DEPARTMENT OF THE INTERIOR

U.S. GEOLOGICAL SURVEY

A user's guide to the PC-based time-series data-management and plotting program BOB

by

Thomas L. Murray[1]

[1] Cascades Volcano Observatory
5400 MacArthur Blvd.
Vancouver, Washington, 98661

Open-File Report 90-56

This report is preliminary and has not been reviewed for conformity with U.S. Geological Survey editorial standards. Any use of trade, product, or firm names is for descriptive purposes only and does not imply endorsement by the USGS. Although this program has been used by the U.S. Geological Survey, no warranty, expressed or implied, is made by the USGS as to the accuracy and functioning of the program and related program material, nor shall the fact of distribution constitute any such warranty, and no responsibility is assumed by the USGS in connection therewith.

Table of Contents

Introduction

PC-BOB is a command-driven FORTRAN program that allows the user to quickly plot and compare time-series data from low-data-rate volcanic predictors (sampling periods greater than or equal to one minute). Data is imported from permanent data files and placed in what is best thought of as columns in an eight column spreadsheet. Once in a column, the data can be averaged, temperature corrected, have offsets removed, or other common techniques for putting the data into a usable format. These modifications do not affect the permanent data files unless specifically instructed to do so. The data is plotted either by itself or with data from any of the other 7 columns for comparison purposes.

Macros can be set up so that these functions can be done automatically. These macros allow users with only a minimal knowledge of computers to also access the data (note that to set up the macros does require a good working knowledge of BOB). A separate menu program (examples are included in the sub-directory MENUS) uses the macros to provide cleaned and scaled plots with little effort.

Read This

This manual does not cover setting up BOB and the data files. This manual assumes BOB is installed, the data files are being periodically updated, and now the user wants to look at the data.

Do read the following section "Columns and Headers" before attempting to produce a plot. It will make things much clearer. And don't be afraid to just try a command to see what it does. Since BOB reads the data into its columns and does all the work on the data there, it is virtually impossible to destroy or foul-up the raw data set no matter how hard you try.

The examples in this manual use the sample data for St. Helens created when BOB was installed from the six diskettes. In the examples, **bold-faced** type indicates your input, normal type indicates the computer's output, and comments are enclosed in brackets. To avoid repetition, non-pertinent computer output is not necessarily shown.

All user commands **must** be UPPER CASE.

To run BOB:
1) Get into the directory BOB.
2) Run BOB by typing **BOB**. The prompt

```
COMMAND?
```

should appear.

Because BOB looks for certain set-up files in relation to the BOB directory, you must run BOB from its directory, not from elsewhere through the use of DOS's PATH statement.

To exit BOB type:

```
COMMAND? BYE
```

To return to command mode when a plot is on the screen, merely press any key.

For VAX BOB users, the major difference between VAX BOB and PC BOB is in the format for the date. VAX BOB uses integer month, integer day, and integer year. PC BOB use integer day, three character month, integer year. For example:

```
05 DEC 87    [PC BOB date format (european style)]
12 05 87     [VAX BOB date format (SAS style)]
```

Columns and Headers

BOB places data in any of eight columns. Each column has two headers associated with it:

The column header indicates from which measurement the column's data came, the time period of the data, the high and low data values, and the number of valid data points in the particular column. The column header's primary function is to provide the user with information about the data contained in the column.
The command **SHOHDR A** displays the header for column A, **SHOHDR B** for column B etc.

The plot header indicates the style of the plot, its title and label, and the y axis scale. The user changes the default values in the plot header to obtain the plot desired.
The command **PLOTHDR** displays the plot header.

A third header, called the fill header (or simply the header) indicates the time period and which measurement's data is to be placed in one of the columns. The parameters in this header are set by the user before using the **FILL** command to place data in a column.
The command **SHOHDR** displays the fill header.

The eight columns are designated A through H. The available disk space is the only limit to the number of data points a column can hold.

The time corresponding to each datum in the column is determined by its place in the column. The time for the initial datum is held in the column's header - the beginning julian day and year. The initial datum is always the first datum for that day, 00:00 hours. The time for succeeding data points is determined by the data rate or points/day. For 24 points/day the interval between suceeding positions in the column is one hour. For 144 points/day it is 10 minutes (1440 minutes/day divided by 144 points/day). From the information in the column header (beginning time and data points/day) and the position of a particular data point (how far from the beginning of the column) its time can be calculated. Note that for this scheme to work, there must be some way to designate that a time slot does not have a valid datum associated with it. If there is no datum for a specific time slot in the column the number -998 is inserted. -998 is the missing data indicator and will not be recognized as a true data value.

The maximum number of days of data a column can hold is limited primarilly by available disk space, but also with the fact that BOB cannot deal with the crossing of a century boundary yet (i.e. 1880-1958).

The minimum number of days that BOB can plot is one. BOB cannot plot, for example, only 4 hours of data. The number of days is always an integer value.

Your First Plot

The following example produces the plot in figure 1. It is intended to show how easy it is to produce a plot and also to insure that BOB has been installed correctly. If figure 1 does not appear, either BOB has not been installed properly or the example data files used throughout this manual have been erased. In either case consult the BOB installation manual or your local BOB expert.

To produce figure 1 enter the following commands:

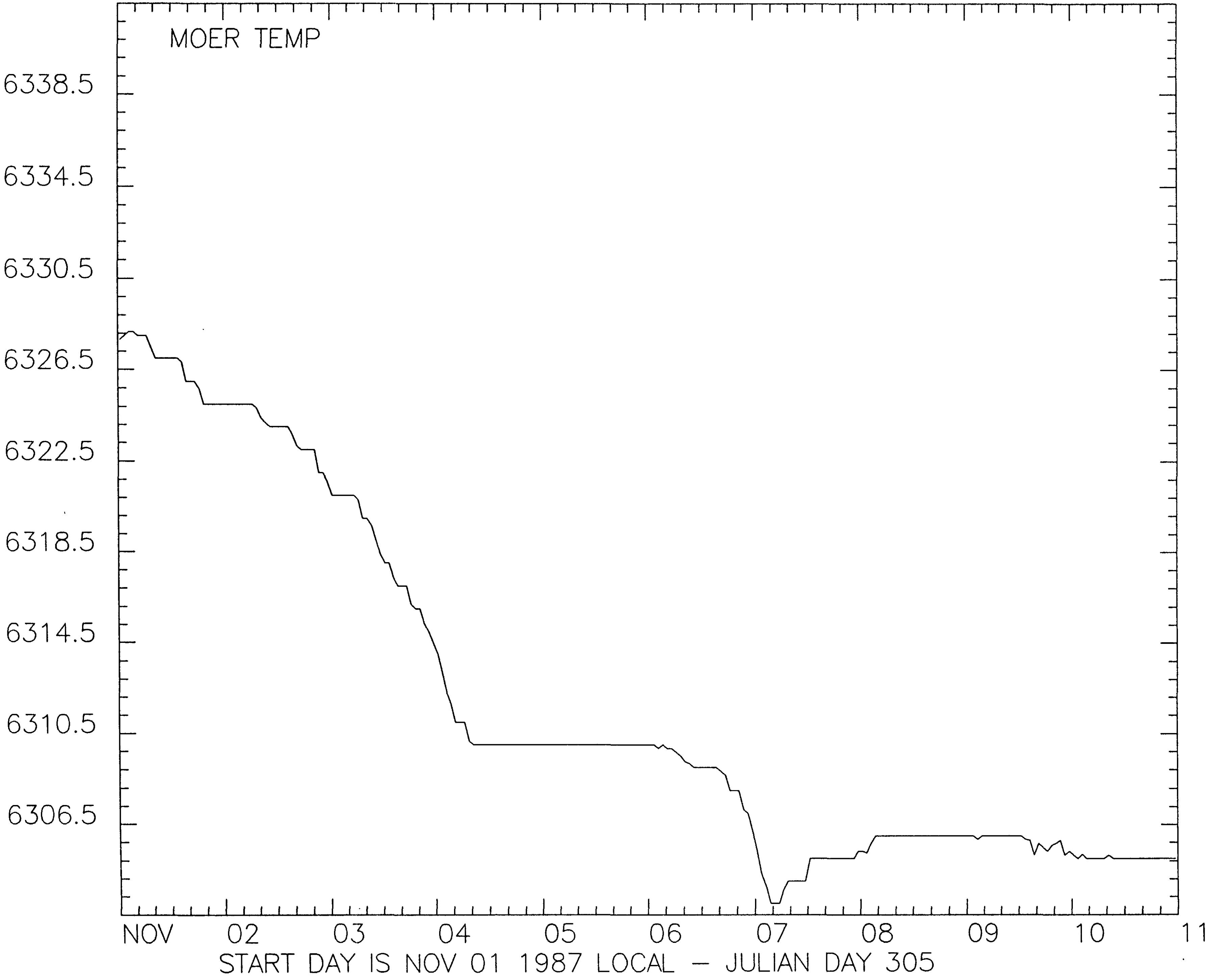
```
                                      [first, indicate the measurement to be
                                         plotted]

COMMAND? STID MOER                    [the field station is set to MOER]
COMMAND? MEAS TEMP                    [the TEMP measurement at station MOER
is                                       selected]

                                      [second, indicate the time period of
                                         interest]
COMMAND? B_DATE 01 NOV 87             [the time period of interest begins 01
                                         Nov 87]
COMMAND? E_DATE 10 NOV 87             [and ends on 10 Nov 87 (inclusive)]

                                      [third, FILL a column with the data]
COMMAND? FILL A                       [put the data into column A]
   HEADER FOR COLUMN A
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=      240  OUT OF       240  HIGH=     6328.17  LOW=     6303.00

                                      [fourth, PLOT the data on the screen]
COMMAND? PLOT 1 A                     [plot the data in column A on the
screen]
```

To erase the screen and return to command mode press any key.

Of course other commands exist to change the style or title of the plot, but the above are the primary commands.

The commands can also be put into a file or MACRO and have that file act as the input for BOB. This allows the setting up of "canned" plotting routines for users not familiar with the BOB command set, or automatic printing of plots from numerous stations. This is described in the section Making and the Use of Macros.

FIG. 1 FAST PLOT



Specifying the Measurement to be put into a Column

To choose the measurement you must specify to BOB the **VOLCANO**, the four character station identifier (STID), and finally the four character measurement designation (MEAS). The following table shows the station identifiers and measurement designations used in the examples for this manual.

STID	MEAS
MOER - tiltmeter located at Moe Rock	TEMP - temperature
	RADL - radial tilt
	TANG - tangential tilt
COSP - Cospec gas measurements	SO2_ - SO2 flux
	WSPD - Wind speed
	ERRB - Error bar
CURL - Curly's Ridge EDM setup	STPD - Curlys to Step distance
	SAUD - Curlys to Sauna Rock distance

Specifying the volcano, station and measurement is done with the commands **VOLCANO**, **STID** and **MEAS**.

```
COMMAND? RESET                         [reset BOB to its start-up state]
COMMAND? SHOHDR                        [display contents of the fill header]
 FILL HEADER
VOLCANO=V1201_05_    STATION=****      MEASUREMENT=****.DAT
BEG. JULIAN DAY=000    BEG. YEAR=00       TOTAL DAYS=0000   POINTS/DAY=0144
                                       [the above are the initial values. the
                                          volcano defaults to Mount St.
                                          St. Helens - V1201_05_]
COMMAND? VOLCANO                       [choose the volcano]
 YOUR CHOICES ARE:
** V1201_05_                            Mount St. Helens, WA, USA
** V1201_04_                            Mount Adams, USA
** BULL__RN_                            Bull Run Watershed, OR, USA

 ENTER THE 9 CHARACTER CHOICE: V1201_05_
                                       [note that only St. Helens has any data
                                          files on the distribution disks. the
                                          other two are purely for this
                                          example.]
COMMAND? STID MOER                     [station id MOER]
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_    STATION=MOER      MEASUREMENT=****.DAT
BEG. JULIAN DAY=000    BEG. YEAR=00       TOTAL DAYS=0000   POINTS/DAY=0144
COMMAND? MEAS RADL                     [measurement RADL]
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_    STATION=MOER      MEASUREMENT=RADL.DAT
BEG. JULIAN DAY=000    BEG. YEAR=00       TOTAL DAYS=0000   POINTS/DAY=0144
```

The SHOHDR command is included only to show how the fill header is changed by each command. It need not be used when setting the STID or MEAS. BOB starts off with a default volcano (usually Mount St. Helens) and the **VOLCANO** command can be eliminated if the desired measurement is located on that volcano.

If you cannot remember the four character station ids for a volcano, enter the command **STATIONS**. This will display the documentation file STATIONS.TXT for that particular volcano and it should include the four character station ids. If you cannot remember the measurement designations for a particular station, enter the command **MEASURES**. This will display the documentation file BOB_stid.TXT for the station and should include the four character measurement designations.

Using the above example, if after indicating the correct volcano (St. Helens - V1201_05_) you were unsure of the four character STID for Moe Rock, you could have typed:

```
COMMAND? STATIONS
```

and a listing of the four character STIDs for St. Helens would be listed along with other information.

If, after choosing **STID MOER**, you needed to know the four character MEAS for temperature you could have typed:

```
COMMAND? MEASURES
```

and a listing of the four character MEASs would be listed along with other information.

*Note that though the files displayed when using the STATIONS and MEASURES commands are supposed to be created when adding new volcanos and stations to BOB, they may not have been. If no information is displayed, complain loudly to the appropriate person and threaten to erase his data unless he creates the files.

Specifying the Time Period of Interest

The time period of interest is defined by the beginning julian day and year for the period, and the total number of days in the period. There are three methods of doing this.

1) <u>Explicitly specifying the beginning date and the length of the time period</u>

The time period of interest's beginning julian day, beginning year, and length in days can be explicitly defined with the commands **BJL**, **BYR**, and **DAYS**. To set the time period to be 310 days long and starting on julian day 13 of 1987 enter:

```
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_     STATION=MOER       MEASUREMENT=RADL.DAT
BEG. JULIAN DAY=000    BEG. YEAR=00         TOTAL DAYS=0000    POINTS/DAY=0144
COMMAND? BJL 13                          [starting julian day is 13]
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_     STATION=MOER       MEASUREMENT=RADL.DAT
BEG. JULIAN DAY=013    BEG. YEAR=00         TOTAL DAYS=0000    POINTS/DAY=0144
COMMAND? BYR 87                          [beginning year is 1987]
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_     STATION=MOER       MEASUREMENT=RADL.DAT
BEG. JULIAN DAY=013    BEG. YEAR=87         TOTAL DAYS=0000    POINTS/DAY=0144
COMMAND? DAYS 310                        [time period is 310 days long]
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_     STATION=MOER       MEASUREMENT=RADL.DAT
BEG. JULIAN DAY=013    BEG. YEAR=87         TOTAL DAYS=0310    POINTS/DAY=0144
```

The commands can be entered in any order and the **SHOHDR** command is again included only to show the effect of the other commands.

A variation on this method is to use the B_DATE and E_DATE commands to specify the time period instead of **BJL**, **BYR**, and **DAYS**. To get the same time period as above you could also have typed:

```
COMMAND? B_DATE 13 JAN 87                [set the beginning date to 13 Jan 87]
                                         [Note that VAX BOB users would type:
                                           COMMAND? B_DATE 01 13 87         ]
COMMAND? E_DATE 28 FEB 87                [and the ending date]
                                         [again, VAX BOB users would type:
                                            COMMAND? E_DATE 02 28 87      ]
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_     STATION=MOER       MEASUREMENT=RADL.DAT
BEG. JULIAN DAY=001    BEG. YEAR=87         TOTAL DAYS=0047    POINTS/DAY=0144
```

Note that E_DATE will not work unless a specific beginning time has been defined either with **BJL** and **BYR** or with **B_DATE**.

2) Indicating the time period is to go back a specific number of days from the present

For instance:

```
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_    STATION=MOER       MEASUREMENT=RADL.DAT
BEG. JULIAN DAY=100    BEG. YEAR=88        TOTAL DAYS=0010    POINTS/DAY=0144
COMMAND? LAST 7                         [get data for the last 7 days]
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_    STATION=MOER       MEASUREMENT=RADL.DAT
BEG. JULIAN DAY=038    BEG. YEAR=88        TOTAL DAYS=0007    POINTS/DAY=0144
```

will set the time period for 7 days with today (as determined by the computers clock) as the last day. In the above example, the computers current date was 13 Feb 88 (julian day 44).

```
COMMAND? LAST 365                       [get data for the last 365 days]
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_    STATION=MOER       MEASUREMENT=RADL.DAT
BEG. JULIAN DAY=045    BEG. YEAR=87        TOTAL DAYS=0365    POINTS/DAY=0144
```

will go back 365 days from the present or one year. The **LAST** command is particularly useful in real-time operations, where you are interested in what's currently happening.

NOTE - Because **LAST** references the current time as indicated by the computer's clock, the results of your typing **LAST** will differ from that listed in the examples.

3) The time period is to be from a specific date in the past to the present

The final method of defining a time period is the inverse of the **LAST** command. The beginning time is set with the **BJL** and **BYR** commands. Using the **TONOW** command will set the days equal to the number of days between the beginning time and the present. This allows you to begin at a specific time and always go right up to the present. To start the time period on Jan. 1, 1988 (julian day 001) and go to the present time, 13 Feb 1988 enter:

```
COMMAND? BJL 1                       [define the starting julian day]
COMMAND? BYR 88                      [and year]
COMMAND? TONOW                       [set the number of days to go from
                                       then up to the present day]
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_    STATION=MOER     MEASUREMENT=RADL.DAT
BEG. JULIAN DAY=001    BEG. YEAR=88      TOTAL DAYS=0044    POINTS/DAY=0144
```

The **SHOHDR** is included just to demonstrate how **TONOW** adjusts the days to the proper number. Note that **TONOW** will not work if no start time is specified.

NOTE - Because **TONOW** references the current time as indicated by the computer's clock, the results of your typing **TONOW** will differ from that listed in the examples.

Getting the Data into a Column

The FILL command is used to read the data from the data files into one of the columns. After setting the fill header with the desired volcano, station id, measurement id, and time period, the FILL command is used to read the data from the data files into one of the eight columns (A-H). An example showing how to put the data into column A:

```
COMMAND? STID MOER                      [define the station and measurement]
COMMAND? MEAS RADL
COMMAND? BJL 305                        [define the time period]
COMMAND? BYR 87
COMMAND? DAYS 10
COMMAND? SHOHDR
 FILL HEADER
VOLCANO=V1201_05_    STATION=MOER       MEASUREMENT=RADL.DAT
BEG. JULIAN DAY=305    BEG. YEAR=87        TOTAL DAYS=0010    POINTS/DAY=0144
COMMAND? FILL A                         [fill column A with the data defined in
                                          the fill header]
   HEADER FOR COLUMN A
STATION=MOER CHANNEL=RADL.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=      239   OUT OF         240   HIGH=     2892.60   LOW=     2872.83
```

If the fill was succesful, the header for the column just filled will come back with information about the data in the column. VALID refers to how many valid (non -998) data points are in the column. OUT OF refers to the total data points in the column (the total equals rate times days). HI is the highest data point in the column, LO the lowest.

To see the data contained in a column, use the **SHODATA** command.

```
COMMAND? SHODATA A                      [display the contents of column A]
   HEADER FOR COLUMN A
STATION=MOER CHANNEL=RADL.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=      239   OUT OF         240   HIGH=     2892.60   LOW=     2872.83
 ------------------------------------------------------------------------------
FOLLOWING DATA STARTS AT 305:00:00  1987
  LOC       +000:00:00 +000:01:00 +000:02:00 +000:03:00 +000:04:00 +000:05:00
 ------------------------------------------------------------------------------
305:00:00      2885.80    2885.00    2884.80    2884.30    2883.40    2884.30
305:06:00      2885.00    2885.00    2885.30    2886.00    2886.30    -998.00
305:12:00      2886.80    2887.00    2886.80    2886.80    2887.30    2888.00

                            [etc.]
```

The data is read from left (earliest) to right. The time of the first reading in each line is at the left side of the line. The time just above the data (on the line starting with LOC -meaning LOCal time) is the time interval to ADD to the the time on the left side of the line to determine the time of the succeding data points on the line. For instance, the time for the datum third from the left on the second line (2885.30) is 305:06:00 + 000:02:00 or 305:08:00, i.e. 08:00 on julian day 305.

NOTE that the value for 305:11:00 is the missing data value -998.00. No data exists for that time.

The above FILL was done from a BOB data file. BOB data files are direct-access binary files holding one year of data for a specific measurement. Their format is the same as a BOB column with a specific starting time and time interval between succeding data points. -998 indicates missing data (see the section Columns and Headers). Because of the similar formats, BOB reads these data files quickly and easily.

The other type of data file BOB can access is called a European-date sequential file. Data stored in these files are aperiodic and therefore do not have a specific interval between succeding data points. Each datum has its time stated explicitly, not calculated from its relative position in the file.

If you try to fill data from a European-date sequential data file you will be prompted for how many data points/day (the RATE). Since the sequential files do not have a fixed time interval between succesive data points and the columns do, you have to tell BOB how many data points/day you want for the FILL. Example:

```
COMMAND? STID COSP
COMMAND? MEAS SO2_                   [S O (as in oxygen) 2, not S 0 (zero) 2]
COMMAND? BJL 290
COMMAND? DAYS 10
COMMAND? BYR 86
COMMAND? FILL A
HOW MANY POINTS/DAY DO YOU WANT FOR THIS FILL ? 1
   HEADER FOR COLUMN A
STATION=COSP MEASMNT=SO2_.DAT BEGIN JL=290 BEGIN YR=86 #DAYS=0010 PTS/DAY=0001
VALID POINTS=        6  OUT OF         10  HIGH=     675.00  LOW=     20.00
```

If more then one data point exists for a specific time slot in the column, the last data point from the file that fits into that slot is the one used. To retrieve the data dropped in this manner, increase the data points/day. In the above example, only 6 valid points were put into the column. By increasing the rate from 1 point/day to 24 points/day, we get 7 valid points, as shown by the example below.

```
COMMAND? FILL A
HOW MANY POINTS/DAY DO YOU WANT FOR THIS FILL ? 24  [go with hourly data]
   HEADER FOR COLUMN A
STATION=COSP MEASMNT=SO2_.DAT BEGIN JL=290 BEGIN YR=86 #DAYS=0010 PTS/DAY=0024
VALID POINTS=        7  OUT OF        240  HIGH=     675.00  LOW=     20.00
```

The extra data point retrieved was in the same day as another data point, hence not retrieved when gathering only one data point/day. But the extra point was in a different hour. By increasing the rate to 24 data points/day, both data points have their own slot in the column and both are retrieved. Experimentation may be needed to determine the optimum rate.

If a data file cannot be found, BOB will ask if you wish to continue the FILL. If you do wish to continue, BOB will insert -998's into the array to indicate that the data was missing.

Cleaning the Data

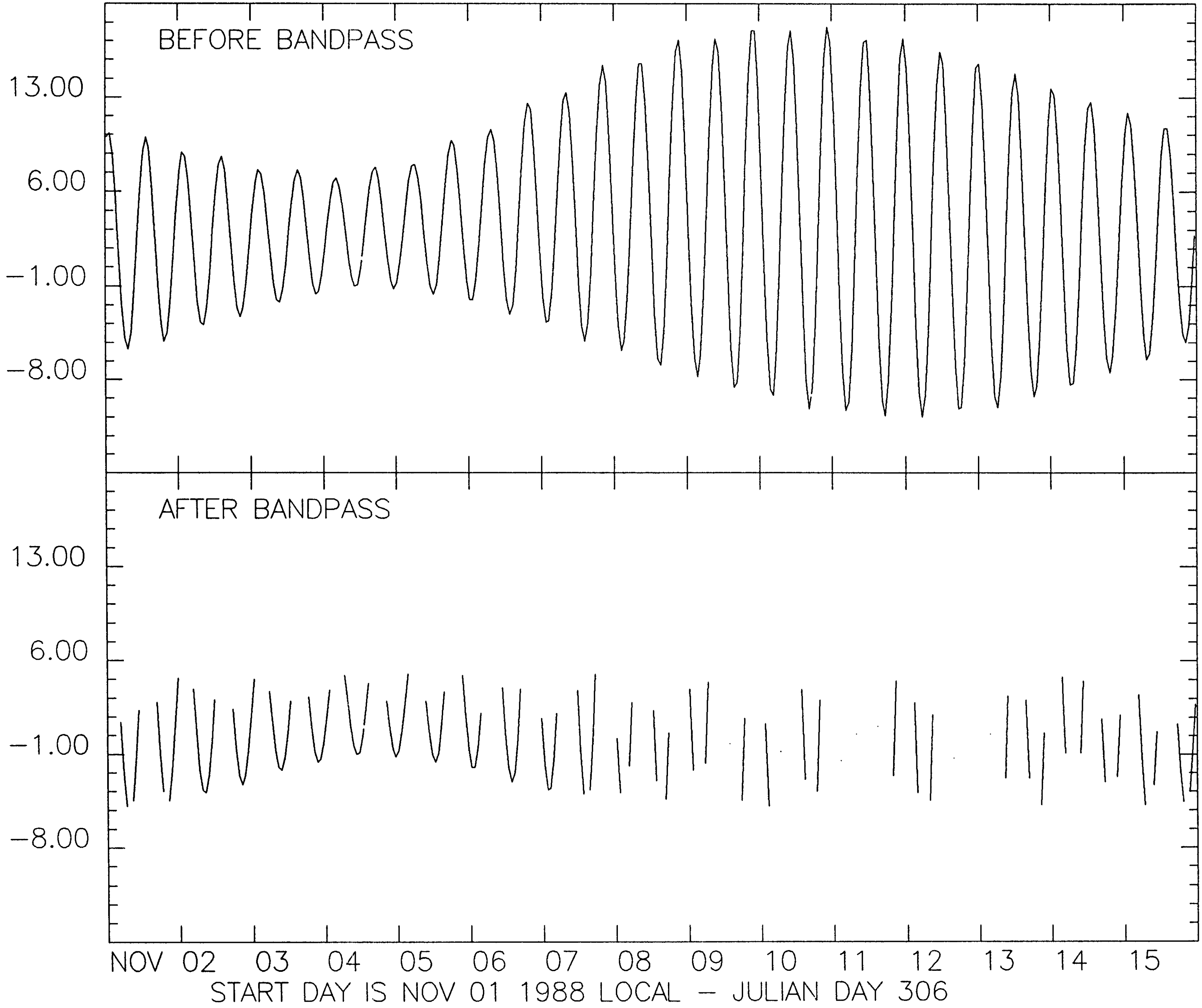
Three commands, **BANDPASS**, **OFFSET**, and **NOSPIKE**, are used to get rid of offsets and spurious data. Once these commands have been executed there is no way to "undo" their action other than to re-**FILL** the column and start the cleaning anew (remember, data in the column is changed by BOB, not data in the file). If the cleaning procedure requires more than a few commands, it is suggested that you periodically **COPY** the column to an unused column to avoid having to go back completely to the **FILL**. No commands currently exist for graphical editing of the data.

BANDPASS is used to get rid of data points above and below set points, allowing only the data in a band to remain in the column. BOB will prompt you for the highest and lowest valid data values. All data outside that range is eliminated (set to -998). For example:

```
COMMAND? STID TIDE
COMMAND? MEAS TIDE                    [in some cases, the STID and MEAS
                                         will be same]
COMMAND? B_DATE 01 NOV 88
COMMAND? DAYS 15
COMMAND? FILL E
   HEADER FOR COLUMN E
STATION=TIDE MEASMNT=TIDE.DAT BEGIN JL=306 BEGIN YR=88 #DAYS=0015 PTS/DAY=0024
VALID POINTS=        360 OUT OF       360  HIGH=        .18  LOW=     -.11
COMMAND? MUL 100 E                    [multiply the values in E by 100]
   HEADER FOR COLUMN E
STATION=TIDE MEASMNT=TIDE.DAT BEGIN JL=306 BEGIN YR=88 #DAYS=0015 PTS/DAY=0024
VALID POINTS=        360 OUT OF       360  HIGH=      18.30  LOW=   -10.80
COMMAND? SCALE E                      [scale the y axis for E]
HIGHEST VALID DATA POINT ? 20
LOWEST VALID DATA POINT ? -15
COMMAND? COPY E TO A                  [copy the contents and header of E to A]
   HEADER FOR COLUMN A
STATION=TIDE MEASMNT=TIDE.DAT BEGIN JL=306 BEGIN YR=88 #DAYS=0015 PTS/DAY=0024
VALID POINTS=        360 OUT OF       360  HIGH=      18.30  LOW=   -10.80
COMMAND? TITLE E BEFORE BANDPASS      [title the plot for E]
COMMAND? BANDPASS A                   [bandpass the data in A]
WHAT IS TO BE THE LOWEST VALID DATA VALUE ? -5
WHAT IS TO BE THE HIGHEST VALID DATA VALUE ? 5
   HEADER FOR COLUMN A
STATION=TIDE MEASMNT=TIDE.DAT BEGIN JL=306 BEGIN YR=88 #DAYS=0015 PTS/DAY=0024
VALID POINTS=        154 OUT OF       360  HIGH=       5.00  LOW=    -4.90
COMMAND? TITLE A AFTER BANDPASS       [title the plot for A]
COMMAND? PLOT 2 E A                   [divide the screen into 2 sections
                                         and plot columns E and A in them]
```

Note the differences before and after the bandpass as shown in HEADER A between the number of VALID POINTS, the HIGH, and the LO. Figure 2 shows the results graphically.

FIG. 2 BANDPASS EXAMPLE

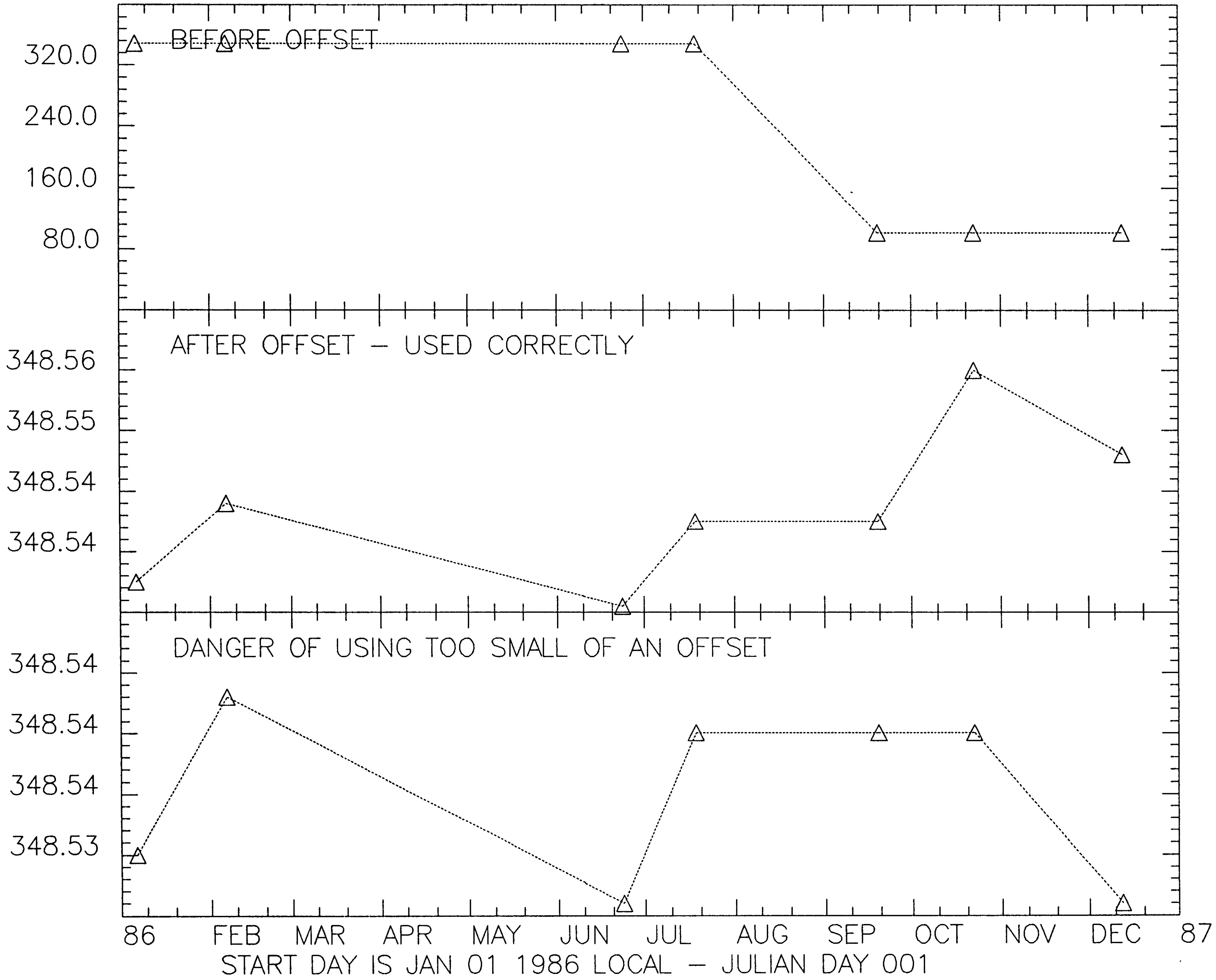


OFFSET is used to get rid of offsets in a column's data, typically the effects of rezeroing an instrument. When using offset, you will be prompted for the maximum allowable offset (i.e. the maximum offset that is ok). BOB then goes through the column, noting the difference between succesive data values. Whenever the difference exceeds the allowed, that data value. and all succeding values are offset by the difference, eliminating the offset. For example:

```
COMMAND? STID CURL
COMMAND? MEAS STPD
COMMAND? B_DATE 01 JAN 86
COMMAND? DAYS 365
COMMAND? FILL A
HOW MANY POINTS/DAY FOR THIS FILE ? 1
   HEADER FOR COLUMN A
STATION=CURL MEASMNT=STPD.DAT BEGIN JL=  1 BEGIN YR=86 #DAYS=0365 PTS/DAY=0001
VALID POINTS=        7  OUT OF        365  HIGH=    348.54  LOW=   101.61
COMMAND? TRIANGLE A                   [draw triangles at A's data points]
COMMAND? TITLE A BEFORE OFFSET
COMMAND? COPY A TO B                  [copy the contents and header of A to B]
   HEADER FOR COLUMN B
STATION=CURL MEASMNT=STPD.DAT BEGIN JL=  1 BEGIN YR=86 #DAYS=0365 PTS/DAY=0001
VALID POINTS=        7  OUT OF        365  HIGH=    348.54  LOW=   101.61
COMMAND? OFFSET B                     [get rid of the offset in B]
WHAT IS THE LARGEST POSSIBLE OFFSET ? 100
   HEADER FOR COLUMN B
STATION=CURL MEASMNT=STPD.DAT BEGIN JL=  1 BEGIN YR=86 #DAYS=0365 PTS/DAY=0001
VALID POINTS=        7  OUT OF        365  HIGH=    348.57  LOW=   348.53
COMMAND? TITLE B AFTER OFFSET - DONE CORRECTLY
COMMAND? COPY A TO C                  [copy the contents and header of A to B]
   HEADER FOR COLUMN C
STATION=CURL MEASMNT=STPD.DAT BEGIN JL=  1 BEGIN YR=86 #DAYS=0365 PTS/DAY=0001
VALID POINTS=        7  OUT OF        365  HIGH=    348.54  LOW=   101.61
COMMAND? OFFSET C                     [get rid of the offset in C, but this
                                         time use too small of a value]
WHAT IS THE LARGEST POSSIBLE OFFSET ? .017
   HEADER FOR COLUMN C
STATION=CURL MEASMNT=STPD.DAT BEGIN JL=  1 BEGIN YR=86 #DAYS=0365 PTS/DAY=0001
VALID POINTS=        7  OUT OF        365  HIGH=    348.54  LOW=   348.53
COMMAND? TITLE C DANGERS OF USING TOO SMALL OF AN OFFSET
COMMAND? PLOT 3 A B C                 [divide the screen into 3 sections
                                         and plot columns A, B, and C in them]
```

See figure 3. The middle plot shows the data as it should be. By using too small of a value for the maximum offset in the bottom plot (column C), you have changed the plot such that there is almost no net change for the time period plotted. Be careful when using **OFFSET**. If the threshold you choose is small enough to affect real data jumps, you can change the data even to the point of reversing a trend.

FIG. 3 OFFSET EXAMPLE

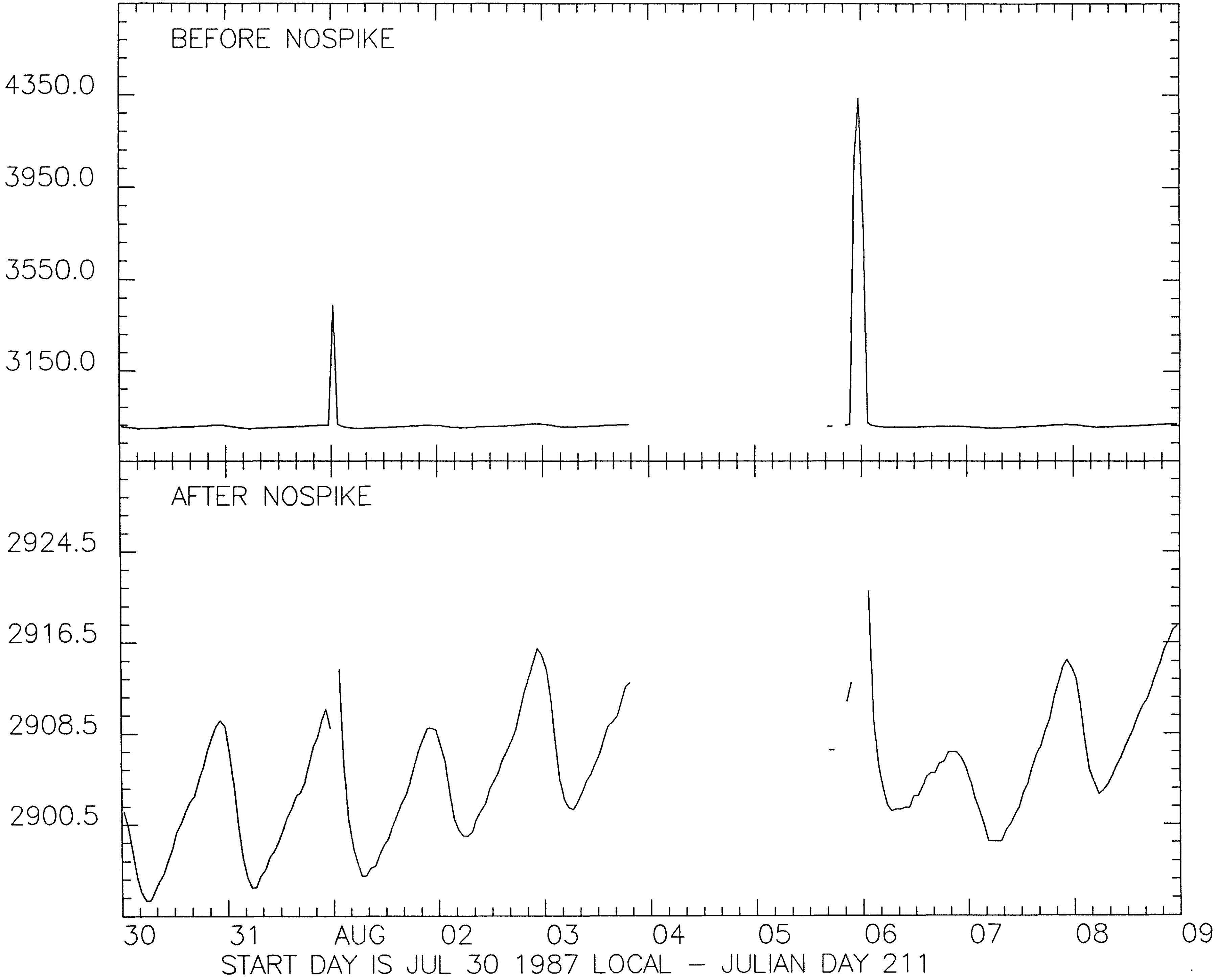


NOSPIKE is used to get rid of spurious spikes in a column's data. As with OFFSET, BOB will prompt you for the maximum allowable "spike" or difference between succeding data values. When BOB detects that a data point exceeds its predecessor by more than the allowable, that data point is eliminated (set to -998). For example:

```
COMMAND? STID MOER
COMMAND? MEAS RADL
COMMAND? B_DATE 30 JUL 87
COMMAND? DAYS 10
COMMAND? FILL A
COMMAND? COPY A TO B
   HEADER FOR COLUMN B
STATION=MOER MEASMNT=RADL.DAT BEGIN JL=211 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=     194  OUT OF        240  HIGH=   4335.67  LOW=    2893.83
COMMAND? TITLE A BEFORE NOSPIKE
COMMAND? TITLE B AFTER NOSPIKE
COMMAND? NOSPIKE B                    [get rid of the data spikes in B]
WHAT IS THE LARGEST POSSIBLE OFFSET ? 100
   HEADER FOR COLUMN B
STATION=MOER MEASMNT=RADL.DAT BEGIN JL=211 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=     190  OUT OF        240  HIGH=   2921.00  LOW=    2893.83
COMMAND? PLOT 2 A B                   [divide the screen into 2 sections
                                         and plot columns A and B in them]
```

Figure 4 shows its use. Note that if your first data point in the array is a spike, NOSPIKE will view it as the valid data point, and the rest of the data as the spikes to be removed.

FIG. 4 NOSPIKE EXAMPLE



Arithmetic Manipulation of the Columns

The commands **ADD**, **SUB**, **MUL**, **and DIV** allow you to mathematically manipulate the data in the columns, either with constants or data from other columns. The primary use of these commands is to convert data that may be stored in the files in units of volts to the actual engineering unit such as degrees celsius, meters, microradians, etc. For example, the formula to convert the MOE Rock TEMPerature data stored in the files as telemetry counts to degrees Celsius is as follows:

CELSIUS=(((TELEMETRY COUNTS)-4097)*.122)-273

NOTE: Conversion information such as this should be displayed when **MEASURES** is executed. Complain to the appropriate person if it is not.

The following commands convert the data to degrees Celsius.

```
COMMAND? STID MOER
COMMAND? MEAS TEMP
COMMAND? B_DATE 01 NOV 87
COMMAND? DAYS 10
COMMAND? FILL B
   HEADER FOR COLUMN B
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=      240  OUT OF         240  HIGH=     6328.17  LOW=     6303.00
COMMAND? SUB 4097 B                    [subtract 4097 from the values in B]
   HEADER FOR COLUMN B
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=      240  OUT OF         240  HIGH=     2231.17  LOW=     2206.00
COMMAND? MUL .122 B                    [multiply the values in B by 0.122]
   HEADER FOR COLUMN B
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=      240  OUT OF         240  HIGH=     272.20   LOW=     269.13
COMMAND? SUB 273 B                     [subtract 273 from the values in B
                                         which leaves B with degrees Celsius]
   HEADER FOR COLUMN B
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=      240  OUT OF         240  HIGH=       -.80  LOW=    -3.87
```

By comparing the HIGH and LOW values before and after the commands, you can see how the in the column is changed. That is why these values are displayed after every change to the column's data. NOTE the commands do not affect the number of valid points.

You can also **SUB**, **ADD**, **MUL** and **DIV** the contents of one column with another (or itself). This capability is useful if you need to subtract the data of one column from that in another, such as would be done in differencing the data between two ocean tide gauges to determine relative crustal tilt between the two stations.

```
COMMAND? COPY B TO E
   HEADER FOR COLUMN E
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=      240  OUT OF        240  HIGH=     6328.17  LOW=     6303.00
COMMAND? SUB B E                       [subtract the values in column B from
                                         those in column E]
   HEADER FOR COLUMN E
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=      240  OUT OF        240  HIGH=        0.00  LOW=        0.00
```

Since column B had the same values as column E, all the subtractions resulted in zero. Column B was unchanged.

By multiplying a column by itself:

```
COMMAND? MUL B B
```

you can square all the values in the column.

Two other commands bear special note, **AVG**, and **ZEROLOW**. Plotting 200 days of 10 minute data can take some time. To shorten this time use the **AVG** command to average the 10 minute data to hourly or even daily data. In the following example, column B starts out as hourly data (rate= 24 data points/day). Succesive **AVG**s average the data to finally get one data point/day.

```
COMMAND? SHOHDR B
   HEADER FOR COLUMN B
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=      240  OUT OF        240  HIGH=       -.80  LOW=     -3.87
COMMAND? AVG 4 B                       [average every 4 data points in B
                                         This leaves B with to 4 hourly data]
   HEADER FOR COLUMN B
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0006
VALID POINTS=       60  OUT OF         60  HIGH=       -.81  LOW=     -3.82
COMMAND? AVG 6 B                       [average every 6 data points in B
                                         This leaves B with daily averages]
   HEADER FOR COLUMN B
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0001
VALID POINTS=       10  OUT OF         10  HIGH=       -.97  LOW=     -3.69
```

Of course the intermediate steps could have been skipped by simply issuing the command

```
COMMAND? AVG 24 B                      [average every 24 data points in B]
```

in the beginning.

ZEROLOW sets the lowest data point in a column to zero, adjusting all the other points up or down accordingly. It is the equivalent of issuing a SUB command with the lowest data value as its argument. This command is useful when the user is interested in relative changes, not absolute values. The following shows the use of the **ZEROLOW** command.

```
COMMAND? FILL C
   HEADER FOR COLUMN C
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=     240  OUT OF       240  HIGH=    6328.17  LOW=    6303.00
COMMAND? ZEROLOW C                     [adjust the data in C so that the low-
                                         est data equals 0.0]
                                       [In the example, this would be the
                                         same as a SUB 6303.00 C]
   HEADER FOR COLUMN C
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=305 BEGIN YR=87 #DAYS=0010 PTS/DAY=0024
VALID POINTS=     240  OUT OF       240  HIGH=      25.17  LOW=       0.00
```

Other arithmetic commands are listed in The BOB Command Set section of this manual. Some are specialized, such as **EVENTS**, used to determine events in seismic amplitude data. Others have greater application such as **PHASE**, used to shift data in a column up or down relative to the time. It is suggested that you at least glance at the list of arithmetic commands to become aware of all of them.

Setting the Plot Parameters

The data contained in the columns can be plotted against time using the PLOT or PRINT command. For instance, the commands

```
COMMAND? STID MOER
COMMAND? MEAS TEMP
COMMAND? B_DATE 01 NOV 87
COMMAND? DAYS 10
COMMAND? FILL A
COMMAND? PLOT 1 A                    [plot A on the whole screen]
```

will produce a plot (figure 1) on the screen of the data from column A. By default, it will be auto-scaled, no label for the Y axis will be given, the title of the graph will be the STID and MEAS of the column and consecutive valid data points will be connected by a line.

Though figure 1 might be good enough to convey the information to people familiar with the station, others would have difficulty deciding what exactly was being measured. All the default parameters were used to produce figure 1. By changing the defaults, the plot can become more readable.

The plotting parameters for all the columns are stored in the plot header. It can be viewed by issuing the command PLOTHDR.

```
COMMAND? PLOTHDR                          [display the plot parameters]
                 COLUMN A        COLUMN B        COLUMN C        COLUMN D
TITLE       MOER TEMP
LABEL
TIME ZONE          LOC             LOC             LOC             LOC
SCALE HI         -998.000        -998.000        -998.000        -998.000
SCALE LO         -998.000        -998.000        -998.000        -998.000
STYLE           LINES           LINES           LINES           LINES
DASHED LINES NO DASHES       NO DASHES       NO DASHES       NO DASHES

                 COLUMN E        COLUMN F        COLUMN G        COLUMN H
TITLE
LABEL
TIME ZONE          LOC             LOC             LOC             LOC
SCALE HI         -998.000        -998.000        -998.000        -998.000
SCALE LO         -998.000        -998.000        -998.000        -998.000
STYLE           LINES           LINES           LINES           LINES
DASHED LINES NO DASHES       NO DASHES       NO DASHES       NO DASHES
```

A -998.000 for both the SCALE HI and SCALE LO indicates that the autoscale will be used.

The plot parameters for a particular column are reset every time that column is FILLed.

By using the TITLE and LABEL commands we can clear up some of the confusion as to what is being measured in figure 1. These commands allow you 45 characters with which to title the plot and 15 characters with which to label the Y axis. For example:

```
COMMAND? PLOTHDR
              COLUMN A        COLUMN B        COLUMN C       COLUMN D
TITLE     MOER TEMP
LABEL
TIME ZONE        LOC             LOC             LOC            LOC
SCALE HI         -998.000        -998.000        -998.000       -998.000
SCALE LO         -998.000        -998.000        -998.000       -998.000
STYLE         LINES           LINES           LINES          LINES
DASHED LINES NO DASHES       NO DASHES       NO DASHES      NO DASHES
```

[plot header for columns E-H would be shown here also]

```
COMMAND? COPY A TO B                  [copy the contents and header of A to B]
COMMAND? TITLE B MOE ROCK TEMPERATURE  [give B a title]
COMMAND? LABEL B CELSIUS              [ label the y axis for B]
COMMAND? PLOTHDR
              COLUMN A        COLUMN B        COLUMN C       COLUMN D
TITLE     MOER TEMP       MOE ROCK TEMPER
LABEL                     CELSUIS
TIME ZONE        LOC             LOC             LOC            LOC
SCALE HI         -998.000        -998.000        -998.000       -998.000
SCALE LO         -998.000        -998.000        -998.000       -998.000
STYLE         LINES           LINES           LINES          LINES
DASHED LINES NO DASHES       NO DASHES       NO DASHES      NO DASHES
```

[plot header for columns E-H would be shown here also]

The command **TOPTITLE** puts a larger character title at the top of the graph. The titles FIG x on all the example graphs in this manual were done using the **TOPTITLE** command. For our current example we would:

```
COMMAND? TOPTITLE FIG. 5
```

To delete the top title type:

```
COMMAND? TOPTITLE
```

with no argument. Deleting a **LABEL** or a **TITLE** is done in a similar fashion.

Though the auto-scaling done in figure 1 allows all the information to be shown, you may wish to override the auto-scale and set them yourself. By using the **SCALE** command you can do this. After telling BOB we wish to scale column A, BOB prompts you for the high and low values for the Y axis.

```
COMMAND? SCALE B                    [set the scale for the yaxis for B]
HIGHEST VALID DATA POINT ? 0
LOWEST VALID DATA POINT ? -5
COMMAND? PLOTHDR
              COLUMN A         COLUMN B         COLUMN C        COLUMN D
TITLE     MOER TEMP        MOE ROCK TEMPER
LABEL                      CELSIUS
TIME ZONE        LOC              LOC              LOC              LOC
SCALE HI         -998.000          0.000           -998.000         -998.000
SCALE LO         -998.000         -5.000           -998.000         -998.000
STYLE         LINES            LINES            LINES           LINES
DASHED LINES NO DASHES        NO DASHES        NO DASHES       NO DASHES

             [plot header for column E-H would be shown here also]
```

Note that this does not affect the data in the column. It only defines the range of the Y axis while plotting. Now if we issue the command

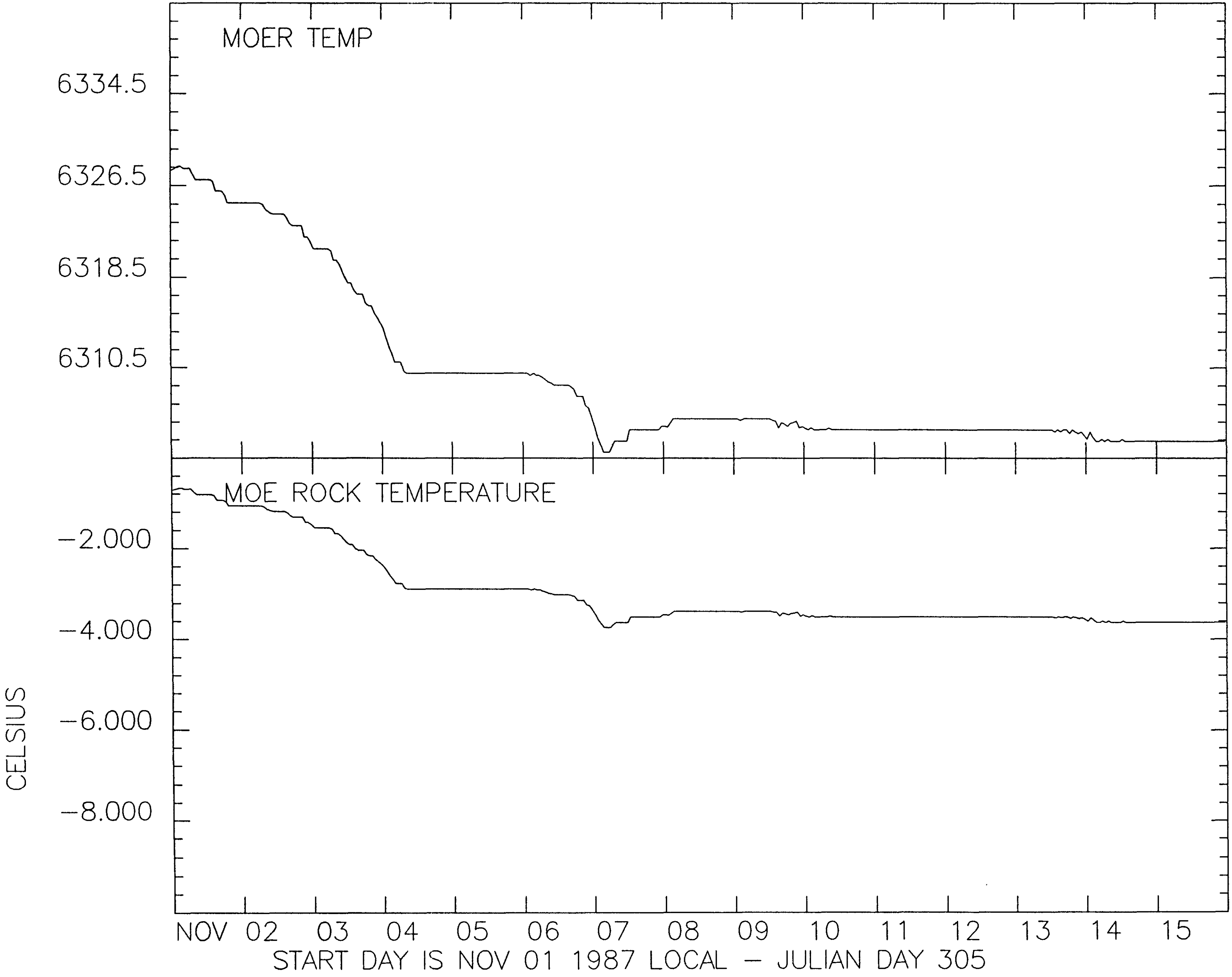
```
COMMAND? PLOT 2 A B                 [divide the screen into 2 sections and
                                      put A in the top, B in the bottom]
```

figure 5 is produced.

Not all data is best shown with a line plot such as in figure 5. The commands **CIRCLE**, **SQUARE**, **TRIANGLE**, **DIAMOND**, and **BAR** change the the style of the plot. BAR produces a bar graph, while the others draw their respective figures at each valid data point. The commands

```
COMMAND? STID COSP
COMMAND? MEAS SO2_
COMMAND? BJL 1
COMMAND? BYR 86
COMMAND? DAYS 365
COMMAND? FILL A
HOW MANY POINTS/DAY DO YOU WANT FOR THIS FILL ? 1
COMMAND? SCALE A
HIGHEST VALID DATA POINT ? 750
LOWEST VALID DATA POINT ? -250
COMMAND? COPY A TO B
COMMAND? COPY B TO C
COMMAND? TITLE A SQUARE, NODASHES
COMMAND? TITLE B TRIANGLE, DASHES
COMMAND? TITLE C BAR
COMMAND? SQUARE A                   [draw squares at A's valid data points]
COMMAND? NODASHES A                 [do not draw dashed lines between the
                                      the squares of A]
```

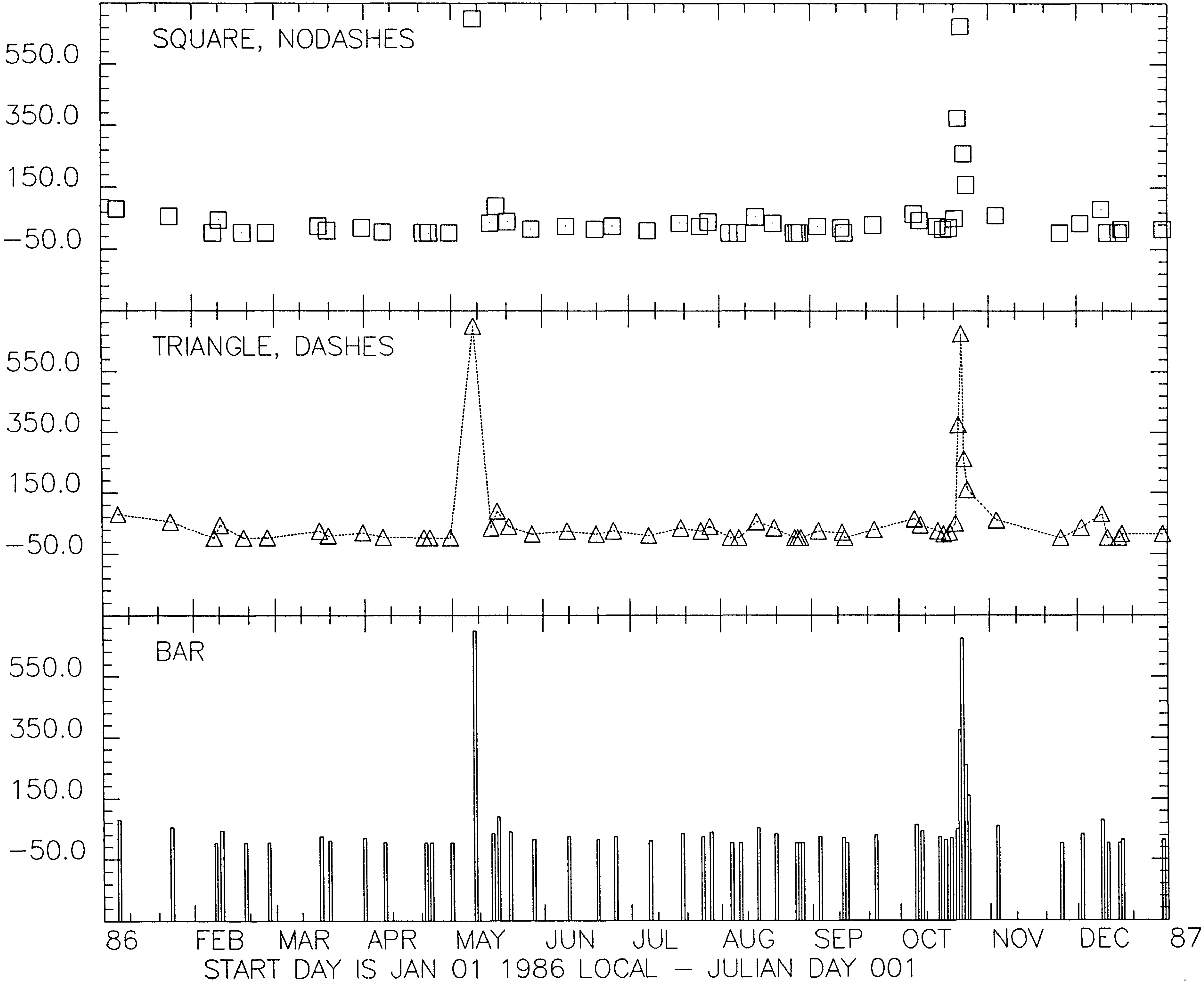
FIG. 5


MOER TEMP
6334.5
6326.5
6318.5
6310.5
MOE ROCK TEMPERATURE
−2.000
−4.000
−6.000
−8.000
CELSIUS
NOV 02 03 04 05 06 07 08 09 10 11 12 13 14 15
START DAY IS NOV 01 1987 LOCAL − JULIAN DAY 305

```
COMMAND? TRIANGLE B                   [draw triangles at B's valid points]
COMMAND? BAR C                        [plot C as a bar graph]
COMMAND? PLOTHDR
               COLUMN A         COLUMN B         COLUMN C         COLUMN D
TITLE     SQUARE, NODASHE   TRIANGLE, DASHE   BAR
LABEL
TIME ZONE         LOC              LOC              LOC              LOC
SCALE HI          750.000          750.000          750.000         -998.000
SCALE LO         -250.000         -250.000         -250.000         -998.000
STYLE           SQUARE           TRIANGLE             BAR         LINES
DASHED LINES NO DASHES             DASHES          DASHES      NO DASHES

           [plot header for columns E-H would be shown here also]

COMMAND? PLOT 3 A B C                 [divide the screen into 3 sections and
                                        put A in the top, B in the middle,
                                        and C at the bottom]
```

produce figure 6. The command **NODASHES** (and its inverse **DASHES**) determines if dashed lines are to be drawn between the data points.

FIG. 6 SYMBOLS



Plotting the Data

The integer after the plot command determines into how many sections the screen will be divided. The previous example was plotted with the command

```
COMMAND? PLOT 3 A B C
```

Figure 6 shows the resulting plot, with A on top, B in the middle, and C at the bottom. If you had wanted B on top and A in the middle you would have used the command

```
COMMAND? PLOT 3 B A C              [this time B goes on top, A in the
                                     middle and C at the bottom]
```

if you had only wanted A plotted in the top third of the screen, leaving the middle blank, and C in the bottom third, you could use the command

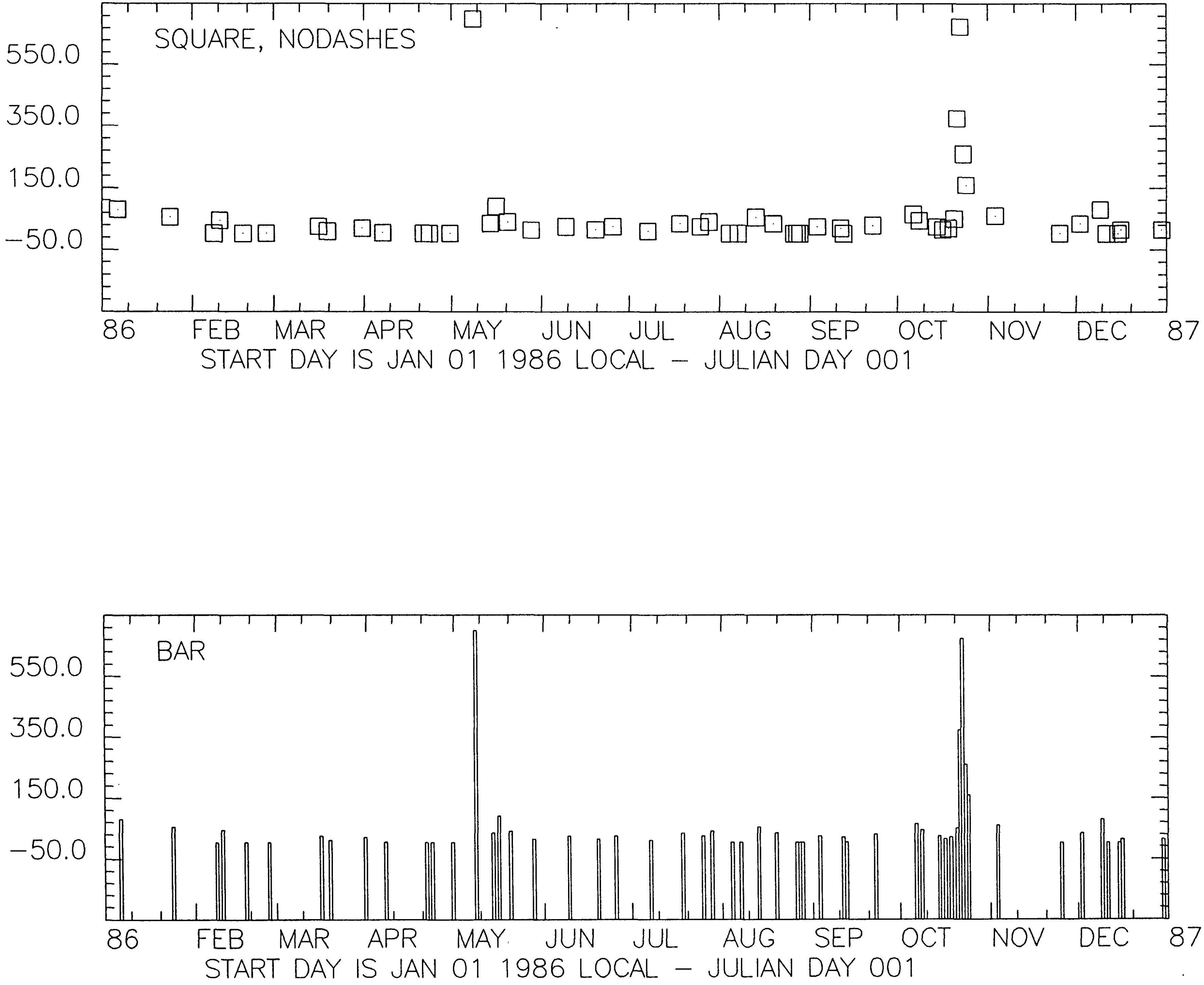
```
COMMAND? PLOT 3 A X C              [divide the screen into 3 sections,
                                     put A in the top, leave the middle
                                     blank, and plot C at the bottom]
```

where X can be any character except that of an array (A-H). Figure 7 is the result.

You can also superimpose the data from two columns on top of each other, by using '+' between columns you want plotted in the same section of the screen. The title, label and Y axis are taken from the first column listed. The other columns are plotted the same as they would be if plotted separately, only no title, label or scale is produced. Example:

```
COMMAND? STID MOER
COMMAND? MEAS RADL
COMMAND? B_DATE 01 NOV 87
COMMAND? DAYS 15
COMMAND? FILL A
COMMAND? MEAS TANG
COMMAND? FILL B
COMMAND? ZEROLOW A
COMMAND? ZEROLOW B
COMMAND? ADD 110 A
COMMAND? SCALE A
HIGHEST VALID DATA POINT ? 200
LOWEST VALID DATA POINT ? 0
```

FIG. 7


SQUARE, NODASHES
550.0
350.0
150.0
−50.0
86
FEB
MAR
APR
MAY
JUN
JUL
AUG
SEP
OCT
NOV
DEC
87
START DAY IS JAN 01 1986 LOCAL − JULIAN DAY 001
BAR
550.0
350.0
150.0
−50.0
86
FEB
MAR
APR
MAY
JUN
JUL
AUG
SEP
OCT
NOV
DEC
87
START DAY IS JAN 01 1986 LOCAL − JULIAN DAY 001

```
COMMAND? SCALE B
HIGHEST VALID DATA POINT ? 200
LOWEST VALID DATA POINT ? 0
COMMAND? COPY A TO C                 [put A into C simply so we can title the
                                        three plots correctly]
COMMAND? TITLE C RADIAL (TOP) AND TANGENTIAL (BOT)
COMMAND? PLOT 3 A C+B B              [the middle plot will have B super-
                                       imposed on C.  since C was mentioned
                                        first, the plot's title and label
                                        will come from it.]
```

would produce figure 8.

If you wish to send the graph to the printer instead of the screen, substitute **PRINT** for **PLOT**. To send the above example to the printer instead of the screen type:

```
COMMAND? PRINT 3 A C+B B
```

If you wish to put the graph into a file that can later be sent to an HP LaserJet printer substitute **LASER** for **PLOT**. BOB will prompt you for a file name.

```
COMMAND? LASER 3 A A+B B
FILE NAME FOR LASER OUTPUT ? ERASEME.LZR
```

This is useful if you do not have an HP LaserJet attached to your computer, but do have access to one. Use **LASER** to generate the graphics file, copy it to a diskette, and take the diskette to the PC attached to the laser printer. To print the file, you must use the program FPLOT which is supplied on Geocomp's Graphics Utilities diskette (Geocomp's Graphics Utilities are required to run BOB so they have to be around somewhere). When BOB is installed it is suggested that this program and the laser driver's file LASER.DRV be installed on the computer with the laser printer. If it is not complain to the appropriate person. To print the file created above, ERASEME.LZR, you will type something like the following.

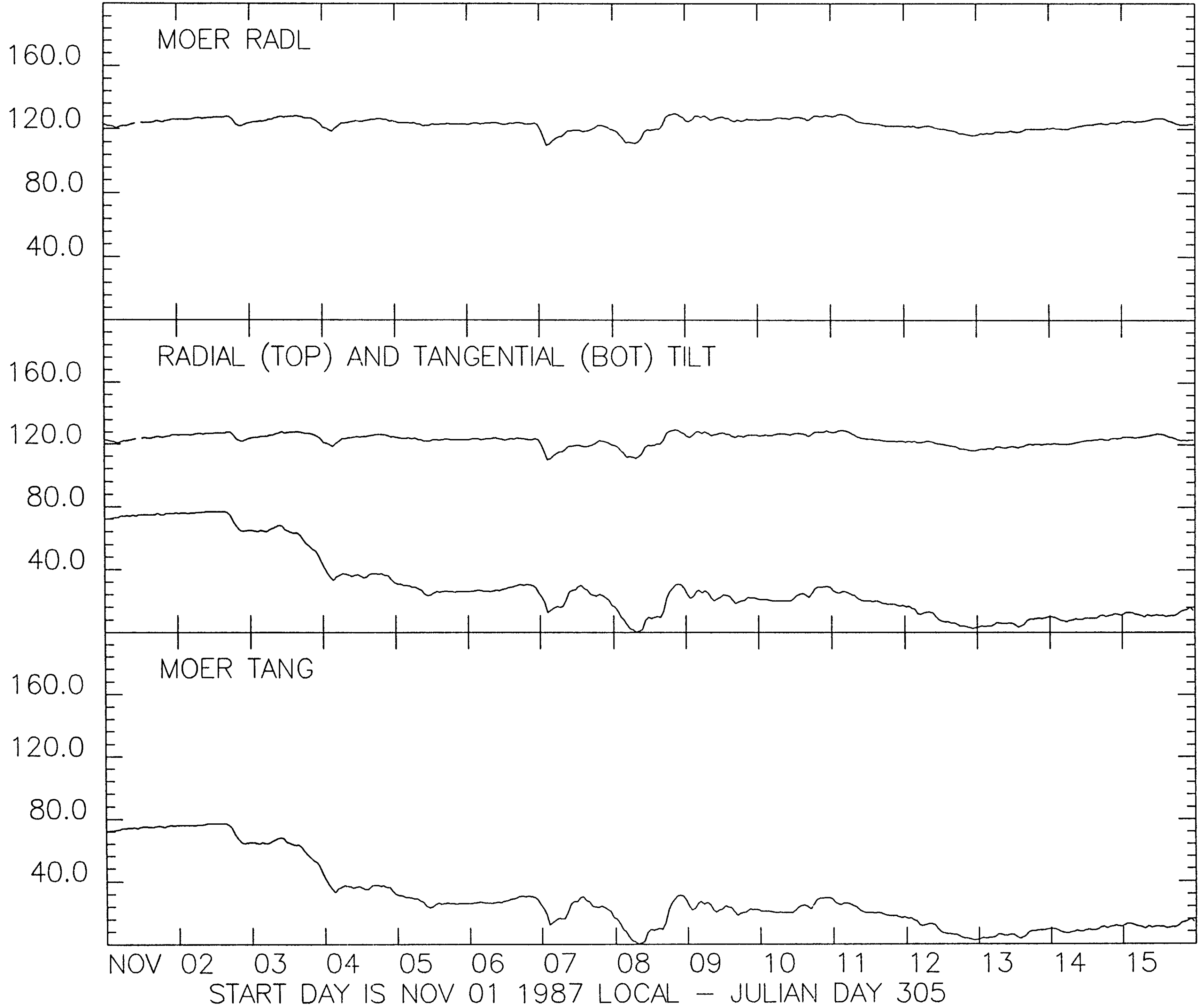
```
FPLOT A: ERASEME.LZR LASER.DRV COM1:
```

The actual command may vary from site to site as set-ups vary. The use of FPLOT is explained in Geocomp's Graphics Utilties manual under FPLOT UTILITY PROGRAM.

Plotting one column against another

BOB also has the ability to generate XY plots, plotting data from one column against data from another. NOTE that THE TWO COLUMNS MUST HAVE THE SAME TIME PERIOD AND NUMBER OF

FIG. 8 SUPERIMPOSING GRAPHS



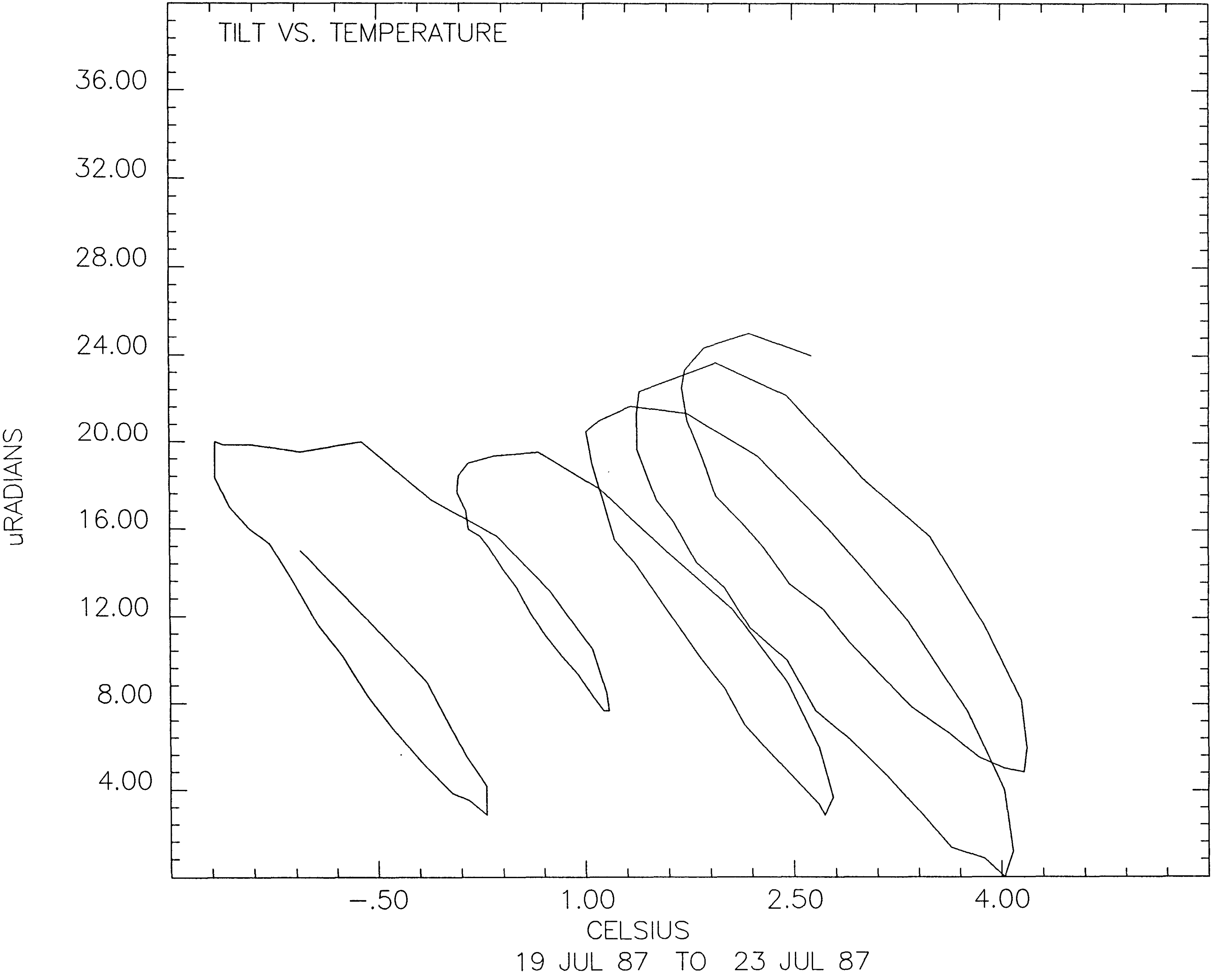
POINTS/DAY. Also you cannot do more than one plot on a screen. Example:

```
COMMAND? STID MOER
COMMAND? BJL 200
COMMAND? BYR 87
COMMAND? DAYS 5
COMMAND? MEAS RADL                      [we'll plot radial tilt vs. temperature]
COMMAND? FILL A
COMMAND? MEAS TEMP
COMMAND? FILL B
COMMAND? ZEROLOW A
COMMAND? SUB 4096 B
COMMAND? MUL .122 B
COMMAND? SUB 273 B
COMMAND? LABEL B CELSIUS
COMMAND? LABEL A uRADIANS
COMMAND? TITLE A TILT VS. TEMPERATURE  [the title comes from the 2nd column
                                            listed in the XYPLOT command]
COMMAND? XYPLOT B A
```

See Figure 9.

Substitute XYPRINT for XYPLOT to send the graph to the printer.

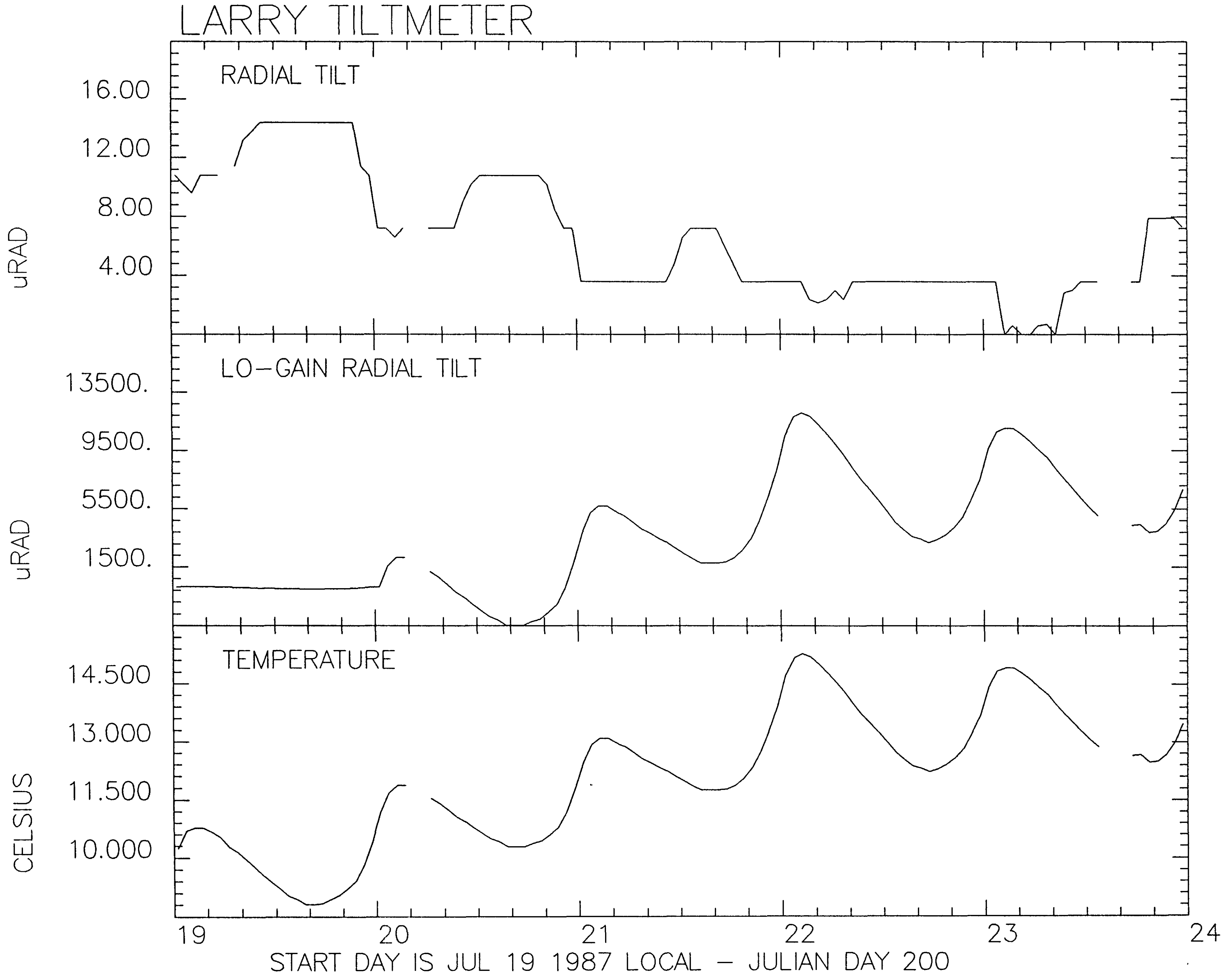
FIG. 9 XY PLOTS



Putting it All Togther

The following is taken from the macro LARR.DMO. It is an example of how to put all the commands together to produce the desired graph. Figure 10 shows the resulting plot.

```
COMMAND? B_DATE 01 JUN 87                [set the beginning day]
COMMAND? DAYS 30                         [do the whole month]
COMMAND? STID LARR                       [station larry tiltmeter]
COMMAND? TOPTITLE FIG. 10 - LARRY TILTMETER  [set the top title]
COMMAND? MEAS RADL                       [radial tilt measurement]
COMMAND? FILL A                          [fill column A]
COMMAND? OFFSET A                        [get rid of any offsets]
WHAT IS THE LARGEST POSSIBLE OFFSET ? 5
COMMAND? ZEROLOW A                       [set the lowest point to zero]
COMMAND? MUL 3.6 A                       [scale it to microradians]
COMMAND? LABEL A uRADS                   [the y axis label]
COMMAND? TITLE A RADIAL TILT             [the title]

COMMAND? MEAS DOLR                       [now get the dollar tilt measurement]
COMMAND? FILL B                          [put it into B]
COMMAND? ZEROLOW B                       [set the lowest point to zero]
COMMAND? MUL 355 B                       [scale it to microradians]
COMMAND? TITLE B LO-GAIN RADIAL TILT      [give B its title]
COMMAND? LABEL B uRAD                    [and its y axis label]

COMMAND? MEAS TEMP                       [now for the temperature]
COMMAND? FILL C                          [put the temperature into C]
COMMAND? SUB 4097 C                      [scale the temp. to degrees
COMMAND? MUL .122 C                         celsius]
COMMAND? SUB 273 C
COMMAND? TITLE C TEMPERATURE             [title C]
COMMAND? LABEL C CELSIUS                 [label the y axis for C]
COMMAND? PLOT 3 A B C                    [plot the data]
```


LARRY TILTMETER
RADIAL TILT
16.00
12.00
8.00
4.00
uRAD
LO–GAIN RADIAL TILT
13500.
9500.
5500.
1500.
uRAD
TEMPERATURE
14.500
13.000
11.500
10.000
CELSIUS
19
20
21
22
23
24
START DAY IS JUL 19 1987 LOCAL – JULIAN DAY 200

Clever Uses of the Arithmetic Commands

Clever use of BOB's arithmetic commands allow you to do things that at first may not seem possible with BOB. As an example, the following shows how to put the decimal julian day representation of the time for the data in column A into column B.

```
COMMAND? STID LARR                    [first get the data for the entire
COMMAND? MEAS TEMP                       year]
COMMAND? BJL 1
COMMAND? BYR 87
COMMAND? DAYS 365
COMMAND? FILL A
   HEADER FOR COLUMN A
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8169   OUT OF       8760  HIGH=      6640.50  LOW=     6341.33
COMMAND? SUB 4097 A                   [convert to degrees celsius]
COMMAND? MUL .122 A
COMMAND? SUB 273 A
   HEADER FOR COLUMN A
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8169   OUT OF       8760  HIGH=        37.31  LOW=          .81
COMMAND? COPY A TO B                  [B will end up with the decimal day]
   HEADER FOR COLUMN B
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8169   OUT OF       8760  HIGH=        37.31  LOW=          .81
COMMAND? VALPTS 1 B                   [VALPTS replaces the data in B with
                                         the number of valid points every
                                         1 data point.  It is used here to
                                         fill in the time slots holding
                                         the missing data value (-998) with a
                                         valid value (0.0).  This ensures B
                                         has no missing data]
   HEADER FOR COLUMN B
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8760   OUT OF       8760  HIGH=         1.00  LOW=          .00
                                      [see that B now has 8760 valid points]
COMMAND? MUL 0 B                      [B will now be filled with zeroes]
   HEADER FOR COLUMN B
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8760   OUT OF       8760  HIGH=          .00  LOW=          .00
COMMAND? ADD 1 B                      [now set B to all ones]
   HEADER FOR COLUMN B
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8760   OUT OF       8760  HIGH=         1.00  LOW=         1.00
COMMAND? CUM B                        [do a running total of the data in B]
   HEADER FOR COLUMN B
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8760   OUT OF       8760  HIGH=      8760.00  LOW=         1.00
```

```
COMMAND? ADD 23 B                       [offset for the fact that the year
                                           doesn't start on 00 Jan.
   HEADER FOR COLUMN B
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8760   OUT OF       8760  HIGH=      8783.00  LOW=       24.00
                                        [The data in column B now starts at 1
                                           and counts up to 8783.  B now has a
                                           count of the number of hours (since
                                           PTS/DAY=0024) from 01 Jan 00:00 to
                                           31 Dec 23:30]
COMMAND? DIV 24 B                       [dividing by 24 converts the hours to
                                           decimal julian days]
   HEADER FOR COLUMN B
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8760   OUT OF       8760  HIGH=       365.96  LOW=        1.00
                                        [you are not done yet.  it would be
                                           best if wherever column A had missing
                                           data, column B also had missing data.
                                           No sense having a time with no data
                                           associated with it]
COMMAND? COPY A TO C                    [need to use column C for this]
   HEADER FOR COLUMN C
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8169   OUT OF       8760  HIGH=        37.31  LOW=         .81
COMMAND? MUL 0 C                        [set all the valid data values in C to
                                           zero]
   HEADER FOR COLUMN C
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8169   OUT OF       8760  HIGH=          .00  LOW=         .00
COMMAND? ADD C B                        [additions result in valid data
only                                                        when both columns
have valid points

                                           in the corresponding time slots.
                                           C only has valid data in the same
                                           spots as A.  By adding C to B we
                                           erase any data in B that doesn't
                                           have a corresponding valid data point
                                           in C (and by extension A).  And since
                                           all the data values in C are zero, no
                                           change is made to the corresponding
                                           data values remaining in B]
   HEADER FOR COLUMN B
STATION=LARR MEASMNT=TEMP.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0365 PTS/DAY=0024
VALID POINTS=     8169   OUT OF       8760  HIGH=       365.96  LOW=        1.00
```

You are left with the temperature values in column A and the corresponding decimal julian day values in column B.

Admittedly, this is a somewhat round-about method for obtaining the desired result. But the point is that BOB is capable of doing it with existing commands. Before determining that BOB is incapable of performing a certain function, think about what is it that you really want to do, and if through clever use of the existing commands it can be achieved.

Making and the Use of Macros

Macros are files consisting of a list of the BOB commands you would normally enter interactively. The macros are located in the BOB sub-directory MACRO. They are constructed either by a word processor (which has to have an ASCII output) or directly from BOB with the use of the **SAVECOMS** and **ENDSAVE** commands.

For instance, the commands (minus those defining the time period and the **PLOT** command) for the example in Putting It All Together (figure 11) are in the macro LARR.DMO. Now, instead of typing in all those commands to get fig.11, you need only type:

```
COMMAND? B_DATE 01 JUN 87          [define the time outside the macro]
COMMAND? DAYS 30
COMMAND? EXE LARR.DMO              [execute the macro LARR.DMO]
COMMAND? PLOT 3 A B C
```

BOB reads through LARR.DMO executing each line in turn as if you had entered the commands interactively.

One method for constructing a macro is with an ASCII word processor such as the QDOS II editor. The other way is to instruct BOB to save the commands you are entering to a file which you can **EXE**cute later. Use **SAVECOMS** and **ENDSAVE** for this.

```
COMMAND? SAVECOMS
 THE FILE NAME FOR THESE EXECUTABLE FILES SHOULD BE OF
 THE FORMAT NAMEEEEE.INT WHERE NAMEEEEE IS A NAME YOU
 GIVE THE FILE AND INT ARE YOUR INITIALS. USING YOUR
 INITIALS WILL PROTECT YOUR FILE FROM OTHERS.
FILENAME ? ERASEME.BOB
 ERASEME.BOB
```

[you would enter your commands here in the normal fashion]

```
COMMAND? STID LARR
COMMAND? MEAS TEMP
```

[and so on until you were done with the commands you wanted saved]

```
COMMAND? PLOT 3 A B C
```

[to stop saving the commands you issue the following command]

```
COMMAND? ENDSAVE
```

As before, to execute the above macro, type

```
COMMAND? EXE ERASEME.BOB
```

Examples of macros are in BOB's subdirectory MACRO. They can be listed while in BOB by issuing the command:

```
COMMAND? MACROS
```

The most obvious use of the macros is to make it easy to generate plots for users that have little knowledge of BOB. They can be given a "cookbook" on how to look at the data from the various stations without having to know a MEAS from a STID. They just would need to know the name of the macro, and how to execute it.

By combining macros for various stations into one (they can be nested) you can create a single macro that will cycle through and print or display on the screen data from stations of interest. The demonstration macro SHOW_M.DMO consists of

```
EXE LARR.DMO
PLOT 3 A B C
EXE MOER.DMO
PLOT 3 A B C
EXE COSP.DMO
PLOT 1 A
BYE
```

where each of the called macros plots a specific station. Now by executing the macro SHOW_M.DMO

```
COMMAND? B_DATE 01 OCT 87
COMMAND? E_DATE 31 OCT 87
COMMAND? EXE SHOW_M.DMO
```

you will get plots for the time period you requested (October 87) for the three stations Larry, Moe, and airborne Cospec.

If you had used the LAST 7 command for defining the time period and included it in the macro, every time SHOW_M.DMO was executed you would get plots of the last 7 days of data, up to and including the most recently received data. This is especially valuable in real-time monitoring.

BOB's menus (in BOB's subdirectory MENUS) also make use of macros for the plots. Using it is the preferred method for plotting numerous stations.

Macros can also be used to "save" plots. Instead of actually saving the plot in a file, you save the commands used to generate the plot. The macro will occupy less memory and be easier to change than the plot. Also, if later you want to plot the data on another style printer (say a laser printer), you won't have to start from scratch to get the plot in laser printer format.

Saving Data from the Columns

At some point you may want to save the data in a column that has been cleaned, i.e. one that has had the spurious data points and offsets removed. This can be done with either the **PUTOUT** or **EUR_FILE** commands.

The **PUTOUT** command writes the data back out to a BOB type file. This file must already exist (having been created with the program NEWFILE.EXE in BOB's subdirectory SUPPORT) and have the following suffix:

CLN for files containing cleaned data with the same number of data points/day as the raw data file

The data cannot be written to a file with the suffix DAT as these files contain the raw data and are to be inviolate. If absolutely necessary, the program BOB_EDIT in BOB's directory can be used to edit the DAT BOB files.

To use PUTOUT you must first change the suffix designation to that of the output file. The following example assumes that you have cleaned the data in column A and now wish to write it out to a CLN file.

```
COMMAND? STID MOER
COMMAND? MEAS TEMP
COMMAND? B_DATE 01 MAR 87
COMMAND? DAYS 31
COMMAND? FILL A
   HEADER FOR COLUMN A
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=060 BEGIN YR=87 #DAYS=0031 PTS/DAY=0024
VALID POINTS=       740 OUT OF        744  HIGH=   6295.00  LOW=  6294.83

    .                                      [assume you have cleaned the data
    .                                         here]
COMMAND? SHOHDR A
   HEADER FOR COLUMN A
STATION=MOER MEASMNT=TEMP.DAT BEGIN JL=060 BEGIN YR=87 #DAYS=0031 PTS/DAY=0024
VALID POINTS=       740 OUT OF        744  HIGH=   6295.00  LOW=  6294.83
COMMAND? FILE A CLN                        [set the file suffix to CLN. remember,
                                              you cannot PUTOUT to a DAT file]
COMMAND? SHOHDR A
   HEADER FOR COLUMN A
STATION=MOER MEASMNT=TEMP.CLN BEGIN JL=060 BEGIN YR=87 #DAYS=0031 PTS/DAY=0024
VALID POINTS=       740 OUT OF        744  HIGH=   6295.00  LOW=  6294.83
COMMAND? PUTOUT A                          [overwrite the current data in the
                                              CLN file with data from column A]
   COLUMN A PUTOUT
```

If you wish to access data from a CLN file, you must first change the file suffix in the fill header.

```
COMMAND? FILE CLN                          [set the file suffix in the fill header
                                              to CLN]
```

```
COMMAND? FILL B
   HEADER FOR COLUMN B
STATION=MOER MEASMNT=TEMP.CLN BEGIN JL=060 BEGIN YR=87 #DAYS=0031 PTS/DAY=0024
VALID POINTS=        360 OUT OF          360  HIGH=   6295.00  LOW=  6294.83
```

Column B now has data from the CLN file instead of the DAT file. NOTE that when doing a **FILL** with the file suffix set to CLN, BOB will look first for a CLN file. If none is present, but a DAT file exists, BOB will **FILL** the column with data from the DAT file. This enables you to access the old cleaned data and the most recent raw data (DAT) with a single **FILL**. The column would contain the best available data. With the file suffix set to DAT, however, BOB only looks for DAT files.

By changing a column header's STID and MEAS, you can re-direct the output to other files and/or directories. For example:

```
COMMAND? STID TIDE                      [get tide data]
COMMAND? MEAS TIDE
COMMAND? FILE DAT
COMMAND? B_DATE 01 JAN 87
COMMAND? DAYS 45
COMMAND? FILL B
   HEADER FOR COLUMN A
STATION=TIDE MEASMNT=TIDE.DAT BEGIN JL=001 BEGIN YR=87 #DAYS=0045 PTS/DAY=0024
VALID POINTS=       1080 OUT OF        1080  HIGH=        .18  LOW=      -.11
                                        [now change the STID and MEAS of
                                           column B]
COMMAND? STID B MOER
COMMAND? MEAS B TEMP
COMMAND? FILE B CLN
COMMAND? SHOHDR B                       [note the changes]
   HEADER FOR COLUMN A
STATION=MOER MEASMNT=TEMP.CLN BEGIN JL=001 BEGIN YR=87 #DAYS=0045 PTS/DAY=0024
VALID POINTS=       1080 OUT OF        1080  HIGH=        .18  LOW=      -.11
COMMAND? PUTOUT B
   COLUMN C PUTOUT                      [the CLN moe rock temperature file
                                           will now contain tide data]
```

Showing that you do have the ability to put tide data into a file normally reserved for temperature data underscores the importance of care in using **PUTOUT**. In the above example, not only is the wrong data set in the temperature file, but the previous data has been erased. It is because of these dangers that **PUTOUT** cannot write to a DAT file.

The ability to change a column's **STID** and **MEAS** is useful in cases such as calculating crustal tilt by differencing data from two ocean tide gauges. The difference is calculated by **SUB**tracting one column from another. By using **STID** and/or **MEAS**, the differences can be stored in a file separate from the original tide files.

You can also put data out to a sequential file using the EUR_FILE command. In this case BOB creates a file for the data and it is up to you to move the file to the proper directory.

```
COMMAND? EUR_FILE A C B              [write the data out to a sequential
                                       file, data from column A in the first
                                       column, C in the second, and B in the
                                       last]
FILE NAME? ERASEME.DAT
 DATA WRITTEN OUT TO FILE ERASEME.DAT
```

Note that time period and points/day must be identical for the columns listed in the **EUR_FILE** command.

Since European-date sequential data files can be edited with a word processor, the **EUR_FILE** command is not used that often for producing cleaned data sets. It is usually easier to simply edit the file with a word processor. The **EUR_FILE** command is most useful for producing an ASCII file of the data that can be used for input by other programs. Using the method outlined in Clever Uses of the Arthmetic Commands, you can produce a file that has one or more columns of data plus a column containing the decimal julian day representation of the time. This file is easily used as input to programs such as LOTUS or GRAPHER, that do not have the facility for a true time-scale X axis as does BOB, but do handle a decimal julian day representation of the time.

The BOB Command Set

The following pages contain a listing of BOB's commands. The first page is a one page listing of the commands, listed as to function. This page can be listed while in BOB with the command **COMMANDS**.

Following the one page listing of commands is an alphabetical listing of the commands with brief explanations as to their functions. The BOB command **HELP** accesses these explanations. For instance, if you wanted to know about the command **EVENTS** then

```
COMMAND? HELP EVENTS
```

would list the explanation of **EVENTS** (and any other commands having the characters **EVENTS** in them. Try a **HELP PLOT** to see how that works).

Not all the BOB commands have been discussed in the previous pages, so it is recommended that you glance through the full listing to see what is available.

```
                 the following is a list of current BOB commands

     I indicates integer    X indicates column (A-H)    R indicates real number
     S indicates a character string          E is either a number or a column
-----------------------------------------------------------------------------
                      getting data into the columns

B_DATE II SSS II   BJL I              BYR I              COPY X TO X
DAYS I             E_DATE II SSS II   FILE SSS           FILL X
LAST I             MEAS SSSS          MERGE X X          RATE IIII
SHOWHDR            STID SSSS          TONOW              VOLCANO
-----------------------------------------------------------------------------
             commands to display the data and the column headers

SHODATA X                      SHOHDR X                      PLOTHDR
-----------------------------------------------------------------------------
                      arithmetic commands

ADD E X            AUTOAVG X          AVG I X            CORRECT X X
CUM X              DIV E X            EVENTS X           GETRATES X
HIGHEST I X        LSTSQR X X         MOVAVG I X         MUL E X
PHASE I X          SETHI X            SUB E X            VALPTS I X
VECTOR X X         ZEROLOW X
-----------------------------------------------------------------------------
                      cleaning commands

BANDPASS X         NOSPIKE X          OFFSET X
-----------------------------------------------------------------------------
                      plot commands

BAR X              CIRCLE X           DASHES X           DIAMOND X
LASER I X+X X      LABEL X SSSSSSS    LINES X            NODASHES X
PLOT I X+X X       PRINT I X+X X      PLOTHDR            POINT  X
SCALE X            SCALEALL           SCROLL I X+X X     SQUARE A
TITLE X SSSSSS     TOPTITLE SSSSSS    TRIANGLE X         TZONE SSS
XYPLOT X X         XYPRINT X X
-----------------------------------------------------------------------------
                      putting data back out to the files

EUR_FILE X X X X   FILE X SSS         MAKEFILE X         MEAS X SSSS
PUTOUT X           STID X SSSS
-----------------------------------------------------------------------------
                      general information

GREG I I           HELP SSSSSSSS      JUL I SSS I        MACROS
MEASURES           STATIONS           COMMANDS
-----------------------------------------------------------------------------
                      building executable "macros"

ENDSAVE            EXE SSSS.SSS       SAVECOMS
-----------------------------------------------------------------------------
                      misc. commands

BYE                RESET              DOS SSSSSSSS       ***
-----------------------------------------------------------------------------
```

PROGRAM BOB

BOB is a FORTRAN program for accessing and manipulating the telemetered low frequency data. When using BOB the keyboard should be in uppercase mode.

The program is centered around 8 data columns - A,B,C,D,E,F,G, and H. each column has a header associated with it. The headers contain the station name, begining julian day, number of days of data, beginning year, data points per day, number of valid data points, the highest data point, and the lowest data point. Whenever a column is updated or changed the header is changed appropriately. There is also a working header that is used as the basis for the information needed to fill the columns with data from storage. It contains the same information as the column headers except (since it has no data points associated with it) the information relating to valid, high, and low data points.

**

The following is a list of the current commands available with BOB. All commands are uppercase and cannot have any spaces in front of the them. When using commands that use qualifiers, spaces and equal signs are considered equivalent. As long as the length of the command does not exceed 60 spaces, as many spaces or equal signs can be put between commands and their qualifiers.

```
******************************************************************************
*                  alphabetical listing of commands                          *
******************************************************************************
```

ADD E X2 ** E can be either a column (A - H) or a real number. X2 must be a column. If E is a real number then each valid data point in column X2 has E added to it. If E is a column then each time E and X2 have corresponding valid data points, their sum is stored in X2, otherwise a missing data point (-998.0) is inserted. E remains unchanged.

AUTOAVG X ** Determines the optimum data rate for plotting the data in column X and then averages the data appropriately. For plots of less than 10 days, 144 points/day is good, less than 30 days is 24 pts/day, less than 60 is 4 points, and anything greater than 60 days is averaged to 1 point/day.

AVG I X ** X is a column (A - H). I is an integer. AVG will go through column X and take the average of every I values. Missing data points are included in the window of I points. Note I must be an integeral divisor of the original data rate. The rate of the column and the rate of the working header are adjusted for the new rate (old rate/I).

B_DATE DA MON YR ** Sets the beginning julian day and year of the fill header to the date corresponding to day DA of month MON, year YR.

BANDPASS X ** X is a column (A thru H). BANDPASS will prompt you for the highest and lowest data values you want in column X. It will then go through the column throwing out any values not within the specified limits.

BAR X ** X is to be a bar graph when plotted.

BJL = I ** I must be an integer, 1 to 366. The beginning julian day in the working header is set to the value of I.

BYE ** Exit the program.

BYR = I ** I must be an integer. The beginning year in the working header is set to the value I.

CIRCLE X ** X will have circles drawn at valid data points when plotted.

COMMANDS ** Types the file command.txt, a short listing of all the BOB commands.

COPY X1 TO X2 ** COPY transfers the contents (including the header) of column X1 to column X2. This command is useful if you are manipulating X1's data with the possibility that you might make a mistake and have to start over with the original data. By having X1's original data copied to column X2 it is much quicker to get X1's original data back via the COPY command from X2 than with another FILL.

CORRECT X1 X2 ** Does a least square fit (just as in the command LSTSQR) and then uses those values and the data in column X1 to correct the data in column X2. X1 is left unchanged.

CUM X ** X is a column (A thru H). CUM goes through column X and replaces each valid number with the sum of it and all the previous numbers (essentially a running total).

DAYS = I ** I must be an integer. The number of days of data in the working header is set to I.

DASHES X ** When using LINE, TRIANGLE, CIRCLE, DIAMOND, or SQUARE, dashed lines wiil be drawn between valid points separted by missing data. See NODASHES.

DIAMOND X ** Diamonds will be drawn at vaild data points when plotting X.

DIV E X2 ** E can be either a column (A - H) or a real number. X2 must be a column. If E is a real number then each valid data point in column X2 is divided by E. If E is a column then each time E and X2 have corresponding valid data points, the result of X2/E is stored in X2, otherwise a missing data point is inserted (-998.0). E is unchanged.

DOS SSSSSSSSS ** DOS causes you to leave BOB and execute the DOS command SSSSSSSSSS (up to 30 characters) such as TIME and then return to where you were in BOB.

E_DATE DA MON YY ** Determines the number of days between the beginning date currently in the fill header and the date day DA of month MON of year YY. This number is put into the days slot of the fill header.

ENDSAVE ** Terminates SAVECOMS session. See SAVECOMS.

EUR_FILE X X X ** Outputs the data in columns X X X to a file with a time tag for each data point (like the serial ASCII files used by BOB to store data). Note that it will only work if the time periods and rates for all the columns are the same.

EVENTS X ** Prompts you for a threshold in determining an event in column X. BOB goes through column X and inserts a -998 (missing data) in all slots except where the value exceeds the previous value by the threshold.

EXE SSSSS.SSS ** The macro SSSSS.SSS will be executed.

EXECUTE SSSS.SSS ** Same as EXE.

FILE X SSS ** If X is ommitted, the working header's file type will be set to SSS (either DAT or CLN). If X is included the header associated with the column X will have its file type changed to SSS.

FILL X ** Fill column X with data according to the parameters of the working header.

GETRATES X ** GETRATES replaces the data in column X with the difference between each data point and the preceding data point, hence determining the rates of change. With points that have missing data between them, it takes the missing points into account by determining the rate for the entire period betwen the valid points.

GREG JUL YR ** JUL must be an integer from 1 to 366. YR is also an integer. GREG returns the month and day corresponding to julian day JUL in the year YR.

HELP SSSSSSSS ** If SSSSSSSS is left blank, a listing and explanation of all the BOB commands will be displayed. If SSSSSSSS is included, the list of commands will be searched and an explanation of command SSSSSSSS will be displayed.

HIGHEST I X ** Determines the highest value in each group of I values in column X.

JUL DA MON YR ** DA must be an integer from 1 to 31, MON a character string denoting the month, and YR must be an integer. JUL returns the julian day corresponding to month MON, day DA, year YR.

LABEL X SSSSSSSS ** LABEL sets the label of the scale axis for any plots of column X to the string (max. of 15 char.) SSSSSSSSSSSS. The label will be reset to a blank with any fill to that column.

LASER I X+X X ** Same as a plot command except outputs the plot to a file in HP laserjet format. The file can be printed at a later date.

LAST I ** LAST gets the current time and date from the computer clock (which may or may not be correct) and sets the header to access the last I days from the current day.

LINES X ** Indicates that when plotting column X, missing data points will not be drawn and lines will be drawn only between consecutive valid data points. This is the default value and any fill to a column will set that column to plot with LINES. Use DASHES for drawing dashed lines between valid data points separated by missing data.

LSTSQR X1 X2 ** LSTSQR takes the data in columns X1 and X2 (A - H) and performs a least squares fit (as in a HP-41) returning the values for A and B for the formula X1=(A*X2)+B. A value for the fit is also returned, with the value 1.0 being perfect correlation and 0.0 being a random fit.

MACROS ** Shows the files currently available that can be executed with the EXE command.

MAKEFILE X ** The data in column X is output to a file in a tabular format.

MEAS SSSS ** SSSS is a 4 character measurement id. The measurement id of the fill header is reassigned the name SSSS.

MEAS X SSSS ** SSSS is a 4 character measurement id. The measurement id of the header for column X is reassigned the name SSSS.

MEASURES ** Lists the file containing the information on all the measurements for a specific stid (in the file BOB_stid.TXT).

MERGE X1 X2 ** X1 and X2 are both columns. where data is missing in X2 but a valid point exists in X1, the valid point in X1 is inserted into X2. X1 is unchanged.

MOVAVG I X ** A moving average is computed for each I valid data points. Missing data points are not included.

MUL E X2 ** E can be either a real number or a column (A - H). X2 must be a column. If E is a real number then each valid data point in column X2 is multiplied by E. If E is an column then each time both E and X2 have valid data points the result of (E * X2) is placed in column X2, otherwise a missing data indicator (-998.0) is inserted. E remains unchanged.

NODASHES X ** No dashed lines will be drawn between succeding valid data points separated by missing data when plotting X using CIRCLE, SQUARE, DIAMOND, LINES, or TRIANGLE.

NOSPIKE X ** X is a column (A - H). NOSPIKE will prompt you for a spike limit. NOSPIKE will then go through the column and discard any data points that differ by more than the limit (+ or -) from the preceding valid data point. Note that if the initial data point is a screwy one, all the following points could be wiped out even if they are valid.

OFFSET X ** X is a column (A - H). OFFSET will prompt you for the maximum allowable offset. OFFSET will then go through the column and if any valid data point differs from its predecessor by more than the maximum allowed, it and all the following data points will be offset by the difference of the two numbers.

PHASE I X ** Shifts the data in column X I spots. Using a plot for the frame of reference, a negative I shifts the data to the left, a positive I to the right. -998s are inserted where the shift would leave missing data.

PLOT I X X X ** PLOT divides the screen into I sections (max. of 10) and then plots the columns X,X, and X starting from the top. If you don't want a certain sector to have a graph use an illegal column descriptor (anything other than A-H) if you want to plot only one graph, full screen, type in PLOT 1 X. To clear the screen and get back into

text mode type <cr> 2 or three times.
If you wish one or more graphs superimposed on each other insert a + between the graphs to be plot ed together.
example: PLOT 2 A+B C
A and B will be ploted on the top graph, C on the bottom.

PLOTHDR ** Prints the current plot parameters.

PRINT I X X X ** Same as PLOT except it automatically will make a hard copy of the plot. Also, one does not have to hit the <cr> twice to exit graphics mode.

POINT X ** When column X is plotted, no connecting lines are to be drawn between data points. Only the points will be drawn. A FILL X will set this back to its default value for X - LINES.

PUTOUT X ** Writes the data in column X (A - H) to the file as per the column header's file type and station id.

RATE I ** Sets the expected rate in the fill header to I.

RESET ** Puts BOB back into its original state.

SAVECOMS ** Prompts you for a filename and then writes each succeding command to that file until an ENDSAVE is typed. This file can then be executed using the EXECUTE command.

SCALE X ** SCALE allows you to set the vertical scale of the plot of column X it will prompt you for the values. If SCALE is not set, the plots will be adjusted to the highest and lowest data values in the column. after a FILL to the column X, the hi and lo are reset to the autoscale values (-998.0 for the both).

SCALEALL ** Same as SCALE except it sets the plot scales for all the columns at the same value.

SCROLL I X+X X ** Same as PLOT except the plot stays on the screen for only 10 seconds. Used for constantly scrolling through the most recent data.

SETHI X ** X is a column (A - H). SETHI will prompt you for what you want to be the highest number in the column. SETHI will go through the column adjusting the numbers accordingly.

SHODATA X ** Displays the data from column X on the screen.

SHOHDR ** Displays the fill header on the screen.

SHOHDR X ** Displays the header of column X on the screen.

SQUARE X ** Squares will be drawn at each valid data point when X is plotted.

STATIONS ** List the file containing all the information about the various stations for the chosen volcano.

STID SSSS ** SSSS is a 4 character station id. the station id of the working header is assigned name SSSS.

STID X SSSS ** SSSS is a 4 character station id. the station id of the header for column X is reassigned the name SSSS.

SUB E X2 ** E can be either a real number or a column (A - H). X2 be a column. If E is a real number then each valid data point of column X2 has E subtracted from it. If E is an column then each time both E and X2 have corresponding valid data points the result of X2-E is placed in X2, otherwise the missing data point indicator (-998.0) is inserted. E remains unchanged.

TITLE X SSSSSSSS ** Does exactly as LABEL but for the title of the plot. A FILL command sets the title to the stid and meas of the fill header.

TONOW ** TONOW looks at the working header's bjl and byr and then sets the days parameter such that all data from that beginning time up to and including the present day will be accessed.

TOPTITLE SSSSSS ** SSSSSS (up to 35 characters) will be printed in large letters at the top of the plot. To remove the TOPTITLE simply issue the command TOPTITLE without any argument.

TRIANGLE X ** Triangles will be drawn at each valid data point when X is plotted.

TZONE SSS ** SSS is the time zone that will appear on the plots. Note

```
                     that this only changes the label, nothing in the plot
                     or its time scale.
VALPTS I X        ** X is a column (A - H). I is an integer. VALPTS goes
                     through column X calculating the number of valid points
                     every interval of I points.
VECTOR X1 X2      ** X1 and X2 are columns (A - H). VECTOR calculates
                     azimuth (in degrees) and magnitude of the vector using
                     X1 as the x leg and X2 as the y leg. The initial point
                     is considered to have a magnitude of 0 and azimuth of 0.
VOLCANO           ** BOB will print a listing of the current volcano ids and
                     prompt you to input your choice.
XYPLOT X1 X2      ** Instead of a time-series plot, plots column X1 (X axis)
                     vs. X2 (Y axis)
XYPRINT X1 X2     ** Same as XYPLOT except output goes to the printer.
ZEROLOW X1        ** ZEROLOW takes the lowest value in the column X1 (A - H)
                     and subtracts it from each valid number in the column. This
                     command is useful with data that has large numbers but only
                     small relative changes (such as the high and low in the
                     column being 100456 and 100450).
***               ** Writes a blank line to the screen.  Useful for delimiting
                     separate sections in macros or batch files without
                     generating an illegal command message.
-----------------------------------------------------------------------------
```