DEPARTMENT OF THE INTERIOR

U. S. GEOLOGICAL SURVEY

Computer Programs for Analyzing Digital Seismic Data

Charles S. Mueller
U. S. Geological Survey, MS 977
345 Middlefield Rd.
Menlo Park, CA 94025

Open-File Report 90-35

1990

Introduction

This report describes and documents four computer programs that are used at the U.S. Geological Survey for routine analysis of digital seismic data.

PLOT1 - Read, modify, and plot a one-component seismogram.

PLOT3 - Read, modify, and plot a three-component seismogram.

PRS - Read, modify, and plot a pseudo-record-section of seismograms. Traces are not plotted as a function of distance, but are plotted sequentially in the order they are submitted ("pseudo"-record-section). They can be independently time shifted, and plotted at an equal scale ("true-amplitude" plotting) or an equal size on the page.

SPY - Read, modify, plot, window, and Fourier analyze a seismogram. Spectra can be modified (for example, smoothed or corrected for instrument response), and spectral ratios can be computed.

These programs are intended to be of general use as written, but readers are encouraged to modify them or use them as models for new programs - they are written in a modular style and are well documented. For users, transcriptions of terminal sessions are presented in the *Examples* section to demonstrate program use. For programmers, all FORTRAN source code for the four main programs and several dozen subroutines is presented in Appendix A and Appendix B, respectively.

These programs have been successfully compiled and run on a DEC VAX-750 computer at the USGS in Menlo Park, California. To modify them, the reader will need to understand several key concepts: the seismogram filename and data formats used at USGS (called VFBB filename and VFBB data in this report); subroutine RVFBB, which reads a VFBB data file, subroutine MVFBB, which modifies a seismogram time series created from a VFBB data file (filter or integrate, for example); and included file VFCOMMON.FI, which defines common blocks that carry information between main programs and subroutines. All the programs are built around these basic elements. USGS-specific features (discussed in more detail below) include a few bits of nonstandard FORTRAN, and calls to three CALCOMP-type plotting subroutines; otherwise, the programs are written in standard FORTRAN-77.

The VFBB filename and VFBB data formats have been used for processing, analyzing, and archiving digital seismic data at USGS since 1979 (Mueller *et al.*, 1988). Waveforms are stored, one evenly sampled component per file, in a compact block-binary format. A VFBB filename is a 13-character string constructed from the start-of-record time, component, and station-instrument name: characters 1-3 = day-of-year (001-366), characters 4-5 = hour (00-23), characters 6-7 = minute (00-59), character 8 = second code (A-T, where A = 0.000-2.999, B = 3.000-5.999, . . . T = 57.000-59.999), character 9 = component code (1-3 for acceleration, 4-6 for velocity, *etc.*),

character 10 = ".", and characters 11-13 = station-instrument name. A typical VFBB data file consists of 512-byte header and data blocks: one integer header block (256 integer*2), followed by one real header block (128 real*4), followed by one-or-more integer data blocks (256 integer*2). Variations are supported by the VFBB data format (extra header blocks, for example), but are seldom used. Subroutine RVFBB reads a VFBB data file and loads a seismogram time series array and common blocks VFGOT, VFCOMP, VFTRNS, and VFRCRD in VFCOMMON.FI. Readers who will use a different data format must replace RVFBB, but if the seismogram array and common blocks are properly loaded, everything else should work. Only a fraction of the headers are typically used; RVFBB looks for the following header information.

I010*	year ($0 < I010 < 200$ or $1900 < I010 < 2100$)
I011*	day ($1 \le I011 \le 366$)
I012*	hour ($0 \le I012 \le 23$)
I013*	minute ($0 \le I013 \le 59$)
I014*	second ($0 \le I014 \le 59$)
I015*	millisecond ($0 \le I015 \le 999$)
I031**	number of data blocks
I032*	index of last sample in last data block ($1 \le I032 \le 256$)
I041*	vertical azimuth of component (degree, $0 \le I041 \le 180$) (0=up, 180=down, 90=horizontal (see I042))
I042*	horizontal azimuth of component (degree, $0 \le I042 \le 359$)
I254*	ground-motion type (1=acc, 2=vel, etc.)
R005**	sample rate (sample/s)
R046*	digitizing constant (count/V)
R047	anti-alias filter corner frequency (Hz)
R048	anti-alias filter rolloff (pole)
R049	sensor frequency (Hz)
R050	sensor damping (fraction of critical)
R051*	sensor sensitivity (V/ground-motion-unit)
R052*	recorder gain (dB)

(Headers that are necessary to interpret data as a time series are denoted by **; RVFBB will complain if they are missing. Headers that are necessary to interpret data as a seismogram are denoted by *; in some cases RVFBB will supply defaults if they are missing.). A sample VFBB data file, converted to an ASCII card-image format, is presented in Appendix C.

Nonstandard FORTRAN

The programs and subroutines described in this report use several DEC extensions to FORTRAN-77. Using VAX FORTRAN with the /STANDARD and /WARNING=GENERAL options, the following cases of nonstandard FORTRAN were identified.

- nonstandard comment: !

- nonstandard statement: INTEGER*2
- nonstandard statement: INCLUDE - incorporate external source code into a program
- nonstandard FORMAT statement item: $ - suppress carriage return (for terminal I/O a typed response follows a prompt on the same line)
- nonstandard FORMAT statement item: Q - obtain the number of characters in the record that remain to be transferred during a read operation
- nonstandard lexical item: FORMAT(I<J>) - variable format expression
- nonstandard name: underscore character in symbolic name
- nonstandard name: symbolic name longer than six characters
- nonstandard keyword: READONLY in OPEN statement
- mixed numeric and character elements in common VFTMOD in VFCOMMON.FI

As a practical matter, some of these cases are worse than others, and the degree of incompatibility will vary from computer to computer. Repairs are left to the reader; they vary from completely straightforward (eliminating !-comments and underscores in symbolic names, for example) to slightly more complicated (a Q-format read can be replaced by a short subprogram that reads and parses input as a character string, for example).

Plotting

Graphics output is accomplished through calls to three pen-plotter emulation subroutines (CalComp compatible) that are supported on the USGS VAXes.

- subroutine PLOTS (0,0,0) - initialize plotting
- subroutine PLOT (X,Y,IPEN) - process coordinate data into vectors
  IPEN=+2, draw to position X,Y
  IPEN=+3, move to position X,Y
  IPEN=-999, end the current plot and reinitialize the system for a new plot
  IPEN=+999, end the current plot and terminate all plotting
- subroutine TEXT (X,Y,HEIGHT,CTEXT,ANGLE,NC) - plot character text
  X,Y = position of lower left-hand corner of the first character to be plotted
  HEIGHT = character height
  CTEXT = character string to be plotted
  ANGLE = text rotation in degrees clockwise from horizontal
  NC = number of characters to be plotted from CTEXT
  (TEXT was written by L. Baker of USGS; it converts character CTEXT to integer ITEXT and calls the standard subroutine SYMBOL (X,Y,HEIGHT,ITEXT,ANGLE,NC))

(In calls to PLOT and TEXT, coordinates are referred to the lower left-hand corner of the plot and all dimensions are inches .)

Examples

In this section, transcriptions of terminal sessions are presented to demonstrate the use of each

program. Several features are briefly described here; see the source code for further details.

- Setup
  Programs initialize various processing and plotting parameters, then allow the user to change default values at "Setup". The user can always return to "Setup" from the filename prompt.
- Non-VFBB filename
  Programs can process VFBB data files with non-VFBB filenames, in most cases program execution will be somewhat clumsier (for example, PLOT3 prompts for 2nd- and 3rd-component filenames, rather than construct them from 1st-component filename).
- Partial filename
  To save typing, programs try to make a new filename by replacing part of the current one, if it is a VFBB filename. The following are interpreted as partial filenames: "<cr>" - use previous filename, "x" - replace component, "xxx" - replace station-instrument, "x.xxx" - replace component and station-instrument, "xx.xxx" - replace second and component and station-instrument, any other string without "]" - replace part of filename after "]" (this also works with non-VFBB filenames).
- Extended filename
  All programs accept an auto-rotate azimuth after the filename.
  PRS accepts an auto-rotate azimuth, shift time, and plot time after the filename.
  Examples: 1290249E2.VEW,135
  2.VEW 135 -0.5 3.0 (PRS only)
  1290249E1.VEW,,-0.5,3.0 (PRS only)
- LADC
  If LADC = .true., the trace is automatically "DC" corrected in MVFBB.
- LART
  If LART = .true., eligible traces are auto-rotated in MVFBB if an auto-rotate azimuth is supplied. If LART = .false., auto-rotation is disabled. Thus, the user can make one file-of-extended-filenames and enable or disable auto-rotation, depending on the circumstance, with LART. PLOT3, which only recognizes vertical-component filenames, uses a variation of this idea; if the filename is extended with auto-rotate azimuth IAZMRT, the two horizontal-component traces are rotated into IAZMRT and IAZMRT+90 for plotting.
- Trace-modify mode
  The user controls preset and interactive trace modification in subroutine MVFBB using parameter NMO. Preset trace-modify options are automatically applied to all traces. If NMO = 0, MVFBB does LADC and LART, then prompts the user for trace-modify options. Similarly, if NMO ≥ 1 and the NMOth option is not "continue", MVFBB does LADC, LART, and NMO preset trace-modify options, then prompts the user for more trace-modify options. Interactive trace modification can be disabled (the prompt can be a nuisance), by setting NMO ≥ 1 with "continue" as the NMOth option. Interactive trace modification is not

supported in PLOT3; the NMOth option must be "continue".

- Enable and disable interactive processing or replotting options in SPY setup
  If these options are enabled, the user will be prompted for processing or replotting information for each trace. (The option can still be skipped; by enabling the option, the user is only asking to be prompted.) If the option is never used, disabling it eliminates the prompt (it can be a nuisance).
- Automated processing
  Rapid processing can be accomplished by using a file-of-filenames with auto-DC, auto-rotate, and preset trace-modify options.

In the following examples, text written by the program is printed in normal style, while text written by the user is printed in outline style.

PLOT1 Examples

```
$ run PLOT1

Plot options available:                                    {initialize plotting (call PLOTS)}
   1 = Terminal only
   2 = Batch only
   3 = Preview and prompt (1 & 2)
   4 = No plots
Option? [I20;CR=3]? <cr>

Terminals available:
   1 = Tektronix 4010 (1024 points)
   2 = Tektronix 4014 (4096 points)
   3 = Retrographics VT640 (VT100)
   4 = Megatek 3355
   5 = Megatek 7255
   6 = Envision 220
   7 = Wyse WY-99GT
   8 = Jupiter 7
   9 = DEC VT240 (in 4014 mode)
Device? [I20;CR=3]? <cr>

Batch devices available:
   1 = Disk file
Device? [I20;CR=1]? <cr>

Enter name for plot file [CR=Batch.plt      ] <cr>

Enter seconds to pause after each plot (-1=wait for CR) [I20;CR=-1]? <cr>

Enter simulated screen height (in.) [F20.0;CR=8.0000]? <cr>

PLOT1 setup ...                                                          {setup}

Use a filelist? Type filelist name [a;<cr>=No]: <cr>

Change default TIME,DECIMATE Parameters? = NO                {change time,decimation parameters}
Type new value [YorN;<cr>=OK]: Y
|
Skip & Read (sec) = 0.0000,10.000
```

Type new values [2f;<cr>=OK]: 0.,4.
|
Read decimate = 1
Type new value [i;<cr>=OK]: <cr>
|
Plot decimate = 2
Type new value [i;<cr>=OK]: <cr>

Change default MODIFY Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default PLOT Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

{Example 1}

Type filename [a;S=Setup;X=Stop]: PUB1:[MUELLER.EQK.TEST]0010101A4.ST1

Figure 1

Save this plot [Y/N]? Y

{Example 2}

Type filename [a;S=Setup;X=Stop]: 5,110 {partial filename, extended with auto-rotate azimuth}

Figure 2

Save this plot [Y/N]? Y

{Example 3}

Type filename [a;S=Setup;X=Stop]: S {return to setup}

Change default TIME,DECIMATE Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default MODIFY Parameters? = NO {change modify parameters}
Type new value [YorN;<cr>=OK]: Y
|
Auto-DC = YES
Type new value [YorN;<cr>=OK]: <cr>
|
Auto-Rotate = YES
Type new value [YorN;<cr>=OK]: <cr>
|
Preset Options:
> Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]
> Hipass filter = 2,FC(Hz)[f]
> Lopass filter = 3,FC(Hz)[f]
> Integrate = 4
> Differentiate = 5
> Remove geophone response = 6,NF[i;<cr>=4]
> Pad with temporary zeros = 7,ZT(s)[f]
> Multiply by scale factor = 8,SF[f]
> Continue (disable interactive modify) = C
How many options? Type no. [0≤i≤9;<cr>=1,Continue]: 1 {use 1 preset option; enable interactive modify}
Choose option #1 [a]: 2,1. {hipass filter (1.0Hz)}

Change default PLOT Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Type filename [a;S=Setup;X=Stop]: 4.ST4 {partial filename}
Options: M=ADC,HP(1.00)
> Rotate = 0,IAZMRT(deg)[i]
> Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]
> Hipass filter = 2,FC(Hz)[f]
> Lopass filter = 3,FC(Hz)[f]
> Integrate = 4
> Differentiate = 5
> Remove geophone response = 6,NF[i;<cr>=4]
> Pad with temporary zeros = 7,ZT(s)[f]
> Multiply by scale factor = 8,SF[f]
> Continue = C or <cr>
> StartOver = S
Choose option: 3,10. {lopass filter (10.0Hz)}
Choose option: <cr> {continue}

Figure 3

Save this plot [Y/N]? Y

{Example 4}
Type filename [a;S=Setup;X=Stop]: S {return to setup}

Change default TIME,DECIMATE Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default MODIFY Parameters? = NO {change modify parameters}
Type new value [YorN;<cr>=OK]: Y
|
Auto-DC = YES
Type new value [YorN;<cr>=OK]: <cr>
|
Auto-Rotate = YES
Type new value [YorN;<cr>=OK]: <cr>
|
Preset Options:
> Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]
> Hipass filter = 2,FC(Hz)[f]
> Lopass filter = 3,FC(Hz)[f]
> Integrate = 4
> Differentiate = 5
> Remove geophone response = 6,NF[i;<cr>=4]
> Pad with temporary zeros = 7,ZT(s)[f]
> Multiply by scale factor = 8,SF[f]
> Continue (disable interactive modify) = C
How many options? Type no. [0≤i≤9;<cr>=1,Continue]: 3 {use 3 preset options; disable interactive modify}
Choose option #1 [a]: 2,1. {hipass filter (1.0Hz)}
Choose option #2 [a]: 3,10. {lopass filter (10.0Hz)}
Choose option #3 [a]: C {continue}

Change default PLOT Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Type filename [a;S=Setup;X=Stop]: <cr> {use previous filename}

Figure 3 again

Save this plot [Y/N]? Y

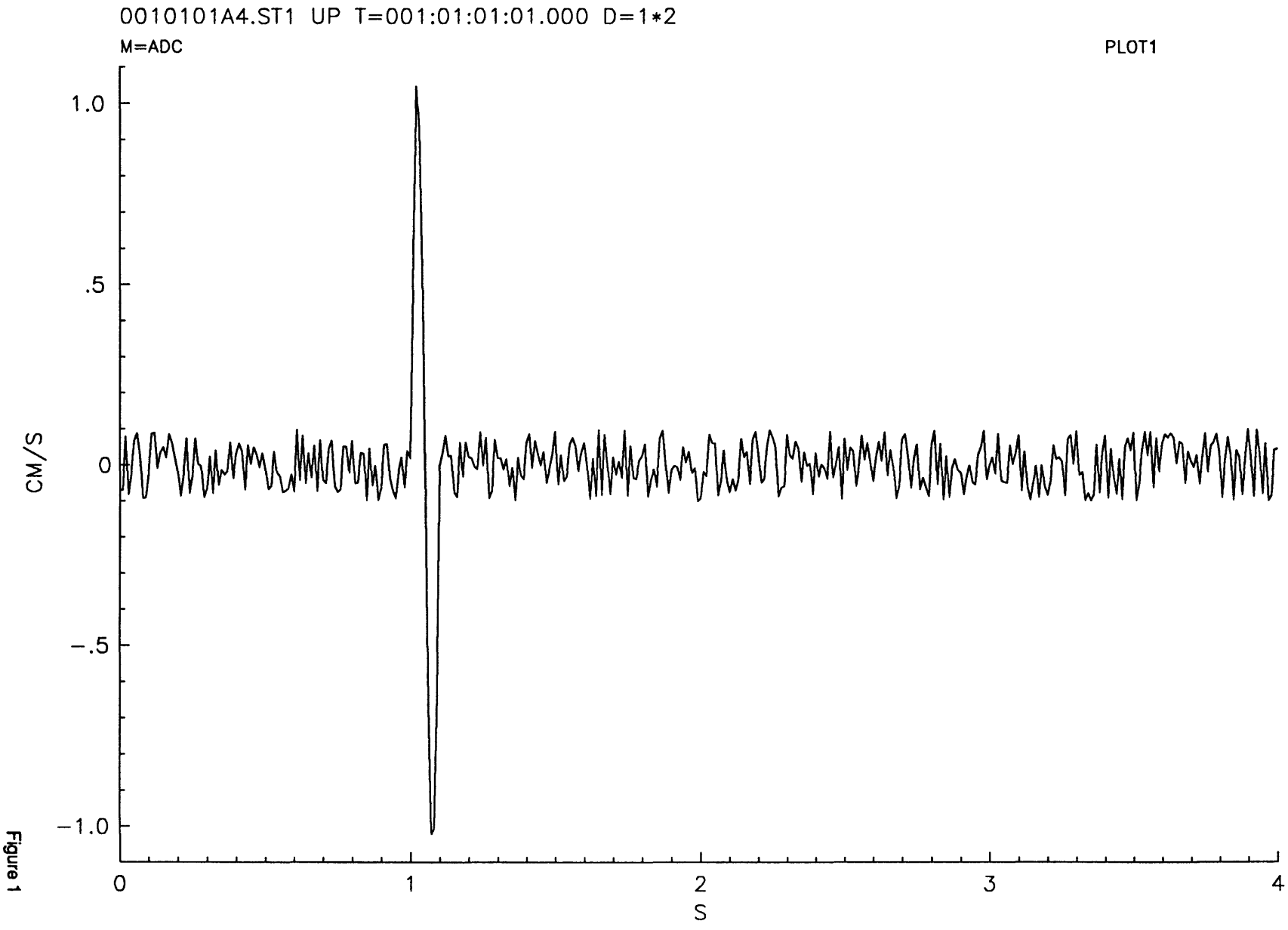
0010101A4.ST1 UP T=001:01:01:01.000 D=1*2
M=ADC
PLOT1
1.0
.5
0
-.5
-1.0
CM/S
0
1
2
3
4
S

Figure 1

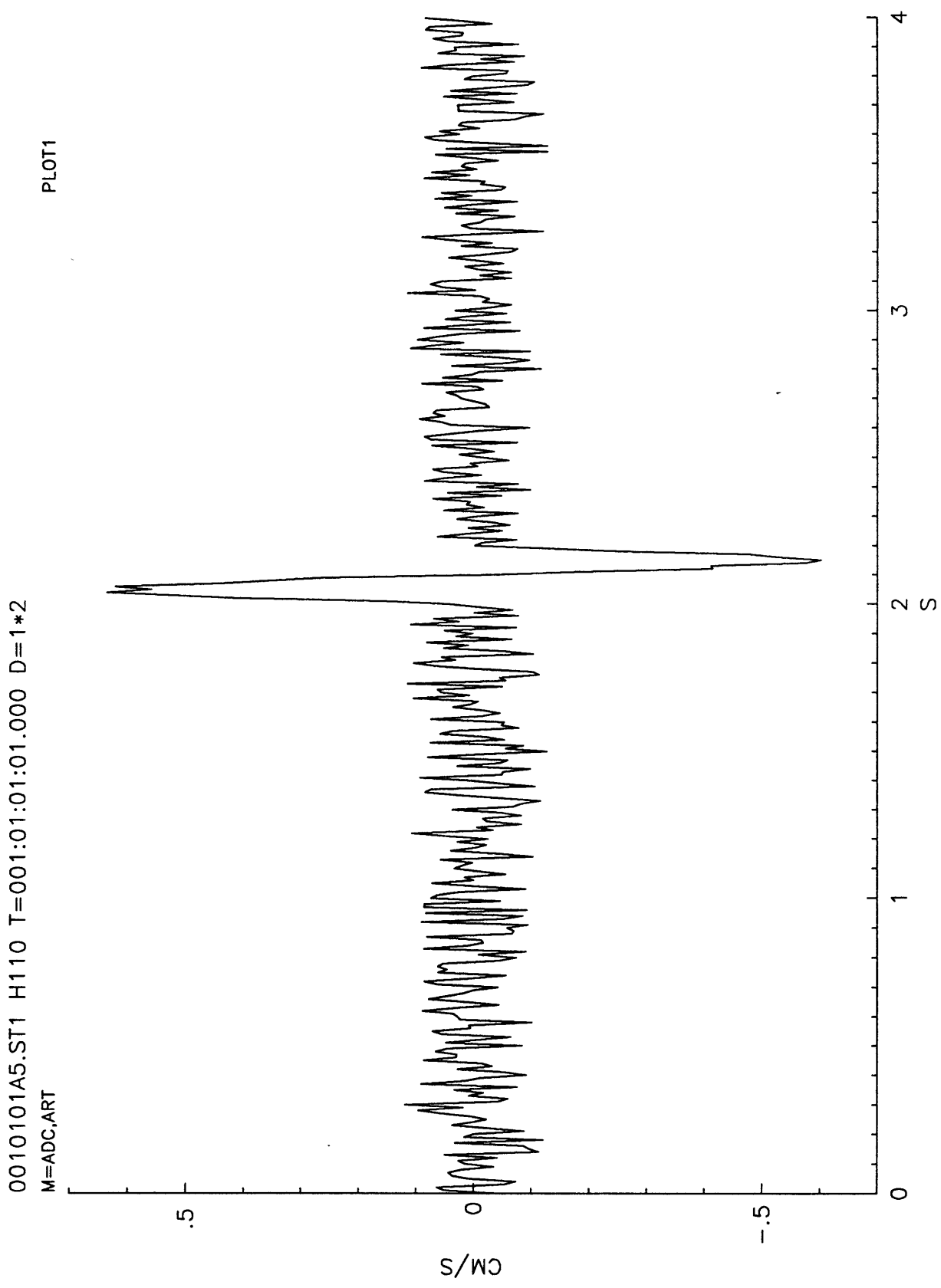
0010101A5.ST1 H110 T=001:01:01:01.000 D=1*2
M=ADC,ART
PLOT1
CM/S
.5
0
-.5
0
1
2
3
4
S

Figure 2

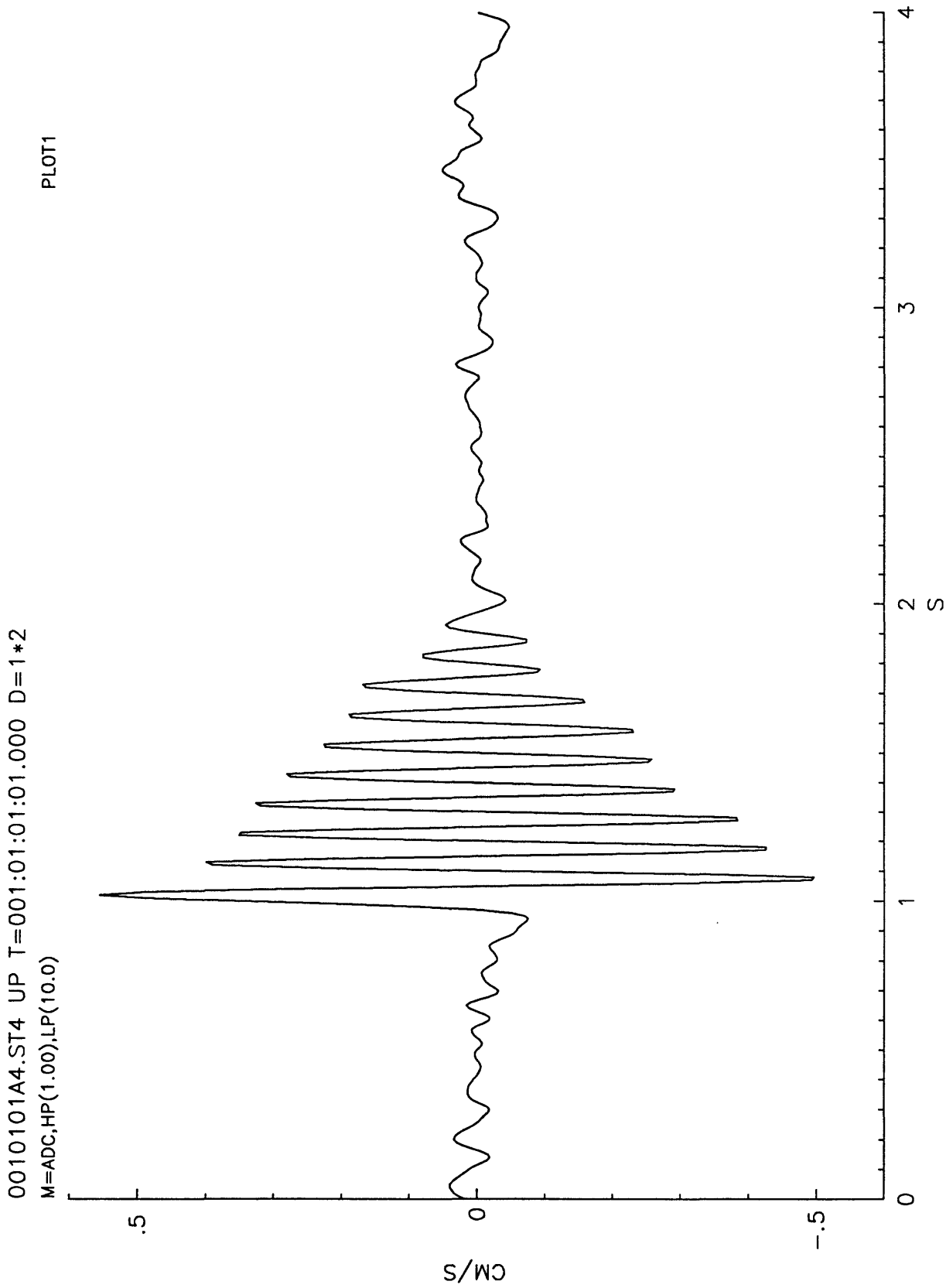
PLOT1
0010101A4.ST4 UP T=001:01:01:01.000 D=1*2
M=ADC,HP(1.00),LP(10.0)
.5
0
-.5
CM/S
0
1
2
3
4
S

Figure 3

{Example 5}

```
Type filename [a;S=Setup;X=Stop]: S                          {return to setup}

Change default TIME,DECIMATE Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default MODIFY Parameters? = NO                       {change modify parameters}
Type new value [YorN;<cr>=OK]: Y
|
Auto-DC = YES
Type new value [YorN;<cr>=OK]: <cr>
|
Auto-Rotate = YES
Type new value [YorN;<cr>=OK]: <cr>
|
Preset Options:
> Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]
> Hipass filter = 2,FC(Hz)[f]
> Lopass filter = 3,FC(Hz)[f]
> Integrate = 4
> Differentiate = 5
> Remove geophone response = 6,NF[i;<cr>=4]
> Pad with temporary zeros = 7,ZT(s)[f]
> Multiply by scale factor = 8,SF[f]
> Continue (disable interactive modify) = C
How many options? Type no. [0≤i≤9;<cr>=1,Continue]: 0   {use 0 preset options; enable interactive modify}

Change default PLOT Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Type filename [a;S=Setup;X=Stop]: <cr>                       {use previous filename}
Options: M=ADC
> Rotate = 0,IAZMRT(deg)[i]
> Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]
> Hipass filter = 2,FC(Hz)[f]
> Lopass filter = 3,FC(Hz)[f]
> Integrate = 4
> Differentiate = 5
> Remove geophone response = 6,NF[i;<cr>=4]
> Pad with temporary zeros = 7,ZT(s)[f]
> Multiply by scale factor = 8,SF[f]
> Continue = C or <cr>
> StartOver = S
Choose option: 2,1.                                          {hipass filter (1.0Hz)}
Choose option: 3,10.                                         {lopass filter (10.0Hz)}
Choose option: <cr>                                          {continue}
```

Figure 3 again

```
Save this plot [Y/N]? Y

Type filename [a;S=Setup;X=Stop]: X

FORTRAN STOP
```

PLOT3 Examples

```
$ run PLOT3
```

```
Plot options available:                                    {initialize plotting (call PLOTS)}
   1 = Terminal only
   2 = Batch only
   3 = Preview and prompt (1 & 2)
   4 = No plots
Option? [I20;CR=3]? <cr>

Terminals available:
   1 = Tektronix 4010 (1024 points)
   2 = Tektronix 4014 (4096 points)
   3 = Retrographics VT640 (VT100)
   4 = Megatek 3355
   5 = Megatek 7255
   6 = Envision 220
   7 = Wyse WY-99GT
   8 = Jupiter 7
   9 = DEC VT240 (in 4014 mode)
Device? [I20;CR=3]? <cr>

Batch devices available:
   1 = Disk file
Device? [I20;CR=1]? <cr>

Enter name for plot file [CR=Batch.plt      ] <cr>

Enter seconds to pause after each plot (-1=wait for CR) [I20;CR=-1]? <cr>

Enter simulated screen height (in.) [F20.0;CR=8.0000]? <cr>

PLOT3 setup ...                                                          {setup}

Use a filelist? Type filelist name [a;<cr>=No]: <cr>

Change default TIME,DECIMATE Parameters? = NO          {change time,decimate parameters}
Type new value [YorN;<cr>=OK]: Y
|
Skip & Read (sec) = 0.0000,10.000
Type new values [2f;<cr>=OK]: 0.,4.
|
Read decimate = 1
Type new value [i;<cr>=OK]: <cr>
|
Plot decimate = 2
Type new value [i;<cr>=OK]: <cr>

Change default MODIFY Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default PLOT Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

                                                                         {Example 1}
Type 1st-component filename [a;S=Setup;X=Stop]: [MUELLER.EQK.TEST]0010101A4.ST1
```

Figure 4

```
Save this plot [Y/N]? Y
                                                                         {Example 2}
Type 1st-component filename [a;S=Setup;X=Stop]: 4,110 {partial filename, extended with auto-rotate azimuth}
```

Figure 5

Save this plot [Y/N]? Y

{Example 3}

Type 1st-component filename [a;S=Setup;X=Stop]: S {return to setup}

Change default TIME,DECIMATE Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default MODIFY Parameters? = NO {change modify parameters}
Type new value [YorN;<cr>=OK]: Y
|
Auto-DC = YES
Type new value [YorN;<cr>=OK]: <cr>
|
Auto-Rotate = YES
Type new value [YorN;<cr>=OK]: <cr>
|
Preset Options:
> Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]
> Hipass filter = 2,FC(Hz)[f]
> Lopass filter = 3,FC(Hz)[f]
> Integrate = 4
> Differentiate = 5
> Remove geophone response = 6,NF[i;<cr>=4]
> Pad with temporary zeros = 7,ZT(s)[f]
> Multiply by scale factor = 8,SF[f]
> Continue (disable interactive modify) = C
How many options? Type no. [1≤i≤9;<cr>=1,Continue]: 3 {use 3 preset options; disable interactive modify}
Choose option #1 [a]: 2,2. {hipass filter (2.0Hz)}
Choose option #2 [a]: 4 {integrate}
Choose option #3 [a]: C {continue}

Change default PLOT Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Type 1st-component filename [a;S=Setup;X=Stop]: 4.ST4 {partial filename}

Figure 6

Save this plot [Y/N]? Y

{Example 4}

Type 1st-component filename [a;S=Setup;X=Stop]: S {return to setup}

Change default TIME,DECIMATE Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default MODIFY Parameters? = NO {change modify parameters}
Type new value [YorN;<cr>=OK]: Y
|
Auto-DC = YES
Type new value [YorN;<cr>=OK]: <cr>
|
Auto-Rotate = YES
Type new value [YorN;<cr>=OK]: <cr>
|

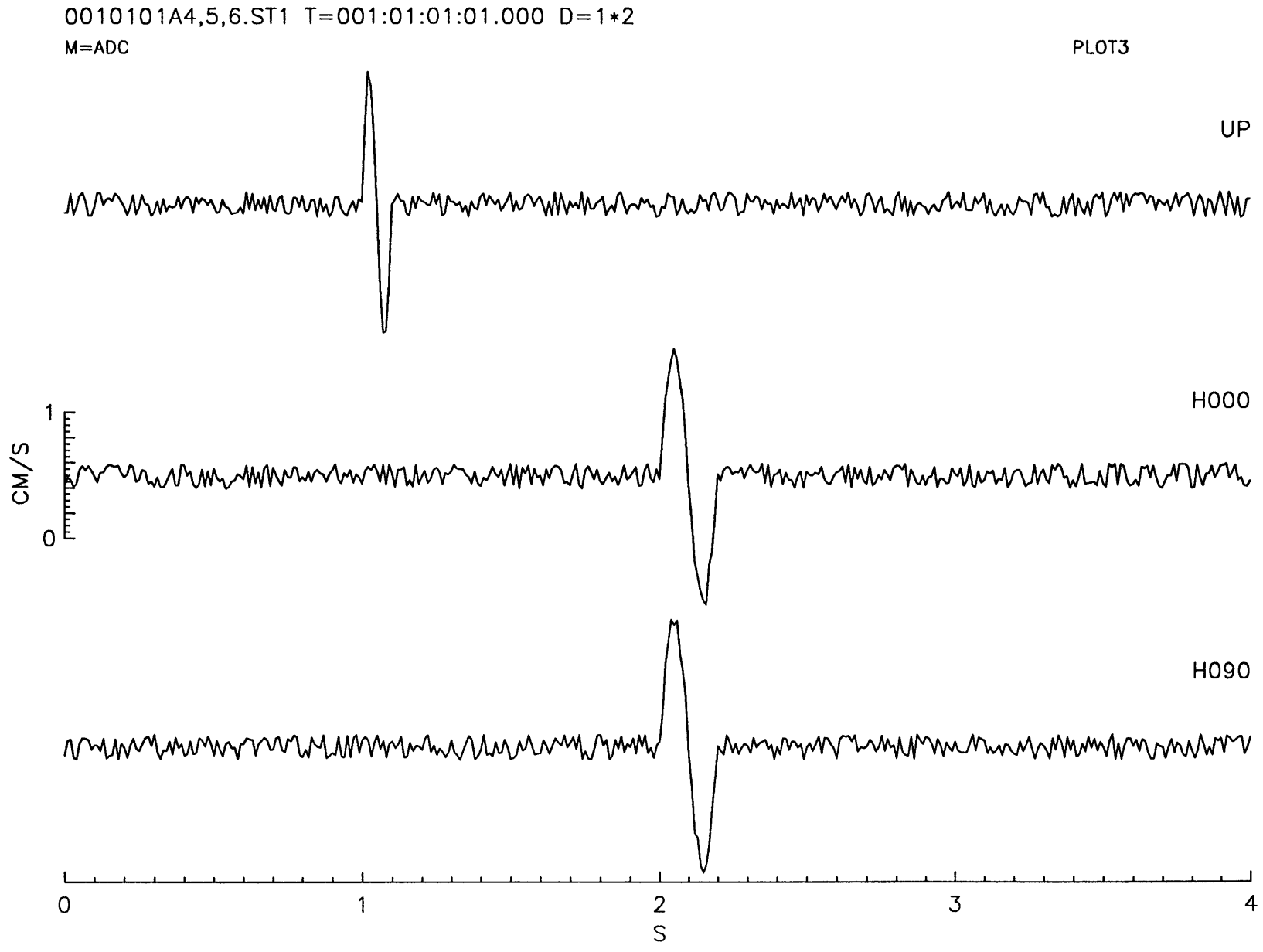
0010101A4,5,6.ST1 T=001:01:01:01.000 D=1*2
M=ADC
PLOT3
UP
H000
H090
CM/S
1
0
0
1
2
3
4
S

Figure 4

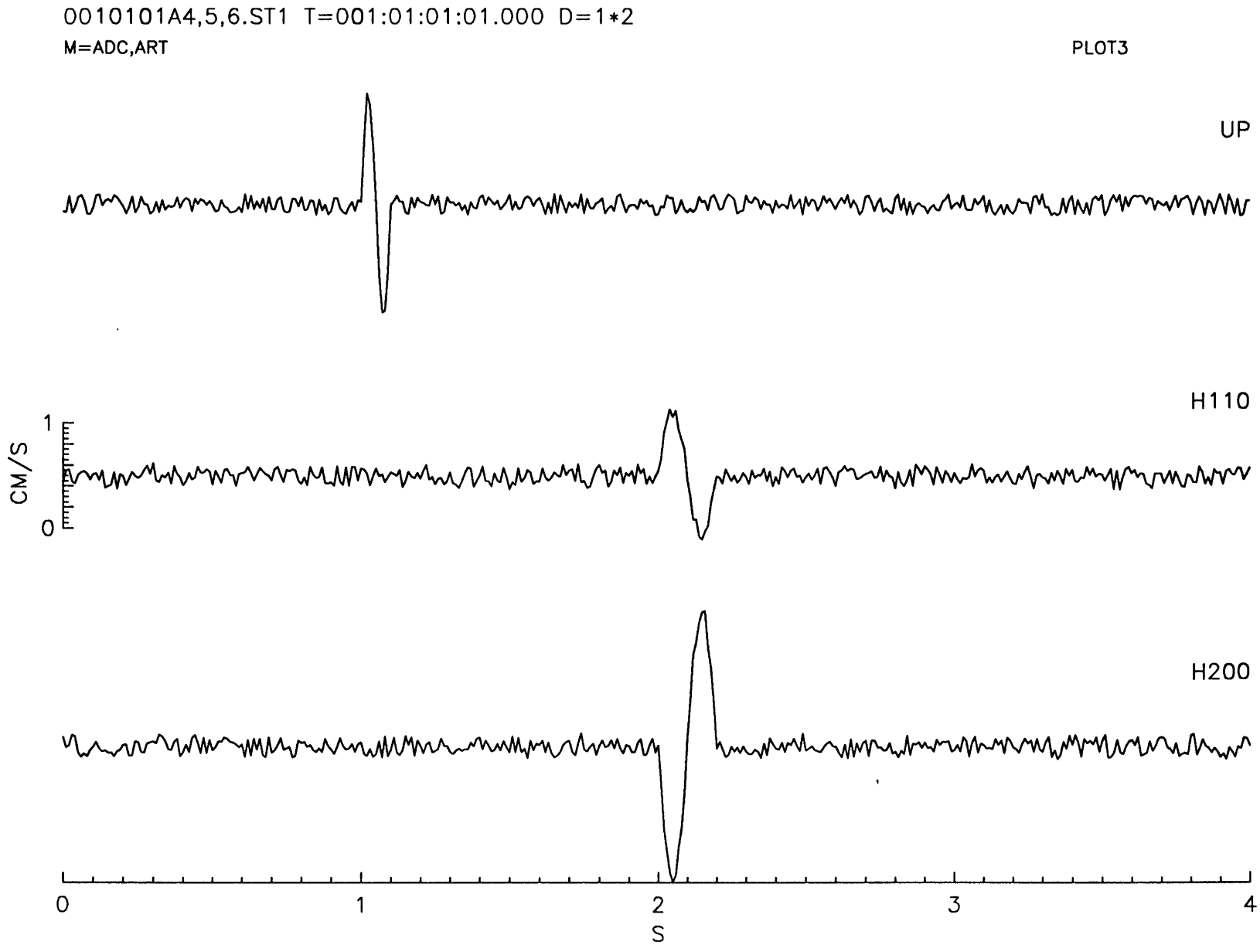
0010101A4,5,6.ST1 T=001:01:01:01.000 D=1*2
M=ADC,ART
PLOT3
UP
H110
H200
CM/S
1
0
0
1
2
3
4
S

Figure 5

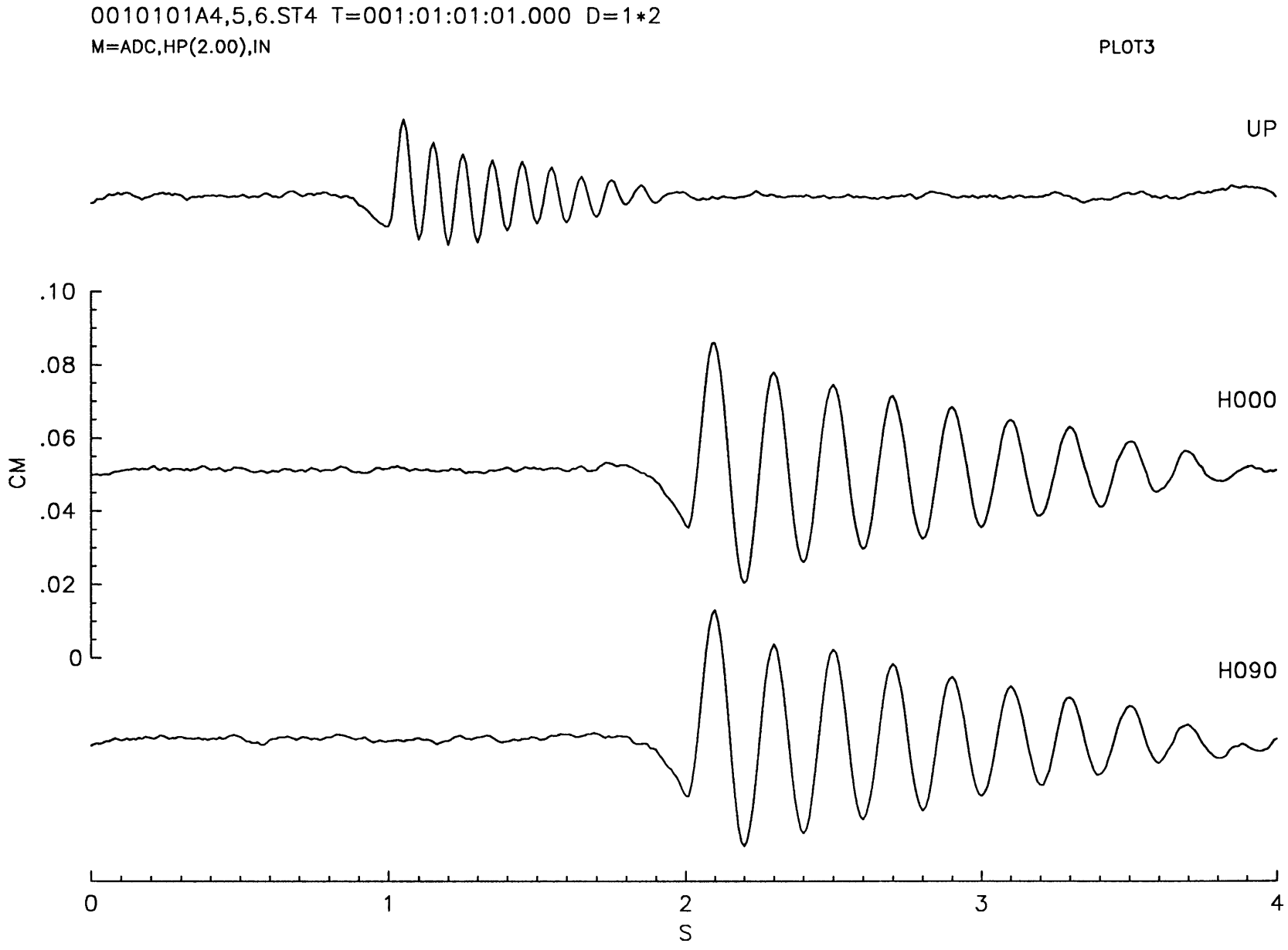
0010101A4,5,6.ST4 T=001:01:01:01.000 D=1*2
M=ADC,HP(2.00),IN
PLOT3
UP
H000
H090
CM
.10
.08
.06
.04
.02
0
0
1
2
3
4
S

Figure 6

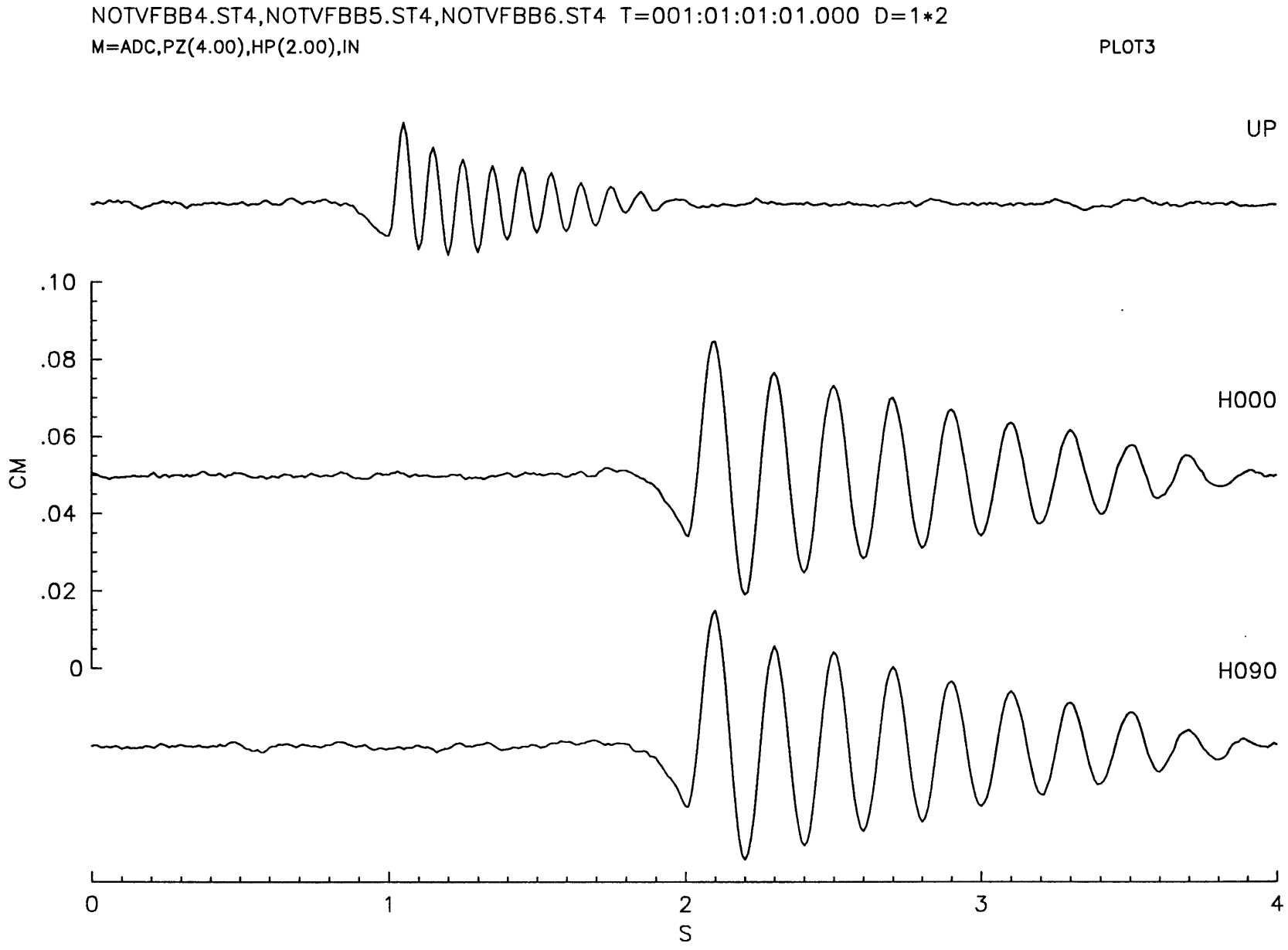
NOTVFBB4.ST4,NOTVFBB5.ST4,NOTVFBB6.ST4 T=001:01:01:01.000 D=1*2
M=ADC,PZ(4.00),HP(2.00),IN
PLOT3
UP
H000
H090
.10
.08
.06
.04
.02
0
CM
0
1
2
3
4
S

Figure 7

```
Preset Options:
> Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]
> Hipass filter = 2,FC(Hz)[f]
> Lopass filter = 3,FC(Hz)[f]
> Integrate = 4
> Differentiate = 5
> Remove geophone response = 6,NF[i;<cr>=4]
> Pad with temporary zeros = 7,ZT(s)[f]
> Multiply by scale factor = 8,SF[f]
> Continue (disable interactive modify) = C
How many options? Type no. [1≤i≤9;<cr>=1,Continue]: 4      {use 4 preset options; disable interactive modify}
Choose option #1 [a]: 7,4.                                  {pad with temporary zeros (4.0s)}
Choose option #2 [a]: 2,2.                                  {hipass filter (2.0Hz)}
Choose option #3 [a]: 4                                     {integrate}
Choose option #4 [a]: C                                     {continue}

Change default PLOT Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Type 1st-component filename [a;S=Setup;X=Stop]: NOTVFBB4.ST4    {non-VFBB filename}
Type 2nd-component filename [a;S=StartOver]: NOTVFBB5.ST4       {non-VFBB filename}
Type 3rd-component filename [a;S=StartOver]: NOTVFBB6.ST4       {non-VFBB filename}
```

Figure 7

```
Save this plot [Y/N]? Y

Type 1st-component filename [a;S=Setup;X=Stop]: X

FORTRAN STOP
```

PRS Examples

```
$ run  [MUELLER.PRS]PRS

Plot options available:                                     {initialize plotting (call PLOTS)}
   1 = Terminal only
   2 = Batch only
   3 = Preview and prompt (1 & 2)
   4 = No plots
Option? [I20;CR=3]? <cr>

Terminals available:
   1 = Tektronix 4010 (1024 points)
   2 = Tektronix 4014 (4096 points)
   3 = Retrographics VT640 (VT100)
   4 = Megatek 3355
   5 = Megatek 7255
   6 = Envision 220
   7 = Wyse WY-99GT
   8 = Jupiter 7
   9 = DEC VT240 (in 4014 mode)
Device? [I20;CR=3]? <cr>

Batch devices available:
   1 = Disk file
Device? [I20;CR=1]? <cr>
```

```
Enter name for plot file [CR=Batch.plt      ] <cr>

Enter seconds to pause after each plot (-1=wait for CR) [I20;CR=-1]? <cr>

Enter simulated screen height (in.) [F20.0;CR=8.0000]? 11.

PRS setup ...                                                          {setup}

Use a filelist? Type filelist name [a;<cr>=No]: <cr>

Change default TIME,DECIMATE Parameters? = NO          {change time,decimate parameters}
Type new value [YorN;<cr>=OK]: Y

Time to plot (sec) = 10.000
Type new value [f;<cr>=OK]: 4.
|
Read decimate = 1
Type new value [i;<cr>=OK]: <cr>
|
Plot decimate = 2
Type new value [i;<cr>=OK]: <cr>

Change default MODIFY Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default PLOT Parameters? = NO                   {change plot parameters}
Type new value [YorN;<cr>=OK]: Y
|
> 1 = Equal size
> 2 = Equal scale ('true amplitude')
Plot amplitude = 1
Type new value [i;<cr>=OK]: <cr>
|
> 1 = Landscape view
> 2 = Portrait view
Plot view = 1
Type new value [i;<cr>=OK]: 2
|
> 1 = Full annotation
> 2 = Sparse annotation
Plot annotation = 1
Type new value [i;<cr>=OK]: <cr>
|
CZ (in) = .10000
Type new value [f;<cr>=OK]: <cr>
XORG,YORG (in) = 1.4000,1.5000
Type new values [2f;<cr>=OK]: <cr>
XLEN,YLEN (in) = 4.8000,8.5000
Type new values [2f;<cr>=OK]: <cr>

                                                                       {Example 1}
Type filename [a;S=Setup;X=Stop]: [MUELLER.EQK.TEST]0010101A4.ST4
Type filename [a;P=Plot;X=Stop]: 5,45          {partial filename, extended with auto-rotate azimuth}
Type filename [a;P=Plot;X=Stop]: 5,135         {partial filename, extended with auto-rotate azimuth}
Type filename [a;P=Plot;X=Stop]: 4,,+0.5                {partial filename, extended with shift time}
Type filename [a;P=Plot;X=Stop]: 4,,-0.5                {partial filename, extended with shift time}
Type filename [a;P=Plot;X=Stop]: 4,,-0.5,2.   {partial filename, extended with shift time and plot time}
Type filename [a;P=Plot;X=Stop]: P
```

Figure 8

```
Save this plot [Y/N]? Y

Type filename [a;S=Setup;X=Stop]: S                                  {return to setup}

Change default TIME,DECIMATE Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default MODIFY Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default PLOT Parameters? = NO                          {change plot parameters}
Type new value [YorN;<cr>=OK]: Y
|
> 1 = Equal size
> 2 = Equal scale ('true amplitude')
Plot amplitude = 1
Type new value [i;<cr>=OK]: 2
|
> 1 = Landscape view
> 2 = Portrait view
Plot view = 2
Type new value [i;<cr>=OK]: <cr>
|
> 1 = Full annotation
> 2 = Sparse annotation
Plot annotation = 1
Type new value [i;<cr>=OK]: <cr>
|
CZ (in) = .10000
Type new value [f;<cr>=OK]: <cr>
XORG,YORG (in) = 1.4000,1.5000
Type new values [2f;<cr>=OK]: <cr>
XLEN,YLEN (in) = 4.8000,8.5000
Type new values [2f;<cr>=OK]: <cr>

                                                                    {Example 2}
Type filename [a;S=Setup;X=Stop]: 4.ST4                             {partial filename}
Type filename [a;P=Plot;X=Stop]: 5,45     {partial filename, extended with auto-rotate azimuth}
Type filename [a;P=Plot;X=Stop]: 5,135    {partial filename, extended with auto-rotate azimuth}
Type filename [a;P=Plot;X=Stop]: 4,,+0.5          {partial filename, extended with shift time}
Type filename [a;P=Plot;X=Stop]: 4,,-0.5          {partial filename, extended with shift time}
Type filename [a;P=Plot;X=Stop]: 4,,-0.5,2.  {partial filename, extended with shift time and plot time}
Type filename [a;P=Plot;X=Stop]: P
```

Figure 9

```
Save this plot [Y/N]? Y

Type filename [a;S=Setup;X=Stop]: X

FORTRAN STOP
```

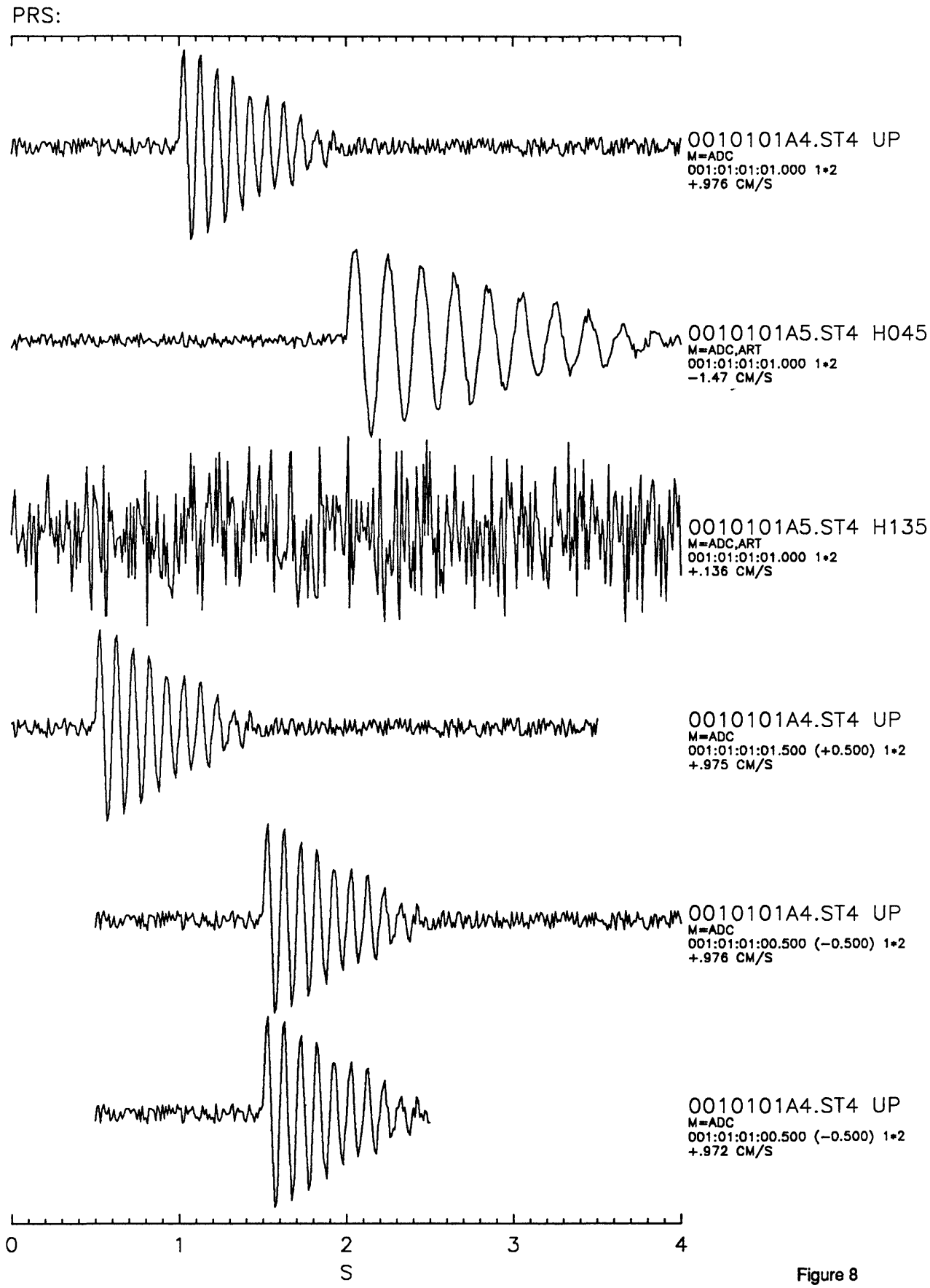
PRS:
0010101A4.ST4 UP
M=ADC
001:01:01:01.000 1*2
+.976 CM/S
0010101A5.ST4 H045
M=ADC,ART
001:01:01:01.000 1*2
-1.47 CM/S
0010101A5.ST4 H135
M=ADC,ART
001:01:01:01.000 1*2
+.136 CM/S
0010101A4.ST4 UP
M=ADC
001:01:01:01.500 (+0.500) 1*2
+.975 CM/S
0010101A4.ST4 UP
M=ADC
001:01:01:00.500 (-0.500) 1*2
+.976 CM/S
0010101A4.ST4 UP
M=ADC
001:01:01:00.500 (-0.500) 1*2
+.972 CM/S
0
1
2
3
4
S

Figure 8

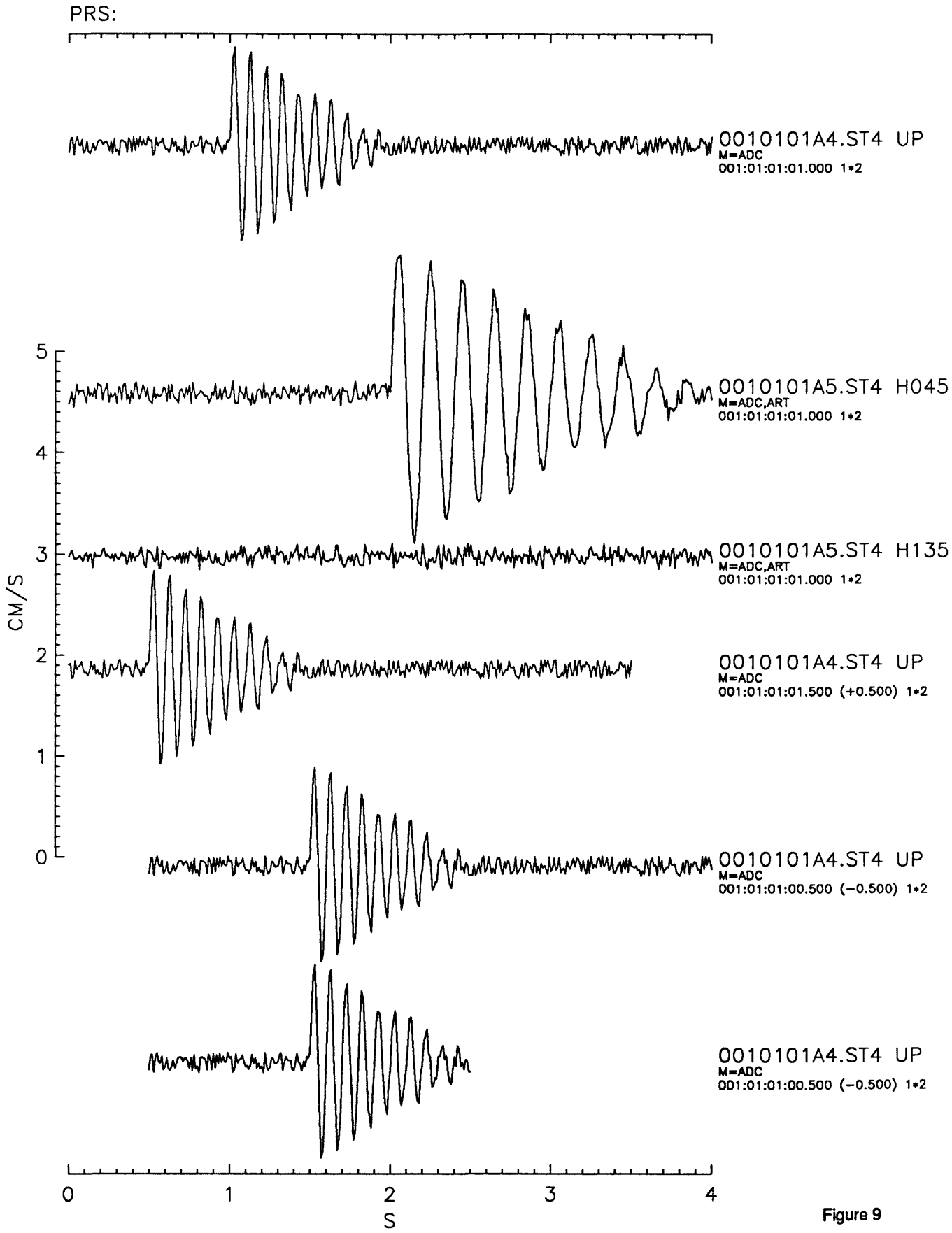
PRS:
0010101A4.ST4 UP
M=ADC
001:01:01:01.000 1*2
0010101A5.ST4 H045
M=ADC,ART
001:01:01:01.000 1*2
0010101A5.ST4 H135
M=ADC,ART
001:01:01:01.000 1*2
0010101A4.ST4 UP
M=ADC
001:01:01:01.500 (+0.500) 1*2
0010101A4.ST4 UP
M=ADC
001:01:01:00.500 (-0.500) 1*2
0010101A4.ST4 UP
M=ADC
001:01:01:00.500 (-0.500) 1*2
CM/S
0
1
2
3
4
5
0
1
2
3
4
S

Figure 9

SPY Examples

```
$ run [MUELLER.SPY]SPY

Plot options available:                                    {initialize plotting (call PLOTS)}
   1 = Terminal only
   2 = Batch only
   3 = Preview and prompt (1 & 2)
   4 = No plots
Option? [I20;CR=3]? <cr>

Terminals available:
   1 = Tektronix 4010 (1024 points)
   2 = Tektronix 4014 (4096 points)
   3 = Retrographics VT640 (VT100)
   4 = Megatek 3355
   5 = Megatek 7255
   6 = Envision 220
   7 = Wyse WY-99GT
   8 = Jupiter 7
   9 = DEC VT240 (in 4014 mode)
Device? [I20;CR=3]? <cr>

Batch devices available:
   1 = Disk file
Device? [I20;CR=1]? <cr>

Enter name for plot file [CR=Batch.plt     ] <cr>

Enter seconds to pause after each plot (-1=wait for CR) [I20;CR=-1]? <cr>

Enter simulated screen height (in.) [F20.0;CR=8.0000]? <cr>

SPY setup ...                                                                    {setup}

Use a filelist? Type filelist name [a;<cr>=No]: <cr>

Change default TIME,DECIMATE Parameters? = NO          {change time,decimate parameters}
Type new value [YorN;<cr>=OK]: Y
|
Skip,Read (sec) = 0.0000,10.000
Type new values [2f;<cr>=OK]: 0.,4.
|
Read decimate = 1
Type new value [i;<cr>=OK]: <cr>
|
Plot decimate = 2
Type new value [i;<cr>=OK]: <cr>

Change default MODIFY Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default PROCESS Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default PLOT Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default ANNOTATE Parameters? = NO
```

Type new value [YorN;<cr>=OK]: <cr>

{Example 1}

Type filename [a;S=Setup;X=Stop]: [MUELLER.EQK.TEST]0010101A4.ST4

Figure 10

(Choose FFT window with cursor)

Save this plot [Y/N]? Y

512 positive frequencies in FFT

Multiply spectrum by Omega**I? I = 0 {skip}
Type new value [i;<cr>=OK;S=Skip]: S

Smooth spectrum? Smooth(Hz) = 0.00 {skip}
Type new value [f;<cr>=OK;S=Skip]: S

Save (denominator) for spectral ratio? = NO
Type new value [YorN;<cr>=OK]: <cr>

> 1 = automatic {automatic plot}
> 2 = customize
> 3 = no plot
Plot mode = 1
Type new value [i;<cr>=OK]: <cr>

Figure 11

(Choose annotation location with cursor)

Save this plot [Y/N]? Y

{Example 2}

Type filename [a;S=Setup;X=Stop]: <cr> {use previous filename}

Figure 10 again

(Choose FFT window with cursor)

Save this plot [Y/N]? Y

512 positive frequencies in FFT

Multiply spectrum by Omega**I? I = 0 {multiply spectrum by power of omega (-1)}
Type new value [i;<cr>=OK;S=Skip]: -1

Smooth spectrum? Smooth(Hz) = 0.00 {smooth spectrum (≈2.0Hz)}
Type new value [f;<cr>=OK;S=Skip]: 2.

Save (denominator) for spectral ratio? = NO
Type new value [YorN;<cr>=OK]: <cr>

> 1 = automatic {customize plot}
> 2 = customize
> 3 = no plot
Plot mode = 1
Type new value [i;<cr>=OK]: 2
Freq axis min,max (Hz) = .19531,99.805
Type new values [2f;<cr>=OK]: .5,50.

```
Freq data min,max (Hz) = .50000,50.000
Type new values [2f;<cr>=OK]: 1.,25.
Ampl axis min,max = .000036755,.0025305
Type new values [2f;<cr>=OK]: 1.E-5,1.E-2
```

Figure 12

(Choose annotation location with cursor)

```
Save this plot [Y/N]? Y

                                                                    {Example 3}
Type filename [a;S=Setup;X=Stop]: S                                 {return to setup }

Change default TIME,DECIMATE Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default MODIFY Parameters? = NO                              {change modify parameters}
Type new value [YorN;<cr>=OK]: Y
|
Auto-DC = YES
Type new value [YorN;<cr>=OK]: <cr>
|
Auto-Rotate = YES
Type new value [YorN;<cr>=OK]: <cr>
|
Preset Options:
> Hipass filter = 2,FC(Hz)[f]
> Lopass filter = 3,FC(Hz)[f]
> Integrate = 4
> Differentiate = 5
> Remove geophone response = 6,NF[i;<cr>=4]
> Pad with temporary zeros = 7,ZT(s)[f]
> Multiply by scale factor = 8,SF[f]
> Continue (disable interactive modify) = C
How many options? Type no. [0≤i≤9;<cr>=1,Continue]: 0      {use 0 preset options; enable interactive mode}

Change default PROCESS Parameters? = NO                             {change process parameters}
Type new value [YorN;<cr>=OK]: Y
|
Taper front,back (%) = 10,10                                        {change taper}
Type new values [2i;<cr>=OK]: 5,40
|
> 1 = linear vs linear
> 2 = log vs log
> 3 = linear vs log
> 4 = log vs linear
Spectrum style = 2
Type new value [i;<cr>=OK]: <cr>
|
Enable (Y) or Disable (N) Interactive Processing:
Multiply by power-of-omega? = YES
Type new value [YorN;<cr>=OK]: <cr>
Smooth? = YES
Type new value [YorN;<cr>=OK]: <cr>
A-A Filter correction? = NO
Type new value [YorN;<cr>=OK]: <cr>
Instrument correction? = NO
Type new value [YorN;<cr>=OK]: <cr>
Attenuation correction? = NO                                        {enable attenuation correction}
```

```
Type new value [YorN;<cr>=OK]: Y

Change default PLOT Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Change default ANNOTATE Parameters? = NO
Type new value [YorN;<cr>=OK]: <cr>

Type filename [a;S=Setup;X=Stop]: <cr>                    {use previous filename}
Options: M=ADC
> Rotate = 0,IAZMRT(deg)[i]
> Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]
> Hipass filter = 2,FC(Hz)[f]
> Lopass filter = 3,FC(Hz)[f]
> Integrate = 4
> Differentiate = 5
> Remove geophone response = 6,NF[i;<cr>=4]
> Pad with temporary zeros = 7,ZT(s)[f]
> Multiply by scale factor = 8,SF[f]
> Continue = C or <cr>
> StartOver = S
Choose option: 7,4.                    {pad with temporary zeros (4.0s)}
Choose option: 2,1.                    {hipass filter (1.0Hz)}
Choose option: 4                       {integrate}
Choose option: <cr>                    {continue}
```

Figure 13

(Choose FFT window with cursor)

```
Save this plot [Y/N]? Y

512 positive frequencies in FFT

Multiply spectrum by Omega**I? I = -1                    {skip}
Type new value [i;<cr>=OK;S=Skip]: S

Smooth spectrum? Smooth(Hz) = 2.00                       {smooth spectrum (≈2.0Hz)}
Type new value [f;<cr>=OK;S=Skip]: <cr>

Attenuation correction?                                  {attenuation correction}
Q0,QETA = 0.00,0.00
Type new values [2f;<cr>=OK;S=Skip]: 100.,0.
Distance(km) = 0.0000
Type new value [f;<cr>=OK]: 10.
Velocity(km/s) = 0.0000
Type new value [f;<cr>=OK]: 3.5

Save (denominator) for spectral ratio? = NO
Type new value [YorN;<cr>=OK]: <cr>

> 1 = automatic                                          {customize plot}
> 2 = customize
> 3 = no plot
Plot mode = 1
Type new value [i;<cr>=OK]: 2
Freq axis min,max (Hz) = .19531,99.805
Type new values [2f;<cr>=OK]: .5,50.
Freq data min,max (Hz) = .50000,50.000
```

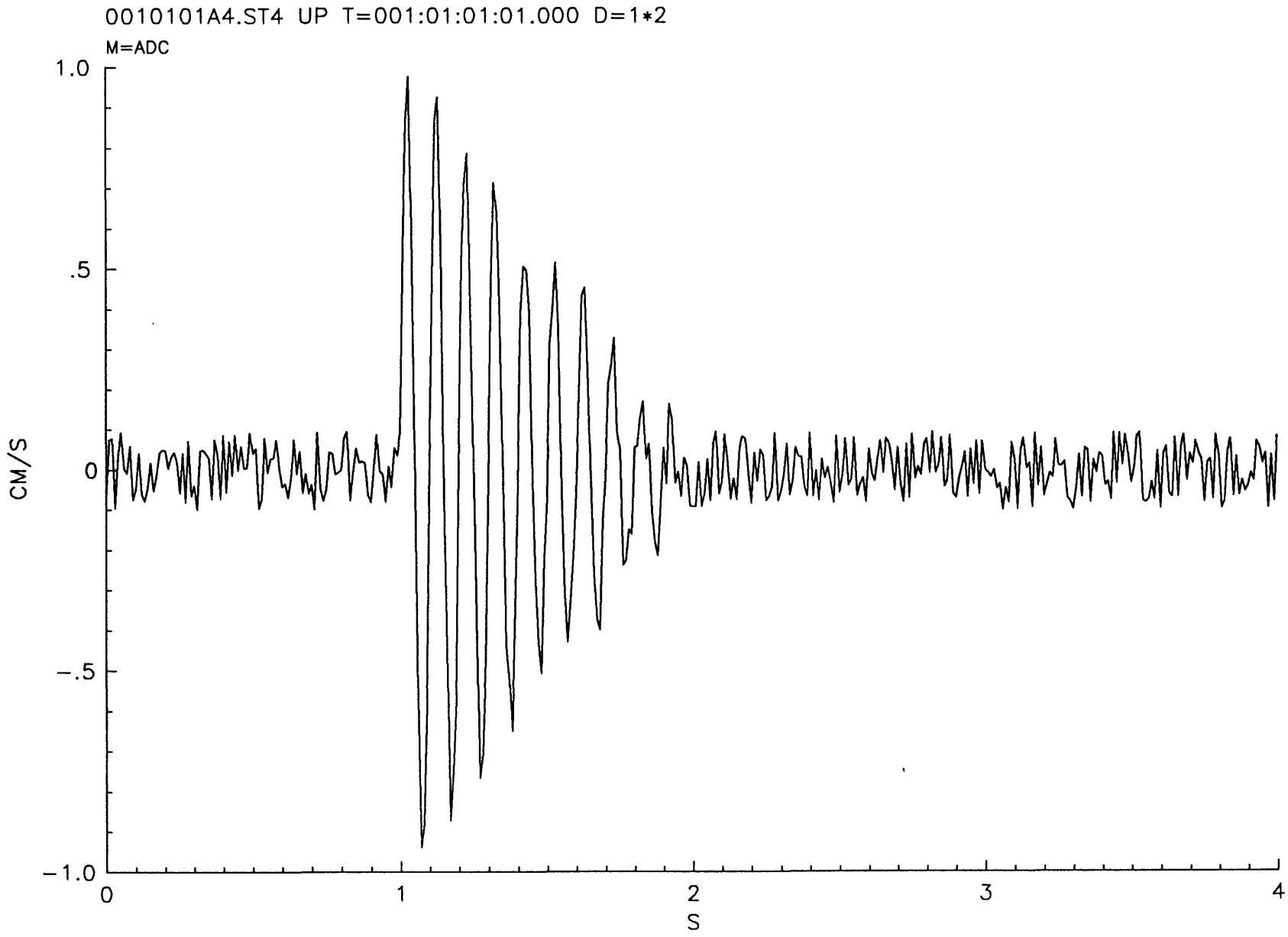
0010101A4.ST4 UP T=001:01:01:01.000 D=1*2
M=ADC
1.0
.5
0
-.5
-1.0
CM/S
0
1
2
3
4
S

Figure 10

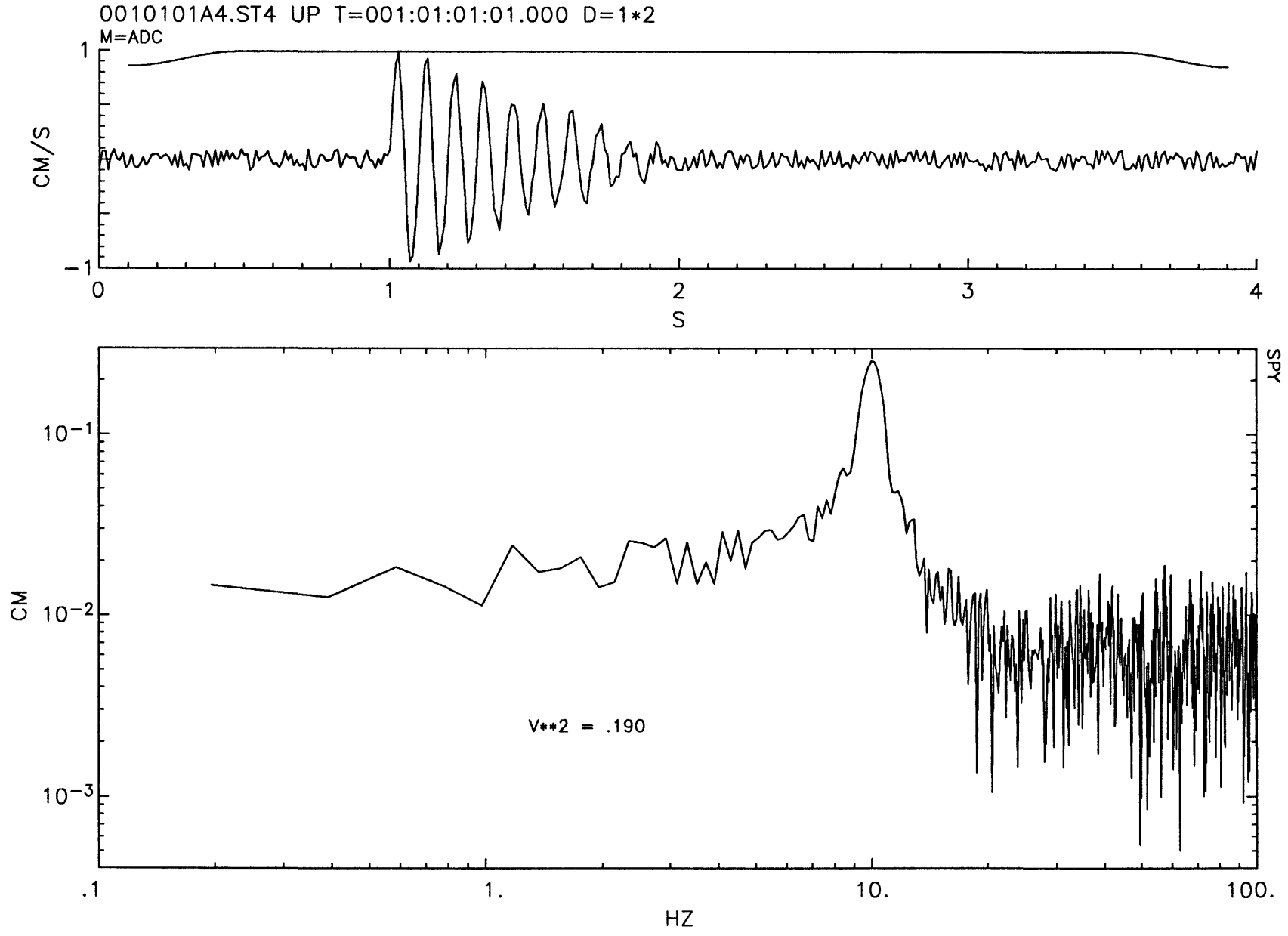
0010101A4.ST4 UP T=001:01:01:01.000 D=1*2
M=ADC
CM/S
1
-1
0
1
2
3
4
S
CM
10^{-1}
10^{-2}
10^{-3}
V**2 = .190
.1
1.
10.
100.
HZ
SPY

Figure 11

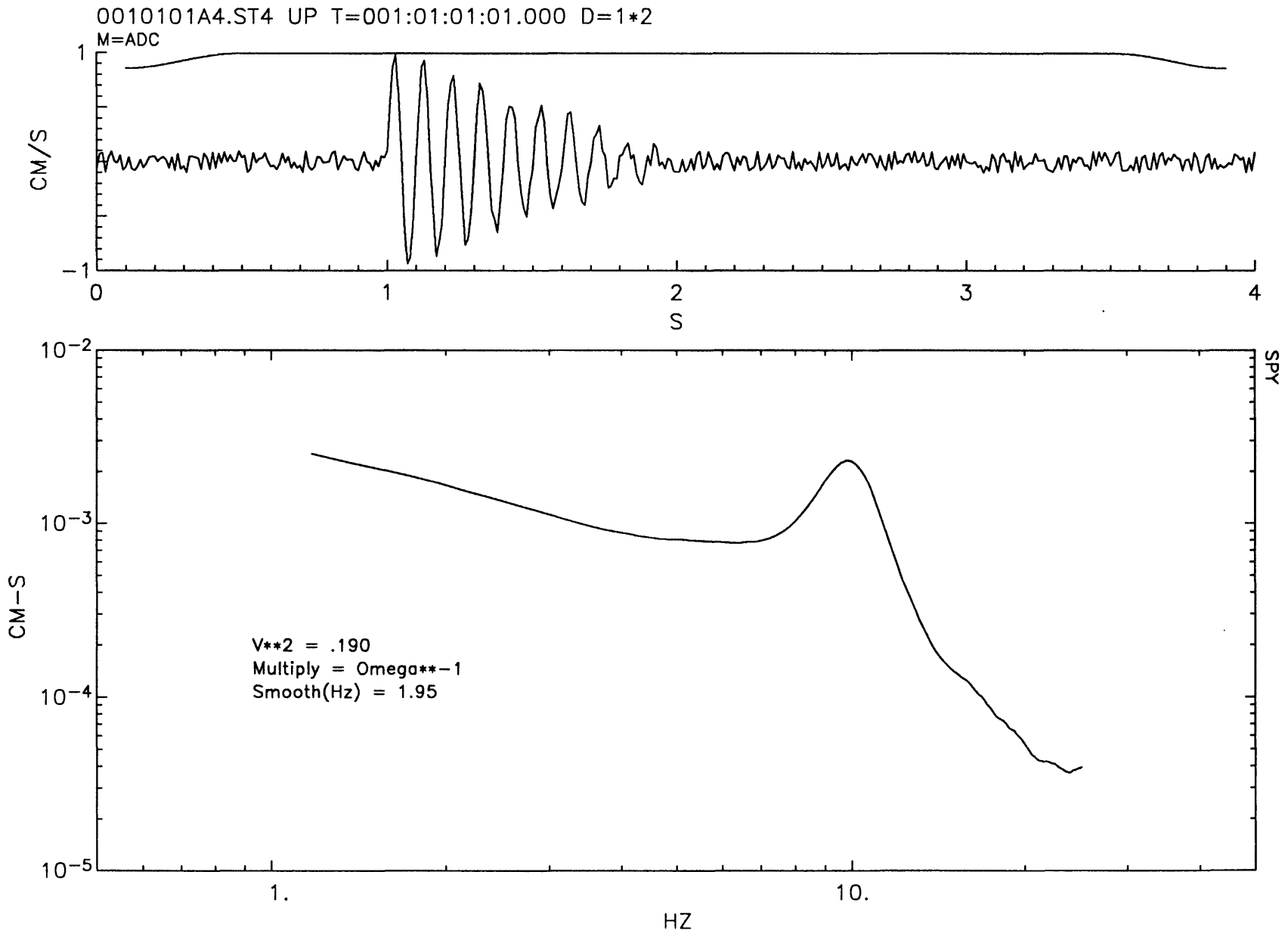
0010101A4.ST4 UP T=001:01:01:01.000 D=1*2
M=ADC
CM/S
1
-1
0
1
2
3
4
S
10^{-2}
10^{-3}
10^{-4}
10^{-5}
CM-S
V**2 = .190
Multiply = Omega**-1
Smooth(Hz) = 1.95
1.
10.
HZ
SPY

Figure 12

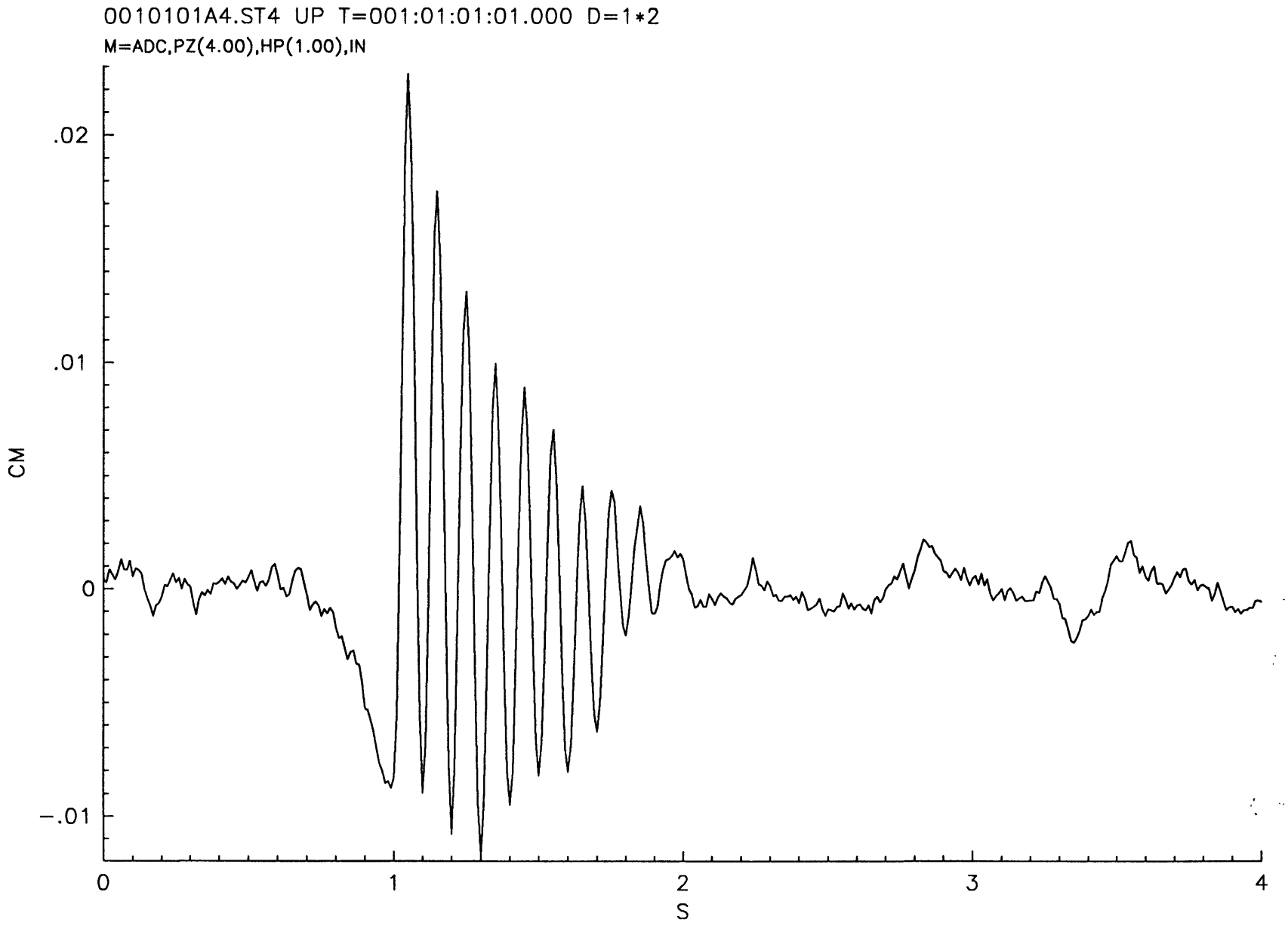
0010101A4.ST4 UP T=001:01:01:01.000 D=1*2
M=ADC,PZ(4.00),HP(1.00),IN
.02
.01
0
-.01
CM
0
1
2
3
4
S

Figure 13

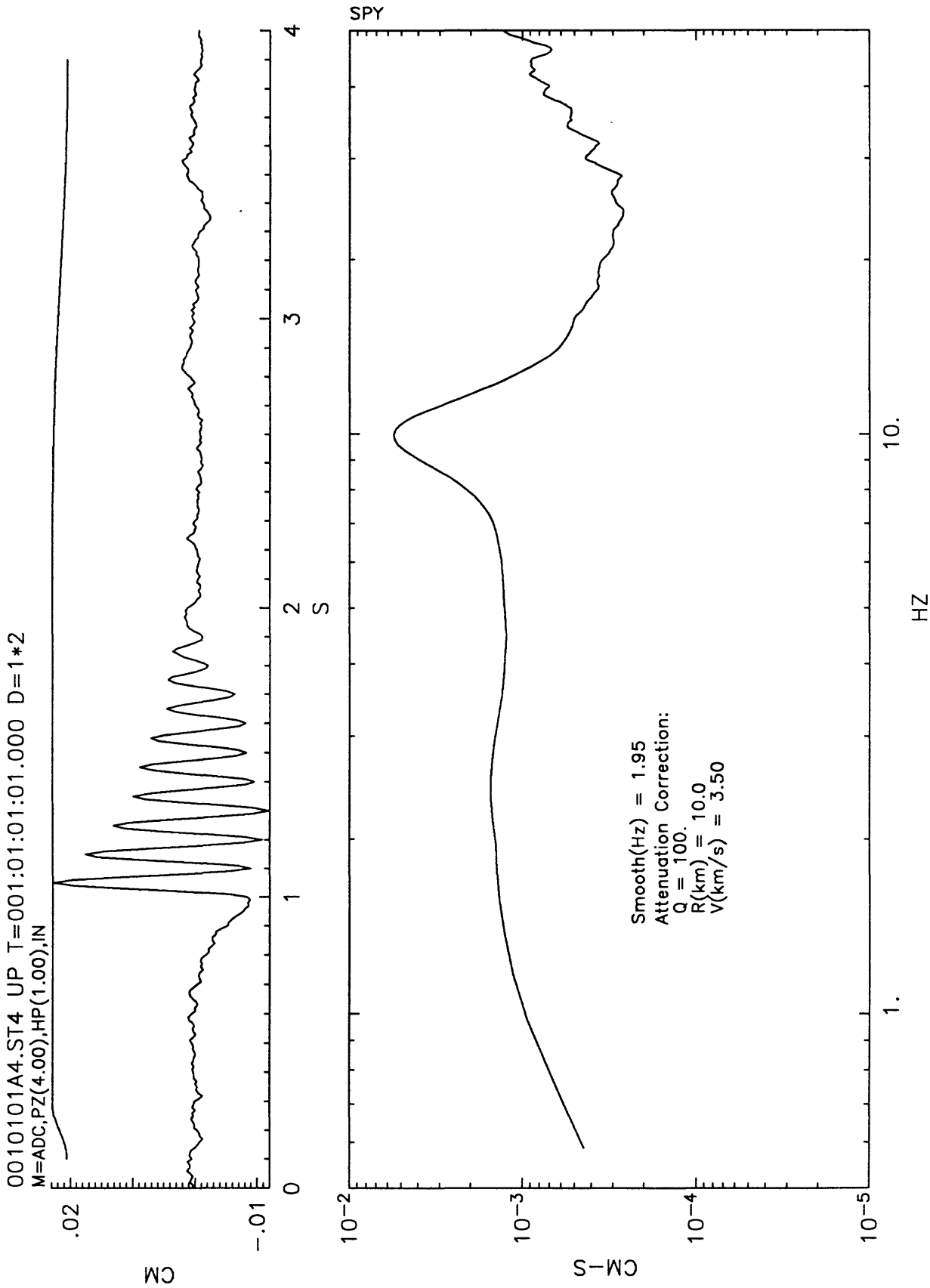
0010101A4.ST4 UP T=001:01:01:01.000 D=1*2
M=ADC,PZ(4.00),HP(1.00),IN
.02
-.01
CM
0
1
2
3
4
S
SPY
10^{-2}
10^{-3}
10^{-4}
10^{-5}
CM-S
Smooth(Hz) = 1.95
Attenuation Correction:
Q = 100.
R(km) = 10.0
V(km/s) = 3.50
1.
10.
HZ

Figure 14

Type new values [2f;<cr>=OK]: <cr>
Ampl axis min,max = .00026168,.0055599
Type new values [2f;<cr>=OK]: 1.E-5,1.E-2

Figure 14

(Choose annotation location with cursor)

Save this plot [Y/N]? Y

Type filename [a;S=Setup;X=Stop]: X

FORTRAN STOP

Acknowledgements

I wish to thank my colleagues at the U. S. Geological Survey in Menlo Park who have tested several versions of these programs. G. Glassmoyer deserves special mention for helping me improve them through several iterations - he is the champion bug finder. D. Boore and E. Cranswick taught me to think about the best ways to process and present seismic data.

References

Mueller, C. S., A. G. Brady, A. M. Converse, and J. C. Watson (1988). Near-source high-frequency seismic waveform data available from the U. S. Geological Survey, *U. S. Geol. Surv. Open-File Report 88-596.*

Appendix A

FORTRAN Source Code: Main programs PLOT1, PLOT3, PRS, and SPY

```
************************************************************ PLOT1  ***

      program PLOT1

c PLOT1
c Plot 1-component ES&G-block-binary seismogram.
c Can extend filename with auto-rotate azm (VFBB only).

      parameter (ISIZ=65536)
      complex CZW
      dimension Z1(ISIZ),ZW(ISIZ),CZW(ISIZ)
      logical LCD,LFL,LVFBB
      character BUF*80,FN1*80,LBL*80,DATBUF*9,C1*1

      include '[MUELLER.FS.SD]VFCOMMON.FI'

      call DATE (DATBUF)
      call PLOTS (0,0,0)
      LUFL = 10 !logical unit for input filelist (file-of-filenames)
      LUVFBB = 11 !logical unit for VFBB data file

90000 format (' ')
90001 format (' |')

c Setup.
10    write (6,90010)
90010 format ('0PLOT1 setup ...')
      TSKIP = 0.
      TREAD = 10.
      IRDEC = 1
      IPDEC = 2
      LADC = .true.
      LART = .true.
      NMO = 1
      NMOBUF(1) = 1
      MOBUF(1) = 'C'
      CZ = 0.10
      XORG = 1.5
      YORG = 1.2
      XLEN = 8.0
      YLEN = 5.5
c Setup data-input mode.
20    write (6,90020)
90020 format (' '/' Use filelist? Type filelist name [a;<cr>=No]: ',$)
      read (5,90021,err=20) NBUF,BUF
90021 format (q,a)
      if (NBUF.eq.0) then
        LFL = .false.
      else
        open (unit=LUFL,file=BUF,err=20,status='OLD',readonly)
        LFL = .true.
      endif
c Re-setup @30.
c Setup trace-time,-decimate parameters.
30    LCD = .false.
      write (6,90000)
      call QPL ('Change default TIME,DECIMATE Parameters?',40,LCD)
      if (.not.LCD) goto 39
      write (6,90001)
      call QPF2 ('Skip & Read (sec)',17,TSKIP,TREAD)
      write (6,90001)
      call QPI ('Read decimate',13,IRDEC)
      write (6,90001)
      call QPI ('Plot decimate',13,IPDEC)
39    continue
c Setup trace-modify parameters.
40    LCD = .false.
      write (6,90000)
      call QPL ('Change default MODIFY Parameters?',33,LCD)
      if (.not.LCD) goto 49
      write (6,90001)
      call QPL ('Auto-DC',7,LADC)
      write (6,90001)
      call QPL ('Auto-Rotate',11,LART)
41    write (6,90041)
90041 format (' |'/' Preset Options:'/
     * ' > Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]'/
     * ' > Hipass filter = 2,FC(Hz)[f]'/
     * ' > Lopass filter = 3,FC(Hz)[f]'/
     * ' > Integrate = 4'/
     * ' > Differentiate = 5'/
     * ' > Remove geophone response = 6,NF[i;<cr>=4]'/
     * ' > Pad with temporary zeros = 7,ZT(s)[f]'/
     * ' > Multiply by scale factor = 8,SF[f]'/
     * ' > Continue (disable interactive modify) = C')
42    write (6,90042) NMOMAX
90042 format (' How many options?
```

```
     * Type no. [0=<i=<',i1,';<cr>=1,Continue]: ',$)
       read (5,90043,err=42) N,NMO
90043  format (q,i12)
       if (N.eq.0) then
         NMO = 1
         NMOBUF(1) = 1
         MOBUF(1) = 'C'
         goto 49
       endif
       if (NMO.lt.0.or.NMO.gt.NMOMAX) goto 42
       do 46 I=1,NMO
44       write (6,90044) I
90044    format (' Choose option #',i1,' [a]: ',$)
         read (5,90045,err=44) NMOBUF(I),MOBUF(I)
90045    format (q,a)
         C1 = MOBUF(I)(1:1)
         if (C1.eq.'c') C1 = 'C'
         if ((C1.lt.'1'.or.C1.gt.'8').and.C1.ne.'C') goto 44
46       continue
49     continue
c Setup trace-plot parameters.
50     LCD = .false.
       write (6,90000)
       call QPL ('Change default PLOT Parameters?',31,LCD)
       if (.not.LCD) goto 59
       write (6,90001)
       call QPF ('CZ (in)',7,CZ)
       call QPF2 ('XORG,YORG (in)',14,XORG,YORG)
       call QPF2 ('XLEN,YLEN (in)',14,XLEN,YLEN)
59     TZ = CZ/3
       write (6,90000)

c Begin main loop.
c Get filename (extend with auto-rotate azm (VFBB only)).
100    if (LFL) then
120      read (LUFL,90120,err=801,end=1000) NBUF,BUF
90120    format (q,a)
         if (NBUF.eq.0.or.BUF(1:1).eq.'|') goto 120
         write (6,90121) BUF(1:NBUF)
90121    format (' ',a)
       else
130      write (6,90130)
90130    format (' '/' Type filename [a;S=Setup;X=Stop]: ',$)
         read (5,90131,err=130) NBUF,BUF
90131    format (q,a)
         if (BUF(1:2).eq.'S '.or.BUF(1:2).eq.'s ') goto 30
         if (BUF(1:2).eq.'X '.or.BUF(1:2).eq.'x ') goto 1001
       endif
       if (NBUF.eq.0) goto 140
       IAZMRT = -1
       call RCI (2,BUF,NBUF,IAZMRT,IR)
       call RCC (1,BUF,NBUF,BUF,NBUF,IR)
       if (LVFBB.and.NBUF.eq.1) then
         FN1(IDT-1:IDT-1) = BUF
       elseif (LVFBB.and.NBUF.eq.3) then
         FN1(IDT+1:IDT+3) = BUF
       elseif (LVFBB.and.NBUF.eq.5) then
         FN1(IDT-1:IDT+3) = BUF
       elseif (LVFBB.and.NBUF.eq.6) then
         FN1(IDT-2:IDT+3) = BUF
       elseif (INDEX(BUF(1:NBUF),']').eq.0) then
         FN1(IRB+1:) = BUF(1:NBUF)//' '
       else
         FN1 = BUF
       endif

c Parse, open, read, and modify. Get Z1.
140    call PARSEF (FN1,IC2,IC1,ILB,IRB,IDT,ISC,IEND,LVFBB,IP)
       if (IP.ne.0) goto 810
       open (unit=LUVFBB,file=FN1,err=811,status='OLD',readonly,
     *       access='DIRECT',recl=128)
       call RVFBB (Z1,ZN1,ISIZ)
       close (unit=LUVFBB)
       if (NP.le.0) goto 812
       call CLRSTR (MOLBL)
       call MVFBB (Z1,ZW,CZW,ZN1,ISIZ,FN1)
       if (MOLBL(1:9).eq.'StartOver') goto 100

c Get LBL and plot.
       call WCC (FN1,IRB+1,ISC-1,LBL,1,NLBL)
       call WCC (' ',1,1,LBL,NLBL+1,NLBL)
       call WCCAZM (IVA,IHA,LBL,NLBL+1,NLBL)
       call WCC (' T=',1,3,LBL,NLBL+1,NLBL)
       call WCCE8 (E8,2,LBL,NLBL+1,NLBL)
       call WCC (' D=',1,3,LBL,NLBL+1,NLBL)
       call WCI (IRDEC,LBL,NLBL+1,NLBL)
       call WCC ('*',1,1,LBL,NLBL+1,NLBL)
       call WCI (IPDEC,LBL,NLBL+1,NLBL)
       T1 = 0.
       T2 = (NP-1)*DT
       call WCCAVD (IAVD,1,BUF,1,NBUF)
```

```
      call PSGM (DT,Z1,ZN1,NP,IPDEC,T1,T2,
     *           XORG,YORG,XLEN,YLEN,CZ,TZ,
     *           'S',1,BUF,NBUF,.false.)
      X = XORG
      Y = YORG+YLEN+3.0*CZ
      call TEXT (X,Y,CZ,LBL,0.,NLBL)
      X = XORG
      Y = YORG+YLEN+1.0*CZ
      call TEXT (X,Y,0.8*CZ,MOLBL,0.,NMOLBL)
      X = XORG+XLEN-(6+9)*0.8*CZ
      call TEXT (X,Y,0.8*CZ,'PLOT1 ',0.,6)
      call TEXT (999.,999.,0.8*CZ,DATBUF,0.,9)

      call PLOT (0.,0.,-999)

      goto 100

c Errors.
801   stop '*PLOT1 ERROR* Can''t read filelist.'
810   write (6,90810) FN1(1:IEND)
90810 format (' *PLOT1 ERROR* Can''t parse ',a)
      goto 899
811   write (6,90811) FN1(1:IEND)
90811 format (' *PLOT1 ERROR* Can''t open ',a)
      goto 899
812   write (6,90812) FN1(1:IEND)
90812 format (' *PLOT1 ERROR* Can''t read ',a)
      goto 899
899   goto 100

c Finish.
1000  write (6,91000)
91000 format (' End of filelist ...')
      close (unit=LUFL)
      LFL = .false.
      goto 100
1001  call PLOT (0.,0.,+999)

      stop

      end

************************************************************** PLOT3  ***

      program PLOT3

c PLOT3
c Plot 3-component ES&G-block-binary seismograms.
c Can extend filename with auto-rotate azm (VFBB only).

      parameter (ISIZ=32768)
      complex CZW
      dimension Z1(ISIZ),Z2(ISIZ),Z3(ISIZ),ZW(ISIZ),CZW(ISIZ)
      double precision E8CHK
      logical LCD,LFL,LVFBB
      character BUF*80,FN1*80,FN2*80,FN3*80,LBL*80,DATBUF*9,C1*1

      include '[MUELLER.FS.SD]VFCOMMON.FI'

      call DATE (DATBUF)
      call PLOTS (0,0,0)
      LUFL = 10 !logical unit for input filelist (file-of-filenames)
      LUVFBB = 11 !logical unit for VFBB data file

90000 format (' ')
90001 format (' |')

c Setup.
10    write (6,90010)
90010 format ('0PLOT3 setup ...')
      TSKIP = 0.
      TREAD = 10.
      IRDEC = 1
      IPDEC = 2
      LADC = .true.
      LART = .true.
      NMO = 1
      NMOBUF(1) = 1
      MOBUF(1) = 'C'
      CZ = 0.10
      XORG = 1.5
      YORG = 1.2
      XLEN = 8.0
      YLEN = 5.5
c Setup data-input mode.
20    write (6,90020)
90020 format (' '/' Use filelist? Type filelist name [a;<cr>=No]: ',$)
      read (5,90021,err=20) NBUF,BUF
90021 format (q,a)
      if (NBUF.eq.0) then
```

```
        LFL = .false.
      else
        open (unit=LUFL,file=BUF,err=20,status='OLD',readonly)
        LFL = .true.
      endif
c Re-setup @30.
c Setup trace-time,-decimate parameters.
30    LCD = .false.
      write (6,90000)
      call QPL ('Change default TIME,DECIMATE Parameters?',40,LCD)
      if (.not.LCD) goto 39
      write (6,90001)
      call QPF2 ('Skip & Read (sec)',17,TSKIP,TREAD)
      write (6,90001)
      call QPI ('Read decimate',13,IRDEC)
      write (6,90001)
      call QPI ('Plot decimate',13,IPDEC)
39    continue
c Setup trace-modify parameters.
40    LCD = .false.
      write (6,90000)
      call QPL ('Change default MODIFY Parameters?',33,LCD)
      if (.not.LCD) goto 49
      write (6,90001)
      call QPL ('Auto-DC',7,LADC)
      write (6,90001)
      call QPL ('Auto-Rotate',11,LART)
41    write (6,90041)
90041 format (' |'/' Preset Options:'/
     * ' > Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]'/
     * ' > Hipass filter = 2,FC(Hz)[f]'/
     * ' > Lopass filter = 3,FC(Hz)[f]'/
     * ' > Integrate = 4'/
     * ' > Differentiate = 5'/
     * ' > Remove geophone response = 6,NF[i;<cr>=4]'/
     * ' > Pad with temporary zeros = 7,ZT(s)[f]'/
     * ' > Multiply by scale factor = 8,SF[f]'/
     * ' > Continue (disable interactive modify) = C')
42    write (6,90042) NMOMAX
90042 format (' How many options?
     * Type no. [1=<i=<',i1,';<cr>=1,Continue]: ',$)
      read (5,90043,err=42) N,NMO
90043 format (q,i12)
      if (N.eq.0) then
        NMO = 1
        NMOBUF(1) = 1
        MOBUF(1) = 'C'
        goto 49
      endif
      if (NMO.lt.1.or.NMO.gt.NMOMAX) goto 42
      do 46 I=1,NMO
44      write (6,90044) I
90044   format (' Choose option #',i1,' [a]: ',$)
        read (5,90045,err=44) NMOBUF(I),MOBUF(I)
90045   format (q,a)
        C1 = MOBUF(I)(1:1)
        if (C1.eq.'c') C1 = 'C'
        if ((C1.lt.'1'.or.C1.gt.'8').and.C1.ne.'C') goto 44
46      continue
      if (C1.ne.'C') goto 42
49    continue
c Setup trace-plot parameters.
50    LCD = .false.
      write (6,90000)
      call QPL ('Change default PLOT Parameters?',31,LCD)
      if (.not.LCD) goto 59
      write (6,90001)
      call QPF ('CZ (in)',7,CZ)
      call QPF2 ('XORG,YORG (in)',14,XORG,YORG)
      call QPF2 ('XLEN,YLEN (in)',14,XLEN,YLEN)
59    TZ = CZ/3
      write (6,90000)

c Begin main loop.
c Get filename (extend with auto-rotate azm (VFBB only)).
100   if (LFL) then
120     read (LUFL,90120,err=801,end=1000) NBUF,BUF
90120   format (q,a)
        if (NBUF.eq.0.or.BUF(1:1).eq.'|') goto 120
        write (6,90121) BUF(1:NBUF)
90121   format (' ',a)
      else
130     write (6,90130)
90130   format (' '/
     *          ' Type 1st-component filename [a;S=Setup;X=Stop]: ',$)
        read (5,90131,err=130) NBUF,BUF
90131   format (q,a)
        if (BUF(1:2).eq.'S '.or.BUF(1:2).eq.'s ') goto 30
        if (BUF(1:2).eq.'X '.or.BUF(1:2).eq.'x ') goto 1001
      endif
```

```
      if (NBUF.eq.0) goto 140
      IAZMR = -1
      call RCI (2,BUF,NBUF,IAZMR,IR)
      call RCC (1,BUF,NBUF,BUF,NBUF,IR)
      if (LVFBB.and.NBUF.eq.1) then
        FN1(IDT-1:IDT-1) = BUF
      elseif (LVFBB.and.NBUF.eq.3) then
        FN1(IDT+1:IDT+3) = BUF
      elseif (LVFBB.and.NBUF.eq.5) then
        FN1(IDT-1:IDT+3) = BUF
      elseif (LVFBB.and.NBUF.eq.6) then
        FN1(IDT-2:IDT+3) = BUF
      elseif (INDEX(BUF(1:NBUF),']').eq.0) then
        FN1(IRB+1:) = BUF(1:NBUF)//' '
      else
        FN1 = BUF
      endif

c Parse, open, read, and modify. Get Z1.
140   call PARSEF (FN1,IC2,IC1,ILB,IRB,IDT,ISC,IEND,LVFBB,IP)
      if (IP.ne.0) goto 810
      C1 = FN1(IDT-1:IDT-1)
      if (LVFBB.and.C1.ne.'1'.and.C1.ne.'4') goto 100
      open (unit=LUVFBB,file=FN1,err=811,status='OLD',readonly,
     *      access='DIRECT',recl=128)
      call RVFBB (Z1,ZN1,ISIZ)
      close (unit=LUVFBB)
      if (NP.le.0) goto 812
      call CLRSTR (MOLBL)
      IAZMRT = -1
      call MVFBB (Z1,ZW,CZW,ZN1,ISIZ,FN1)
      if (MOLBL(1:9).eq.'StartOver') goto 100
      NP1 = NP
      IVA1 = IVA
      IHA1 = IHA
      DTCHK = DT
      E8CHK = E8
      IAVDCHK = IAVD

c Get 2nd- and 3rd-component filenames.
150   FN2 = FN1
      FN3 = FN1
      if (LVFBB) then
        FN2(IDT-1:IDT-1) = CHAR(ICHAR(FN1(IDT-1:IDT-1))+1)
        FN3(IDT-1:IDT-1) = CHAR(ICHAR(FN1(IDT-1:IDT-1))+2)
      else
151     if (LFL) then
          read (LUFL,90120,err=801,end=1000) NBUF,BUF
        else
152       write (6,90152)
90152     format (' Type 2nd-component filename [a;S=StartOver]: ',$)
          read (5,90131,err=152) NBUF,BUF
          if (BUF(1:2).eq.'S '.or.BUF(1:2).eq.'s ') goto 100
        endif
        if (INDEX(BUF(1:NBUF),']').eq.0) then
          FN2(IRB+1:) = BUF(1:NBUF)//' '
        else
          FN2 = BUF
        endif
153     if (LFL) then
          read (LUFL,90120,err=801,end=1000) NBUF,BUF
        else
154       write (6,90154)
90154     format (' Type 3rd-component filename [a;S=StartOver]: ',$)
          read (5,90131,err=154) NBUF,BUF
          if (BUF(1:2).eq.'S '.or.BUF(1:2).eq.'s ') goto 100
        endif
        if (INDEX(BUF(1:NBUF),']').eq.0) then
          FN3(IRB+1:) = BUF(1:NBUF)//' '
        else
          FN3 = BUF
        endif
      endif

c Open, read, and modify. Get Z2,Z3.
160   open (unit=LUVFBB,file=FN2,err=811,status='OLD',readonly,
     *      access='DIRECT',recl=128)
      call RVFBB (Z2,ZN2,ISIZ)
      close (unit=LUVFBB)
      if (NP.le.0) goto 812
      call CLRSTR (MOLBL)
      if (IAZMR.ge.0) IAZMRT = IAZMR
      call MVFBB (Z2,ZW,CZW,ZN2,ISIZ,FN2)
      if (MOLBL(1:9).eq.'StartOver') goto 100
      NP2 = NP
      IVA2 = IVA
      IHA2 = IHA
      if (DT.ne.DTCHK) goto 813
      if (E8.ne.E8CHK) goto 814
      if (IAVD.ne.IAVDCHK) goto 815
```

```
161   open (unit=LUVFBB,file=FN3,err=811,status='OLD',readonly,
     *      access='DIRECT',recl=128)
      call RVFBB (Z3,ZN3,ISIZ)
      close (unit=LUVFBB)
      if (NP.le.0) goto 812
      call CLRSTR (MOLBL)
      if (IAZMR.ge.0) IAZMRT = IAZMR+90
      call MVFBB (Z3,ZW,CZW,ZN3,ISIZ,FN3)
      if (MOLBL(1:9).eq.'StartOver') goto 100
      NP3 = NP
      IVA3 = IVA
      IHA3 = IHA
      if (DT.ne.DTCHK) goto 813
      if (E8.ne.E8CHK) goto 814
      if (IAVD.ne.IAVDCHK) goto 815
170   NP = MAX(NP1,NP2,NP3)
      do 171 I=NP1+1,NP
171      Z1(I) = ZN
      do 172 I=NP2+1,NP
172      Z2(I) = ZN2
      do 173 I=NP3+1,NP
173      Z3(I) = ZN3

c Get LBL and plot.
      if (LVFBB) then
        call WCC (FN1,IRB+1,IDT-1,LBL,1,NLBL)
        call WCC (',',1,1,LBL,NLBL+1,NLBL)
        call WCC (FN2,IDT-1,IDT-1,LBL,NLBL+1,NLBL)
        call WCC (',',1,1,LBL,NLBL+1,NLBL)
        call WCC (FN3,IDT-1,IDT-1,LBL,NLBL+1,NLBL)
        call WCC (FN1,IDT,ISC-1,LBL,NLBL+1,NLBL)
      else
        call WCC (FN1,IRB+1,ISC-1,LBL,1,NLBL)
        call WCC (',',1,1,LBL,NLBL+1,NLBL)
        call WCC (FN2,IRB+1,ISC-1,LBL,NLBL+1,NLBL)
        call WCC (',',1,1,LBL,NLBL+1,NLBL)
        call WCC (FN3,IRB+1,ISC-1,LBL,NLBL+1,NLBL)
      endif
      call WCC (' T=',1,3,LBL,NLBL+1,NLBL)
      call WCCE8 (E8,2,LBL,NLBL+1,NLBL)
      call WCC (' D=',1,3,LBL,NLBL+1,NLBL)
      call WCI (IRDEC,LBL,NLBL+1,NLBL)
      call WCC ('*',1,1,LBL,NLBL+1,NLBL)
      call WCI (IPDEC,LBL,NLBL+1,NLBL)
      T1 = 0.
      T2 = (NP-1)*DT
      call WCCAVD (IAVD,1,BUF,1,NBUF)
      call PSGM3 (DT,Z1,ZN1,Z2,ZN2,Z3,ZN3,NP,IPDEC,T1,T2,
     *            XORG,YORG,XLEN,YLEN,CZ,TZ,
     *            'S',1,BUF,NBUF,.false.)
      call WCCAZM (IVA1,IHA1,BUF,1,NBUF)
      X = XORG+XLEN-NBUF*CZ
      Y = YORG+(11./12.)*YLEN
      call TEXT (X,Y,CZ,BUF,0.,NBUF)
      call WCCAZM (IVA2,IHA2,BUF,1,NBUF)
      X = XORG+XLEN-NBUF*CZ
      Y = YORG+(7./12.)*YLEN
      call TEXT (X,Y,CZ,BUF,0.,NBUF)
      call WCCAZM (IVA3,IHA3,BUF,1,NBUF)
      X = XORG+XLEN-NBUF*CZ
      Y = YORG+(3./12.)*YLEN
      call TEXT (X,Y,CZ,BUF,0.,NBUF)
      X = XORG
      Y = YORG+YLEN+3.0*CZ
      call TEXT (X,Y,CZ,LBL,0.,NLBL)
      X = XORG
      Y = YORG+YLEN+1.0*CZ
      call TEXT (X,Y,0.8*CZ,MOLBL,0.,NMOLBL)
      X = XORG+XLEN-(6+9)*0.8*CZ
      call TEXT (X,Y,0.8*CZ,'PLOT3 ',0.,6)
      call TEXT (999.,999.,0.8*CZ,DATBUF,0.,9)

      call PLOT (0.,0.,-999)

      goto 100

c Errors.
801   stop '*PLOT3 ERROR* Can''t read filelist.'
810   write (6,90810) FN1(1:IEND)
90810 format (' *PLOT3 ERROR* Can''t parse ',a)
      goto 899
811   write (6,90811) FN1(1:IDT-2),'*',FN1(IDT:IEND)
90811 format (' *PLOT3 ERROR* Can''t open ',a,a,a)
      goto 899
812   write (6,90812) FN1(1:IDT-2),'*',FN1(IDT:IEND)
90812 format (' *PLOT3 ERROR* Can''t read ',a,a,a)
      goto 899
813   write (6,90813) FN1(1:IDT-2),'*',FN1(IDT:IEND)
90813 format (' *PLOT3 ERROR* DT incompatible ',a,a,a)
      goto 899
```

```
814   write (6,90814) FN1(1:IDT-2),'*',FN1(IDT:IEND)
90814 format (' *PLOT3 ERROR* E8 incompatible ',a,a,a)
      goto 899
815   write (6,90815) FN1(1:IDT-2),'*',FN1(IDT:IEND)
90815 format (' *PLOT3 ERROR* IAVD incompatible ',a,a,a)
      goto 899
899   goto 100

c Finish.
1000  write (6,91000)
91000 format (' End of filelist ...')
      close (unit=LUFL)
      LFL = .false.
      goto 100
1001  call PLOT (0.,0.,+999)

      stop

      end

************************************************************ PRS    ***

       program PRS

c PRS
c Plot pseudo-record-section; plot up to NZMAX traces.
c Can extend filename with auto-rotate azm (VFBB only), shift time, plot time.

      parameter (ISIZ=32768,NZMAX=16)
      complex ZC
      dimension Z1(ISIZ),Z2(ISIZ),ZC(ISIZ),
     *          Z(ISIZ,NZMAX),
     *          ZNZ(NZMAX),ZEZ(NZMAX),DTZ(NZMAX),NPZ(NZMAX)
      logical LVFBB
      character BUF*80,FN*80,DATBUF*9,C1*1,
     *          LBL1*80,LBL2*80,LBL3*80,LBL4*80
      dimension LBL1(NZMAX),NLBL1(NZMAX),
     *          LBL2(NZMAX),NLBL2(NZMAX),
     *          LBL3(NZMAX),NLBL3(NZMAX),
     *          LBL4(NZMAX),NLBL4(NZMAX)

      include '[MUELLER.FS.SD]VFCOMMON.FI'

      call PLOTS (0,0,0)
      call DATE (DATBUF)
      LUFL = 10 !logical unit for input filelist (file-of-filenames)
      LUVFBB = 11 !logical unit for VFBB data file

90000 format (' ')
90001 format (' |')

c Setup.
10    write (6,90010)
90010 format ('0PRS setup ...')
      TPLOT = 10.
      IRDEC = 1
      IPDEC = 2
      LADC = .true.
      LART = .true.
      NMO = 1
      NMOBUF(1) = 1
      MOBUF(1) = 'C'
      IPS = 1
      IPV = 1
      IPA = 1
      CZ = 0.10
      XORG = 1.4
      YORG = 1.5
      XLENL = 7.0
      YLENL = 6.2
      XLENP = 4.8
      YLENP = 8.5
      XLEN = XLENL
      YLEN = YLENL
c Setup data-input mode.
20    write (6,90020)
90020 format (' */' Use filelist? Type filelist name [a;<cr>=No]: ',$)
      read (5,90021,err=20) NBUF,BUF
90021 format (q,a)
      if (NBUF.eq.0) then
        LFL = .false.
      else
        open (unit=LUFL,file=BUF,err=20,status='OLD',readonly)
        LFL = .true.
      endif
c Re-setup @30.
c Setup trace-time,-decimate parameters.
30    LCD = .false.
      write (6,90000)
      call QPL ('Change default TIME,DECIMATE Parameters?',40,LCD)
```

```
      if (.not.LCD) goto 39
      write (6,90001)
31    call QPF ('Time to plot (sec)',18,TPLOT)
      if (TPLOT.eq.0.) then
        write (6,90031)
90031   format (' *PRS ERROR* TPLOT=0')
        goto 31
      endif
      write (6,90001)
      call QPI ('Read decimate',13,IRDEC)
      write (6,90001)
      call QPI ('Plot decimate',13,IPDEC)
39    continue
c Setup trace-modify parameters.
40    LCD = .false.
      write (6,90000)
      call QPL ('Change default MODIFY Parameters?',33,LCD)
      if (.not.LCD) goto 49
      write (6,90001)
      call QPL ('Auto-DC',7,LADC)
      write (6,90001)
      call QPL ('Auto-Rotate',11,LART)
41    write (6,90041)
90041 format (' |'/' Preset Options:'/
     * ' > Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]'/
     * ' > Hipass filter = 2,FC(Hz)[f]'/
     * ' > Lopass filter = 3,FC(Hz)[f]'/
     * ' > Integrate = 4'/
     * ' > Differentiate = 5'/
     * ' > Remove geophone response = 6,NF[i;<cr>=4]'/
     * ' > Pad with temporary zeros = 7,ZT(s)[f]'/
     * ' > Multiply by scale factor = 8,SF[f]'/
     * ' > Continue (disable interactive modify) = C')
42    write (6,90042) NMOMAX
90042 format (' How many options?
     * Type no. [0=<i=<',i1,';<cr>=1,Continue]: ',$)
      read (5,90043,err=42) N,NMO
90043 format (q,i12)
      if (N.eq.0) then
        NMO = 1
        NMOBUF(1) = 1
        MOBUF(1) = 'C'
        goto 49
      endif
      if (NMO.lt.0.or.NMO.gt.NMOMAX) goto 42
      do 46 I=1,NMO
44      write (6,90044) I
90044   format (' Choose option #',i1,' [a]: ',$)
        read (5,90045,err=44) NMOBUF(I),MOBUF(I)
90045   format (q,a)
        C1 = MOBUF(I)(1:1)
        if (C1.eq.'c') C1 = 'C'
        if ((C1.lt.'1'.or.C1.gt.'8').and.C1.ne.'C') goto 44
46      continue
49    continue
c Setup trace-plot parameters.
50    LCD = .false.
      write (6,90000)
      call QPL ('Change default PLOT Parameters?',31,LCD)
      if (.not.LCD) goto 59
51    write (6,90051)
90051 format (' |'/
     *        ' > 1 = Equal size'/
     *        ' > 2 = Equal scale (''true amplitude'')')
52    call QPI ('Plot amplitude',14,IPS)
      if (IPS.lt.1.or.IPS.gt.2) goto 52
53    write (6,90053)
90053 format (' |'/
     *        ' > 1 = Landscape view'/
     *        ' > 2 = Portrait view')
54    call QPI ('Plot view',9,IPV)
      if (IPV.eq.1) then
        XLEN = XLENL
        YLEN = YLENL
      elseif (IPV.eq.2) then
        XLEN = XLENP
        YLEN = YLENP
      else
        goto 54
      endif
55    write (6,90055)
90055 format (' |'/
     *        ' > 1 = Full annotation'/
     *        ' > 2 = Sparse annotation')
56    call QPI ('Plot annotation',15,IPA)
      if (IPA.lt.1.or.IPA.gt.2) goto 56
      write (6,90001)
      call QPF ('CZ (in)',7,CZ)
      call QPF2 ('XORG,YORG (in)',14,XORG,YORG)
      call QPF2 ('XLEN,YLEN (in)',14,XLEN,YLEN)
```

```
59    TZ = CZ/3
      write (6,90000)

c Begin main loop.
c Get filename (extend with auto-rotate azm (VFBB only), shift time, plot time).
100   IZ = 0
101   IZ = IZ+1
      if (IZ.gt.NZMAX) goto 200
110   continue
      if (LFL) then
120     read (LUFL,90120,err=801,end=1000) NBUF,BUF
90120   format (q,a)
        if (NBUF.eq.0.or.BUF(1:1).eq.'|') goto 120
        if (BUF(1:2).eq.'P '.or.BUF(1:2).eq.'p '.or.
     *      BUF(1:5).eq.'PLOT '.or.BUF(1:5).eq.'plot ') goto 200
        write (6,90121) BUF(1:NBUF)
90121   format (' ',a)
      else
130     if (IZ.eq.1) write (6,90130)
90130   format (' '/' Type filename [a;S=Setup;X=Stop]: ',$)
        if (IZ.gt.1) write (6,90131)
90131   format (' '/' Type filename [a;P=Plot;X=Stop]: ',$)
        read (5,90132,err=130) NBUF,BUF
90132   format (q,a)
        if (IZ.eq.1.and.
     *      (BUF(1:2).eq.'S '.or.BUF(1:2).eq.'s ')) goto 20
        if (IZ.gt.1.and.
     *      (BUF(1:2).eq.'P '.or.BUF(1:2).eq.'p '.or.
     *       BUF(1:5).eq.'PLOT '.or.BUF(1:5).eq.'plot ')) goto 200
        if (BUF(1:2).eq.'X '.or.BUF(1:2).eq.'x ') goto 1001
      endif
      if (NBUF.eq.0) goto 140
      IAZMRT = -1
      call RCI (2,BUF,NBUF,IAZMRT,IR)
      TSHIFT = 0.
      call RCF (3,BUF,NBUF,TSHIFT,IR)
      TREAD = TPLOT
      call RCF (4,BUF,NBUF,TREAD,IR)
      call RCC (1,BUF,NBUF,BUF,NBUF,IR)
      if (LVFBB.and.NBUF.eq.1) then
        FN(IDT-1:IDT-1) = BUF
      elseif (LVFBB.and.NBUF.eq.3) then
        FN(IDT+1:IDT+3) = BUF
      elseif (LVFBB.and.NBUF.eq.5) then
        FN(IDT-1:IDT+3) = BUF
      elseif (LVFBB.and.NBUF.eq.6) then
        FN(IDT-2:IDT+3) = BUF
      elseif (INDEX(BUF(1:NBUF),']').eq.0) then
        FN(IRB+1:) = BUF(1:NBUF)//' '
      else
        FN = BUF
      endif

c Parse, open, read, and modify.
c TSHIFT>0 for left shift; TSHIFT<0 for right shift.
140   call PARSEF (FN,IC2,IC1,ILB,IRB,IDT,ISC,IEND,LVFBB,IP)
      if (IP.ne.0) goto 810
      open (unit=LUVFBB,file=FN,err=811,
     *      status='OLD',readonly,
     *      access='DIRECT',recl=128)
      TSKIP = MAX(0.,TSHIFT)
      call RVFBB (Z1,Z1NL,ISIZ)
      close (unit=LUVFBB)
      if (NP.le.0) goto 812
      call CLRSTR (MOLBL)
      call MVFBB (Z1,Z2,ZC,Z1NL,ISIZ,FN)
      if (MOLBL(1:9).eq.'StartOver') goto 110
      if (IZ.eq.1) IAVD1 = IAVD
      if (IPS.eq.2.and.IAVD.ne.IAVD1) goto 813

c Shift Z1 and compute Z1EX.
150   Z1EX = 0.
      if (TSHIFT.lt.0.) then
        NPLOT = 1+INT(TPLOT/DT)
        NSHIFT = INT(-TSHIFT/DT)
        NP = MIN(NP,NPLOT-NSHIFT)
        do 151 I=NP,1,-1
           ZI = Z1(I)
           Z1(I+NSHIFT) = ZI
           if (ABS(ZI).gt.ABS(Z1EX).and.ZI.ne.Z1NL) Z1EX = ZI
151        continue
        do 152 I=1,NSHIFT
           Z1(I) = Z1NL
152        continue
        NP = NP+NSHIFT
        E8 = E8-NSHIFT*DT
      else
        do 153 I=1,NP
           ZI = Z1(I)
           if (ABS(ZI).gt.ABS(Z1EX).and.ZI.ne.Z1NL) Z1EX = ZI
153        continue
```

```
      endif

c Save labels.
160   call WCC (FN,IRB+1,ISC-1,LBL1(IZ),1,N)
      call WCC (' ',1,1,LBL1(IZ),N+1,N)
      call WCCAZM (IVA,IHA,LBL1(IZ),N+1,N)
      NLBL1(IZ) = N
      LBL2(IZ) = MOLBL
      NLBL2(IZ) = NMOLBL
      call WCCE8 (E8,2,LBL3(IZ),1,N)
      if (TSHIFT.ne.0.) then
        call WCC (' (',1,2,LBL3(IZ),N+1,N)
        if (TSHIFT.gt.0.) call WCC ('+',1,1,LBL3(IZ),N+1,N)
        call WCT4 (TSHIFT,LBL3(IZ),N+1,N)
        call WCC (')',1,1,LBL3(IZ),N+1,N)
      endif
      call WCC (' ',1,1,LBL3(IZ),N+1,N)
      call WCI (IRDEC,LBL3(IZ),N+1,N)
      call WCC ('*',1,1,LBL3(IZ),N+1,N)
      call WCI (IPDEC,LBL3(IZ),N+1,N)
      NLBL3(IZ) = N
      N = 0
      if (Z1EX.gt.0.) call WCC ('+',1,1,LBL4(IZ),N+1,N)
      call WCF (Z1EX,3,3,LBL4(IZ),N+1,N)
      call WCC (' ',1,1,LBL4(IZ),N+1,N)
      call WCCAVD (IAVD,1,LBL4(IZ),N+1,N)
      NLBL4(IZ) = N

c Save Z,ZNZ,ZEZ,DTZ,NPZ.
170   if (IPS.eq.1) then
        SF = ABS(Z1EX)
        do 171 I=1,NP
           ZI = Z1(I)
           if (ZI.ne.Z1NL) ZI = ZI/SF
           Z(I,IZ) = ZI
171        continue
        Z1EX = Z1EX/SF
      elseif (IPS.eq.2) then
        do 172 I=1,NP
           Z(I,IZ) = Z1(I)
172        continue
      endif
      ZNZ(IZ) = Z1NL
      ZEZ(IZ) = Z1EX
      DTZ(IZ) = DT
      NPZ(IZ) = NP

      goto 101

c Plot.
200   NZ = IZ-1
      T1 = 0.
      T2 = TPLOT
      call LNABS (0,T1,T2,XSCL,
     *            XORG,YORG,XLEN,YLEN,CZ,TZ,'S',1,.true.)
      call WCC ('PRS: ',1,5,BUF,1,NBUF)
      call WCC (DATBUF,1,9,BUF,NBUF+1,NBUF)
      call TEXT (XORG,YORG+YLEN+CZ,CZ,BUF,0.,NBUF)
210   SS = 0.
      do 211 I=1,NZ
         SS = SS+2*ABS(ZEZ(I))
211      continue
      YSCL = (YLEN-2.*CZ)/SS
      if (IPS.eq.2) then
        S1 = 0.
        S2 = (.667)*SS
        IY = LOG10(S2)
        if (IY.gt.LOG10(S2)) IY = IY-1
        X = S2/(10.**IY)
        if (X.lt.1.5) then
          IX = 1
        elseif (X.lt.3.5) then
          IX = 2
        elseif (X.lt.7.5) then
          IX = 5
        else
          IX = 10
        endif
        S2 = IX*(10.**IY)
        YLENS = S2*YSCL
        YORGS = YORG+(YLEN-YLENS)/2.
        call WCCAVD (IAVD,1,BUF,1,NBUF)
        call LNORD (0,S1,S2,YSCL,
     *              XORG-CZ,YORGS,XLEN,YLENS,CZ,TZ,BUF,NBUF,.false.)
      endif
220   XO = XORG
      YO = YORG+YLEN-CZ
      XL = XORG+XLEN+0.5*CZ
      do 221 I=1,NZ
         ZA = ABS(ZEZ(I))
```

```
        YO = YO-YSCL*2*ZA
        call P1 (0.,DTZ(I),Z(1,I),ZNZ(I),IPDEC,1,NPZ(I),
     *           T1,-ZA,XO,YO,XSCL,YSCL)
        if (IPA.eq.1) then
          YL = YO+YSCL*ZA
          call TEXT (XL,YL,CZ,LBL1(I),0.,NLBL1(I))
          YL = YL-CZ
          call TEXT (XL,YL,0.6*CZ,LBL2(I),0.,NLBL2(I))
          YL = YL-CZ
          call TEXT (XL,YL,0.6*CZ,LBL3(I),0.,NLBL3(I))
        elseif (IPA.eq.2) then
          YL = YO+YSCL*ZA-0.5*CZ
          call TEXT (XL,YL,CZ,LBL1(I),0.,NLBL1(I))
        endif
        if (IPS.eq.1) then
          YL = YL-CZ
          call TEXT (XL,YL,0.6*CZ,LBL4(I),0.,NLBL4(I))
        endif
221     continue

      call PLOT (0.,0.,-999)

      goto 100

c Errors.
801   stop '*PRS ERROR* Can''t read filelist.'
810   write (6,90810) FN(1:IEND)
90810 format (' *PRS ERROR* Can''t parse ',a)
      goto 899
811   write (6,90811) FN(1:IEND)
90811 format (' *PRS ERROR* Can''t open ',a)
      goto 899
812   write (6,90812) FN(1:IEND)
90812 format (' *PRS ERROR* Can''t read ',a)
      goto 899
813   write (6,90813) FN(1:IEND)
90813 format (' *PRS ERROR* IAVD incompatible ',a)
      goto 899
899   goto 110

c Finish
1000  write (6,91000)
91000 format (' End of filelist ...')
      close (unit=LUFL)
      LFL = .false.
      goto 100
1001  call PLOT (0.,0.,+999)

      stop

      end

************************************************************* SPY    ***

      program SPY

c SPY
c Compute spectra and spectral ratios of VFBB data.
c Can extend filename with auto-rotate azm (VFBB only).

      parameter (ISIZ=32768)
      complex ZC
      dimension Z1(ISIZ),Z1_(ISIZ),Z2(ISIZ),Z2_(ISIZ),ZP(ISIZ),ZC(ISIZ)
      logical LCD,LFL,LVFBB,
     *        LRATIO,LSAVE,
     *        LMBPO,LSMTH,LCAAF,LCINS,LCATT,LRP,LANNS,LANNPH
      character BUF*80,FNBUF*80,DATBUF*9,C1*1,
     *          FNLBL*80,FNLBL_*80,OPLBL*80,OPLBL_*80

      common /SPEC/ ILL,DF,INYQ,ISPAVD
      common /AVGM/ ARMS,V2,ARMS_,V2_
      common /MBPO/ LMBPO,IMBPO,IMBPO_
      common /SMTH/ LSMTH,SW,SW2,SW_,SW2_
      common /CAAF/ LCAAF,CAAFF,CAAFR,CAAFF_,CAAFR_
      common /CINS/ LCINS,CINSF,CINSD,CINSF_,CINSD_
      common /CATT/ LCATT,Q0,QETA,RQ,VQ,Q0_,QETA_,RQ_,VQ_
      common /PSGMCOM/ TMIN,TMAX,ZMIN,ZMAX,XSCL,YSCL

      include '[MUELLER.FS.SD]VFCOMMON.FI'

      call DATE (DATBUF)
      call PLOTS (0,0,0)
      LUFL = 10 !logical unit for input filelist (file-of-filenames)
      LUVFBB = 11 !logical unit for VFBB data file

90000 format (' ')
90001 format (' |')

c Setup.
10    write (6,90010)
```

```
90010 format ('OSPY setup ...')
      TSKIP = 0.
      TREAD = 10.
      IRDEC = 1
      IPDEC = 2
      LADC = .true.
      LART = .true.
      NMO = 1
      NMOBUF(1) = 1
      MOBUF(1) = 'C'
      IFCT = 10
      IBCT = 10
      LMBPO = .true.
      LSMTH = .true.
      LCAAF = .false.
      LCINS = .false.
      LCATT = .false.
      ILL = 2
      CZ = 0.10
      SXO = 1.5
      SYO = 1.2
      SXL = 8.0
      SPYL = 3.6
      SPYLR = 2.6
      SMYL = 1.5
      SMYLR = 0.8
      LRP = .false.
      LANNS = .true.
      LANNPH = .true.
c Setup data-input mode.
20    write (6,90020)
90020 format (' '/' Use filelist? Type filelist name [a;<cr>=No]: ',$)
      read (5,90021,err=20) NBUF,BUF
90021 format (q,a)
      if (NBUF.eq.0) then
        LFL = .false.
      else
        open (unit=LUFL,file=BUF,err=20,status='OLD',readonly)
        LFL = .true.
      endif
c Re-setup @30.
c Setup trace-time,-decimate parameters.
30    LCD = .false.
      write (6,90000)
      call QPL ('Change default TIME,DECIMATE Parameters?',40,LCD)
      if (.not.LCD) goto 39
      write (6,90001)
      call QPF2 ('Skip,Read (sec)',15,TSKIP,TREAD)
      write (6,90001)
      call QPI ('Read decimate',13,IRDEC)
      write (6,90001)
      call QPI ('Plot decimate',13,IPDEC)
39    continue
c Setup trace-modify parameters.
40    LCD = .false.
      write (6,90000)
      call QPL ('Change default MODIFY Parameters?',33,LCD)
      if (.not.LCD) goto 49
      write (6,90001)
      call QPL ('Auto-DC',7,LADC)
      write (6,90001)
      call QPL ('Auto-Rotate',11,LART)
41    write (6,90041)
90041 format (' |'/' Preset Options:'/
     * ' > Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]'/
     * ' > Hipass filter = 2,FC(Hz)[f]'/
     * ' > Lopass filter = 3,FC(Hz)[f]'/
     * ' > Integrate = 4'/
     * ' > Differentiate = 5'/
     * ' > Remove geophone response = 6,NF[i;<cr>=4]'/
     * ' > Pad with temporary zeros = 7,ZT(s)[f]'/
     * ' > Multiply by scale factor = 8,SF[f]'/
     * ' > Continue (disable interactive modify) = C')
42    write (6,90042) NMOMAX
90042 format (' How many options?
     * Type no. [0=<i=<',i1,';<cr>=1,Continue]: ',$)
      read (5,90043,err=42) N,NMO
90043 format (q,i12)
      if (N.eq.0) then
        NMO = 1
        NMOBUF(1) = 1
        MOBUF(1) = 'C'
        goto 49
      endif
      if (NMO.lt.0.or.NMO.gt.NMOMAX) goto 42
      do 46 I=1,NMO
44      write (6,90044) I
90044   format (' Choose option #',i1,' [a]: ',$)
        read (5,90045,err=44) NMOBUF(I),MOBUF(I)
90045   format (q,a)
```

```
        C1 = MOBUF(I)(1:1)
        if (C1.eq.'c') C1 = 'C'
        if ((C1.lt.'1'.or.C1.gt.'8').and.C1.ne.'C') goto 44
46      continue
49    continue
c Setup trace&spectrum process parameters.
50    LCD = .false.
      write (6,90000)
      call QPL ('Change default PROCESS Parameters?',34,LCD)
      if (.not.LCD) goto 59
      write (6,90001)
      call QPI2 ('Taper front,back (%)',20,IFCT,IBCT)
51    write (6,90051)
90051 format (' |'/
     *        ' > 1 = linear vs linear'/
     *        ' > 2 = log vs log'/
     *        ' > 3 = linear vs log'/
     *        ' > 4 = log vs linear')
52    call QPI ('Spectrum style',14,ILL)
      if (ILL.lt.1.or.ILL.gt.4) goto 52
53    write (6,90053)
90053 format (' |'/' Enable (Y) or Disable (N) Interactive Processing:')
      call QPL ('Multiply by power-of-omega?',27,LMBPO)
      call QPL ('Smooth?',7,LSMTH)
      call QPL ('A-A Filter correction?',22,LCAAF)
      call QPL ('Instrument correction?',22,LCINS)
      call QPL ('Attenuation correction?',23,LCATT)
59    continue
c Setup trace&spectrum plot parameters.
60    LCD = .false.
      write (6,90000)
      call QPL ('Change default PLOT Parameters?',31,LCD)
      if (.not.LCD) goto 69
      write (6,90001)
      call QPF ('CZ (in)',7,CZ)
      call QPF2 ('SXO,SYO (in)',12,SXO,SYO)
      call QPF ('SXL (in)',8,SXL)
      call QPF2 ('SPYL,SPYLR (in)',15,SPYL,SPYLR)
      call QPF2 ('SMYL,SMYLR (in)',15,SMYL,SMYLR)
      write (6,90001)
      call QPL ('Enable (Y) or Disable (N): Replotting?',38,LRP)
69    TZ = CZ/3
      SMYO = SYO+SPYL+5.5*CZ
      SMYORD = SYO+SPYLR+5.5*CZ
      SMYORN = SMYORD+SMYLR+7.0*CZ
c Setup trace&spectrum annotation parameters.
70    LCD = .false.
      write (6,90000)
      call QPL ('Change default ANNOTATE Parameters?',35,LCD)
      if (.not.LCD) goto 79
      write (6,90001)
      call QPL ('Annotate SPY label?',19,LANNS)
      call QPL ('Annotate processing history?',28,LANNPH)
79    continue
      write (6,90000)

c Begin main loop.
c Get filename (extend with auto-rotate azm (VFBB only)).
100   if (LFL) then
120     read (LUFL,90120,err=801,end=1000) NBUF,BUF
90120   format (q,a)
        if (NBUF.eq.0.or.BUF(1:1).eq.'|') goto 120
        write (6,90121) BUF(1:NBUF)
90121   format (' ',a)
122     write (6,90122)
90122   format (' Type <cr>=OK [N=Next;P=Previous;S=Setup;X=Stop]: ',$)
        read (5,90123,err=122) C1
90123   format (a)
        if (C1.eq.'N'.or.C1.eq.'n') goto 120
        if (C1.eq.'P'.or.C1.eq.'p') then
          backspace (LUFL)
          backspace (LUFL)
          goto 120
        endif
        if (C1.eq.'S'.or.C1.eq.'s') goto 20
        if (C1.eq.'X'.or.C1.eq.'x') goto 1001
      else
130     write (6,90130)
90130   format (' '/' Type filename [a;S=Setup;X=Stop]: ',$)
        read (5,90131,err=130) NBUF,BUF
90131   format (q,a)
        if (BUF(1:2).eq.'S '.or.BUF(1:2).eq.'s ') goto 20
        if (BUF(1:2).eq.'X '.or.BUF(1:2).eq.'x ') goto 1001
      endif
      if (NBUF.eq.0) goto 140
      IAZMRT = -1
      call RCI (2,BUF,NBUF,IAZMRT,IR)
      call RCC (1,BUF,NBUF,BUF,NBUF,IR)
      if (LVFBB.and.NBUF.eq.1) then
        FNBUF(IDT-1:IDT-1) = BUF
```

```
      elseif (LVFBB.and.NBUF.eq.3) then
        FNBUF(IDT+1:IDT+3) = BUF
      elseif (LVFBB.and.NBUF.eq.5) then
        FNBUF(IDT-1:IDT+3) = BUF
      elseif (LVFBB.and.NBUF.eq.6) then
        FNBUF(IDT-2:IDT+3) = BUF
      elseif (INDEX(BUF(1:NBUF),']').eq.0) then
        FNBUF(IRB+1:) = BUF(1:NBUF)//' '
      else
        FNBUF = BUF
      endif

c Parse, open, read, and modify. Get seismogram Z1.
140   call PARSEF (FNBUF,IC2,IC1,ILB,IRB,IDT,ISC,IEND,LVFBB,IP)
      if (IP.ne.0) goto 810
      open (unit=LUVFBB,file=FNBUF,err=811,
     *      status='OLD',readonly,
     *      access='DIRECT',recl=128)
      call RVFBB (Z1,Z1NL,ISIZ)
      close (unit=LUVFBB)
      if (NP.le.0) goto 812
      call CLRSTR (MOLBL)
141   call MVFBB (Z1,Z2,ZC,Z1NL,ISIZ,FNBUF)
      if (MOLBL(1:9).eq.'StartOver') goto 100
      OPLBL = MOLBL
      NOPLBL = NMOLBL

c Get FNLBL and plot.
      call CLRSTR (FNLBL)
      call WCC (FNBUF,IRB+1,ISC-1,FNLBL,1,NFNLBL)
      call WCC (' ',1,1,FNLBL,NFNLBL+1,NFNLBL)
      call WCCAZM (IVA,IHA,FNLBL,NFNLBL+1,NFNLBL)
      call WCC (' T=',1,3,FNLBL,NFNLBL+1,NFNLBL)
      call WCCE8 (E8,2,FNLBL,NFNLBL+1,NFNLBL)
      call WCC (' D=',1,3,FNLBL,NFNLBL+1,NFNLBL)
      call WCI (IRDEC,FNLBL,NFNLBL+1,NFNLBL)
      call WCC ('*',1,1,FNLBL,NFNLBL+1,NFNLBL)
      call WCI (IPDEC,FNLBL,NFNLBL+1,NFNLBL)
      T1 = 0.
      T2 = (NP-1)*DT
      call WCCAVD (IAVD,1,BUF,1,NBUF)
      call PSGM (DT,Z1,Z1NL,NP,IPDEC,T1,T2,
     *           1.5,1.2,8.0,5.5,CZ,TZ,
     *           'S',1,BUF,NBUF,.false.)
      call TEXT (1.5,7.00,CZ,FNLBL,0.,NFNLBL)
      call TEXT (1.5,6.80,0.8*CZ,OPLBL,0.,NOPLBL)

c Get FFT window with cursor and compute spectrum Z2.
      call CURSOR (X1,Y1)
      call CURSOR (X2,Y2)
      call PLOT (0.,0.,-999)
      if (X1.eq.X2) goto 141 !"double-click" to get MVFBB again
      TW1 = AMAX1((AMIN1(X1,X2)-1.5)/XSCL,0.)
      TW2 = AMIN1((AMAX1(X1,X2)-1.5)/XSCL,(NP-1)*DT)
      ISPAVD = IAVD
      CINSF = TFRQ
      CINSD = TDMP
      CAAFF = AAFC
      CAAFR = AAFR
      call SPYS (DT,Z1,Z2,ZC,Z1NL,IFCT,IBCT,TW1,TW2)

c Get LRATIO and LSAVE.
      write (6,90000)
      LRATIO = .false.
      if (DF.eq.DF_.and.INYQ.eq.INYQ_)
     * call QPL ('Compute spectral ratio?',23,LRATIO)
      LSAVE = .false.
      call QPL ('Save (denominator) for spectral ratio?',38,LSAVE)

c Load ZP and setup spectrum/seismogram plot.
200   if (LRATIO) then
        do 201 I=1,INYQ
201       ZP(I) = Z2(I)/Z2_(I)
      else
        do 202 I=1,INYQ
202       ZP(I) = Z2(I)
      endif
210   IPM = 1
      write (6,90210)
90210 format ('0> 1 = automatic'/
     *        ' > 2 = customize'/
     *        ' > 3 = no plot')
211   call QPI ('Plot mode',9,IPM)
      if (IPM.lt.1.or.IPM.gt.3) goto 211
      if (IPM.eq.3) goto 500
220   FFLR = DF
      FCEL = (INYQ-1-1)*DF
221   FAX1 = FFLR
      FAX2 = FCEL
      if (IPM.eq.2) call QPF2 ('Freq axis min,max (Hz)',22,FAX1,FAX2)
```

```
      if (FAX2.le.FAX1) goto 221
      if ((ILL.eq.2.or.ILL.eq.3).and.FAX1.le.0.) goto 221
222   F1 = AMAX1(FAX1,FFLR)
      F2 = AMIN1(FAX2,FCEL)
      if (IPM.eq.2) call QPF2 ('Freq data min,max (Hz)',22,F1,F2)
      if (F1.lt.AMAX1(FAX1,FFLR)) goto 222
      if (F2.gt.AMIN1(FAX2,FCEL)) goto 222
      if (F2.le.F1) goto 222
      if ((ILL.eq.2.or.ILL.eq.3).and.F1.le.0.) goto 222
230   IF1 = NINT(F1/DF)+1
      if ((IF1-1)*DF.lt.F1) IF1 = IF1+1
      IF2 = NINT(F2/DF)+1
      if ((IF2-1)*DF.gt.F2) IF2 = IF2-1
240   ZPMIN = ZP(IF1)
      ZPMAX = ZPMIN
      do 241 I=IF1,IF2
         ZPI = ZP(I)
         if (ZPI.lt.ZPMIN) ZPMIN = ZPI
         if (ZPI.gt.ZPMAX) ZPMAX = ZPI
241      continue
250   ZP1 = ZPMIN
      ZP2 = ZPMAX
      if (LRATIO.and.(ILL.eq.2.or.ILL.eq.4)) then
        if (ZP1.gt.0.1) ZP1 = 0.1
        if (ZP2.lt.10.) ZP2 = 10.
      endif
      if (IPM.eq.2) call QPF2 ('Ampl axis min,max',17,ZP1,ZP2)
      if (ZP2.le.ZP1) goto 250
      if ((ILL.eq.2.or.ILL.eq.4).and.ZP1.le.0.) goto 250

c If .not.LRATIO, plot spectrum and one windowed seismogram.
      if (.not.LRATIO) then
      call WCCAVD (ISPAVD,2,BUF,1,N)
      call PSPC (DF,ZP,INYQ,FAX1,FAX2,F1,F2,.true.,ZP1,ZP2,ILL,
     *           SXO,SYO,SXL,SPYL,CZ,TZ,
     *           'HZ',2,BUF,N,.true.)
      if (LANNS) then
        call TEXT (SXO+SXL+0.8*CZ,SYO+SPYL,0.8*CZ,'SPY ',270.,4)
        call TEXT (999.,999.,0.8*CZ,DATBUF,270.,9)
      endif
      T1 = 0.
      T2 = (NP-1)*DT
      call WCCAVD (IAVD,1,BUF,1,N)
      call PSGM (DT,Z1,Z1NL,NP,IPDEC,T1,T2,
     *           SXO,SMYO,SXL,SMYL,CZ,TZ,
     *           'S',1,BUF,N,.false.)
      call SPYW (DT,Z1,Z1NL,IFCT,IBCT,TW1,TW2,SXO,SMYO,CZ)
      call TEXT (SXO,SMYO+SMYL+2.0*CZ,CZ,FNLBL,0.,NFNLBL)
      call TEXT (SXO,SMYO+SMYL+0.5*CZ,0.8*CZ,OPLBL,0.,NOPLBL)
      endif

c If LRATIO, plot spectral ratio and two windowed seismograms.
      if (LRATIO) then
      call WCCAVD (-1,2,BUF,1,N)
      call PSPC (DF,ZP,INYQ,FAX1,FAX2,F1,F2,.true.,ZP1,ZP2,ILL,
     *           SXO,SYO,SXL,SPYLR,CZ,TZ,
     *           'HZ',2,BUF,N,.true.)
      if (LANNS) then
        call TEXT (SXO+SXL+0.8*CZ,SYO+SPYLR,0.8*CZ,'SPY ',270.,4)
        call TEXT (999.,999.,0.8*CZ,DATBUF,270.,9)
      endif
      T1 = 0.
      T2 = (NP_-1)*DT_
      call WCCAVD (IAVD_,1,BUF,1,N)
      call PSGM (DT_,Z1_,Z1NL_,NP_,IPDEC,T1,T2,
     *           SXO,SMYORD,SXL,SMYLR,CZ,TZ,
     *           'S',1,BUF,N,.false.)
      call SPYW (DT_,Z1_,Z1NL_,IFCT_,IBCT_,TW1_,TW2_,SXO,SMYORD,CZ)
      call TEXT (SXO,SMYORD+SMYLR+2.0*CZ,CZ,FNLBL_,0.,NFNLBL_)
      call TEXT (SXO,SMYORD+SMYLR+0.5*CZ,0.8*CZ,OPLBL_,0.,NOPLBL_)
      T1 = 0.
      T2 = (NP-1)*DT
      call WCCAVD (IAVD,1,BUF,1,N)
      call PSGM (DT,Z1,Z1NL,NP,IPDEC,T1,T2,
     *           SXO,SMYORN,SXL,SMYLR,CZ,TZ,
     *           ' ',1,BUF,N,.false.)
      call SPYW (DT,Z1,Z1NL,IFCT,IBCT,TW1,TW2,SXO,SMYORN,CZ)
      call TEXT (SXO,SMYORN+SMYLR+2.0*CZ,CZ,FNLBL,0.,NFNLBL)
      call TEXT (SXO,SMYORN+SMYLR+0.5*CZ,0.8*CZ,OPLBL,0.,NOPLBL)
      endif

c Annotate.
      if (LANNPH) then
        call CURSOR (XC,YC)
        SYL = SPYL
        if (LRATIO) SYL = SPYLR
        if ((XC.ge.SXO.and.XC.le.SXO+SXL).and.
     *      (YC.ge.SYO.and.YC.le.SYO+SYL)) call SPYA (XC,YC,CZ,LRATIO)
      endif
```

```
      call PLOT (0.,0.,-999)

c Replot.
290   if (.not.LRP) goto 299
291   write (6,90291)
90291 format (' '/' Type <cr>=OK [R=Replot]: ',$)
      read (5,90292,err=291) C1
90292 format (a)
      IPM = 2
      if (C1.eq.'R'.or.C1.eq.'r') goto 220
299   continue

c Save a denominator.
500   if (.not.LSAVE) goto 509
      do 501 I=1,NP
501      Z1_(I) = Z1(I)
      Z1NL_ = Z1NL
      do 502 I=1,INYQ
502      Z2_(I) = Z2(I)
      FNLBL_ = FNLBL
      NFNLBL_ = NFNLBL
      OPLBL_ = OPLBL
      NOPLBL_ = NOPLBL
      DT_ = DT
      NP_ = NP
      IAVD_ = IAVD
      DF_ = DF
      INYQ_ = INYQ
      IFCT_ = IFCT
      IBCT_ = IBCT
      TW1_ = TW1
      TW2_ = TW2
      ARMS_ = ARMS
      V2_ = V2
      IMBPO_ = IMBPO
      SW_ = SW
      SW2_ = SW2
      CAAFF_ = CAAFF
      CAAFR_ = CAAFR
      CINSF_ = CINSF
      CINSD_ = CINSD
      Q0_ = Q0
      QETA_ = QETA
      RQ_ = RQ
      VQ_ = VQ
509   continue

c End main loop.
      goto 100

c Errors.
801   stop '*SPY ERROR* Can''t read filelist.'
810   write (6,90810) FNBUF(1:IEND)
90810 format (' *SPY ERROR* Can''t parse ',a)
      goto 899
811   write (6,90811) FNBUF(1:IEND)
90811 format (' *SPY ERROR* Can''t open ',a)
      goto 899
812   write (6,90812) FNBUF(1:IEND)
90812 format (' *SPY ERROR* Can''t read ',a)
      goto 899
899   goto 100

c Finish.
1000  write (6,91000)
91000 format (' End of filelist ...')
      close (unit=LUFL)
      LFL = .false.
      goto 100
1001  call PLOT (0.,0.,+999)

      stop

      end

      subroutine SPYA (XC,YC,CSIZE,LRATIO)

c Annotate spectrum plot at (XC,YC).

      logical LRATIO,LMBPO,LSMTH,LAAF,LINS,LATT
      character BUF*80

      common /AVGM/ ARMS,V2,ARMS_,V2_
      common /MBPO/ LMBPO,IMBPO,IMBPO_
      common /SMTH/ LSMTH,SW,SW2,SW_,SW2_
      common /CAAF/ LAAF,CAAFF,CAAFR,CAAFF_,CAAFR_
      common /CINS/ LINS,CINSF,CINSD,CINSF_,CINSD_
      common /CATT/ LATT,Q0,QETA,RQ,VQ,Q0_,QETA_,RQ_,VQ_

      C = 0.8*CSIZE
      YSKIP1 = 2.0*C
```

```
      YSKIP2 = 1.6*C

      X = XC
      Y = YC-C

10    if (ARMS.ne.0..or.(LRATIO.and.ARMS_.ne.0.)) then
        call WCC ('ARMS = ',1,7,BUF,1,NBUF)
        if (ARMS.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (ARMS.ne.0.) call WCF (ARMS,3,1,BUF,NBUF+1,NBUF)
        if (.not.LRATIO) goto 11
        call WCC ('/',1,1,BUF,NBUF+1,NBUF)
        if (ARMS_.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (ARMS_.ne.0.) call WCF (ARMS_,3,1,BUF,NBUF+1,NBUF)
11      call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP1
      endif

20    if (V2.ne.0..or.(LRATIO.and.V2_.ne.0.)) then
        call WCC ('V**2 = ',1,7,BUF,1,NBUF)
        if (V2.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (V2.ne.0.) call WCF (V2,3,1,BUF,NBUF+1,NBUF)
        if (.not.LRATIO) goto 21
        call WCC ('/',1,1,BUF,NBUF+1,NBUF)
        if (V2_.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (V2_.ne.0.) call WCF (V2_,3,1,BUF,NBUF+1,NBUF)
21      call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP1
      endif

30    if (IMBPO.ne.0.or.(LRATIO.and.IMBPO_.ne.0)) then
        call WCC ('Multiply = ',1,11,BUF,1,NBUF)
        if (IMBPO.eq.0) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (IMBPO.ne.0) then
          call WCC ('Omega**',1,7,BUF,NBUF+1,NBUF)
          call WCI (IMBPO,BUF,NBUF+1,NBUF)
        endif
        if (.not.LRATIO) goto 31
        call WCC ('/',1,1,BUF,NBUF+1,NBUF)
        if (IMBPO_.eq.0) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (IMBPO_.ne.0) then
          call WCC ('Omega**',1,7,BUF,NBUF+1,NBUF)
          call WCI (IMBPO_,BUF,NBUF+1,NBUF)
        endif
31      call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP1
      endif

40    if (SW2.ne.0..or.(LRATIO.and.SW2_.ne.0.)) then
        call WCC ('Smooth(Hz) = ',1,13,BUF,1,NBUF)
        if (SW2.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (SW2.ne.0.) call WCF (SW2,3,1,BUF,NBUF+1,NBUF)
        if (.not.LRATIO) goto 41
        call WCC ('/',1,1,BUF,NBUF+1,NBUF)
        if (SW2_.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (SW2_.ne.0.) call WCF (SW2_,3,1,BUF,NBUF+1,NBUF)
41      call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP1
      endif

50    if (CAAFF.ne.0..or.(LRATIO.and.CAAFF_.ne.0.)) then
        call WCC ('A-A Filter Correction:',1,22,BUF,1,NBUF)
        call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP2
        call WCC (' CFq(Hz) = ',1,11,BUF,1,NBUF)
        if (CAAFF.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (CAAFF.ne.0.) call WCF (CAAFF,3,1,BUF,NBUF+1,NBUF)
        if (.not.LRATIO) goto 51
        call WCC ('/',1,1,BUF,NBUF+1,NBUF)
        if (CAAFF_.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (CAAFF_.ne.0.) call WCF (CAAFF_,3,1,BUF,NBUF+1,NBUF)
51      call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP2
        call WCC (' RO(dB/oct) = ',1,14,BUF,1,NBUF)
        if (CAAFF.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (CAAFF.ne.0.) call WCF (CAAFR,3,1,BUF,NBUF+1,NBUF)
        if (.not.LRATIO) goto 52
        call WCC ('/',1,1,BUF,NBUF+1,NBUF)
        if (CAAFF_.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (CAAFF_.ne.0.) call WCF (CAAFR_,3,1,BUF,NBUF+1,NBUF)
52      call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP1
      endif

60    if (CINSF.ne.0..or.(LRATIO.and.CINSF_.ne.0.)) then
        call WCC ('Instrument Correction:',1,22,BUF,1,NBUF)
        call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP2
        call WCC (' NFq(Hz) = ',1,11,BUF,1,NBUF)
        if (CINSF.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (CINSF.ne.0.) call WCF (CINSF,3,1,BUF,NBUF+1,NBUF)
```

```
        if (.not.LRATIO) goto 61
        call WCC ('/',1,1,BUF,NBUF+1,NBUF)
        if (CINSF_.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (CINSF_.ne.0.) call WCF (CINSF_,3,1,BUF,NBUF+1,NBUF)
61      call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP2
        call WCC (' Damp(frct) = ',1,14,BUF,1,NBUF)
        if (CINSF.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (CINSF.ne.0.) call WCF (CINSD,3,1,BUF,NBUF+1,NBUF)
        if (.not.LRATIO) goto 62
        call WCC ('/',1,1,BUF,NBUF+1,NBUF)
        if (CINSF_.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (CINSF_.ne.0.) call WCF (CINSD_,3,1,BUF,NBUF+1,NBUF)
62      call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP1
      endif

70    if (Q0.ne.0..or.(LRATIO.and.Q0_.ne.0.)) then
        call WCC ('Attenuation Correction:',1,23,BUF,1,NBUF)
        call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP2
        call WCC (' Q = ',1,5,BUF,1,NBUF)
        if (Q0.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (Q0.ne.0.) then
          call WCF (Q0,3,1,BUF,NBUF+1,NBUF)
          if (QETA.ne.0.) then
            call WCC (',',1,1,BUF,NBUF+1,NBUF)
            call WCF (QETA,3,1,BUF,NBUF+1,NBUF)
          endif
        endif
        if (.not.LRATIO) goto 71
        call WCC ('/',1,1,BUF,NBUF+1,NBUF)
        if (Q0_.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (Q0_.ne.0.) then
          call WCF (Q0_,3,1,BUF,NBUF+1,NBUF)
          if (QETA_.ne.0.) then
            call WCC (',',1,1,BUF,NBUF+1,NBUF)
            call WCF (QETA_,3,1,BUF,NBUF+1,NBUF)
          endif
        endif
71      call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP2
        call WCC (' R(km) = ',1,9,BUF,1,NBUF)
        if (Q0.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (Q0.ne.0.) call WCF (RQ,3,1,BUF,NBUF+1,NBUF)
        if (.not.LRATIO) goto 72
        call WCC ('/',1,1,BUF,NBUF+1,NBUF)
        if (Q0_.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (Q0_.ne.0.) call WCF (RQ_,3,1,BUF,NBUF+1,NBUF)
72      call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP2
        call WCC (' V(km/s) = ',1,11,BUF,1,NBUF)
        if (Q0.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (Q0.ne.0.) call WCF (VQ,3,1,BUF,NBUF+1,NBUF)
        if (.not.LRATIO) goto 73
        call WCC ('/',1,1,BUF,NBUF+1,NBUF)
        if (Q0_.eq.0.) call WCC ('-',1,1,BUF,NBUF+1,NBUF)
        if (Q0_.ne.0.) call WCF (VQ_,3,1,BUF,NBUF+1,NBUF)
73      call TEXT (X,Y,C,BUF,0.,NBUF)
        Y = Y-YSKIP1
      endif

      return

      end

      subroutine SPYS (DT,Z1,Z2,ZC,ZNULL,IFCT,IBCT,TW1,TW2)

c Compute Fourier amplitude spectrum of windowed seismogram Z1.
c Use Aki-Richards sign convention.
c Scale FORK output by DT*SQRT(IPW2).

      complex ZC
      dimension Z1(1),Z2(1),ZC(1)
      logical LMBPO,LSMTH,LAAF,LINS,LATT
      character BUF*80,BUF2*80

      common /SPEC/ LL,DF,INYQ,ISPAVD
      common /AVGM/ ARMS,V2,ARMS_,V2_
      common /MBPO/ LMBPO,IMBPO,IMBPO_
      common /SMTH/ LSMTH,SW,SW2,SW_,SW2_
      common /CAAF/ LAAF,CAAFF,CAAFR,CAAFF_,CAAFR_
      common /CINS/ LINS,CINSF,CINSD,CINSF_,CINSD_
      common /CATT/ LATT,Q0,QETA,RQ,VQ,Q0_,QETA_,RQ_,VQ_

      PI = 4.*ATAN(1.)
90000 format (q,a)

c Put windowed part of Z1 in Z2. Recompute TW1,TW2.
      I1 = NINT(TW1/DT)+1
```

```
      TW1 = (I1-1)*DT
      I2 = NINT(TW2/DT)+1
      TW2 = (I2-1)*DT
      do 1 I=I1,I2
1        Z2(I-I1+1) = Z1(I)
      IPTS = I2-I1+1
c If acceleration from acc. transducer, compute ARMS in the window.
c See Hanks(1979), equation (12).
      ARMS = 0.
      if (ISPAVD.eq.1) then
        SQA = 0.
        do 2 I=1,IPTS
2          SQA = SQA+Z2(I)**2
        ARMS = SQRT(SQA/IPTS)
      endif
c If velocity, compute cumulative V**2 in the window.
      V2 = 0.
      if (ISPAVD.eq.2) then
        do 3 I=1,IPTS
3          V2 = V2+Z2(I)*Z2(I)
        V2 = V2*DT
      endif
c Window and pad.
      call FBCTPR (Z2,ZNULL,IPTS,DT,IFCT,IBCT)
      call PADPW2 (Z2,ZNULL,IPTS,DT)
      IPW2 = IPTS
c Compute amplitude spectrum.
      do 4 I=1,IPW2
4        ZC(I) = CMPLX(Z2(I),0.)
      call FORK (IPW2,ZC,+1.)
      DF = 1./(IPW2*DT)
      INYQ = IPW2/2+1
      SFACT = DT*SQRT(FLOAT(IPW2))
      do 5 I=1,INYQ
         ZC(I) = SFACT*ZC(I)
5        Z2(I) = CABS(ZC(I))
      call WCI (INYQ-1,BUF,1,NBUF)
      call WCC (' positive frequencies in FFT',1,28,BUF,NBUF+1,NBUF)
6     write (6,90006) BUF(1:NBUF)
90006 format ('0',a)

c Multiply spectrum by integer power-of-omega?
10    I = 0
      if (.not.IMBPO) goto 13
      call WCI (IMBPO,BUF,1,NBUF)
11    write (6,90011) BUF(1:NBUF)
90011 format ('0Multiply spectrum by Omega**I? I = ',a/
     *        ' Type new value [i;<cr>=OK;S=Skip]: ',$)
      read (6,90000,err=11) NBUF2,BUF2
      if (BUF2(1:2).eq.'S '.or.BUF2(1:2).eq.'s ') goto 13
      I = IMBPO
      if (NBUF2.eq.0) goto 13
      call RCI (1,BUF2,NBUF2,I,IR)
      if (IR.ne.0) goto 11
13    IMBPO = I
      if (IMBPO.eq.0) goto 19
      ISPAVD = ISPAVD-IMBPO
      do 14 I=2,INYQ
         F = (I-1)*DF
         W = 2*PI*F
         Z2(I) = Z2(I)*(W**IMBPO)
14       continue
19    continue

c Smooth spectrum?
c Number of points in the smoothing triangle base = 2*ISW+1;
c the two outside points get zero weight, so ISW<=1 implies no smooth.
20    F = 0.
      if (.not.LSMTH) goto 23
      call WCF (SW,3,1,BUF,1,NBUF)
21    write (6,90021) BUF(1:NBUF)
90021 format ('0Smooth spectrum? Smooth(Hz) = 'a/
     *        ' Type new value [f;<cr>=OK;S=Skip]: ',$)
      read (5,90000) NBUF2,BUF2
      if (BUF2(1:2).eq.'S '.or.BUF2(1:2).eq.'s ') goto 23
      F = SW
      if (NBUF2.eq.0) goto 23
      call RCF (1,BUF2,NBUF2,F,IR)
      if (IR.ne.0.or.(F.lt.0.)) goto 21
23    SW = F
      ISW = NINT(SW/DF)
      if (ISW.eq.1) ISW = 0
      SW2 = ISW*DF
      if (LL.eq.1.or.LL.eq.3) call SMTHS (Z2,IPW2,DF,ISW,1)
      if (LL.eq.2.or.LL.eq.4) call SMTHS (Z2,IPW2,DF,ISW,2)
29    continue

c Anti-alias filter correction?
30    F1 = 0.
      F2 = 0.
```

```
      if (.not.LAAF) goto 33
      call WCC (' CornerFq(Hz),Rolloff(dB/oct) = ',1,32,BUF,1,NBUF)
      call WCF (CAAFF,3,1,BUF,NBUF+1,NBUF)
      call WCC (',',1,1,BUF,NBUF+1,NBUF)
      call WCF (CAAFR,3,1,BUF,NBUF+1,NBUF)
31    write (6,90031) BUF(1:NBUF)
90031 format ('0A-A Filter correction?'/a/
     *       ' Type new values [2f;<cr>=OK;S=Skip]: ',$)
      read (5,90000) NBUF2,BUF2
      if (BUF2(1:2).eq.'S '.or.BUF2(1:2).eq.'s ') goto 33
      F1 = CAAFF
      F2 = CAAFR
      if (NBUF2.eq.0) goto 33
      call RCF (1,BUF2,NBUF2,F1,IR)
      if (IR.ne.0.or.(F1.lt.0.)) goto 31
      call RCF (2,BUF2,NBUF2,F2,IR)
      if (IR.ne.0.or.(F2.lt.0.)) goto 31
33    CAAFF = F1
      CAAFR = F2
      if (CAAFF.eq.0.) goto 39
      call CORAAF (Z2,IPW2,DF,CAAFF,CAAFR)
39    continue

c Instrument correction?
40    F1 = 0.
      F2 = 0.
      if (.not.LINS) goto 43
      call WCC (' NaturalFq(Hz),Damping(fraction) = ',1,35,BUF,1,NBUF)
      call WCF (CINSF,3,1,BUF,NBUF+1,NBUF)
      call WCC (',',1,1,BUF,NBUF+1,NBUF)
      call WCF (CINSD,3,1,BUF,NBUF+1,NBUF)
41    write (6,90041) BUF(1:NBUF)
90041 format ('0Instrument correction?'/a/
     *       ' Type new values [2f;<cr>=OK;S=Skip]: ',$)
      read (5,90000) NBUF2,BUF2
      if (BUF2(1:2).eq.'S '.or.BUF2(1:2).eq.'s ') goto 43
      F1 = CINSF
      F2 = CINSD
      if (NBUF2.eq.0) goto 43
      call RCF (1,BUF2,NBUF2,F1,IR)
      if (IR.ne.0.or.(F1.lt.0.)) goto 41
      call RCF (2,BUF2,NBUF2,F2,IR)
      if (IR.ne.0.or.(F2.lt.0.)) goto 41
43    CINSF = F1
      CINSD = F2
      if (CINSF.eq.0.) goto 49
      call CORINS (Z2,IPW2,DF,CINSF,CINSD)
49    continue

c Attenuation correction?
50    F1 = 0.
      F2 = 0.
      if (.not.LATT) goto 53
      call WCC (' Q0,QETA = ',1,11,BUF,1,NBUF)
      call WCF (Q0,3,1,BUF,NBUF+1,NBUF)
      call WCC (',',1,1,BUF,NBUF+1,NBUF)
      call WCF (QETA,3,1,BUF,NBUF+1,NBUF)
51    write (6,90051) BUF(1:NBUF)
90051 format ('0Attenuation correction?'/a/
     *       ' Type new values [2f;<cr>=OK;S=Skip]: ',$)
      read (5,90000) NBUF2,BUF2
      if (BUF2(1:2).eq.'S '.or.BUF2(1:2).eq.'s ') goto 53
      F1 = Q0
      F2 = QETA
      if (NBUF2.eq.0) goto 53
      call RCF (1,BUF2,NBUF2,F1,IR)
      if (IR.ne.0.or.(F1.lt.0.)) goto 51
      call RCF (2,BUF2,NBUF2,F2,IR)
      if (IR.ne.0) goto 51
53    Q0 = F1
      QETA = F2
      if (Q0.eq.0.) goto 59
      call QPF ('Distance(km)',12,RQ)
      call QPF ('Velocity(km/s)',14,VQ)
      call CORATT (Z2,IPW2,DF,Q0,QETA,RQ,VQ)
59    continue

c Ensure no zeros in spectrum.
      call SPFLR (Z2,IPW2,DF,1.0E12)

      return

      end

      subroutine SPYW (DT,Z,ZNULL,IFCT,IBCT,TW1,TW2,XORG,YORG,CZ)

c Plot tapered window shape above seismogram Z.
c 1. SPY calls PSGM to plot Z and get TW1,TW2.
c 2. SPY calls PSGM and SPYW to plot Z and window above spectrum.
```

```
      dimension Z(1)

      common /PSGMCOM/ TMIN,TMAX,ZMIN,ZMAX,XSCL,YSCL

      PI = 4.*ATAN(1.)
      TT = TW2-TW1
      TTF = TT*(IFCT/100.)
      TTB = TT*(IBCT/100.)
      I1 = NINT(TW1/DT)+1
      I2 = NINT((TW1+TTF)/DT)+1
      I3 = NINT((TW2-TTB)/DT)+1
      I4 = NINT(TW2/DT)+1
      ZZ = Z(I2)
      do 1 I=I2,I3
         ZI = Z(I)
         if (ZI.eq.ZNULL) goto 1
         if (ZI.gt.ZZ) ZZ = ZI
1     continue
      Y0 = YORG+(ZZ-ZMIN)*YSCL-0.9*CZ
10    T = (I1-1)*DT
      X = XORG+T*XSCL
      Y = Y0
      call PLOT (X,Y,+3)
20    SF = PI/(I2-I1+1)
      do 21 I=I1,I2
         T = (I-1)*DT
         X = XORG+T*XSCL
         Y = Y0+0.5*(1.-COS(SF*(I-I1)))*CZ
         call PLOT (X,Y,+2)
21    continue
30    T = (I3-1)*DT
      X = XORG+T*XSCL
      Y = Y0+CZ
      call PLOT(X,Y,+2)
40    SF = PI/(I4-I3+1)
      do 41 I=I3,I4
         T = (I-1)*DT
         X = XORG+T*XSCL
         Y = Y0+CZ-0.5*(1.-COS(SF*(I-I3)))*CZ
         call PLOT (X,Y,+2)
41    continue
      return

      end
```

Appendix B

FORTRAN Source Code: VFCOMMON.FI and subroutines

```
************************************************************ VFCOMMON.FI

c VFCOMMON.FI: common blocks for reading and modifying VFBB data files.
c /VFGET/ input info:
c  LUVFBB= logical unit number for opening and reading VFBB data file
c  TSKIP= time to skip (sec)
c  TREAD= time to (try to) read (sec)
c  IRDEC= read decimation factor
c /VFGOT/ output info:
c  IS1= first sample index
c  IS2= last sample index
c  NP= number of points
c  DT= delta T (sec)
c  E8= epoch time of first sample (sec from 00:00, 1 Jan 1970)
c       (TSKIP + header time + ?clock corr someday? + ?sample lag someday?)
c /VFCOMP/ component info:
c  IVA= vertical azimuth (usually 0 or 90)
c  IHA= horizontal azimuth (0<=I42A<=360)
c  IAVD= motion type (I255)
c /VFTRNS/ sensor info:
c  ITRNS= sensor type code (I043)
c  TFRQ= natural frequency (Hz)
c  TDMP= damping (% of critical damping)
c /VFRCDR/ recorder info:
c  IRCDR= recorder type code (I037)
c  AAFC= anti-alias corner frequency (Hz)
c  AAFR= anti-alias rolloff (dB/octave)
c /VFTMOD/ trace-modify info for subroutine MVFBB:
c  If NMO=0: Do LADC & LART, prompt user for options.
c  If 1=<NMO=<NMOMAX: Do LADC & LART, read NMO preset options from MOBUF.
c                     If some MOBUF='C'or'c' or NMOBUF=0 return,
c                     otherwise prompt user for more options.
c  LADC= logical variable for auto-dc
c  LART= logical variable for auto-rotate
c  IAZMRT= auto-rotate azimuth
c  NMO= number of preset modify options
c  MOBUF(I)= character string containing Ith preset modify option
c  NMOBUF(I)= number of characters in MOBUF(I)
c  MOLBL= character string containing modify history for labeling
c  NMOLBL= number of characters in MOLBL

      double precision E8
      logical LADC,LART
      parameter (NMOMAX=9)
      character MOBUF*20,MOLBL*80
      dimension MOBUF(NMOMAX),NMOBUF(NMOMAX)
      common /VFGET/ LUVFBB,TSKIP,TREAD,IRDEC
      common /VFGOT/ IS1,IS2,NP,DT,E8
      common /VFCOMP/ IVA,IHA,IAVD
      common /VFTRNS/ ITRNS,TFRQ,TDMP
      common /VFRCRD/ IRCDR,AAFC,AAFR
      common /VFTMOD/ LADC,LART,IAZMRT,NMO,MOBUF,NMOBUF,MOLBL,NMOLBL

************************************************************ CLRSTR ***

      subroutine CLRSTR (CSTR)

c CLRSTR - [MUELLER.FS.GEN]CSMGENLB
c Clear character string CSTR.

      character CSTR*(*)

      do 1 I=1,LEN(CSTR)
1        CSTR(I:I) = ' '
      return

      end

************************************************************ CMODY  ***

      subroutine CMODY (IYR4,IDOY,ICMO,ICDY,ISTAT)

c CMODY - [MUELLER.FS.GEN[CSMGENLB
c Convert 4-digit year and day-of-year (IYR4 and IDOY)
c  to calendar month and day (ICMO and ICDY).
c Valid for 1901=<IYR4=<2099 (simple leap-year test for this range).
c ISTAT= 0 for success;
c      = 1 if IYR4 is out-of-range;
c      = 2 if IDOY is out-of-range.
c      If ISTAT>0, return with ICMO=0 and ICDY=0.

      dimension MDAYS(12)

      ISTAT = 0
      ICMO = 0
      ICDY = 0
      if (IYR4.lt.1901.or.IYR4.gt.2099) then
        ISTAT = 1
        return
      endif
```

```
      ILEAP = 0
      if (MOD(IYR4,4).eq.0) ILEAP = 1
      if (IDOY.lt.1.or.IDOY.gt.365+ILEAP) then
        ISTAT = 2
        return
      endif
      MDAYS(1) = 31
      MDAYS(2) = 28+ILEAP
      MDAYS(3) = 31
      MDAYS(4) = 30
      MDAYS(5) = 31
      MDAYS(6) = 30
      MDAYS(7) = 31
      MDAYS(8) = 31
      MDAYS(9) = 30
      MDAYS(10) = 31
      MDAYS(11) = 30
      MDAYS(12) = 31
      M = 0
      I = IDOY
1     M = M+1
      I = I-MDAYS(M)
      if (I.gt.0) goto 1
      ICMO = M
      ICDY = I+MDAYS(M)
      return

      end

************************************************************* CORAAF ***

      subroutine CORAAF (Z,IPW2,DF,AAC,AAR)

c CORAAF - [MUELLER.FS.SD]CSMSDLB
c Remove anti-alias filter response from amplitude spectrum Z.
c Assume freq=0.0 at Z(1); correct Z from DF to Nyquist.
c AAC= corner frequency of a-a filter (Hz)
c AAR= rolloff of a-a filter (dB/octave)

      dimension Z(IPW2)

      INYQ = IPW2/2+1
      do 1 I=2,INYQ
         F = (I-1)*DF
         A = (F/AAC)**(2.*AAR/6.)
         B = 1./(1.+A)
         Z(I) = Z(I)/SQRT(B)
1        continue
      return

      end

************************************************************* CORATT ***

      subroutine CORATT (Z,IPW2,DF,Q0,QETA,RQ,VQ)

c CORATT - [MUELLER.FS.SD]CSMSDLB
c Remove attenuation factor from amplitude spectrum Z.
c Assume freq=0.0 at Z(1); correct Z from DF to Nyquist.
c Attenuation model: Q=Q0*F**QETA (Q0=Q at 1 Hz).
c RQ= hypocentral distance (km)
c VQ= velocity (km/sec)

      dimension Z(IPW2)

      PI = 4.*ATAN(1.)
      INYQ = IPW2/2+1
      do 1 I=2,INYQ
         F = (I-1)*DF
         Q = Q0*F**QETA
         X = (PI*F*RQ)/(Q*VQ)
         Z(I) = Z(I)*EXP(X)
1        continue
      return

      end

************************************************************* CORINS ***

      subroutine CORINS (Z,IPW2,DF,FRQ,DMP)

c CORINS - [MUELLER.FS.SD]CSMSDLB
c Remove geophone response shape from amplitude spectrum Z.
c Assume freq=0.0 at Z(1); correct Z from DF to Nyquist.
c FRQ= natural frequency (Hz)
c DMP= damping (fraction of critical)

      dimension Z(IPW2)

      TWOPI = 8.*ATAN(1.)
```

```
      W0 = TWOPI*FRQ
      INYQ = IPW2/2+1
      do 1 I=2,INYQ
         F = (I-1)*DF
         W = TWOPI*F
         A = (W*W)
         B = (W0*W0-W*W)**2
         C = (2.*W*W0*DMP)**2
         D = A/SQRT(B+C)
         Z(I) = Z(I)/D
1        continue
      return

      end

************************************************************ DC     ***

      subroutine DC (Z1,ZNULL1,NP,DT,IFIT1,IFIT2)

c DC - [MUELLER.FS.SD]CSMSDLB
c Apply "DC" correction to seismogram Z1.
c Compute mean of array Z1 between indices IFIT1 and IFIT2;
c  subtract mean between indices 1 and NP.

      dimension Z1(1)

      NSUM = 0
      ZSUM = 0.
      do 1 I=IFIT1,IFIT2
         if (Z1(I).eq.ZNULL1) goto 1
         NSUM = NSUM+1
         ZSUM = ZSUM+Z1(I)
1        continue
      ZDC = ZSUM/NSUM
      do 2 I=1,NP
         if (Z1(I).eq.ZNULL1) goto 2
         Z1(I) = Z1(I)-ZDC
2        continue
      return

      end

************************************************************ DIFRNC ***

      subroutine DIFRNC (Z1,Z2,ZNULL1,NP,DT)

c DIFRNC - [MUELLER.FS.SD]CSMSDLB
c Differentiate seismogram Z1.

      dimension Z1(1),Z2(1)

      do 1 I=1,NP
         Z2(I) = Z1(I)
         if (Z1(I).eq.ZNULL1) Z1(I) = 0.
1        continue
      do 2 I=1,NP-1
         Z1(I) = (Z1(I+1)-Z1(I))/DT
2        continue
      do 3 I=NP,2,-1
         Z1(I) = Z1(I-1)
3        continue
      Z1(1) = 0.
      do 4 I=1,NP
         if (Z2(I).eq.ZNULL1) Z1(I) = ZNULL1
4        continue
      return

      end

************************************************************ DOY    ***

      subroutine DOY (IYR4,ICMO,ICDY,IDOY,ISTAT)

c DOY - [MUELLER.FS.GEN]CSMGENLB
c Convert 4-digit year, calendar month and calendar day
c  (IYR4, ICMO, and ICDY) to day of year (IDOY).
c Valid for 1901=<IYR4=<2099 (simple leap-year test for this range).
c ISTAT= 0 for success;
c      = 1 if IYR4 is out-of-range;
c      = 2 if ICMO is out-of-range;
c      = 3 if ICDY is out-of-range.
c      If ISTAT>0, return with IDOY=0.

      dimension MDAYS(12)

      ISTAT = 0
      IDOY = 0
      if (IYR4.lt.1901.or.IYR4.gt.2099) then
        ISTAT = 1
        return
```

```
      endif
      ILEAP = 0
      if (MOD(IYR4,4).eq.0) ILEAP = 1
      MDAYS(1) = 31
      MDAYS(2) = 28+ILEAP
      MDAYS(3) = 31
      MDAYS(4) = 30
      MDAYS(5) = 31
      MDAYS(6) = 30
      MDAYS(7) = 31
      MDAYS(8) = 31
      MDAYS(9) = 30
      MDAYS(10) = 31
      MDAYS(11) = 30
      MDAYS(12) = 31
      if (ICMO.lt.0.or.ICMO.gt.12) then
        ISTAT = 2
        return
      endif
      if (ICDY.lt.0.or.ICDY.gt.MDAYS(ICMO)) then
        ISTAT = 3
        return
      endif
      IDOY = ICDY
      do 1 I=1,ICMO-1
1        IDOY = IDOY+MDAYS(I)
      return

      end

************************************************************ EYDHMS ***

      subroutine EYDHMS (E8,IYR4,IDY,IHR,IMN,ISC,IMSC,ISTAT)

c EYDHMS - [MUELLER.FS.GEN]CSMGENLB
c Compute 4-digit year (IYR4), day-of-year (IDY),
c  hour (IHR), minute (IMN), second (ISC), and millisecond (IMSC)
c  from epoch seconds (E8, relative to 00:00 1 Jan 1970).
c Valid for E1901=<E8<E2100 (simple leap-year test for this range).
c EYDHMS doesn't know about leap seconds.
c ISTAT= 0 for success;
c      = 1 if E8 is out-of-range.
c       If ISTAT>0, return with IYR4=1900,IDY=1,IHR-IMSC=0.

      real*8 E8,E8W,ETEST,ETESTT,E1901,E2100
      data E1901 /-2177452800.0d0/
      data E2100 /+4102444800.0d0/

      E8W = E8+0.0005 !try 521198464.000
      ISTAT = 0
      IYR4 = 1900
      IDY = 1
      IHR = 0
      IMN = 0
      ISC = 0
      IMSC = 0
      if (E8W.lt.E1901.or.E8W.ge.E2100) then
        ISTAT = 1
        return
      endif
      ETEST = E1901
      do 1 IYR4=1901,2099
         if (MOD(IYR4,4).eq.0) then
           ETESTT = ETEST+31622400.0d0
         else
           ETESTT = ETEST+31536000.0d0
         endif
         if (ETESTT.gt.E8W) goto 2
1        ETEST = ETESTT
2     ETEST = E8W-ETEST
      IDY = INT(ETEST/86400.0d0)+1
      ETEST = ETEST-(IDY-1)*86400.0d0
      IHR = INT(ETEST/3600.0d0)
      ETEST = ETEST-IHR*3600.0d0
      IMN = INT(ETEST/60.0d0)
      ETEST = ETEST-IMN*60.0d0
      ISC = INT(ETEST)
      ETEST = (ETEST-ISC)*1.0d3
      IMSC = INT(ETEST)
      return

      end

************************************************************ FBCTPR ***

      subroutine FBCTPR (Z1,ZNULL1,NP,DT,IFRNT,IBACK)

c FBCTPR - [MUELLER.FS.SD]CSMSDLB
c Apply IFRNT% and IBACK% cosine tapers to seismogram Z1.
```

```
      dimension Z1(1)

      PI = 4.*ATAN(1.)
      LP = NINT(NP*IFRNT/100.)
      SF = PI/LP
      do 1 I=1,LP
         if (Z1(I).eq.ZNULL1) goto 1
         F = 0.5*(1.-COS(SF*(I-1)))
         Z1(I) = Z1(I)*F
1     continue
      LP = NINT(NP*IBACK/100.)
      SF = PI/LP
      do 2 I=NP,NP-LP+1,-1
         if (Z1(I).eq.ZNULL1) goto 2
         F = 0.5*(1.-COS(SF*(NP-I)))
         Z1(I) = Z1(I)*F
2     continue
      return

      end

************************************************************** FHP2   ***

      subroutine FHP2 (Z1,Z2,ZNULL1,NP,DT,FC)

c FHP2 - [MUELLER.FS.SD]CSMSDLB
c Apply zero-phase-shift hipass filter to seismogram Z1.
c (Subroutine HIPAS is in USERLIB.)

      dimension Z1(1),Z2(1)

      do 1 I=1,NP
         Z2(I) = Z1(I)
         if (Z1(I).eq.ZNULL1) Z1(I) = 0.
1     continue
      FNY = 2.*DT*FC
      call HIPAS (1,FNY,Z1,NP,0)
      call HIPAS (1,FNY,Z1,NP,1)
      do 2 I=1,NP
         if (Z2(I).eq.ZNULL1) Z1(I) = ZNULL1
2     continue
      return

      end

************************************************************** FLP2   ***

      subroutine FLP2 (Z1,Z2,ZNULL1,NP,DT,FC)

c FLP2 - [MUELLER.FS.SD]CSMSDLB
c Apply zero-phase-shift lopass filter to seismogram Z1.
c (Subroutine LOPAS is in USERLIB.)

      dimension Z1(1),Z2(1)

      do 1 I=1,NP
         Z2(I) = Z1(I)
         if (Z1(I).eq.ZNULL1) Z1(I) = 0.
1     continue
      FNY = 2.*DT*FC
      call LOPAS (1,FNY,Z1,NP,0)
      call LOPAS (1,FNY,Z1,NP,1)
      do 2 I=1,NP
         if (Z2(I).eq.ZNULL1) Z1(I) = ZNULL1
2     continue
      return

      end

************************************************************** GPHONE ***

      subroutine GPHONE (Z1,Z2,ZC,ZNULL1,NP,DT,FRQ,DMP,NF)

c GPHONE - [MUELLER.FS.SD]CSMSDLB
c Remove geophone response shape from seismogram Z1;
c  use equation from Aki-Richards and A-R sign conventions.
c FRQ= natural frequency of geophone (Hz).
c DMP= damping (fraction of critical).
c NF= corner-frequency factor for low-cut noise filter (see below).
c Strategy...
c 1. Taper and pad Z1; load ZC; forward FFT.
c 2. Compute geophone response (shape) and divide.
c 3. Apply low-cut filter with corner at F=FRQ/NF.
c    (modified spectrum might be noisy at low frequencies; try NF=4.)
c 4. Inverse FFT; reload Z1.

      complex ZC,CINS
      dimension Z1(1),Z2(1),ZC(1)

      do 1 I=1,NP
```

```
         Z2(I) = Z1(I)
         if (Z1(I).eq.ZNULL1) Z1(I) = 0.
1        continue
      PI = 4.*ATAN(1.)
      WS = 2.*PI*FRQ
      WLC = WS/NF
      N = NP
      call FBCTPR (Z1,ZNULL1,N,DT,1,1)
      call PADPW2 (Z1,ZNULL1,N,DT)
      do 2 I=1,N
         ZC(I) = CMPLX(Z1(I),0.)
2        continue
      call FORK (N,ZC,+1.)
      INYQ = N/2+1
      TT = (N-1)*DT
      DW = 2.*PI*(1./TT)
      ZC(1) = CMPLX(0.,0.)
      ZC(INYQ) = CMPLX(ABS(ZC(INYQ)),0.)
      do 3 I=2,INYQ-1
         W = (I-1)*DW
         W2 = W*W
         CINS = CMPLX(0.,2.*DMP*WS*W)
         CINS = W2/(W2+CINS-WS*WS)
         FACTLC = ((W/WLC)**4)/(1.+(W/WLC)**4)
         ZC(I) = ZC(I)*FACTLC/CINS
         ZC(N-I+2) = CONJG(ZC(I))
3        continue
      call FORK (N,ZC,-1.)
      do 4 I=1,NP
         Z1(I) = REAL(ZC(I))
         if (Z2(I).eq.ZNULL1) Z1(I) = ZNULL1
4        continue
      return

      end

************************************************************ INTGRT ***

      subroutine INTGRT (Z1,Z2,ZNULL1,NP,DT)

c INTGRT - [MUELLER.FS.SD]CSMSDLB
c Integrate seismogram Z1.
c You might want to "DC"-correct and/or filter Z1.

      dimension Z1(1),Z2(1)

      do 1 I=1,NP
         Z2(I) = Z1(I)
         if (Z1(I).eq.ZNULL1) Z1(I) = 0.
1        continue
      HDT = DT/2.
      Z1(1) = HDT*(Z1(1)+Z1(2))
      do 2 I=2,NP-1
         Z1(I) = Z1(I-1)+HDT*(Z1(I)+Z1(I+1))
2        continue
      do 3 I=NP,2,-1
         Z1(I) = Z1(I-1)
3        continue
      Z1(1) = 0.
      do 4 I=1,NP
         if (Z2(I).eq.ZNULL1) Z1(I) = ZNULL1
4        continue
      return

      end

************************************************************ LGABS  ***

      subroutine LGABS (IABSSF,ABSMN,ABSMX,XSCL,
     *                  XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)

c LGABS - [MUELLER.FS.PLT]CSMPLTLB
c Draw and label logarithmic abscissa axis.
c Input:  IABSSF - log10 scale factor (ABSSF=10**IABSSF).
c         ABSMN,ABSMX - min and max (abscissa units).
c         XORG,YORG - bottom left-hand corner of box (inches).
c         XLEN,YLEN - box dimensions (inches).
c         CZ - character size (inches).
c         TZ - tick size (inches).
c         ALBL,NALBL - label.
c         LTR = .true. if you want (unlabeled) top axis.
c Output: ABSMN,ABSMX - might be rounded from input.
c         XSCL (inches per decade).

      logical LTR
      character ALBL*(*),C10*10

c Scale and round QMIN and QMAX.
100   ABSSF = 10.**IABSSF
      QMIN = (1.001*ABSMN)/ABSSF !kludge
```

```
      QMAX = (0.999*ABSMX)/ABSSF !kludge
      IQMINY = ALOG10(QMIN)
      if (IQMINY.gt.ALOG10(QMIN)) IQMINY = IQMINY-1
      IQMAXY = ALOG10(QMAX)
      if (IQMAXY.gt.ALOG10(QMAX)) IQMAXY = IQMAXY-1
      IQMINX = QMIN/(10.**IQMINY)
      if (IQMAXY.eq.IQMINY) IQMINX = 1
      IQMAXX = QMAX/(10.**IQMAXY)+1
      QMIN = IQMINX*(10.**IQMINY)
      QMAX = IQMAXX*(10.**IQMAXY)
      QSCALE = XLEN/ALOG10(QMAX/QMIN)
c Draw box, tick marks, and label.
200   call PLOT (XORG,YORG,+3)
      call PLOT (XORG+XLEN,YORG,+2)
      if (LTR) then
        call PLOT (XORG,YORG+YLEN,+3)
        call PLOT (XORG+XLEN,YORG+YLEN,+2)
      endif
210   IQX = IQMINX-1
      IQY = IQMINY
211   IQX = IQX+1
      if (IQX.eq.10) then
        IQX = 1
        IQY = IQY+1
      endif
      Q = IQX*(10.**IQY)
      if (Q.gt.QMAX) goto 219
      X = XORG+QSCALE*ALOG10(Q/QMIN)
      TZF = 1.
      if (IQX.eq.1) TZF = 2.
      call PLOT (X,YORG,+3)
      call PLOT (X,YORG+TZF*TZ,+2)
      if (LTR) then
        call PLOT (X,YORG+YLEN,+3)
        call PLOT (X,YORG+YLEN-TZF*TZ,+2)
      endif
      if (IQX.eq.1) then
        if (QMIN.ge.0.001.and.QMAX.le.1000.) then
          call WCF (Q,1,3,C10,1,N)
          if (Q.lt.1.) call TEXT (X-1.25*CZ,YORG-2.*CZ,CZ,C10,0.,N)
          if (Q.ge.1.) call TEXT (X-(N-1.75)*CZ,YORG-2.*CZ,CZ,C10,0.,N)
        else
          call WCI (IQY,C10,1,N)
          call TEXT (X-1.25*CZ,YORG-2.*CZ,CZ,'10',0.,2)
          call TEXT (X+0.75*CZ,YORG-1.5*CZ,0.75*CZ,C10,0.,N)
        endif
      endif
      goto 211
219   continue
220   if (IABSSF.ne.0) then
        call WCC (' *(10**',1,7,ALBL,NALBL+1,NALBL)
        call WCI (IABSSF,ALBL,NALBL+1,NALBL)
        call WCC (')',1,1,ALBL,NALBL+1,NALBL)
      endif
      XX = AMAX1(XORG,XORG+(XLEN-NALBL*CZ)/2.)
      YY = YORG-4.*CZ
      call TEXT (XX,YY,CZ,ALBL,0.,NALBL)
c Compute ABSMN,ABSMX,XSCL.
300   ABSMN = QMIN*ABSSF
      ABSMX = QMAX*ABSSF
      XSCL = QSCALE
      return

      end

************************************************************ LGORD  ***

      subroutine LGORD (IORDSF,ORDMN,ORDMX,YSCL,
     *                  XORG,YORG,XLEN,YLEN,CZ,TZ,OLBL,NOLBL,LTR)

c LGORD - [MUELLER.FS.PLT]CSMPLTLB
c Draw and label logarithmic ordinate axis.
c Input:  IORDSF - ALOG10 scale factor (ORDSF=10.**IORDSF).
c         ORDMN,ORDMX - min and max (ordinate units).
c         XORG,YORG - bottom left-hand corner of box (inches).
c         XLEN,YLEN - box dimensions (inches).
c         CZ - character size (inches).
c         TZ - tick size (inches).
c         OLBL,NOLBL - label.
c         LTR = .true. if you want (unlabeled) right axis.
c Output: ORDMN,ORDMX - might be rounded from input.
c         YSCL (inches per decade).

      logical LTR
      character OLBL*(*),C10*10

c Scale and round QMIN and QMAX.
100   ORDSF = 10.**IORDSF
      QMIN = (1.001*ORDMN)/ORDSF !kludge
      QMAX = (0.999*ORDMX)/ORDSF !kludge
      IQMINY = ALOG10(QMIN)
```

```
      if (IQMINY.gt.ALOG10(QMIN)) IQMINY = IQMINY-1
      IQMAXY = ALOG10(QMAX)
      if (IQMAXY.gt.ALOG10(QMAX)) IQMAXY = IQMAXY-1
      IQMINX = QMIN/(10.**IQMINY)
      if (IQMAXY.eq.IQMINY) IQMINX = 1
      IQMAXX = QMAX/(10.**IQMAXY)+1
      QMIN = IQMINX*(10.**IQMINY)
      QMAX = IQMAXX*(10.**IQMAXY)
      QSCALE = YLEN/ALOG10(QMAX/QMIN)
c Draw box, tick marks, and label.
200   call PLOT (XORG,YORG,+3)
      call PLOT (XORG,YORG+YLEN,+2)
      if (LTR) then
        call PLOT (XORG+XLEN,YORG,+3)
        call PLOT (XORG+XLEN,YORG+YLEN,+2)
      endif
210   XXMIN = XORG
      IQX = IQMINX-1
      IQY = IQMINY
211   IQX = IQX+1
      if (IQX.eq.10) then
        IQX = 1
        IQY = IQY+1
      endif
      Q = IQX*(10.**IQY)
      if (Q.gt.QMAX) goto 219
      Y = YORG+QSCALE*ALOG10(Q/QMIN)
      TZF = 1.
      if (IQX.eq.1) TZF = 2.
      call PLOT (XORG,Y,+3)
      call PLOT (XORG+TZ*TZF,Y,+2)
      if (LTR) then
        call PLOT (XORG+XLEN,Y,+3)
        call PLOT (XORG+XLEN-TZ*TZF,Y,+2)
      endif
      if (IQX.eq.1) then
        call WCI (IQY,C10,1,N)
        XX = XORG-(2.+N*0.75+0.5)*CZ
        call TEXT (XX,Y-0.5*CZ,CZ,'10',0.,2)
        call TEXT (XX+2.*CZ,Y,0.75*CZ,C10,0.,N)
        if (XX.lt.XXMIN) XXMIN = XX
      endif
      goto 211
219   continue
220   if (IORDSF.ne.0) then
        call WCC (' *(10**',1,7,OLBL,NOLBL+1,NOLBL)
        call WCI (IORDSF,OLBL,NOLBL+1,NOLBL)
        call WCC (')',1,1,OLBL,NOLBL+1,NOLBL)
      endif
      XX = XXMIN-CZ
      YY = AMAX1(YORG,YORG+(YLEN-NOLBL*CZ)/2.)
      call TEXT (XX,YY,CZ,OLBL,90.,NOLBL)
c Compute ORDMN,ORDMX,YSCL.
300   ORDMN = QMIN*ORDSF
      ORDMX = QMAX*ORDSF
      YSCL = QSCALE
      return

      end

************************************************************ LNABS  ***

      subroutine LNABS (IABSSF,ABSMN,ABSMX,XSCL,
     *                  XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)

c LNABS - [MUELLER.FS.PLT]CSMPLTLB
c Draw and label linear abscissa axis.
c Input:  IABSSF - ALOG10 scale factor (ABSSF=10.**IABSSF)
c         ABSMN,ABSMX - min and max (abscissa units).
c         XORG,YORG - bottom left-hand corner of box (inches).
c         XLEN,YLEN - box dimensions (inches).
c         CZ - character size (inches).
c         TZ - tick size (inches).
c         ALBL,NALBL - label.
c         LTR = .true. if you want (unlabeled) top axis.
c Output: ABSMN,ABSMX - might be rounded from input.
c         XSCL - inches per unscaled unit.

      logical LTR
      character ALBL*(*),C10*10

c Scale and round QMIN and QMAX.
c Find JMIN, JMAX, JNXT, etc.
100   ABSSF = 10.**IABSSF
      QMIN = ABSMN/ABSSF
      QMAX = ABSMX/ABSSF
      QK = (QMAX-QMIN)/1000. !kludge
      QMIN = QMIN+QK
      QMAX = QMAX-QK
      Q = QMAX-QMIN
```

```
      IQY = ALOG10(Q/10.)
      if (IQY.gt.ALOG10(Q/10.)) IQY = IQY-1
      IQX = 1
      if (Q/(10.**IQY).gt.50.) IQX = 5
      QTCK = IQX*(10.**IQY)
      JMIN = QMIN/QTCK
      if (JMIN*QTCK.gt.QMIN) JMIN = JMIN-1
      QMIN = JMIN*QTCK
      JMAX = QMAX/QTCK
      if (JMAX*QTCK.lt.QMAX) JMAX = JMAX+1
      QMAX = JMAX*QTCK
      QSCL = XLEN/(QMAX-QMIN)
      J = JMAX-JMIN+1
      if (IQX.eq.1.and.J.le.25) JNXT = 5
      if (IQX.eq.1.and.J.gt.25) JNXT = 10
      if (IQX.eq.5) JNXT = 4
      QNXT = JNXT*QTCK
      IQNXTY = ALOG10(QNXT)
      if (IQNXTY.gt.ALOG10(QNXT)) IQNXTY = IQNXTY-1
c Draw box.
200   call PLOT (XORG,YORG,+3)
      call PLOT (XORG+XLEN,YORG,+2)
      if (LTR) then
        call PLOT (XORG,YORG+YLEN,+3)
        call PLOT (XORG+XLEN,YORG+YLEN,+2)
      endif
c Draw tick marks.
210   do 211 J=JMIN,JMAX
         YTZ = TZ
         if (MOD(J,JNXT).eq.0) YTZ = 2.*TZ
         Q = J*QTCK
         X = XORG+(Q-QMIN)*QSCL
         call PLOT (X,YORG,+3)
         call PLOT (X,YORG+YTZ,+2)
         if (LTR) then
           call PLOT (X,YORG+YLEN,+3)
           call PLOT (X,YORG+YLEN-YTZ,+2)
         endif
211      continue
c Get IW (0 = "i" format, 1 = "f" format, 2 = "e" format).
220   IW = 0
      do 221 J=JMIN,JMAX
         if (MOD(J,JNXT).ne.0) goto 221
         Q = J*QTCK
         if (Q.eq.0.) goto 221
         IQY = ALOG10(ABS(Q))
         if (IQY.gt.ALOG10(ABS(Q))) IQY = IQY-1
         NSIG = IQY+1-IQNXTY
         Q = NINT(Q)/Q
         QMEPS = 1.-10.**(-NSIG)
         QPEPS = 1.+10.**(-NSIG)
         if (Q.lt.QMEPS.or.Q.gt.QPEPS) IW = 1
221      continue
      Q = ABS(QMIN)
      if ((Q.lt.0.0001.or.Q.gt.10000.).and.Q.ne.0.) IW = 2
      Q = ABS(QMAX)
      if ((Q.lt.0.0001.or.Q.gt.10000.).and.Q.ne.0.) IW = 2
c Draw labels.
230   do 231 J=JMIN,JMAX
         if (MOD(J,JNXT).ne.0) goto 231
         Q = J*QTCK
         if (IW.eq.0.or.Q.eq.0.) then
           call WCI (NINT(Q),C10,1,N)
         else
           IQY = ALOG10(ABS(Q))
           if (IQY.gt.ALOG10(ABS(Q))) IQY = IQY-1
           NSIG = IQY+1-IQNXTY
           call WCF (Q,NSIG,IW,C10,1,N)
         endif
         X = XORG+(Q-QMIN)*QSCL-0.5*N*CZ
         Y = YORG-2.*CZ
         call TEXT (X,Y,CZ,C10,0.,N)
231      continue
240   if (IABSSF.ne.0) then
        call WCC (' *(10**',1,7,ALBL,NALBL+1,NALBL)
        call WCI (IABSSF,ALBL,NALBL+1,NALBL)
        call WCC (')',1,1,ALBL,NALBL+1,NALBL)
      endif
      X = AMAX1(XORG,XORG+(XLEN-NALBL*CZ)/2.)
      Y = YORG-4.*CZ
      call TEXT (X,Y,CZ,ALBL,0.,NALBL)
c Compute ABSMN,ABSMX,XSCL.
300   ABSMN = QMIN*ABSSF
      ABSMX = QMAX*ABSSF
      XSCL = QSCL/ABSSF
      return

      end

************************************************************ LNORD  ***
```

```
      subroutine LNORD (IORDSF,ORDMN,ORDMX,YSCL,
     *                  XORG,YORG,XLEN,YLEN,CZ,TZ,OLBL,NOLBL,LTR)

c LNORD - [MUELLER.FS.PLT]CSMPLTLB
c Draw and label linear ordinate axis.
c Input:  IORDSF - ALOG10 scale factor (ORDSF=10.**IORDSF)
c         ORDMN,ORDMX - min and max (ordinate units).
c         XORG,YORG - bottom left-hand corner of box (inches).
c         XLEN,YLEN - box dimensions (inches).
c         CZ - character size (inches).
c         TZ - tick size (inches).
c         OLBL,NOLBL - label.
c         LTR = .true. if you want (unlabeled) right axis.
c Output: ORDMN,ORDMX - might be rounded.
c         YSCL - inches per unscaled unit.

      logical LTR,LSPARS
      character OLBL*(*),C10*10

c Scale and round QMIN and QMAX.
c Find JMIN, JMAX, JNXT, etc.
100   ORDSF = 10.**IORDSF
      QMIN = ORDMN/ORDSF
      QMAX = ORDMX/ORDSF
      QK = (QMAX-QMIN)/1000. !kludge
      QMIN = QMIN+QK
      QMAX = QMAX-QK
      Q = QMAX-QMIN
      IQY = ALOG10(Q/10.)
      if (IQY.gt.ALOG10(Q/10.)) IQY = IQY-1
      IQX = 1
      if (Q/(10.**IQY).ge.50.) IQX = 5
      QTCK = IQX*(10.**IQY)
      JMIN = QMIN/QTCK
      if (JMIN*QTCK.gt.QMIN) JMIN = JMIN-1
      QMIN = JMIN*QTCK
      JMAX = QMAX/QTCK
      if (JMAX*QTCK.lt.QMAX) JMAX = JMAX+1
      QMAX = JMAX*QTCK
      QSCL = YLEN/(QMAX-QMIN)
      J = JMAX-JMIN+1
      if (IQX.eq.1.and.J.le.25) JNXT = 5
      if (IQX.eq.1.and.J.gt.25) JNXT = 10
      if (IQX.eq.5) JNXT = 4
      QNXT = JNXT*QTCK
      IQNXTY = ALOG10(QNXT)
      if (IQNXTY.gt.ALOG10(QNXT)) IQNXTY = IQNXTY-1
c Find LSPARS and first and last labeled ticks.
      LSPARS = .false.
      if (YLEN/CZ.lt.20.) LSPARS = .true.
      JF = (JMIN/JNXT)*JNXT
      if (JF.lt.JMIN) JF = JF+JNXT
      JL = (JMAX/JNXT)*JNXT
      if (JL.gt.JMAX) JL = JL-JNXT
c Draw box.
200   call PLOT (XORG,YORG,+3)
      call PLOT (XORG,YORG+YLEN,+2)
      if (LTR) then
        call PLOT (XORG+XLEN,YORG,+3)
        call PLOT (XORG+XLEN,YORG+YLEN,+2)
      endif
c Draw tick marks.
210   do 211 J=JMIN,JMAX
         XTZ = TZ
         if (MOD(J,JNXT).eq.0) XTZ = 2.*TZ
         Q = J*QTCK
         Y = YORG+(Q-QMIN)*QSCL
         call PLOT (XORG,Y,+3)
         call PLOT (XORG+XTZ,Y,+2)
         if (LTR) then
           call PLOT (XORG+XLEN,Y,+3)
           call PLOT (XORG+XLEN-XTZ,Y,+2)
         endif
211      continue
c Get IW (0 = "i" format, 1 = "f" format, 2 = "e" format).
220   IW = 0
      do 221 J=JMIN,JMAX
         if (LSPARS.and.J.ne.JF.and.J.ne.JL) goto 221
         if (MOD(J,JNXT).ne.0) goto 221
         Q = J*QTCK
         if (Q.eq.0.) goto 221
         IQY = ALOG10(ABS(Q))
         if (IQY.gt.ALOG10(ABS(Q))) IQY = IQY-1
         NSIG = IQY+1-IQNXTY
         Q = NINT(Q)/Q
         QMEPS = 1.-10.**(-NSIG)
         QPEPS = 1.+10.**(-NSIG)
         if (Q.lt.QMEPS.or.Q.gt.QPEPS) IW = 1
221      continue
```

```
      Q = ABS(QMIN)
      if ((Q.lt.0.001.or.Q.gt.1000.).and.Q.ne.0.) IW = 2
      Q = ABS(QMAX)
      if ((Q.lt.0.001.or.Q.gt.1000.).and.Q.ne.0.) IW = 2
c Draw labels.
230   XMIN = XORG
      do 231 J=JMIN,JMAX
         if (LSPARS.and.J.ne.JF.and.J.ne.JL) goto 231
         if (MOD(J,JNXT).ne.0) goto 231
         Q = J*QTCK
         if (IW.eq.0.or.Q.eq.0.) then
           call WCI (NINT(Q),C10,1,N)
         else
           IQY = ALOG10(ABS(Q))
           if (IQY.gt.ALOG10(ABS(Q))) IQY = IQY-1
           NSIG = IQY+1-IQNXTY
           call WCF (Q,NSIG,IW,C10,1,N)
         endif
         X = XORG-(N+0.5)*CZ
         Y = YORG+(Q-QMIN)*QSCL-0.5*CZ
         call TEXT (X,Y,CZ,C10,0.,N)
         if (X.lt.XMIN) XMIN = X
231      continue
240   if (IORDSF.ne.0) then
        call WCC (' *(10**',1,7,OLBL,NOLBL+1,NOLBL)
        call WCI (IORDSF,OLBL,NOLBL+1,NOLBL)
        call WCC (')',1,1,OLBL,NOLBL+1,NOLBL)
      endif
      X = XMIN-CZ
      Y = AMAX1(YORG,YORG+(YLEN-NOLBL*CZ)/2.)
      call TEXT (X,Y,CZ,OLBL,90.,NOLBL)
c Compute ORDMN,ORDMX,YSCL.
300   ORDMN = QMIN*ORDSF
      ORDMX = QMAX*ORDSF
      YSCL = QSCL/ORDSF
      return

      end

************************************************************ MVFBB  ***

      subroutine MVFBB (Z1,Z2,ZC,ZNULL1,ISIZ,FNM)

c MVFBB - [MUELLER.FS.SD]CSMSDLB
c Modify seismogram Z1. See also RVFBB.FOR and VFBBCOMMON.FI.
c MVFBB mode if LFIRST=.true. (first MVFBB call for Z1) ->
c  If NMO=0 - Do LADC & LART, prompt user for options.
c  If 1=<NMO=<NMOMAX - Do LADC & LART, read NMO preset options from MOBUF;
c                      if some MOBUF='C'or'c' or NMOBUF=0 return,
c                      otherwise prompt user for more options.
c MVFBB mode if LFIRST=.false. ->
c  Prompt user for trace-modify options.
c Options:
c  0 - Rotate (parameter IAZMRT)
c  1 - Subtract DC (parameters TFIT1,TFIT2)
c  2 - Hipass filter (parameter FC)
c  3 - Lopass filter (parameter FC)
c  4 - Integrate
c  5 - Differentiate
c  6 - Remove geophone response (parameter NF)
c  7 - Pad with temporary zeros (parameter ZT)
c  8 - Multiply by scale factor (parameter SF)
c  C or c or <cr> - Return and Continue
c  S or s - Return and StartOver (test for MOLBL='StartOver' in main prog)
c Arguments:
c  Z1= seismogram array (ZNULL1= seismogram null, ISIZ= dimension)
c  Z2,CZ= work-space arrays
c  FNM = filename corresponding to Z1
c MOLBL should be blank for first call with a new trace;
c  after that, MVFBB will handle multiple calls properly.

      complex ZC
      dimension Z1(1),Z2(1),ZC(1)
      logical LFIRST,LPROMPT,LVFBB
      character FNM*(*),BUF*80,C1*1

      include '[MUELLER.FS.SD]VFCOMMON.FI'

      NZT = 0

c Parse MOLBL to determine LFIRST.
10    NMOLBL = INDEX(MOLBL,' ')-1
      LFIRST = .false.
      if (NMOLBL.eq.0) LFIRST = .true.

c If LFIRST, do LADC (auto-DC) and LART (auto-rotate (LVFBB only)).
c You can suppress auto-rotation by passing IAZMRT<0 to MVFBB.
20    if (.not.LFIRST) goto 29
      call WCC ('M=',1,2,MOLBL,1,NMOLBL)
      if (LADC) then
```

```
        call DC (Z1,ZNULL1,NP,DT,1,NP)
        call WCC ('ADC',1,3,MOLBL,NMOLBL+1,NMOLBL)
      endif
      if (LART.and.IAZMRT.ge.0) then
        BUF = FNM
        call PARSEF (BUF,IC2,IC1,ILB,IRB,IDT,ISC,IEND,LVFBB,IP)
        if (.not.LVFBB.or.IP.ne.0) goto 8020
        I = IDT-1
        C1 = '?'
        if (BUF(I:I).eq.'2') C1 = '3'
        if (BUF(I:I).eq.'3') C1 = '2'
        if (BUF(I:I).eq.'5') C1 = '6'
        if (BUF(I:I).eq.'6') C1 = '5'
        BUF(I:I) = C1
        open (unit=LUVFBB,file=BUF(1:IEND),err=8020,
     *        status='OLD',readonly,
     *        access='DIRECT',recl=128)
        NPS = NP
        DTS = DT
        IVAS = IVA
        IHAS = IHA
        IAVDS = IAVD
        call RVFBB (Z2,ZNULL2,ISIZ)
        close (unit=LUVFBB)
        if (NP.le.0) goto 8020
        if (LADC) call DC (Z2,ZNULL2,NP,DT,1,NP)
        NP = MIN(NPS,NP)
        if (DT.ne.DTS) goto 8020
        if (IVA.ne.90.or.IVAS.ne.90) goto 8020
        MH = ABS(MOD(IHA-IHAS,360))
        if (MH.ne.90.and.MH.ne.270) goto 8020
        if (IAVD.ne.IAVDS) goto 8020
        call ROT12 (Z1,ZNULL1,IHAS,Z2,ZNULL2,IHA,NP,DT,IAZMRT)
        IHA = IHAS
        if (NMOLBL.gt.2) call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC ('ART',1,3,MOLBL,NMOLBL+1,NMOLBL)
      endif
29    continue

c Test LFIRST and NMO to determine MVFBB mode.
30    LPROMPT = .true.
      if (LFIRST.and.NMO.ge.1) LPROMPT = .false.

c Get options.
      NCOUNT = 0
100   if (LPROMPT) write (6,90100) MOLBL(1:NMOLBL)
90100   format (' Options: ',a/
     * ' > Rotate = 0,IAZMRT(deg)[i]'/
     * ' > Subtract DC = 1,TFIT1(s),TFIT2(s)[2f;<cr>=wholetrace]'/
     * ' > Hipass filter = 2,FC(Hz)[f]'/
     * ' > Lopass filter = 3,FC(Hz)[f]'/
     * ' > Integrate = 4'/
     * ' > Differentiate = 5'/
     * ' > Remove geophone response = 6,NF[i;<cr>=4]'/
     * ' > Pad with temporary zeros = 7,ZT(s)[f]'/
     * ' > Multiply by scale factor = 8,SF[f]'/
     * ' > Continue = C or <cr>'/
     * ' > StartOver = S')
110   NCOUNT = NCOUNT+1
      if (LPROMPT) then
        write (6,90110)
90110   format (' Choose option: ',$)
        read (5,90111,err=8100) NBUF,BUF
90111   format (q,a)
      else
        BUF = MOBUF(NCOUNT)
        NBUF = NMOBUF(NCOUNT)
      endif
      if (BUF(1:1).eq.'S'.or.BUF(1:1).eq.'s') goto 910
      if (BUF(1:1).eq.'C'.or.BUF(1:1).eq.'c'.or.NBUF.eq.0) goto 920
      call RCI (1,BUF,NBUF,IOPT,IR)
      if (IR.ne.0) goto 8100

c Rotate. If you rotate, it must be the first option.
200   if (IOPT.eq.0) then
        if (LADC.and.NMOLBL.ne.5) goto 8200
        if (.not.LADC.and.NMOLBL.ne.2) goto 8200
        call RCI (2,BUF,NBUF,IAZMRT,IR)
        if (IR.ne.0) goto 8100
201     BUF = FNM
        call PARSEF (BUF,IC2,IC1,ILB,IRB,IDT,ISC,IEND,LVFBB,IP)
        if (IP.ne.0) goto 8200
        if (LVFBB) then
          I = IDT-1
          C1 = '?'
          if (BUF(I:I).eq.'2') C1 = '3'
          if (BUF(I:I).eq.'3') C1 = '2'
          if (BUF(I:I).eq.'5') C1 = '6'
          if (BUF(I:I).eq.'6') C1 = '5'
          BUF(I:I) = C1
```

```
        else
          write (6,90201) BUF(IRB+1:ISC-1)
90201     format (' 1st trace = ',a/
     *            ' 2nd trace = ',$)
          read (5,90202,err=201) BUF(IRB+1:)
90202     format (a)
          call PARSEF (BUF,IC2,IC1,ILB,IRB,IDT,ISC,IEND,LVFBB,IP)
          if (IP.ne.0) goto 201
        endif
        open (unit=LUVFBB,file=BUF(1:IEND),err=8200,
     *      status='OLD',readonly,
     *      access='DIRECT',recl=128)
        NPS = NP
        DTS = DT
        IVAS = IVA
        IHAS = IHA
        IAVDS = IAVD
        call RVFBB (Z2,ZNULL2,ISIZ)
        close (unit=LUVFBB)
        if (NP.le.0) goto 8200
        if (LADC) call DC (Z2,ZNULL2,NP,DT,1,NP)
        NP = MIN(NPS,NP)
        if (DT.ne.DTS) goto 8200
        if (IVA.ne.90.or.IVAS.ne.90) goto 8200
        MH = ABS(MOD(IHA-IHAS,360))
        if (MH.ne.90.and.MH.ne.270) goto 8200
        if (IAVD.ne.IAVDS) goto 8200
        call ROT12 (Z1,ZNULL1,IHAS,Z2,ZNULL2,IHA,NP,DT,IAZMRT)
        IHA = IHAS
        if (NMOLBL.gt.2) call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC ('RT',1,2,MOLBL,NMOLBL+1,NMOLBL)

c Subtract DC.
210   elseif (IOPT.eq.1) then
        if (NBUF.eq.1) then
          TFIT1 = 0
          TFIT2 = (NP-1)*DT
        else
          call RCF (2,BUF,NBUF,TFIT1,IR1)
          call RCF (3,BUF,NBUF,TFIT2,IR2)
          if (IR1.ne.0.or.IR2.ne.0) goto 8100
        endif
        IFIT1 = NINT(TFIT1/DT+1)
        IFIT2 = NINT(TFIT2/DT+1)
        if (IFIT1.lt.1.or.IFIT1.gt.NP) goto 8200
        if (IFIT2.lt.1.or.IFIT2.gt.NP) goto 8200
        if (IFIT1.ge.IFIT2) goto 8200
        call DC (Z1,ZNULL1,NP,DT,IFIT1,IFIT2)
        if (NMOLBL.gt.2) call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC ('DC(',1,3,MOLBL,NMOLBL+1,NMOLBL)
        call WCF (TFIT1,3,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCF (TFIT2,3,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC (')',1,1,MOLBL,NMOLBL+1,NMOLBL)

c Hipass filter.
220   elseif (IOPT.eq.2) then
        call RCF (2,BUF,NBUF,FC,IR)
        if (IR.ne.0) goto 8100
        F1 = 1./((NP-1)*DT)
        F2 = 1./(2.*DT)
        if (FC.lt.F1.or.FC.gt.F2) goto 8200
        call FHP2 (Z1,Z2,ZNULL1,NP,DT,FC)
        if (NMOLBL.gt.2) call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC ('HP(',1,3,MOLBL,NMOLBL+1,NMOLBL)
        call WCF (FC,3,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC (')',1,1,MOLBL,NMOLBL+1,NMOLBL)

c Lopass filter.
230   elseif (IOPT.eq.3) then
        call RCF (2,BUF,NBUF,FC,IR)
        if (IR.ne.0) goto 8100
        F1 = 1./((NP-1)*DT)
        F2 = 1./(2.*DT)
        if (FC.lt.F1.or.FC.gt.F2) goto 8200
        call FLP2 (Z1,Z2,ZNULL1,NP,DT,FC)
        if (NMOLBL.gt.2) call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC ('LP(',1,3,MOLBL,NMOLBL+1,NMOLBL)
        call WCF (FC,3,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC (')',1,1,MOLBL,NMOLBL+1,NMOLBL)

c Integrate.
240   elseif (IOPT.eq.4) then
        call INTGRT (Z1,Z2,ZNULL1,NP,DT)
        if (IAVD.eq.1.or.IAVD.eq.2.or.IAVD.eq.3.or.
     *      IAVD.eq.10.or.IAVD.eq.20.or.IAVD.eq.30) then
          IAVD = IAVD+1
        else
          IAVD = 0
        endif
```

```
        if (NMOLBL.gt.2) call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC ('IN',1,2,MOLBL,NMOLBL+1,NMOLBL)

c Differentiate.
250   elseif (IOPT.eq.5) then
        call DIFRNC (Z1,Z2,ZNULL1,NP,DT)
        if (IAVD.eq.2.or.IAVD.eq.3.or.IAVD.eq.4) then
          IAVD = IAVD-1
        else
          IAVD = 0
        endif
        if (NMOLBL.gt.2) call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC ('DF',1,2,MOLBL,NMOLBL+1,NMOLBL)

c Remove geophone response.
260   elseif (IOPT.eq.6) then
        if (NBUF.eq.1) then
          NF = 4
        else
          call RCI (2,BUF,NBUF,NF,IR)
          if (IR.ne.0) goto 8100
        endif
        if (NF.le.0) goto 8200
        if (ITRNS.ne.2.or.(TFRQ.eq.0..or.TDMP.eq.0.)) goto 8200
        if (INDEX(MOLBL,'GC').ne.0) goto 8200
        call GPHONE (Z1,Z2,ZC,ZNULL1,NP,DT,TFRQ,TDMP,NF)
        if (NMOLBL.gt.2) call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC ('GC(',1,3,MOLBL,NMOLBL+1,NMOLBL)
        call WCF (TFRQ,3,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCF (TDMP,3,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCI (NF,MOLBL,NMOLBL+1,NMOLBL)
        call WCC (')',1,1,MOLBL,NMOLBL+1,NMOLBL)
        continue

c Pad with temporary zeros (watch out for NP+NPAD+NPAD crashing into ISIZ).
270   elseif (IOPT.eq.7) then
        call RCF (2,BUF,NBUF,ZT,IR)
        if (IR.ne.0) goto 8100
        if (INDEX(MOLBL,'PZ').ne.0) goto 8200
        NZT = ZT/DT
        if (NZT.eq.0) goto 274
        NPAD = ABS(ZT)/DT
        do 271 I=NP+NPAD,1+NPAD,-1
          Z1(I) = Z1(I-NPAD)
271       continue
        do 272 I=1,NPAD
          Z1(I) = 0.
272       continue
        do 273 I=NP+NPAD+1,NP+NPAD+NPAD
          Z1(I) = 0.
273       continue
        NP = NP+NPAD+NPAD
        E8 = E8-DT*NPAD
        if (NMOLBL.gt.2) call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC ('PZ(',1,3,MOLBL,NMOLBL+1,NMOLBL)
        call WCF (ZT,3,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC (')',1,1,MOLBL,NMOLBL+1,NMOLBL)
274     continue

c Multiply by scale factor.
280   elseif (IOPT.eq.8) then
        if (INDEX(MOLBL,'SF').ne.0) goto 8200
        call RCF (2,BUF,NBUF,SF,IR)
        if (IR.ne.0) goto 8100
        do 281 I=1,NP
           if (Z1(I).ne.ZNULL1) Z1(I) = SF*Z1(I)
281        continue
        if (NMOLBL.gt.2) call WCC (',',1,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC ('SF(',1,3,MOLBL,NMOLBL+1,NMOLBL)
        call WCF (SF,3,1,MOLBL,NMOLBL+1,NMOLBL)
        call WCC (')',1,1,MOLBL,NMOLBL+1,NMOLBL)

      else
        goto 8100

      endif

300   if (.not.LPROMPT.and.NCOUNT.eq.NMO) then !finished preset options
        LPROMPT = .true.
        goto 100
      else
        goto 110
      endif

910   MOLBL = 'StartOver'
      return
920   if (NZT.le.0) return
      NP = NP-NPAD-NPAD
      E8 = E8+DT*NPAD
```

```
      do 921 I=1,NP
         Z1(I) = Z1(I+NPAD)
921      continue
      return


8020  write (6,98020)
98020 format (' *MVFBB ERROR* Can''t auto-rotate')
      goto 29
8100  write (6,98100)
98100 format (' *MVFBB ERROR* Can''t interpret input')
      goto 8999
8200  write (6,98200) IOPT
98200 format (' *MVFBB ERROR* Option = ',i1)
      goto 8999
8999  goto 110

      end

************************************************************* PADPW2 ***

      subroutine PADPW2 (Z1,ZNULL1,NP,DT)

c PADPW2 - [MUELLER.FS.SD]CSMSDLB
c Pad end of seismogram Z1 with zeroes up to power-of-2 samples.
c Warning - Watch out for NPPW2 crashing into ISIZ!

      dimension Z1(1)

      M = 0
1     M = M+1
      NPPW2 = 2**M
      if (NPPW2.lt.NP) goto 1
      if (NPPW2.eq.NP) return
      do 2 I=NP+1,NPPW2
         Z1(I) = 0.
2        continue
      NP = NPPW2
      return

      end

************************************************************* PARSEF ***

      subroutine PARSEF (F,IC2,IC1,ILB,IRB,IDT,ISC,IEND,LVFBB,ISTAT)

c PARSEF - [MUELLER.FS.SD]CSMSDLB
c Try to interpret character string F as a filename.
c F should contain one-or-more terminating blanks and no leading blanks.
c ISTAT = 0 for success; = 1-6 for problems.

      logical LVFBB
      character F*(*)

      IC2 = 0  !double colon
      IC1 = 0  !single colon
      ILB = 0  !left square bracket
      IRB = 0  !right square bracket
      IDT = 0  !dot
      ISC = 0  !semicolon
      IEND = 0 !end
      LVFBB = .false.

c IEND: If no terminating ' ' or if leading ' ' -> IEND=0, ISTAT=1, return.
10    ISTAT = 1
      I = INDEX(F,' ')
      if (I.eq.0.or.I.eq.1) return
      IEND = I-1
c IC2: If no '::' -> IC2=0, continue.
c       If wierd -> IC2=0, ISTAT=2, return.
20    ISTAT = 2
      I = INDEX(F(1:IEND),'::')
      if (I.eq.1.or.INDEX(F(I+1:IEND),'::').ne.0) return
      IC2 = I
c IC1: If no ':' and IC2=0 -> IC1=0, continue.
c       If wierd -> IC1=0, ISTAT=3, return.
30    ISTAT = 3
      if (IC2.gt.1.and.INDEX(F(1:IC2-1),':').ne.0) return
      if (IC2.eq.0) I = INDEX(F(1:IEND),':')
      if (IC2.gt.0) I = IC2+1+INDEX(F(IC2+2:IEND),':')
      if (I.eq.IC2+1.or.INDEX(F(I+1:IEND),':').ne.0) return
      IC1 = I
c ILB & IRB: If no '[' and no ']' -> ILB=IRB=IC1, continue.
c            If wierd -> ILB=0 or IRB=0, ISTAT=4, return.
40    ISTAT = 4
      if (IC1.gt.1.and.INDEX(F(1:IC1-1),'[').ne.0) return
      I = IC1+INDEX(F(IC1+1:IEND),'[')
      if (I.gt.IC1+1.or.INDEX(F(I+1:IEND),'[').ne.0) return
      ILB = I
      if (ILB.gt.1.and.INDEX(F(1:ILB-1),']').ne.0) return
```

```
      I = ILB+INDEX(F(ILB+1:IEND),']')
      if ((ILB.eq.IC1.and.I.ne.IC1).or.
     *    INDEX(F(I+1:IEND),']').ne.0) return
      IRB = I
c IDT: If no '.' or if wierd -> IDT=0, ISTAT=5, return.
50    ISTAT = 5
      I = IRB+INDEX(F(IRB+1:IEND),'.')
      if (I.lt.IRB+2.or.INDEX(F(I+1:IEND),'.').ne.0) return
      IDT = I
c ISC: If no ';' -> F(IEND+1:IEND+3)=';0 ', IEND=IEND+2, continue.
c      If wierd -> ISC=0, ISTAT=6, return.
60    ISTAT = 6
      I = IRB+INDEX(F(IRB+1:IEND),';')
      if (I.eq.IRB) then
        F(IEND+1:IEND+3) = ';0 '
        I = IEND+1
        IEND = IEND+2
      endif
      if (I.lt.IDT.or.INDEX(F(I+1:IEND),';').ne.0) return
      ISC = I
      call VFBBF (F(IRB+1:IEND),LVFBB)
      ISTAT = 0
      return

      end

************************************************************ PSGM   ***

      subroutine PSGM (DT,Z,ZNULL,NZ,IPDEC,T1,T2,
     *                 XORG,YORG,XLEN,YLEN,CZ,TZ,
     *                 ALBL,NALBL,OLBL,NOLBL,LTR)

c PSGM - [MUELLER.FS.PLT]CSMPLTLB
c Plot evenly-sampled seismogram Z;
c  plot every IPDECth point from T1 to T2.
c Parameters:
c DT= delta T (sec).
c Z= array. ZNULL= null (not plotted). NZ= number of points in Z.
c IPDEC= plot decimation factor.
c T1,T2= min,max time to plot (sec).
c XORG,YORG,XLEN,YLEN,CZ,TZ= plot, character, and tick dimensions (inch).
c ALBL,NALBL,OLBL,NOLBL= abscissa,ordinate label.
c LTR= .true. if you want (unlabeled) top and right axes.
c Parameters returned in PSGMCOM:
c TMIN,TMAX= min,max time (sec).
c ZMIN,ZMAX= min,max amplitude (ordinate_unit).
c XSCL,YSCL= scale factor (inch/abscissa_unit,inch/ordinate_unit).
c Note: PSGM assumes time = 0 at Z(1).
c       User should annotate absolute time at Z(1) outside PSGM;
c       this is OK even if T1>0, because PSGM labels the time axis correctly.

      real Z(1)
      logical LTR,L1NULL,L2NULL
      character ALBL*(*),OLBL*(*)

      common /PSGMCOM/ TMIN,TMAX,ZMIN,ZMAX,XSCL,YSCL

      I1 = MAX(1,NINT(T1/DT)+1)
      if (DT*(I1-1).lt.T1) I1 = I1+1
      I2 = MIN(NZ,NINT(T2/DT)+1)
      if (DT*(I2-1).gt.T2) I2 = I2-1
c Time scale.
      TMIN = T1
      TMAX = T2
      call LNABS (0,TMIN,TMAX,XSCL,
     *            XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)
c Amplitude scale.
      ZMIN = Z(I1)
      ZMAX = ZMIN
      do 1 I=I1,I2
         ZZ = Z(I)
         if (ZZ.eq.ZNULL) goto 1
         if (ZZ.lt.ZMIN) ZMIN = ZZ
         if (ZZ.gt.ZMAX) ZMAX = ZZ
1        continue
      call LNORD (0,ZMIN,ZMAX,YSCL,
     *            XORG,YORG,XLEN,YLEN,CZ,TZ,OLBL,NOLBL,LTR)
c Plot Z using TMIN, ZMIN, XSCL, and YSCL.
      call P1 (0.,DT,Z,ZNULL,IPDEC,I1,I2,
     *         TMIN,ZMIN,XORG,YORG,XSCL,YSCL)
      return

      end

************************************************************ PSGM3  ***

      subroutine PSGM3 (DT,Z1,ZNULL1,Z2,ZNULL2,Z3,ZNULL3,NZ,IPDEC,T1,T2,
     *                  XORG,YORG,XLEN,YLEN,CZ,TZ,
     *                  ALBL,NALBL,OLBL,NOLBL,LTR)
```

```
c PSGM3 - [MUELLER.FS.PLT]CSMPLTLB
c Plot three-component evenly-sampled seismograms Z1,Z2,Z3;
c  plot every IPDECth point from T1 to T2.
c See subroutine PSGM for description of parameters and note about usage.

      real Z1(1),Z2(1),Z3(1)
      logical LTR,L1NULL,L2NULL
      character ALBL*(*),OLBL*(*)

      common /PSGMCOM/ TMIN,TMAX,ZMIN,ZMAX,XSCL,YSCL

      I1 = MAX(1,NINT(T1/DT)+1)
      if (DT*(I1-1).lt.T1) I1 = I1+1
      I2 = MIN(NZ,NINT(T2/DT)+1)
      if (DT*(I2-1).gt.T2) I2 = I2-1
c Time scale.
      TMIN = T1
      TMAX = T2
      call LNABS (0,TMIN,TMAX,XSCL,
     *            XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)
c Amplitude scale.
      ZEXT = 0.
      do 1 I=I1,I2
         ZZ = Z1(I)
         if (ABS(ZZ).gt.ZEXT.and.ZZ.ne.ZNULL1) ZEXT = ABS(ZZ)
         ZZ = Z2(I)
         if (ABS(ZZ).gt.ZEXT.and.ZZ.ne.ZNULL2) ZEXT = ABS(ZZ)
         ZZ = Z3(I)
         if (ABS(ZZ).gt.ZEXT.and.ZZ.ne.ZNULL3) ZEXT = ABS(ZZ)
1        continue
      ZMIN = -ZEXT
      SS = 6.*ZEXT
      YSCL = YLEN/SS
      S1 = 0.
      S2 = 0.9*SS
      IX = 1
      IY = ALOG10(S2)
      if (IY.gt.ALOG10(S2)) IY = IY-1
      S2 = IX*(10.**IY)
      YLENS = YSCL*S2
      YORGS = YORG+(YLEN-YLENS)/2.
      call LNORD (0,S1,S2,YSCL,
     *            XORG,YORGS,XLEN,YLENS,CZ,TZ,OLBL,NOLBL,LTR)
c Plot Z1,Z2,Z3 using TMIN, ZMIN, XSCL, and YSCL.
      XO = XORG
      YO = YORG+0.667*YLEN
      call P1 (0.,DT,Z1,ZNULL1,IPDEC,I1,I2,
     *         TMIN,ZMIN,XO,YO,XSCL,YSCL)
      YO = YORG+0.333*YLEN
      call P1 (0.,DT,Z2,ZNULL2,IPDEC,I1,I2,
     *         TMIN,ZMIN,XO,YO,XSCL,YSCL)
      YO = YORG
      call P1 (0.,DT,Z3,ZNULL3,IPDEC,I1,I2,
     *         TMIN,ZMIN,XO,YO,XSCL,YSCL)
      return

      end

************************************************************ PSPC   ***

      subroutine PSPC (DF,Z,NZ,FAX1,FAX2,F1,F2,LU,UMN,UMX,LL,
     *                 XORG,YORG,XLEN,YLEN,CZ,TZ,
     *                 ALBL,NALBL,OLBL,NOLBL,LTR)

c PSPC - [MUELLER.FS.PLT]CSMPLTLB
c Plot evenly-sampled real spectrum from F1 to F2.
c Parameters:
c DF= delta F (Hz).
c Z= spectrum array. NZ = number of points in Z.
c FAX1,FAX2= min,max frequency - axis range (Hz).
c F1,F2= min,max frequency - data range (Hz).
c LU,UMN,UMX= ordinate scaling (auto-scaling if LU=.false.).
c LL= 1(lin vs lin), 2(log vs log), 3(lin vs log), 4(log vs lin).
c XORG,YORG,XLEN,YLEN,CZ,TZ= plot, character, and tick dimensions (inches).
c ALBL,NALBL,OLBL,NOLBL= abscissa and ordinate labels.
c LTR= .true. if you want (unlabeled) top and right axes.
c Parameters returned in PSPCCOM:
c FMIN,FMAX= min,max frequency (Hz).
c ZMIN,ZMAX= min,max amplitude (ordinate_unit).
c XSCL,YSCL= scale factor (inches/unit or inches/decade).
c Note: PSPC assumes freq = 0 at Z(1) and skips this point.

      real Z(1)
      logical LU,LTR
      character ALBL*(*),OLBL*(*)

      common /PSPCCOM/ FMIN,FMAX,ZMIN,ZMAX,XSCL,YSCL

      I1 = MAX(2,NINT(F1/DF)+1)
      if (DF*(I1-1).lt.F1) I1 = I1+1
```

```
      I2 = MIN(NZ,NINT(F2/DF)+1)
      if (DF*(I2-1).gt.F2) I2 = I2-1
c Frequency scale.
      FMIN = FAX1
      FMAX = FAX2
      if (LL.eq.1.or.LL.eq.4) call LNABS
     * (0,FMIN,FMAX,XSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)
      if (LL.eq.2.or.LL.eq.3) call LGABS
     * (0,FMIN,FMAX,XSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)
c Amplitude scale.
      if (LU) then
        ZMIN = UMN
        ZMAX = UMX
      else
        ZMIN = Z(I1)
        ZMAX = ZMIN
        do 1 I=I1,I2
           ZZ = Z(I)
           if (ZZ.lt.ZMIN) ZMIN = ZZ
           if (ZZ.gt.ZMAX) ZMAX = ZZ
1          continue
      endif
      if (LL.eq.1.or.LL.eq.3) call LNORD
     * (0,ZMIN,ZMAX,YSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,OLBL,NOLBL,LTR)
      if (LL.eq.2.or.LL.eq.4) call LGORD
     * (0,ZMIN,ZMAX,YSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,OLBL,NOLBL,LTR)
c Plot Z using FMIN, ZMIN, XSCL, and YSCL.
c Check for points outside the boundary.
      ZNULL = 0.
      Y1 = YORG
      Y2 = YORG+YLEN
10    goto (11,12,13,14) LL
11    call P1CB (0.,DF,Z,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
12    call P2CB (0.,DF,Z,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
13    call P3CB (0.,DF,Z,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
14    call P4CB (0.,DF,Z,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return

      end

************************************************************ PSPC2  ***

      subroutine PSPC2 (DF,Z1,Z2,NZ,FAX1,FAX2,F1,F2,LU,UMN,UMX,LL,
     *                  XORG,YORG,XLEN,YLEN,CZ,TZ,
     *                  ALBL,NALBL,OLBL,NOLBL,LTR)

c PSPC2 - [MUELLER.FS.PLT]CSMPLTLB
c Plot two evenly-sampled real spectra from F1 to F2.
c See subroutine PSPC for description of parameters and note about usage.

      real Z1(1),Z2(1)
      logical LU,LTR
      character ALBL*(*),OLBL*(*)

      common /PSPCCOM/ FMIN,FMAX,ZMIN,ZMAX,XSCL,YSCL

      I1 = MAX(2,NINT(F1/DF)+1)
      if (DF*(I1-1).lt.F1) I1 = I1+1
      I2 = MIN(NZ,NINT(F2/DF)+1)
      if (DF*(I2-1).gt.F2) I2 = I2-1
c Frequency scale.
      FMIN = FAX1
      FMAX = FAX2
      if (LL.eq.1.or.LL.eq.4) call LNABS
     * (0,FMIN,FMAX,XSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)
      if (LL.eq.2.or.LL.eq.3) call LGABS
     * (0,FMIN,FMAX,XSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)
c Amplitude scale.
      if (LU) then
        ZMIN = UMN
        ZMAX = UMX
      else
        ZMIN = Z1(I1)
        ZMAX = ZMIN
        do 1 I=I1,I2
           ZZ = Z1(I)
           if (ZZ.lt.ZMIN) ZMIN = ZZ
           if (ZZ.gt.ZMAX) ZMAX = ZZ
           ZZ = Z2(I)
           if (ZZ.lt.ZMIN) ZMIN = ZZ
           if (ZZ.gt.ZMAX) ZMAX = ZZ
1          continue
      endif
```

```
      if (LL.eq.1.or.LL.eq.3) call LNORD
     * (0,ZMIN,ZMAX,YSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,OLBL,NOLBL,LTR)
      if (LL.eq.2.or.LL.eq.4) call LGORD
     * (0,ZMIN,ZMAX,YSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,OLBL,NOLBL,LTR)
c Plot Z1,Z2 using FMIN, ZMIN, XSCL, and YSCL.
c Check for points outside the boundary.
      ZNULL = 0.
      Y1 = YORG
      Y2 = YORG+YLEN
10    goto (11,12,13,14) LL
11    call P1CB (0.,DF,Z1,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      call P1CB (0.,DF,Z2,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
12    call P2CB (0.,DF,Z1,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      call P2CB (0.,DF,Z2,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
13    call P3CB (0.,DF,Z1,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      call P3CB (0.,DF,Z2,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
14    call P4CB (0.,DF,Z1,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      call P4CB (0.,DF,Z2,ZNULL,1,I1,I2,
     *           FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return

      end

************************************************************ PSPCG  ***

      subroutine PSPCG (FG1,FGR,Z,NZ,FAX1,FAX2,F1,F2,LU,UMN,UMX,LL,
     *                  XORG,YORG,XLEN,YLEN,CZ,TZ,
     *                  ALBL,NALBL,OLBL,NOLBL,LTR)

c PSPCG - [MUELLER.FS.PLT]CSMPLTLB
c Plot geometrically-sampled real spectrum from F1 to F2.
c Parameters:
c FG1= frequency at Z(1) (Hz).
c FGR= frequency ratio (FGR=freq(Z(I+1))/freq(Z(I))).
c Z= spectrum array. NZ = number of points in Z.
c FAX1,FAX2= min,max frequency - axis range (Hz).
c F1,F2= min,max frequency - data range (Hz).
c LU,UMN,UMX= ordinate scaling (auto-scaling if LU=.false.).
c LL= 1(lin vs lin), 2(log vs log), 3(lin vs log), 4(log vs lin).
c XORG,YORG,XLEN,YLEN,CZ,TZ= plot, character, and tick dimensions (inches).
c ALBL,NALBL,OLBL,NOLBL= abscissa and ordinate labels.
c LTR= .true. if you want (unlabeled) top and right axes.
c Parameters returned in PSPCCOM:
c FMIN,FMAX= min,max frequency (Hz).
c ZMIN,ZMAX= min,max amplitude (ordinate_unit).
c XSCL,YSCL= scale factor (inches/unit or inches/decade).

      real Z(1)
      logical LU,LTR
      character ALBL*(*),OLBL*(*)

      common /PSPCCOM/ FMIN,FMAX,ZMIN,ZMAX,XSCL,YSCL

      I1 = MAX(1,NINT(LOG10(F1/FG1)/LOG10(FGR))+1)
      if (FG1*FGR**(I1-1).lt.F1) I1 = I1+1
      I2 = MIN(NZ,NINT(LOG10(F2/FG1)/LOG10(FGR))+1)
      if (FG1*FGR**(I2-1).gt.F2) I2 = I2-1
c Frequency scale.
      FMIN = FAX1
      FMAX = FAX2
      if (LL.eq.1.or.LL.eq.4) call LNABS
     * (0,FMIN,FMAX,XSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)
      if (LL.eq.2.or.LL.eq.3) call LGABS
     * (0,FMIN,FMAX,XSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)
c Amplitude scale.
      if (LU) then
        ZMIN = UMN
        ZMAX = UMX
      else
        ZMIN = Z(I1)
        ZMAX = ZMIN
        do 1 I=I1,I2
           ZZ = Z(I)
           if (ZZ.lt.ZMIN) ZMIN = ZZ
           if (ZZ.gt.ZMAX) ZMAX = ZZ
1          continue
      endif
      if (LL.eq.1.or.LL.eq.3) call LNORD
     * (0,ZMIN,ZMAX,YSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,OLBL,NOLBL,LTR)
      if (LL.eq.2.or.LL.eq.4) call LGORD
```

```
     * (0,ZMIN,ZMAX,YSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,OLBL,NOLBL,LTR)
c Plot Z using FMIN, ZMIN, XSCL, and YSCL.
c Check for points outside the boundary.
      ZNULL = 0.
      Y1 = YORG
      Y2 = YORG+YLEN
10    goto (11,12,13,14) LL
11    call P1GCB (FG1,FGR,Z,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
12    call P2GCB (FG1,FGR,Z,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
13    call P3GCB (FG1,FGR,Z,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
14    call P4GCB (FG1,FGR,Z,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return

      end

************************************************************ PSPCG2 ***

      subroutine PSPCG2 (FG1,FGR,Z1,Z2,NZ,FAX1,FAX2,F1,F2,LU,UMN,UMX,LL,
     *                   XORG,YORG,XLEN,YLEN,CZ,TZ,
     *                   ALBL,NALBL,OLBL,NOLBL,LTR)

c PSPCG2 - [MUELLER.FS.PLT]CSMPLTLB
c Plot two geometrically-sampled real spectra from F1 to F2.
c See subroutine PSPCG for description of parameters and note about usage.

      real Z1(1),Z2(1)
      logical LU,LTR
      character ALBL*(*),OLBL*(*)

      common /PSPCCOM/ FMIN,FMAX,ZMIN,ZMAX,XSCL,YSCL

      I1 = MAX(1,NINT(LOG10(F1/FG1)/LOG10(FGR))+1)
      if (FG1*FGR**(I1-1).lt.F1) I1 = I1+1
      I2 = MIN(NZ,NINT(LOG10(F2/FG1)/LOG10(FGR))+1)
      if (FG1*FGR**(I2-1).gt.F2) I2 = I2-1
c Frequency scale.
      FMIN = FAX1
      FMAX = FAX2
      if (LL.eq.1.or.LL.eq.4) call LNABS
     * (0,FMIN,FMAX,XSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)
      if (LL.eq.2.or.LL.eq.3) call LGABS
     * (0,FMIN,FMAX,XSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,ALBL,NALBL,LTR)
c Amplitude scale.
      if (LU) then
        ZMIN = UMN
        ZMAX = UMX
      else
        ZMIN = Z1(I1)
        ZMAX = ZMIN
        do 1 I=I1,I2
           ZZ = Z1(I)
           if (ZZ.lt.ZMIN) ZMIN = ZZ
           if (ZZ.gt.ZMAX) ZMAX = ZZ
           ZZ = Z2(I)
           if (ZZ.lt.ZMIN) ZMIN = ZZ
           if (ZZ.gt.ZMAX) ZMAX = ZZ
1          continue
      endif
      if (LL.eq.1.or.LL.eq.3) call LNORD
     * (0,ZMIN,ZMAX,YSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,OLBL,NOLBL,LTR)
      if (LL.eq.2.or.LL.eq.4) call LGORD
     * (0,ZMIN,ZMAX,YSCL,XORG,YORG,XLEN,YLEN,CZ,TZ,OLBL,NOLBL,LTR)
c Plot Z1,Z2 using FMIN, ZMIN, XSCL, and YSCL.
c Check for points outside the boundary.
      ZNULL = 0.
      Y1 = YORG
      Y2 = YORG+YLEN
10    goto (11,12,13,14) LL
11    call P1GCB (FG1,FGR,Z1,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      call P1GCB (FG1,FGR,Z2,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
12    call P2GCB (FG1,FGR,Z1,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      call P2GCB (FG1,FGR,Z2,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
13    call P3GCB (FG1,FGR,Z1,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      call P3GCB (FG1,FGR,Z2,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return
```

```
14    call P4GCB (FG1,FGR,Z1,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      call P4GCB (FG1,FGR,Z2,ZNULL,1,I1,I2,
     *            FMIN,ZMIN,XORG,YORG,XSCL,YSCL,Y1,Y2)
      return

      end

************************************************************ P1     ***

      subroutine P1 (A1,AD,B,BNULL,IPDEC,I1,I2,
     *               AORG,BORG,XORG,YORG,XSCL,YSCL)

c P1 - [MUELLER.FS.PLT]CSMPLTLB
c Plot evenly-spaced array B (linear vs linear);
c  plot every IPDECth point from index I1 to I2.
c Parameters:
c A1 = abscissa of B(1).
c AD = abscissa_unit/point (e.g. delta T for evenly-sampled seismogram).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/abscissa_unit,inch/ordinate_unit).

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = A1+AD*(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*(AI-AORG)
         Y = YORG+YSCL*(BI-BORG)
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         call PLOT (X,Y,+2)
1        L2NULL = L1NULL
2        continue
      return

      end

************************************************************ P2     ***

      subroutine P2 (A1,AD,B,BNULL,IPDEC,I1,I2,
     *               AORG,BORG,XORG,YORG,XSCL,YSCL)

c P2 - [MUELLER.FS.PLT]CSMPLTLB
c Plot evenly-spaced array B (log vs log);
c  plot every IPDECth point from index I1 to I2.
c Parameters:
c A1 = abscissa of B(1).
c AD = abscissa_unit/point (e.g. delta T for evenly-sampled seismogram).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/decade,inch/decade).

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = A1+AD*(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*ALOG10(AI/AORG)
         Y = YORG+YSCL*ALOG10(BI/BORG)
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         call PLOT (X,Y,+2)
1        L2NULL = L1NULL
2        continue
      return

      end

************************************************************ P3     ***

      subroutine P3 (A1,AD,B,BNULL,IPDEC,I1,I2,
```

```
     *               AORG,BORG,XORG,YORG,XSCL,YSCL)

c P3 - [MUELLER.FS.PLT]CSMPLTLB
c Plot evenly-spaced array B (linear vs log);
c  plot every IPDECth point from index I1 to I2.
c Parameters:
c A1 = abscissa of B(1).
c AD = abscissa_unit/point (e.g. delta T for evenly-sampled seismogram).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/decade,inch/ordinate_unit).

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = A1+AD*(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*ALOG10(AI/AORG)
         Y = YORG+YSCL*(BI-BORG)
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         call PLOT (X,Y,+2)
1        L2NULL = L1NULL
2        continue
      return

      end

************************************************************* P4     ***

      subroutine P4 (A1,AD,B,BNULL,IPDEC,I1,I2,
     *               AORG,BORG,XORG,YORG,XSCL,YSCL)

c P4 - [MUELLER.FS.PLT]CSMPLTLB
c Plot evenly-spaced array B (log vs linear);
c  plot every IPDECth point from index I1 to I2.
c Parameters:
c A1 = abscissa of B(1).
c AD = abscissa_unit/point (e.g. delta T for evenly-sampled seismogram).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/abscissa_unit,inch/decade).

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = A1+AD*(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*(AI-AORG)
         Y = YORG+YSCL*ALOG10(BI/BORG)
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         call PLOT (X,Y,+2)
1        L2NULL = L1NULL
2        continue
      return

      end

************************************************************* P1CB   ***

      subroutine P1CB (A1,AD,B,BNULL,IPDEC,I1,I2,
     *                 AORG,BORG,XORG,YORG,XSCL,YSCL,Y1,Y2)

c P1CB - [MUELLER.FS.PLT]CSMPLTLB
c Plot evenly-spaced array B (linear vs. linear);
c  plot every IPDECth point from index I1 to I2.
c  Like P1, but checks for points outside the plotting boundary (Y1,Y2).
c Parameters:
c A1 = abscissa of B(1).
c AD = abscissa_unit/point (e.g. delta T for evenly-sampled seismogram).
c B = array. BNULL = null (not plotted).
```

```
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/abscissa_unit,inch/ordinate_unit).
c Y1,Y2 = don't plot Y<Y1 or Y>Y2.

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = A1+AD*(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*(AI-AORG)
         Y = YORG+YSCL*(BI-BORG)
         ICB1 = 0
         if (Y.lt.Y1) ICB1 = -1
         if (Y.gt.Y2) ICB1 = +1
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         if (ICB1.eq.0.and.ICB2.eq.0) then
           call PLOT (X,Y,+2)
         else
           SLOPE = (Y-YOLD)/(X-XOLD)
           if (ICB1.eq.-1.or.ICB2.eq.-1) YB = Y1
           if (ICB1.eq.+1.or.ICB2.eq.+1) YB = Y2
           XB = (YB-YOLD)/SLOPE+XOLD
           if (ICB1.eq.0) then
             call PLOT (XB,YB,+3)
             call PLOT (X,Y,+2)
           endif
           if (ICB2.eq.0) then
             call PLOT (XOLD,YOLD,+3)
             call PLOT (XB,YB,+2)
           endif
         endif
1        L2NULL = L1NULL
         XOLD = X
         YOLD = Y
         ICB2 = ICB1
2        continue
      return

      end

************************************************************ P2CB   ***

      subroutine P2CB (A1,AD,B,BNULL,IPDEC,I1,I2,
     *                 AORG,BORG,XORG,YORG,XSCL,YSCL,Y1,Y2)

c P2CB - [MUELLER.FS.PLT]CSMPLTLB
c Plot evenly-spaced array B (log vs. log);
c  plot every IPDECth point from index I1 to I2.
c  Like P2, but checks for points outside the plotting boundary (Y1,Y2).
c Parameters:
c A1 = abscissa of B(1).
c AD = abscissa_unit/point (e.g. delta T for evenly-sampled seismogram).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/decade,inch/decade).
c Y1,Y2 = don't plot Y<Y1 or Y>Y2.

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = A1+AD*(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*ALOG10(AI/AORG)
         Y = YORG+YSCL*ALOG10(BI/BORG)
         ICB1 = 0
         if (Y.lt.Y1) ICB1 = -1
         if (Y.gt.Y2) ICB1 = +1
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         if (ICB1.eq.0.and.ICB2.eq.0) then
           call PLOT (X,Y,+2)
```

```
          else
            SLOPE = (Y-YOLD)/(X-XOLD)
            if (ICB1.eq.-1.or.ICB2.eq.-1) YB = Y1
            if (ICB1.eq.+1.or.ICB2.eq.+1) YB = Y2
            XB = (YB-YOLD)/SLOPE+XOLD
            if (ICB1.eq.0) then
              call PLOT (XB,YB,+3)
              call PLOT (X,Y,+2)
            endif
            if (ICB2.eq.0) then
              call PLOT (XOLD,YOLD,+3)
              call PLOT (XB,YB,+2)
            endif
          endif
1         L2NULL = L1NULL
          XOLD = X
          YOLD = Y
          ICB2 = ICB1
2         continue
      return

      end

************************************************************* P3CB   ***

      subroutine P3CB (A1,AD,B,BNULL,IPDEC,I1,I2,
     *                 AORG,BORG,XORG,YORG,XSCL,YSCL,Y1,Y2)

c P3CB - [MUELLER.FS.PLT]CSMPLTLB
c Plot evenly-spaced array B (linear vs. log);
c  plot every IPDECth point from index I1 to I2.
c  Like P3, but checks for points outside the plotting boundary (Y1,Y2).
c Parameters:
c A1 = abscissa of B(1).
c AD = abscissa_unit/point (e.g. delta T for evenly-sampled seismogram).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/decade,inch/ordinate_unit).
c Y1,Y2 = don't plot Y<Y1 or Y>Y2.

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = A1+AD*(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*ALOG10(AI/AORG)
         Y = YORG+YSCL*(BI-BORG)
         ICB1 = 0
         if (Y.lt.Y1) ICB1 = -1
         if (Y.gt.Y2) ICB1 = +1
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         if (ICB1.eq.0.and.ICB2.eq.0) then
           call PLOT (X,Y,+2)
         else
           SLOPE = (Y-YOLD)/(X-XOLD)
           if (ICB1.eq.-1.or.ICB2.eq.-1) YB = Y1
           if (ICB1.eq.+1.or.ICB2.eq.+1) YB = Y2
           XB = (YB-YOLD)/SLOPE+XOLD
           if (ICB1.eq.0) then
             call PLOT (XB,YB,+3)
             call PLOT (X,Y,+2)
           endif
           if (ICB2.eq.0) then
             call PLOT (XOLD,YOLD,+3)
             call PLOT (XB,YB,+2)
           endif
         endif
1        L2NULL = L1NULL
         XOLD = X
         YOLD = Y
         ICB2 = ICB1
2        continue
      return

      end

************************************************************* P4CB   ***

      subroutine P4CB (A1,AD,B,BNULL,IPDEC,I1,I2,
     *                 AORG,BORG,XORG,YORG,XSCL,YSCL,Y1,Y2)
```

```
c P4CB - [MUELLER.FS.PLT]CSMPLTLB
c Plot evenly-spaced array B (log vs. linear);
c  plot every IPDECth point from index I1 to I2.
c  Like P4, but checks for points outside the plotting boundary (Y1,Y2).
c Parameters:
c A1 = abscissa of B(1).
c AD = abscissa_unit/point (e.g. delta T for evenly sampled seismogram).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/abscissa_unit,inch/decade).
c Y1,Y2 = don't plot Y<Y1 or Y>Y2.

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = A1+AD*(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*(AI-AORG)
         Y = YORG+YSCL*ALOG10(BI/BORG)
         ICB1 = 0
         if (Y.lt.Y1) ICB1 = -1
         if (Y.gt.Y2) ICB1 = +1
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         if (ICB1.eq.0.and.ICB2.eq.0) then
           call PLOT (X,Y,+2)
         else
           SLOPE = (Y-YOLD)/(X-XOLD)
           if (ICB1.eq.-1.or.ICB2.eq.-1) YB = Y1
           if (ICB1.eq.+1.or.ICB2.eq.+1) YB = Y2
           XB = (YB-YOLD)/SLOPE+XOLD
           if (ICB1.eq.0) then
             call PLOT (XB,YB,+3)
             call PLOT (X,Y,+2)
           endif
           if (ICB2.eq.0) then
             call PLOT (XOLD,YOLD,+3)
             call PLOT (XB,YB,+2)
           endif
         endif
1        L2NULL = L1NULL
         XOLD = X
         YOLD = Y
         ICB2 = ICB1
2        continue
      return

      end


************************************************************ P1G    ***

      subroutine P1G (AG1,AGR,B,BNULL,IPDEC,I1,I2,
     *                AORG,BORG,XORG,YORG,XSCL,YSCL)

c P1G - [MUELLER.FS.PLT]CSMPLTLB
c Plot geometrically-spaced array B (linear vs. linear);
c  plot every IPDECth point from index I1 to I2.
c Parameters:
c AG1 = abscissa of B(1).
c AGR = abscissa ratio (AGR=abscissa(B(I+1))/abscissa(B(I))).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/abscissa_unit,inch/ordinate_unit).

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = AG1*AGR**(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*(AI-AORG)
         Y = YORG+YSCL*(BI-BORG)
         if (L1NULL.or.L2NULL) then
```

```
            call PLOT (X,Y,+3)
            goto 1
          endif
          call PLOT (X,Y,+2)
1         L2NULL = L1NULL
2         continue
      return

      end

************************************************************ P2G    ***

      subroutine P2G (AG1,AGR,B,BNULL,IPDEC,I1,I2,
     *                AORG,BORG,XORG,YORG,XSCL,YSCL)

c P2G - [MUELLER.FS.PLT]CSMPLTLB
c Plot geometrically-spaced array B (log vs. log);
c  plot every IPDECth point from index I1 to I2.
c Parameters:
c AG1 = abscissa of B(1).
c AGR = abscissa ratio (AGR=abscissa(B(I+1))/abscissa(B(I))).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/decade,inch/decade).

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = AG1*AGR**(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*ALOG10(AI/AORG)
         Y = YORG+YSCL*ALOG10(BI/BORG)
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         call PLOT (X,Y,+2)
1        L2NULL = L1NULL
2        continue
      return

      end

************************************************************ P3G    ***

      subroutine P3G (AG1,AGR,B,BNULL,IPDEC,I1,I2,
     *                AORG,BORG,XORG,YORG,XSCL,YSCL)

c P3G - [MUELLER.FS.PLT]CSMPLTLB
c Plot geometrically-spaced array B (linear vs. log);
c  plot every IPDECth point from index I1 to I2.
c Parameters:
c AG1 = abscissa of B(1).
c AGR = abscissa ratio (AGR=abscissa(B(I+1))/abscissa(B(I))).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/decade,inch/ordinate_unit).

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = AG1*AGR**(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*ALOG10(AI/AORG)
         Y = YORG+YSCL*(BI-BORG)
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         call PLOT (X,Y,+2)
1        L2NULL = L1NULL
2        continue
      return

      end
```

```
*************************************************************** P4G    ***

      subroutine P4G (AG1,AGR,B,BNULL,IPDEC,I1,I2,
     *                AORG,BORG,XORG,YORG,XSCL,YSCL)

c P4G - [MUELLER.FS.PLT]CSMPLTLB
c Plot geometrically-spaced array B (log vs. linear);
c  plot every IPDECth point from index I1 to I2.
c Parameters:
c AG1 = abscissa of B(1).
c AGR = abscissa ratio (AGR=abscissa(B(I+1))/abscissa(B(I))).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/abscissa_unit,inch/decade).

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = AG1*AGR**(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*(AI-AORG)
         Y = YORG+YSCL*ALOG10(BI/BORG)
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         call PLOT (X,Y,+2)
1        L2NULL = L1NULL
2        continue
      return

      end

*************************************************************** P1GCB  ***

      subroutine P1GCB (AG1,AGR,B,BNULL,IPDEC,I1,I2,
     *                  AORG,BORG,XORG,YORG,XSCL,YSCL,Y1,Y2)

c P1GCB - [MUELLER.FS.PLT]CSMPLTLB
c Plot geometrically-spaced array B (linear vs. linear);
c  plot every IPDECth point from index I1 to I2.
c  Like P1G, but checks for points outside the plotting boundary (Y1,Y2).
c Parameters:
c AG1 = abscissa of B(1).
c AGR = abscissa ratio (AGR=abscissa(B(I+1))/abscissa(B(I))).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/abscissa_unit,inch/ordinate_unit).
c Y1,Y2 = don't plot Y<Y1 or Y>Y2.

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = AG1*AGR**(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*(AI-AORG)
         Y = YORG+YSCL*(BI-BORG)
         ICB1 = 0
         if (Y.lt.Y1) ICB1 = -1
         if (Y.gt.Y2) ICB1 = +1
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         if (ICB1.eq.0.and.ICB2.eq.0) then
           call PLOT (X,Y,+2)
         else
           SLOPE = (Y-YOLD)/(X-XOLD)
           if (ICB1.eq.-1.or.ICB2.eq.-1) YB = Y1
           if (ICB1.eq.+1.or.ICB2.eq.+1) YB = Y2
           XB = (YB-YOLD)/SLOPE+XOLD
           if (ICB1.eq.0) then
             call PLOT (XB,YB,+3)
             call PLOT (X,Y,+2)
           endif
           if (ICB2.eq.0) then
```

```
            call PLOT (XOLD,YOLD,+3)
            call PLOT (XB,YB,+2)
          endif
        endif
1       L2NULL = L1NULL
        XOLD = X
        YOLD = Y
        ICB2 = ICB1
2       continue
      return

      end

************************************************************ P2GCB  ***

      subroutine P2GCB (AG1,AGR,B,BNULL,IPDEC,I1,I2,
     *                  AORG,BORG,XORG,YORG,XSCL,YSCL,Y1,Y2)

c P2GCB - [MUELLER.FS.PLT]CSMPLTLB
c Plot geometrically-spaced array B (log vs. log);
c  plot every IPDECth point from index I1 to I2.
c  Like P2G, but checks for points outside the plotting boundary (Y1,Y2).
c Parameters:
c AG1 = abscissa of B(1).
c AGR = abscissa ratio (AGR=abscissa(B(I+1))/abscissa(B(I))).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/decade,inch/decade).
c Y1,Y2 = don't plot Y<Y1 or Y>Y2.

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = AG1*AGR**(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*ALOG10(AI/AORG)
         Y = YORG+YSCL*ALOG10(BI/BORG)
         ICB1 = 0
         if (Y.lt.Y1) ICB1 = -1
         if (Y.gt.Y2) ICB1 = +1
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         if (ICB1.eq.0.and.ICB2.eq.0) then
           call PLOT (X,Y,+2)
         else
           SLOPE = (Y-YOLD)/(X-XOLD)
           if (ICB1.eq.-1.or.ICB2.eq.-1) YB = Y1
           if (ICB1.eq.+1.or.ICB2.eq.+1) YB = Y2
           XB = (YB-YOLD)/SLOPE+XOLD
           if (ICB1.eq.0) then
             call PLOT (XB,YB,+3)
             call PLOT (X,Y,+2)
           endif
           if (ICB2.eq.0) then
             call PLOT (XOLD,YOLD,+3)
             call PLOT (XB,YB,+2)
           endif
         endif
1        L2NULL = L1NULL
         XOLD = X
         YOLD = Y
         ICB2 = ICB1
2        continue
      return

      end

************************************************************ P3GCB  ***

      subroutine P3GCB (AG1,AGR,B,BNULL,IPDEC,I1,I2,
     *                  AORG,BORG,XORG,YORG,XSCL,YSCL,Y1,Y2)

c P3GCB - [MUELLER.FS.PLT]CSMPLTLB
c Plot geometrically-spaced array B (linear vs. log);
c  plot every IPDECth point from index I1 to I2.
c  Like P3G, but checks for points outside the plotting boundary (Y1,Y2).
c Parameters:
c AG1 = abscissa of B(1).
c AGR = abscissa ratio (AGR=abscissa(B(I+1))/abscissa(B(I))).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
```

```
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/decade,inch/ordinate_unit).
c Y1,Y2 = don't plot Y<Y1 or Y>Y2.

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = AG1*AGR**(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*ALOG10(AI/AORG)
         Y = YORG+YSCL*(BI-BORG)
         ICB1 = 0
         if (Y.lt.Y1) ICB1 = -1
         if (Y.gt.Y2) ICB1 = +1
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         if (ICB1.eq.0.and.ICB2.eq.0) then
           call PLOT (X,Y,+2)
         else
           SLOPE = (Y-YOLD)/(X-XOLD)
           if (ICB1.eq.-1.or.ICB2.eq.-1) YB = Y1
           if (ICB1.eq.+1.or.ICB2.eq.+1) YB = Y2
           XB = (YB-YOLD)/SLOPE+XOLD
           if (ICB1.eq.0) then
             call PLOT (XB,YB,+3)
             call PLOT (X,Y,+2)
           endif
           if (ICB2.eq.0) then
             call PLOT (XOLD,YOLD,+3)
             call PLOT (XB,YB,+2)
           endif
         endif
1        L2NULL = L1NULL
         XOLD = X
         YOLD = Y
         ICB2 = ICB1
2        continue
      return

      end

************************************************************ P4GCB  ***

      subroutine P4GCB (AG1,AGR,B,BNULL,IPDEC,I1,I2,
     *                  AORG,BORG,XORG,YORG,XSCL,YSCL,Y1,Y2)

c P4GCB - [MUELLER.FS.PLT]CSMPLTLB
c Plot geometrically-spaced array B (log vs. linear);
c  plot every IPDECth point from index I1 to I2.
c  Like P4G, but checks for points outside the plotting boundary (Y1,Y2).
c Parameters:
c AG1 = abscissa of B(1).
c AGR = abscissa ratio (AGR=abscissa(B(I+1))/abscissa(B(I))).
c B = array. BNULL = null (not plotted).
c IPDEC = plot decimation factor.
c I1,I2 = min,max index to plot.
c AORG,BORG = abscissa,ordinate corresponding to XORG,YORG.
c XORG,YORG = origin (inch from lower-lefthand corner).
c XSCL,YSCL = scale factor (inch/abscissa_unit,inch/decade).
c Y1,Y2 = don't plot Y<Y1 or Y>Y2.

      dimension B(1)
      logical L1NULL,L2NULL

      L2NULL = .true.
      do 2 I=I1,I2,IPDEC
         AI = AG1*AGR**(I-1)
         BI = B(I)
         L1NULL = .false.
         if (BI.eq.BNULL) L1NULL = .true.
         X = XORG+XSCL*(AI-AORG)
         Y = YORG+YSCL*ALOG10(BI/BORG)
         ICB1 = 0
         if (Y.lt.Y1) ICB1 = -1
         if (Y.gt.Y2) ICB1 = +1
         if (L1NULL.or.L2NULL) then
           call PLOT (X,Y,+3)
           goto 1
         endif
         if (ICB1.eq.0.and.ICB2.eq.0) then
           call PLOT (X,Y,+2)
         else
           SLOPE = (Y-YOLD)/(X-XOLD)
```

```
            if (ICB1.eq.-1.or.ICB2.eq.-1) YB = Y1
            if (ICB1.eq.+1.or.ICB2.eq.+1) YB = Y2
            XB = (YB-YOLD)/SLOPE+XOLD
            if (ICB1.eq.0) then
              call PLOT (XB,YB,+3)
              call PLOT (X,Y,+2)
            endif
            if (ICB2.eq.0) then
              call PLOT (XOLD,YOLD,+3)
              call PLOT (XB,YB,+2)
            endif
          endif
1         L2NULL = L1NULL
          XOLD = X
          YOLD = Y
          ICB2 = ICB1
2         continue
      return

      end

************************************************************ QPF    ***

      subroutine QPF (TEXT,NTEXT,F)

c QPF - [MUELLER.FS.GEN]CSMGENLB
c Prompt for real parameter change.
c NTRY prevents runaway "batch" jobs.

      character TEXT*(*),BUF*80

      BUF = TEXT(1:NTEXT)
      call WCC (' = ',1,3,BUF,NTEXT+1,NBUF)
      call WCF (F,5,1,BUF,NBUF+1,NBUF)
      NTRY = 0
1     NTRY = NTRY+1
      if (NTRY.gt.3) stop '*QPF ERROR* NTRY>3'
      write (6,90001) BUF(1:NBUF)
90001 format (' ',a)
      write (6,90002)
90002 format (' Type new value [f;<cr>=OK]: ',$)
      read (5,90003,err=1) N,FF
90003 format (q,f15.0)
      if (N.ne.0) F = FF
      return

      end

************************************************************ QPF2   ***

      subroutine QPF2 (TEXT,NTEXT,F1,F2)

c QPF2 - [MUELLER.FS.GEN]CSMGENLB
c Prompt for two real parameter changes.
c NTRY prevents runaway "batch" jobs.

      character TEXT*(*),BUF*80

      BUF = TEXT(1:NTEXT)
      call WCC (' = ',1,3,BUF,NTEXT+1,NBUF)
      call WCF (F1,5,1,BUF,NBUF+1,NBUF)
      call WCC (',',1,1,BUF,NBUF+1,NBUF)
      call WCF (F2,5,1,BUF,NBUF+1,NBUF)
      NTRY = 0
1     NTRY = NTRY+1
      if (NTRY.gt.3) stop '*QPF2 ERROR* NTRY>3'
      write (6,90001) BUF(1:NBUF)
90001 format (' ',a)
      write (6,90002)
90002 format (' Type new values [2f;<cr>=OK]: ',$)
      read (5,90003,err=1) N,FF1,FF2
90003 format (q,2f15.0)
      if (N.ne.0) then
        F1 = FF1
        F2 = FF2
      endif
      return

      end

************************************************************ QPI    ***

      subroutine QPI (TEXT,NTEXT,I)

c QPI - [MUELLER.FS.GEN]CSMGENLB
c Prompt for integer parameter change.
c NTRY prevents runaway "batch" jobs.

      character TEXT*(*),BUF*80
```

```
      BUF = TEXT(1:NTEXT)
      call WCC (' = ',1,3,BUF,NTEXT+1,NBUF)
      call WCI (I,BUF,NBUF+1,NBUF)
      NTRY = 0
1     NTRY = NTRY+1
      if (NTRY.gt.3) stop '*QPI ERROR* NTRY>3'
      write (6,90001) BUF(1:NBUF)
90001 format (' ',a)
      write (6,90002)
90002 format (' Type new value [i;<cr>=OK]: ',$)
      read (5,90003,err=1) N,II
90003 format (q,i12)
      if (N.ne.0) I = II
      return

      end

************************************************************ QPI2   ***

      subroutine QPI2 (TEXT,NTEXT,I1,I2)

c QPI2 - [MUELLER.FS.GEN]CSMGENLB
c Prompt for two integer parameter changes.
c NTRY prevents runaway "batch" jobs.

      character TEXT*(*),BUF*80

      BUF = TEXT(1:NTEXT)
      call WCC (' = ',1,3,BUF,NTEXT+1,NBUF)
      call WCI (I1,BUF,NBUF+1,NBUF)
      call WCC (',',1,1,BUF,NBUF+1,NBUF)
      call WCI (I2,BUF,NBUF+1,NBUF)
      NTRY = 0
1     NTRY = NTRY+1
      if (NTRY.gt.3) stop '*QPI2 ERROR* NTRY>3'
      write (6,90001) BUF(1:NBUF)
90001 format (' ',a)
      write (6,90002)
90002 format (' Type new values [2i;<cr>=OK]: ',$)
      read (5,90003,err=1) N,II1,II2
90003 format (q,2i12)
      if (N.ne.0) then
        I1 = II1
        I2 = II2
      endif
      return

      end

************************************************************ QPL    ***

      subroutine QPL (TEXT,NTEXT,L)

c QPL - [MUELLER.FS.GEN]CSMGENLB
c Prompt for logical parameter change.
c NTRY prevents runaway "batch" jobs.

      character TEXT*(*),BUF*80,C1*1

      BUF = TEXT(1:NTEXT)
      call WCC (' = ',1,3,BUF,NTEXT+1,NBUF)
      if (L) then
        call WCC ('YES',1,3,BUF,NBUF+1,NBUF)
      else
        call WCC ('NO',1,2,BUF,NBUF+1,NBUF)
      endif
      NTRY = 0
1     NTRY = NTRY+1
      if (NTRY.gt.3) stop '*QPL ERROR* NTRY>3'
      write (6,90001) BUF(1:NBUF)
90001 format (' ',a)
      write (6,90002)
90002 format (' Type new value [YorN;<cr>=OK]: ',$)
      read (5,90003,err=1) N,C1
90003 format (q,a)
      if (N.eq.0) return
      if (C1.eq.'Y'.or.C1.eq.'y') then
        L = .true.
      elseif (C1.eq.'N'.or.C1.eq.'n') then
        L = .false.
      else
        goto 1
      endif
      return

      end

************************************************************ RCC    ***

      subroutine RCC (KPARM,CSTR,NCSTR,C,NC)
```

```
c RCC - [MUELLER.FS.GEN]CSMGENLB
c Read character string C(1:NC), the KPARMth element in CSTR(1:NCSTR).
c  CSTR contains one-or-more mixed-type fields
c  separated by blanks or commas to the right of an optional '='.
c  KPARMth field contains C to the right of an optional '>'.
c If the KPARMth field is nonexistent or empty, RCC sets NC=0 and returns.
c  This simulates a <carriage-return> in a Q-format read.
c  (An empty field must be separated by commas, not blanks.)

      character CSTR*(*),C*(*)

c Eliminate trailing white space.
      N = NCSTR+1
1     N = N-1
      if (CSTR(N:N).eq.' '.or.CSTR(N:N).eq.',') goto 1
c Find optional '='.
      I2 = INDEX(CSTR(1:N),'=')
c Find start and end of KPARMth field.
      do 4 K=1,KPARM
         I1 = I2
2        I2 = I1
3        I1 = I1+1
         if (CSTR(I1:I1).eq.' ') goto 3
         if (CSTR(I2:I2).eq.' '.and.CSTR(I1:I1).eq.',') goto 2
         IBLANK = INDEX(CSTR(I1:N),' ')
         ICOMMA = INDEX(CSTR(I1:N),',')
         if (IBLANK.eq.0.and.ICOMMA.eq.0.and.K.lt.KPARM) goto 801
         if (IBLANK.eq.0) IBLANK = N-I1+2
         if (ICOMMA.eq.0) ICOMMA = N-I1+2
         I2 = I1+MIN(IBLANK,ICOMMA)-1
4        continue
c Check for empty field.
      if (I2.eq.I1) goto 801
c Find optional '>'.
      I1 = I1+INDEX(CSTR(I1:I2),'>')
c Get C.
      C = CSTR(I1:I2-1)
      NC = I2-I1
      return
c Errors.
801   NC = 0
      return

      end

************************************************************ RCF    ***

      subroutine RCF (KPARM,CSTR,NCSTR,F,ISTAT)

c RCF - [MUELLER.FS.GEN]CSMGENLB
c Read real F, the KPARMth element in CSTR(1:NCSTR).
c  CSTR contains one-or-more mixed-type fields
c  separated by blanks or commas to the right of an optional '='.
c  KPARMth field contains F to the right of an optional '>'.
c ISTAT= 0 for success;
c      = 1 if CSTR contains fewer than KPARM fields;
c      = 2 if KPARMth field is empty;
c      = 3 if KPARMth field is unreadable;
c      if ISTAT>0, return with F unchanged.

      character CSTR*(*)

      ISTAT = 0
c Eliminate trailing white space.
      N = NCSTR+1
1     N = N-1
      if (CSTR(N:N).eq.' '.or.CSTR(N:N).eq.',') goto 1
c Find optional '='.
      I2 = INDEX(CSTR(1:N),'=')
c Find start and end of KPARMth field.
      do 4 K=1,KPARM
         I1 = I2
2        I2 = I1
3        I1 = I1+1
         if (CSTR(I1:I1).eq.' ') goto 3
         if (CSTR(I2:I2).eq.' '.and.CSTR(I1:I1).eq.',') goto 2
         IBLANK = INDEX(CSTR(I1:N),' ')
         ICOMMA = INDEX(CSTR(I1:N),',')
         if (IBLANK.eq.0.and.ICOMMA.eq.0.and.K.lt.KPARM) goto 801
         if (IBLANK.eq.0) IBLANK = N-I1+2
         if (ICOMMA.eq.0) ICOMMA = N-I1+2
         I2 = I1+MIN(IBLANK,ICOMMA)-1
4        continue
c Check for empty field.
      if (I2.eq.I1) goto 802
c Find optional '>'.
      I1 = I1+INDEX(CSTR(I1:I2),'>')
c Get F.
10    read (CSTR(I1:I2-1),90010,err=803) F
```

```
90010 format (f15.0)
      return
c Errors.
801   ISTAT = 1
      return
802   ISTAT = 2
      return
803   ISTAT = 3
      return

      end

************************************************************ RCI    ***

      subroutine RCI (KPARM,CSTR,NCSTR,I,ISTAT)

c RCI - [MUELLER.FS.GEN]CSMGENLB
c Read integer I, the KPARMth element in CSTR(1:NCSTR).
c  CSTR contains one-or-more mixed-type fields
c  separated by blanks or commas to the right of an optional '='.
c  KPARMth element contains I to the right of an optional '>'.
c ISTAT= 0 for success;
c      = 1 if CSTR contains fewer than KPARM fields;
c      = 2 if KPARMth field is empty;
c      = 3 if KPARMth field is unreadable;
c      if ISTAT>0, return with I unchanged.

      character CSTR*(*)

      ISTAT = 0
c Eliminate trailing white space.
      N = NCSTR+1
1     N = N-1
      if (CSTR(N:N).eq.' '.or.CSTR(N:N).eq.',') goto 1
c Find optional '='.
      I2 = INDEX(CSTR(1:N),'=')
c Find start and end of KPARMth field.
      do 4 K=1,KPARM
         I1 = I2
2        I2 = I1
3        I1 = I1+1
         if (CSTR(I1:I1).eq.' ') goto 3
         if (CSTR(I2:I2).eq.' '.and.CSTR(I1:I1).eq.',') goto 2
         IBLANK = INDEX(CSTR(I1:N),' ')
         ICOMMA = INDEX(CSTR(I1:N),',')
         if (IBLANK.eq.0.and.ICOMMA.eq.0.and.K.lt.KPARM) goto 801
         if (IBLANK.eq.0) IBLANK = N-I1+2
         if (ICOMMA.eq.0) ICOMMA = N-I1+2
         I2 = I1+MIN(IBLANK,ICOMMA)-1
4        continue
c Check for empty field.
      if (I2.eq.I1) goto 802
c Find optional '>'.
      I1 = I1+INDEX(CSTR(I1:I2),'>')
c Get I.
10    read (CSTR(I1:I2-1),90010,err=803) I
90010 format (i12)
      return
c Errors.
801   ISTAT = 1
      return
802   ISTAT = 2
      return
803   ISTAT = 3
      return

      end

************************************************************ ROT12  ***

      subroutine ROT12 (Z1,ZN1,IAZM1,Z2,ZN2,IAZM2,NP,DT,IAZMR)

c ROT12 - [MUELLER.FS.SD]CSMSDLB
c Rotate seismograms Z1 and Z2 into azimuth IAZMR.
c Assume MOD(ABS(IAZM1-IAZM2),360) = 90.
c  Original Z1: null=ZN1,azimuth=IAZM1
c  Original Z2: null=ZN2,azimuth=IAZM2
c  Rotated Z1: null=ZN1,azimuth=IAZMR(=newIAZM1)
c  Rotated Z2: null=ZN2,azimuth=IAZMR+90(=newIAZM2)

      dimension Z1(1),Z2(1)

      PI = 4.*ATAN(1.)
      DTOR = PI/180.
      C1 = COS((IAZMR-IAZM1)*DTOR)
      C2 = COS((IAZMR-IAZM2)*DTOR)
      C3 = COS((IAZMR+90-IAZM1)*DTOR)
      C4 = COS((IAZMR+90-IAZM2)*DTOR)
      do 1 I=1,NP
         ZZ1 = Z1(I)
```

```
          ZZ2 = Z2(I)
          if (ZZ1.eq.ZN1.or.ZZ2.eq.ZN2) then
            Z1(I) = ZN1
            Z2(I) = ZN2
            goto 1
          endif
          Z1(I) = ZZ1*C1+ZZ2*C2
          Z2(I) = ZZ1*C3+ZZ2*C4
1         continue
        IAZM1 = MOD(IAZMR,360)
        IAZM2 = MOD(IAZMR+90,360)
        return

        end

************************************************************* RVFBB  ***

        subroutine RVFBB (Z1,ZNULL1,ISIZ)

c RVFBB - [MUELLER.FS.SD]CSMSDLB
c Read VFBB file into array Z1.
c See also MVFBB.FOR and VFCOMMON.FI.
c If (CRORI='R' and 0<R103<10), assume data have been through AGRAM.
c  (Header GAIN is not in dB for AGRAM data.).
c On error, return with NP<0:
c NP= -1 for read-error on integer header block;
c   = -2 for read-error on real header block;
c   = -3 for read-error on data block;
c   = -4 for data-type header info missing or inconsistent;
c   = -5 for block or sample header info missing or inconsistent;

        integer*2 IBLK
        dimension IBLK(256),RBLK(128),Z1(1)
        logical LAGRAM
        character C4*4,CRORI*1

        include '[MUELLER.FS.SD]VFCOMMON.FI'

c Read headers and get no-entry values.
        read (LUVFBB,rec=1,err=8001) IBLK
        read (LUVFBB,rec=2,err=8002) RBLK
        INE = IBLK(003)
        RNE = RBLK(002)

c Get data-type header info.
100     continue
c Header version.
        if (IBLK(005).eq.2) goto 8100
c CRORI, NSIZE, LAGRAM.
        LAGRAM = .false.
        CRORI = 'I'
        NSIZE = 256
        if (IBLK(004).eq.1.or.IBLK(004).eq.4) then
          CRORI = 'R'
          NSIZE = 128
          if (RBLK(103).ge.0..and.RBLK(103).le.10.) LAGRAM = .true.
        endif

c Get block and sample header info.
200     continue
c Number of header blocks.
        NEIHB = IBLK(001)
        if (IBLK(001).eq.INE) NEIHB = 0
        NETHB = IBLK(002)
        if (IBLK(002).eq.INE) NETHB = 0
        NERHB =  NINT(RBLK(001))
        if (RBLK(001).eq.RNE) NERHB = 0
        NHBLK = 2+NEIHB+NETHB+NERHB
c Number of data blocks.
c Index of last point in last data block.
        NDBLK = IBLK(031)
        if (NDBLK.eq.INE.or.NDBLK.le.0) goto 8200
        J1 = IBLK(032)
        if (J1.lt.0.or.J1.gt.NSIZE) J1 = NSIZE
        J2 = IBLK(256)-(NDBLK-1)*NSIZE
        if (J2.lt.0.or.J2.gt.NSIZE) J2 = NSIZE
        ILPLDB = MIN(J1,J2)
c Sampling rate.
        SAMP = RBLK(005)
        if (SAMP.eq.RNE.or.SAMP.le.0) goto 8200
c Index of the first point.
        IS1 = 1+NINT(TSKIP*SAMP)
        IMAX = (NDBLK-1)*NSIZE+ILPLDB
        IS1 = MIN(IS1,IMAX)
c Index of the last point.
        IS2 = IS1+INT(TREAD*SAMP)
        IMAXX = IS1+(ISIZ-1)*IRDEC
        IS2 = MIN(IS2,IMAX,IMAXX)
c Number of data blocks to skip.
        NDBSKP = (IS1-1)/NSIZE
```

```
c Number of data points to skip in first data block.
      NDPSKP = (IS1-1)-NDBSKP*NSIZE

c Get scaling and ground-motion header info.
c (RBBSF=1.,IAVD=0 if scaling unknown).
c (IAVD=0 if ground-motion unknown).
300   if (LAGRAM) then
        RBBSF = 1.
        IAVD = IBLK(254)
      else
        DCON = RBLK(46)
        CCON = RBLK(51)
        GDB = RBLK(52)
        if ((DCON.eq.RNE.or.DCON.lt.0.).or.
     *      (CCON.eq.RNE.or.CCON.le.0.).or.
     *      (GDB.eq.RNE.or.GDB.lt.0.)) then
          RBBSF = 1.
          IAVD = 0
        else
          GAIN = 10.**(GDB/20.)
          RBBSF = DCON*CCON*GAIN
          IAVD = IBLK(254)
        endif
      endif
      if (IAVD.lt.0.or.IAVD.gt.100) IAVD = 0

c Get other header info.
400   continue
c Epoch time of first datum (if IY>0, YDHMSE returns with E8=E1900).
      IYR4 = IBLK(010)
      if (IYR4.gt.0.and.IYR4.lt.200) IYR4 = 1900+IYR4
      IDY = IBLK(011)
      IHR = IBLK(012)
      IMN = IBLK(013)
      ISC = IBLK(014)
      IMSC = IBLK(015)
      call YDHMSE (IYR4,IDY,IHR,IMN,ISC,IMSC,E8,IY)
      if (IY.eq.0) E8 = E8+(IS1-1)/SAMP
c IVA,IHA (IVA,IHA=-1 if unknown or wierd).
      IVA = IBLK(041)
      if (IVA.ne.0.and.IVA.ne.90.and.IVA.ne.180) IVA = -1
      IHA = IBLK(042)
      if (IHA.lt.0.or.IHA.gt.359) IHA = -1
      if (IVA.eq.180) then
        RBBSF = -RBBSF
        IVA = 0
      endif
c Transducer parameters
c (ITRNS=0 if unknown; TFRQ=TDMP=0 if unknown).
      ITRNS = 0
      if (LAGRAM) then
        ITRNS = 4
      else
        write (C4,90400,iostat=IS) RBLK(39)
90400   format (a4)
        if (C4(1:3).eq.'FBA') then
          ITRNS = 1
        elseif (C4(1:3).eq.'VEL') then
          ITRNS = 2
        elseif (C4(1:3).eq.'SMA') then
          ITRNS = 4
        else
          ITRNS = IAVD !kludge
        endif
      endif
      TFRQ = RBLK(49)
      if (TFRQ.eq.RNE) TFRQ = 0.
      TDMP = RBLK(50)
      if (TDMP.eq.RNE) TDMP = 0.
c Recorder parameters
c (IRCDR=0 if unknown; AAFC=AAFR=0 if unknown).
      IRCDR = 0
      AAFC = RBLK(47)
      if (AAFC.eq.RNE) AAFC = 0.
      AAFR = RBLK(48)
      if (AAFR.eq.RNE) AAFR = 0.
      AAFR = 6.*AAFR !get dB/octave from poles

c Read data.
500   ZNULL1 = -32768/RBBSF
      IPT = 0
c Read the first data record.
510   IAV = NHBLK+NDBSKP+1
      IB1 = NDPSKP+1
      IB2 = MIN(NSIZE,IS2-(IAV-NHBLK-1)*NSIZE)
520   if (CRORI.eq.'I') then
        read (LUVFBB,rec=IAV,err=8003) IBLK
        do 521 IB=IB1,IB2,IRDEC
           IPT = IPT+1
521        Z1(IPT) = IBLK(IB)/RBBSF
      elseif (CRORI.eq.'R') then
```

```
        read (LUVFBB,rec=IAV,err=8003) RBLK
        do 522 IB=IB1,IB2,IRDEC
           IPT = IPT+1
522        Z1(IPT) = RBLK(IB)/RBBSF
      endif
c Read the rest of the data records.
530   IAV = IAV+1
      IB1 = IB-NSIZE
531   if (IB1.gt.NSIZE) then
        IAV = IAV+1
        IB1 = IB1-NSIZE
        goto 531
      endif
      IB2 = MIN(NSIZE,IS2-(IAV-NHBLK-1)*NSIZE)
      if (IB2.lt.IB1) goto 600
540   if (CRORI.eq.'I') then
        read (LUVFBB,rec=IAV,err=8003) IBLK
        do 541 IB=IB1,IB2,IRDEC
           IPT = IPT+1
541        Z1(IPT) = IBLK(IB)/RBBSF
      elseif (CRORI.eq.'R') then
        read (LUVFBB,rec=IAV,err=8003) RBLK
        do 542 IB=IB1,IB2,IRDEC
           IPT = IPT+1
542        Z1(IPT) = RBLK(IB)/RBBSF
      endif
      goto 530

600   DT = IRDEC/SAMP
      NP = IPT
      return

8001  NP = -1
      goto 8999
8002  NP = -2
      goto 8999
8003  NP = -3
      goto 8999
8100  NP = -4
      goto 8999
8200  NP = -5
      goto 8999
8999  write (6,98999) NP
98999 format (' *RVFBB ERROR* NP = ',i2)
      return

      end

************************************************************* SMTHS  ***

      subroutine SMTHS (Z,IPW2,DF,ISW,IMODE)

c SMTHS - [MUELLER.FS.SD]CSMSDLB
c Smooth amplitude spectrum Z with a triangle window.
c Assume freq=0.0 at Z(1); smooth from DF to Nyquist.
c If IMODE = 1, smooth amplitudes.
c If IMODE = 2, smooth log amplitudes.
c Smoothing triangle base has odd integer number of points (2*ISW+1);
c  the two outside points get zero weight, so ISW<=1 gives no smoothing.

      dimension Z(IPW2)

      if (ISW.le.1) return
      ISHIFT = IPW2/2
10    do 13 I=1,ISW
         FACT = 1./I
         WSUM = 0.
         ZSUM = 0.
         if (IMODE.eq.1) then
           do 11 J=1,2*I-1
              W = (I-IABS(I-J))*FACT
              WSUM = WSUM+W
11            ZSUM = ZSUM+W*Z(J+1)
           Z(I+1+ISHIFT) = ZSUM/WSUM
         elseif (IMODE.eq.2) then
           do 12 J=1,2*I-1
              W = (I-IABS(I-J))*FACT
              WSUM = WSUM+W
12            ZSUM = ZSUM+W*ALOG10(Z(J+1))
           Z(I+1+ISHIFT) = 10.**(ZSUM/WSUM)
         endif
13       continue
20    FACT = 1./ISW
      do 23 I=ISW+1,IPW2/2-(ISW+1)
         WSUM = 0.
         ZSUM = 0.
         if (IMODE.eq.1) then
           do 21 J=I-ISW+1,I+ISW-1
              W = (ISW-IABS(I-J))*FACT
              WSUM = WSUM+W
```

```
21            ZSUM = ZSUM+W*Z(J+1)
            Z(I+1+ISHIFT) = ZSUM/WSUM
          elseif (IMODE.eq.2) then
            do 22 J=I-ISW+1,I+ISW-1
               W = (ISW-IABS(I-J))*FACT
               WSUM = WSUM+W
22             ZSUM = ZSUM+W*ALOG10(Z(J+1))
            Z(I+1+ISHIFT) = 10.**(ZSUM/WSUM)
          endif
23        continue
30      do 33 I=IPW2/2-ISW,IPW2/2-1
          FACT = 1./(IPW2/2-I)
          WSUM = 0.
          ZSUM = 0.
          if (IMODE.eq.1) then
            do 31 J=IPW2/2-2*(IPW2/2-I)+1,IPW2/2-1
               W = (ISW-IABS(I-J))*FACT
               WSUM = WSUM+W
31             ZSUM = ZSUM+W*Z(J+1)
            Z(I+1+ISHIFT) = ZSUM/WSUM
          elseif (IMODE.eq.2) then
            do 32 J=IPW2/2-2*(IPW2/2-I)+1,IPW2/2-1
               W = (ISW-IABS(I-J))*FACT
               WSUM = WSUM+W
32             ZSUM = ZSUM+W*ALOG10(Z(J+1))
            Z(I+1+ISHIFT) = 10.**(ZSUM/WSUM)
          endif
33        continue
40      do 41 I=2,IPW2/2
41        Z(I) = Z(I+ISHIFT)
        return

        end

************************************************************* SPFLR  ***

        subroutine SPFLR (Z,IPW2,DF,FCTR)

c SPFLR - [MUELLER.FS.SD]CSMSDLB
c Ensure there are no zeros in amplitude spectrum Z.
c Assume freq=0.0 at Z(1); correct from DF to Nyquist.
c A typical value for FCTR might be 1.0E20.

        dimension Z(IPW2)

        INYQ = IPW2/2+1
        ZMAX = Z(2)
        do 1 I=2,INYQ
          if (Z(I).gt.ZMAX) ZMAX = Z(I)
1         continue
        FLR = ZMAX/FCTR
        do 2 I=2,INYQ
          Z(I) = AMAX1(Z(I),FLR)
2         continue
        return

        end

************************************************************* VFBBF  ***

        subroutine VFBBF (F,LVFBB)

c VFBBF - [MUELLER.FS.SD]CSMSDLB
c Test F to see if it's a VFBB filename:
c  F(01:03) = Day of year
c  F(04:05) = hour
c  F(06:07) = minute
c  F(08:08) = seconds (letter code)
c  F(09:09) = component
c  F(10:10) = '.'
c  F(11:13) = station-instrument name.

        logical LVFBB
        character F*(*)
        character*1 C1,C2,C3,C4,C5,C6,C7,C8,C9

        LVFBB = .true.
        if (LEN(F).lt.13) goto 1
        if (F(10:10).ne.'.') goto 1
        C1 = F(1:1)
        C2 = F(2:2)
        C3 = F(3:3)
        C4 = F(4:4)
        C5 = F(5:5)
        C6 = F(6:6)
        C7 = F(7:7)
        C8 = F(8:8)
        C9 = F(9:9)
        if (C1.lt.'0'.or.C1.gt.'3') goto 1
        if (C1.eq.'3') then
```

```
        if (C2.lt.'0'.or.C2.gt.'6') goto 1
        if (C2.eq.'6') then
          if (C3.lt.'0'.or.C3.gt.'6') goto 1
        else
          if (C3.lt.'0'.or.C3.gt.'9') goto 1
        endif
      else
        if (C2.lt.'0'.or.C2.gt.'9') goto 1
        if (C3.lt.'0'.or.C3.gt.'9') goto 1
      endif
      if (C4.lt.'0'.or.C4.gt.'2') goto 1
      if (C4.eq.'2') then
        if (C5.lt.'0'.or.C5.gt.'3') goto 1
      else
        if (C5.lt.'0'.or.C5.gt.'9') goto 1
      endif
      if (C6.lt.'0'.or.C6.gt.'5') goto 1
      if (C7.lt.'0'.or.C7.gt.'9') goto 1
      if ((C8.lt.'a'.or.C8.gt.'t') .and.
     *    (C8.lt.'A'.or.C8.gt.'T')) goto 1
      if (C9.lt.'1'.or.C9.gt.'9') goto 1
      return
1     LVFBB = .false.
      return

      end

************************************************************* WCC    ***

      subroutine WCC (C,IC1,IC2,CSTR,ICSTR1,ICSTR2)

c WCC - [MUELLER.FS.GEN]CSMGENLB
c Write character string C(IC1:IC2) into CSTR(ICSTR1:).

      character C*(*),CSTR*(*)

      LCSTR = LEN(CSTR)
      if (ICSTR1.gt.LCSTR) goto 801
      ICSTR2 = (ICSTR1-1)+(IC2-IC1+1)
      if (ICSTR2.gt.LCSTR) goto 802
      CSTR(ICSTR1:) = C(IC1:IC2)
      return
801   ICSTR1 = LCSTR
802   ICSTR2 = ICSTR1
      CSTR(ICSTR1:) = '?'
      return

      end

************************************************************* WCCAVD ***

      subroutine WCCAVD (IAVD,IMODE,CSTR,ICSTR1,ICSTR2)

c WCCAVD - [MUELLER.FS.GEN]CSMGENLB
c Translate IAVD into ground-motion units and write into CSTR(ICSTR1:).
c IMODE=1 for labeling seismogram; IMODE=2 for labeling spectrum.
c IAVD should correspond (more or less) to header I254 in VFBB data.
c WCCAVD recognizes:
c IAVD=-1 (spectrum only) for spectral ratio (RATIO);
c      -2 (spectrum only) for spectral amplification (AMPLIFICATION);
c       1 for acceleration (seismogram=CM/S**2,spectrum=CM/S);
c       2 for velocity (CM/S,CM);
c       3 for displacement (CM;CM-S);
c       4 for integrated displacement (CM-S,CM-S**2);
c      10 for squared acceleration (CM**2/S**4,CM**2/S**3);
c      11 for integrated squared acceleration (CM**2/S**3,CM**2/S**2);
c      20 for squared velocity (CM**2/S**2,CM**2/S);
c      21 for integrated squared velocity (CM**2/S,CM**2);
c      30 for squared displacement (CM**2,CM**2-S);
c      31 for integrated squared displacement (CM**2-S,CM**2-S**2);
c      50 for volumetric strain (MICROSTRAIN,MICROSTRAIN-S).

      logical L1,L2
      character CSTR*(*)

      if (IMODE.lt.1.or.IMODE.gt.2) goto 801
      L1 = .false.
      L2 = .false.
      if (IMODE.eq.1) L1 = .true.
      if (IMODE.eq.2) L2 = .true.
      if (IAVD.eq.-1.and.L2) then
        call WCC ('RATIO',1,5,CSTR,ICSTR1,ICSTR2)
      elseif (IAVD.eq.-2.and.L2) then
        if (L2) call WCC ('AMPLIFICATION',1,13,CSTR,ICSTR1,ICSTR2)
      elseif (IAVD.eq.1) then
        if (L1) call WCC ('CM/S**2',1,7,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('CM/S',1,4,CSTR,ICSTR1,ICSTR2)
      elseif (IAVD.eq.2) then
        if (L1) call WCC ('CM/S',1,4,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('CM',1,2,CSTR,ICSTR1,ICSTR2)
```

```
      elseif (IAVD.eq.3) then
        if (L1) call WCC ('CM',1,2,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('CM-S',1,4,CSTR,ICSTR1,ICSTR2)
      elseif (IAVD.eq.4) then
        if (L1) call WCC ('CM-S',1,4,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('CM-S**2',1,7,CSTR,ICSTR1,ICSTR2)
      elseif (IAVD.eq.10) then
        if (L1) call WCC ('CM**2/S**4',1,10,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('CM**2/S**3',1,10,CSTR,ICSTR1,ICSTR2)
      elseif (IAVD.eq.11) then
        if (L1) call WCC ('CM**2/S**3',1,10,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('CM**2/S**2',1,10,CSTR,ICSTR1,ICSTR2)
      elseif (IAVD.eq.20) then
        if (L1) call WCC ('CM**2/S**2',1,10,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('CM**2/S',1,7,CSTR,ICSTR1,ICSTR2)
      elseif (IAVD.eq.21) then
        if (L1) call WCC ('CM**2/S',1,7,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('CM**2',1,5,CSTR,ICSTR1,ICSTR2)
      elseif (IAVD.eq.30) then
        if (L1) call WCC ('CM**2',1,5,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('CM**2-S',1,7,CSTR,ICSTR1,ICSTR2)
      elseif (IAVD.eq.31) then
        if (L1) call WCC ('CM**2-S',1,7,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('CM**2-S**2',1,10,CSTR,ICSTR1,ICSTR2)
      elseif (IAVD.eq.50) then
        if (L1) call WCC ('MICROSTRAIN',1,11,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('MICROSTRAIN-S',1,13,CSTR,ICSTR1,ICSTR2)
      else
        if (L1) call WCC ('SEISMOGRAM',1,10,CSTR,ICSTR1,ICSTR2)
        if (L2) call WCC ('SPECTRUM',1,8,CSTR,ICSTR1,ICSTR2)
      endif
      return
801   call WCC ('?',1,1,CSTR,ICSTR1,ICSTR2)
      return

      end

************************************************************* WCCAZM ***

      subroutine WCCAZM (IVA,IHA,CSTR,ICSTR1,ICSTR2)

c WCCAZM - [MUELLER.FS.GEN]CSMGENLB
c Write IVA,IHA into CSTR(ICSTR1:).
c IVA,IHA should correspond (more or less) to headers I41,I42 in VFBB data.

      character CSTR*(*)

      if (IVA.eq.0) then
        call WCC ('UP',1,2,CSTR,ICSTR1,ICSTR2)
      elseif (IVA.eq.180) then
        call WCC ('DOWN',1,4,CSTR,ICSTR1,ICSTR2)
      elseif (IVA.eq.90.and.(IHA.ge.0.and.IHA.le.359)) then
        if (ICSTR1+3.gt.LEN(CSTR)) goto 801
        call WCC ('HHHH',1,4,CSTR,ICSTR1,ICSTR2)
1       write (CSTR(ICSTR1+1:),90001) IHA
90001   format (i3.3)
      else
        goto 801
      endif
      return
801   call WCC ('?',1,1,CSTR,ICSTR1,ICSTR2)
      return

      end

************************************************************* WCCE8  ***

      subroutine WCCE8 (E8,IMODE,CSTR,ICSTR1,ICSTR2)

c WCCE8 - [MUELLER.FS.GEN]CSMGENLB
c Translate epoch time E8 to DHMS form and write into CSTR(ICSTR1:).
c IMODE= 1 call EYDHMS and write doy:hr:mn:sc;
c      = 2 call EYDHMS and write doy:hr:mn:sc.msc;

      real*8 E8
      character CSTR*(*),C16*16

      if (IMODE.lt.1.or.IMODE.gt.2) goto 801
      call EYDHMS (E8,IYR4,IDY,IHR,IMN,ISC,IMSC,ISTAT)
      C16 = '000:00:00:00.000'
      write (C16(01:03),90001) IDY
      write (C16(05:06),90002) IHR
      write (C16(08:09),90002) IMN
      write (C16(11:12),90002) ISC
      write (C16(14:16),90001) IMSC
90001 format (i3.3)
90002 format (i2.2)
      if (IMODE.eq.1) call WCC (C16,1,12,CSTR,ICSTR1,ICSTR2)
      if (IMODE.eq.2) call WCC (C16,1,16,CSTR,ICSTR1,ICSTR2)
      return
```

```
801   call WCC ('?',1,1,CSTR,ICSTR1,ICSTR2)
      return

      end

************************************************************ WCCSEC ***

      subroutine WCCSEC (SEC,CSTR,ICSTR1,ICSTR2)

c WCCSEC - [MUELLER.FS.GEN]CSMGENLB
c Translate SEC to VFBB letter code and write into CSTR(ICSTR1:).

      character CSTR*(*)

      if (SEC.lt.0..or.SEC.ge.60.) goto 801
      if (SEC.lt.3.) then
        call WCC ('A',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.6.) then
        call WCC ('B',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.9.) then
        call WCC ('C',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.12.) then
        call WCC ('D',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.15.) then
        call WCC ('E',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.18.) then
        call WCC ('F',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.21.) then
        call WCC ('G',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.24.) then
        call WCC ('H',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.27.) then
        call WCC ('I',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.30.) then
        call WCC ('J',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.33.) then
        call WCC ('K',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.36.) then
        call WCC ('L',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.39.) then
        call WCC ('M',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.42.) then
        call WCC ('N',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.45.) then
        call WCC ('O',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.48.) then
        call WCC ('P',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.51.) then
        call WCC ('Q',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.54.) then
        call WCC ('R',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.57.) then
        call WCC ('S',1,1,CSTR,ICSTR1,ICSTR2)
      elseif (SEC.lt.60.) then
        call WCC ('T',1,1,CSTR,ICSTR1,ICSTR2)
      endif
      return
801   call WCC ('?',1,1,CSTR,ICSTR1,ICSTR2)
      return

      end

************************************************************ WCF    ***

      subroutine WCF (F,NSIGF,IMODE,CSTR,ICSTR1,ICSTR2)

c WCF - [MUELLER.FS.GEN]CSMGENLB
c Write real F (NSIGF figures) into CSTR(ICSTR1:).
c IMODE= 1 "f" format;
c      = 2 "e" format;
c      = 3 WCF chooses format, depending on size of F.

      character CSTR*(*),CBUF*14

      LCSTR = LEN(CSTR)
      if (ICSTR1.gt.LCSTR) goto 801
      if (NSIGF.lt.1.or.NSIGF.gt.7) goto 802
      if (IMODE.lt.1.or.IMODE.gt.3) goto 802
1     write (CBUF,90001,err=802) F
90001 format (sp,1pe<NSIGF+6>.<NSIGF-1>)
2     read (CBUF(1:NSIGF+2),90002,err=802) FF
90002 format (f)
3     read (CBUF(NSIGF+4:NSIGF+6),90003,err=802) IFF
90003 format (i)
      if (IMODE.eq.2) goto 20
      if (IMODE.eq.3.and.(IFF.lt.-3.or.IFF.gt.+3)) goto 20
c "f" format
10    I1 = 0
      if (FF.lt.0.) I1 = 1
      if (IFF.lt.0) then
```

```
        I2 = -(IFF+1)
        ICSTR2 = (ICSTR1-1)+NSIGF+1+I1+I2
        if (ICSTR2.gt.LCSTR) goto 802
11      write (CSTR(ICSTR1:),90011) F
90011   format (f<ICSTR2-ICSTR1+1>.<NSIGF+I2>)
      else
        I2 = MAX(IFF+1-NSIGF,0)
        ICSTR2 = (ICSTR1-1)+NSIGF+1+I1+I2
        if (ICSTR2.gt.LCSTR) goto 802
        I3 = MAX(NSIGF-(IFF+1),0)
12      write (CSTR(ICSTR1:),90012) F
90012   format (f<ICSTR2-ICSTR1+1>.<I3>)
      endif
      return
c "e" format
20    I1 = 0
      if (FF.lt.0.) I1 = 1
      I2 = 0
      if (IFF.lt.0) I2 = 1
      I3 = 0
      if (IFF.le.-10.or.IFF.ge.+10) I3 = 1
      ICSTR2 = (ICSTR1-1)+NSIGF+3+I1+I2+I3
      if (ICSTR2.gt.LCSTR) goto 802
      CSTR(ICSTR1:) = CBUF(2-I1:NSIGF+3+I2)//CBUF(NSIGF+6-I3:NSIGF+6)
      return
801   ICSTR1 = LCSTR
802   ICSTR2 = ICSTR1
      CSTR(ICSTR1:) = '?'
      return

      end

********************************************************** WCI    ***

      subroutine WCI (I,CSTR,ICSTR1,ICSTR2)

c WCI - [MUELLER.FS.GEN]CSMGENLB
c Write integer I into CSTR(ICSTR1:).

      character CSTR*(*)

      LCSTR = LEN(CSTR)
      if (ICSTR1.gt.LCSTR) goto 801
      I1 = 0
      if (I.lt.0) I1 = 1
      I2 = 1
      if (I.ne.0) I2 = LOG10(ABS(REAL(I)))+1
      ICSTR2 = (ICSTR1-1)+I1+I2
      if (ICSTR2.gt.LCSTR) goto 802
1     write (CSTR(ICSTR1:),90001) I
90001 format (i<I1+I2>)
      return
801   ICSTR1 = LCSTR
802   ICSTR2 = ICSTR1
      CSTR(ICSTR1:) = '?'
      return

      end

********************************************************** WCT4   ***

      subroutine WCT4 (T4,CSTR,ICSTR1,ICSTR2)

c WCT4 - [MUELLER.FS.GEN]CSMGENLB
c Write time T4 (second) to millisecond precision into CSTR(ICSTR1:).

      character CSTR*(*)

      LCSTR = LEN(CSTR)
      if (ICSTR1.gt.LCSTR) goto 801
      I1 = 0
      if (T4.lt.0.) I1 = 1
      I2 = 5
      if (T4.ne.0.) I2 = I2+MAX(0,INT(ALOG10(ABS(T4))))
      ICSTR2 = (ICSTR1-1)+I1+I2
      if (ICSTR2.gt.LCSTR) goto 802
1     write (CSTR(ICSTR1:),80001) T4
80001 format (f<ICSTR2-ICSTR1+1>.3)
      return
801   ICSTR1 = LCSTR
802   ICSTR2 = ICSTR1
      CSTR(ICSTR1:) = '?'
      return

      end

********************************************************** WCT8   ***

      subroutine WCT8 (T8,CSTR,ICSTR1,ICSTR2)
```

```
c WCT8 - [MUELLER.FS.GEN]CSMGENLB
c Write time T8 (second) to millisecond precision into CSTR(ICSTR1:).

      real*8 T8
      character CSTR*(*)

      LCSTR = LEN(CSTR)
      if (ICSTR1.gt.LCSTR) goto 801
      I1 = 0
      if (T8.lt.0.) I1 = 1
      I2 = 5
      if (T8.ne.0.) I2 = I2+MAX(0,INT(DLOG10(DABS(T8))))
      ICSTR2 = (ICSTR1-1)+I1+I2
      if (ICSTR2.gt.LCSTR) goto 802
1     write (CSTR(ICSTR1:),80001) T8
80001 format (f<ICSTR2-ICSTR1+1>.3)
      return
801   ICSTR1 = LCSTR
802   ICSTR2 = ICSTR1
      CSTR(ICSTR1:) = '?'
      return

      end

************************************************************* YDHMSE ***

      subroutine YDHMSE (IYR4,IDY,IHR,IMN,ISC,IMSC,E8,ISTAT)

c YDHMSE - [MUELLER.FS.GEN]CSMGENLB
c Compute epoch seconds (E8, relative to 00:00 1 Jan 1970)
c  from 4-digit year (IYR4), day-of-year (IDY), hour (IHR),
c  minute (IMN), second (ISC), and millisecond (IMSC).
c Valid for 1901=<IYR4=<2099 (simple leap-year test for this range).
c YDHMSE doesn't know about leap seconds.
c ISTAT= 0 for success;
c      = 1 if IYR4 is out-of-range;
c      = 2 if IDY is out-of-range;
c      = 3 if IHR is out-of-range;
c      = 4 if IMN is out-of-range;
c      = 5 if ISC is out-of-range;
c      = 6 if IMSC is out-of-range.
c      If ISTAT>0, return with E8=E1900.

      real*8 E8,E1900,E1901
      data E1900 /-2208988800.0d0/
      data E1901 /-2177452800.0d0/

      ISTAT = 0
      E8 = E1900
      if (IYR4.lt.1901.or.IYR4.gt.2099) then
        ISTAT = 1
        return
      endif
      ILEAP = 0
      if (MOD(IYR4,4).eq.0) ILEAP = 1
      if (IMSC.lt.0.or.IMSC.gt.999) ISTAT = 6
      if (ISC.lt.0.or.ISC.gt.59) ISTAT = 5
      if (IMN.lt.0.or.IMN.gt.59) ISTAT = 4
      if (IHR.lt.0.or.IHR.gt.23) ISTAT = 3
      if (IDY.lt.1.or.IDY.gt.365+ILEAP) ISTAT = 2
      if (ISTAT.ne.0) return
      NYLEAP = 0
      NNLEAP = 0
      do 1 I=1901,IYR4-1
         if (MOD(I,4).eq.0) then
           NYLEAP = NYLEAP+1
         else
           NNLEAP = NNLEAP+1
         endif
1        continue
      E8 = E1901+
     *     NYLEAP*31622400.0d0+NNLEAP*31536000.0d0+
     *     (IDY-1)*86400.0d0+
     *     IHR*3600.0d0+
     *     IMN*60.0d0+
     *     ISC*1.0d0+
     *     IMSC*1.0d-3
      return

      end
```

Appendix C

Sample VFBB data file: Synthetic data file converted from block-binary to ASCII.

(Note an important difference between real and synthetic data... For real data, R046 is an instrument-dependent digitizing constant (3276.8 count/V for GEOS data, 204.8 count/V for DR100 data, etc.); it is used with R051 and R052 to convert counts into ground motion. For synthetic data, R046 is simply a scale factor used to convert counts into synthetic ground motion.)

PUB1:[MUELLER.EQK.TEST]0010101A4.ST1

Headers
I(010) = 1988
I(011) = 1
I(012) = 1
I(013) = 1
I(014) = 1
I(015) = 0
I(016) = INE
I(031) = 4
I(032) = 33
I(041) = 0
I(042) = 0
I(254) = 2
I(256) = INE
R(005) = 0.2000000E+03
R(046) = 0.3034838E+05
R(047) = RNE
R(048) = RNE
R(049) = RNE
R(050) = RNE
R(051) = 0.1000000E+01
R(052) = 0.0000000E+00
Data block #00001
-2058 -856 -2092 -2068 2451 76 -2459 1762 -695 -2694 2049 -627 2723
-2936 863 -93 -2773 1648 -2705 -2478 -678 -2682 2676 849 2764 2345
-244 1373 1116 -860 1550 -1767 656 -2354 2634 1206 1912 2353 667
-1112 -574 -363 -2592 -1075 -738 1746 2258 2915 -2342 191 -821 1592
2272 1376 146 1581 -35 260 -2689 2458 -1918 1192 745 -876 -2319
-665 1262 -1985 -1651 -1555 -334 -2250 -761 -304 -438 -1399 1934 -2183
-1084 473 999 1193 1854 2224 1191 -61 -2067 1430 1700 -1325 149
-1660 1538 -1368 915 88 -160 266 1005 1595 -320 -999 -2008 2947
-1740 -224 1195 236 -611 1780 -894 571 -2265 286 -2114 1230 -1927
2469 -747 -1122 -2233 -2064 2992 2184 -1242 -811 2506 1482 -1483 -1319
1019 -699 -1061 -2875 1665 -2258 -2194 1316 2102 1023 -1237 -2573 -1513
-1036 1391 1352 2120 -1188 -1772 890 -2248 1414 -2030 -2459 1604 -1762
1573 2159 -577 -2504 2071 1067 -1533 1146 -1455 -1271 1056 -1320 997
-1164 -2948 -427 1410 2590 -1687 -2620 -12 -1686 -2927 -374 -1849 -77
1769 1264 1777 2200 -1138 2515 -2176 -1602 -2769 431 -576 -2111 657
-2671 -1831 -1755 1236 972 618 7511 17901 26541 31762 32735 28222 22008
16354 8792 -323 -8728 -20184 -25666 -30932 -30690 -30540 -26988 -19153 -7271 -88
662 1024 -1777 2534 -4 772 708 800 2908 -2305 2770 -2678 -1909
1922 -2410 -916 -2230 1934 732 736 2941 599 -2752 -130 -2240 -300
769 2816 -1224 -32 -2337 2314 1136 -2784 2982
Data block #00002
-2142 2214 2138 -1743 609 2413 623 -907 -366 2801 527 1395 -1759
-540 -203 2584 -2962 -2776 656 1379 -890 -2809 -1179 -430 1924 1915
2647 -2673 -204 1768 2088 2453 960 1550 22 -390 1153 -747 -1503
39 -327 -2701 888 -501 2859 -2399 -1627 -1706 852 -2952 -1331 -884
-925 -1512 1739 42 2323 2212 1634 -1474 -460 2137 1105 2491 1899
-2809 -162 654 -2826 2655 557 440 -2570 2178 2990 -2374 -2489 -1829
2613 1254 104 1175 -2461 -807 716 -2445 -746 -3029 351 110 -1047
108 2949 -550 -2535 2786 1636 2643 -1063 1734 -1135 -1764 399 -1932
766 767 1790 2922 -2607 -1288 -1212 490 -310 -1591 -1826 2664 2233
-2972 2928 215 221 1268 -2332 2347 -429 682 -43 2935 -99 -2820
-1249 1578 1536 2616 248 -299 1137 -184 -638 -317 -219 458 -3031
-2934 -2794 -2060 -507 2902 -923 207 2613 -1697 1874 -1646 1889 867
-2557 -1700 -1308 2439 1259 -1330 -1281 -2693 -2255 2741 -1180 2919 -2130
-849 -1184 -2102 2235 2281 684 -1978 1153 -996 -1579 -2261 2154 -1079
2857 903 985 1388 -1436 -1213 -1180 605 1513 704 2975 2885 2358
2633 1575 -1645 -2608 2176 -1839 -2253 -1758 2913 2601 674 851 -1834
594 3015 2030 1598 1416 -2204 -1370 2423 1102 1333 -44 642 172
1394 -2415 -2956 1080 1713 -945 -685 86 2002 -201 931 -1128 -207
2839 -2779 -991 1903 50 -1720 1563 1434 -2802 3017 2270 2122 -400
204 1504 434 1164 259 -1745 -1843 -131 2811
Data block #00003
2511 1358 482 -2160 1861 2619 302 -483 -1334 2555 644 1147 2008
-2278 387 -949 2796 38 -850 1057 1305 -348 -211 600 -2797 83
-1664 -1048 2244 -1846 2649 -1078 833 -1933 -1887 1017 597 1939 1789
1202 -1993 2448 -1056 31 -1838 1791 -2575 1069 1791 718 2917 -759
-1573 -2998 1849 766 -2851 818 793 -542 -2676 203 -266 2610 517
2406 -362 -2093 -622 2364 -2429 61 -839 1101 -3 -1162 -1411 1124
-1609 -1236 888 1473 1536 -525 2932 1456 -1145 796 -41 2325 698
-132 -688 757 2665 671 -1266 2848 -1386 230 -1462 886 1726 -825
126 -709 1038 -2114 2557 -1427 -2101 -1393 1190 2489 -1806 2052 -2910
2755 -1422 -2346 -19 -2634 -2674 -304 28 -38 -1766 -1989 -2513 2183
-1249 -109 1728 1235 548 -1506 706 3006 346 -1083 -2520 683 2228

```
  904  2570    59   151  -409  2902 -2629  -787 -2687  -568  2126 -2941  -124
-2348  1661 -2966  -559 -2450  -685  1760  -683 -2283  1731  1097   910  2501
 -821 -2759  -974  1401  -503 -1022  -198 -2402 -1269   542  1298 -2861   648
 1513  2618  2334  1593  1257  2768  2781  1263 -2953 -2459 -1538  1395  1300
 -674  2861  1360   858   118  2846 -1838 -1844  2800  2304  2888  -455  1483
 1885  -842  2646  2635  2371 -1751  2727  1145  2372 -1893   186    13  2023
-2127  1847  3001 -1468  -448  1170  -876   428  -228  -125 -1180   879 -2682
-1573 -1958   593 -1844  2717   -94  -530 -1888  1682 -2552  2005   774  2661
 1619  1101  -778 -2708  2884   479   359  2401  2142
Data block #00004
  901  -770 -2889  2371  1278 -2720   749  2654 -2461  -775   505 -1749  3027
  573   738  1063 -2570   610  3017  2247   976  1284 -2387 -2505  1876  -495
-2937  -200 -2537 -2982  1334  -603  1412     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0
```