be zoomed to 'display' coords. as required.
90       */
91
92      log_scale(ax,side)          /* Label log axis */
93      int ax, side;
94      {
95              int k, l, top, dtop, bot, range, x, xs;
96              int ltic, mtic, stic;
97              flip=ax;
98              bot=max(w->pl[ax],w->po[ax]);
99              dtop=to_plt(w->dc[ax]<<1,ax);
100             top=min(w->p2[ax],dtop);
101     /*
102             printf("in log_scale: bot,dtop,top,pl,po= %d %d %d %d %d \n",
103               bot,dtop,top,w->pl[ax],w->po[ax]);
104     */
105             if (side==0)
106                     Axtop[ax]=to_dis(top,ax);
107             if (side)
108             {
109                     ltic=-Ltic; mtic=-Mtic; stic=-Stic;
110             }
111             else
112             {
```

```
113                 ltic=Ltic; mtic=Mtic; stic=Stic;
114             }
115             range=dtop-bot;
116             if (ax==2)
117                 col_scale();
118             else
119             {
120                 color_pg(6);
121                 flip_pg(M,0,side);
122                 flip_pg(D,to_dis(top,ax),side);
123             }
124             color_pg(6);
125             flip_pg(M,0,side);
126             for (k=0; k<25; k++)
127             {
128                 x=lten[k];
129                 if (x<bot-848)
130                     continue;
131                 if (x>top)
132                     break;
133                 if (x>bot)
134                 {
135                     flip_pg(M,to_dis(x,ax),side);  /* *10 tic */
136                     flip_pg(DR,0,ltic);
137                     flip_pg(MR,0,-ltic);
138                     if (side==0 &&
139                           (range<8192 || k&1==0) )  /* *10 label */
140                     {
141                       flip_pg(MR,0,ax>0?-24:-12);
142                       draw_pg(MR,-12,-8);
143                       text_pg("10");          /* 10 */
144                       draw_pg(MR,20,6);
145                       sprintf(dbuf,"%d\0",k-12);
146                       text_pg(dbuf);         /* Power */
147                     }
148                 }
149                 if (range>8192)
150                     continue;
151                 xs=x+lsub[5];
152                 if (range>4096 && xs>bot && xs<top)
153                 {
154                     flip_pg(M,to_dis(xs,ax),side);
155                     flip_pg(DR,0,mtic);       /* *5 tic */
156                     continue;
157                 }
158                 for (l=2; l<9; l+=2)
159                 {
160                     xs=x+lsub[l];
161                     if (xs<bot || xs>top)
162                         continue;
163                     flip_pg(M,to_dis(xs,ax),side);
164                     flip_pg(DR,0,mtic);       /* *2 tic */
165                 }
166                 if (range>2048)
167                     continue;
168                 for (l=1; l<10; l+=2)
```

```
169                     {
170                             xs=x+lsub[l];
171                             if (xs<bot || xs>top)
172                                     continue;
173                             flip_pg(M,to_dis(xs,ax),side);
174                             flip_pg(DR,0,stic);     /* units tic */
175                     }
176                     send(txt_jst[2],3);
177             }
178     }
179     lg_int(n,ax)      /* Take log2 of int & map to displ. coords. */
180     int n, ax;        /*  iff axis is log  */
181     {
182             return(to_dis(lg_plt(n,ax),ax));
183     }
184     lg_plt(n,ax)      /* Take log2 of int & map to plot coords. */
185     int n, ax;        /*  iff axis is log  */
186     {
187             int pow2;
188             if (w->lf[ax]!=1)
189                     return(n/(int)psf[ax]);
190             if (n<=0)
191                     return(-100);
192             pow2=0;
193             while (n<256)
194             {
195                     n<<=1; pow2-=256;
196             }
197             while (n>511)
198             {
199                     n>>=1; pow2+=256;
200             }
201             return(tabl[n-256]+pow2);
202     }
203     lin_scale(ax,side)         /* Label linear axis */
204     int ax;
205     {
206             int jt, x, xl, xb, x2, dx2, t;
207             int tax, sl;
208             int ltic, mtic, stic;
209             float tx,round_off(),fabs();
210
211             tax=min(ax,1);
212             flip=ax;
213             xl=max(w->pl[ax],w->po[ax]);
214             dx2=to_plt(w->dc[ax]<<1,ax);
215             x2=min(w->p2[ax],dx2);
216             if (side==0)
217                     Axtop[ax]=to_dis(x2,ax);
218             if (side)
219             {
220                     ltic=-Ltic; mtic=-Mtic;
221             }
222             else
223             {
224                     ltic=Ltic; mtic=Mtic;
```

```
225             }
226             jt=max(0,indx(dx2-x1,tic,14)-3);
227             t=tic[jt];
228             xb=(x1/t)*t;
229             if (ax==2)
230                     col_scale();
231             else
232             {
233                     color_pg(6);
234                     flip_pg(M,0,side);
235                     flip_pg(D,to_dis(x2,ax),side);
236             }
237             color_pg(6);
238             flip_pg(M,0,side);
239             for(x=xb; x<=x2; x+=t)
240             {
241                     if (x<x1)
242                             continue;
243                     flip_pg(M,to_dis(x,ax),side);
244                     flip_pg(DR,0,ltic);              /* Tic   */
245                     if (side)
246                             continue;
247                     flip_pg(MR,0,-ltic-4);
248                     tx=round_off(x*psf[ax]);
249                     if(fabs((double)tx)>10000. || fabs((double)tx) <.0001)
250                             sprintf(dbuf,"%6.2e\0",tx);
251                     else
252                             sprintf(dbuf,"%2g\0",tx);
253                     sl=-strlen(dbuf);
254                     if (ax)
255                             draw_pg(MR,4*sl-5,-4);
256                     else
257                             draw_pg(MR,sl<<2,-10);
258                     text_pg(dbuf);                   /* Label */
259             }
260             send(txt_jst[2],3);
261             if (jt<0)
262                     return;
263             xb-=t;
264             t=tic[jt-2];
265             for(x=xb; x<=x2; x+=t)
266             {
267                     if (x<x1)
268                             continue;
269                     flip_pg(M,to_dis(x,ax),side);
270                     flip_pg(DR,0,mtic);              /* Minor tic */
271             }
272     }
273
274     float round_off(x)
275     float x;
276     {
277             float rx;
278             int *iptr,is,j;
279             char *cptr,cs;
280
```

```
281             rx=x;
282             iptr=&rx;
283             cptr=&rx;
284     /*
285             printf("%g  ,  ",rx);
286             for (j=0;j<8;j++)
287                     printf("%02x ",cptr[j]);
288             printf ("\n");
289     */
290             if ((is=(((*iptr>>3)<<3)||0x0007)) == 0xFFFF)
291             {
292     /*
293                     printf ("*iptr,is,rx= %04x %04x %08x \n",*iptr,is,rx);
294     */
295                     *iptr=is;
296                     rx=rx+0x01000000;
297             }
298             else
299                     *iptr=(*iptr>>3)<<3;
300
301             return(rx);
302     }
303
304     col_scale()     /* Plot color scale */
305     {
306             int i, x;
307             for (i=0; i<256; i++)
308             {
309                     color_pg(i);
310                     draw_pg(M,550,i);
311                     draw_pg(DR,Zbox?(i+8>>4)+1:2,0);
312             }
313     }
314
315     to_dis(plt,axis)   /* Plot coord -> display coord  */
316     int plt, axis;
317     {
318             int zoom;
319
320     /*      printf ("in to_d(int): plt=%d \n", plt);    */
321             zoom=w->zf[axis];
322             if (zoom>1)
323                     return((plt-w->po[axis])/zoom);
324             else if (zoom<1)
325                     return((plt-w->po[axis])*(2-zoom));
326             else
327                     return(plt-w->po[axis]);
328     }
329     to_plt(dis,axis)   /* Display coord -> Plot coord  */
330     int dis, axis;
331     {
332             int zoom;
333             zoom=w->zf[axis];
334             if (zoom>1)
335                     return(dis*zoom+w->po[axis]);
336             else if (zoom<1)
```

```
337                     return(dis/(2-zoom)+w->po[axis]);
338             else
339                     return(dis+w->po[axis]);
340     }
341
342     to_bin(dis,axis)   /* Display coord -> Histo. bin coord.  */
343     int dis, axis;
344     {
345             int zoom, plt;
346             plt=to_plt(dis,axis);
347             zoom=max(w->zf[axis],w->zf[axis+3]);
348             if (zoom>1)
349                     return((plt-w->po[axis])/zoom);
350             else
351                     return(plt-w->po[axis]);
352     }
353     bin_to_dis(bin,axis)   /* Histo bin coord -> displ coord  */
354     int bin, axis;
355     {
356             int zoom, plt;
357             zoom=max(w->zf[axis],w->zf[axis+3]);
358             if (zoom>1)
359                     plt=bin*zoom+w->po[axis];
360             else
361                     plt=bin+w->po[axis];
362             return(to_dis(plt,axis));
363     }
364
365     line(x0,y0,x1,y1)  /* Draw line */
366     int x0,y0,x1,y1;
367     {
368             draw_pg(M,x0,y0);
369             draw_pg(D,x1,y1);
370     }
371     point(x,y,c,lx,ly)       /* Plot point in color c on PGA */
372     int x, y, c, lx, ly;     /* x,y are old Display coords.  */
373     {
374             char  pt=PT;
375             int wid, wid2;
376             int zoom;
377             if (c<0)
378                     return;
379             if (c>247)
380                     c=247;
381             x+=Dif_d[0];    /* Correct for shifts since binning */
382             if (Ishist!=1)
383                     y+=Dif_d[1];
384             if (x<0 || x>Axtop[0])
385                     return;
386             if (y<0 || (y>Axtop[1] && Ishist!=1))
387                     return;
388             draw_pg(M,x,y);
389             color_pg(c+8);
390
391             if (Ishist==1)
392             {
```

```
393     /*              draw_pg(D,x,1);
394     */
395                     if (y>0)
396                             draw_pg(RR,bin_to_dis(1,0),-y);
397             }
398             else if (Zbox)
399             {
400                     wid=(c>>4)+1;
401                     draw_pg(CI,wid);
402
403             }
404             else
405             {
406                     draw_pg(PT,0,0);
407                     if (w->zf[0]<1 && w->lf[0]!=1 && lx) /* Extend x length */
408                     {
409                             if (w->zf[1]<1 && w->lf[1]!=1 && ly) /* y length */
410                             {
411                                     draw_pg(RR,1,1);
412                             }
413                     }
414             }
415     }
416
417     /* Revised version of routines to handle floating pt. (real*4) nos.
418
419        Note as of 8/20/86:  It appears that the scaling between plot coordinates
420        (which I assume are still in "physical units") and display coordinates
421        (which I assume are in pixels of the PG screen) must all be buried in the
422        "zoom" factor, although this seems a bit inconsistent with how I thought
423        the zoom factor was being handled (i.e. zoom=1 for the nominal 1st plot
424        scaling.  On this basis, the conversion from floating point to integer
425        display coordinates must take place in to_dis and to_plt to avoid losing
426        floating resolution by prematurely converting to int format.
427     */
428
429     lg_float(f,err,ax,indx)  /* Take log2 of float (iff axis log), map to displ. co
       ords. */
430     int ax, err[2],indx;
431     float f[2];
432     {
433             float lgf_plt(), g;
434             g=lgf_plt(f[0],ax,indx);
435             return(tof_dis(g,f,err,ax,indx));
436     }
437
438     float lgf_plt(f,ax,indx)
439     int ax,indx;
440     float f;
441     {
442             float g,h, log();
443             long *lptr;
444
445             if (w->lf[ax]!=1)
446             {
447                     return(f/psf[ax]);
```

```
448
449     /*              if (indx !=0)
450                             return(f/psf[ax]);
451                     else
452                     {
453                             h=f;
454                             lptr=&h;
455                             g=*lptr/(long)psf[ax];
456                             printf("f,h,*lptr,psf[ax],g= %f %f %ld %f %f \n",
457                               f,h,*lptr,psf[ax],g);
458                             return(g);
459                     }
460     */
461             }
462
463             if (f<=0.)
464                     return(-100.);
465
466             g= 369.3299305*log((double)f);
467             return(g);
468     }
469
470     tof_dis(plt,f,err,axis,indx)   /* Plot coord -> display coord  */
471     int axis, err[2],indx;
472     float plt, f[2];
473     {
474             float zoom, origin, errf[2], rat;
475             float divide_ch(),lgf_plt();
476             int dis, j;
477             zoom=w->zf[axis];
478             origin=w->po[axis];
479
480             if(w->lf[axis])
481             {
482                     if(plt ==0.)
483                     {
484                             rat=0.;
485                             errf[0]=0.;
486                             errf[1]=lgf_plt(f[0]+f[1],axis,indx);
487                     }
488                     else
489                     {
490                             rat=divide_ch(f[1],f[0]);
491                             errf[0]=(1.-rat)*plt;
492                             errf[1]=(1.+rat)*plt;
493                     }
494             }
495             else
496             {
497                     errf[1]=lgf_plt(f[1],axis,indx);
498                     {
499                             errf[0]=plt-errf[1];
500                             errf[1]=plt+errf[1];
501                     }
502     /*
503                     if(indx !=0)
```

```
504                     {
505                             errf[0]=plt-errf[1];
506                             errf[1]=plt+errf[1];
507                     }
508                     else
509                     {
510                             errf[0]=plt;
511                             plt=(errf[0]+errf[1])/2;
512                     }
513     */
514             }
515
516             if (zoom>1.)
517             {
518                     dis=((plt-origin)/zoom);
519                     for(j=0;j<2;j++)
520                             err[j]=((errf[j]-origin)/zoom);
521             }
522             else if (zoom<1.)
523             {
524                     dis=((plt-origin)*(2.-zoom));
525                     for(j=0;j<2;j++)
526                             err[j]=(errf[j]-origin)*(2.-zoom);
527             }
528             else
529             {
530                     dis=plt-origin;
531                     for(j=0;j<2;j++)
532                             err[j]=errf[j]-origin;
533             }
534     /*
535             printf ("r,o,z,plt=%f %f %f %f \n",rat,origin,zoom,plt);
536             printf ("f,errf=%f %f %f %f \n",f[0],f[1],errf[0],errf[1]);
537             printf ("dis,err=%d %d %d \n",dis,err[0],err[1]);
538     */
539             return(dis);
540     }
541
542     float divide_ch (x,y)
543     float x,y;
544     {
545             float rat;
546             if(x==0.)
547                     rat=0.;
548             else if(y==0.)
549                     rat=1.e10;
550             else
551                     rat=x/y;
552             return(rat);
553     }
554
555     pointf (x,erx,y,ery,c,erc,lx,ly)            /* Plot point in color c on PGA */
556     int x,y,c, lx,ly, erx[2],ery[2],erc[2];     /* x,y are old Display coords.  */
557     {
558             char  pt=PT;
559             int wid, wid2;
```

```
560             int zoom;
561             if (c<0)
562                     return;
563             if (c>247)
564                     c=247;
565             x+=Dif_d[0];    /* Correct for shifts since binning */
566             erx[0]+=Dif_d[0];
567             erx[1]+=Dif_d[0];
568             if (Ishist!=1)
569             {
570                     y+=Dif_d[1];
571                     ery[0]+=Dif_d[1];
572                     ery[1]+=Dif_d[1];
573             }
574             if (x<0 || x>Axtop[0])
575                     return;
576             if (y<0 || (y>Axtop[1] && Ishist!=1))
577                     return;
578
579             if (erx[0]<0)
580                     erx[0]=0;
581             if (erx[1]>Axtop[0])
582                     erx[1]=Axtop[0];
583             if (Ishist!=1)
584             {
585                     if (ery[0]<0)
586                             ery[0]=0;
587                     if (ery[1]>Axtop[1])
588                             ery[1]=Axtop[1];
589             }
590
591             draw_pg(M,x,y);
592             color_pg(c+8);
593
594             if (Ishist==1)
595             {
596                     if (y>0)
597                             draw_pg(RR,bin_to_dis(1,0),-y);
598             }
599             else if (Zbox)
600             {
601                     wid=(c>>4)+1;
602                     draw_pg(CI,wid);
603                     line(erx[0],y,erx[1],y);
604                     line(x,ery[0],x,ery[1]);
605             }
606             else
607             {
608                     draw_pg(PT,0,0);
609                     if (w->zf[0]<1 && w->lf[0]!=1 && lx) /* Extend x length */
610                     {
611                             if (w->zf[1]<1 && w->lf[1]!=1 && ly) /* y length */
612                             {
613                                     draw_pg(RR,1,1);
614                             }
615                     }
```

```
616             }
617     }
618
619     draw_pg(cmd,x,y)   /* Send a move/draw cmd to PGA */
620     char cmd;
621     int x, y;
622     {
623             int nul=0;
624             send(&cmd,1);
625             if (cmd!=PT)
626             {
627                     send(&x,2);
628                     send(&nul,2);
629                     if (cmd!=CI)
630                     {
631                             send(&y,2);
632                             send(&nul,2);
633                     }
634             }
635             if (Laser)
636                     draw_can(cmd,x,y);
637     }
638     flip_pg(cmd,x,y)   /* Send cmd with axis reversal */
639     char cmd;
640     int x, y;
641     {
642             if (flip==2)
643             {
644                     if (cmd==M || cmd==D)
645
646                             draw_pg(cmd,y+540,x+8);
647                     else
648                             draw_pg(cmd,y,x);
649             }
650             else if (flip==1)
651                     draw_pg(cmd,y,x);
652             else
653                     draw_pg(cmd,x,y);
654     }
655     text_pg(str)    /* Send text string to PGA */
656     char *str;
657     {
658             char cmd=T, quote='"';
659             send(&cmd,1);
660             send(&quote,1);
661             send(str,strlen(str));
662             send(&quote,1);
663             if (Laser)
664                     text_can(str);
665     }
666     color_pg(color) /* Send color to PGA */
667     int color;
668     {
669             char cmd[2];
670             *cmd=C;
671             cmd[1]=color;
```

```
672             send(cmd,2);
673             if (Laser)
674                     color_can(color);
675     }
676
677     send_asc(buf)   /* Send ASCII string to PGA */
678     char *buf;
679     {
680             send(buf,strlen(buf));
681     }
682     send(buf,n)     /* Send binary string to PGA */
683     char *buf;
684     int n;
685     {
686             char cwr[9];
687             int wrt, nxt, rd, i;
688             i=0;
689             while (i<n)
690             {
691                     _lmove(9,0,0xc630,cwr,_showds());
692                     wrt=cwr[0]; rd=cwr[1];
693                     nxt=wrt+1&0x00ff;
694                     if (nxt!=rd)
695                     {
696                             _poke(buf[i++],wrt,0xc600);
697                             _poke(nxt,0,0xc630);
698                     }
699             }
700     }
701     /*
702             Laser Printer Subroutines
703     */
704     #define ESC         '\033'              /* 0x1b */
705     #define CSI         '\233'              /* 0x9b */
706     #define IS1         '\037'              /* 0x1f */
707     #define IS2         '\036'              /* 0x1e */
708     #define ISO_MODE    "\033;"             /* C48 (Diablo) */
709     #define SOFT_RESET  "\033<"             /* C 1 (ISO) */
710     #define PAINT_FULL  "\2332&z"           /* C 5 (ISO) */
711     #define ENTER_VECT  "\233&}"            /* C 6 (ISO) */
712     #define BEGIN_PICT  "\043\036"          /* C 1 (VDM) */
713     #define BEGIN_BODY  "\044\036"          /* C 2 (VDM) */
714     #define SCALE_MODE  "!0\043\061\036"    /* C11 (VDM) =300dpi, x1 */
715     #define END_PICT    "\045\036"          /* C 3 (VDM) */
716
717     #define ORIGIN      "}\042"             /* C54 (VDM) x,y */
718     #define SOLID_LINE  "E10\036"           /* C31 (VDM) line type 0 */
719     #define THIN_LINE   "F1\042\036"        /* C32 (VDM) line width -2 */
720     #define MARK_TYPE   "A"                 /* C34 (VDM) Index */
721     #define MARK_SIZE   "B"                 /* C35 (VDM) Size */
722     #define FILL_STYLE  "I21\036"           /* C36 (VDM) interior 2 */
723     #define POLY_LINE   "\061"              /* C52 (VDM) x,y,...*/
724     #define CIRCLE      "\065"              /* C53 (VDM) x,y,R */
725     #define POLY_MARK   "\060"              /* C56 (VDM) x,y,...*/
726     #define POLYGON     "\062"              /* C57 (VDM) x,y,...*/
727     #define ELLIPSE     "}\067"             /* C58 (VDM) Cx,Cy,Rx,Ry,A,Q,T */
```

```
728     #define TO_TEXT   "}p"                  /* C71 (VDM) x,y */
729
730     int c_x0=0, c_y0=0;
731     char cbuf[160], *canbuf;
732
733     plot_Laser()    /* Copy a plot to the Laser printer */
734     {
735             int i, j;
736             unsigned l;
737            Laser=fopen(p_las,"a");
738             fputs(SOFT_RESET,Laser);
739             fputs(PAINT_FULL,Laser);
740             for (i=0, l=0; i<25; i++, l+=160)
741             {
742                     fputs("\n",Laser);
743                     _lmove(160,l,0xb800,cbuf,_showds());
744                     for (j=0; j<160 && isprint(cbuf[j]); j+=2)
745                             cbuf[j>>1]=cbuf[j];
746                     cbuf[j>>1]=0;
747                     cbuf[80]=0;
748                     fputs(cbuf,Laser);
749             }
750             fputs(ENTER_VECT,Laser);
751             fputs(BEGIN_PICT,Laser);
752             fputs(SCALE_MODE,Laser);
753             fputs(BEGIN_BODY,Laser);
754             canbuf=cbuf;
755             fputs(ORIGIN    ,Laser);
756             put_int(300);
757             put_int(3000);
758             *canbuf++=IS2;
759             *canbuf++=0;
760             fputs(cbuf,Laser);
761
762             canbuf=cbuf;
763             fputs(MARK_TYPE ,Laser);
764             put_int(Zbox);
765             *canbuf++=IS2;
766             *canbuf++=0;
767             fputs(cbuf,Laser);
768
769             fputs(SOLID_LINE,Laser);
770             fputs(FILL_STYLE,Laser);
771             fputs(THIN_LINE ,Laser);
772            plot_matrix();
773            fputs("\f",Laser);
774             fputs(SOFT_RESET,Laser);
775            close(Laser);
776            Laser=0;
777     }
778     draw_can(cmd,x,y) /* Execute move/draw on Cannon Laser */
779     char cmd;
780     int x, y;
781     {
782             x=(x<<1)+x;
783             y=-(y<<1)-y;
```

```
784             if (cmd==M)
785             {
786                     c_x0=x; c_y0=y;
787             }
788             else if (cmd==MR)
789             {
790                     c_x0+=x; c_y0+=y;
791             }
792             else if (cmd==D)
793             {
794                     canbuf=cbuf;
795                     fputs(POLY_LINE ,Laser);
796                     put_int(c_x0);
797                     put_int(c_y0);
798                     put_int(x-c_x0);
799                     put_int(y-c_y0);
800                     *canbuf++=IS2;
801                     *canbuf++=0;
802                     fputs(cbuf,Laser);
803                     c_x0=x; c_y0=y;
804             }
805             else if (cmd==DR)
806             {
807                     canbuf=cbuf;
808                     fputs(POLY_LINE ,Laser);
809                     put_int(c_x0);
810                     put_int(c_y0);
811                     put_int(x);
812                     put_int(y);
813                     *canbuf++=IS2;
814                     *canbuf++=0;
815                     fputs(cbuf,Laser);
816                     c_x0+=x; c_y0+=y;
817             }
818             else if (cmd==RR)
819             {
820                     canbuf=cbuf;
821                     fputs(POLYGON   ,Laser);
822                     put_int(c_x0);
823                     put_int(c_y0);
824                     put_int(x);
825                     put_int(0);
826                     put_int(0);
827                     put_int(y);
828                     put_int(-x);
829                     put_int(0);
830                     put_int(0);
831                     put_int(-y);
832                     *canbuf++=IS2;
833                     *canbuf++=0;
834                     fputs(cbuf,Laser);
835                     c_x0+=x; c_y0+=y;
836             }
837             else if (cmd==CI || cmd==PT)
838             {
839                     canbuf=cbuf;
```

```
840                     fputs(POLY_MARK ,Laser);
841                     put_int(c_x0);
842                     put_int(c_y0);
843                     *canbuf++=IS2;
844                     *canbuf++=0;
845                     fputs(cbuf,Laser);
846             }
847     }
848     color_can(color)     /* Translate color to marker size */
849     {
850             if (color<8)
851                     return;
852             canbuf=cbuf;
853             fputs(MARK_SIZE ,Laser);
854             put_int(color>>2);
855             *canbuf++=IS2;
856             *canbuf=0;
857             fputs(cbuf,Laser);
858     }
859
860     put_int(x)  /* Put int x into string at *canbuf in ISO format */
861     int x;
862     {
863             char msk, big=0;
864             msk=x>=0?0x30:0x20;
865             x=abs(x);
866             if (x>1023)
867             {
868                     *canbuf++=0x40|(x>>10);
869                     x&=1023;
870                     big=1;
871             }
872             if (x>15 || big)
873             {
874                     *canbuf++=0x40|(x>>4);
875                     x&=15;
876             }
877             *canbuf++=msk|x;
878     }
879
880     text_can(str)   /* Print string at x0,y0 */
881     {
882             canbuf= cbuf;
883             fputs(TO_TEXT   ,Laser);
884             put_int(c_x0);
885             put_int(c_y0);
886             *canbuf++=IS2;
887             *canbuf++=0;
888             fputs(cbuf,Laser);
889             fputs(str,Laser);
890
891             fputs(ENTER_VECT,Laser);
892
893     }
894
895     indx(i,list,n)  /* Index j of i<=list[j] */
```

```
896     int i, list[], n;
897     {
898             int j;
899             for (j=0; j<n && i>list[j]; j++)
900                     ;
901             return(j);
902     }
903     min(x,y)
904     int x, y;
905     {
906             return(x<y?x:y);
907     }
908     max(x,y)
909     int x, y;
910     {
911             return(x>y?x:y);
912     }
913
914
```

Blank Spacer Page

```
1   /*
2       program in11 - converts 1170 files to at real files.
3   */
4
5   #define MXBUF  20000    /* Data buffer size in integer*2 words */
6   #define BLK      150
7   #define MCOL     300    /* Max. no. data quantities (cols) */
8   #define L_LABL    50    /* Max. length of axis label */
9
10  int infi, outfi;
11  char *defi="\\1170dat\\evmast4.flm   \0 ";
12  char *defo="out11.plt                \0 ";
13
14  int Data[MXBUF];
15  char tbuf[BLK], obuf[BLK], *ib=tbuf, *ob=obuf;
16
17  long Ldata;  /* Data location on file */
18  long Lbinl;  /* Bin label location on file */
19
20  int Nevs,Bincr,No_rec,Reclen_int,Nrec_pblk,Brec_no,Ints_perblk ;
21  int Outflag,*Out_point,Out_event;
22  int Idtype=1;
23
24  char *cptr;
25  int *iptr;
26  float *fptr;
27  double *dptr;
28  long *lptr;
29
30  main(argc,argv)
31  int argc;
32  char *argv[];
33  {
34          int i, j,  ax_type, nevents, int_incr;
35          unsigned num,nitems,lrecl, nitems2;
36          float axr_lim[MCOL*2];
37          char ax_lab[MCOL][L_LABL];
38
39          int nbin,lcount,irecs,lrecl,iblk,ibyr;
40          double tstart,tend,avet, time;
41
42          long offset,itstart,itend,iavet;
43          long off_1970(), to_1970();
44
45          int fread(),fwrite();
46          float minr(),maxr();
47          long fseek(),lseek(),rewind();
48          int *get_data();
49
50          if(--argc==1)
51                  cptr=argv[1];
52          else
53                  cptr=defi;
54
55          printf("in file = %10.28s \n",cptr);
56
```

```
57              if (!(infi=fopen(cptr,"r")))
58              {
59                      puts("Cannot open input file ");
60                      puts(cptr);
61                      exit(1);
62              }
63              cptr=defo;
64              if ((outfi=creat(cptr))<0)
65              {
66                      puts("Cannot open output file ");
67                      puts(cptr);
68                      exit(1);
69              }
70
71              Ldata=0;
72              nevents=0;
73
74              fread(Data,sizeof(*Data),66,infi);
75
76              printf ("qname= %10.28s \n",cptr);
77
78              iptr=&Data[14];
79              nbin=iptr[0];
80              lcount=iptr[1];
81              irecs=iptr[2];
82              lrecl=iptr[3];
83              iblk=iptr[4];
84              ibyr=iptr[5];
85              nevents=lcount-nbin-2;
86
87              printf ("nbin,lc,ir,lr,ib,iy= %d %d %d %d %d %d \n",
88                nbin,lcount,irecs,lrecl,iblk,ibyr);
89
90              dptr=&Data[20];
91
92              convr8_11_at(3,&Data[20]);
93              dptr=&Data[20];
94              tstart=dptr[0];
95              tend=dptr[1];
96              avet=dptr[2];
97
98              printf ("ts,te,av= %e %e %e \n", tstart,tend,avet);
99
100             if(avet<0.)
101                     avet=-avet;
102             else
103                     ibyr=1977;
104
105             offset=off_1970(ibyr);
106             itstart=to_1970(tstart,offset);
107             itend=to_1970(tend,offset);
108             iavet=avet;
109
110             printf ("off,its,ite,ia= %12ld %12ld %12ld %12ld \n",
111               offset,itstart,itend,iavet);
112
```

```
113             printf ("run date = %12.12s \n",&Data[32]);
114             printf ("run time = %12.12s \n",&Data[38]);
115             printf ("tape type label= %8.8s \n",&Data[44]);
116     /*
117             nevents=20;
118     */
119             data_rewind(0L,lcount+1,lrecl/2);
120             cptr=0;
121
122             cptr=get_data(cptr);
123             cptr=get_data(cptr);
124             fprintf(outfi,"%8d%8dReal    %10ld\n", nbin,nevents,Ldata);
125
126             cptr=get_data(cptr);
127             fprintf(outfi,"%50.50s\n",cptr);
128
129             ax_type=0;
130
131             sprintf (&ax_lab[0][0],"Start/End Time of Event (seconds of 1970)\0");
132             axr_lim[0]=tstart;
133             axr_lim[1]=tend;
134             fprintf(outfi,"%-8d%-8d%-8.1e%-8.1e%-50.50s\n",
135               0,ax_type,axr_lim[0],axr_lim[1],&ax_lab[0][0]);
136
137             Lbinl=fseek(outfi,0L,2);
138             for(i=1;i<nbin+1;i++)
139             {
140                     cptr=get_data(cptr);
141                     strncpy(&ax_lab[i][0],&cptr[5],50);
142                     axr_lim[i*2]=1.e20;
143                     axr_lim[i*2+1]=-1.e20;
144
145                     fprintf(outfi,"%-8d%-8d%-8.1e%-8.1e%-50.50s\n",
146                       0,ax_type,axr_lim[i*2],axr_lim[i*2+1],&ax_lab[i][0]);
147             }
148
149             Ldata=fseek(outfi,0L,2);
150             set_outflag(cptr,1);
151             fptr=cptr;
152             nitems=(nbin+1)*2;
153             for(i=0;i<nevents;i++)
154             {
155                     fptr=get_data(fptr);
156                     dptr=fptr;
157                     lptr=fptr;
158
159                     convr8_11_at (1,dptr);
160                     lptr[0]=to_1970 (dptr[0],offset);
161                     lptr[1]=lptr[0]+iavet;
162                     printf ("lptr= %12ld %12ld \n",lptr[0],lptr[1]);
163
164                     convr4_11_at (nitems-2,&fptr[2]);
165                     for(j=2;j<nitems;j+=2)
166                     {
167                             if(fptr[j+1]>=0)
168                             {
```

```
169                                     axr_lim[j]=minr(axr_lim[j],fptr[j]);
170                                     axr_lim[j+1]=maxr(axr_lim[j+1],fptr[j]);
171                             }
172                     }
173
174             }
175
176             set_outflag(fptr,0);
177             lseek(outfi,Lbin1,0);
178             for(i=1;i<nbin+1;i++)
179             {
180                     if(axr_lim[i*2] == axr_lim[i*2+1])
181                             axr_lim[i*2+1]=axr_lim[i*2]+1.;
182                     if(axr_lim[i*2] == 1.e20)
183                     {
184                             axr_lim[i*2]=-1.;
185                             axr_lim[i*2+1]=1.;
186                     }
187                     if(axr_lim[i*2]>0)
188                             axr_lim[i*2]=axr_lim[i*2]*0.9;
189                     else
190                             axr_lim[i*2]=axr_lim[i*2]*1.1;
191                     if(axr_lim[i*2+1]<0)
192                             axr_lim[i*2+1]=axr_lim[i*2+1]*0.9;
193                     else
194                             axr_lim[i*2+1]=axr_lim[i*2+1]*1.1;
195
196                     fprintf(outfi,"%-8d%-8d%-8.1e%-8.1e%50.50s\n",
197                       0,ax_type,axr_lim[i*2],axr_lim[i*2+1],&ax_lab[i][0]);
198             }
199
200             lseek(outfi,0L,0);
201             fprintf(outfi,"%8d%8dReal    %10ld\n", nbin+1,nevents,Ldata);
202             close(outfi);
203     }
204
205     min(x,y)
206     int x,y;
207     {
208             return(x<y?x:y);
209     }
210     max(x,y)
211     int x,y;
212     {
213             return(x>y?x:y);
214     }
215     float minr(x,y)
216     float x,y;
217     {
218             return(x<y?x:y);
219     }
220     float maxr(x,y)
221     float x,y;
222     {
223             return(x>y?x:y);
224     }
```

```
225
226     next_line()     /* Get next line, expand tabs */
227     {
228             char buf[BLK], *s=buf, *d=tbuf;
229             int good;
230             while (good=fgets(buf,BLK,infi))
231             {
232                     if (*s=='\n' || *s=='#')
233                             continue;
234                     _setmem(d,BLK,' ');
235                     while (*s && *s!='\n')
236                     {
237                             if (*s=='\t')
238                             {
239                                     d=tbuf+(d-tbuf+8&0x0ff8);
240                                     s++;
241                             }
242                             else
243                                     *d++=*s++;
244                     }
245                     d[BLK-1]='\0';
246                     break;
247             }
248             return good;
249     }
250     getmins(sptr)     /* Return minutes from UT at *sptr */
251     char *sptr;       /*   Return -1 if blank            */
252     {
253             int hr, min;
254             char buf[3];
255
256             hr=0; min=-1;
257             buf[2]=0;
258             strncpy(buf,sptr,2);
259             sscanf(buf,"%2d",&hr);
260             strncpy(buf,sptr+2,2);
261             sscanf(buf,"%2d",&min);
262             return 60*hr+min;
263     }
264
265     convr4_11_at (n,iptr)
266     int n;
267     unsigned *iptr;
268     {
269             int i;
270             unsigned t;
271
272             for(i=0;i<n*2;i+=2)
273             {
274                     if(iptr[i]>0x0100)
275                     {
276                             t=iptr[i];
277                             iptr[i]=iptr[i+1];
278                             iptr[i+1]=t-0x0100;
279                     }
280                     else
```

```
281                     {
282                             iptr[i]=0;
283                             iptr[i+1]=0;
284                     }
285             }
286     }
287
288     convr8_11_at (n,iptr)
289     int n;
290     unsigned *iptr;
291     {
292             int i,j;
293             unsigned t;
294
295             for(i=0;i<n*4;i+=4)
296             {
297                     if(iptr[i]>0x0100)
298                     {
299                             t=iptr[i];
300                             iptr[i]=iptr[i+3];
301                             iptr[i+3]=t-0x0100;
302                             t=iptr[i+1];
303                             iptr[i+1]=iptr[i+2];
304                             iptr[i+2]=t;
305
306                             iptr[i]=iptr[i+1]<<13 | iptr[i]>>3;
307                             iptr[i+1]=iptr[i+2]<<13 | iptr[i+1]>>3;
308                             iptr[i+2]=iptr[i+3]<<13 | iptr[i+2]>>3;
309                             iptr[i+3]=((iptr[i+3]&0x3FFF)>>3) |
310                               (iptr[i+3]&0xC000);
311
312                     }
313                     else
314                     {
315                             iptr[i]=0;
316                             iptr[i+1]=0;
317                             iptr[i+2]=0;
318                             iptr[i+3]=0;
319                     }
320             }
321     }
322
323     int *get_data(phloc)
324     int *phloc;
325     {
326             int *read_data();
327             if(Brec_no<0)
328             {
329                     Bincr=0;
330                     phloc=read_data();
331             }
332             else
333             {
334                     Bincr+=Reclen_int;
335                     phloc+=Reclen_int;
336             }
```

```
337
338             if(Bincr>=Ints_perblk)
339             {
340                     phloc=read_data();
341                     Bincr=0;
342             }
343
344             if(Outflag==1)
345             {
346                     Outflag=2;
347                     Out_point=phloc;
348                     Out_event=Bincr;
349             }
350
351             return(phloc);
352     }
353
354     int *read_data()              /*read next data block, no selection-search*/
355     {
356             int n,i, col[MCOL], err,outs;
357             int *phloc;
358             char tbuf[BLK];
359
360             printf("enter redd: Outf,Outp,Outev,phloc,Bincr= %d %d %d %d %d \n",
361               Outflag,Out_point,Out_event,phloc,Bincr);
362
363             if(Outflag==2)
364                     if((outs=Bincr-Out_event)>0)
365                             fwrite(Out_point,sizeof(*Out_point),
366                               outs,outfi);
367             phloc=Data;
368             if(Brec_no<0)
369             {
370                     Nevs=min(Nrec_pblk,No_rec);
371                     Brec_no=0;
372                     Out_point=Data;
373                     Out_event=0;
374             }
375             else
376             {
377                     Nevs=min(Nrec_pblk,No_rec-(Brec_no+Nrec_pblk));
378                     Brec_no=Brec_no+Nevs;
379                     Out_point=Data;
380                     Out_event=0;
381             }
382
383             err=fread(phloc,sizeof(*phloc),Nevs*Reclen_int,infi);
384             if(err<=0)
385                     printf("read error: err,Phloc,Nevs= %d %d %d \n",
386                       err,phloc,Nevs);
387
388             return(phloc);
389
390     }
391
392     data_rewind(lposi,n1,n2)
```

```
393     int n1,n2;
394     long lposi;
395     {
396             int err;
397             err=fseek(infi,lposi,0);
398             if(err< 0)
399                     printf("fseek error: Ldata= %d \n",Ldata);
400             Brec_no=-1;
401
402             No_rec=n1;
403             Reclen_int=n2;
404             Nrec_pblk=MXBUF/Reclen_int;
405             Ints_perblk=Nrec_pblk*Reclen_int;
406
407             Outflag=0;
408             Out_point=Data;
409             Out_event=0;
410     }
411
412     set_outflag(iptr,mflag)
413     int *iptr,mflag;
414     {
415             int outs;
416
417             outs=0;
418             if((Outflag==2) && (mflag==0))
419             {
420                     outs=Bincr-Out_event+Reclen_int;
421                     fwrite(Out_point,sizeof(*Out_point),outs,outfi);
422             }
423
424             Outflag=mflag;
425             Out_point=Data;
426             Out_event=0;
427
428             printf("exit oflg: Of,mf,Op,Oe,ip,Bi,outs= %d %d %d %d %d %d %d \n",
429               Outflag,mflag,Out_point,Out_event,iptr,Bincr,outs);
430
431     }
432
433     long off_1970 (ibyr)
434     int ibyr;
435     {
436             long offset;
437             int ndays;
438
439             ndays=( (ibyr-1969)/4 + (ibyr-1970)*365 -1);
440             offset=(long)ndays *86400;
441             return (offset);
442     }
443
444     long to_1970 (time,offset)
445     double time;
446     long offset;
447     {
448             long itime;
```

```
449             itime=(long)time + offset;
450             return(itime);
451     }
452
```

```
sptr        250   251#  258   260
sscanf      259   261
strncpy     141   258   260
t           270   276   278   293   299   301   302   304
tbuf         15#   15#  228   239   239   358
tend         40    95    98   107   133
time         40   444   445#  449
to_1970      43   106   107   160   444#
tstart       40    94    98   106   132
x           205   206#  208   208   210   211#  213   213   215   216#  218   218   220
            221#  223   223
y           205   206#  208   208   210   211#  213   213   215   216#  218   218   220
            221#  223   223
```

Blank Spacer Page

Cross - Graphic Cross-plot Program

for the: IBM Professional Graphics Adapt.
and the: HP Laserjet Plus printer.

Key definitions as of Apr 20, 1986 beta test.

- D. Reames

Graphic Mode:

F1	Select Menu mode
F3/F4	Expand/Contract z (color) axis
F5/F6	Expand/Contract y axis
F7/F8	Expand/Contract x axis
Escape	Replot data
Enter	" "
^C	Exit the program
^P	Add plot to printer file (plot.las)
^S	Shift current cursor coord to origin
^X	Toggle cursor - Box/Tracking cross/Off
^Z	Toggle z(color)-axis display
Arrow keys	Step cursor left, right, up, down
^ Right arrow	Move right 1/2 page (+x)
^ Left arrow	Move left 1/2 page (-x)
Page up	Move up 1/2 page (+y)
Page down	Move down 1/2 page (-y)
^ Page up	Step z (color) scale up
^ Page down	Step z (color) scale down
Home	Move curs. to origin (lower left) on screen
End	Move curs. to upper right corner
^Home	Move plot to origin (lower left) of plot space
^End	Move plot to upper right corner of plot space

(Left-over features, not fully implemented)

ASCII char	Type char on screen
^R	Rubber-band-line toggle
^W	Re-initialize windows

Menu mode:

F1	Select graphics mode
Escape	" " "
up/down arrows	Move cursor one row
Enter	Select or change current row (see below)

Axis select:

Escape	Return to menu mode -no changes
F1	" " "
up/down arrows	Move cursor one row
Enter	Select current row for menu
^C	Exit the program

Limit select:

Escape	Return to menu mode -no changes
left/rt arrows	Move cursor one column
0-9,-,blank	Type the char.
Enter	Select current values for menu
^C	Exit the program

cntl c   cntl z
alt z   color bar

1068Hpd0b12d@0Bs0s0B8Usp10h12v0s0Bs0s0B
k12H   Cross - Graphic Cross-plot P
rogram

for the: IBM Professional Graphics Adapt.
and the: HP Laserjet Plus printer.

Key definitions as of Apr 20, 1986 beta test.
- D. Reames

Graphic Mode:

F1	Select Menu mode
F3/F4	Expand/Contract z (color) axis
F5/F6	Expand/Contract y axis
F7/F8	Expand/Contract x axis
Escape	Replot data
Enter	" "
^C	Exit the program
^P	Add plot to printer file (plot.las)
^S	Shift current cursor coord to origin
^X	Toggle cursor - Box/Tracking cross/Off
^Z	Toggle z(color)-axis display
Arrow keys	Step cursor left, right, up, down
^ Right arrow	Move right 1/2 page (+x)
^ Left arrow	Move left 1/2 page (-x)
Page up	Move up 1/2 page (+y)
Page down	Move down 1/2 page (-y)
^ Page up	Step z (color) scale up
^ Page down	Step z (color) scale down
Home	Move curs. to origin (lower left) on screen
End	Move curs. to upper right corner
^Home	Move plot to origin (lower left) of plot space
^End	Move plot to upper right corner of plot space

(Left-over features, not fully implemented)

ASCII char	Type char on screen
^R	Rubber-band-line toggle
^W	Re-initialize windows

Menu mode:

F1	Select graphics mode
Escape	" " "
up/down arrows	Move cursor one row
Enter	Select or change current row (see below)

Axis select:

Escape	Return to menu mode -no changes
F1	" " "
up/down arrows	Move cursor one row
Enter	Select current row for menu
^C	Exit the program

Limit select:

ESC	Return to menu mode -no changes
left/rt arrows	Move cursor one column
0-9,-,blank	Type the char.
Enter	Select current values for menu
^C	Exit the program

116C
lo8osP&OBBBG$dB@BH12vOsOB18C

```
/*
    phatree.c

    Subroutine library to store, count
    & recall incoming pulse heights
    using self-referential tree structure

*/

#include <stdio.h>
#define MAXWORD 20

struct tnode
{
    int ph[2];      /* Pulse heights    */
    int count;
    struct tnode *left; /* Left child       */
    struct tnode *right;    /* Right child      */
};

struct tnode *root, *tree();
int nodes=0, phas=0, maxn=1;

extern int ishist;

clear_tree()         /* Initialize tree */
{
    root=NULL;
    nodes=0;
    phas=0;
    maxn=1;
    freeall(2048);
}
add_ph(new)     /* File new pulse height */
int *new;
{
    phas++;
    root=tree(root,new);
}

struct tnode *tree(p,new) /* Install w in tree or count it */
struct tnode *p;
int *new;
{
    char *malloc();
    int cond;

    if (p==NULL)
    {                       /*  Make new node */
        p=malloc(sizeof(struct tnode));
        p->ph[0]=new[0];
        p->ph[1]=new[1];
        p->count=1;
        p->left=p->right=NULL;
        nodes++;
    }
    else if ((cond=ph_cmp(new,p->ph))==0) /* Count old PH    */
    {
        p->count++;
        maxn=max(maxn,p->count);
```

```
61          }
62          else if (cond < 0)            /* Smaller, to left */
63              p->left=tree(p->left,new);
64          else                          /* Larger, to right */
65              p->right=tree(p->right,new);
66          return(p);
67      }
68
69      print_ph()      /* Print all */
70      {
71          treeprint(root);
72      }
73
74      treeprint(p)        /* Print the tree */
75      struct tnode *p;
76      {
77          if (p != NULL)
78          {
79              treeprint(p->left);
80              if (ishist==0)
81                  point(p->ph[0],p->ph[1],lg_int(p->count,2),1,1);
82              else if (ishist==1)
83                  point(bin_to_dis(p->ph[0],0),
84                      lg_int(p->count,1),255,1,0);
85              else
86                  point(bin_to_dis(p->ph[0],0),
87                      bin_to_dis(p->ph[1],1),
88                      lg_int(p->count,2),1,1);
89              treeprint(p->right);
90          }
91      }
92
93
94      ph_cmp(new,old)      /* Compare PH's */
95      int *new, *old;
96      {
97          int dif;
98          dif=new[1]-old[1];
99          if (dif)
100             return(dif);
101         else
102             return(new[0]-old[0]);
103     }
```

```
/*
        WINDO - Graphic Window Controller for PGC

    Windo provides interactive keyboard control of the display
characteristics of multiple windows into graphic plot space.  It
allows selection of window size and location and permits pan and
zoom of graphic data into the window. Simple editing of the plot
area such as typed labels and added and deleted lines are also
supported but the main plotting facilities are elsewhere.

                    D. Reames

*/

#define XMAX     640        /* Plot width in pixels     */
#define YMAX     480        /* Plot height in pixels    */
#define ZMAX      32        /* Max zoom         */

#define NWIND      5        /* Number of windows        */
#include "windo.h"

#define M   0x10    /* Move */

struct window
    w0= { 256,210,128, 256,210,148, 8,8,4, 0,0,0,
          0,0,0, 2048,2048,100, 1,1,-6, 1,1,1, 0,0,0, 0, 0 },
    w1= { 256,210,128, 256,210,148, 8,8,4, 0,0,0,
          0,0,0, 2048,2048,100, 4,4,-6, 1,1,1, 0,0,0, 0, 0 },
    w2= { 256,210,128, 256,210,148, 8,8,4, 0,0,0,
          -10,-10,-10, 2048,2048,100, 1,1,-6, 1,1,1, 0,0,0, 0, 0 },
    w3= { 256,210,128, 256,210,148, 8,8,4, -100,-100,0,
          -100,-100,0,    3500,3500,1000, 8,8,0, 1,1,1, 1,1,0, 0, 0 }
,
    w4= { 256,210,128, 256,210,148, 8,8,4, -100,-100,-100,
          -100,-100,-100, 3500,3500,3500, 4,4,6, 1,1,1, 1,1,1, 0, 0 }
;
struct window ww[NWIND], *w=&ww[1];
int entry=1;
extern int Ishist;

int lf1=0x01eb, lf0=0x00eb;  /* LF cmd: 1=xor, 0=plot color */

#include "keydef.h"

int k=1, displ=2, widp, htp, yw;
int e_rt, e_left, e_top, e_bot;
int rubr=0, xr, yr;     /* Rubber-band line */
int ksave=1;
int sav_po[3], Dif_d[3];
int curs_col;
winit()         /* Initialize windows   */
{
    _move(sizeof(w0),&w0,&ww[0]);  /* init. windows */
    _move(sizeof(w1),&w0,&ww[1]);
    _move(sizeof(w2),&w0,&ww[2]);
    _move(sizeof(w3),&w0,&ww[3]);
    _move(sizeof(w4),&w0,&ww[4]);
    rubr=0;
    w_read();
/*
```

```
59          w_save();
60      */
61          k=1;
62          w=&ww[k];
63      }
64      w_read()    /* Read in window struct */
65      {
66          char buf[80];
67          int *p, siz, i, j, n;
68          int wfi;
69          siz=sizeof(w0)>>1;
70          if ((wfi=fopen("windo.def","r")) == 0)
71          {
72              puts("Cannot open windo.def");
73              return 0;
74          }
75
76          while (1)
77          {
78              if (fscanf(wfi,"Window %d:\n",&i)<0)
79                  break;
80              printf("Window %d:\r",i);
81              p=&ww[i];
82              for(j=0; j<siz; )
83              {
84                  if (!fgets(buf,80,wfi))
85                      break;
86                  n=sscanf(buf,"%8d%9d%9d",p,p+1,p+2);
87                  p+=n; j+=n;
88              }
89
90          }
91          close(wfi);
92          return 1;
93      }
94
95      w_save()    /* Save window structure */
96      {
97          int *p, siz, i, j;
98          int wfi;
99          siz=sizeof(w0)>>1;
100         if ((wfi=fopen("windo.out","w")) == 0)
101         {
102             puts("Cannot open windo.out");
103             return 0;
104         }
105
106         for  (i=0; i<NWIND; i++)
107         {
108             p=&ww[i];
109             fprintf(wfi,"Window %d:\n",i);
110             for(j=0; j<siz; j++)
111                 fprintf(wfi,"%8d%c%",*p++,j%3==2?'\n':' ');
112             fputs("\n",wfi);
113         }
114         close(wfi);
115         return 1;
116     }
117
118     show_curs(kk, pk) /* Return coords of cursor kk */
```

```
119     int kk, *pk;
120     {
121         *pk=ww[kk].cp;
122     }
123     set_curs(kk,xk,yk) /* Set cursor kk to xk,yk   */
124     int kk, xk, yk;
125     {
126         ww[kk].cp[0]=xk;
127         ww[kk].cp[1]=yk;
128     }
129     set_wind(kk)    /* Select window (from 0) */
130     {
131         ksave=kk;
132         k=kk;
133         w=&ww[k];
134     }
135
136     windo()
137     {
138         int c, offs, bw, bh, xd, yd, xh;
139         send_asc("CA DI 0 CX ");
140         set_edge();
141         w=&ww[k];
142         sav_po[0]=w->po[0];
143         sav_po[1]=w->po[1];
144         sav_po[2]=w->po[2];
145         if (entry)
146         {
147             displ=max(plot_stat(k),2);
148             entry=0;
149         }
150         else
151             displ=2;
152
153         while ( (c=key())!=0x1b && c!=13 ) /* Return on Esc & CR */
154         {
155             if (c<0 || c>0x0184)
156                 c=-1;
157             else if (c>0x0171)  /*  ^ numpad, alt 1...= */
158                 numpad(c);
159             else if (c>0x0153)  /* sh F, ^ F, or alt F */
160                 fn_key(c);
161             else if (c>0x0146)  /* numpad */
162                 numpad(c);
163             else if (c>0x013a)  /* F */
164                 fn_key(c);
165             else if (c>0x007f)  /* high ASCII, alt a... */
166                 c=alt_key(c);
167             else if (c>0x001f)  /* ASCII chars */
168             {
169                 plot_chr(c);
170             }
171             else if (c>=8 && c<=13)
172             {
173                 ;
174             }
175             else            /* ASCII control */
176                 control(c);
177             if (c==3)       /* ^C - Exit    */
178             {
```

```
179                 send_asc("CA DI 1 ");
180                 exit(1);
181             }
182             if (displ>0)
183             {
184                 set_edge();
185                 if (displ==2 && k>0)
186                 {
187                     Dif_d[0]=to_dis(sav_po[0],0);
188                     Dif_d[1]=to_dis(sav_po[1],1);
189                     Dif_d[2]=to_dis(sav_po[2],2);
190                     plot_matrix();
191                 }
192                 else if (displ>2 && k>0)
193                 {
194                     send_asc("CA F 0 CX ");
195                     clear_tree();
196                     scale();
197                     if (Ishist)
198                         box(0,0,bin_to_dis(1,0),
199                         Ishist==2?bin_to_dis(1,1):210);
200                 }
201                 cursor();
202                 displ=0;
203             }
204         }
205         send_asc("CA DI 1 ");
206         return(c);
207     }
208     cursor()
209     {
210         int x0, y0, x1, y1;
211
212         x0=max(w->cp[0]-4,0);   x1=min(w->cp[0]+4,e_rt);
213         y0=max(w->cp[1]-4,0);   y1=min(w->cp[1]+4,e_top);
214         send(&lf1,2);
215         if (w->ct==2)
216         {
217             line(0,w->cp[1],e_rt,w->cp[1]);
218             line(w->cp[0],e_top,w->cp[0],0);
219         }
220         else if (w->ct==1)
221         {
222             box(x0,y0,x1,y1);
223             xpoint(w->cp[0],w->cp[1]);
224         }
225         else if (w->ct==3)
226         {
227             box(x0,y0,x1+150,y1+10);
228         }
229         if (rubr)
230             line(xr,yr,w->cp[0],w->cp[1]);
231         send(&lf0,2);
232     }
233     set_edge()  /* Set display edges */
234     {
235     /*  e_left=0;               window edge- left pix*/
236         e_rt=w->dc[0]<<1;        /*      right   */
237         e_top=w->dc[1]<<1;       /*      top */
238     /*  e_bot=0;                    bottom  */
```

```
239     }
240
241     box(x0,y0,x1,y1)
242     int x0, y0, x1, y1;
243     {
244         line(x0,y0,x0,y1);
245         line(x1,y0,x1,y1);
246         line(x0,y0,x1,y0);
247         line(x0,y1,x1,y1);
248     }
249     xpoint(x0,y0)
250     int x0, y0;
251     {
252         line(x0,y0,x0,y0);
253     }
254
255     numpad(c)   /* process keys on the numeric pad */
256     int c;
257     {
258         int zsx, zsy, ik;
259
260         zsx=w->ss[0];
261         zsy=w->ss[1];
262         cursor();
263         displ=1;
264         if (c==K_DN)            /* Step down */
265             step_dn(zsy);
266         else if (c==K_UP)       /* Step up */
267             step_up(zsy);
268         else if (c==K_RT)       /* Step right */
269             step_rt(zsx);
270         else if (c==K_LEFT)     /* Step left */
271             step_lft(zsx);
272         else if (c==C_PGDN)     /* Step z (color) down  */
273         {
274             w->po[2]=max(w->p1[2],
275                 to_plt(-max(w->ss[2],2-w->zf[2]),2));
276             displ=2;
277         }
278         else if (c==C_PGUP)     /* Step z (color) up  */
279         {
280             w->po[2]=min(w->p2[2],
281                 to_plt(max(w->ss[2],2-w->zf[2]),2));
282             displ=2;
283         }
284         else if (c==K_PGDN)     /* Page down (1/2 displ) */
285         {
286             w->po[1]=max(w->p1[1],to_plt(-w->dc[1],1));
287             displ=2;
288         }
289         else if (c==K_PGUP)     /* Page up */
290         {
291             w->po[1]=min(w->p2[1],to_plt(w->dc[1],1));
292             displ=2;
293         }
294         else if (c==C_RT)       /* Page right */
295         {
296             w->po[0]=min(w->p2[0],to_plt(w->dc[0],0));
297             displ=2;
298         }
```

```
299         else if (c==C_LEFT)     /* Page left */
300         {
301             w->po[0]=max(w->p1[0],to_plt(-w->dc[0],0));
302             displ=2;
303         }
304         else if (c==K_HOME) /* Lower left corner of window */
305         {
306             w->cp[0]=0; w->cp[1]=0;
307             displ=1;
308         }
309         else if (c==K_END)  /* Upper rt corner of window */
310         {
311             w->cp[0]=e_rt; w->cp[1]=e_top;
312             displ=1;
313         }
314         else if (c==C_HOME) /* Lower left corner of plot */
315         {
316             w->po[0]=w->p1[0];
317             w->po[1]=w->p1[1];
318             w->cp[0]=0;
319             w->cp[1]=0;
320             displ=2;
321         }
322     #if 0
323     /* Disabled:
324         else if (c==C_END)  / Lower left corner of plot /
325         {
326             w->cp[0]=0;         w->x=widp;
327             w->cp[1]=YMAX;      w->y=YMAX-htp+1;
328             displ=2;
329         }
330         else if (c==C_PGUP) / Upper right corner of plot /
331         {
332             w->cp[0]=XMAX;      w->x=XMAX-widp+1;
333             w->cp[1]=0;         w->y=htp;
334             displ=2;
335         }
336         else if (c==C_PGDN) / Lower right corner of plot /
337         {
338             w->cp[0]=XMAX;      w->x=XMAX-widp+1;
339             w->cp[1]=YMAX;      w->y=YMAX-htp+1;
340             displ=2;
341         }
342     */
343         else if (c>=A_1)    /* Select window by number  */
344         {
345             ik=c-A_1+1;
346             if (c==A_0)
347                 ik=NWIND;
348             if (ik<NWIND)
349             {
350                 k=ik; w=&ww[k];
351                 displ=3;
352                 rubr=0;
353             }
354         }
355     #endif
356     }
357
358     fn_key(c)       /* Process function keys */
```

```
359     int c;
360     {
361         int key, zstep, alt;
362         key=c-K_F1+1;
363         alt=c-A_F1+1;
364         if (key<11 || alt<11)
365         {
366             if (key==2) /* F2 - Update data plot    */
367             {
368     /*
369                 if (k==0)
370                     k=ksave;
371                 displ=2;
372                 w=ww[k];
373                 rubr=0;
374     */
375             }
376             else if (key==1)    /* F1 - window 0 select,  */
377             {           /*    -  status window    */
378                 if (k>0)
379                 {
380                     displ=plot_stat(ksave);
381                 }
382                 w=ww[k];
383                 rubr=0;
384             }
385             else if (key==3)    /* F3 - zoom color in   */
386             {
387                 zstep=(abs(w->zf[2])>>3)+1;
388                 w->zf[2]=max(w->zf[2]-zstep,-128);
389                 displ=3;
390             }
391             else if (key==4)    /* F4 - zoom color out  */
392             {
393                 zstep=(abs(w->zf[2])>>3)+1;
394                 w->zf[2]=min(w->zf[2]+zstep,128);
395                 displ=3;
396             }
397             else if (key==5)    /* F5 - zoom y in    */
398             {
399                 zstep=(abs(w->zf[1])>>3)+1;
400                 w->zf[1]=max(w->zf[1]-zstep,-128);
401                 displ=3;
402             }
403             else if (key==6)    /* F6 - zoom y out   */
404             {
405                 zstep=(abs(w->zf[1])>>3)+1;
406                 w->zf[1]=min(w->zf[1]+zstep,128);
407                 displ=3;
408             }
409             else if (key==7)    /* F7 - zoom x in   */
410             {
411                 zstep=(abs(w->zf[0])>>3)+1;
412                 w->zf[0]=max(w->zf[0]-zstep,-128);
413                 displ=3;
414             }
415             else if (key==8)    /* F8 - zoom x out  */
416             {
417                 zstep=(abs(w->zf[0])>>3)+1;
418                 w->zf[0]=min(w->zf[0]+zstep,128);
```

```
419                 displ=3;
420             }
421             else if (alt==3)    /* a_F3 - zoom color bin in   */
422             {
423                 zstep=(abs(w->zf[5])>>3)+1;
424                 w->zf[5]=max(w->zf[5]-zstep,1);
425                 displ=3;
426             }
427             else if (alt==4)    /* a_F4 - zoom color bin out  */
428             {
429                 zstep=(abs(w->zf[5])>>3)+1;
430                 w->zf[5]=min(w->zf[5]+zstep,128);
431                 displ=3;
432             }
433             else if (alt==5)    /* a_F5 - zoom y bin in    */
434             {
435                 zstep=(abs(w->zf[4])>>3)+1;
436                 w->zf[4]=max(w->zf[4]-zstep,1);
437                 displ=3;
438             }
439             else if (alt==6)    /* a_F6 - zoom y bin out   */
440             {
441                 zstep=(abs(w->zf[4])>>3)+1;
442                 w->zf[4]=min(w->zf[4]+zstep,128);
443                 displ=3;
444             }
445             else if (alt==7)    /* a_F7 - zoom x bin in   */
446             {
447                 zstep=(abs(w->zf[3])>>3)+1;
448                 w->zf[3]=max(w->zf[3]-zstep,1);
449                 displ=3;
450             }
451             else if (alt==8)    /* a_F8 - zoom x bin out  */
452             {
453                 zstep=(abs(w->zf[3])>>3)+1;
454                 w->zf[3]=min(w->zf[3]+zstep,128);
455                 displ=3;
456             }
457     #if 0
458             else if (key==5)    /* F5 - dec step, speed  */
459             {
460                 w->ss[1]=max(w->ss[1]>>1,1);
461             }
462             else if (key==6)    /* F6 - inc step, speed  */
463             {
464                 w->ss[1]=min(w->ss[1]<<1,32);
465             }
466         }
467         if (c==C_F10)       /* ^F10 - init & clear */
468         {
469     /*      init_card(1);     */
470             clear_all();
471             displ=2;
472     #endif
473         }
474     }
475     alt_key(c)      /* Alt. keys        */
476     int c;
477     {
478         if (c==A_C)     /* Alt-C - Color palette */
```

```
479         {
480             color_pg(7);
481             draw_pg(M,512,300);
482             text_pg("Set Color!");
483             curs_col=w->cp[2];
484             draw_pg(M,512,285);
485             color_pg(curs_col);
486             text_pg(" -<text>-");
487             c=palette('L');
488             w->cp[2]=curs_col;
489             color_pg(0);
490             draw_pg(M,512,300);
491             text_pg("Set Color!");
492             draw_pg(M,512,285);
493             text_pg(" -<text>-");
494             return(c);
495         }
496     }
497     control(c)      /* ASCII control chars  */
498     int c;
499     {
500         if (c==4)       /* ^D - Clear display mem   */
501             ;
502         else if (c==0x0010) /* ^P - Print plot to laserjet  */
503         {
504             plot_laser();
505         }
506         else if (c==0x0012) /* ^R - Rubber band line    */
507         {
508             if (rubr==0)
509             {
510                 rubr=1;
511                 xr=w->cp[0];
512                 yr=w->cp[1];
513             }
514             else
515             {
516                 send(&lf1,2);       /* Xor old line */
517                 line(xr,yr,w->cp[0],w->cp[1]);
518                 send(&lf0,2);
519                 color_pg(w->cp[2]);   /* Draw new line */
520                 line(xr,yr,w->cp[0],w->cp[1]);
521                 rubr=0;
522             }
523         }
524         else if (c==0x0013) /* ^S- Shift curs to origin po   */
525         {
526             w->po[0]=to_plt(w->cp[0],0);
527             w->po[1]=to_plt(w->cp[1],1);
528             w->cp[0]=0;
529             w->cp[1]=0;
530             displ=2;
531         }
532         else if (c==0x0017) /* ^W - Re-initialize windows   */
533         {
534             winit();
535             displ=3;
536         }
537         else if (c==0x0018 && k>0) /* ^X - Toggle tracking cross  */
538         {               /*          vs. cursor     */
```

```
539             cursor();
540             w->ct=++w->ct%3;
541             displ=1;
542         }
543         else if (c==0x1a)     /* ^Z - Z-axis, color scale toggle */
544         {
545             w->cse^=1;
546             displ=2;
547         }
548     }
549     plot_chr(c)     /* Plot char at current xc,yc   */
550     int c;
551     {
552         int x, y;
553         cursor();
554         x=max(0,w->cp[0]-3);
555         y=max(0,w->cp[1]-3);
556         draw_pg(M,x,y); /*  Move */
557         color_pg(w->cp[2]);
558         text_pg(&c);
559         step_rt(8);
560         displ=1;
561     }
562     step_rt(dx)
563     int dx;
564     {
565         int new;
566         new=w->cp[0]+dx;
567         if (new<e_rt)
568             w->cp[0]=new;
569         else
570         {
571             new=to_plt(to_dis(w->po[0],0)+dx,0);
572             w->po[0]=min(w->p2[0],new);
573             displ=2;
574         }
575     }
576     step_lft(dx)
577     int dx;
578     {
579         int new;
580         new=w->cp[0]-dx;
581         if (new>=0)
582             w->cp[0]=new;
583         else
584         {
585             new=to_plt(to_dis(w->po[0],0)-dx,0);
586             w->po[0]=max(w->p1[0],new);
587             displ=2;
588         }
589     }
590     step_up(dx)
591     int dx;
592     {
593         int new;
594         new=w->cp[1]+dx;
595         if (new<e_top)
596             w->cp[1]=new;
597         else
598         {
```

```
599             new=to_plt(to_dis(w->po[1],1)+dx,1);
600             w->po[1]=min(w->p2[1],new);
601             displ=2;
602         }
603     }
604     step_dn(dx)
605     int dx;
606     {
607         int new;
608         new=w->cp[1]-dx;
609         if (new>=0)
610             w->cp[1]=new;
611         else
612         {
613             new=to_plt(to_dis(w->po[1],1)-dx,1);
614             w->po[1]=max(w->p1[1],new);
615             displ=2;
616         }
617     }
618
619     key()   /* Read keyboard with immediate return.
620             - Remap keys preceeded by null byte.
621             - Return -1 if no key pressed.
622         */
623     {
624         int c;
625         if ((c=chi())==0)
626             c=0x100|chi();
627         return(c);
628     }
629     chi()   /* Call for keyboard char, return -1 if none */
630     {
631     #asm
632         mov     dl,0ffh
633         mov     ax,0600h
634         int     21h
635         mov     ah,0
636         jnz     exit
637         mov     ax,0ffffh
638     exit:
639         nop
640     #
641     }
642     update()
643     {
644     }
```

```
 1
 2      /*  palette.c - subs for palmain.c
 3       *
 4       *  List of routines:
 5       *      palette()
 6       *      setcolors()
 7       */
 8
 9      #include "keydef.h"
10
11      #define NVCOLORS   248
12      #define VCOLORS      8
13      #define M   0x10    /* Move */
14
15      extern int curs_col;
16
17      struct  {
18          float   bright;     /* Roll colors up/down  */
19          float   contrast;   /* Color gradient   */
20          char    scheme;
21          } palor = { 0.5, 0.5, 'r' };    /* Default parameters */
22
23      palette(cmd)
24      char cmd;       /* A valid cmd will be exeuted with immediate */
25                  /* return, cmd=='L' will loop on key input.   */
26      {
27          int a;
28          unsigned char b, c, d;
29          static unsigned x=255, y=382;
30          a=cmd;
31
32          do {
33
34              if (toupper(cmd)=='L')
35                  a=key();
36
37              switch( a ) {
38
39              case 'q':
40              case 'Q':
41              case 0x7f:
42              case 0x3:
43              case 0x1b:
44                  return a;
45
46
47              case 'g':       /* all color names here */
48              case 'G':
49              case 'a':
50              case 'A':
51              case 'i':
52              case 'I':
53              case 'r':
54              case 'R':
55                  palor.scheme = tolower(a);
56                  break;
57              case K_LEFT:
58                  y=(y+8)&511;
59                  break;
60              case K_RT:
```

```
61                  y=(y-8)&511;
62                  break;
63              case K_UP:
64                  x=(x-8)&511;
65                  break;
66              case K_DN:
67                  x=(x+8)&511;
68                  break;
69              case C_LEFT:
70                  y=(y+16)&511;
71                  break;
72              case C_RT:
73                  y=(y-16)&511;
74                  break;
75              case K_PGUP:
76                  x=(x-16)&511;
77                  break;
78              case K_PGDN:
79                  x=(x+16)&511;
80                  break;
81              case 'p':
82                  palor.scheme = ' ';             /* pass blank */
83                  break;
84              case '-':
85                  curs_col-=8;
86              case '+':
87                  curs_col+=4;
88                  curs_col&=255;
89                  draw_pg(M,512,285);
90                  color_pg(curs_col);
91                  text_pg(" -<text>-");
92                  continue;
93              default:
94                  continue;
95              }
96
97
98
99              palor.bright = x / 511.0;
100             palor.contrast = y / 511.0;
101
102             setcolors();
103
104         } while (toupper(cmd)=='L');
105
106         return a;
107
108     }
109     /*  setcolors.c - a sub for palette and other mains
110      *
111      */
112
113
114     unsigned char   tab[ 5 ];
115
116     short pal_aw, pal_rl, pal_rw;
117
118     setcolors()
119     {
120         short   aw, ac, al, ah;
```

```
121
122         /* aw is the width of the dynamic range */
123         aw = NVCOLORS * ( 1.0 - palor.contrast );
124
125         /* ac is the center of the dynamic range */
126         ac = NVCOLORS * ( 1.0 - palor.bright );
127
128         /* al is the lower limit of the dynamic range */
129         al = ac - aw / 2;
130
131         /* ah is the upper end of the dynamic range */
132         ah = al + aw;
133
134         switch( palor.scheme ) {
135
136         case 'g':       /* Simple grey scale (jpd 2/18/84) */
137         default:
138             {
139                 register short i;
140                 register unsigned char v;
141
142                 for( i = 0; i < NVCOLORS; i++ ) {
143                     if( i <= al ) v = 0;
144                     else if( i >= ah ) v = 255;
145                     else v = ( i - al )*255L  / aw;
146                     tab[ 0 ] = v;
147                     tab[ 1 ] = v;
148                     tab[ 2 ] = v;
149
150                     send_colr(i);
151                 }
152             }
153             break;
154
155         case 'a':       /* "apple" ( jpd 2/18/84 )*/
156             {
157                 register short i;
158                 short gl, bl, gw, bw;
159
160                 gw = aw / 2;
161                 gl = ah - gw;
162                 bw = gw / 2;
163                 bl = ah - bw;
164
165                 for( i = 0; i < NVCOLORS; i++ ) {
166
167                     if( i <= al ) {
168                         tab[ 0 ] = 0;
169                         tab[ 1 ] = 0;
170                         tab[ 2 ] = 0;
171                         send_colr(i);
172                         continue;
173                     }
174
175                     if( i >= ah ) {
176                         tab[ 0 ] = 255;
177                         tab[ 1 ] = 255;
178                         tab[ 2 ] = 255;
179                         send_colr(i);
180                         continue;
```

```
181                     }
182
183                     tab[ 0 ] = (( i - al ) * 255L ) / aw;
184
185                     if( i <= gl ) tab[ 1 ] = 0;
186                     else tab[ 1 ] = (( i - gl ) * 255L ) / gw;
187
188                     if( i <= bl ) tab[ 2 ] = 0;
189                     else tab[ 2 ] = (( i - bl ) * 255L ) / bw;
190
191                     send_colr(i);
192                 }
193             }
194             break;
195
196         case 'i':                   /* Inferno (2/18/84 jpd)*/
197             {
198                 register short i;
199                 register short v;
200                 short rl, rh, rw, gl, gh, gw, bl, bh, bw;
201
202                 rl = al;
203                 rw = aw / 3;
204                 rh = rl + rw;
205                 gl = rh;
206                 gw = rw;
207                 gh = gl + gw;
208                 bl = gh;
209                 bh = ah;
210                 bw = bh - bl;
211
212                 for( i = 0; i < NVCOLORS; i++ ) {
213
214                     if( i <= rl ) v = 0;
215                     else if( i >= rh ) v = 255;
216                     else v = (( i - rl ) * 255L ) / rw;
217                     tab[ 0 ] = v;
218
219                     if( i <= gl ) v = 0;
220                     else if( i >= gh ) v = 255;
221                     else v = (( i - gl ) * 255L ) / gw;
222                     tab[ 1 ] = v;
223
224                     if( i <= bl ) v = 0;
225                     else if( i >= bh ) v = 255;
226                     else v = (( i - bl ) * 255L ) / bw;
227                     tab[ 2 ] = v;
228                     send_colr(i);
229                 }
230             }
231             break;
232         case 'r':                   /* rainbow (6/20/86 dvr)*/
233             {
234                 register short i;
235
236                 aw=aw<12?12:aw;
237                 aw<<=1;
238                 ac=aw>NVCOLORS?aw*(1.0-palor.bright):ac;
239                 pal_rl = ac-aw;
240                 pal_rw = aw / 3;
```

```
241                 pal_aw=aw;
242
243                 for( i = 0; i < NVCOLORS; i++ ) {
244
245                     tab[ 0 ] = ramp(i,0);
246                     tab[ 1 ] = ramp(i,1);
247                     tab[ 2 ] = ramp(i,2);
248                     send_colr(i);
249                 }
250             }
251             break;
252
253         }
254     }
255     send_colr(i)
256     char i;
257     {
258         char buf[8];
259
260         buf[0]=0xee;
261         buf[1]=i+VCOLORS;
262         buf[2]=tab[0]>>4;
263         buf[3]=tab[1]>>4;
264         buf[4]=tab[2]>>4;
265         send( buf, 5 );
266
267     }
268     ramp(i,rgb) /* Gen. color ramp for "rainbow" */
269     int i, rgb;
270     {
271         int phase, k, v;
272         phase=i-pal_r1+rgb*pal_rw;
273         while (phase<0)
274             phase+=pal_aw;
275         k=phase/pal_rw%3;
276         phase%=pal_aw;
277         if (k==0)
278             return 0;
279         else if (k==1)
280             v=abs(phase-pal_rw)*511L/pal_rw;
281         else
282             v=abs(pal_aw-phase)*511L/pal_rw;
283
284         return v<256?v:255;
285     }
286
```

```
struct window              /* Defines mapping from plot to display */
{
        int dc[3];         /* Center of display (orig 0,0)  Dx,y,c */
        int cp[3];         /* Cursor posit. display coords. Dx,y,c */
        int ss[3];         /* Cursor step size              Dx,y,c */
        int po[3];         /* Plot origin (lower left)     Px,y,c  */
        int p1[3];         /* Plot lower bound             Px,y,c  */
        int p2[3];         /* Plot upper bound             Px,y,c  */
        int zf[6];         /* Zoom factor zf=Pu/Du     if zf>1     */
                           /*               =-Du/Pu+2     zf<1     */
                           /* Color scale zoom=Counts/color if>1   */
                           /*                 =-col/count+2 if<1   */
                           /* Last 3 are for histogram binning     */
        int lf[3];         /* Log flag =0 for lin, =1 for log      */
        int ct;            /* Cursor type =0 off, =1 box, =2 cross */
        int cse;           /* Color scale enable                   */
};
```

```
/*
        Key definitions -

         Prefix:          K_              Standard key
                          C_              Control (^)
                          S_              Shift
                          A_              Alt
         Two-byte keys with leading null are offset
         by 0x0100.

*/
#define K_UP     0x0148
#define K_DN     0x0150
#define K_RT     0x014D
#define K_LEFT   0x014B
#define K_PGUP   0x0149
#define K_PGDN   0x0151
#define K_HOME   0x0147
#define K_END    0x014F
#define K_INS    0x0152
#define K_DEL    0x0153

#define C_RT     0x0174
#define C_LEFT   0x0173
#define C_PGUP   0x0184
#define C_PGDN   0x0176
#define C_HOME   0x0177
#define C_END    0x0175

#define K_F1     0x013B
#define K_F10    0x0144
#define S_F1     0x0154
#define S_F10    0x015D
#define C_F1     0x015E
#define C_F10    0x0167
#define A_F1     0x0168
#define A_F10    0x0171

#define K_CR     0x0D
#define C_CR     0X0A
#define K_BKS    0x08
#define K_TBR    0x09
#define K_TBL    0x010F
#define K_ESC    0x1B
#define C_PRTSC  0x10

#define A_1      0x0178
#define A_9      0x0180
#define A_0      0x0181

#define A_A      0x011E
#define A_B      0x0130
#define A_C      0x012E
#define A_D      0x0120
#define A_E      0x0112
#define A_F      0x0121
#define A_G      0x0122
#define A_H      0x0123
#define A_I      0x0117
#define A_J      0x0124
#define A_K      0x0125
#define A_L      0x0126
#define A_M      0x0132
#define A_N      0x0131
#define A_O      0x0118
#define A_P      0x0119
#define A_Q      0x0110
#define A_R      0x0113
```

```
#define A_S     0x011F
#define A_T     0x0114
#define A_U     0x0116
#define A_V     0x012F
#define A_W     0x0111
#define A_X     0x012D
#define A_Y     0x0115
#define A_Z     0x012C
```

```
/*
        tabl[i]=256*log2(i+256) for 0<=i<=255
 */
int tabl[256]={
2048,2049,2050,2052,2053,2055,2056,2057,2059,2060,2062,2063,2064,2066,2067,2069,
2070,2071,2073,2074,2075,2077,2078,2079,2081,2082,2083,2085,2086,2087,2088,2090,
2091,2092,2094,2095,2096,2097,2099,2100,2101,2102,2104,2105,2106,2107,2109,2110,
2111,2112,2113,2115,2116,2117,2118,2119,2121,2122,2123,2124,2125,2126,2128,2129,
2130,2131,2132,2133,2135,2136,2137,2138,2139,2140,2141,2142,2144,2145,2146,2147,
2148,2149,2150,2151,2152,2153,2154,2156,2157,2158,2159,2160,2161,2162,2163,2164,
2165,2166,2167,2168,2169,2170,2171,2172,2173,2174,2175,2176,2177,2179,2180,2181,
2182,2183,2184,2185,2186,2187,2188,2188,2189,2190,2191,2192,2193,2194,2195,2196,
2197,2198,2199,2200,2201,2202,2203,2204,2205,2206,2207,2208,2209,2210,2210,2211,
2212,2213,2214,2215,2216,2217,2218,2219,2220,2221,2221,2222,2223,2224,2225,2226,
2227,2228,2229,2229,2230,2231,2232,2233,2234,2235,2236,2236,2237,2238,2239,2240,
2241,2242,2242,2243,2244,2245,2246,2247,2248,2248,2249,2250,2251,2252,2253,2253,
2254,2255,2256,2257,2257,2258,2259,2260,2261,2262,2262,2263,2264,2265,2266,2266,
2267,2268,2269,2270,2270,2271,2272,2273,2273,2274,2275,2276,2277,2277,2278,2279,
2280,2280,2281,2282,2283,2283,2284,2285,2286,2287,2287,2288,2289,2290,2290,2291,
2292,2293,2293,2294,2295,2295,2296,2297,2298,2298,2299,2300,2301,2301,2302,2303
};
```

Page Intentionally Left Blank

```
**** TSO FOREGROUND HARDCOPY ****
DSNAME=SE2NL.XTAPEPC.SOURCE                              ($BUILD$ )

//ZW2NLBLD JOB (SB013,BF3,5),'BUILD XTAPE',TIME=(0,30),CLASS=A,          0000
//  MSGCLASS=A                                                          0000
//  EXEC  FORTRAN,FVLINES=58                                            0000
//FORT.SYSIN DD DSN=ZW2NL.XTAPE.SOURCE(MAIN),DISP=SHR                   0000
//  EXEC  ASMG,REGION=300K,TERM=TERM,OPTION='IS=70',                    0000
//  LIB1='SYS2.MACLIB',LIB2='SYS1.MACLIB'                               0000
//SOURCE.SYSIN DD DSN=ZW2NL.XTAPE.SOURCE(XTAPE),DISP=SHR                0000
//  EXEC  LINK                                                          0000
//LINK.SYSLMOD DD DSN=ZW2NL.XTAPE.LOAD,DISP=SHR                         0000
//LINK.OBJECT DD *                                                      0000
 NAME XTAPE(R)                                                        0000
//  EXEC  NOTIFYTS                                                      0000
```

**** TSO FOREGROUND HARDCOPY ****
DSNAME=SE2NL.XTAPEPC.SOURCE ($NOTES$)

XTAPE creates a file that is directly usable by the 11/70 in TIMPLT and similar programs. Input to XTAPE consists of two files prepared by FLUXPLOT. The first of these files is the index to

```
**** TSO FOREGROUND HARDCOPY ****
DSNAME=SE2NL.XTAPEPC.SOURCE                                    (MAIN    )

      CHARACTER PARMST*120,NULL*4,FILENM*8                                  0000
      INTEGER COUNT                                                         0000
      DATA NULL/'NULL'/                                                     0000
      CALL PARM(COUNT,PARMST)                                               0000
      IF (COUNT .GT. 4) STOP 8                                              0000
      IF (COUNT .EQ. 4 .AND. PARMST(1:4) .EQ. NULL) GO TO 100               0000
         FILENM = 'FILE'                                                    0000
         IF (COUNT .GT. 0) FILENM = PARMST(1:COUNT)//'FILE'                 0000
         WRITE (6,10) FILENM                                                0000
   10    FORMAT(' OUTPUT FILE NAME: ',A)                                    0000
         CALL XTAPE                                                         0000
         STOP                                                               0000
  100 CONTINUE                                                              0000
      STOP                                                                  0000
      END                                                                   0000
```

**** TSO FOREGROUND HARDCOPY ****
DSNAME=SE2NL.XTAPEPC.SOURCE (PROGLOG)

Changes of July 1984 00001
Length of saetellite code (SATCODE) was changed to 2, and 00001
satellite code was moved into columns 6 and 7 of bin 00001
definition record. Earlier the length was 3 and the code 00001
occupied first three bytes of the bin definition record. 00001

```
//

**** TSO FOREGROUND HARDCOPY ****
DSNAME=SE2NL.XTAPEPC.SOURCE                                    (XTAPE    )

         MACRO                                                          0000
&NAME    I2    &LOC                                                     0000
         LA    1,&LOC                                                   0000
         LH    0,0(1)                                                   0000
         STC   0,0(1)                                                   0000
         SRL   0,8                                                      0000
         STC   0,1(1)                                                   0000
         MEND                                                           0000
*                                                                       0000
         MACRO                                                          0000
&NAME    I4    &LOC                                                     0000
         LA    1,&LOC                                                   0000
         L     0,0(1)                                                   0000
         STC   0,0(1)                                                   0000
         SRL   0,8                                                      0000
         STC   0,1(1)                                                   0000
         SRL   0,8                                                      0000
         STC   0,2(1)                                                   0000
         SRL   0,8                                                      0000
         STC   0,3(1)                                                   0000
         MEND                                                           0000
*                                                                       0000
         MACRO                                                          0000
&NAME    R4    &LOC                                                     0000
         SR    14,14                                                    0000
         SR    1,1                                                      0000
         L     15,&LOC                                                  0000
         SLDL  14,1                                                     0000
         LR    0,14                                                     0000
         SLL   0,8                                                      0000
         SR    14,14                                                    0000
         SLDL  14,7                                                     0000
         IC    0,EXPTAB(14)                                             0000
         SR    14,14                                                    0000
         SLDL  14,4                                                     0000
         IC    1,ADDTAB(14)                                             0000
         SR    0,1                                                      0000
         IC    1,SHIFTAB(14)                                            0000
         SRDL  14,0(1)                                                  0000
         LR    14,0                                                     0000
         SRDL  14,9                                                     0000
         LA    1,&LOC                                                   0000
         STC   15,0(1)                                                  0000
*        SRL   15,8                                                     0000
         STCM  15,B'0010',1(1)                                          0000
*        SRL   15,8                                                     0000
         STCM  15,B'0100',2(1)                                          0000
*        SRL   15,8                                                     0000
         STCM  15,B'1000',3(1)                                          0000
         MEND                                                           0000
*                                                                       0000
         MACRO                                                          0000
&NAME    R8    &LOC,&WREG=9                                             0000
         GBLA  &TIMETHR
         AIF   (&TIMETHR GT 0).NOT1ST
         B     R8BR

...
```

```
          DS    0F
MASKSIGN  DC    X'80000000'
MASKCOMP  DC    X'7FFFFFFF'
BIAS      DC    X'38000000'
R8BR      DS    0H
.NOT1ST   ANOP
&TIMETHR  SETA  &TIMETHR+1
          SR    14,14                                                  0000
          SR    1,1                                                    0000
          L     15,&LOC                                                0000
          SLDL  14,1                                                   0000
          LR    0,14                                                   0000
          SLL   0,8                                                    0000
          SR    14,14                                                  0000
          SLDL  14,7                                                   0000
          IC    0,EXPTAB(14)                                           0000
          SR    14,14                                                  0000
          SLDL  14,4                                                   0000
          SR    &WREG.,&WREG                                           0000
          IC    &WREG,ADDTAB(14)                                       0000
          SR    0,&WREG                                                0000
          IC    1,SHIFTAB(14)                                          0000
          SRDL  14,0(1)                                                0000
          LR    14,0                                                   0000
          SLDL  14,20(1)                                               0000
          LA    15,&LOC                                                0000
          L     15,4(15)                                               0000
          SLDL  14,0(&WREG.)                                           0000
          LR    0,14
          N     0,MASKSIGN
          A     0,BIAS
          N     14,MASKCOMP
          SRDL  14,3
          AR    14,0
          LA    1,&LOC                                                 0000
          STC   14,4(1)                                                0000
          STCM  14,B'0010',5(1)                                        0000
          STCM  14,B'0100',6(1)                                        0000
          STCM  14,B'1000',7(1)                                        0000
          STC   15,0(1)                                                0000
          STCM  15,B'0010',1(1)                                        0000
          STCM  15,B'0100',2(1)                                        0000
          STCM  15,B'1000',3(1)                                        0000
          MEND                                                         0000
XTAPE     PROC  LINKAGE=(OS,CSECT)                                     0000
          CALL  FREAD,(DTHEADER,UNIT31,LEN31),VL                       0000
          IF    (LTR,15,15,NZ)                                         0000
                ABEND 1113,DUMP                                        0000
          FI                                                           0000
          LH    0,DTNOBINS                                             0000
          ST    0,XNUMBIN                                              0000
          LH    0,DTLCOUNT          ???                                0000
*                                                                      0000
          LH    0,DTLRECL                                              0000
          ST    0,XLRECL                                               0000
          GETMAIN R,LV=(0)                                             0000
          LR    2,1                                                    0000
*                                                                      0000
          I2    DTNOBINS                                               0000
          I2    DTLCOUNT                                               0000
```

```
         I2    DTIRECS                                                  0000
         I2    DTLRECL                                                  0000
         I2    DTBLKSIZ                                                 0000
         I2    DTKBYR                                                   0000
         R8    DTSTIME                                                  0000
         R8    DTETIME                                                  0000
         R8    DTAVET                                                   0000
*                                                                       0000
         L     3,XNUMBIN                                                0000
         LA    3,1(3)                                                   0000
         SDR   0,0                                                      0000
         LR    4,2                                                      0000
         DO    UNTIL,(BCT,3)                                            0000
               STD   0,0(4)                                             0000
               LA    4,8(4)                                             0000
         OD                                                             0000
         TR    DTFNAME,EBCASC                                           0000
         TR    DTPNAME,EBCASC                                           0000
         CALL  ZTIME,(ZTOD,ZCODE),VL                                    0000
         MVC   DTDATE(2),ZDATE+8                                        0000
         MVI   DTDATE+2,C'-'                                            0000
         MVC   DTDATE+3(3),ZDATE+4                                      0000
         MVI   DTDATE+6,C'-'                                            0000
         MVC   DTDATE+7(2),ZDATE+13                                     0000
         MVC   DTDATE+9(3),ZDOY+3                                       0000
         MVC   DTTIME,ZTOD                                              0000
         MVI   DTTIME+2,C'-'                                            0000
         MVI   DTTIME+5,C'-'                                            0000
*                                                                       0000
         TR    DTDATE,EBCASC                                            0000
         TR    DTTIME,EBCASC                                            0000
         MVC   0(132,2),DTHEADER                                        0000
         CALL  FWRITE,((2),UNIT33,XLRECL),VL                            0000
         LA    0,132                                                    0000
         LCR   0,0                                                      0000
         A     0,XLRECL                                                 0000
         ST    0,X132                                                   0000
*                                                                       0000
         CALL  FREAD,(AREAADR,UNIT32L,LEN32),VL                         0000
         L     3,AREAADR                                                0000
         MVC   (DDLCOUNT-DDHEADER)(2,3),DTLCOUNT                        0000
         MVC   (DDNOBINS-DDHEADER)(2,3),DTNOBINS                        0000
         MVC   (DDKBYR-DDHEADER)(2,3),DTKBYR                            0000
         MVC   (DDAVET-DDHEADER)(8,3),DTAVET                            0000
         L     0,LEN32                                                  0000
         SRDA  0,32                                                     0000
         D     0,XLRECL                                                 0000
* R1=NUMBER OF LOGICAL RECORDS. COMPUTE NUMBER OF RECORDS THAT          0000
* THAT PRECEDE DATA RECORDS                                             0000
         LR    7,1                                                      0000
         LA    8,1   NUMBER OF LOGICAL RECORDS PROCESSED                0000
         LA    6,1   NUMBER OF LEGEND RECORDS PROCESSED                 0000
         L     5,XNUMBIN                                                0000
         LA    5,2(5)                                                   0000
         A     3,XLRECL                                                 0000
         DO    WHILE,(CR,6,5,LT)                                        0000
               DO    WHILE,(CR,8,7,LT),AND,(CR,6,5,LT)                  0000
                     LA    0,1                                          0000
                     IF    (CR,0,6,EQ) IS THE FIRST                     0000
                           LA    15,STABLE                              0000
```

```
                        L       1,NUMSAT                                    0000
                        DO      UNTIL,(BCT,1)                               0000
                                IF      (CLC,0(12,3),0(15),EQ),EXIT=*       0000
                                LA      15,16(15)                           0000
                        ONEXIT                                              0000
                                MVC     SATCODE,12(15)                      0000
                        ATEND                                               0000
                                ABEND 1123,DUMP                             0000
                        OD                                                  0000
                    ELSE                                                    0000
                                MVC     5(L'SATCODE,3),SATCODE              0000
                    FI                                                      0000
                    TR      0(132,3),EBCASC                                 0000
                    LA      0,132(3)                                        0000
                    L       1,X132                                          0000
                    SR      15,15                                           0000
                    LA      14,132(2)                                       0000
                    MVCL    0,14                                            0000
                    A       3,XLRECL                                        0000
                    LA      6,1(6)                                          0000
                    LA      8,1(8)                                          0000
                OD                                                          0000
                IF      (CR,8,7,LT)                                         0000
                        SR      7,8                                         0000
                        LR      1,7                                         0000
                        BAL     14,CONVERT                                  0000
                FI                                                          0000
                CALL    FWRITE,(AREAADR,UNIT33L,LEN32),VL                   0000
                IF      (CR,6,5,LT)                                         0000
                        CALL FREAD,(AREAADR,UNIT32L,LEN32),VL               0000
                        L       3,AREAADR                                   0000
                        SR      8,8                                         0000
                        L       1,LEN32                                     0000
                        SR      0,0                                         0000
                        D       0,XLRECL                                    0000
                        LR      7,1                                         0000
                FI                                                          0000
            OD                                                              0000
*                                                                           0000
            DO                                                              0000
                CALL FREAD,(AREAADR,UNIT32L,LEN32),VL                       0000
                IF      (LTR,15,15,NZ),EXIT=*                               0000
                L       3,AREAADR                                           0000
                L       0,LEN32                                             0000
                SRDA    0,32                                                0000
                D       0,XLRECL                                            0000
                BAL     14,CONVERT                                          0000
                CALL FWRITE,(AREAADR,UNIT33L,LEN32),VL                      0000
            OD                                                              0000
            CORP                                                            0000
SATCODE     DS      CL2                                                     0000
XNUMBIN     DS      F                                                       0000
XLRECL      DS      F                                                       0000
X132        DS      F                                                       0000
AREAADR     DS      F                                                       0000
UNIT31      DC      F'31'                                                   0000
UNIT32      DS      F'32'                                                   0000
UNIT32L     DC      F'-32'                                                  0000
UNIT33      DC      F'33'                                                   0000
UNIT33L     DC      F'-33'                                                  0000
```

```
LEN31    DS    F                                                      0000
LEN32    DS    F                                                      0000
ZTOD     DS    C'HH.MM.SS.TH '                                        0000
ZDATE    DS    C'DDD.MMM.DD.YYYY '                                    0000
ZDOY     DS    C'YY.DDD '                                             0000
         DS    0F                                                     0000
ZCODE    DC    A(1+2+4)                                               0000
*                                                                     0000
CONVERT  PROC  SAVE=14                                                0000
         LR    4,1                                                    0000
         ST    3,CONVR3                                               0000
         DO    UNTIL,(BCT,4)                                          0000
               LD    0,0(3)                                           0000
               R8    0(3)                                             0000
               LA    3,8(3)                                           0000
               L     5,XNUMBIN                                        0000
               DO    UNTIL,(BCT,5)                                    0000
                     R4    0(3)                                       0000
                     R4    4(3)                                       0000
                     LA    3,8(3)                                     0000
               OD                                                     0000
               L     3,CONVR3                                         0000
               A     3,XLRECL                                         0000
               ST    3,CONVR3                                         0000
         OD                                                           0000
*                                                                     0000
         CORP                                                         0000
CONVR3   DS    F                                                      0000
EXPTAB    DC    32X'00'                                              0000
          DC    X'0002060A0E12161A1E22262A2E32363A'                  0000
          DC    X'3E42464A4E52565A5E62666A6E72767A'                  0000
          DC    X'7E82868A8E92969A9EA2A6AAAEB2B6BA'                  0000
          DC    X'BEC2C6CACED2D6DADEE2E6EAEEF2F6FA'                  0000
          DC    32X'FF'                                              0000
SHIFTAB   DC    X'00000101020202020303030303030303'                  0000
ADDTAB    DC    X'00030202010101010000000000000000'                  0000
MAXIMUM   DS    X'60FFFFFFFFFFFF'                                    0000
MINIMUM   DS    X'21100000000000'                                    0000
DTHEADER DS    0D                                                     0000
DTFNAME  DC    CL28'FLXPLT.FLM'                                       0000
DTNOBINS DS    H                                                      0000
DTLCOUNT DC    H'0'                                                   0000
DTIRECS  DS    H                                                      0000
DTLRECL  DS    H                                                      0000
DTBLKSIZ DS    H                                                      0000
DTKBYR   DS    H                                                      0000
DTSTIME  DS    D                                                      0000
DTETIME  DS    D                                                      0000
DTAVET   DS    D                                                      0000
DTDATE   DS    C'DD-MMM-YRDDD'                                        0000
DTTIME   DS    C'HH:MM:SS'                                            0000
         DC    2H'0'                                                  0000
DTPNAME  DC    CL8'FLUXPLOT'                                          0000
DTUSED   EQU   *-DTHEADER                                            0000
         DC    XL(132-DTUSED)'00'                                     0000
         DS    0D                                                     0000
*                                                                     0000
         DS    F                                                      0000
DDHEADER DS    0F                                                     0000
DDNOBINS DS    H                                                      0000
```

```
DDLCOUNT DS    H                                                        0000
DDAVET   DS    D                                                        0000
DDKBYR   DS    H                                                        0000
DDUSED   EQU   *-DDHEADER                                               0000
         DS    0D                                                       0000
EBCASC   DC    X'000102030409067F08090A0B0C0D0E0F'                      0000
         DC    X'10111213141508171819 1A1B1C1D1E1F'                     0000
         DC    32X'3F'                                                  0000
         DC    X'203F3F3F3F3F3F3F3F5B2E3C282B21'                        0000
         DC    X'263F3F3F3F3F3F3F3F5D242A293B5E'                        0000
         DC    X'2D2F3F3F3F3F3F3F3F7C2C255F3E3F'                        0000
         DC    X'3F3F3F3F3F3F3F3F3F603A2340273D22'                      0000
         DC    X'3F6162636465666768693F3F3F3F3F3F'                      0000
         DC    X'3F6A6B6C6D6E6F7071723F3F3F3F3F3F'                      0000
         DC    X'3F7E737475767778797A3F3F3F3F3F3F'                      0000
         DC    16X'3F'                                                  0000
         DC    X'7B41424344454647484 93F3F3F3F3F3F'                     0000
         DC    X'7D4A4B4C4D4E4F5051523F3F3F3F3F3F'                      0000
         DC    X'5C3F535455565758595A3F3F3F3F3F3F'                      0000
         DC    X'30313233343536373839 3F3F3F3F3F3F'                     0000
STABLE   DS    0D                                                       0000
         DC    CL12'ISEE-3',CL4'IC'                                     0000
         DC    CL12'HELIOS-A',CL4'HA'                                   0000
         DC    CL12'HELIOS-B',CL4'HB'                                   0000
         DC    CL12'VOYAGER-1',CL4'VA'                                  0000
         DC    CL12'VOYAGER-2',CL4'VB'                                  0000
         DC    CL12'PIONEER-F',CL4'PF'                                  0000
         DC    CL12'PIONEER-G',CL4'PG'                                  0000
NUMSAT   DC    F'7'                                                     0000
         END                                                            0000
```