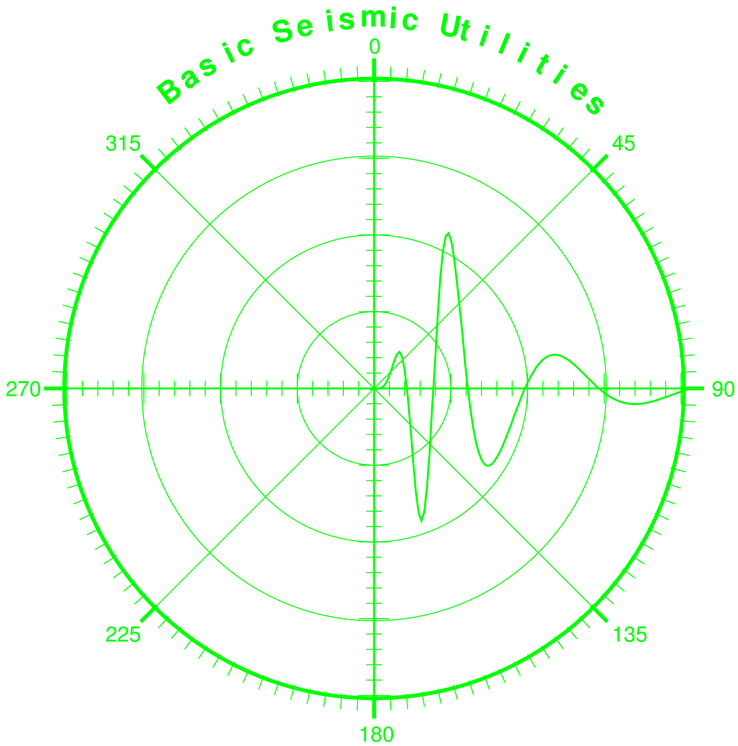
Running Basic Seismic Utilities (BSU)

Dr. P. Michaels, PE
<pmsolidx@gmail.com>
<paulmichaels@boisestate.edu>

June 17, 2024
Version 3.0.3



Michaels Engineering Geophysics
Boise, Idaho

Boise State Geosciences, Engineering Geophysics
Boise State University
Boise, Idaho 83725

Contents

List of Figures

1 Acknowledgements

This software is an updated version of the revised release that was done in April 2018. The user guide for version 3.0.1 is still largely valid, and this document fills the need to guide a user to which utilities might be of use to them. Building on earlier versions, the goal has been portability. That is, while some new programs have been added, much of the software has been carried forward, and thus is now available for use on a variety of operating systems. Thus, I remain indebted to the work of many others in the development of this package. I would like to thank Enders A. Robinson and the Holden-Day Inc., Liquidation Trust (1259 S.W. 14th Street, Boca Raton, FL 33486, Phone: 561.750-9229 Fax: 561.394.6809) for license to include and distribute under the GNU license subroutines found in Dr. Robinson's 1967 book Robinson (1967) , Multichannel time series analysis with digital computer programs. This book is currently out of print, but contains a wealth of algorithms, several of which I have found useful and included in the BSU Fortran77/gfortran subroutine library (sublib4.a). This has saved me considerable time.

In other cases, subroutines taken from the book Numerical Recipes Press *et al.* (1989) had to be replaced (the publisher did not give permission to distribute). While this is an excellent book, and very instructional for those interested in the theory of the algorithms, future authors of software should know that the algorithms given in that book are NOT GNU. Replacement software was found in the *GNU Scientific Library (GSL)*, and in the *CMLIB*.

For plotting, I remain indebted to the developers of *PLPLOT*. *PLPLOT* credits have grown to be too many to list. However, there are a number of instances where I ran into dependency problems with some operating systems, particularly the Microsoft family. So I have added GNUPLOT alternatives.

Where there was a need to solve for eigenvalues, or invert a matrix, I have relied on *LAPACK*. This excellent package is well worth installing, and I acknowledge the contributions of the many authors of *LAPACK* and *BLAS*.

Lastly, the author acknowledges financial support over the years from various clients. Financial support included that provided by grant DAAH04-96-1-0318 from the U.S. Army Research Office. Views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Army Research Office or the U.S. Government. This material is also based upon work supported by the National Science Foundation under Grant No. 0321233. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation. Other support has been provided by the Idaho State Board of Education, Boise State University Sabbatical Committee, Idaho Transportation Department, and Idaho Power. Again, views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the views of those who have provided the author support.

2 Conventions

- **Capitalization** In the text of this document, program names are often typed in upper case letters. This is to make the name stand out. However, when actually executing a program, use lower case letters for command line execution in a terminal. *Linux is case sensitive*.

- **File Naming** BSU programs expect file names with no more than 12 characters. If a file xxxx.seg is input to a program byyy, then the output data file will be byyyxxxx.seg. Naming scrolls right. So if data file byyyxxxx.seg is input to program bzzz, the output will be named bzzzbyyy.seg. This is predictable and a bit of a pain at times. It goes back to being able to write bash scripts and be able to predict the names of output for later stages in a sequence of processes. The only exception is program **mseed2seg**. If you input a file with a long name, then you will get this message:
  [ABORT] Input length violation [13]

- **File Suffixes** Seismic data in BSEGY format are *.seg files. Seismic data in SEGY format are *.sgy files. Files in SEG-2 format are *.DAT files. Bison files don't have a suffix (unless compressed as *.gz) and are 8 characters long, typically upper case alpha numeric. Some BSU programs will output GNUPLOT *.gp files.

3 Converting Between Data Formats

There are two types of seismic data formats.

- Data Exchange Formats
- Data Processing Formats

3.0.0.1 Exchange Formats These are formats meant to publicly defined so that users may exchange data regardless of what processing software they may ordinarily use. Examples would include SEGY, SEG-2, Bison, and SEG-D. The professional society, Seismic Exploration Geophysicists (SEG) has created all of these except the Bison format which was developed by makers of an engineering seismograph. For example, SEG-2 was developed for personal computers, Pullan (1990). There are other types of data exchange formats. For example, for Computer Aided Drafting, there is DXF (data exchange format). BSU software also can convert header information into DXF format which can then be used to make base maps.

Bison format is really just the format output by Bison seismographs. It can be integer or floating point. However, the floating point is extremely unusual, and I know of no other software than BSU which can process the Bison floats correctly. In particular,

```
Float 16 bit 2's complement mantissa, 4 bit exponent normalized float
generated from 32 bit internal fixed format standard instrument storage
format. Transmitted as follows: in groups of 5 words

LSbyte of first sample mantissa
MSbyte of first sample mantissa
LSbyte of second sample mantissa
MSbyte of second sample mantissa
LSbyte of third sample mantissa
MSbyte of third sample mantissa
LSbyte of fourth sample mantissa
MSbyte of fourth sample mantissa

Exponent word:
  Least significant nibble of first byte for fourth word.
  Most significant nibble of first byte for third word.
  Least significant nibble of second byte for second word.
  Most significant nibble of first byte for first word.
```

```
One must Check to see if 4 bit exponent is greater than 7
(implies negative exponent in that case).  Decode to negative
4 bit number using assumption of 2's complement
```

NOTE: SEGY uses a 16 bit integer header value for the number of samples. Fortran does not support unsigned integers, so it is likely that using an unsigned integer may present problems with some other party Fortran software. Beginning with BSU-3.0.3 codes, I use 16 bit unsigned integers for the number of samples in a trace. I mix C and Fortran to permit the Fortran codes to work this way. The result is that the maximum number of samples per trace in BSU codes is now 65536 samples.

3.0.0.2 Processing Formats There are a number of these. Seismic Unix (SU) has their format which is derived from SEGY. It is good for reflection seismic processing and is focused on vertical component geophones when it comes to headers. BSU uses what it calls BSEGY. It is very similar to SU, but headers include polarizations in 3-component directions for both sources and geophones. This is because the problems addressed by BSU include 3-components for both the source and geophone.

Both SU and BSEGY are similar to the exchange format, SEGY in that they use a 240 byte trace header followed by a data block. This way, the headers are locked to the data. Neither SU or BSEGY uses a reel header, since that goes back to the days of 9 inch tape.

BSU also supports dumping data to text and comma separated variable formats for those who just want to get the binary format out into something they can use in software like Matlab or Spread Sheets.

3.1 Conversion Utilities

The following programs do data format conversions:

- **BA2S** 3.1.1 Converts an ASCII text file to BSEGY format.
- **BSWP** 3.1.2 Byte swap. BSEGY <–> SUXDR
- **BCNV** 3.1.3 Converts between SEGY <–> BSEGY
- **BIS2SEG** 3.1.4 Converts from Bison to BSEGY
- **SEG2DUMP** 3.1.5 Raw text dump of SEG-2 file
- **EGG2SEG** 3.1.6 Converts between SEG-2 <–> BSEGY
- **GENB2S** 3.1.7 Generates bash script, Bison to BSEGY
- **SEG2TXT** 3.1.8 Converts BSEGY to ASCII text
- **SEG2CSV** 3.1.9 Converts BSEGY to Comma Separated Variable
- **BSG2** 3.1.10 Converts BSEGY to SEG-2 format.
- **SEGD2SEG** 3.1.11 Converts SEG-D to BSEGY format.
- **WAV2TXT** 3.1.12 Converts an audio WAV file to ASCII text.
- **MSEED2SEG** 3.1.13 Extracts a seismic trace from a Miniseed archive. Requires *libmseed* 3.1.13.1.
- **SAC2SEG** 3.1.14 Converts a SAC format file to BSEGY format. Also produces an ASCII and GNUPLOT file.
- **SEG2SU** 3.1.15 Bash script to convert from BSEGY format to Seismic Unix SU format (**WITHOUT XDR**). If you have compiled Seismic Unix for XDR, then you may use BSWP 3.1.2 to byte swap and convert SU files between XDR or NO XDR.
- **SU2SEG** 3.1.16 Bash script to convert from Seismic Unix SU format (**WITHOUT XDR**) to BSEGY format. If you have compiled Seismic Unix for XDR, then you may use BSWP 3.1.2 to byte swap and convert SU files between XDR or NO XDR.

3.1.1 BA2S

Converts an ASCII text file to BSEGY format.

```
ba2s  infile iorder ncol nrow dt

    infile   =  input file name
    iorder   =  1 each row is a trace, columns are time axis (int)
             =  0 each col is a trace, rows are time axis (int)
    ncol     =  number of columns in matrix (int)
    nrow     =  number of rows in matrix (int)
    dt       =  sample interval in micro seconds (int)
```

3.1.2 BSWP

BSEGY and SU (before XDR) share many headers. However, if SU is compiled with XDR, the data and headers become byte swapped. SU with XDR is referred as SUXDR.

```
bswp  infile   idfc  npts

     infile  = input file name (4char minimum)

     Data format
     idfc   1=4 byte floating point
            2=4 byte integer
            3=2 byte integer
     npts    =number of samples per trace
```

3.1.3 BCNV

Converts from SEG-Y to BSEGY, or from BSEGY to SEG-Y.

```
 bcnv  infile endian compliance idirec idfc iunits hedfil

     infile   =  input file name
     endian  0= Little Endian host (Linux PC)
             1= Big    Endian host (IBM Mainframe)
  compliance 1= SEGY Compliant (EBCDIC, IBM Float, BigEndian)
             0= (ASCII Reel Header, Float and endian of host)
     idirec  0= BSEGY ==>SEG-Y (new *.sgy)
             1= SEG-Y ==>BSEGY (new *.seg)
     idfc    1= floating point 4 byte output
             2= long integer   4 byte output
             3= short integer  2 byte output
             5= IEEE Floating Point 4 byte output
             (uses reel header if SEG-Y input data)
     iunits  1= meters
             2= feet
             (uses reel header if SEG-Y input data)
NOTE:
 hedfil only input if idirec=0  BSEGY --> SEG-Y
 (hedfil contains 3200 bytes, 40 records, 80char each
 If hedfil='none', then blank lines after C used
```

3.1.4 BIS2SEG

No geometry setting of headers, just converts Bison to BSEGY. See GEN2BS 3.1.7 for building a bash script to convert a large number of files. See TOPCON 10.1.3 for setting headers with survey *.nez file.

```
 bis2seg  infile

     infile   =  input file name
```

3.1.5 SEG2DUMP

Raw dump to a text file of SEG-2 data formatted data. Engineering seismographs like those available from Geometrics or that designed by the author SeisRecorder.

The text file contains the data samples without applying any scaling due to fold or stack or instrument scaling factors. The input file *.DAT is output as a *.lst file name. File descriptor block and trace descriptor blocks for each trace are listed before the samples.

For example, if the data were integer recorded, the raw values as integers are displayed followed by the sample times.

This program is handy for debugging and evaluating recording parameters. If one wants a text file with all scale factors applied, consider converting the data to BSEGY format (program egg2seg) and apply either SEG2TXT or SEG2CSV programs which can produce text or comma separated value files.

For an input file 0000.DAT, output file 0000.lst sample follows:

```
 ALL DATA VALUES ARE RAW SEG-2, No Scale Factors

 Confirmed SEG-2 Data: 0X3A55

-------------------------------------------------------------------------

FILE DESCRIPTOR BLOCK
-------------------------------------------------------------------------
 Size of trace pointer block = 32
 Number of traces = 1
 first string terminator character = 0
 second string terminator character = 0
 first line terminator character = 0XA
 second line terminator character = 0
 trace pointer (0) = 0XB8
 number of characters=33
 ACQUISITION_DATE 31/May/2019
 Date (dayofyear.year: 151.2019
 number of characters=30
 ACQUISITION_TIME 13:45:52
 TIME: 13 45 52
-------------------------------------------------------------------------

TRACE DESCRIPTOR BLOCKS
-------------------------------------------------------------------------
  idfc = 2
  npts = 500
 CHANNEL_NUMBER 0
 ALIAS_FILTER 100 6
 HIGH_CUT_FILTER 100 6
 FIXED_GAIN 19
 DELAY 0.00
 LINE_ID 0001
 RAW_RECORD 0000.DAT
 RECEIVER VERTICAL_GEOPHONE
 SOURCE HAMMER
 SOURCE_STATION_NUMBER 0000
 RECEIVER_LOCATION 1.00 0.00 0.00
 SOURCE_LOCATION 0.00 0.00 0.00
 SAMPLE_INTERVAL 0.001000
 DESCALING_FACTOR  1.1220185E-04
 STACK 1
-------------------------------------------------------------------------
NOTE: All Values are Raw, No Scale Factors
[Indx]  Raw Value  (time)
```

```
[0]     -10521  (0.00000)
[1]     -10216  (0.00100)
[2]      -8080  (0.00200)
[3]       2373  (0.00300)
[4]      18318  (0.00400)
[5]      37544  (0.00500)
[6]      56084  (0.00600)
[7]      66536  (0.00700)
[8]      60356  (0.00800)
[9]      32890  (0.00900)
[10]     -11589  (0.01000)
[11]     -58586  (0.01100)
 .          .        .
 .          .        .
 .          .        .
```

3.1.6 EGG2SEG

This is sample conversion of SEG-2 data to BSEGY It is just a simple conversion without any attempt to correct headers. To correct headers, see GENWAW 10.1.1. Or if survey data (*.nez) are available, consider TOPCON2 10.1.5.

```
  egg2seg  infile

     infile   =  input file name
```

TOPCON2 Alternative This alternative is to use a survey *.nez file and topcon2
EXAMPLE:

```
topcon2 a10001.nez 1061.dat 00A1 0.0 1 6 0186 0181 1061 0.  270 135 0 270
```

BHED Fix Headers This is an alternative to create a partial header file that can be edited
EXAMPLE:

```
   bhed 1061.seg 1061.hed 1

Edit 1061.hed file for correct elevations, x, y, and z.  Also, any ^@
characters like in PHONE= here need to be replaced with ascii
characters.

Then, run

  bhed 1061.seg 1061.hed 0

to upload the new headers producing a file bhed1061.seg
```

3.1.7 GENB2S

Generates bash script to convert BISON to BSEGY This is used typically to QC data, and minimal headers are created (geometry is zeroed out). It is an interactive program. **In a terminal**, run the command, `genb2s`

```
 enter 5_LETTER ALPHA PREFIX
LOGNO
 enter 3digit FIRST FILE number
001
 enter 3digit LAST FILE number
007
  OUTPUT =====> gob2s

Make gob2s executable
     chmod +x gob2s
Then run gob2s

Then run BMRG to perhaps look at the first trace
from each file (vertical component down hole here)
     bmrg r00 001 007 1 1 1
```

3.1.8 SEG2TXT

This program converts a BSEGY seismic file to an ASCII text file. Columns are traces, rows are sample time.

```
seg2txt infil tmin tmax fstrc lstrc timelist

 infil =input file name
 tmin  =minimum time
 tmax  =maximum time
 fstrc =first trace
 lstrc =last trace
 timelist = 0 do NOT add column of sample times
          = 1 Do add first column of sample times
```

One can convert only a portion of the BSEGY file, selecting a time and spatial window. If desired, an extra column of sample time values can be output as the first column.

3.1.9 SEG2CSV

Program SEG2CSV converts an entire BSEGY seismic file to a comma separated variable (csv) file with an additional column of sample times. Each column is a seismic trace.

```
  seg2csv  infile

     infile  = input file name (4char minimum)
```

3.1.10 BSG2

Program BSG2 converts an entire BSEGY seismic file to a SEG-2 format file (dat). SEG-2 is commonly used as a format output from seismic field instruments. One should always check the SEG-2 headers (3.1.5) to make sure that they have been transfered correctly. Not all header content maps 1 to 1 in different formats. Possible applications might include converting a BISON file to SEG-2 format when other software packages do not support BISON format. To do this, one would run a flow: BIS2SEG(3.1.4) → BSU BSEGY FORMAT → BSG2. (3.1.10).

```
 bsg2  infile number

     infile  = input file name (4char minimum)
     number  = output file number, zb. 0001
```

EXAMPLE:

```
bsg2 c008.seg 0008
```

would produce an output file `0008.DAT`.

3.1.11 SEGD2SEG

Program SEGD2SEG reads a SEG-D data file and outputs in BSU's BSEGY format. SEG-D data files are generated by seismic instruments commonly used in the energy industry. WARNING: THIS CODE IS A WORK IN PROGRESS. SEG-D standard is very variable in terms of options. Designed originally for tape with multiple scan channel sets, with headers in a mix of packed BCD and binary integers, it is a swamp of possibilities. This code has only been tested on a single set of data that are in 8058 format (32 bit IEEE demultiplexed floats using IEEE 754-1985 big endian byte order), (Zhang *et al.*, 2020). This code assumes it is run on a little endian (ie. PC computer), which is the reason for the byte swapping. Further, the code assumes no more than 2 channel sets, and it assumes that if there are two, the first channel set is a few aux traces (like a vibrator sweep and an autocorrelation for example). It labels the output file with a "aux" prefix to distinguish it from the shot gather of seismic traces. This code can be adapted for other data formats (see makefloat() function). The code does capture the source and geophone coordinates, filter settings, etc. if provided in the headers. From BSEGY format, one can then use program BCNV 3.1.3 to convert to SEG-Y format.

3.1.11.1 Updated for Geode Instruments The current version of the code has been upgraded to add another instrument capability. Both Sercel and now Geode instruments have support. The SEGD standard revision 1 does not have a byte code for the Geometrics Geode instruments. The Geode appears to set this code to zero. Program segd2seg run on Geode recorded data will assume that the geometry has not been included in the headers, and default dummy header geometry is the result when run on Geode recordings.

3.1.11.2 Adding Geometry to Headers If the acquisition system is a land streamer, the code SETSTREAM 10.1.18 can be used. In addition, program BHED 10.1.4 is another route to editing header information. Programs TOPCON 10.1.3 or TOPCON2 10.1.5 may also be considered.

```
segd2seg  infile

    infile  = input file name (4char minimum,
     13 character maximum )
    Example:     xxxx.segd
    (use symbolic link with shorter name if a problem)
```

3.1.12 WAV2TXT

Program WAV2TXT converts an audio WAV file to an ASCII text file. An application might be to use a sound card on a computer to record seismic data. To record on the sub audio range, one would build an AM or FM modulator circuit to modulate a audio carrier signal (say at 1 KHz for example) with the seismic signal (perhaps 0 to 100 Hz).

```
 wav2seg  infile n1  nlast

    infile  = input file name (4char minimum)
    n1   = first sample to output
    nlast    = last  sample to output
```

If the WAV file is mono, the output file will have 3 columns, (sample number, sample time, sample value). If it is a 2 channel, stereo file, then the output file will have 4 columns (sample number, sample time, ch1_value, ch2_value). Note that the output file may be quite large, and have more than the 65536 samples allowed in BSU's BSEGY format. However, one can convert the ASCII text file to BSEGY format using the program BA2S (3.1.1). If there are more than 65536 samples, one would do it in chunks. Another route would be to write some code to demodulate the carrier and resample the seismic signal at a larger sample interval.

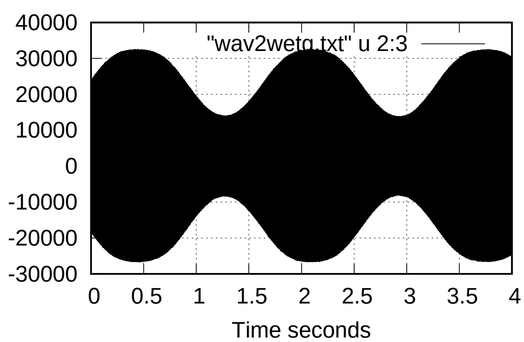


Figure 1: Example of an AM modulated carrier sampled at 44100 samples/second.

3.1.13 MSEED2SEG

Miniseed is a data format used primarily in passive seismic data archiving (see https://www.iris.edu). A Miniseed file is an archive with one or more recordings of data. Headers are very limited. One can select a specific signal by specifying the Station, Channel, Network, Location and Sample Rate. The command line arguments are:

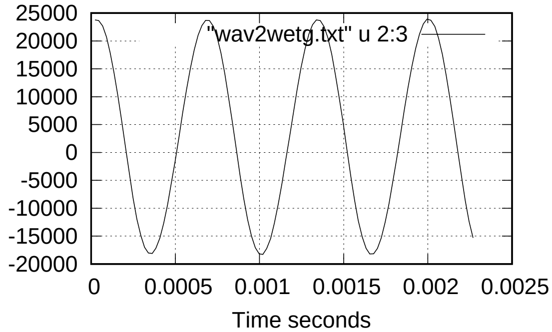


Figure 2: Blowup of the carrier frequency (about 1500 Hz), first 100 samples of Figure 1.

```
mseed2seg  infile scansw station channel network location samprate

     infile   = input file name (4char minimum, 131 char max.)
     scansw =0 apply filters, extract waveform data (int)
            =1 scan headers only, 1 line per record (int)
            =2 scan headers only,  more detail (int)
            =3+ scan headers only, with increasing detail (int)
     station  = desired station name (char) ex. COR
     channel  = desired channel (char) ex. BHZ
     network  = desired network (char) ex. IU             | OR x matches all
     location = desired location at station (char) ex. 10 | OR x matches all
     samprate = desired sample rate (double) ex. 40.0
```

Note that the ***infile*** *argument is unusual and may be very large, up to 131 characters. The earthquake and passive recording community will typically use long file names, so we deviate here.*

In a typical use, it is wise to first scan the file to determine what data are included. This is done by specifying a **scansw** value > 0. The larger the value, the greater the detail in the listing file. Usually a value of 1 is enough. Once one has found the spelling of the parameters required, a second run can filter the file for just that signal with a second run. Below is a sample from a listing file named **CORBHZ.lst**.

```
     FILES:
     Input   File= 2020-03-31-ml68-western-idaho.miniseed
     Output  File= CORBHZ.seg
     Data File  File= CORBHZ.dat
     Listing File= CORBHZ.lst

     INPUT PARAMETERS:
     Select Station = COR
     Select Channel = BHZ
     Select Network = IU
     Select Location= 10
     Select Sample Rate = 40.000000
---------------------------------------------------------------------


     SELECTED RECORDS:
IU_COR_10_BHZ, 606650, M, 512, 317 samples, 40 Hz, 2020,091,23:52:56.019538
IU_COR_10_BHZ, 606651, M, 512, 340 samples, 40 Hz, 2020,091,23:53:03.944538
IU_COR_10_BHZ, 606652, M, 512, 315 samples, 40 Hz, 2020,091,23:53:12.444538
.
.
.
```

Note that the output file name is built from the [StationChannel], in this case **CORBHZ.lst**. COR is the station name (Corvalis) and BHZ is the vertical component. When run to extract data, the **scansw** option is set to zero. This requires the additional command line arguments to specify what is desired. Example:

```
mseed2seg 2020-03-31-ml68-western-idaho.miniseed 0 COR BHZ IU 10 40.
```

To illustrate the naming convention, output files for this example are:

- **CORBHZ.seg** BSEGY formatted data file. If the number of samples exceeds 65535, then the trace is broken up into sub traces of equal length. Some data at the end may be dropped.
- **CORBHZ.dat** An ASCII file with two columns (time,data). Comments are included with each segment and start with a hash "#". This file includes the entire trace data and is avaliable for the GNUPLOT script that is created. To remove the # comment lines for import into a program like MATLAB or OCTAVE, a bash command can be used. For example,

  ```
  cat CORBHZ.dat | gawk ' !/#/ {print $0 }' >datafile
  ```

  In this case, the file, **datafile**, will only contain the two columns, (time, data value).
- **CORBYHZ.gp** A GNUPLOT script which is generated and reads the *.gp file to plot the entire data trace. Assuming GNUPLOT is installed, the plot is created with a command like this:

  ```
  gnuplot -p CORBHZ.gp
  ```

  The -p option is for persistence so it can remain on the screen. Also output is a Postscript file, **CORBHZ.ps** in this instance.

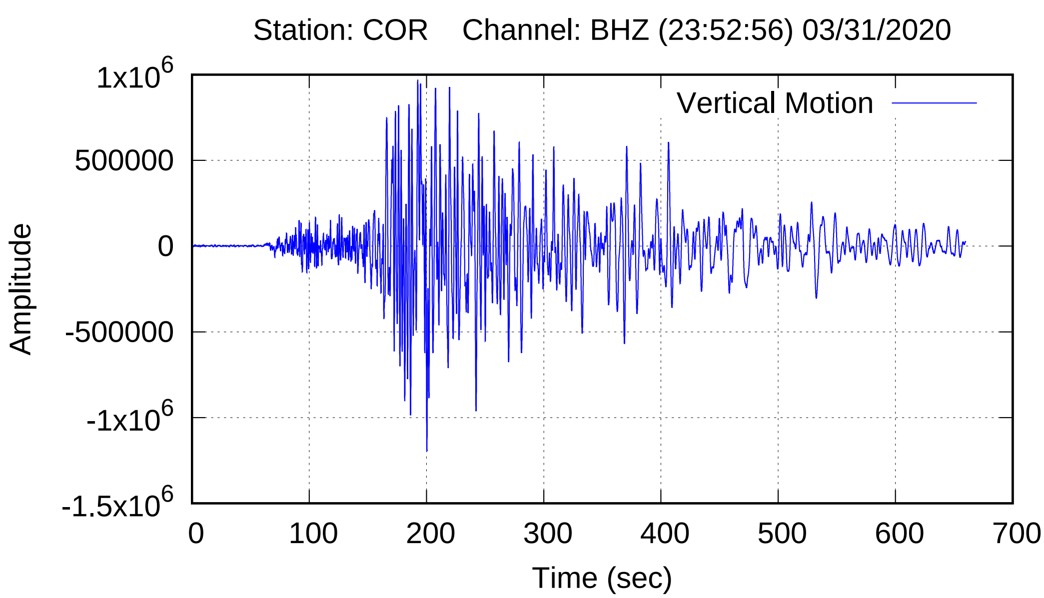


Figure 3: Image produced from Gnuplot and the output *.gp file from mseed2seg. The title gives the Station, Channel, initial time of day for zero time on the plot, and the date of the earthquake.

3.1.13.1 Required Library Program **MSEED2SEG** requires that the **libmseed** be installed. This library is available with the Debian package manager, APT, and may be available with package managers on other operating systems. If not on your OS, go to https://github.com/iris-edu/dataselect/releases. BSU-3.0.3 has been tested with **libmseed-2.19.4.tar.gz**.

3.1.14 SAC2SEG

SAC is a data format used primarily in passive seismic data archiving (see https://www.iris.edu). It differs from MiniSeed in that the headers contain more information. Further, one expects that a SAC file will only have a single recording. In a typical download from IRIS, one expects that the file name will also indicate the Station, Channel, Network, Sample rate, etc. The command line arguments are:

```
sac2seg  infile

     infile   = input file name (4char minimum, 131 char max.)
```

For example:

```
sac2seg IU.COR.10.BHZ.M.2020.091.235256.SAC
```

The above command will output files CORBHZ.lst, CORBHZ.seg, CORBHZ.dat, and CORBHZ.gp. These are the listing, BSEGY, ASCII, and Gnuplot files respectively. The file CORBHZ.lst is:

```
 Program sac2seg
 Infile: IU.COR.10.BHZ.M.2020.091.235256.SAC

 Station Name: COR
 Station Latitude = 44.585499 degrees (+North)
 Station Longitude= -123.304604 degrees (+East)
 Station Elevation= 110.000000 meters
 Station Below Surface Depth= 14.530000 meters
 Station to Event Azimuth = 88.368439
 Station to Event Distance = 649.61 km
 Station to Event Great Circle Arc 5.84 deg
 Channel: BHZ
------------------------------------------------

 ALL TIMES are GMT
 Year=2020  Julian Day=91
 DATE: 03/31/2020
 Hour=23  Minute=52  Second=56  mSecond=19
 tzero=7948376.019000 seconds
------------------------------------------------

 Event latitude  (degres, + North) = 44.460300
 Event longitude (degres, + East ) = -115.135597
 Event Depth                       = 14.530000
 Event to Sation  Azimuth          = 274.081879
------------------------------------------------

 npts=26400
 delt=0.025000
 Begin Time: 0.000538 seconds
 End   Time: 659.975525 seconds
------------------------------------------------
 Type: time series data
```

The plot will be identical to that shown in Figure 3 since they are the same recordings, one downloaded in MiniSeed and the other in SAC format. Note that there is more information in a SAC header.

3.1.15 SEG2SU

This is a bash script for converting a file from the internal processing format, BSEGY, to Seismic Unix (***WITHOUT XDR***) SU format. If SU is compiled and installed under `/usr/local` directory tree, then look for installed script in the `/usr/local/share/scripts` directory. Or you can copy it from this listing. See SU2SEG 3.1.16 for a script that goes the opposite way.

```
#!/bin/bash
# script converts BSEGY *.seg file to Seismic Unix *.su file
#Author: P Michaels  <https:pmhub.org>
#  $Id: seg2su,v 1.2 2024/01/27 23:06:16 pm Exp $
# REQUIRES Both bsu-3.0.3 and SU installed, no XDR

if test "$1" = "-h"
then
echo "USAGE: seg2su infile.seg outfile_prefix"
else
        if test "$1" = ''
        then
        echo 'Enter infile.seg name'
        read FILEN
        else
        FILEN=$1
        fi

        if test "$2" = ''
        then
        echo 'outfile_prefix'
        read OFILE
        else
        OFILE=$2
        fi

NAME=`basename $FILEN .seg `
bcnv $FILEN 0 1  0  1  1 none
segyread  tape=bcnv${FILEN:0:4}.sgy >$OFILE.su
rm -f bcnv$NAME.sgy
rm -f binary
rm -f header
rm -f bcnv$NAME.lst

fi
```

3.1.16 SU2SEG

This script will convert a Seismic Unix (SU ***WITHOUT XDR***) file to the internal processing format of BSU-3.0.3 (BSEGY). Put this or any other scripts in your executable path for ease of use. See SEG2SU 3.1.15 for a script that goes the opposite way.

```
 #!/bin/bash
# script converts Seismic Unix *.su file to BSEGY *.seg file
#Author: P Michaels  <https:pmhub.org>
#  $Id: su2seg,v 1.1 2024/01/23 01:04:36 pm Exp $
# REQUIRES Both bsu-3.0.3 and SU installed, no XDR

if test "$1" = "-h"
then
echo "USAGE: su2seg infile.su  outfile_prefix"
else
        if test "$1" = ''
        then
        echo 'Enter infile.su name'
        read FILEN
        else
        FILEN=$1
        fi

        if test "$2" = ''
        then
        echo 'outfile_prefix'
        read OFILE
        else
        OFILE=$2
        fi

        L=${#OFILE}
        if [ $L  -gt  4 ]
        then
        MSG="NOTE: truncating output prefix to 4 characters"
        TMP=${OFILE:0:4}
        OFILE=$TMP
        fi

segyhdrs <$FILEN  bfile=binary  hfile=scratch
segywrite <$FILEN hfile=scratch bfile=binary ebcdic=1 tape=$OFILE.sgy
bcnv $OFILE.sgy 0 1 1
mv bcnv$OFILE.seg $OFILE.seg
rm $OFILE.sgy
rm scratch
rm bcnv*.lst
rm binary
bdump $OFILE.seg 0
echo $MSG
fi
```

4 Header Information

There are several codes that dump information about file contents. These are:

- **BDUMP** 4.0.1 Shows partial headers of a BSEGY file
- **SEG2DUMP** 3.1.5 Raw dump of SEG-2 acquisition files
- **BHELP** 5.0.1 Lists all the programs in Basic Seismic Utilities
- **man pages** 5.0.2 An information system in Linux or Unix

4.0.1 BDUMP

Most user interest is in a single shot gather, so the program focuses largely on receiver headers plus a single header for the shot. In that case, one may supress shot header display after the first. Sometimes, as in reciprocal refraction shooting, the interest is in a geophone gather. In that case, one should display all shot headers. The command is issued from a terminal:

```
bdump infile iskip

infile:    =name of input file
iskip:     =0 supress shot header display after 1st
           =1 display all shot headers
```

The output file is bdump.lst which may be viewed in any editor or cat to the screen. A sample of the output:

```
                     |-------------------------------|
                     |   PARTIAL SEGY HEADER DUMP    |
                     |                               |
                     |          k007.seg             |
                     |                               |
                     |-------------------------------|
-----------------------------------------------------------------------------------
   Length =  2000 samples              |  Shot Elevation =      998.4
   Sample Interval =    0.00025 sec.   |  Shot Depth =        0.0
   Delay Time =    0 msec.             |  Up Hole Time =     0 msec
   Low Cut Filter  =    8 Hz.          |  Shot X-COORD =    9927.00
   High Cut Filter =  500 Hz.          |  Shot Y-COORD =    9773.13
   Line ID: BNK2                       |  Shot Date (year.moday) = 1995.0628
   Shot Orientation:                   |  Shot Time (hr:min)    = 12:29
   Azimuth=  0 Deg.  Vertical=180 Deg.|  Charge Size (grams)=     0
 -----------------------------------------------------------------------------------
 TRACE|SHOT| STATION | OFFSET|         RECEIVER          |VERT|1STBRK|K-GAIN|AZI|VER|
   #  |REC.|SHOT  REC|       |  ELEV.  X-COORD   Y-COORD|FOLD|(SEC.)| (dB) |   |   |
 -----|----|---------|-------|----------------------------|--|------|------|---|---|
    1 |   7| 024  001|  74.01| 1024.97   9981.25   9815.88|20|0.0580|   40 |  0|  0|
    2 |   7| 024  002|  73.46| 1024.38   9980.50   9816.25|20|0.0552|   40 |  0|  0|
    3 |   7| 024  003|  72.95| 1023.97   9979.76   9816.54|20|0.0540|   40 |  0|  0|
    4 |   7| 024  004|  72.31| 1023.50   9978.88   9816.79|20|0.0531|   40 |  0|  0|
    5 |   7| 024  005|  71.75| 1022.84   9978.09   9817.17|20|0.0557|   40 |  0|  0|
```

Each trace row is a geophone associated with the shot.

4.0.2 SEG2DUMP

This is described above in section 3.1.5, since it is a complete text conversion of a SEG-2 exchange format program.

5 Software Documentation

5.0.1 BHELP

This code can be used to see a brief description of all the BSU codes streamed to a terminal screen. You may wish to pipe it through less or more programs. For example:

```
bhelp  | more
  or
bhelp  | less
```

Here is a partial output of using “bhelp | less”.

```
babs.c    rectify seismic traces
bagc.c    automatic gain control of traces (scale in time and space)
ba2s.c    FORMAT CONVERSION: ASCII TEXT ---> BSEGY (no geometry setting)
bamp.F90  amplitude analysis by frequency (K-V Solid)Downhole Sph. Div.
bamx.F90  amplitude analysis by frequency (K-V Solid)SurfaceWaves Cyl.Div
bbal.c    balances two data sets to have same MAV (mean absolute value)
bcad.f    plot seismic traces as CAD (*.dxf; digital exchange file)
bcar.c    apply moving average (box car) filter as function of time
bcnv.c    FORMAT CONVERSION: BSEGY <---> SEG-Y  (BSU=*.seg, SEG-Y=*.sgy)
bcrd.f    coordinate rotation and translation, BSEGY geometry headers
bdat.c    datuming program for refraction data (easier for picking)
bdcn.f    deconvolution (profile or trace mode), prediction or error out
bdif.f    differentiates w.r.t. time using Bilinear Transform method
bdum.f    generate dummy data set with user defined impulse position
bdump.f   generate a dump of selected BSEGY header values
bedt.f    edit BSEGY seismic file (traces, time, sample interval, etc.)
bequ.c    trace equalize data by L2 norm or Maximum Absolute Value
bext.c    extract traces from a merged data set based on header values
bfil.f    ARMA FILTER of seismic traces (low-, band-, or high-pass)
bfit.f    Solves for interval velocity from times in headers (VSP)
bftr.f    FILTER traces with other *.seg traces, or namelist from bdump
bfxt.f    F-X Transform of seismic traces
bgar.c    exponential GAIN recovery, by range specification
bgaz.c    exponential GAIN recovery, by depth specification
bhed.f    up/down load selected header information from/to a text file
bhelp.c   this listing of BSU package contents
bhod.F90  hodogram by PCA to determine down-hole tool orientation
bint.f    numerical integration of seismic traces (trapezoidal rule)
bis2seg.c FORMAT CONVERSION: BISON ---> BSEGY  (no geometry setting)
bkil.f    either kill (delete) or zero seismic traces
bmed.f    median mix of seismic traces (spatial)
bmix.f    mean mix of seismic traces (spatial)
bmrg.f    merge traces from many files to a single file
bmrk.f    mark first break picks with a delta function on waveform
bmst.f    MASTER illustrates programing in BSU, FORTRAN
bnez.c    GEOMETRY: Create survey *.nez (Northing,Easting,Elevation) file
bnfd.c    MODELING:  computes near and far field in elastic whole space
bnos.f    MODELING:  generate band-limited random noise traces
```

5.0.2 man pages

From a terminal, type: `man program name`.

For example,
`man babs`

```
 babs(1)                    Basic Seismic Utilities BSU                    babs(1)

NAME
       babs - BSU program rectifies seismic traces (C-Language Version)

SYNOPSIS
       babs [  -h  |  infile   ]

DESCRIPTION
       Basic  Seismic  Utilities  (BSU) rectifies seismic traces by taking the absolute
       value.  Functionally equivalent to the master program, cmst.c, this  version  is
       cleaned  up  and  reflects the fact that there is only one command line argument
       necessary.  C-Language Version.

   Options
       -h     Online help giving details on command line arguments

       infile Input file name

   NOTE:
       If invoked with no options, will prompt user for input parameters.

   EXAMPLE:
       babs w001.seg

       File w001.seg is processed by babs.  Output traces are rectified.

FILES
       babsxxxx.seg
              named according to convention (first 4char babs, the next 4char  are  the
              first 4char of the input file name, suffix .seg)

       standard output
              produces a progress bar

       babsxxxx.lst
              Echo check of input parameters in listing file.

SEE ALSO
       bhelp(1),    cmst(1)
```

6 Plotting

BSU uses a number of ways to plot seismic data. Depending on how BSU is compiled, it can employ PLPLOT libraries, GNUPLOT libraries, OCTAVE, and old style line printer inspired text plots.

- **TRAPLT** 6.0.1 Line printer inspired trace and spectrum (ASCII text)
- **BPLT** 6.0.2 Plot seismic data with choice of output formats (PLPLOT or GNUPLOT depending on how compiled)
- **TPLT** 6.0.3 Plot seismic trace (GNUPLOT)
- **QPLT** 6.0.4 Plot seismic traces scaled by max amplitude (GNUPLOT)
- **CAPLOT** 6.0.5 down-hole dispersion and amplitude decay (PLPLOT or GNUPLOT)
- **Octave TRAPLT** 6.0.6 Octave version, trace and FFT spectrum
- **Octave YULE WALKER** 6.0.7 Octave ALL POLE spectrum
- **Octave SEISAZI** 6.0.8 Octave plot azimuth of down-hole horizontal components
- **Octave REFPLOT** 6.0.13 Octave first break analysis
- **Octave PROFPLOT** 6.0.11 Octave plot of a shot profile
- **Octave HODOPLOT** 6.0.9 Octave hodogram plot of two channels in same file
- **Octave HODO2PLOT** 6.0.10 Octave hodogram plot of two channels in different files

6.0.1 TRAPLT

Inspired by old school line printer plots, the code produces a text file, traplt.lst, that can be viewed with terminal commands like MORE or LESS, or in fullscreen editors like VI.

```
traplt infile tmin tmax trace# tzero ilin

infile: =name of input file
tmin:   =start time in seconds
tmax:   =end time in seconds
trace#: =trace number to list and plot
tzero:=zero time for phase reference; spectral plot
ilin:  spectrum plot 1=linear 0=dB
```

Here are some portions of the listing for an example case. The "j" column is the sample number, x(j), column is the sample amplitude in microvolts.

```
  max= 0.1924409E+06 min=-0.1765663E+06
  j            x(j) .....................................................
 51   0.4216680E+05 |                      .******                      |
 52   0.4289184E+05 |                      .******                      |
 53   0.3646188E+05 |                      .*****                       |
 54   0.2400264E+05 |                      .****                        |
 55   0.7193160E+04 |                      .*                           |
 56  -0.1169604E+05 |                     *.                            |
 57  -0.2911608E+05 |                  ****.                            |
 58  -0.4088844E+05 |                 *****.                            |
 59  -0.4132728E+05 |                 *****.                            |
 60  -0.2852460E+05 |                   ***.                            |
 61  -0.3510720E+04 |                      *                            |
```

```
62   0.2860092E+05 |                       .****                       |
63   0.6088428E+05 |                       .*********                  |
64   0.8744362E+05 |                       .************               |
65   0.1032228E+06 |                       .**************             |
66   0.1058558E+06 |                       .***************            |
67   0.9560988E+05 |                       .*************              |
68   0.7389681E+05 |                       .**********                 |
69   0.4399848E+05 |                       .******                     |
70   0.9291957E+04 |                       .**                         |
71  -0.2699820E+05 |                    ***.                           |
72  -0.6130404E+05 |               ********.                           |
73  -0.9099250E+05 |           ************.                           |
74  -0.1149951E+06 |        ***************.                           |
75  -0.1331593E+06 |     ******************.                           |
76  -0.1470877E+06 |   ********************.                           |
77  -0.1580969E+06 |  *********************.                           |
78  -0.1668928E+06 | **********************.                           |
79  -0.1731701E+06 |***********************.                           |
80  -0.1765663E+06 |***********************.                           |
81  -0.1765091E+06 |***********************.                           |
82  -0.1723306E+06 |***********************.                           |
83  -0.1633629E+06 | **********************.                           |
84  -0.1505412E+06 |   ********************.                           |
85  -0.1346094E+06 |     ******************.                           |
86  -0.1165979E+06 |        ***************.                           |
87  -0.9709806E+05 |          *************.                           |
88  -0.7677788E+05 |             **********.                           |
89  -0.5510304E+05 |                *******.                           |
90  -0.3155832E+05 |                   ****.                           |
91  -0.6506277E+04 |                       *
```

A portion of the spectrum listing:

```
 maxa=0.636E+07 mina=0.231E+03 maxp=0.180E+03 minp=-.180E+03 tzero= 0.000
 j   freq    amp    phz .....linear scale.....-180..........0.........+180
 1 .00000    0.3 -180.0 |*                      |*          .           |
 2    3.9    0.4   40.6 |*                      |           . *         |
 3    7.8    0.6  -93.3 |**                     |     *     .           |
 4   11.7    1.1  158.3 |***                    |           .        *  |
 5   15.6    2.0   70.8 |*****                  |           .   *       |
 6   19.5    3.3   -6.6 |********               |          *.           |
 7   23.4    5.2  -81.0 |************           |      *    .           |
 8   27.3    7.2 -155.5 |****************       | *         .           |
 9   31.2    9.0  129.5 |********************   |           .      *    |
10   35.2   10.0   54.8 |***********************|           .  *        |
11   39.1    9.8  -18.3 |********************** |         * .           |
12   43.0    8.7  -88.4 |********************   |     *     .           |
13   46.9    7.3 -154.4 |*****************      | *         .           |
14   50.8    6.0  143.6 |**************         |           .       *   |
15   54.7    5.1   83.5 |************           |           .    *      |
16   58.6    4.3   23.8 |**********             |           .*          |
17   62.5    3.5  -33.7 |********               |        *  .           |
18   66.4    3.0  -87.6 |*******                |     *     .           |
19   70.3    2.6 -139.2 |******                 |  *        .           |
20   74.2    2.3  169.6 |******                 |           .         * |
```

6.0.2 BPLT

Depending on how conditionally compiled, the program will either us PLPLOT or GNUPLOT libraries. The command line arguments are:

```
bplt  infile idev iorient itype 1tr Ltr tmin tmax istyl amp percnt xaxis yaxis

  infile   =  input file name
  idev     =  output device
          0=    xwin/wxt  (Linux/MS Windows)
          1=    Post Script
          2=    xfig
          3=    jpeg
          4=    PDF
  iorient  =  orientation
          0=    landscape
          1=    portrait
  itype    =  select non-time axis type
          0=    trace number
          1=    offset
          2=    geophone z-coord
          3=    geophone x-coord
          4=    geophone y-coord
          5=    shot z-coord
          6=    shot x-coord
          7=    shot y-coord
  1tr      =  first trace to plot
  Ltr      =  last trace to plot
  tmin     =  minimum time to plot
  tmax     =  maximum time to plot
  istyl    =  style of plot
          0=    wiggle plot
          1=   black/white variable area
          2=   black/grey variable area
  amp      =  amplitude for 1 trace deflection
  percnt   =  percent overplot 100= 1 trace
  xaxis    =  length of x-axis (non-time) in inches
  yaxis    =  length of y-axis (time) in inches
  (if xaxis and yaxis absent, 6.0 by 4.0 inches
```

```
bplt c008.seg 4 0 1 1 100 0 0.6 1 2.E+4 200 7.0 3.0
```

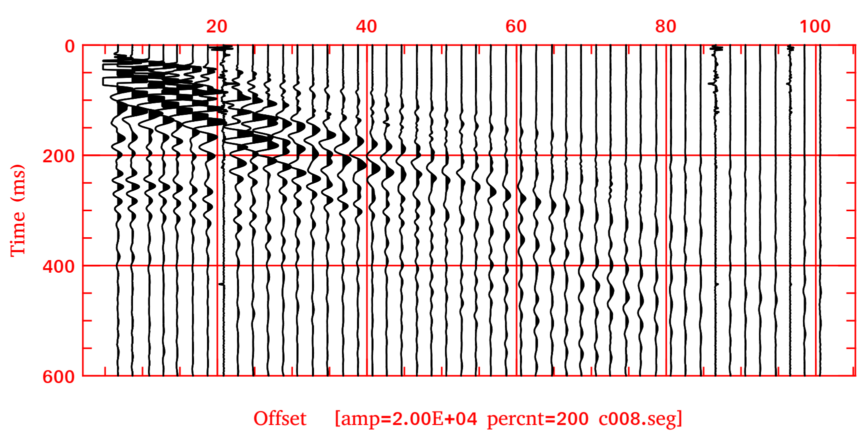


Figure 4: Example of a trace by offset in meters plot, written to file bplt.pdf

6.0.2.1 Trace Equalization

Program BPLT is a true amplitude plot, the x-axis label indicates the amplitude of a single trace deflection. At times, there is a need to see detail in both low and large amplitude portions of a shot gather. This can be done by running program BEQU on the data, and then plotting that. For example,

```
bequ c008.seg 0.  0.5
```

equalizes each trace amplitude using the L2 norm. Then run BPLT on file bequc008.seg.

```
bplt bequc008.seg 4 0 1 1 100 0 0.6 1 2.0 200 7.0 3.0
```

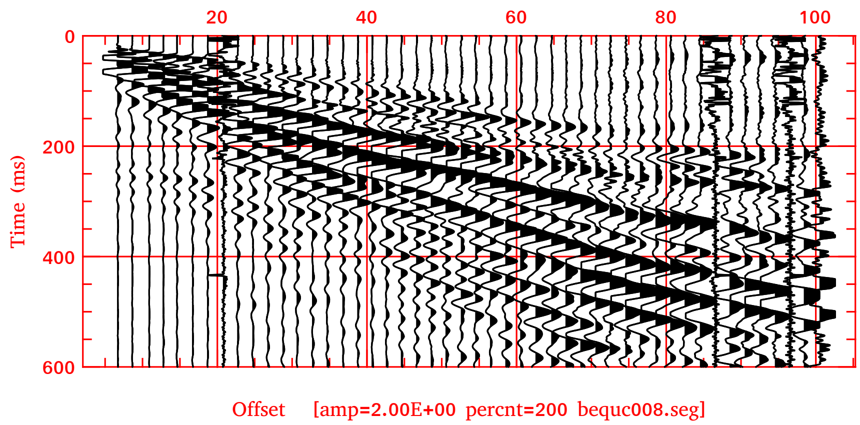


Figure 5: Trace equalized version of Figure 4

6.0.2.2 xplot bash script

As an example of how to set up a script to do the above sequence with the additional flexibility of scaling by one of the following normalizations:

- Peak Absolute Value of profile
- L2 Norm of profile
- Trace by trace L2 norm (this matches the above example in Figure 5)

```
#!/bin/bash
 OR=0   #orientation 0=landscape 1=portrait
if test "$1" = "-h"
then
echo "USAGE: xplot  filename  tmax  scaling"
echo 'Scaling Choices:'
echo '1= Peak Absolute Value of profile'
echo '2= L2 Norm of profile'
echo '3= Trace by trace L2 Norm'
else
if test "$1" = ''
then
echo 'Enter input file name'
read FILEN
else
FILEN=$1
fi
if test "$2" = ''
then
echo 'Enter tmax'
read TMAX
else
TMAX=$2
fi
```

```
if test "$3" = ''
then
echo 'Enter Scaling Choice'
echo '1= Peak Absolute Value of profile'
echo '2= L2 Norm of profile'
echo '3= Trace by trace L2 Norm'
read SCL
else
SCL=$3
fi

NAME=`basename $FILEN .seg`
NAME4=`echo $NAME | gawk -F "" '{print $1$2$3$4}' `
case $SCL in
1)
bscl $FILEN 1 5000 3
AMP=`gawk '/Peak Absolute Value/ {print $4}' bscl$NAME4.lst`
rm -f bscl$NAME4.*
PFILEN=$FILEN
bplt $PFILEN 0 $OR 0 1 500 0 $TMAX 1 $AMP 200
 ;;
2)
bscl $FILEN 1 5000 1
AMP=`gawk '/L2 Norm of Data Set=/ {print $6}' bscl$NAME4.lst`
rm -f bscl$NAME4.*
PFILEN=$FILEN
bplt $PFILEN 0 $OR 0 1 500 0 $TMAX 1 $AMP 200
;;
3)
bequ $FILEN 0 $TMAX
PFILEN=bequ$NAME4.seg
AMP=4
bplt $PFILEN 0 $OR 0 1 5000 0 $TMAX 1 $AMP 200
rm -f bequ$NAME4.*
;;
esac
rm -f bplt*.lst
fi
```

6.0.3 TPLT

This program plots a single trace to an X11 screen. The command is

```
tplt  infile  trace_number tmin tmax

    infile  = input file name (4char minimum)
    trace_number = trace number to plot
    tmin = minimum time in seconds
    tmax = maximum time in seconds
```

It also outputs a GNUPLOT file, graph.gp which can be edited or not, and run as a bash script. For example, to create a PDF file of the plot, comment out (insert a # symbol at the beginning of the line) the “set terminal” command and replace it as follows:

```
#set terminal x11 persist
set terminal pdf
set output "graph.pdf"
```

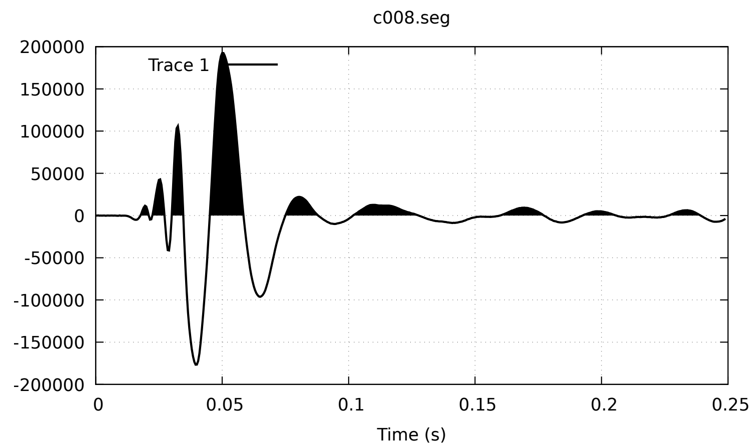


Figure 6: TPLT: Plot of the first trace in the file c008.seg. Units are microvolts if only instrument corrections have been applied. Of course that will change depending on the processing history.

6.0.4 QPLT

A quick quality control plot in which each trace is scaled by its maximum value and the displayed by GNUPLOT to the X11 window. One can modify the display interactively. Pressing enter in the terminal will freeze the plot. Also output is a file, qgraph.gp, which can be edited for an alternative terminal. The program can be run with the following command line arguments:

```
qplt  infile  tmin tmax

    infile  = input file name (4char minimum)
    tmin = minimum time in seconds
    tmax = maximum time in seconds
```

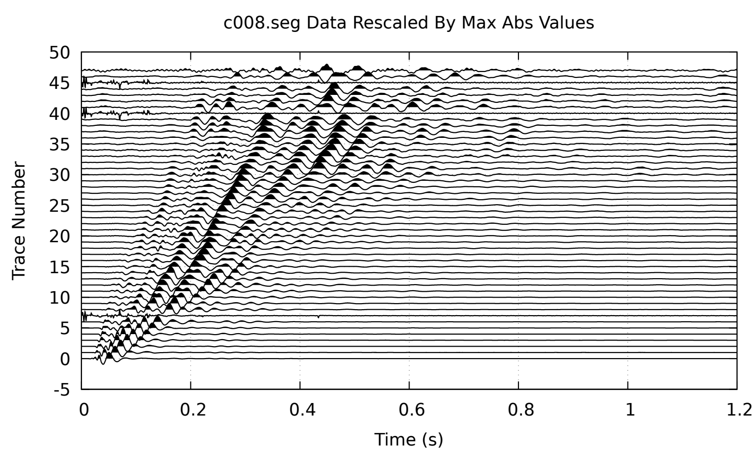


Figure 7: QPLT: Quality control plot showing traces, **each scaled by the maximum value in the trace**. Output includes X11 and qgraph.gp (GNUPLOT). Edit qgraph.gp to change terminal type (ie. postscript) and run qgraph.gp with gnuplot.

6.0.5 CAPLOT

Down-hole surveys can determine stiffness and damping of soils in shear. Two BSU programs, are used to measure S-wave velocity dispersion (BVAS) 8.3.5 and amplitude decay with distance traveled (BAMP) 8.3.6. These programs produce two files, bvas.his and bamp.his which can then be used in a joint inversion scheme to determine stiffness and damping. Program CAPLOT may be used to create an image file displaying the dispersion and decay measurements with 95% confidence bars.

```
caplot  bvas_file bamp_file emin emax well year date idev

bvas_file =input file ( bvas.his)
bamp_file =input file ( bamp.his)

Title info from bvas and bamp runs:
 emin     =minimum elevation (real)
 emax     =maximum elevation (real)
 well     =well name (char 4)
 year     = year of survey (integer, 4 digits)
 date     = 4 digit integer mmdd
 idev     =device for plotting
         0=X window display
         1=Post Script *.ps file
         2=Xfig *.fig file
         3=JPEG *.jpeg file
         4=PDF  *.pdf  file
```

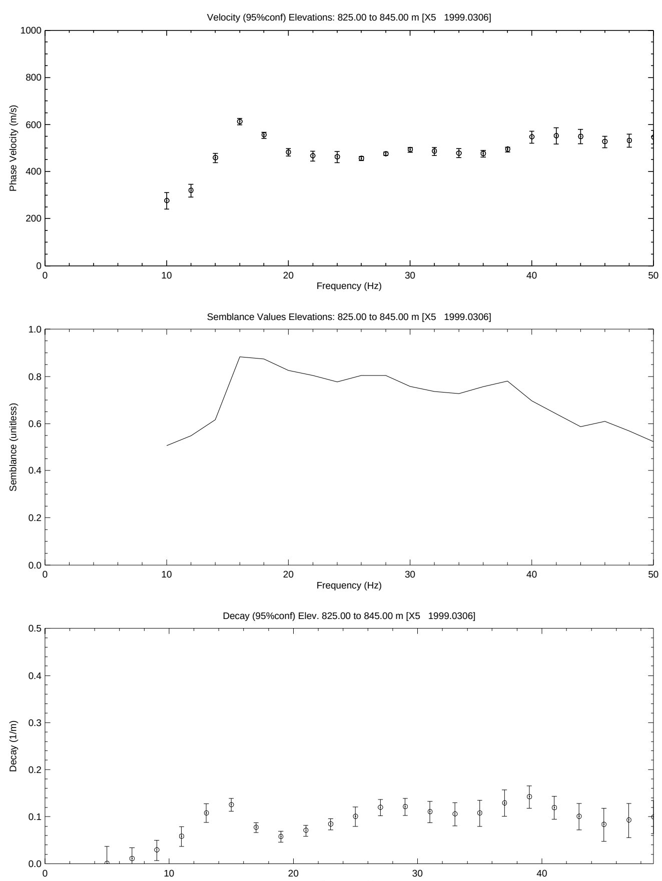


Figure 8: CAPLOT: Display S-wave dispersion and amplitude decay from program outputs of BVAS and BAMP programs.

6.0.6 OCTAVE TRAPLT

This is the octave version of TRAPLT. The following files are required to be in the directory where octave is started.

- **bsegin.m** Reads traces from BSEGY files
- **segyinfo.m** Reads header information from BSEGY files
- **traplt.m** Actually does the plotting

```
 Start an octave session and then type
        traplt

You will be prompted for a file name, channel
phase reference, and maximum frequency to
display.

First shows a plot of the selected trace, mouse
used to pick time zero for phase. Click OK then use mouse.

Uses FFT for amplitude spectrum
```

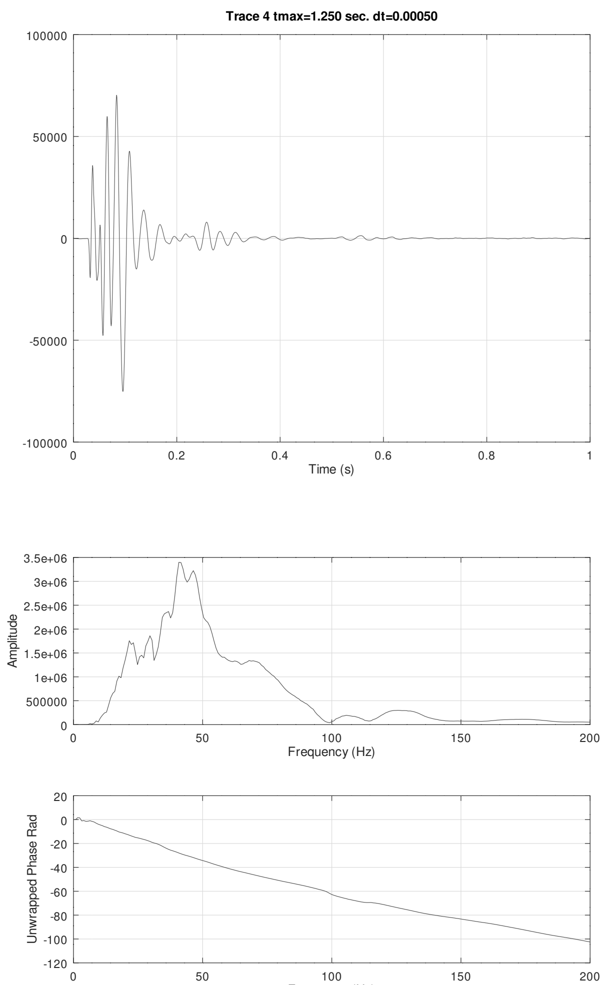


Figure 9: OCTAVE TRAPLT: Octave version of TRAPLT program. Octave is a mathematical interactive program like Matlab. Compare this plot to the all pole yule walker spectrum of Figure 10

6.0.7 OCTAVE YULEWALKER

This program computes the ALL POLE spectrum for a signal. In addition to the octave plots, a GNUPLOT (plotspec.gp) file is output along with a data file yw.dat which is readable by the plotspec.gp file. The code is interactive and one uses the mouse to pick the order of the process on the autocorrelation.

```
 The yulewalker.m program is run in octave.
You can run it on BSEGY data or
you can run it on the autocorrelation of
BSEGY data.

In either case, the spectrum will be an
all pole spectrum.  You will be prompted
to use your mouse to select the maximum
lag in the autocorrelation.

If you run BXCR followed by BSTK, you can
compute an average autocorrelation for the
entire shot gather, and run yulewalker.m on
that.

Other *.m files required (bsegin.m and segyinfo.m)
must be in the same directory before yulewalker.m
is run in an octave session.
```

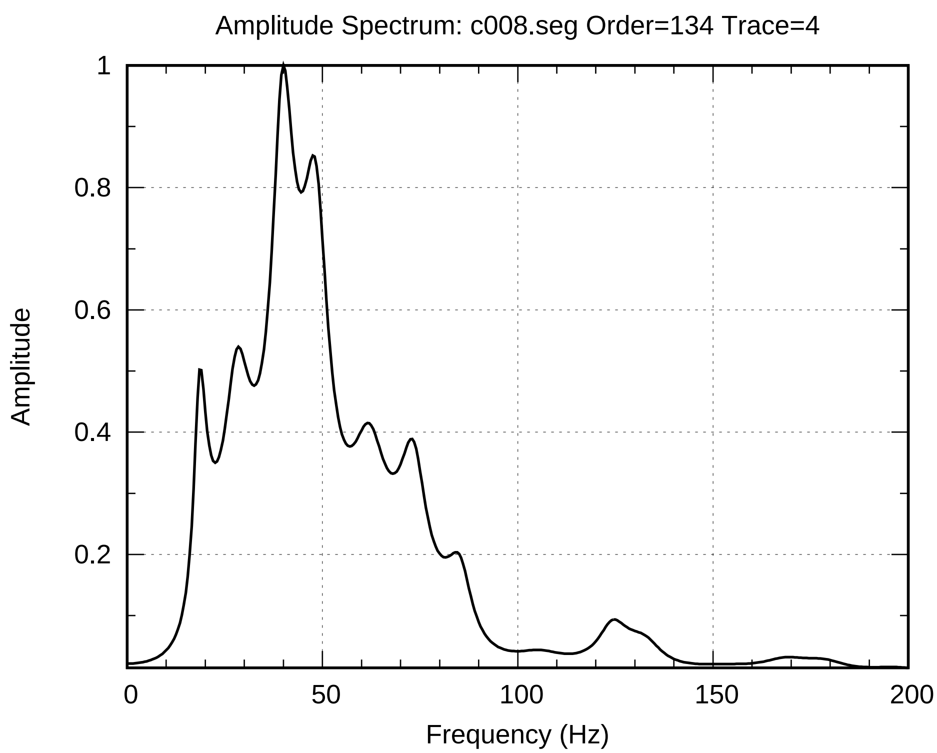


Figure 10: OCTAVE YULE WALKER: Octave program which computes the ALL POLE spectrum. Input can be either a seismic trace data or an autocorrelation of trace data (either must be in BSEGY format, *.seg file). Compare to Figure 9 FFT plot. See BXCR 12.0.16 for how to create an autocorrelation as input.

6.0.8 OCTAVE SEISAZI

This program is run in an octave session. Start octave and type seisazi. The program will request an *.seg file name. It will then display a GUI showing the number of traces and the sample interval. Click on OK. The program will generate a plot of the horizontal component geophone. In a typical application, one extracts a single horizontal component from a collection of multi-component files using the BMRG program. The information is extracted from the headers.

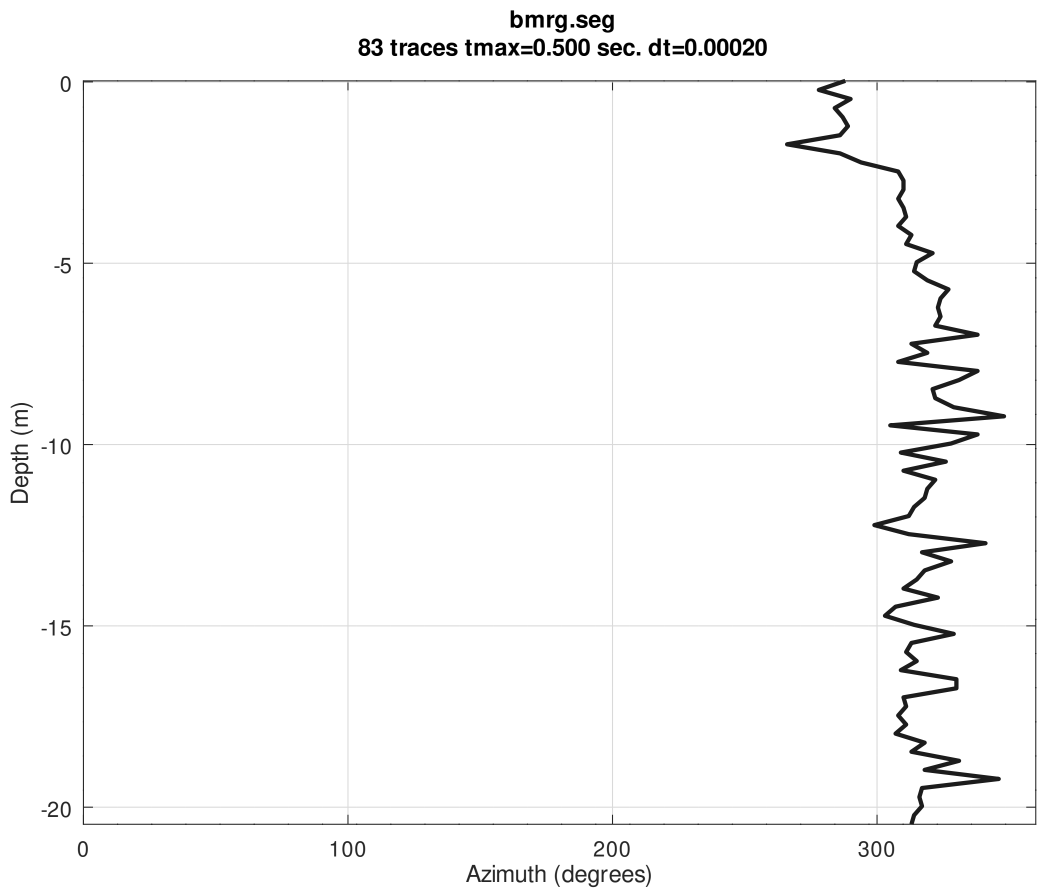


Figure 11: OCTAVE SEISAZI: Plots a horizontal component azimuth from the headers of an *.seg file. Here, the plot is of the T-component from a down-hole survey. Phone orientation was determined using program BHOD.

6.0.9 OCTAVE HODOPLOT

This program is run in an octave session. Start an octave session and type hodoplot. This program is for the case of two components in the same *.seg file.

```
 Prompts follow:
1. enter file name.  Example: 1001.seg hardwired as a 6 channel shot record with 3
components down-hole and 3 components stationary reference phone at the surface.
Down-hole Phone
ch1=Vert
ch2=Radial
ch3=Transverse

Reference Phone
ch4=V
ch5=R
ch6=T

2. GUI, informative.
3. GUI, choose either the down-hole or surface reference phone.
4. GUI, choose component for X-axis, say T
5. GUI, choose component for Y-axis, say R
6. GUI, choose a scale factor, or default
7. GUI, choose a Tmax, say .05 seconds
8. GUI, click continue
9. GUI, choose the next Tmax, say 0.10 seconds
10. GUI, click continue

KEY POINTS:
* There will always be a sign convention.  Here, on the vertical phone,
upward velocity produces a negative voltage.  Do a tap test for your
equipment.

* The hodoplot.m program is hard wired to relate components to the GUI
choices.  If your phones use different channels for V, R, T, then you may
need to modify the code.  By the way, R and T are just arbitrary labels
considering that down-hole phones will twist there orientation as they
travel up or down the hole.
```

What you are doing is progressively working down in time, plotting the particle motion. With each step, just change the Tmax value, the Tmin value is automatically adjusted to the last step tmax value. Here the first arrival largest motion is in the direction of the R component. This means that it is mostly aligned with the source blow to the West. The T component is oriented mostly orthogonal to the source.

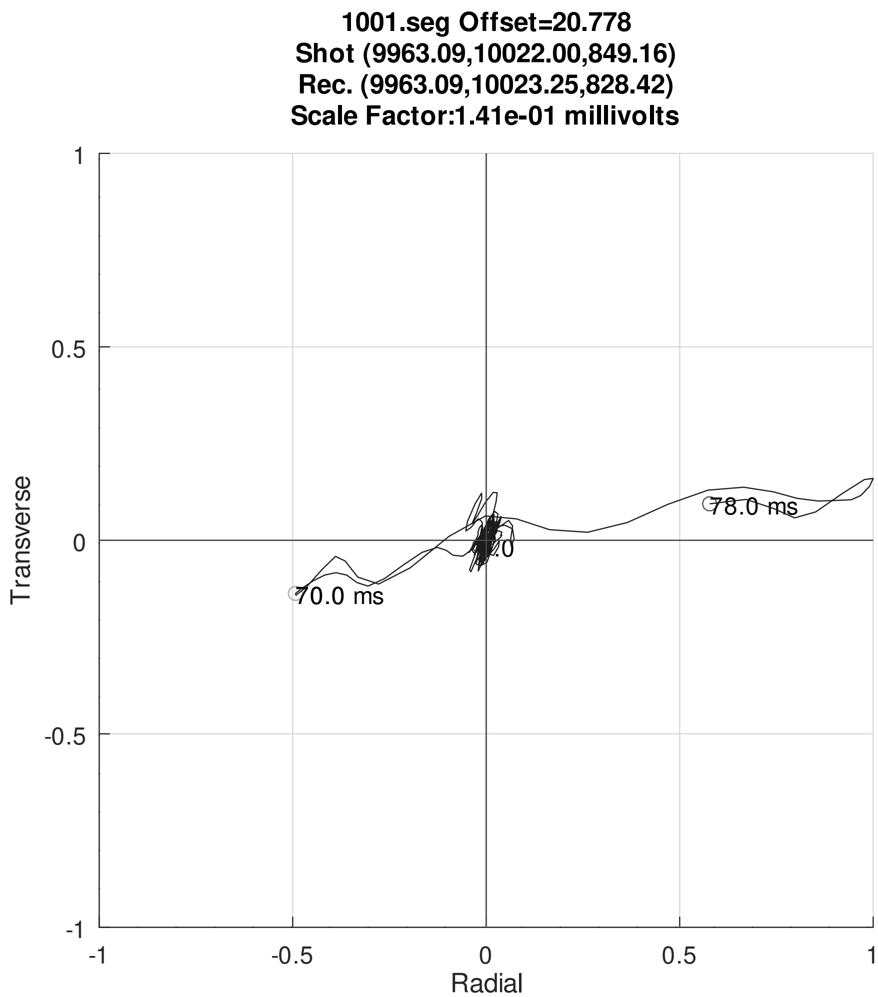


Figure 12: OCTAVE HODOPLOT: Plotting particle motion on the down-hole horizontal R- and T- components which are channels in the same *.seg file. If components are in different files, use HODO2PLOT program instead (see 6.0.10).

6.0.10 OCTAVE HODO2PLOT

This is an octave program that reads two *.seg files for plotting a hodogram. The files are checked to insure that the sample intervals and other parameters match.

```
 Start an octave session and type
        hodo2plot

This hodo2plot.m file is different from hodoplot.m in that the components
to be plotted are from two different *.seg files, rather than a single
*.seg file.  An example would be a Rayleigh wave problem
where the vertical and radial components reside in different files.

1. prompt for which file is X-axis, which is Y-axis.
The code will check that the same number of traces and time sampling are
used for both files.
2. GUI, choose a trace to plot.
3. GUI, choose scale factor or just default.
4. GUI, choose a Tmax value, say .05 seconds.
5. GUI, click continue
6. GUI, choose a new Tmax value, say .10 seconds
7. GUI, click continue
What you are doing is plotting the particle motion for increasing
time steps.
KEY POINT:
* There will always be a sign convention. Upward velocity produces a negative voltage on vertical
phone.  Do a tap test for your equipment.
```

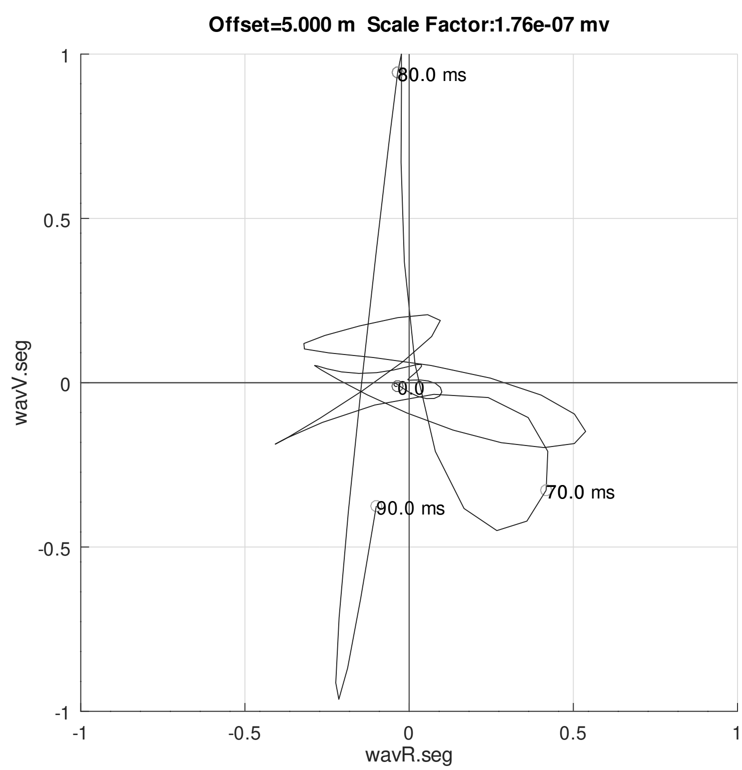


Figure 13: OCTAVE HODO2PLOT: Plotting particle motion on the Radial and Vertical components of a Rayleigh wave problem in which the channels reside in different *.seg files. If components are in a single file, use HODOPLOT program instead (see 6.0.9).

6.0.11 OCTAVE PROFPLOT

This program just does a simple trace plot of a shot gather in octave. Depending on your octave installation, it is likely that you can zoom in for detail on the plot. This program also serves as an example of how to read a BSEGY *.seg file. Requires segyinfo.m and bsegin.m in the same directory.

```
 Start an octave session then type
        profplot

The program will prompt for a file name. Type a full file name like:
        c008.seg
for example.
```

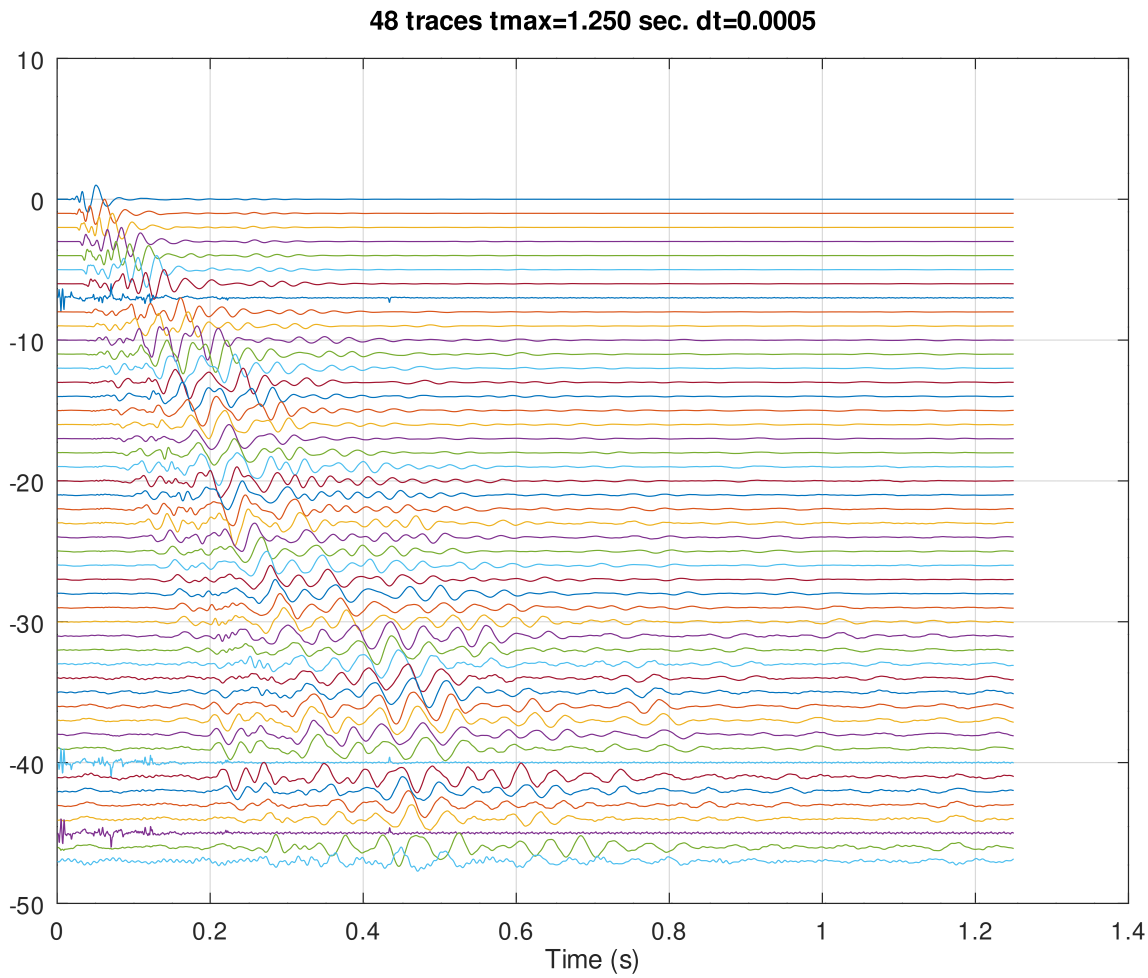


Figure 14: OCTAVE PROFPLOT: Plots a shot gather of traces in BSEGY formated file, *.seg. Traces are individually scaled by the maximum value. Compare to images Figure 5 and Figure 7.

6.0.12 OCTAVE SEGPIC

The program segpic.m is run in octave to plot each trace and permit picking with a mouse. The output is a file ending in *.pic. Program BPIC can be used to insert the pics into the *.seg file headers. See section 8.5 for an example of using **segpic.m** in conjunction with datuming program BREF 8.5.6. Also see BPIC 8.5.3.

```
 Start an octave session, then type
    segpic

1. prompt for file name, like k007.seg for example.
2. GUI shows number of traces and tmax.
3. GUI prompt for a clip factor and reduced tmax for good
plotting resolution.  Suggest clip factor of 3 and maybe .1 for
tmax, depending on when arrivals come in.
4. GUI page through each trace, using mouse to pic first arrival,
typically a down motion with SEGY polarity conventions.

Pics are output to an ascii *.pic file.  Trace (number, pick time):
1  0.05695853
2  0.05534562
3  0.04336406
4  0.05281106
5  0.05396313
for example.

Use program BPIC to insert pics to headers, for example:

  bpic k007.seg 1 k007.pic 0.

The above command would be executed from an terminal, after octave
session is ended.
```

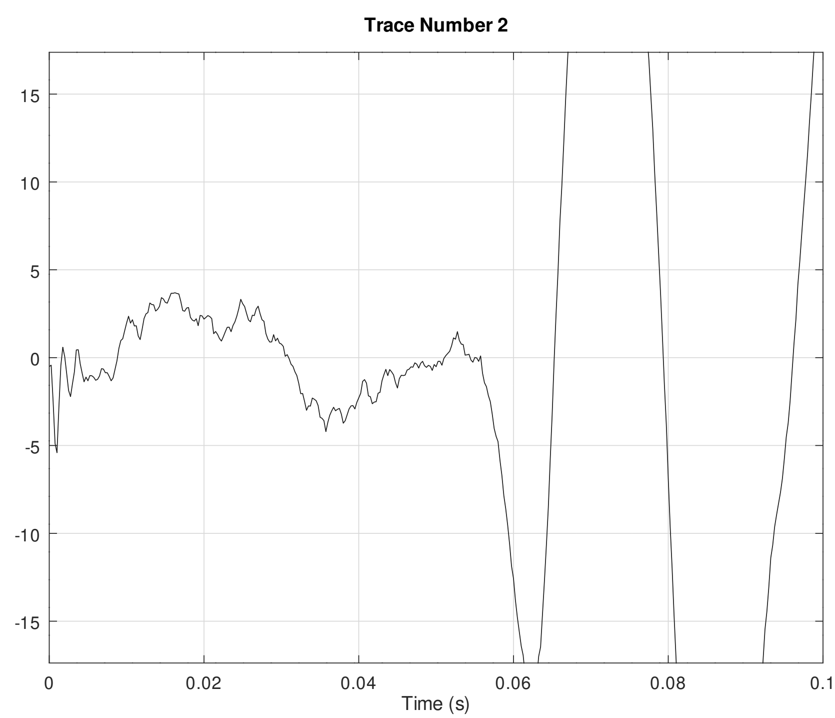


Figure 15: OCTAVE SEGPIC: Example of a trace for picking with mouse. First arrival refraction is at about 0.055 seconds.

6.0.13 OCTAVE REFPLOT

This program can be used to both plot and measure apparent velocities of refraction arrival time picks. It is up to the user to know how to determine which arrivals are refractions. The first arrival picks must have been done first and inserted into the *.seg file headers (see programs segpic.m, BDAT 8.5.5, BPIC 8.5.3, PICRESTORE 8.5.1).

```
 Start an octave session, then type
     refplot

One is prompted for the file name. For example:
     k008.seg

Choose either stations or offsets
Pick a segment to get started,
Click yes
Then 3 mouse clicks, click a near offset
then a far offset limit, a line will be
fit (OLS), a third click will print the
estimated velocity and 95% confidence
values where you click on the plot.
Chi^2 info GUI, then choose to do another
segment or not.

When picking the near and far offset
limits, only the horizontal, x-axis
position of the mouse matters.

A Postscript output file, plot.ps, is created and
can be viewed with ghostscript.
```

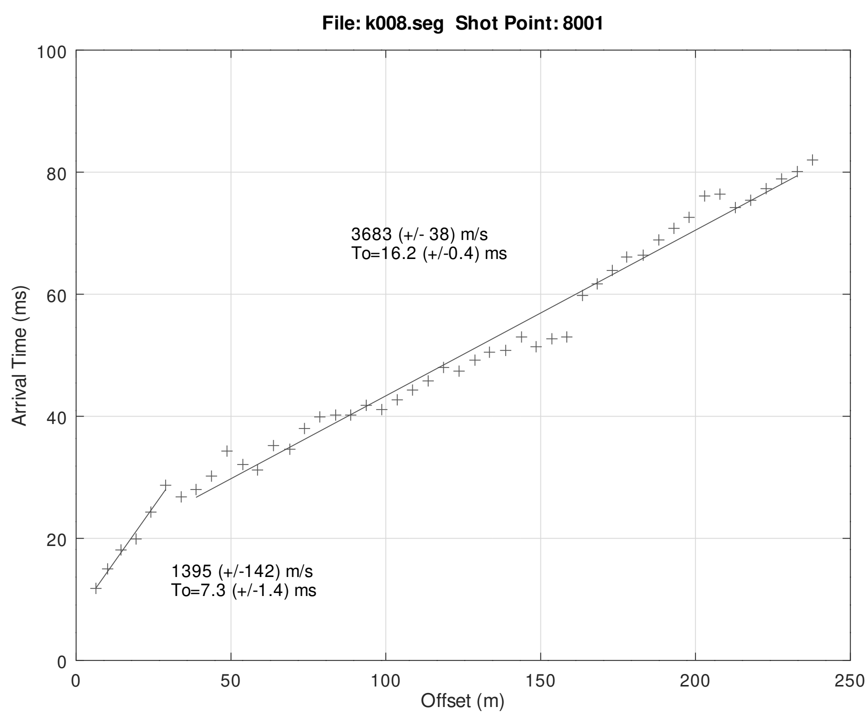


Figure 16: OCTAVE REFPLOT: Plots first break picks which have been added to headers with BPIC. Then use mouse to pick line segment (start,end), followed by a mouse click to plot refractor apparent velocity result. See section 8.5.6, estimating a cross-over distance for program BREF.

7 Surface Seismic

7.0.1 BRED

Correctional velocity can be applied to a data set to static shift data into a linear alignment (direct waves or refracted head waves). Alternatively, one can apply hyperbolic (NMO, reflection) correction to the data. Flattening the data on a refracted arrival can make picking first breaks easier in some cases (see section 8.5)

```
bred  infile  iopt  vred  t0  tshift

infile:  =name of input file
iopt:    =statics option
        1=linear trend
        0=hyperbolic NMO trend
vred:    =reduction velocity
t0:      =zero offset time (iopt=0 ONLY)
tshift:  =bulk static shift
```

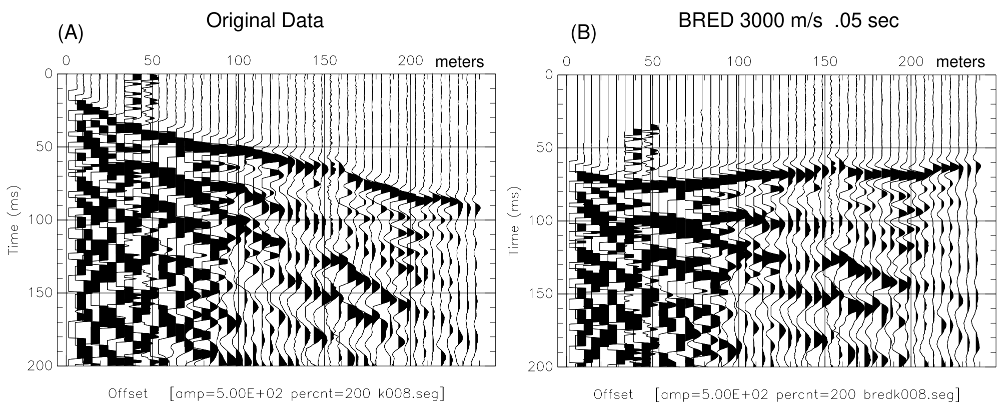


Figure 17: (A) Plot of a shot gather, (B) BRED: linear trend, 3000 m/s reduction velocity, .05 seconds offset. See section 8.5.0.1 for an example of picking data with BRED.

7.0.2 BVAX

Run BVAX for surface wave dispersion measurement. A number of image files are created, and the file bvax.his is available for use in the inversion program invR1.m (run in octave). To run invR1.m in octave, execute build_disper_oct script to build an extension to octave. Edit the bvax.his file to remove any measurements that are zero or bogus velocities. **NOTE: BVAX determines *PHASE* velocities in the time domain**. The code has been revised to sort by offset before the limiting of the filter step. This permits correct inclusion of offsets in split spread and other types of acquisition geometries. Older versions, up to bsu-3.0.2 will require running BSRT 12.0.6 program first.

```
 bvax infile xmin xmax vmin vmax nvel . . .
          fmin fmax delf bwd iskp ivscn

infile  =input file name
xmin    =minimum offset (float)
xmax    =maximum offset (float)
vmin    =minimum velocity
vmax    =maximum velocity
nvel    =number of velocity increments
fmin    =minimum frequency Hz
fmax    =maximum frequency Hz
delf    =frequency increment Hz
bwd     =filter bandwidth Hz
iskp    =skip filtering (if files already exist)
          1=YES  0=NO (-1=NO and delete when done)
ivscn   =output velocity scan data sets
          1=YES  0=NO

EXAMPLE: bvax c008.seg 1.0 100. 100. 500. 200 10. 50. 1. 1. -1 0
```

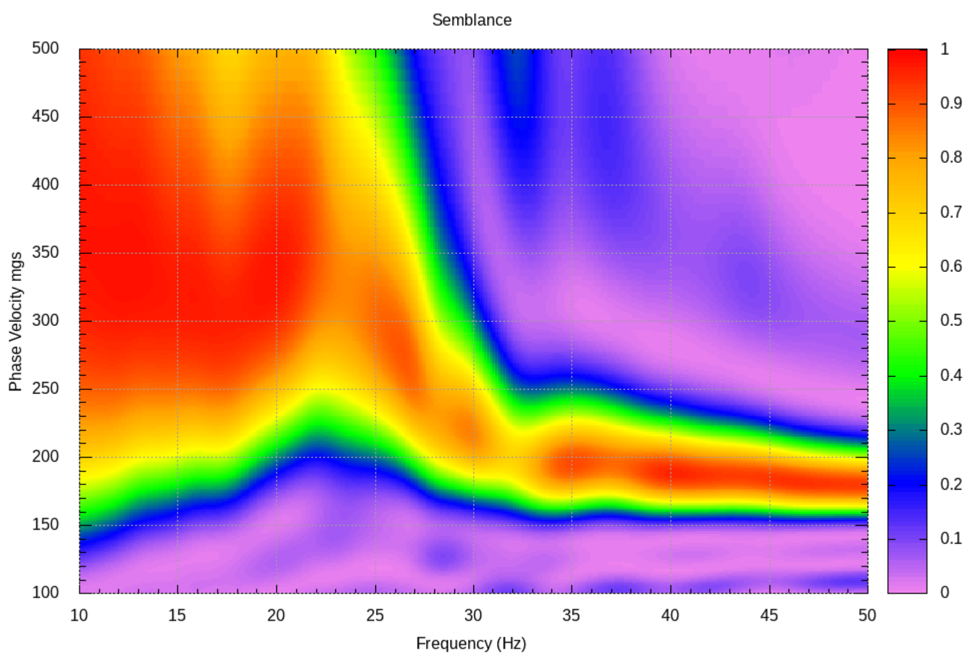


Figure 18: BVAX: Phase velocity semblance display file, clrplot.png. For details on semblance, see Sheriff (1991). Semblance provides a measure of the degree to which the data were aligned at a trial velocity. Offset range was limited: 5 → 19 meters.

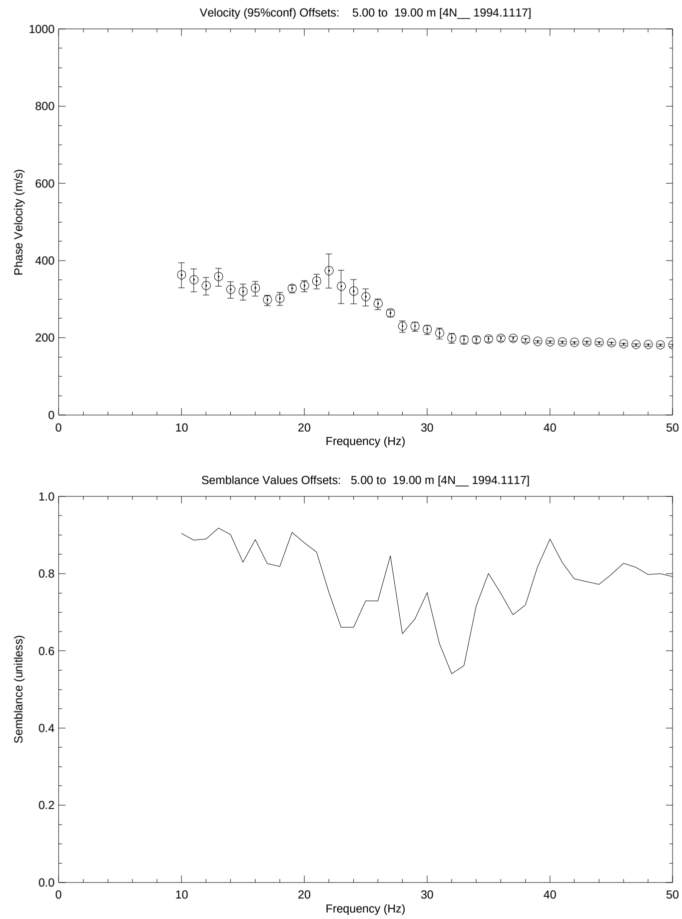


Figure 19: BVAX: Phase velocity semblance display file, bvax.ps. These are the autopicks from figure 18.

One must decide which picks are fundamental mode, which might be higher modes. The file, bvax.his should be reviewed and edited where necessary to remove questionable data, or in some cases where zero velocity is returned due to a failure to find a phase velocity. That happens when the range of velocities scanned is too limited, or when there is no signal. Once edited, an inversion in Octave can be done with program invR1.m (8.2.1).

7.0.3 BAMX

Program BAMX computes amplitude decay with frequency. The code attempts to measure the viscoelastic alternative to an elastic earth. It is similar to the BAMP code which is used in down-hole measurements of viscoelasticity. BSU software does NOT have code to invert surface wave data under a viscoelastic representation, at this time. In this case, the decay is modest, but if large decay were present, one might wish to develop a viscoelastic surface wave inversion code.

```
bamx infile rmin rmax fmin fmax delf bw tmax

infile  =input file name
rmin    =scan gate: (min_offset)
rmax    =scan gate: (max_offset)
fmin    =min band pass center frequency (Hz)
fmax    =max band pass center frequency (Hz)
delf    =frequency step (Hz)
bw      =bandwidth of filter (Hz)
tmax    =time gate: max_time
```

8 Inversion Codes

Forward problems take an earth representation and compute a corresponding geophysical expression of that representation. The inverse problem goes the other way and computes an earth representation from geophysical data. Basic Seismic Utilities (BSU) is focused on near surface problems. The typical representation of the earth is a soil profile with S- or P- wave velocities as a function of depth being the object of interest.

8.1 Objective Functions

The general inversion procedure compares field data or features extracted from field data to those of synthetic data. The physics and corresponding forward problem software are used to generate the synthetic data for a given subsurface profile. When comparing field and synthetics, a method to measure distance must be first chosen. Here, distance is a measure of how closely synthetic and field match. An objective function employs that measure of distance, and the goal is to minimize the distance by a chosen inverse method. Once minimized, the corresponding soil profile which generated the synthetic data is taken to be a likely representation of the actual subsurface conditions.

By far, the most popular objective functions employ the L2 (sum of squared differences) norm. In some cases, the L1 norm (sum of absolute value of differences) is chosen when outliers in the data are evident. It is likely that different objective functions will lead to different solutions for the same field data. Thus, all inverse problems are non-unique until some way to measure distance is chosen.

Basic Seismic Utilities (BSU) software follows this L2 practice with one exception. Wave form inversion of surface waves takes a different approach in the BWFI 8.2.4 code. When comparing features like dispersion curves, the L2 approach is use. When comparing Rayleigh wave shot gather wave forms, the objective function mixes an angle between the data vectors, modified by an arrival time component that addresses the possibility of non-overlap that may occur when the data are very dissimilar.

For wave form inversions, see the BOBF 12.0.20 code for details on how the angle between seismic trace is modified (see BAGL 12.0.19) to address the non-overlap possibility.

8.2 Surface Waves

The surface waves of interest in BSU software are Rayleigh waves. These are a mixture of SV- and P-wave motion that satisfy Hookes law $F = k \cdot x$ and Newton's law of motion, $F = m \cdot a$. The particle motion is largely elliptical and can be measured on both vertical and in-line radial (horizontal) component geophones. BSU codes compute features from seismic data, specifically a dispersion curve. The soil profile representation is 1-D, varying only in depth. Inversion is done in Octave. The difference between **SASW** (section 8.2.2) and **saswv** (8.2.3) codes is in the type of file read. **SASW** read a BSEGY formatted file, **saswv** reads a text file of cross power spectrum. Program **invR1** reads a bvax.his file (see 8.2.1 and 7.0.2).

8.2.1 OCTAVE invR1, Rayleigh Wave Inversion

This program uses the BVAX output file, bvax.his, to invert Rayleigh wave, fundamental mode, under an elastic representation. See section 7.0.2 for details on BVAX. A companion forward problem octave code is **FwdR1.m**, see section 9.2.1.

To run invR1.m in octave, first execute **build_disper_oct** script to build an extension to octave. Edit the bvax.his file to remove any measurements that are zero or bogus velocities. Program invR1.m is hard wired to read bvax.his, so that should be the name of any edited file that will be used by invR1.m. The octave files are located at the `/usr/local/share/octave/site-m/` directory.

The code is an iterative inversion which runs for a user number of inversion steps. Default is 2, but recommend 5 as a useful number. Increasing the number of singular values employed will provide additional detail in the inversion result. However, if you use too many, noise in the data may inject details in the result that are not reliable. Or the code can become unstable if too many singular values are used. EXAMPLE:

```
1). Enter initial soil representation file:
File model.txt is used to set an initial model of control points
For example, with 3 control points:

3
200 300 500
.0 2.0 15.

Velocity   Depth
200 m/s    0 m
300 m/s    2.0 m
500 m/s   15.0 m
2). GUI Choose P-wave velocity option. Click on Vp/Vs ratio OR Vp=fixed (if fixed,
GUI enter Vp m/s and Density kg/m3
3). GUI Choose density parameters, Poisson Ratio, grain density, porosity,
degree water saturation.
4). GUI Informs user of Vp/Vs ratio and constant density to be used.
NOTE: Code will seek a S-wave velocity profile consistent with these results.
5). GUI Select number of singular values to use, layer thickness (constant in meters),
locksw (switch to lock some conditions), and number of iterations to do.  Typically,
5 or more are good, but for the first run, 2 is wise in case things go sidewise.
locksw   meaning
0        free bottom control, velocity and depth
1        (default) lock bottom depth, free bottom velocity
2        lock bottom velocity, free bottom depth
3        lock both bottom velocity and depth

The bottom is the deepest control point, top of the bounding half-space.
Program FwdR1.m is a manual forward program that can be used to do the inversion
manually, or to explore the fast and slow limits based on confidence limits.
```

The edited bvax.his file from the example in section 7.0.2 was run using a constant VpVs ratio for just 5 iterations. Density was held constant. Other settings from a GUI are Poisson Ratio=0.33, grain density 2.67 g/cc, porosity 0.33, degree water saturation 1.00. This results in VpVs=1.99 and density=2169 kg/m^3. Only 3 singular values included, layer thickness, deltz= 0.1 meters. The resulting velocity model
Other output includes text files of the solution as well as a fast and slow 95% limit cases.

```
solution.txt (Three rows: number of control, S-velocities, Depths)
 3.00000000e+00
 9.28725027e+01 2.76581965e+02 5.00000000e+02
 0.00000000e+03.00000000e+00

 slow.txt
 9.08153279e+01 2.75856068e+02 5.00000000e+02
 0.00000000e+00 3.59321540e+00 1.50000000e+010 3.58000000e+00 1.50000000e+01

 fast.txt
 3.00000000e+00
 9.49296774e+01 2.77307861e+02 5.00000000e+02
 0.00000000e+00 3.56678460e+00 1.50000000e+01
```

8.2.1.1 Solution Uncertainty The above *.txt files can be converted to alternative slow and fast plots of the soil profiles showing S-velocity with depth.

- One can uncomment the two plotvel() functions under the Plus and Minus Limits section of the invR1.m code. These are lines 620 and 630 of the current version of invR1.m Running the octave program, invR1.m, with the same parameters, but now with the fast and slow plotvel() calls will display the solution surrounded

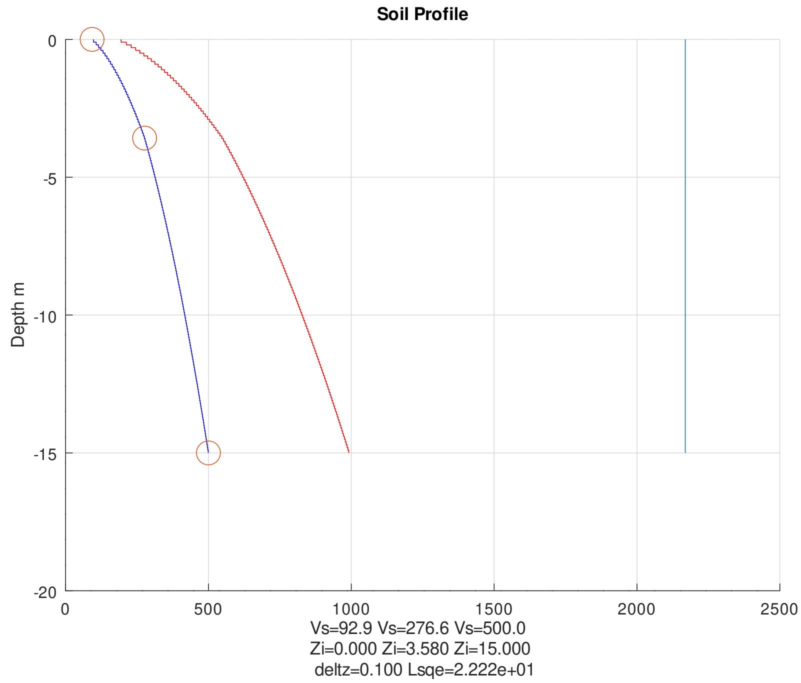


Figure 20: invR1: After 5 iterations, the resulting soil model is shown. The S-wave velocity with inverted control points is shown as the Blue curve (m/s). The Red curve is the P-wave velocity, and at the far right is the constant density (kg/m^3)

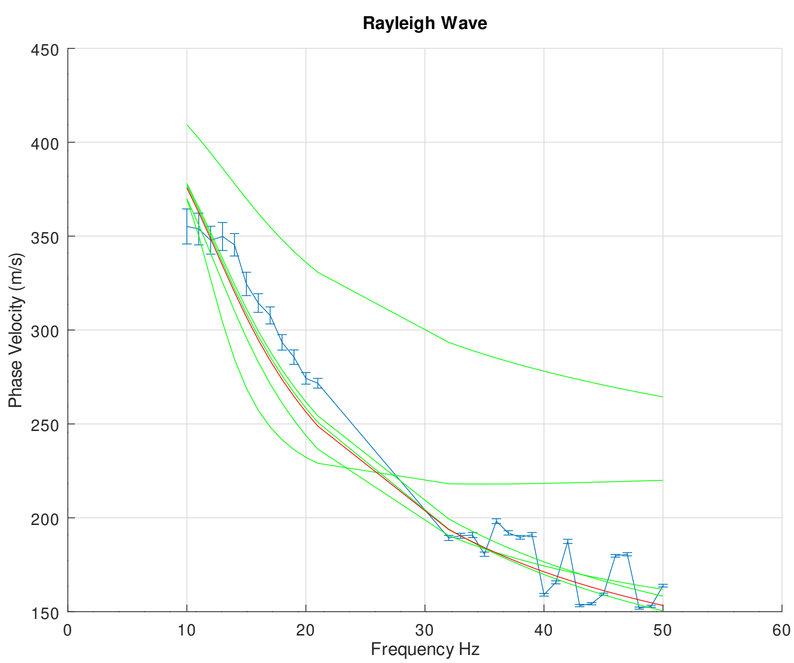


Figure 21: invR1: Progress of the inversion. The initial model dispersion is the fastest green curve. The green curve is the dispersion after 5 iterations. Data from bvax.his is in blue.

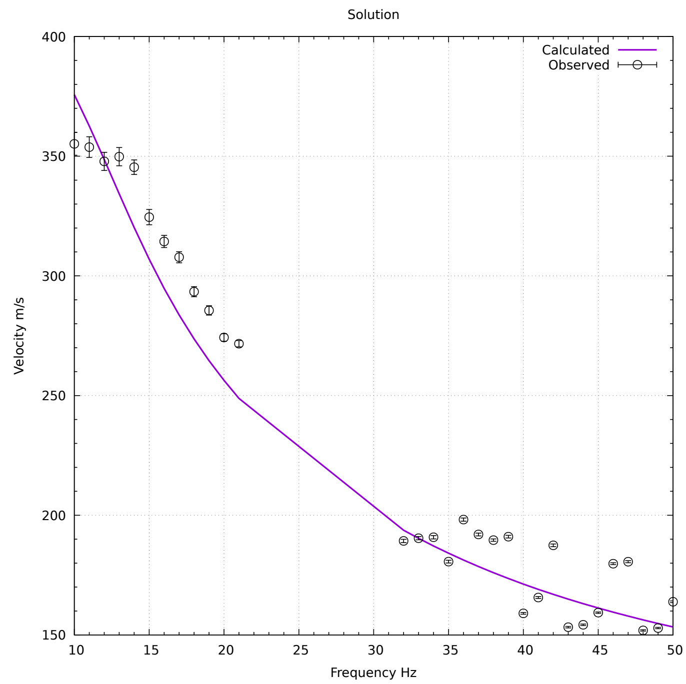


Figure 22: invR1: The code also generates a GNUPLOT file, dispcrv.gp, which shows the final solution when run with the gnuplot program.

by the fast and slow solutions. In some cases, there will not be much difference, but using the zoom function on Figure 2 of the octave program output can be used to see the difference.

- The other option is to use the octave program, FwdR1.m (see section 9.2.1), and when prompted for the model, enter either fast.txt or slow.txt (instead of model.txt) to compute these cases and their fit to the data. After the first plot, end the program with the GUI and it will generate additional plots showing both the fast or slow model and fit to the data.

8.2.2 OCTAVE SASW

In theory, only two traces are needed to compute a dispersion curve. Program SASW.m permits one to select two traces and compute Rayleigh-wave velocity dispersion. Depending on the trace spacing and spectral selection, the code recommends a maximum spacing between the two traces (to avoid aliasing).

```
 Start an octave session and then type
       SASW

Note, capital letters are important since that
agrees with the file SASW.m

This code takes two signals from a shot gather to
compute a cross spectrum leading to a dispersion
curve.

Prompts:
1). enter file name, example: c008.seg
2). GUI Pop up to select fmin, fmax vmin vmax
3), Info GUI pops up and shows both time and
spatial sample intervals.

Recommended trace separation is indicated on
last line. If high frequencies are chosen, then
too large a separation between the two geophone
stations can lead to aliasing.

4). GUI enter tmax, near trace number, far trace number.
For example:
tmax =1.0
trace R1 = 2
trace R2 = 3
(this would follow a recommendation that
R2-R1 be no larger than 1)

When only two offsets are used,
one should always look at the entire shot gather
first and select traces likely to be dominated by
the fundamental mode (typically close to the source).
```

The program produces two figures. One shows the cross power spectrum and coherence (Figure 23). The other figure shows the dispersion over a range of frequencies selected in the GUI prompt when the code is run (Figure 24).

The code determines a time shift, Δt, between geophones with a separation of Δx to compute a phase velocity at each frequency of interest. If Φ is the unwrapped phase angle at a frequency of interest, then

$$\Delta t = \frac{\Phi}{2\pi} \cdot T \quad , \tag{1}$$

where T is the period for a frequency f ($T = 1/f$). The phase velocity at the frequency f is

$$C(f) = \frac{\Delta x}{\Delta t}. \tag{2}$$

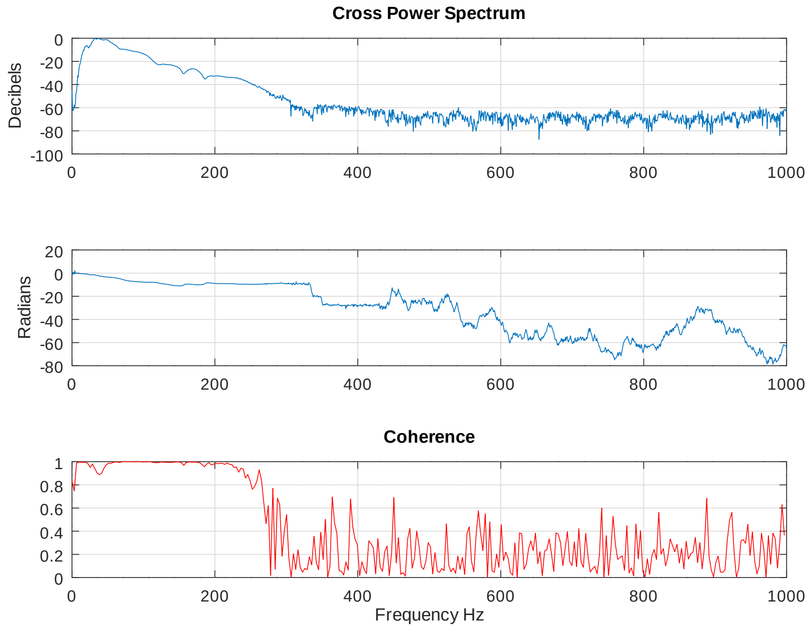


Figure 23: SASW: Cross spectrum amplitude and coherence reveal what range of frequencies provides useful dispersion information.

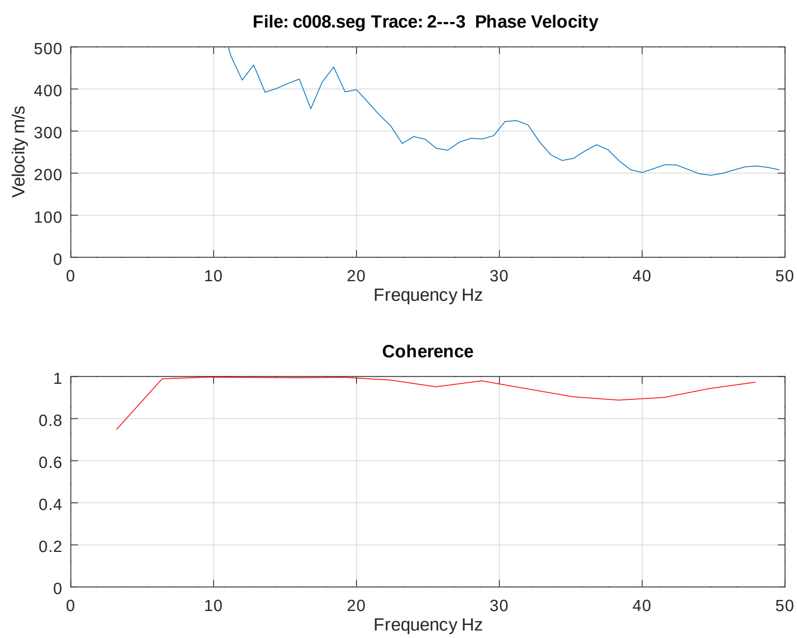


Figure 24: SASW: Dispersion computation over limited range of frequencies selected in the GUI.

8.2.3 OCTAVE saswv

This program was developed to read a text file with a measured cross power spectrum. It was used in a Benchmark Test sponsored by the Geo-Institute of the American Society of Civil Engineers (ASCE). The format of the text file is shown by the following first few lines of one instance:

```
dX = 32
R1= 45 R2 = 77 S = -7 T-Rex Shaker
  forward
f (Hz) |Gxy| (volts) Ph (Gxy) (deg) Coherence

5 0 -66.75 1
5.5 0.01 -74.25 1
6 0.01 -77.12 1
6.5 0.01 -79.08 1
7 0.01 -88.2 1
7.5 0.02 -103.03 1
8 0.02 -111.59 1
.
.
.

To run the program, start octave and type
     saswv

1). Enter the text file with the cross spectrum
2). Select a range of frequencies to plot
```

Details on the data set are found in Michaels (2014).

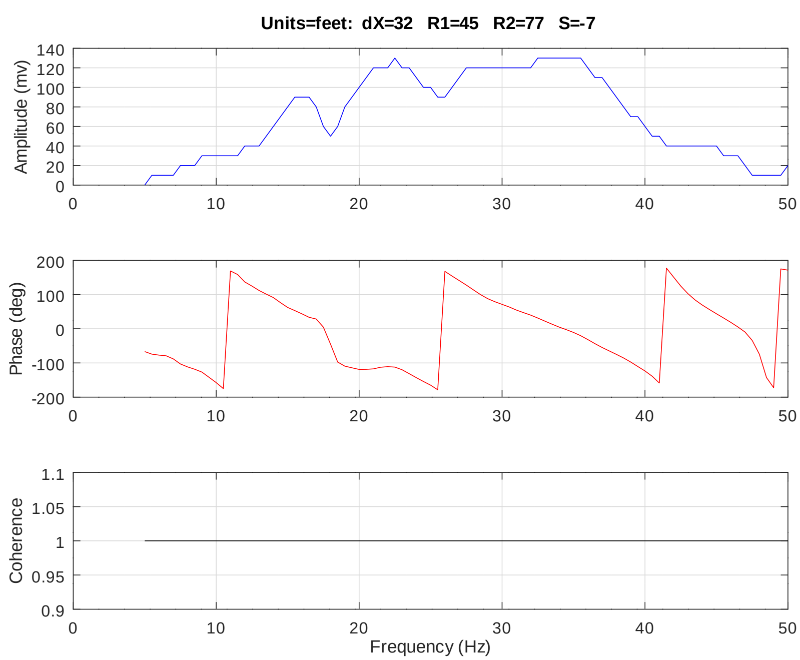


Figure 25: saswv: Cross power spectrum from data Michaels (2014).

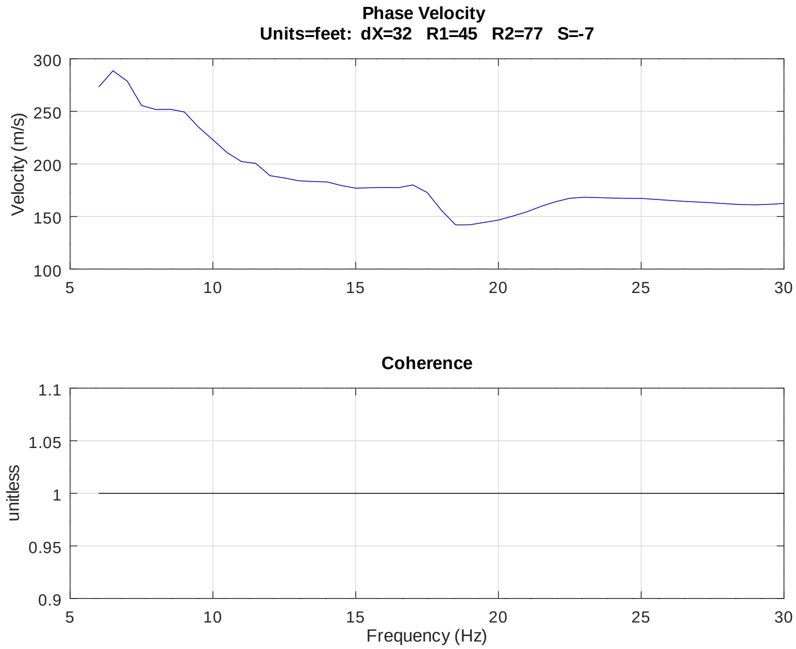


Figure 26: saswv: Dispersion computed from data Michaels (2014).

8.2.4 BWFI Wave Form Inversion Rayleigh Waves

This C-language code does a grid search inversion of Rayleigh Wave shot gather data. It provides a template for additional software development, and uses the objective function, **BOBF** 12.0.20 to serve as a metric for comparison of waveforms between a theoretical and field shot gather.

8.2.4.1 Grid Search The code **BWFI** loops through a grid of subsurface profiles. For each profile, the dispersion curve for up to 9 modes of Rayleigh waves is computed by the function **rwv_**, and then the waveforms computed by the **wavsub_** function. In order to compute a shot gather, one needs additional information. Temporal sample interval, source and receiver geometries are matched to the field data which are being inverted. The objective function, **BOBF** 12.0.20 measures the distance between each computed theoretical shot gather and the one field gather. The case where this distance is minimum is noted as the best case. The distance is based only on the waveforms, no computation of dispersion is used.

8.2.4.2 Subsurface Representation The subsurface profile is limited to only 2 paramters being varied in the grid search. The style of model is described in Michaels (2018). Figure 20 provides a sample of this type of representation. There are 3 control points.

- **Surface**: Depth fixed to zero, S-wave velocity fixed.
- **Moving Point**: Depth varies in a grid scan, S-wave velocity varies in a grid scan.
- **Half Space**: Depth fixed (input parameter), S-wave velocity fixed (input parameter)

Only the moving point (between the other two) is varied in the grid search. Between the control points the elastic parameters are linearly interpolated in a 1-D layered structure. The layer thickness is a fixed input parameter. The parameters varied are the S-wave velocity and depth of the control point. All other parameters, like mass density, layer thickness, P-wave velocity, etc. are interpolated. The Vp/Vs ratio is a fixed input parameter, so P-wave velocity follows the S-wave velocities. A slice of the solution space is provided in the output. In that slice, the two axes are S-wave velocity and depth of the middle control point.

The following are the input parameters for **BWFI**. The parameter infile are the field data to be inverted:

```
  bwfi  infile   Vo Vmn Vmx NV Zmn Zmx Nz
                Vhsp Zhsp VpVs Rho deltz segsw
                fmin fmax firvsel tmin tmax

    infile  = input field data (4char minimum)
                      FIXED
    Vo     = S-velocity at surface      (double)
                      SCAN
    Vmn    = Minimum S-velocity scan    (double)
    Vmx    = Maximum S-velocity scan    (double)
    NV     = Number of Vs grid steps     (int)
    Zmn    = Minimum Depth scan         (double)
    Zmx    = Maximum Depth scan         (double)
    NZ     = Number of depth grid steps  (int)
                      FIXED
    Vhsp   = Velocity of the half-space (double)
    Zhsp   = Depth top of the half-space (double)
    VpVs   = P to S wave velocity ratio (double)
    Rho    = Mass density               (double)
    deltz  = Depth integration step     (double)
    segsw  = Switch=1 output BSEGY files(double)
             Switch=0 No Output BSEGY   (double)
 Note: Octave *.m files always output
    fmin   = minimum frequency Hz (example 1.0)
    fmax   = maximum frequency Hz (example 100.0
 NOTE: if fmin=1., fmax=100. expect minimum phase
 wavelet with 3db bandwidth 4-60 Hz
 and no amplitude < 1.0 Hz or > 100.0 Hz
    firvsel = select component geophone
          0. = vertical
          1. = horizontal
    tmin = start time for time window in seconds
    tmax = end time for time window in seconds
    (trace samples outside window ignored
     when computing inner products and angles)
```

8.2.4.3 Example Run of BWFI In this example, the synthetic shot gather is computed for a layer over a half-space. The program **bwfi** does a grid search using the following command:

```
bwfi data.seg 200. 100. 400. 12  1. 5. 16 400. 5.01 2.5 1900 .1 1 1.0 80. \
0 0. 0.5
```

The grid consists of 221 points. The points are shown in Figure 27 below. The soil profiles corresponding to the search points are defined by linear interpolation of the elastic moduli between the two blue points. So there is a velocity reversal in the near surface for some cases (points to the left of the surface blue point). The interpolation is non-linear in velocity, but linear in terms of elastic moduli. Figure 28 shows the best fit of the grid search. "Best Fit" is defined in terms of the objective function (see **BOBF** 12.0.20).

We note that the two soil profiles differ in detail, but result in shot gathers that are nearly the same. This illustrates non-uniqueness which often is present in geophysical inversions. A lot depends on the context for doing geophysics. If this were a Vs30 problem, then the solution and the true profile are a match.

Figure 29 shows a contour plot of the objective function for this example. We can see that velocity resolution is better than depth resolution of the soil profile by the blue elongated minima region.

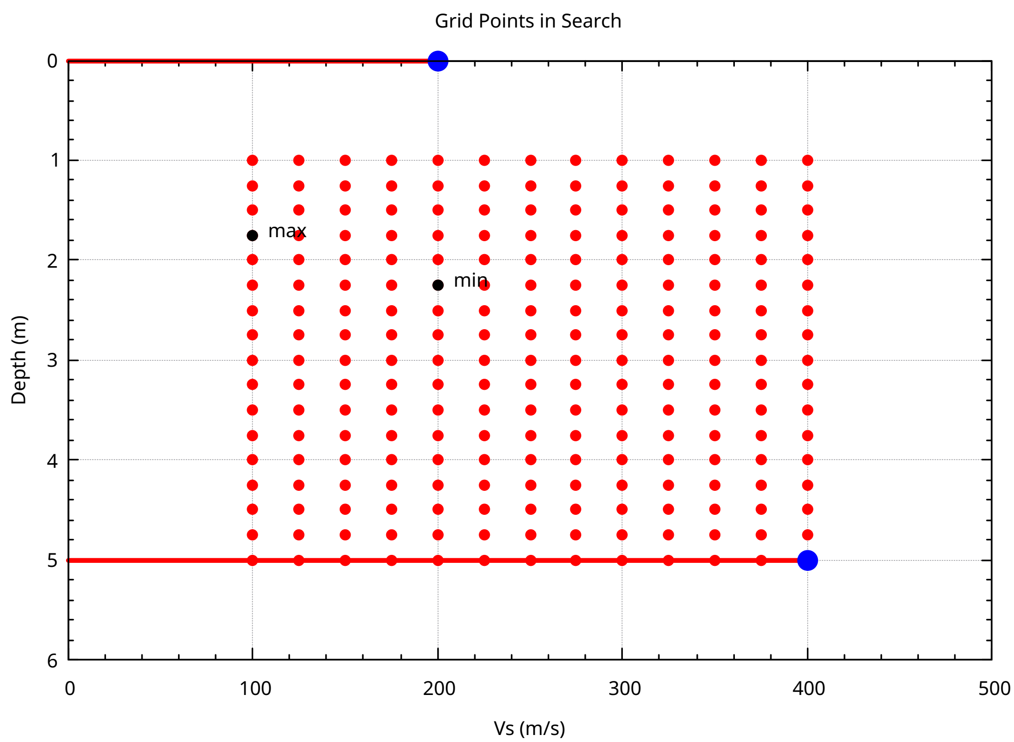


Figure 27: Grid points searched between fixed surface and half-space points in blue.

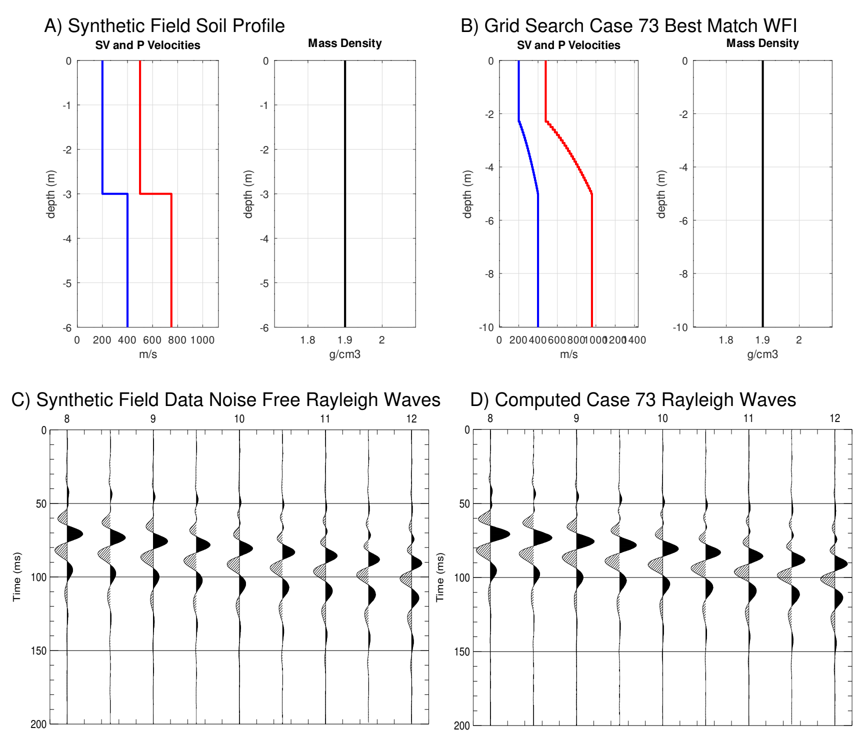


Figure 28: The soil profiles and computed shot gathers for the synthetic field data (A) and the best fit (B) (grid point labled min on figure 27).

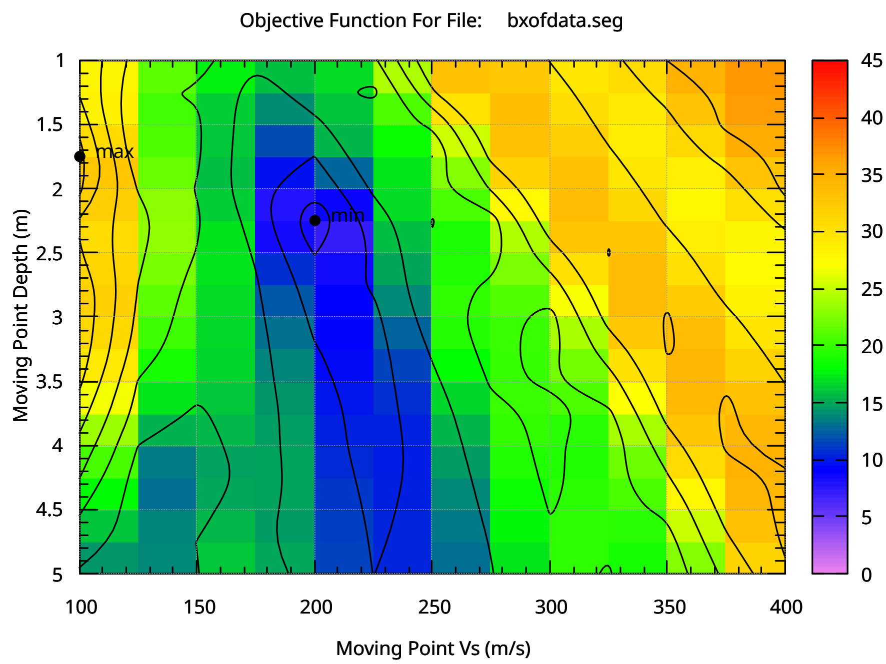


Figure 29: Contour plot of the objective function for the **BWFI** example run. The soil profile and waveforms of the objective function minimum are shown in Figure 28) (B) and (D).

8.2.4.4 Coding Details While **BWFI** is written in C-language, it calls functions which are Fortran subroutines. The Fortran subroutines are modified from main Fortran programs **DISPER** 9.2.6 and **WAVES** 9.2.8. The Fortran subroutines are located in the `bsu-3.0.3/src/sublibF4/` directory, and are called **rwv.f** and **wavsub.F90**. The subroutines are included in **libsubF4** which is accessed during the compilation of **BWFI**. The executable uses the shared library `libsubF4.so.0`.

The C-function prototypes are:

```
void rwv_(int *nlay, double rho[], double mu[], double lame[],
  double zi[], double *deltz, double *freq, int *maxmod, double pvf[]);

void wavsub_(int *nfreq, double Freq[], double pv[], double *fsamin2, double *firvsel,
  double sxyz[],int *nrec, double Rx[], double Ry[], double Rz[], int *nlay,
  double rho[], double mu[], double lame[], double zi[], int *Nxout,
  double xout[], double *segsw, int *IDcase);
```

Call statements in **BWF** are shown here. First is the dispersion computation, **rwv**.

```
rwv_(&nlay, rho, mu, lame, zi, &deltz, &freq, &maxmod, pvf);
```

where $nlay = 3$ in this case, and is actually the control points, not actual layers. **Rho** is mass density, **mu** is shear modululs, **lame** is Lame's constant, **zi** are the depths of the control points, **deltz** is the thickness of the interpolated layers, **freq** is the current frequency in the loop, $maxmod = 9$, and **pvf** is the array of Rayleigh wave velocities at the current frequency.

The other call statement actually generates the shot gather waveforms, **wavsub**.

```
wavsub_(&Nfreq,Freq,pv,&fsamin2,&firvsel,sxyz,&nrec,Rx,Ry,Rz,&nlay,
  rho,mu,lame,zi,&Nxout,xout,&segsw,&IDcase);
```

where **Nfreq** is the number of frequencies included in the simulation, **Freq** is the array of frequencies. The frequency interval depends on the length of the traces, and this trace length depends on the field data, but is designed for a power of two adjustment (see code variable **delf**). The array **pv** contains the Rayleigh wave velocities for all the modes found, **fsamin2** is the sample interval (matches field data), **firvsel** is the component of motion (either vertical or radial), **sxyz** is a 3 component array of source locaction in x,y, and z. The variable **nrec** is the number of geophone receivers (to match field record), and arrays **Rx, Ry, Rz** are the locactions of the geophones to match those of the field data. **nlay** is the number of control points (set to 3 currently), **rho** is mass density, **mu** is the shear modulus at each control point, **lame** are the Lame's constants for each control point, and **zi** are the depths of the 3 control points. All the waveforms are in an array, **xout**, the dimension of which is **Nxout**. The variable **segsw** is a switch controling output $1 = output\ the\ BSEGY\ files,\ 0 = don't\ save\ the\ BSEGY\ files$. The variable **IDcase** is the number of the case being processed in the call. The cases are those of a grid search.

8.3 Down Hole Seismic

Down-hole processing codes include:

- **BFIT** 8.3.1 Fit an interval velocity to vertical times
- **BVEL** 8.3.2 applies correctional velocity by static shift
- **OCTAVE VFITW** 8.3.3 fit interval velocities to vertical times of picked down-hole data.
- **OCTAVE VPLOT** 8.3.3.1 replot VFITW solutions, nice axes
- **BVSP** 8.3.4 fits 3-layer model to picked down-hole data
- **BVAS** 8.3.5 measures body wave dispersion SH-wave data
- **BAMP** 8.3.6 measures body wave decay SH-wave data
- **OCTAVE CAINV3** 8.3.7 inverts for stiffness and damping from **BVAS** and **BAMP** results.

8.3.1 BFIT

Vertical times correct for the source horizontal offset. If the vertical distance between the source and the geophone is Z, if the horizontal offset of the source from the bore hole is H, and if the straight line slant distance from source to geophone is S, then the cosine of the angle, θ, between the vertical and the slant is $cos(\theta) = \frac{Z}{S}$. The slant time is $T_s = \frac{S}{V_i}$ where V_i is the interval velocity. Typically we don't measure S, but do measure H. So the angle, $\theta = \arctan(\frac{H}{Z})$. The vertical time is then:

$$T_v = T_s \cdot cos(\theta) \quad = \frac{Z}{V_i} = \frac{S}{V_i} \cdot cos(\theta), \tag{3}$$

where T_s is the observed arrival time. Except in large horizontal offsets, the correction is modest. This program computes a straight line fit to vertical times, T_v. A similar program in OCTAVE is VFITW 8.3.3. The command line arguments are:

```
bfit  infile   emin   emax  labl

    infile  = input file name (4char minimum)

    emin    =minimum elevation for interval
    emax    =maximum elevation for interval
    labl    =2 character ID label for interval
```

Example for the X5 borehole:

```
bfit twave.seg 820.  840.  X5
```

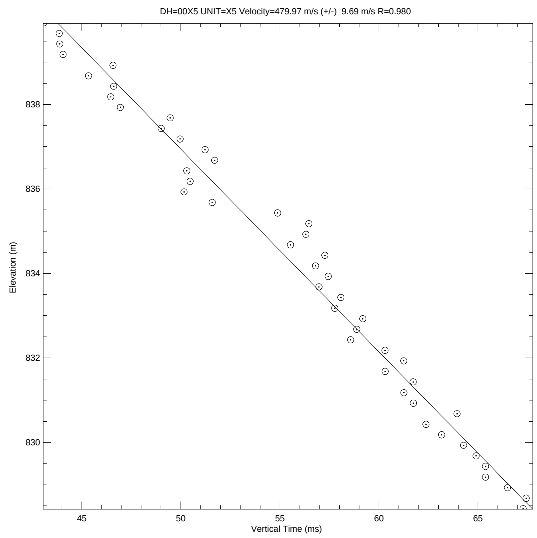


Figure 30: BFIT: Straight line fit yields interval velocity by least squares. Title has the value of the velocity, 479 ±10 m/s.

8.3.2 BVEL

This program can apply a correctional velocity to a down-hole data set. See **BRED** 7.0.1 for the same process on surface data. The command line arguments are:

```
  bvel infile  vel  ifast
 infile:   =input file name
 vel       =phase velocity to apply
 ifast     0=slow option (fft phase rotation
           1=fast option (sample shift)
Example:
bvel twave.seg 500. 1
```

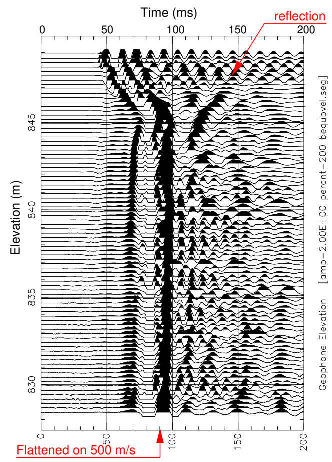


Figure 31: BVEL: Data flattened on 500 m/s (direct wave in bedrock). Overburden is slower (about 100 m/s). Reflection off top of bedrock shown.

8.3.3 OCTAVE VFITW

Given a BSEGY (*.seg) down-hole data set with first break pics in the headers (see **SEGPIC** 6.0.12), this program computes vertical times from the observed slant times. The user uses the mouse to select a start and ending depth, and the vertical velocity is computed. Then the mouse is used to place a label. After all the desired intervals are picked, the user may elect to replot the data using the **OCTAVE VPLOT** 8.3.3.1 program.

```
Start an octave session
  vfitw
  enter file name
  GUI: choose units         <feet | meters>
  GUI: choose vertical axis <depth | elevation>
  GUI: enter title
  GUI: save to disk ?       <yes | no>   (recommend yes here)
  GUI: do an interval?      <yes | no >
  loop here, use mouse to click start and end depth
  /|\           use mouse to place label with velocity (+/- m/s)
   |           GUI: do another interval? <yes | no >
   |--if yes if no---|
                    \|/
                    exit
```

Output Files:
filename.seg.vt and **filename.seg.vt2** (required for **VPLOT**).

8.3.3.1 OCTAVE VPLOT

The results of 8.3.3 are custom plotted as follows:

```
Start octave session
vplot
enter file name
GUI: Axes Limits  <xmin | xmax | ymin | ymax> (x=vertical time,
y=depth/elevation)
save plot to postscript
```

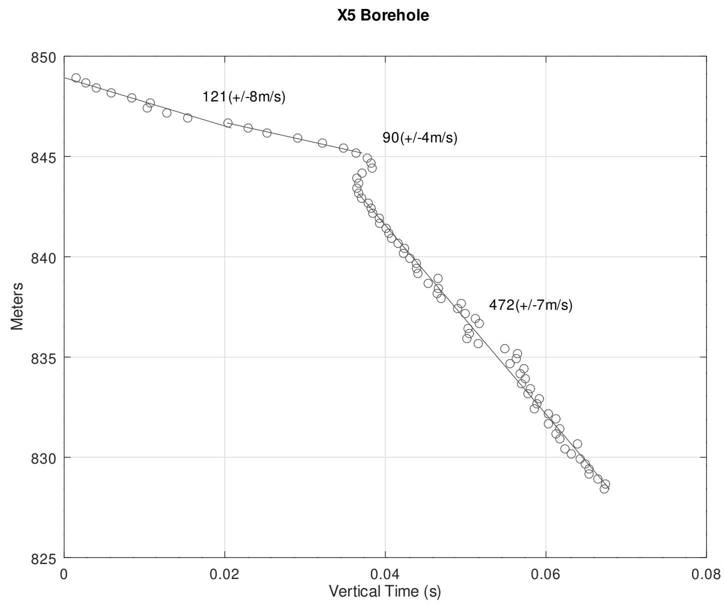


Figure 32: VFITW –>VPLOT: Plot of vertical time vs. elevation, and interval velocities. Axes and placement of velocity labels by mouse.

8.3.4 BVSP

This code reads a down-hole survey in BSEGY seismic format. It fits a 3 layer velocity model to the arrival times. The output *.lst file identifies the first arrivals as direct or refracted ray paths.

```
bvsp infile itmax zmin zmax
infile:   =input file name
itmax     =maximum number of iterations
zmin      =minimum depth to include
zmax      =maximum depth to include
```

EXAMPLE OUTPUT SAMPLE FROM *.lst

Iteration	LSQE	V1	V2	V3	Z1	Z2
0	0.02201	107.5	206.7	305.8	4.8	3.8
1	0.01985	108.6	221.8	316.6	4.7	3.9
2	0.01791	109.7	238.5	327.0	4.7	4.0
3	0.01616	110.6	253.2	337.0	4.6	4.1
4	0.01458	111.3	269.3	346.4	4.5	4.2
5	0.01202	112.0	285.5	355.4	4.5	2.0
6	0.01087	112.6	314.0	364.2	4.5	2.0
7	0.00984	113.1	345.0	372.6	4.5	2.0
8	0.00891	113.5	379.0	380.5	4.5	2.0
9	0.00807	113.9	416.9	388.0	4.5	2.0
10	0.00732	114.3	459.7	395.1	4.5	2.0

The values Z1 and Z2 are layer thicknesses. So for iteration 10, the first layer is 4.5 meters thick. The second layer is 2 meters thick, placing the top of the half space at 6.5 meters depth.

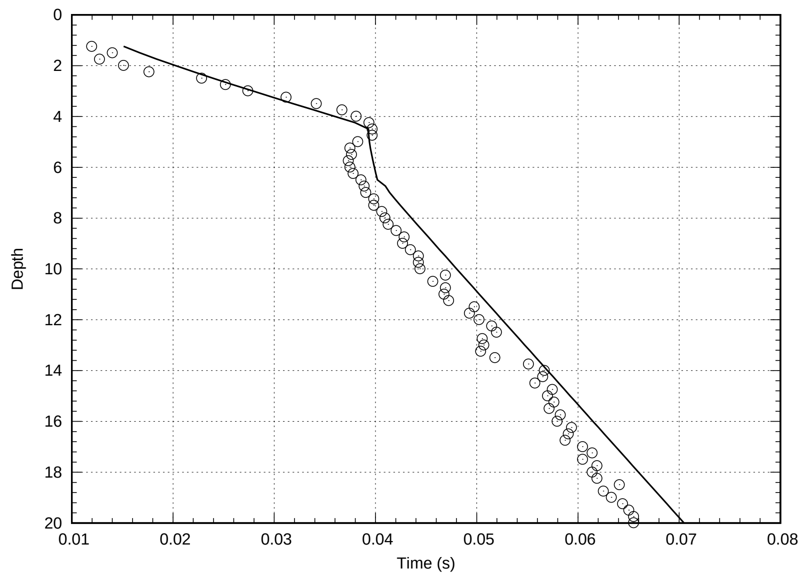


Figure 33: BVSP: Solution is a first layer, 4.5 meter thick Vs=114.3 m/s, second layer 2.0 meter thick, Vs=459.7 m/s, on top of a half-space with Vs=395.1 m/s.

8.3.5 BVAS

Programs BVAS and BAMP 8.3.6 assume a viscoelastic medium. The results are inverted under a Kelvin-Voigt (KV) constituative model with program **OCTAVE_CAINV3** 8.3.7 Michaels (1998). Inversion under a Kelvin-Voigt-Maxwell-Biot (KVMB) model may be used to estimate hydraulic conductivity if porosity is available using **OCTAVE_KD4kvmb** program (Michaels, 2006). SH wave enhanced down-hole data in BSEGY format (*.seg files) are processed for body wave dispersion. The **delf**, frequency increment should be no smaller than the reciprocal of the record length. For a 0.5 second recording, 2 Hz is the finest resolution.

```
bvas infile emin emax vmin vmax nvel fmin fmax delf bwd iskp ivscn

infile  =input file name
emin    =minimum receiver elevation (float)
emax    =maximum receiver elevation (float)
vmin    =minimum velocity
vmax    =maximum velocity
nvel    =number of velocity increments
fmin    =minimum frequency Hz
fmax    =maximum frequency Hz
delf    =frequency increment Hz
bwd     =filter bandwidth Hz
iskp    =skip filtering (if files already exist)
          1=YES  0=NO (-1=NO and delete when done)
ivscn   =output velocity scan data sets
          1=YES  0=NO
```

The output includes a file, **bvas.his** which can be processed by the inversion code, **cainv3** (section 8.3.7). The columns of the bvas.his file are also defined at the end of the *.lst file which contains details of the run. For example:

```
Frequency      Phase Vel.     +/- m/s       Semblance       Tbar       Tvar

   10.00         275.70     17.588079        0.5065       0.0342      0.0083
   12.00         319.28     13.666108        0.5476       0.0144      0.0059
```

14.00	458.14	9.981112	0.6169	0.0053	0.0019
16.00	612.12	6.796203	0.8824	0.0016	0.0007
18.00	554.49	6.774652	0.8730	0.0022	0.0008
20.00	481.10	7.960438	0.8251	0.0028	0.0014
22.00	466.08	11.081243	0.8044	0.0026	0.0021
24.00	461.89	12.072690	0.7758	0.0018	0.0022
26.00	454.63	4.133945	0.8045	0.0012	0.0007
28.00	475.22	3.958418	0.8036	0.0011	0.0007

Other outputs include a Postscript plot, bvas.ps, a QC plot, bvasqc.ps and a number of semplance plots.

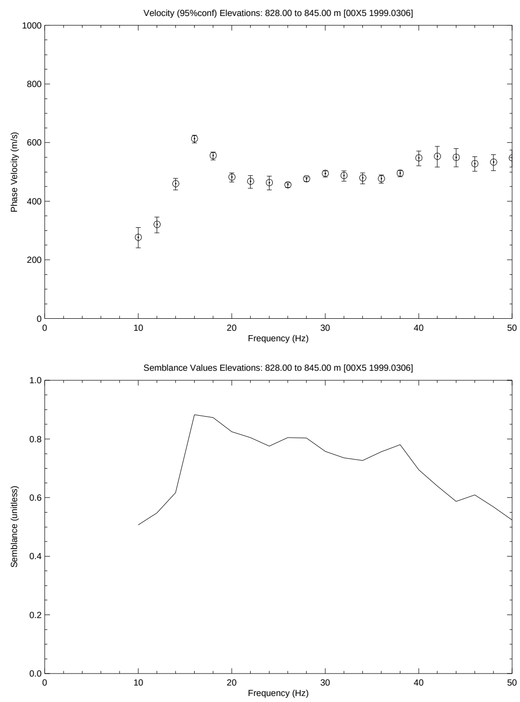


Figure 34: BVAS: SH body-wave dispersion and semblance results for down-hole data. These are the automated picks for maximum semblance as seen in Figure 35. Viscous, Kelvin-Voit behavior is an increase in velocity with frequency (Michaels, 1998).

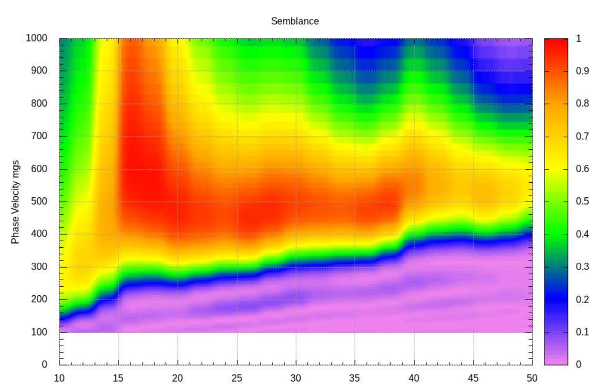


Figure 35: BVAS: SH body-wave semblance results for down-hole data.

8.3.6 BAMP

SH-wave enhanced down-hole data in BSEGY format (*.seg files) are processed for amplitude decay with frequency. Beam spreading is corrected for assuming spherical divergence. The frequency increment, **delf**, should be no smaller than the reciprocal of the record length. For a 0.5 second recording, 2 Hz is the finest resolution.

```
bamp infile emin emax fmin fmax delf bw tmax

infile  =input file name
emin    =scan gate: (min_elev)
emax    =scan gate: (max_elev)
fmin    =min band pass center frequency (Hz)
fmax    =max band pass center frequency (Hz)
delf    =frequency step (Hz)
bw      =bandwidth of filter (Hz)
tmax    =time gate: max_time
```

Output includes a file, **bamp.his** which can be used in procedure **cainv3** (see section 8.3.7). Also output are Postscript files, **bamp.ps** and **bampqc.ps**. The bamp.his file is 3 columns (frequency, decay 1/meters, standard deviation). For example:

```
 5.00     0.0000     0.0186
 7.00     0.0104     0.0121
 9.00     0.0287     0.0109
11.00     0.0576     0.0105
13.00     0.1077     0.0099
15.00     0.1251     0.0070
17.00     0.0771     0.0055
19.00     0.0574     0.0059
21.00     0.0703     0.0059
23.00     0.0840     0.0062
```

8.3.7 OCTAVE CAINV3

Down-hole SH-wave data are inverted for stiffness and damping with this Octave program. Required are the *.his file results from programs **BVAS** (section 8.3.5) and **BAMP** (section 8.3.6). Files **bvas.his** and **bamp.his** are required for each depth interval of interest. In order to compute uncertainty error bars, the depth interval should include as many subsurface stations as possible. Since **cainv3.m** is a joint inversion of body wave velocity dispersion and amplitude decay (corrected for beam divergence), the *.his files do not need to include exactly the same subsurface stations, as when there is a need to remove poor data from one or both *.his files. A companion program that computes the forward problem is **cafwd3.m** see section 9.1.1 .

The governing differential equation for this problem is a 3rd order PDE that is formulated as a 1-D plane wave problem (hence the need for the BAMP program to correct for beam divergence.

$$\frac{\partial^2 u}{\partial t^2} = C_1 \frac{\partial^2 u}{\partial x^2} + C_2 \frac{\partial^3 u}{\partial t \partial x^2} \tag{4}$$

where "u" is particle displacement, "t" is time, "x" is the coordinate in the direction of wave propagation, $C_1$is the stiffness $\left(\frac{m^2}{s^2}\right)$, and C_2 is the damping $\left(\frac{m^2}{s}\right)$. Equation (4) reduces to the elastic wave equation when the damping value, $C_2 = 0$. In that case, the phase velocity is constant for all frequencies, and the wave does not experience any decay (for a 1-D plane wave). In the elastic case, the phase velocity will be $\sqrt{C_1}$.

In the more general case, $C_2 \neq 0$, and there will be both velocity dispersion and exponential, inelastic amplitude decay. A solution of equation (4) is

$$u(x,t) = \exp(-\alpha x) \cdot \cos(\beta x - \omega t),$$

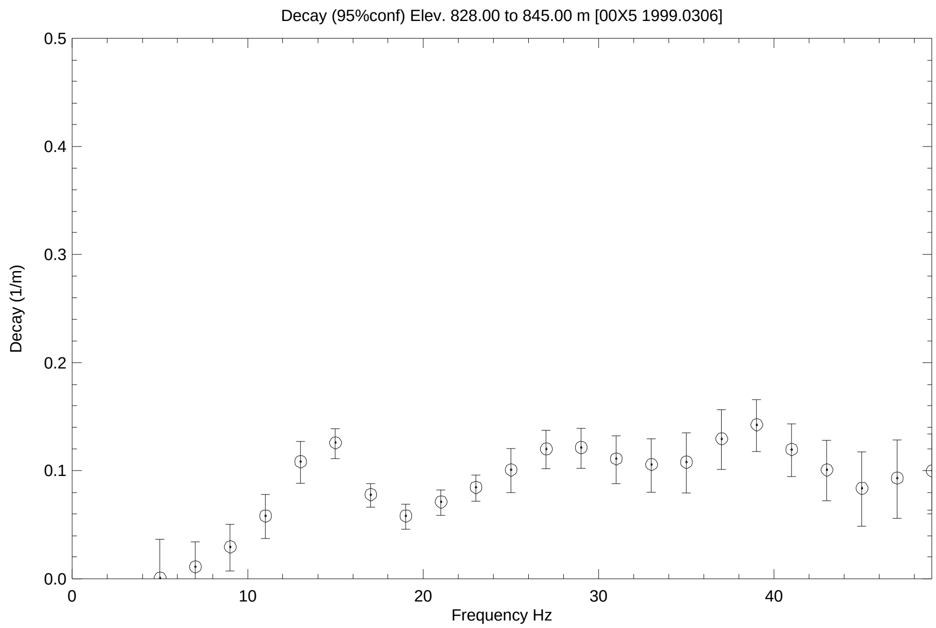


Figure 36: BAMP: SH body-wave amplitude decay for down-hole data same as seen in Figure 34 velocity dispersion. Corrected for beam spreading, a viscous, Kelvin-Voigt material, the decay should increase with frequency (Michaels, 1998).

where the wavenumber is complex and given by $\beta + i\alpha$. Michaels Michaels (1998) shows that the inelastic decay of a plane wave will be given by

$$\alpha = \frac{4\sqrt{D}\omega^2 C_2}{(2\omega C_2)^2 + D^2}$$

where ω is angular frequency (rad/s) and the quantity, D, is given by

$$D = 2\left(C_1 + \sqrt{C_1^2 + \omega^2 C_2^2}\right). \tag{5}$$

The phase velocity, c, varies with frequency according to the following relationship

$$c = \frac{2\omega^2 C_2}{D\alpha}. \tag{6}$$

The values for C_1 and C_2 can be expressed in terms of the following :

$$C_1 = \frac{(\beta^2 - \alpha^2)\,\omega^2}{(\beta^2 + \alpha^2)^2}, \tag{7}$$

and

$$C_2 = \frac{2\alpha\beta\omega}{(\beta^2 + \alpha^2)^2}. \tag{8}$$

Determination of C_1 and C_2 is by nonlinear joint inversion of the phase velocity, c, and inelastic decay, α , over a range of frequencies. The inversion is currently performed in the Octave procedure, *cainv3.m*. Initial estimates of stiffness and damping are obtained at the frequency corresponding to the largest α measured by *bamp*. First, C_1 is found by evaluation of equation (7). In that computation, $\beta = \frac{\omega}{c}$. Then, C_2 is estimated from equation (8).

```
RUNNING CAINV3:
Start an octave session, type cainv3
GUI, use mouse to pick min and max frequencies, click OK
and then use the mouse.  Horizontal position is all that
is read.  Focus one of the panels.
You can exclude some frequencies, and that will create
an fbx vector.  If you include all frequencies, you may get
an error statement (since it can't write out something that
does not exist).  Typically not a problem when you run
caplot3.m later.  Don't worry about it.

GUI, C1=stiffness, C2=damping initial estimate for the
3rd order wave equation.
GUI, Choose weighting
GUI, Choose balance between damping and velocity, .5 good idea
Plots, update as inversion progresses
GUI, continue LSQE plot
GUI, continue Chi squared plot
GUI, save results to disk, yes if you want to run caplot3.m
```

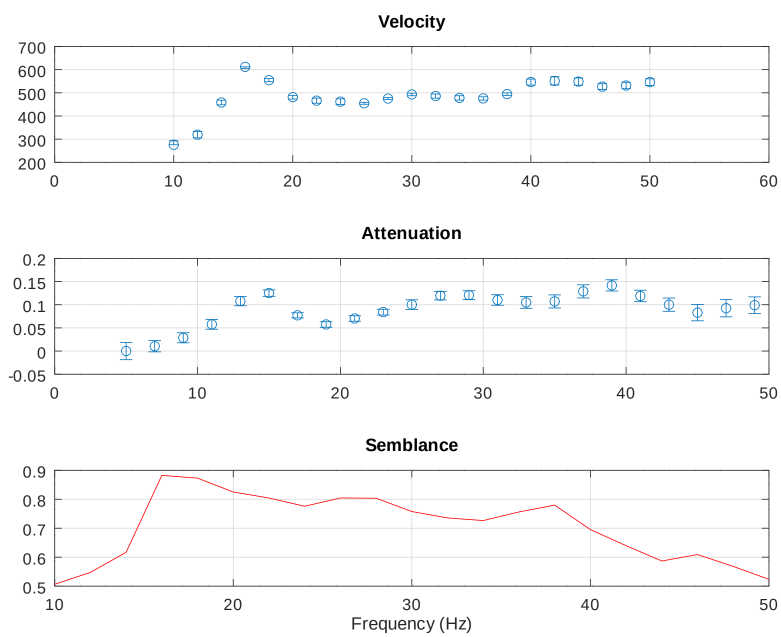


Figure 37: CAINV3: First display. Use mouse to pick frequency limits for analysis, low and then high.

After running **cainv3**, you may wish to make nice plots. For this, there is program **caplot3** (8.3.8).

8.3.8 OCTAVE CAPLOT3

Once an acceptable solution is determined for stiffness and damping by running **cainv3** (8.3.7), improved plotting of results are done with this program.

```
 Running caplot3:
Start an octave session, type caplot3
GUI, show grid?
GUI, enter limits to plot
GUI, only plot data used, or plot all data (recall mouse selection
of range of data to include above)
Plot, save or plot, shows fit and error bars with data observed
GUI, Go to Attenuation?
GUI, select axes limits
Plot, save or plot, shows fit and error bars with data observed
GUI, go to Chi^2 plot?
GUI, select axes limits
Plot shows change in Chi^2 by iteration

NOTE:
Plot Titles show C1, C2, Relaxation time
```

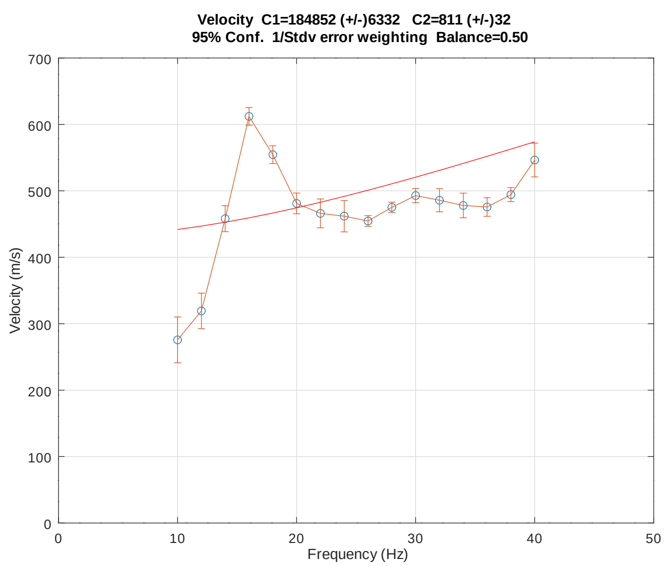


Figure 38: CAPLOT3: Plot of velocity dispersion, measure and calculated (solid line) only over frequency range used in cainv3 (8.3.7). Weighting by reciprocal of standard deviations.

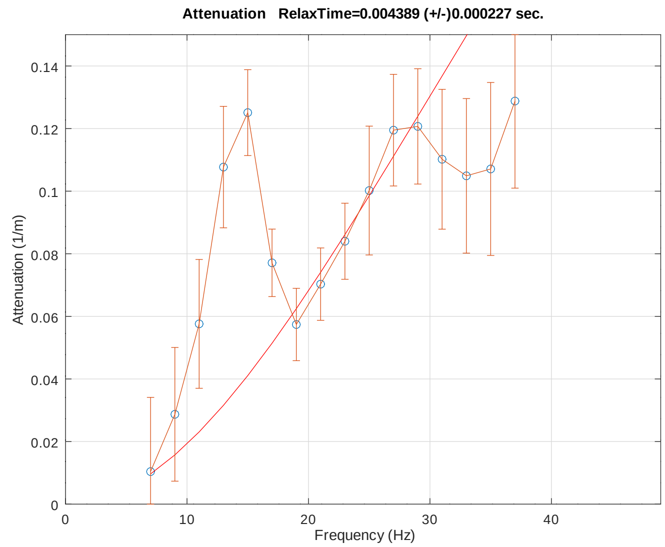


Figure 39: CAPLOT3: Plot of decay, measure and calculated (solid line) only over frequency range used in cainv3 (8.3.7). Weighting by reciprocal of standard deviations. Relaxation time about 4 msec. Relaxation time is $T_r = \frac{C_2}{C_1}$.

8.4 Relating Permeability to Damping KVMB

If porosity information is available, it can be combined with stiffness and damping results (viscoelastic, KV) under an alternative constitutive model (KVMB) to estimate permeablility. This would not be absolute, but rather relative permeablility (hydraulic conductivity, units of meters/second). The theory is found in Michaels (2006).

While the constitutive model is structured on highly simplified assumptions, it captures the behavior of granular soils saturated with water or other fluids when shaken by S-waves. Inertial damping resulting from shaking is predicted to peak at some hydraulic conductivity. Damping decreases on one side of the peak due to pore sizes being too small to permit significant relative motion between the frame and fluids. On the other side of the peak, damping decreases because the pores are so large that fluid moves easily with respect to the frame.

There are four octave programs provided with BSU that may be used with the KVMB soil model. Note, the intention is that this model is only valid in the context of **granular soils** under the assumption of **inertial damping** and **laminar flow**. The first three are forward problems, the 4th listed below is an inversion program.

- **OCTAVE_kdKVMBscan.m** computes and plots KV damping ratio as a function of either hydraulic conductivity or uniform pore diameter (user option). User provides porosity and frequency of shaking. See Figure 40.

- **OCTAVE_fqKVMBscan.m** computes and plots KV damping ratio as a function of frequency for a user provided porosity and hydraulic conductivity. See Figure 41.

- **OCTAVE_kvKVMBscan.m** computes and plots KV (Kelvin-Voigt) damping ratio vs. KVMB (Kelvin-Voigt-Maxwell-Biot) damping ratio. See Michaels (2006).

- **OCTAVE_KD4kvmb.m** 8.4.1 Inversion code that combines **frequency**, **porosity**, **stiffness**, **damping** to compute KV **damping ratios** and KD **hydraulic conductivity**. See the procedure 8.4.1.1.

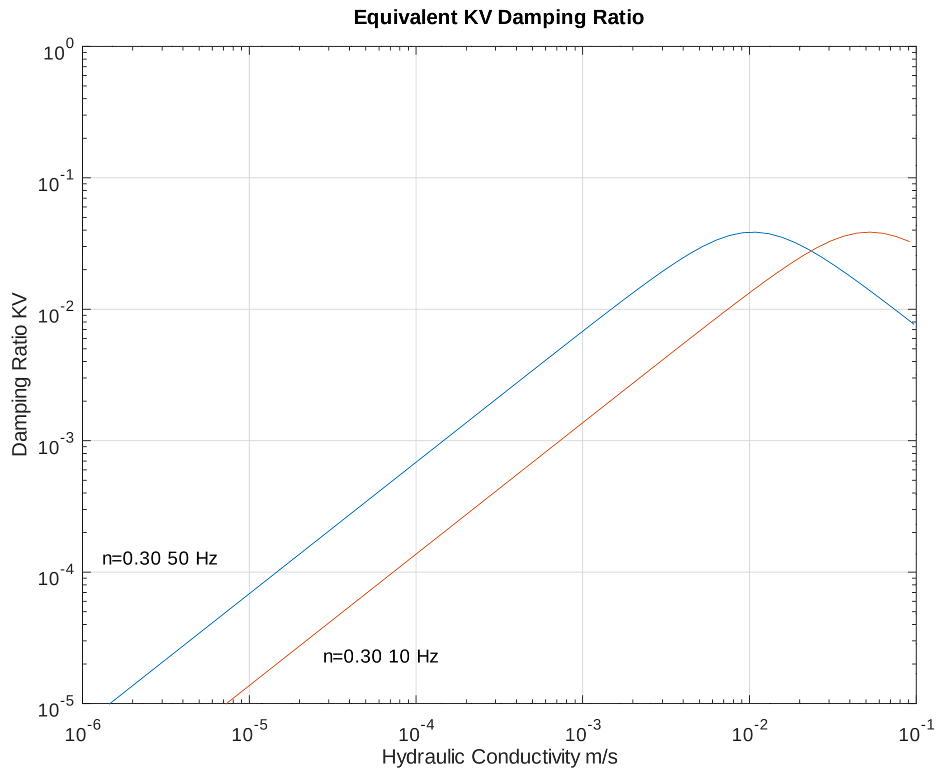


Figure 40: kdKVMBscan.m: Plots Kelvin-Voigt damping ratio vs. hydraulic conductivity for user provided porosity and frquency of shaking. Here, porosity is 30% and frequencies are 10 and 50 Hz. Left of the peak is coupled motion (small pores, fluid largely moves with frame). Right of the peak is uncoupled motion (large pores).

8.4.1 OCTAVE_KD4kvmb

This Octave program combines the **C1, stiffness** and **C2, damping** wave equation coefficients (Equation 4) determined from body wave dispersion inversion (CAINV3, 8.3.7) with **porosity** and a desired frequency to return solutions for the Kelvin-Voigt (KV) **damping ratio**. There will be up to two possible solutions. The likely solution is the coupled solution, but the program returns the uncoupled solution as well. The coupled solution is more likely since most earth materials are not permeable enough to result in the uncoupled solution.

8.4.1.1 Hydraulic Conductivity Procedure

The general procedure is as follows:

1. Drill a bore hole.
2. Do a down hole survey with a source that generates SH-waves.
3. Invert body wave dispersion and decay with CAINV3 8.3.7. This will provide **stiffness**, C1, and **damping**, C2 values under a Kelvin-Voigt model (Equation 4).
4. Select a relevant **frequency** and **porosity** and compute Kelvin-Voigt **damping ratios** using KD4kvmb 8.4.1. The coupled damping ratio is the most likely one. The program will also return the corresponding hydraulic conductivities, KD in m/s.

   Note this solution should agree with figure like Figure 40 when that figure is computed for the same relevant frequency.

Why use the Kelvin-Voigt (KV) constitutive representation for a soil? The problem with the KV model is that frame and fluid masses are lumped together as one. The KVMB representation frees the two masses to move which leads to an estimate of permeability. Engineering practice has been to use the KV representation, as in

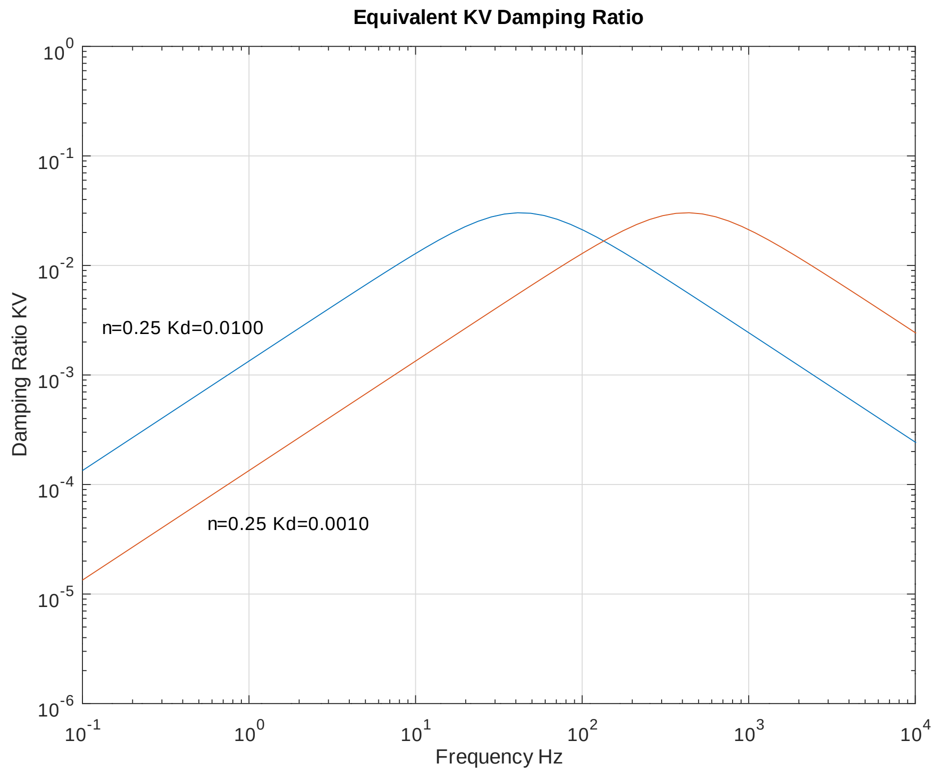


Figure 41: fqKVMBscan.m: Plots Kelvin-Voigt damping ratio vs. frequency fo user defined porosity and hydraulic conductivities. Here, porosity set at 0.25, two different cases of hydraulic conductivity $K_d = .01 \;\; K_d = .001 \;\; m/s$.

resonant column analysis. The CAINV3 joint inversion of wave dispersion and decay to values of stiffness and damping follows that same KV practice. However, the KVMB representation can also be used to map up to two (coupled and uncoupled) cases of equivalent KV damping. It is also possible that there will be only one result of KV damping if one is at the peak of the curve. Consider drawing a horizontal line to intersect a curve like those in Figure 40.

An example of running KD4kvmb is shown to illustrate the final step of the procedure 8.4.1.1. The results of the run shown in Figure 42 are:

```
SOLUTION (+/- 95 Percent Confidence)
Freq=12(Hz) Resonator_L=1.33(m)
Damping Ratios: Peak=0.030293 Wave=0.018850 (+/-0.01450)
Coupled (b_case): DR=0.018850 KD=0.01224(+/-0.0117m/s)
UnCoupled (a_case): DR=0.018849 KD=0.09169(+/-0.0881m/s)
Porosity: 0.250 (+/-0.038)
Relaxation Time Tr=C2/C1=0.50 msec
```

The notation is as follows:

- **Peak** damping ratio (DR) is the theoretical maximum for the case at hand.
- **Wave** damping ratio is the result of the CAINV3 joint inversion of a down-hole SH-wave survey.
- **KD** is the estimate of hydraulic conductivity (m/s). There are two of these unless the wave equals the peak damping ratio.
- Relaxation time is in milliseconds. It is analogous to the time it takes a sponge to recover its shape when squeezed under water and then released. The more permeable the sponge, the quicker the water can enter the sponges pores.

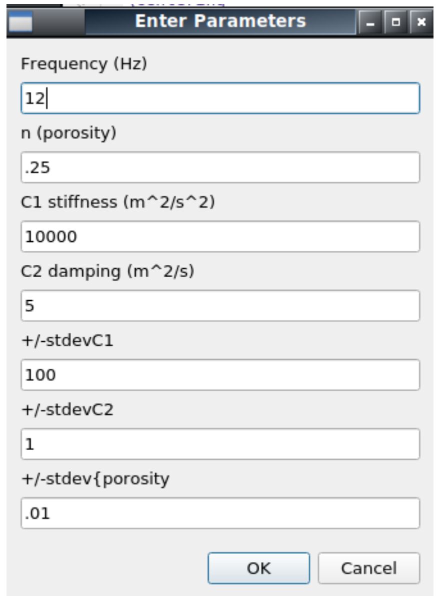


Figure 42: Prompt for input in KD4kvmb.m run

8.5 Refraction Shooting

There are a number of programs for refraction surveys.

- **BRED** (7.0.1) apply linear or hyperbolic correction in time to traces. This sometimes makes picking first breaks easier. It can also be used in flattening reflections for a totally different type of problem. See section 7.0.1 for more.
- **PICRESTORE** (8.5.1) Restore segpic.m picks on data reduced by BRED. 6.0.12
- **BPIC** (8.5.3) automatic first break picker, or inserts picks from a file (segpic.m 6.0.12 or suxpicker SU program).
- **BSHF** (12.1.1) static shift by header pics to QC picks.
- **BDAT** (8.5.5) datuming program for refraction data (adjusts data to the shot elevation using an overburden velocity)
- **BREF** (8.5.6) Direct wave (8.5.6.1), and Refraction (8.5.6.3) analysis setup.
- **OCTAVE DELAYTM** (8.5.6.3) delay time solution.
- **OCTAVE DELAYTMR** (8.5.6.4) reciprocal delay time solution (cross-river shooting).

8.5.0.1 BRED Example Flow Using BRED to flatten the refracted arrivals in time is not necessary, but can make it easier to pick first breaks. See **BVEL** 8.3.2 for the down-hole version. That is, it aids in identification of the refracted arrival (if the reduction velocity is approximately equal to the apparent horizontal velocity of the refracted arrival). It can also get the refracted arrival in a more confined time window, and this permits scaling the display for better resolution of the first arrivals. Here is an example flow. The situation is sandy soil over bedrock. The bedrock velocity is about 3000 m/s. See figure 17 for the original and reduced in time data plot.

```
bred k008.seg 1 3000.  .05
```

```
start octave
   segpic
     bredk008.seg (input file prompt answer)
     3 for clip, 0.15 for maximum time
     use mouse to pic first major down deflection
     (this produces output file bredk008.pic )
     exit

picrestore bredk008.dat bredk008.pic > out.pic

bpic k008.seg 1 out.pic 0.

mv bpick008.seg k008.seg

bshf k008.seg 0  1  .05
```

First, the data are reduced by a 3000 m/s velocity, bulk shift of .05 seconds to make picking easier. An octave session is started and the segpic program is run (requires segpic.m, segyinfo.m, bsegin.m files be in the directory. Exit octave, and note that file bredk008.pic contains the picks as two column text file (channel, pic time). Picks are corrected for the reduction velocity by running picrestore (see 8.5.1). The picks are then uploaded to file k008.seg by program BPIC 8.5.3. This produces file bpick008.seg which is then renamed k008.seg using the move command. A quality control (QC) check is done by using program BSHF 12.1.1, static shifting the data to align on .05 seconds using the header values uploaded during the BPIC step. Figure 43 shows the data aligned on .05 seconds (red arrow). Now file k008.seg is ready for further refraction analysis.

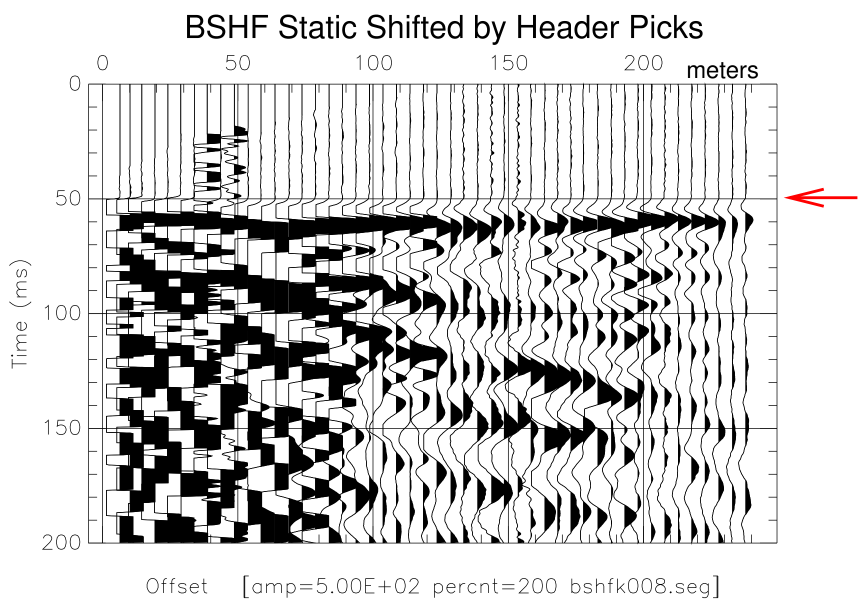


Figure 43: BSHF: After picks uploaded to headers with BPIC, data are static shifted to align on .05 seconds using header values. This is a quality control step. See example flow, section 8.5.0.1.

8.5.1 PICRESTORE

Only required if data have been reduced by a velocity with BRED (7.0.1). Since the data have been reduced, we need to adjust the pic file to original recording time. File **bredk008.dat** has recorded the static shift that were applied in the BRED process. This *.dat file is combined with the *.pic file in **picrestore** to restore the picks themselves to the original time. Program BPIC is then run with this corrected pic file (out.pic in the example, created by redirection).

```
usage: picrestore bredxxxx.dat bredxxxx.pic >out.pic
  FLOW
 1)bred flattens data, saved: bredxxxx.dat
 2)OCTAVE: segpic.m pics 1st breaks
 segpic.m  outputs bredxxxx.pic (tr,pic)
 3)picrestore stdout stream:
 adjusted pics, removing the flattening
 4)bpic xxxx.seg 1 out.pic
 (pics inserted into original file before bred)
```

8.5.2 BMRK

Picking times on seismic data can be done a number of ways, and program **BPIC** 8.5.3 is used to insert the pick times into the headers. This program will insert a spike (either positive or negative) at the pick time for quality control.

```
bmrk  infile  ipol  fscl

infile  =name input file to mark picks
ipol    =polarity of tick mark
      -1=negative
       1=positive
fscl    =scale factor to multiply tick mark size
```

Figure 44 shows an example of how it displays.

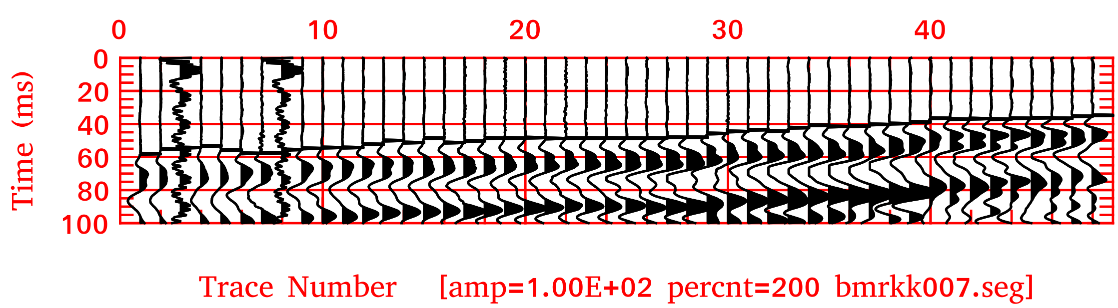


Figure 44: BMRK: Inserting a + spike to mark pick times.

8.5.3 BPIC

This program can either auto pick first breaks, or accept a text file with picks from another process like **segpic** 6.0.12. I recommend NOT using the autopicker (iswop1=0). It is usually better to manually pick data, particularly in noisy environments. The command line arguments to consider are below the dashed line, IF ISWOP1 = 1 or 2. See section 8.5.0.1 for an example flow where this program is used. The autopic command line arguments can be see by typing **bpic -h** in a terminal, or using the man page for BPIC (**man bref**).

```
-----IF ISWOP1 = 1 or 2-----PIC FILE----------------

bpic infile iswop1 picfil bulkst

infil
 iswop1:   option 0=autopic
                  1=UPload pic file to header
                  2=DOWNload headers to pic file
 picfil    =name of pic file
 bulkst    =bulk static shift to add to picks in sec
```

8.5.4 BSHF

This program can use static shifts to align data on either header pick values or values in a separate *.pic file.

```
bshf infile ipic ishf tshift picfil

 infile:   =input file name
 ipic:    1=static shift data by picks pick file
          0=static shift data by picks in headers
 ishf:    0=add static shifts
          1=subtract static shifts
 tshift:   =bulk static to add to picks
 picfil:  =file name with picks (for ipic=0)
```

For an example, see the flow in secion 8.5.0.1. See Figure 43 for a QC alignment check.

8.5.5 BDAT

Not all refraction data are shot with a source on the surface of the ground, and the ground is not always flat. BDAT can shift data by static shifts to datum the data to the shot elevation. This program is often used with buried explosive shots. The up-hole time can provide a value for the overburden velocity.

```
   bdat  infile vel iapply

     infile   =  input file name
     vel      =  velocity of overburden
     iapply   =  switch
            1=  apply static to datum
           -1=  remove static, restore to observed

NOTES: 1 Statics are computed from geometry in headers.
        The receiver is datumed to the shot elevation.

       2 Intended use is to make recognition of first
        arrivals easier, and picking easier.

       3 First break pick headers are adjusted with
        each datum change
```

BDAT uses a weathering zone approach, and computes the static shift using the header values:

- **sz** Shot elevation (top of hole if buried)
- **sd** Shot depth
- **rz** Geophone elevation

The static shift is computed as:

$$T_s = \frac{(sz - sd - rz)}{vw} \tag{9}$$

where *vw* is the weathering velocity.

8.5.6 BREF

BREF can build matricies for ground consistent inversion of either direct waves, or refracted head waves. The method is the delay time method for refractions, and the formulation is limited to a single refractor under overburden (Michaels, 1995). One typically decides on the offset where the transition from direct wave to refracted wave arrivals occurs (cross-over distance, Figure 16, **OCTAVE refplot**). The arguments are:

```
bref line# nshots rmin rmax irefdir irecip infil1 infil2 infil3

line#:  =line number for refraction/direct analysis
nshots  =number of shot records to use (<=10)
rmin    =minimum offset to include
rmax    =maximum offset to include
irefdir =0 refraction analysis
        =1 direct wave analysis
irecip  =0 normal shooting, shot fixed
        =1 reciprocal shooting, receiver fixed
infil1  =input file 1
infil2  =input file 2
infil3  =input file 3 etc....
```

Note that the argument, **irefdir** determines if the setup will be for direct wave or refracted wave. The following data examples are taken from Michaels (1999). Also, see the **bsu-user-guide3-1.pdf** for more.

8.5.6.1 Direct Wave In this example, we have 3 shot profiles, k004 is split spread. The minimum and maximum offset range for direct wave analysis is 0 to 30 meters. See Figure 16 to see how a cross-over distance is estimated. The command issued from a terminal is:

```
bref 008 3 0.  30.  1 0 k004.seg k008.seg k009.seg
```

The output files are:

- **G008** system matrix columns: (shot, receiver, offset)
- **D008** data vector, direct wave (arrival time,channel)
- **E008** elevation vector, (trace number, station number, elevation)

Analysis requires program **direct.m**. Start an Octave session and type **direct** to start the program. Answer the GUI questions. The result and a file **plot.ps** will be output (see Figure 45).

8.5.6.2 Theory The basic travel time equation for the direct wave between shot A and geophone 1 is

$$X_{a1} \cdot \frac{1}{V_1} = t_{a1} \tag{10}$$

where X_{a1} is the distance between the shot A and geophone 1. The overburden velocity is given by V_1 and the observed first arrival time is t_{a1}.

We set up a matrix problem in the form

$$G \cdot m = d \tag{11}$$

which expands to

$$\begin{bmatrix} X_{a1} \\ X_{a2} \\ X_{b8} \\ X_{b9} \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{V_1} \end{bmatrix} = \begin{bmatrix} t_{a1} \\ t_{a2} \\ t_{b8} \\ t_{b9} \end{bmatrix} \tag{12}$$

The ordinary least squares (OLS) solution is given by Menke (1989)

$$m = \left[G^T G\right]^{-1} G^T \cdot d \tag{13}$$

It follows that the overburden velocity determination is $V_1 = \frac{1}{m}$.

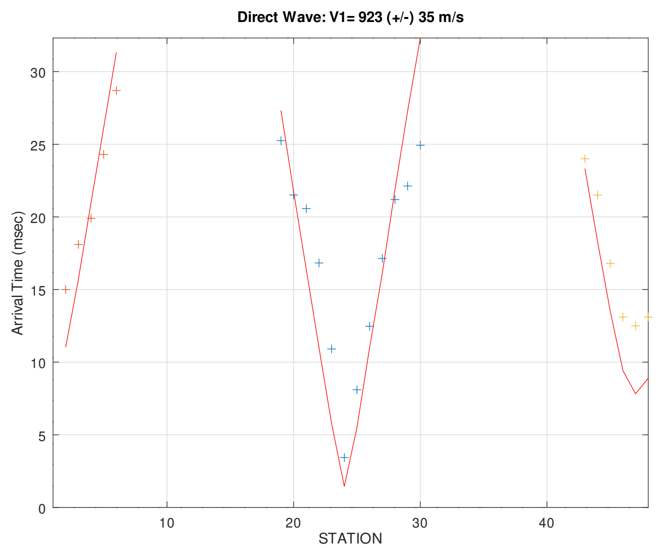


Figure 45: BREF: Output plot.ps for direct wave analysis. Title shows the least squares solution for the overburden velocity, $923 \pm 35 m/s$. Range of offsets 0 –> 30 m.

8.5.6.3 Normal Delay Time Refraction In normal shooting, we work with shot gathers rather than geophone gathers in the reciprocal approach 8.5.6.4. Shot profiles k008.seg and k009.seg are used in this example. Offsets from 50 to 250 meters are used. The BREF command is

```
bref 0809 2 50.  250.  0 0 k008.seg k009.seg
```

The output files are:

- **G0809** system matrix columns: (shot, receiver, offset)
- **D0809** data vector, refracted wave (arrival time,channel)
- **E0809** elevation vector, (trace number, station number, elevation)

An observant reader will ask, why not include the split spread, k004.seg? That would be better, and one would normally do that. However, by taking only two reverse profiles, I can show you how to add constraint equations when needed. With only the two lines, the system matrix, G0809, will be singular. The problem is a lack of reverse profiles in the near offset ranges. Receivers in the offsets 50 meters and beyond all receive signals from both sources at the end of the lines, so they are OK. To get a solution, we need to add a couple of extra lines at the bottom of the G0809 matrix.

Constraint Make the shot and nearest geophone for that shot have the same delay time.

The first two columns of the G matrix correspond to the shots, k008 and k009 (columns 1 and 2 respectively). The first shot, k008, has a near geophone in column 13. We create a new row at the bottom of the matrix by placing a 1 in column 1 (for the shot) and a -1 in column 13 (for the nearest geophone with a refraction). In the last column, we put a zero instead of a distance since this is a constraint equation, $T_{shot} - T_{geophone} = 0$. To get the zero, we add a row to the bottom of the data vector, D0809. We put two zeros in this last row (one for the time column, one for the station number). Again, this is a constraint equation, not a data equation.

We do this again, adding one more row to the G matrix. This time, a 1 in column 2 (for shot k009). The negative one (-1) goes in column 39 corresponding to the nearest geophone with a refraction for shot k009. We then edit D0809 data vector with a pair of zeros as above. Note: *We leave E0809 alone, no need to change it.*

The rest of the matrix above the constraint equations are simply delay time equations.

$$T_{shot} + T_{geophone} + \frac{X_{sg}}{V_2} = Tobs \quad , \tag{14}$$

where T_{shot} is the shot delay time, $T_{geophone}$ is the geophone delay time, X_{sg} is the distance on the surface of the ground between shot and geophone, and T_{obs} is the observed arrival time from the first break pick. The refractor velocity is given by V_2. Details on this approach are given in Michaels (1995). The solution is found by a weighted least squares (weighting minimizes the roughness of the solution, ie. makes it smoother at the slight cost of a poorer fit).

```
 RUNNING delaytm.m
 Start octave, type
        delaytm
GUI, change G001 to G0809, etc
GUI, number of shots = 2
GUI, smoothness weight 0.1
GUI, shows refractor velocity =4122 m/s and shot delay times of 10.3 and
12.4 msec, OK
Plot showing delay times for geophones
GUI, overburden velocity 923
Plot shows ground elevation and refractor indicating a variable soil
thickness. Alternative solution, default to 10 meters, Plot shows how
an alternative extreme solution of constant soil thickness with overburden
velocity varying.
GUI, 2 constraint equations, OK
Plot shows fit of solution to observed times.
Preference is for variable soil thickness solution based on geologic context.
```

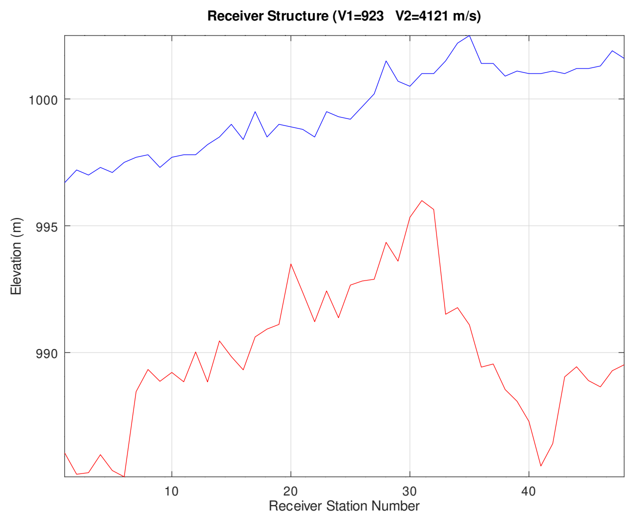


Figure 46: OCTAVE DELAYTM: Structure solution for shots k008 and k009. Ground surface in blue, top of bedrock in red. Soil velocity 923 m/s between blue and red. Bedrock velocity 4121 m/s.

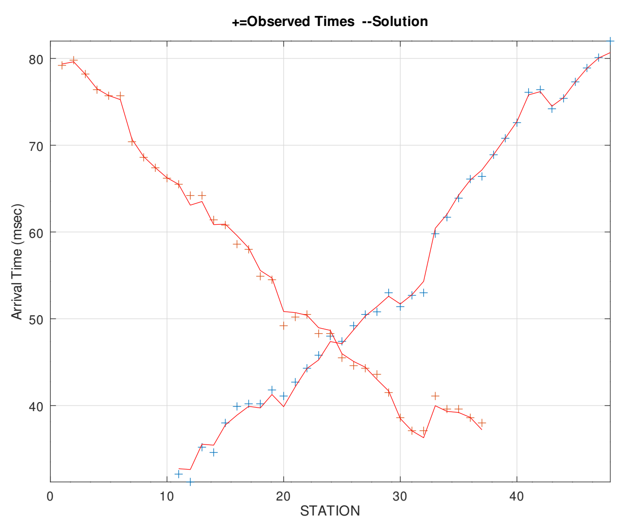


Figure 47: OCTAVE DELAYTM: Computed solution and observed times for k008 and k009.

8.5.6.4 Reciprocal Delaytime Refraction The difference between conventional refraction shooting 8.5.6.3 and reciprocal refraction shooting is that the former employs shot gathers and in this case, geophone gathers. The analysis is essentially the same. Reciprocal shooting is applied when crossing a river. Placing geophone in the river would subject the geophones to a noisy environment, particularly with a strong current bouncing the phones around. When an existing bridge needs replacement, one can deploy an airgun source from the bridge at stations of about 5 meters and record into geophone arrays on the river banks. An example of this approach is shown in Figure 48.

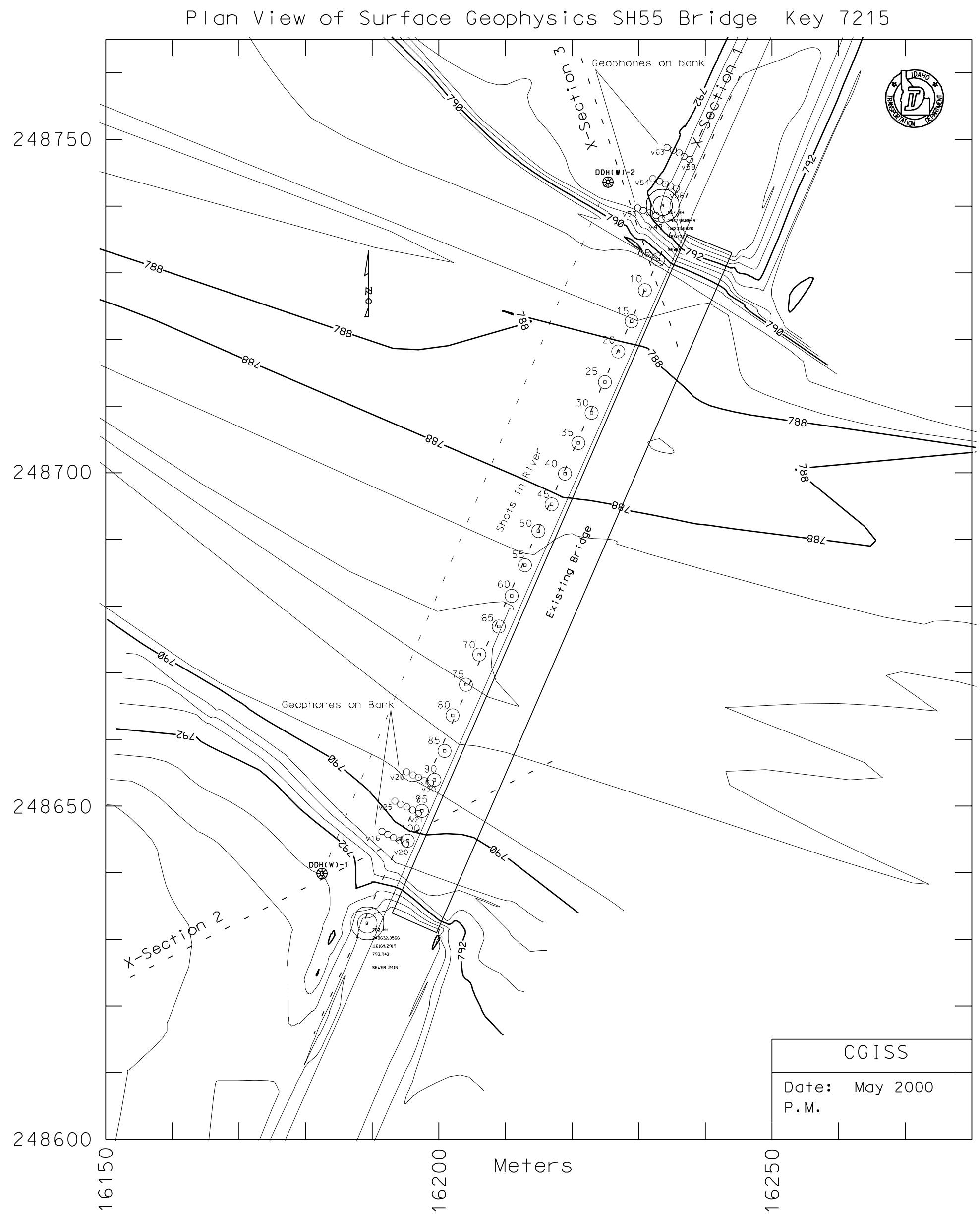


Figure 48: OCTAVE DELAYTMR: Reciprocal shooting across a river. Airgun source deployed at stations across bridge (Michaels, 2001a).

The geophone arrays can be summed to cancel traffic noise, beam forming to receive signals from the air guns. Recorded data are sorted into 3 northern geophone gathers, and 3 southern geophone gathers using the BEXT program (N000.seg, N001.seg, N002.seg, S001.seg, S002.seg, S003.seg). BREF is run with the following command:

```
bref 001 6 10.  110.  0 1 N000.seg N001.seg N002.seg S000.seg S001.seg S002.seg
```

The BREF program produces output files G001, D001, and E001 as in the normal shooting example (8.5.6.3). Some editing is required. The BREF code detected one constraint equation on this run. Further, a need for 2 more constraints was found so that all delaytimes were made equal at stations 90, 95, 100. These constraints were strongly weighted (factor of 50), and the G001 matrix was edited as shown below (relevant tail of the matrix):

```
0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0      45.363
0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0      40.364
0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0      35.382
0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0      30.329
0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0      20.336
0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0      15.075
0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0      10.564
0  0  0 50  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 -50 0  0       0.000
0  0  0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 -50 50  0      0.000
0  0  0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 -50 0   50     0.000
```

The first 6 columns are for the 6 geophone gathers. The D001 matrix tail is shown here:

```
0.03300     55
0.03120     60
0.02840     65
0.02660     70
0.02390     80
0.01880     85
0.01720     90
0.00000      0
0.00000      0
0.00000      0
```

One extra file is required when shooting this way. One needs a water depth file which gives the depth to the river bottom below the surface shots. This is a two column ASCII text file (depth, shot#). One starts an octave session and then runs delaytmR.m:

```
delaytmR
GUI enter names of the files G001, D001, E001 and wds.data
GUI number of geophones on shore, 6
GUI smoothness weight, 0.1 a good value
GUI displays refractor velocity (2216 m/s) and geophone delay
times below geophones (7.8, 9.9, 11.0, 2.5, 8.7, 8.6 ms), OK
PLOT shows delay times under the trans-river sources
GUI enter an overburden velocity in m/s, for example 1500 m/s
PLOT showing structure if overburden velocity is as assumed
(ground or water surface, bottom of river, refractor structure)
GUI enter alternative limiting case of constant depth refractor
PLOT of alternative, constant depth refractor, V1 varies
GUI enter number of constraint equations, 3 OK
PLOT of the observed data and fit to the equations.
```

The resulting structural plot is shown in Figure 49 and the arrival time fit in Figure 50. Delay times, T_{dt}, are related to the distance from the surface to the refractor, H by the critical angle, θ_c:

$$T_{dt} = \frac{Hcos(\theta_c)}{V_1} \tag{15}$$

In normal shooting, one solves for H using the critical angle $\theta_c = sin^{-1}(V_1/V_2)$, the refractor velocity, V_2, being the last unknown in the G matrix setup, and a result of obtaining a solution. This is straight forward for the delay times on the river banks, under the geophones. Under the shots, the water layer creates some additional complexity to the problem (see the code, delaytmR.m). In short, if the refractor velocity, V_2, is greater than water velocity, then it is simple ray optics. Velocity V_1 is the soil layer between the bottom of the river and the top of the refractor. To keep it simple in this example, I made $V_1 = 1500$, water velocity, and that would mean no ray bending from water to overburden soil. Recall that the ray parameter, p, is a constant, $p = sin(\theta_j)/V_j$, and to get a critical refraction at bedrock, we need to know V_2 relative to water velocity.

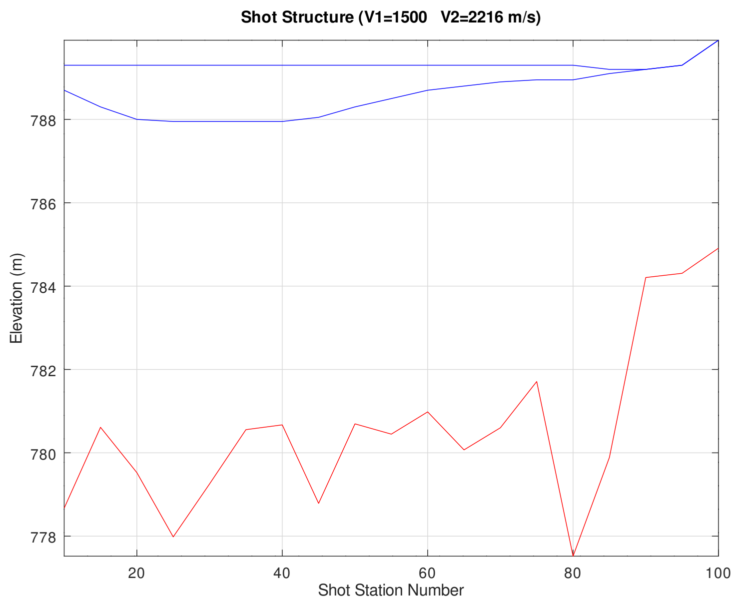


Figure 49: OCTAVE DELAYTMR: Structure assuming an overburden velocity of 1500 m/s. River water surface and bottom of river bottom in blue. Refractor structure in red.

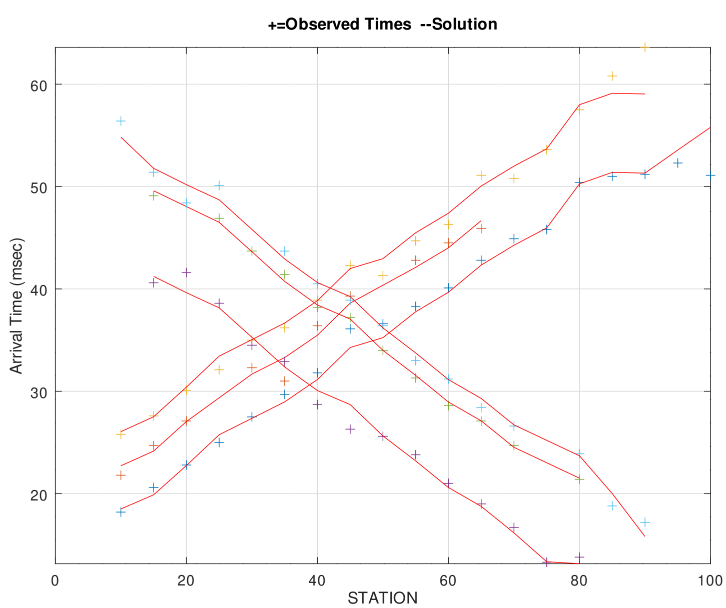


Figure 50: OCTAVE DELAYTMR: Observed arrival times and fit assuming an overburden velocity of 1500 m/s.

9 Forward Problem Codes

In the forward problem, a soil profile representation is used to compute a corresponding geophysical expression. The resulting geophysical data can then be compared to actual data. Another use is in planning geophysical surveys.

9.1 Down-Hole Seismic

Given an earth representation, compute the geophysical expression (ie. synthetic geophysical data).

9.1.1 OCTAVE cafwd3

This code uses the same formulation as in **cainv3** (8.3.7). It can be run with data to compare to, or as a simple stand alone.

```
 Start an octave session then type
        cafwd3

FORWARD PROBLEM
Computes S-wave velocity dispersion and damping by manual input for
C1=stiffness m^2/s^2 and C2=damping m^2/s

GUI, with or without data? (requires bvas.his and bamp.his if with data)
GUI, use mouse to pick range of frequencies, OK
Plot, use mouse to limit,
GUI, C1 and C2 estimate, OK
computes fit, then
GUI, make changes to stiffness and damping for next forward calculation
iterate
```

The code also computes "Q" quality factor as a function of frequency.

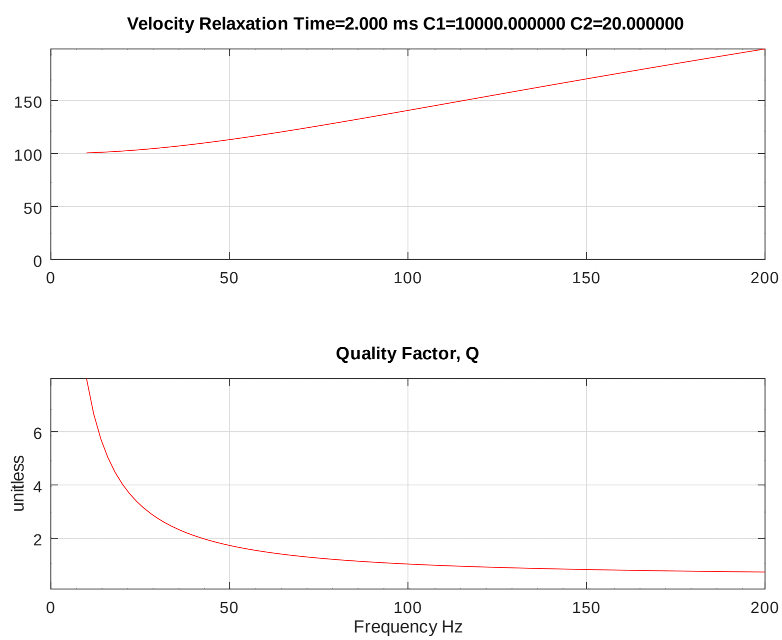


Figure 51: CAFWD3: Example without data, program's second plot showing quality factor, Q, The program's first figure plot expresses damping in terms of decay ($1/m$ units) as in Figures 36, 37, and 39.

9.2 Surface Waves

BSU codes include C-language, Fortran, and Octave procedures.

9.2.1 OCTAVE FwdR1

This code uses the same formulation as invR1.m program (see section 8.2.1). The function **rwv.f** is built from **DISPER** (9.2.6). It computes a single forward problem described in the same way as in the inverse problem.

```
 This program computes a demonstration Rayleigh wave dispersion
curve.  It uses rwv.f fortran function which must be compiled and
linked to expand octave capability.  The required files include:
rwv.f
wrapper.cpp
build_disper_oct (make executable, chmod +x if not)

Either run the build script first or start an octave session
and be prompted to build the extension inside the octave run.

A file, like model.txt, contains the layered earth velocity model.
Control points are interpolated by layers linearly interpolated
by elastic parameters, not velocity.

Control Point file, like model.txt, is as follows:
1. first line is number of layers
2. second line is S-wave velocities
3. third line is depth values where those velocities apply.
   |Example: for nlay=3 vi=shear velocity, zi=depth layer top   |
   | nlay                                                       |
   | v1 v2 v3                                                   |
   | z1 z2 z3
GUI prompts are available for relating P-wave velocity to S-wave velocity.
The layer thickness, once entered, will be held constant during
the run of all other model changes.

Also plotted are actual observed data (see line 275).  Running this program over
and over again, one can manually invert the observed data in
bvax.his (see program BVAX), by trying to fit the observed data
with a computed curve.

The disper() function returns a vector pv with fundamental
and any higher modes.  Here, it is demonstrated how to
select and plot fundamental mode.
```

9.2.2 LAMB

Program LAMB computes a solution to Lamb's problem. This solution includes surface and body waves that radiate from a vertical impact on a half-space medium. The code is specific to a single medium property where the P- to S-wave velocity is fixed, $V_p/V_s = \sqrt{3}$ (Lamb, 1904). For additional theory, see Mooney (1974). The command line arguments are:

```
  lamb  xmin dx np tmax fsamin vs den itype sfrq sdamp gfrq gdamp pol stab
 xmin = minimum geophone offset (m)
 dx    = spacing of geophones (m)
 np    = number of geophones
 tmax1 = maximum time for seismogram (s)
 fsamin= sample interval (seconds)
 vs    = shear wave velocity (m/s)
 den   = mass density (kg/m3)
 itype = type of traces output
      1= ground displacement, step function source
      2= ground particle velocity, step function source
         (or ground displacement, impulse source)
      3= ground displacement, source wavelet=damped resonator
      4= ground particle velocity, source wavelet=damped resonator
      5= geophone displacement with source wavelet
```

```
      6= geophone particle velocity with source wavelet (geophone voltage)
      7= source wavelet displacement (at source)
      8= source wavelet velocity (at source)
      9= source wavelet geophone displacement (at source)
     10= source wavelet geophone velocity (at source)
 sfrq  = source wavelet high-cut frequency (hz)
 sdamp = source damping (fraction of critical, example .7)
 gfrq  = geophone resonant frequency (hz)
 gdamp = geophone damping (fraction of critical, example .7)
 pol   = polarity switch
     -1= SEG Sign Convention (up motion = negative = trough)
      0= TEST MODE, for display of normalized solution, NO 1/R etc.
     +1= REVERSE SEG Sign Convention (up motion = positive = peak
 stab  = stability factor, for derivative, moves pole off unit circle
         (not generally needed except for itype=8 (try stab=.16)
          since most other outputs have enough low-pass filtering)
          See function deriv for more
```

The code computes both vertical and horizontal motion (files **lambv.seg** and **lambh.seg**). The **itype** parameter selects the type of output signal.

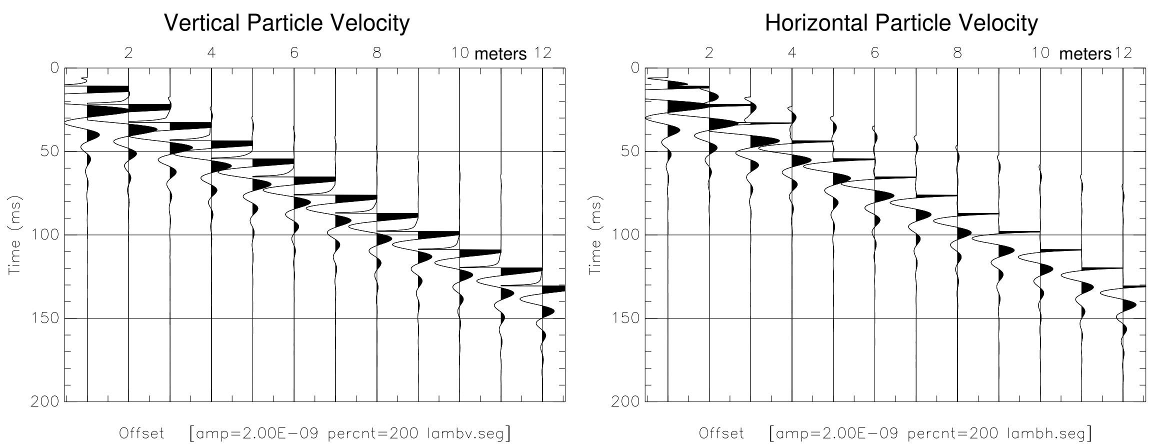


Figure 52: LAMB:**Ground** particle velocity solution for Lamb's problem, $itype = 4$.

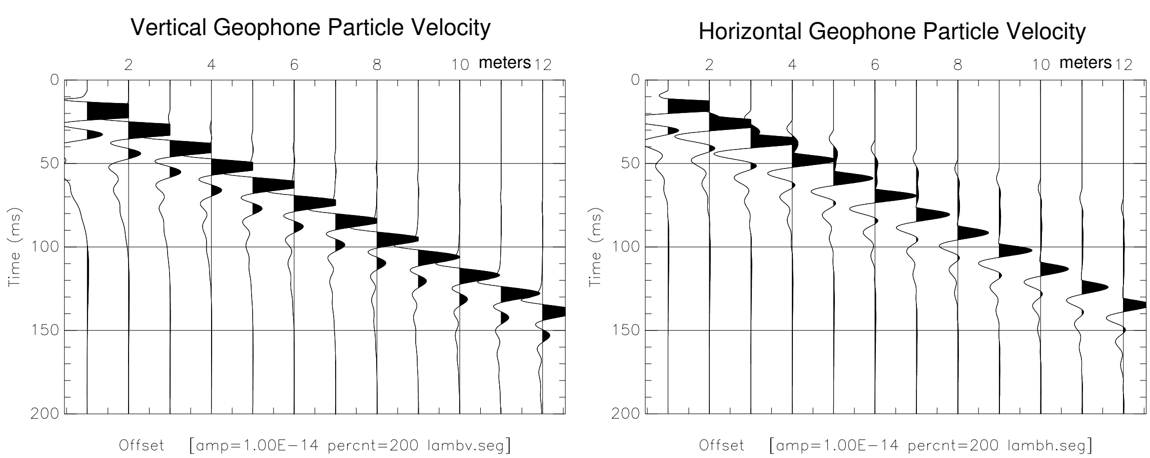


Figure 53: LAMB: **Geophone** (10 Hz, 0.7 damping) response, $itype = 6$.

The LAMB command corresponding to Figure 52 ($itype = 4$).

```
lamb 1 1.  12 .5 .0001 100.  1700.  4 70 .2 10.  .7 -1 .2
```

The above command computes the ground particle velocity for offsets 1 to 12 meters, .5 second record, sample interval of 0.1 msec, $V_s = 100m/s$, density 1700 kg/m^3.

LAMB was re-run to compute the particle velocity of the geophone element (which corresponds to geophone voltage) by changing itype to 6. This is shown in Figure 53.

```
lamb 1 1.  12 .5 .0001 100.  1700.  6 70 .2 10.  .7 -1 .2
```

9.2.3 Near Field BNFD

The BNFD program computes the near field seismic radiation for a homogeneous whole space (Aki & Richards, 1980) (Eq 4.23,Vol. I). While not likely to correspond to any field recording, it aids in understanding the transition from near to far field without the complexity of any boundaries being present. Command line parameters follow:

```
bnfd  infile xforce vp vs den alpha fc icomp ifield

    infile   =  input file name (sets geometry, tmax, ntraces)
    xforce   =  point force direction
            1=  in positive x-axis direction
            2=  in positive y-axis direction
            3=  in positive (down) z-axis direction

    vp       =  p-wave velocity
    vs       =  s-wave velocity
    den      =  mass density

    alpha    =  exponential decay factor (pos) for wavelet
    fc       =  center frequency of wavelet Hz
                (for example, try 50 for alpha and fc)

    icomp    =  component of motion to output
            1=  radial
            2=  transverse
            3=  vertical
    ifield   =  fields to include
                 (SPN) binary coded
            0=  wavelet only
            1=  near field only
            2=  far p-wave only
            4=  far s-wave only
            3=  near and far p-wave only
            5=  near and far s-wave only
            6=  far p- and  s-wave only
            7=  ALL: far S, far P, Near Field
```

Abbreviation SPN: S-wave, P-wave, and Near-field. Thus, (SPN)=(111)=7=all far S, far P, and N. For far-field P-wave only, (SPN)=(010)=2.

In the following example, all motions (ifield=7) are computed on the vertical and radial components. The headers are copied from an actual data set, c008.seg, to set number of samples, sample interval, and geometry. The commands are:

```
bnfd c008.seg 3 800.  200.  1800.  50 50 1 7
```

```
bnfd c008.seg 3 800.  200.  1800.  50 50 3 7
```

The first is for the radial (icomp=1) motion, the second is for the vertical (icomp=3) motion. Figure 54 shows motion in the vertical and radial directions. The template file, c008.seg, provides header data generating offset for each trace. While there are different definitions of near field in the literature, the bandwidth of the propagating wavelet (which sets the wavelet duration) and the difference between P- and S-wave velocity play the major role. The plots have been trace equalized using program BEQU to compensate for the dynamic variation in amplitude with offset.

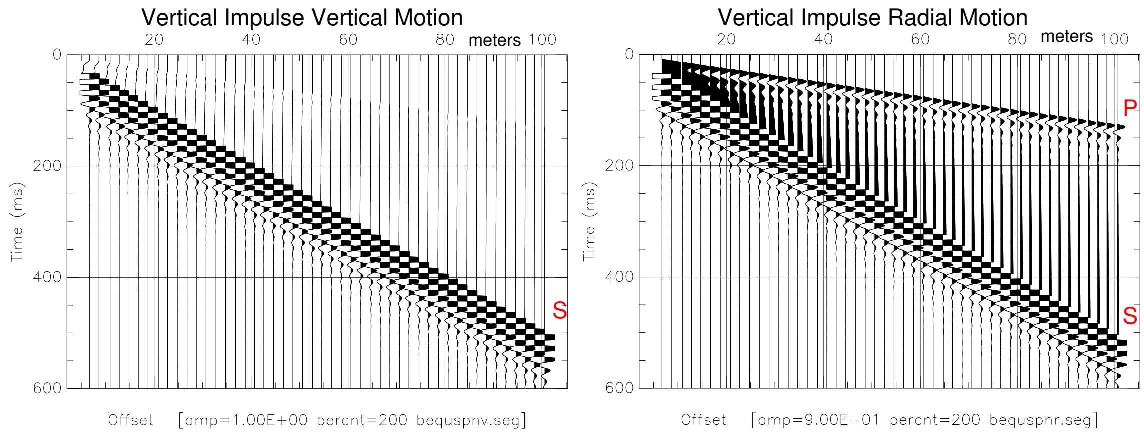


Figure 54: BNFD: Computing all fields (S-wave, P-wave, Near-field) The geometry is taken from a template file, c008.seg, and spans offsets from 7 to 100 meters. As offset increases, the far field P- and S-wave motion waxes as the near field wanes.

9.2.4 HALFSP

The Rayleigh wave solution for a half-space elastic medium is computed by HALFSP. The man page is read by typing `man halfsp` in a terminal. The output includes the velocity of the Rayleigh wave (stdout) and a file,**halfsp.tmp** which gives the motion-stress vectors for the frequency and range of depths required. An example log of this interactive program follows.**One types from a terminal**: `halfsp`.

```
ENTER RHO,VP,VS
```

```
1600,800,200
```

```
ENTER FREQ,NZ,ZO,ZEND
```

```
15,50,0,30
```

```
PHASE VEL=  190.2245
```

The units are kg/m^3 for RHO (density), m/s for P- and S-wave velocities (VP and VS), Hz for FREQ (frequency), and meters for the top and bottom of the interval computed (Z0 and ZEND). NZ is the number of depth points computed. The top of the halfsp.tmp file:

```
   HALFSP.F OUTPUT:
  RHO=0.1600E+04
  MU=0.6400E+08
  LAME=0.8960E+09
  FREQ=0.1500E+02
  P-WAVE VELOCITY=0.8000E+03
  S-WAVE VELOCITY=0.2000E+03
  RAYLEIGH WAVE PHASE VEL=  190.2245
 R1=Horiz. Displacement R2=Vertical Displacement
 R3=Horiz. Stress       R4=Vertical Stress


    DEPTH      R1          R2          R3          R4

      0.0   0.2241030E+00  -0.3974480E+00   0.3780026E+01  -0.0000000E+00
      0.6   0.1236450E+00  -0.4410695E+00   0.4977292E+07   0.2806469E+07
      1.2   0.5226342E-01  -0.4611813E+00   0.8269704E+07   0.4662912E+07
      1.8   0.2325356E-02  -0.4647821E+00   0.1033810E+08   0.5829188E+07
      2.4  -0.3185774E-01  -0.4570187E+00   0.1152439E+08   0.6498086E+07
      3.0  -0.5452060E-01  -0.4416587E+00   0.1208169E+08   0.6812320E+07
      3.6  -0.6881093E-01  -0.4214445E+00   0.1219681E+08   0.6877232E+07
      4.2  -0.7706446E-01  -0.3983572E+00   0.1200719E+08   0.6770310E+07
      4.8  -0.8101030E-01  -0.3738140E+00   0.1161344E+08   0.6548294E+07
      5.4  -0.8192512E-01  -0.3488157E+00   0.1108883E+08   0.6252494E+07
      6.0  -0.8074815E-01  -0.3240562E+00   0.1048634E+08   0.5912774E+07
```

To make a quick plot of the motion vectors, you can do something like this. Copy the halfsp.tmp file to a file like data.dat:

```
cp halfsp.tmp data.dat
```

Delete the first lines down to the first depth. So the top of the file becomes just columns of data:

```
0.0   0.2241030E+00  -0.3974480E+00   0.3780026E+01  -0.0000000E+00
0.6   0.1236450E+00  -0.4410695E+00   0.4977292E+07   0.2806469E+07
1.2   0.5226342E-01  -0.4611813E+00   0.8269704E+07   0.4662912E+07
1.8   0.2325356E-02  -0.4647821E+00   0.1033810E+08   0.5829188E+07
2.4  -0.3185774E-01  -0.4570187E+00   0.1152439E+08   0.6498086E+07
3.0  -0.5452060E-01  -0.4416587E+00   0.1208169E+08   0.6812320E+07
```

Then write a short **Gnuplot** script to plot the second and third columns as a function of the negative of the depth. Call it **plot.gp**:

```
set grid
set ylabel 'Depth (meters)'
p 'data.dat' u ($2):(-1)*($1) w l t 'horiz',\
'data.dat' u ($3):(-1)*($1) w l t 'vert'
set terminal pdf
set output 'plot.pdf'
replot
```

Run the Gnuplot program from a terminal command line to produce Figure 55:
`gnuplot -p plot.gp`

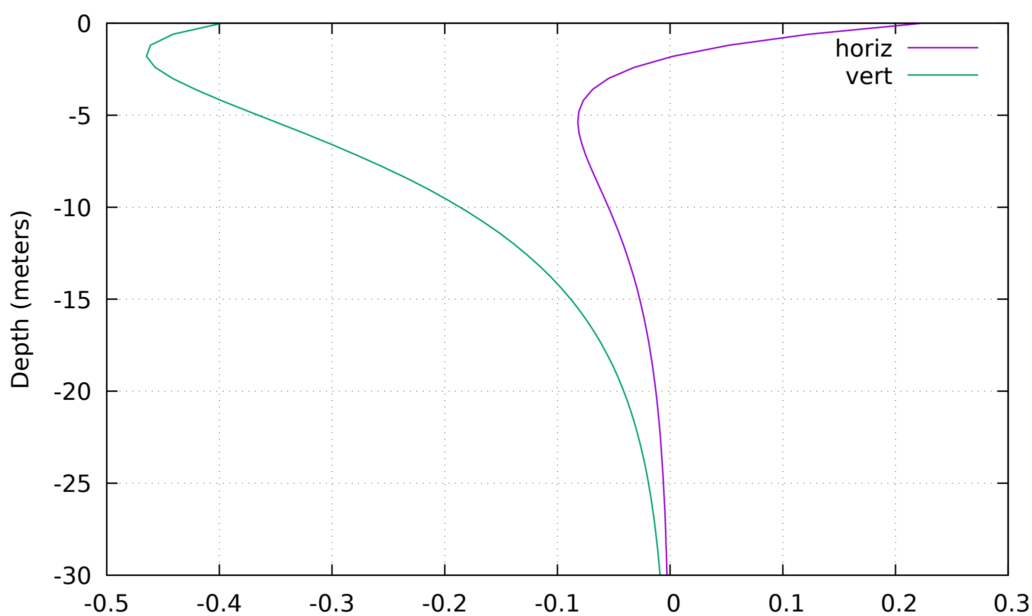


Figure 55: Gnuplot image created by the **plot.gp** script. The **-p** command line option of the gnuplot command makes the X11 plot persistent. Press the **q** key while mouse focus is in the figure to end the display. Then view the **plot.pdf** file with your favorite PDF viewer.

9.2.5 GENDIS

GENDIS is an interactive program used to prepare an input file for the **DISPER** program (9.2.6). **DISPER** is written in Fortran, and employs a namelist file for input. Use the man page for documentation (man gendis). There is no **-h** command line option since it is interactive. The following is an example log of a GENDIS run.

```
    enter: name of output file ( < 40 char)
disper.d
   enter: sample interval in seconds
.001
   enter: tmax for trace
1
   enter: minimum frequency
1
   enter: maximum frequency
100
   enter: maximum mode #
9
   enter: deltz step size
1.
   enter: number of control
3
  layer( 1) enter: beta,alpha,rho,depth(top)
100., 800., 1600., 0
  layer( 2) enter: beta,alpha,rho,depth(top)
100., 800., 1600., 1.0
  layer( 3) enter: beta,alpha,rho,depth(top)
300., 1500., 1700., 1.01
 twice npts= 2048
 twice tmax=    2.0480
  output====>disper.d
```

The output file, disper.d, in this case is:

```
  &disper
  nlay=    3,
rho= 0.1600E+04, 0.1600E+04, 0.1700E+04,
mu= 0.1600E+08, 0.1600E+08, 0.1530E+09,
lame= 0.9920E+09, 0.9920E+09, 0.3519E+10,
zi= 0.0000E+00, 0.1000E+01, 0.1010E+01,
  deltz=    1.0000,
  modemx=9,
  nfreq=202, flo=  0.1000000E+01, delf= 0.48828122E+00,   jsmax=300, ksw=0,
   pvlcty=0.0, pfreq=0.0, zend=100.0,
  ofile='disper.tmp',
  octav1='phase.m', octav2='mat2.m',
  curve='earth.crv', /
```

9.2.5.1 SHOWMDL

This program provides an easier human view of a disper.d file. Type **showmdl disper.d** to display the file named disper.d:

```
 show.tmp
      &disper
      nlay=    3,
    rho= 0.1600E+04, 0.1600E+04, 0.1700E+04,
    mu= 0.1600E+08, 0.1600E+08, 0.1530E+09,
    lame= 0.9920E+09, 0.9920E+09, 0.3519E+10,
    zi= 0.0000E+00, 0.1000E+01, 0.1010E+01,
      deltz=    1.0000,
      modemx=9,
      nfreq=202, flo=  0.1000000E+01, delf= 0.48828122E+00,   jsmax=300, ksw=0
       pvlcty=0.0, pfreq=0.0, zend=100.0,
      ofile='disper.tmp',
      octav1='phase.m', octav2='mat2.m',
      curve='earth.crv', /
        1        0.000      100.00      800.00   1600.0
        2        1.000      100.00      800.00   1600.0
        3        1.010      300.00     1500.00   1700.0
```

9.2.6 DISPER

After **gendis** (9.2.5) is run, a namelist file can be run to compute dispersion. The output includes a text file, **disper.tmp**, a data file capturing dispersion, **earth.crv**, and some Octave files, **model.m** and **phase.m**.

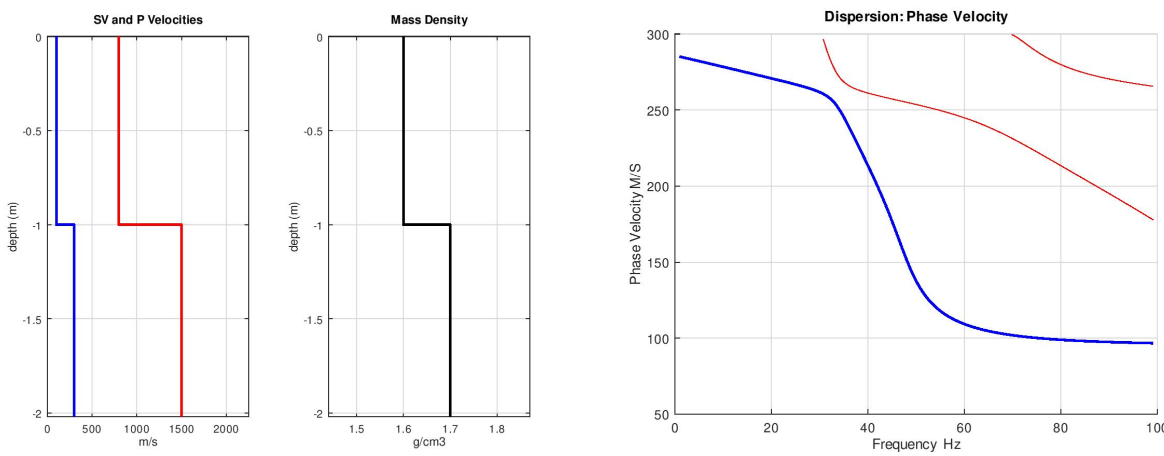


Figure 56: DISPER: The model and phase velocity plots. The model.m plot shows P-wave (red), S-wave (blue) velocity, and density (black). This is a layer over a half-space model. On the right is the phase.m generated plot showing the fundamental mode (blue) and two higher modes (red). The model (soil profile) was generated in gendis (9.2.5)

The computational function is the same as **rwv.f** used in the octave programs **FwdR1.m** 9.2.1 and **invR1** 8.2.1.

9.2.6.1 Motion-Stress from disper.d The file, **disper.d** can be edited and disper run in a different way, computing the motion-stress vector for a given frequency and mode. For example, from the file disper.tmp, note the phase velocity for a particular mode. Pick a frequency of interest, say 32.2265605 Hz. We scroll down disper.tmp:

```
31.2499981 |    259.6905012   291.4923132
31.7382793 |    258.6609733   286.9434926
32.2265605 |    257.4416940   282.8358906
32.7148417 |    255.9845241   279.2012189
33.2031230 |    254.2462253   276.0619743
```

We edit the **pvlcty=** line of disper.d to use the phase velocity. For the fundamental mode, the replacement is:

```
 <    pvlcty=0.0, pfreq=0.0, zend=100.0,
---
>    pvlcty=257.4416940, pfreq=32.2265605, zend=10.0,
```

We also change zend, choosing a more relevant depth of interest which depends on the frequency. Low frequencies extend deeper than high frequencies. We also change the computational depth interval, deltz. The replacement is:

```
 <   deltz=    1.0000,
---
>   deltz=    .0010,
```

We change the computational increment from 1 meter to .001 meters. Review the relevant lines of the disper.d file shown above in section 9.2.5 for example. Running disper with the edited **disper.d** file will replace disper.tmp with a new version listing the motion-stress vector computed at the new deltz interval. Figure 57 is the result.

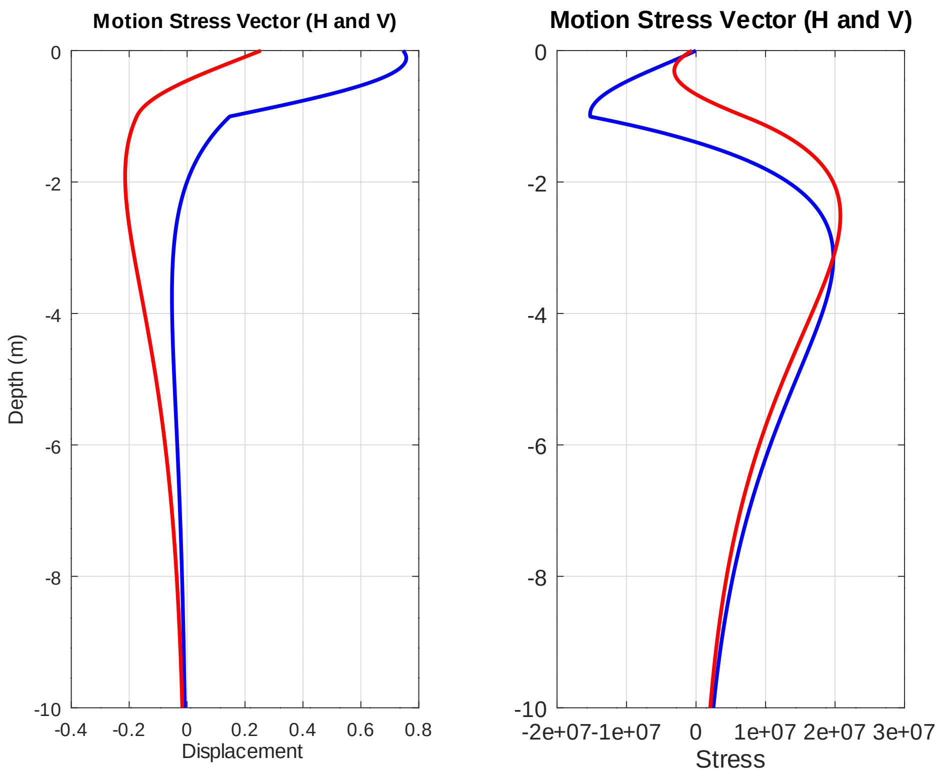


Figure 57: DISPER: Re-running **disper** to compute the motion stress vectors. See section 9.2.6.1 for how to edit **disper.d**. The file, mat2.m created this plot. Blue is horizontal, Red is vertical.

9.2.7 GENWAV

The output file **earth.crv** generated by DISPER (9.2.6) can be used as input to the program **WAVES** which will compute a synthetic Rayleigh wave only seismic data profile. This program, **genwav**, is used to interactively build a namelist file for program **WAVES** (see 9.2.8).
IMPORTANT: The tmax and sample interval must agree when running gendis and genwav so that frequency sampling matches. The following is a log of an example run of this interactive program.

```
 genwav
   enter name of namelist file (40 char)
   Example: waves.d
waves.d
   enter name of dispersion curve file
   (this is file from disper.f)
   Example: earth.crv
earth.crv
   enter near offset:  xnear
1
   enter group interval:  delx
1
   enter number of receivers:  nrec
24
   enter minimum group velocity expected
100
  RECOMMENDED minimum tmax=    0.4800
   enter: maximum trace time, tmax
1.0
   enter: sample interval (seconds), fsamin
.001
   enter frequencies: fmin, fmax
1., 100.
```

```
   enter maximum mode to include
9
   enter ksw switch 0=c plot, 1=k plot
0
   enter type of plot format, mapmat
   0=octave (Matlab)  1=scilab
0
   enter Output option 0=Vertical  1=Radial
0
   enter source depth
0
  enter (3) diagonal elements, moment tensor
0,0,1
  Padded Radix 2 tmax=    2.0480
  Number of points in signal= 2048
--------------------------------------------------
  .......Frequency interval=  0.48828122
  NOTE: Frequency Interval MUST MATCH DISPER OUTPUT
  WAVES will output signal length = 1.0/delf
  IF MISMATCHED: CHANGE sample rate in WAVES
                 or RERUN DISPER
--------------------------------------------------
  Number of frequencies=  204
  output in =====>waves.d
```

9.2.7.1 Frequency Increment Note that following the RECOMMENDED minimum tmax, the next two questions must agree with the **gendis** run, specifically enter **tmax** (the maximum recording time) and **fsamin** (the sample interval in seconds). The "Frequency interval" must be the same for both **disper** (which generates **earth.crv**) and the intended **waves** program run. If unsure, open the earth.crv file in your favorite editor, and compute the difference between consecutive frequencies (column 1). This frequency interval must match the one near the end of the genwav run (written to the terminal, see above for example). In this example:

```
0.14648436600000000102883178E+01 -0.97656244000000000315813509E+00 = 0.48828
```

9.2.7.2 Explanation of genwav parameters

- **xnear** near offset in meters
- **delx** spacing between channels in meters
- **nrec** number of receivers (ie. traces in shot gather)
- **minimum group velocity expected** used in estimating needed record length.
- **tmax** length of traces in seconds (need not match the recommended, depending on gendis run).
- **fsamin** sample interval in time, seconds
- **fmin, fmax** minimum and maximum frequencies. Recommend that these be wider than what you think you need, then filter back for your final result using a filter program (like BFIL). NOTE: Wavelet used is minimum phase, set by fmin and fmax.
- **maximum mode to include** Must be $modemax \leq 9$
- **ksw** dispersion plot, sets wavenumber (1=k) or velocity (0=c)
- **mapmat** Format for plots. Recommend Octave (Matlab)
- **output option, irvsel** signals will be vertical or inline radial.
- **source depth** depth of source in meters
- **moment tensor diagonal** (radial,transverse,vertical). 0,0,1 is a vertical impulse. You can edit the waves.d file if you want a double couple instead.

The waves.d listing for this example:

```
 &waves
  ksw= 0, stepz=20,
  modes=1,2,3,4,5,6,7,8,9,
  fmin=    1.0000, fmax=  100.0000,
  fsamin=   0.00100,
  curve='earth.crv',
  mapmat=0,
  matlb1='matc.m',   scilb1='matc.sci',
  matlb2='matu.m',   scilb2='matu.sci',
  irvsel=0,
  ofile='waves.tmp', /
  &source
  tm= 0.0, 0.0, 0.0,
     0.0, 0.0, 0.0,
     0.0, 0.0, 1.0, /
  sz=    0.00, sy=0.00, sx=0.00, /
  &recvr
  nrec=24,
  rz=24*0.0,
  ry=24*0.0,
rx=      1.000,     2.000,     3.000,     4.000,     5.000,
     6.000,     7.000,     8.000,     9.000,    10.000,
    11.000,    12.000,    13.000,    14.000,    15.000,
    16.000,    17.000,    18.000,    19.000,    20.000,
    21.000,    22.000,    23.000,    24.000,
  /
```

This file can be edited in case the **genwav** options don't cover what you want. If you want only the fundamental mode, for example, change the modes line:

```
modes=1,0,0,0,0,0,0,0,0,
```

The **irvsel** parameter is an easy way to change between **vertical** or **horizontal radial** signals on the receivers (see Figure 59). For guidance on the moment tensor, **tm**, see Aki & Richards (1980)

While the default is off-end shooting with the source at origin

```
sz= 0.00, sy=0.00, sx=0.00, /
```

This can be edited to place the shot somewhere else. For example, one can create a split spread by editing the example above:

```
sz= 0.00, sy=1.00, sx=12.00, /
```

which would place the shot slightly off line in the middle. Of course, the receivers can also be edited, changing rx, ry, and rz.

9.2.8 WAVES

WAVES computes elastic Rayleigh waves. Start with **GENDIS** 9.2.5, then run **DISPER** 9.2.6. Run **GENWAV** 9.2.7 to define a simulation geometry and parameters. IMPORTANT: Make sure the frequency increments are consistent between disper and waves (see 9.2.7.1). The **waves.d** file will then be input to **WAVES**. Outputs from WAVES include:

- **matu.m** Octave program to plot group velocity dispersion
- **matc.m** Octave program to plot phase velocity dispersion (redundant with phase.m output from disper).
- **m0.m** Octave program to plot wavelet and spectrum.
- **wavV.seg or wavR.seg** seismic shot gathers in BSEGY format.
- **waves.tmp** listing file for waves run.
- **waves.his** scaled lagrangian maximum for all runs made in the current directory. Smaller the better since integrating stiff equations.

The following are some plots generated from these outputs.

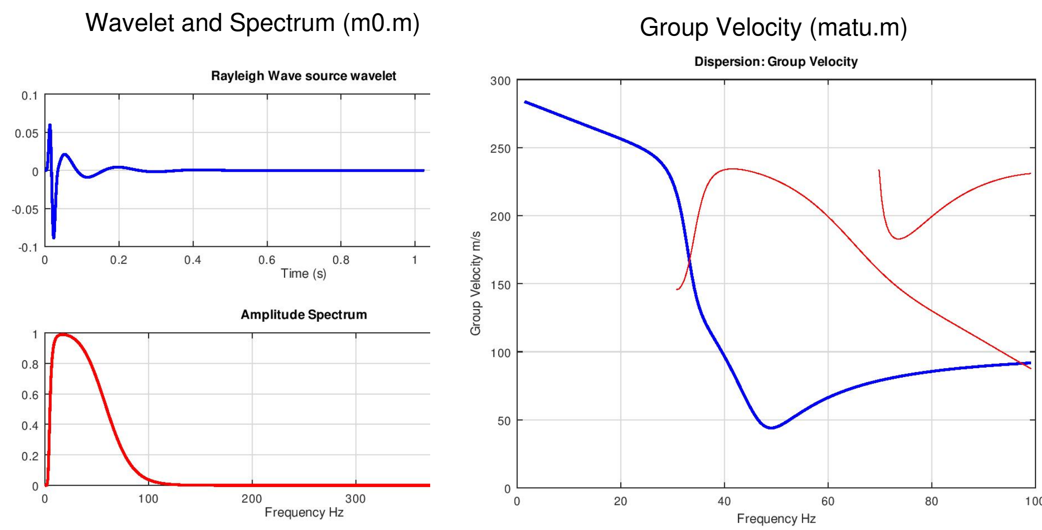


Figure 58: WAVES: Wavelet on left, group velocity dispersion on right. No significance to curve colors except that in the dispersion plot, the fundamental is Blue and higher modes are in Red. Soil representation is layer over half-space as shown in 9.2.5.1 and Figure 56 above.

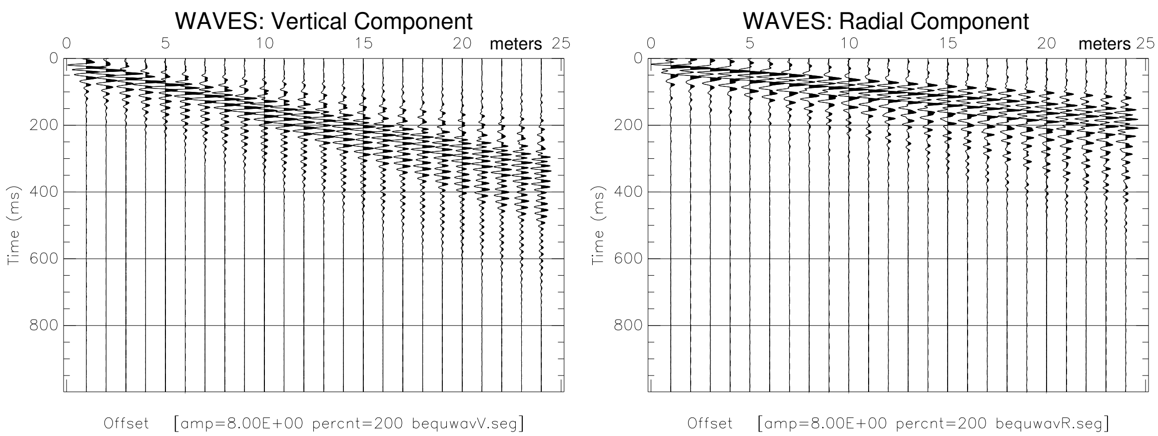


Figure 59: WAVES: Synthetic seismograms for Vertical (wavV.seg) and horizontal (wavR.seg) motion.

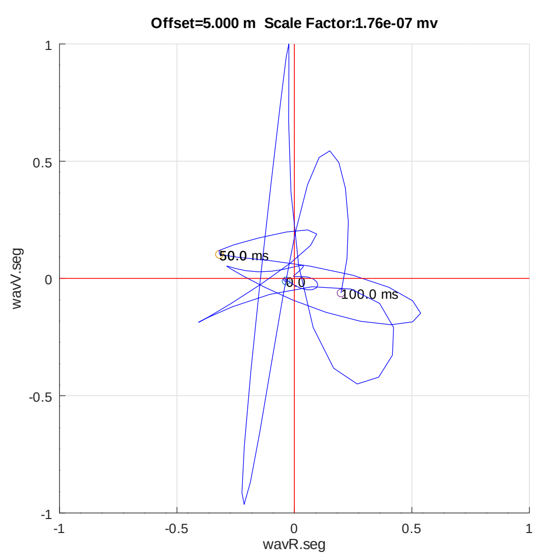


Figure 60: Hodogram for offset 5 meters. Requires bsegin.m, segyinfo.m, and hodo2plot.m in directory with wavV.seg and wavR.seg files (see 6.0.10).

While Rayleigh waves are often described to have elliptical retrograde motion, that is not always the case. Depending on the source depth, receiver depth, and the geologic profile, the motion can be either prograde or retrograde. In Figure 60 we see that the motion is complex, starting out with an ellipse with a sub-horizontal major axis, evolving to a vertical major axis of elliptical motion. An alternative case that illustrates retrograde elliptical motion is computed for the same waves.d. The revised disper.d uses a homogeneous half-space model (two points to describe). The result is shown in Figure 61:

```
show.tmp
      &disper
      nlay=    2,
    rho= 0.1700E+04, 0.1700E+04,
    mu= 0.6800E+08, 0.6800E+08,
    lame= 0.9520E+09, 0.9520E+09,
    zi= 0.0000E+00, 0.1000E+01,
      deltz=    1.0000,
      modemx=1,
      nfreq=202, flo=  0.1000000E+01, delf= 0.48828122E+00,   jsmax=300, ksw=0
       pvlcty=0.0, pfreq=0.0, zend=100.0,
      ofile='disper.tmp',
      octav1='phase.m', octav2='mat2.m',
      curve='earth.crv', /
        1        0.000      200.00      800.00    1700.0
        2        1.000      200.00      800.00    1700.0
```

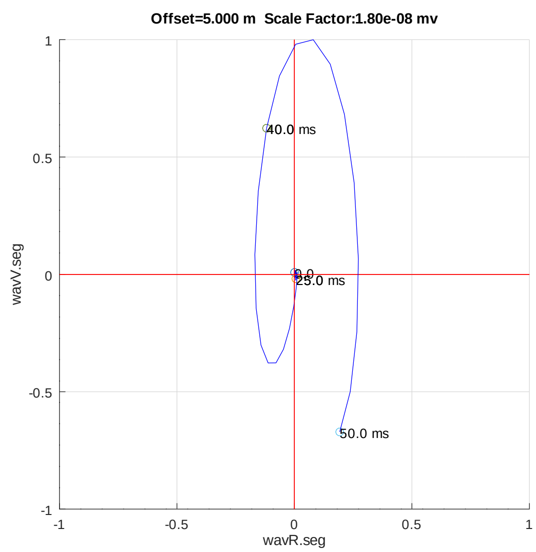


Figure 61: Hodogram at offset 5 meters for alternative half-space soil model, see show.tmp above. Sign conventions need to be taken into account when determining type of motion.

9.2.9 BDUM

This program reads a *.seg file, adopts the headers and replaces the data with an impulse at a user specified time. It can be used to present an impulse response or to benchmark software and do testing. In the following example, the filter program is illustrated. See Figure 62.

```
 bdum   infile   time_impulse

EXAMPLE: filter an impulse at 0.1 seconds, headers from c008.seg
bdum c008.seg .1
bfil bdumc008.seg 1 6 40 40 1
```

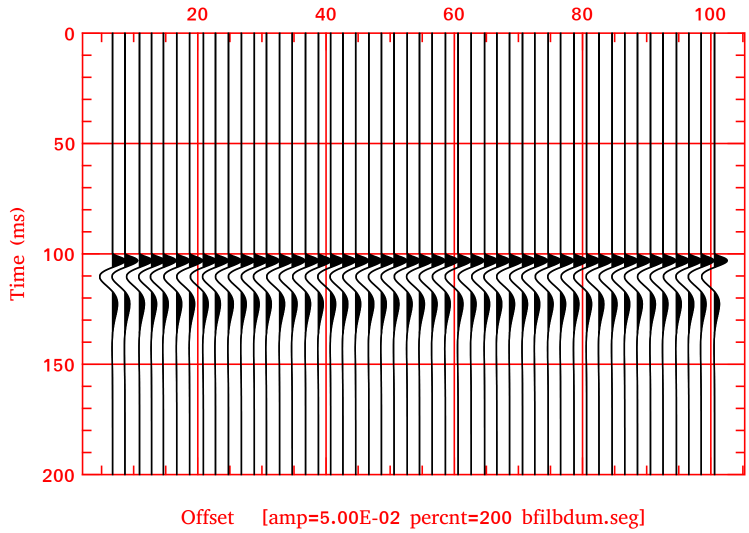


Figure 62: BDUM: Impuse replaced original data and filtered by BFIL program (band-pass 6 pole 40 Hz center, 40 Hz bandwidth, minimum phase).

9.2.10 OCTAVE rayleigh.m

Demonstration program on how to use the dispersion computation function **rwv.f** in an octave program. Required octave functions are:

- **rwv.f** computes dispersion, requires compilation and linking into the octave engine.
- **wrapper.cpp** wrapper code
- **build_disper_oct** script to compile rwv.f and build **disper.oct**
- **rayleigh.m** the demonstration code.

The build script has one active line plus some informational echo commands. The active line is:

```
mkoctfile rwv.f wrapper.cpp -o disper.oct
```

In order for this to work, one must have the development package installed for Octave. In Debian 10 Linux, the packages needed installation are:

- liboctave-dev
- octave-common

Of course, these are not the only packages, Octave has a lot of packages that are of use. But the above packages will install the mkoctfile. The wrapper.cpp and all BSU *.m files will be installed in `/usr/local/share/octave/site-m/` if you build BSU from the source tar archive. The locate command can also be helpful in other situations.

The disper() function returns a vector pv with fundamental and any higher modes. Here, it is demonstrated how to select and plot both fundamental and first higher mode. The higher mode becomes possible and recognized when the returned pv(2) value becomes > zero. This demonstration code searches backward to find the first non-zero case of the second component being non-zero.

Another similar code, **moho.m** is included that illustrates the same points as above. It shows how to display dispersion as a function of period rather than frequency.

10 Surveying, Setting Geometry, and Mapping

Setting geometry is the act of creating headers that include the locations of seismic sources and geophones. BSU includes programs for setting geometry as well as making maps and computer aided drafting (CAD) files from headers once they are set.

- **GENWAW** 10.1.1 Labor intensive interactive conversion of SEG-2 (*.DAT files) and setting geometry for each source and receiver.
- **GENREF** 10.1.2 interactive, generates bash scripts for setting geometry on CDP reflection shooting. Bison data ONLY.
- **TOPCON** 10.1.3 reads a survey *.nez file and **Bison** seismograph file, creates *.xyz file.
- **BHED** 10.1.4 down-load or up-load header data from or to an *.seg file.
- **TOPCON2** 10.1.5 converts **SEG-2** (*.DAT) to *.seg format while setting geometry from command line arguments. GENVSP 10.1.8 can be used first to set up bash scripts that use a *.nez file and calls to TOPCON2.
- **GENSETG** 10.1.6 interactive program creates files for setting geometry where phones fixed, shots move (reciprocal shooting)
- **SETGEOM** 10.1.7 Run after GENSETG, takes the shot.txt, phones.txt, and *nez files created by GENSETG and applies them. A *.nez file is Northing, Easting, Elevation text file.
- **GENVSP** 10.1.8 interactive program for setting geometry in down-hole surveys.
- **GENBHOD** 10.1.9 SH-wave source interactive program generates bash scripts to determine down-hole tool orientation by principle component analysis (PCA) of shot records. Program BHOD 10.1.11 does the actual PCA.
- **GENBHODV** 10.1.10 Vertical impact source interactive program generates bash scripts to determine down-hole tool orientation by PCA. Experimental, uses Rayleigh wave on horizontal component. Program BHOD 10.1.11 does the actual PCA.
- **BHOD**10.1.11 performs PCA on down-hole data.
- **BNEZ** 10.1.12 generates a *.nez file from rules. Typically run twice, once for shots, once for geophones.
- **TOP2NEZ** 10.1.13 converts a raw Topcon Total Station survey file to NEZ format.
- **TOP2DXF** 10.1.14 reads a *.nez file and converts it to a *.dxf (CAD) file.
- **TOPBCRD** 10.1.15 applies rotation and translations to coordinates in an *.nez file. Program BCRD 10.1.16 does this on *.seg files.
- **BCRD** 10.1.16 rotates and translates header geometry coordinates in an *.seg file.
- **BCAD** 10.1.17 creates a CAD *.dxf file from *.seg file headers.
- **SETSTREAM** 10.1.18 can be used to add geometry in the case of a land streamer.
- **GENSCRIPT** 10.1.18.1 Example Bash script to generate a script to process multiple land streamer files.

10.1 Setting Geometry

These are interactive codes for setting geometry. They are run from a terminal with a question and answer format.

10.1.1 GENWAW

Basic Seismic Utilities (BSU) interactive program for setting geometry. Code optimal for a walk-a-way type of data collection. Code is for SEG-2 format (*.DAT) files. This code prompts the user for shot and geophone locations. It should be run in the directory where the SEG-2 *.DAT files are located. The code scans the directory contents and builds a list of the files needing to have headers corrected for geometry. One use that makes geometry setting less of a burden is to set geometry for temporary local coordinates (ex. line along x-axis), then employ BSU program BCRD to rotate and translate coordinates to a final system. This program is run if the SEG-2 headers were not correctly set during acquisition (a common occurance).

EXAMPLE:
Here, the application is a walk-a-way with a fixed source at 0,0,0 and a single moving vertical component geophone, starting at an offset of 10 meters walking in 1 meter steps toward the source. **This is labor intensive, but very flexible. In a terminal, type the command:** `genwaw`

```
Enter Number of Channels 1
GEOPHONE ORIENTATIONS
Geophone Az 90=East Ver 180=Down
Channel=0 Enter Geophone Orientation  Az  Ver
0 180
     |----------------------------------------|
     |     Copyright (C) 2017 P. Michaels     |
     |           All rights reserved          |
     |     See GNU General Public License     |
     |----------------------------------------|
     waw:  TIME: 15:38:11 DATE: 29/May/2020
Nsources= 10
SOURCE LOCATION-------------------------------------
 0000.DAT: Enter Source X  Y  Z
0 0 0
 Trace:00 Enter Receiver X  Y  Z
10. 0 0
SOURCE LOCATION-------------------------------------
 0001.DAT: Enter Source X  Y  Z
0 0 0
 Trace:00 Enter Receiver X  Y  Z
9 0 0
SOURCE LOCATION-------------------------------------
 0002.DAT: Enter Source X  Y  Z
0 0 0
 Trace:00 Enter Receiver X  Y  Z
8. 0 0
.
. etc....
.
SOURCE LOCATION-------------------------------------
 0009.DAT: Enter Source X  Y  Z
0 0 0
 Trace:00 Enter Receiver X  Y  Z
1.0 0 0
```

NOTE: If there were more than one trace in each *.DAT file, there would be additional “Trace:” questions to answer.

Output includes the creation of a child directory, LST, in which the list files for each *.DAT file are stored. These are the result of the interactive program calling EGG2SEG 3.1.6. In this example, there will be files 0000.seg through 0009.seg, each with a single trace, now with headers as entered in GENWAW. To merge these into a single file, use BMRG (see 11.0.1). The command would be
`bmrg 000 0 9 1 1 1`
and that produces a file bmrg.seg. The headers can be checked by running BDUMP 4.0.1:

```
----------------------------------------------------------------------------------
   Length =   500 samples               |  Shot Elevation =        0.0
   Sample Interval =     0.00100 sec.   |  Shot Depth =        0.0
   Delay Time =    0 msec.              |  Up Hole Time =     0 msec
   Low Cut Filter  =     0 Hz.          |  Shot X-COORD =        0.00
   High Cut Filter =  100 Hz.           |  Shot Y-COORD =        0.00
   Line ID:  000                        |  Shot Date (year.moday) = 2019.0423
   Shot Orientation:                    |  Shot Time (hr:min)    = 15:20
   Azimuth=  0 Deg.  Vertical=  0 Deg.|  Charge Size (grams)=     0
 ----------------------------------------------------------------------------------
 TRACE|SHOT| STATION  | OFFSET|         RECEIVER           |VERT|1STBRK|K-GAIN|AZI|VER|
   #  |REC.|SHOT  REC|       |  ELEV.  X-COORD   Y-COORD|FOLD|(SEC.)| (dB)  |   |   |
 -----|----|---------|-------|----------------------------|--|------|------|---|---|
   1  |    0|0000 0001|  10.00|   0.00     10.00        0.00|  1|0.0000|   19  |   0|180|
   2  |    1|0000 0001|   9.00|   0.00      9.00        0.00|  1|0.0000|   19  |   0|180|
   3  |    2|0000 0001|   8.00|   0.00      8.00        0.00|  1|0.0000|   19  |   0|180|
   4  |    3|0000 0001|   7.00|   0.00      7.00        0.00|  1|0.0000|   19  |   0|180|
   5  |    4|0000 0001|   6.00|   0.00      6.00        0.00|  1|0.0000|   19  |   0|180|
   6  |    5|0000 0001|   5.00|   0.00      5.00        0.00|  1|0.0000|   19  |   0|180|
   7  |    6|0000 0001|   4.00|   0.00      4.00        0.00|  1|0.0000|   19  |   0|180|
   8  |    7|0000 0001|   3.00|   0.00      3.00        0.00|  1|0.0000|   19  |   0|180|
   9  |    8|0000 0001|   2.00|   0.00      2.00        0.00|  1|0.0000|   19  |   0|180|
  10  |    9|0000 0001|   1.00|   0.00      1.00        0.00|  1|0.0000|   19  |   0|180|
```

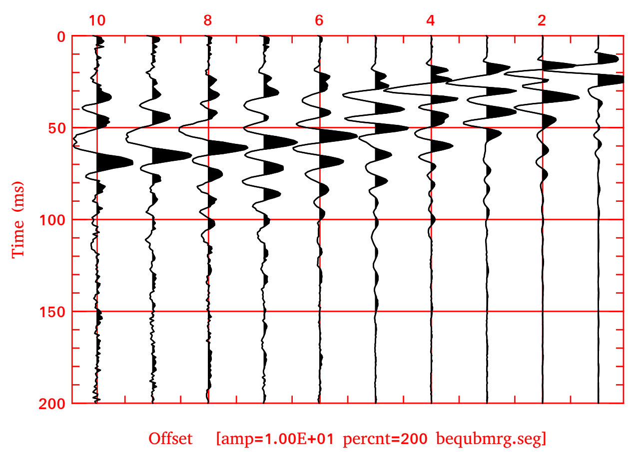


Figure 63: GENWAW: Example data from a small hammer source, trace equalized with BEQU 12.0.9.

10.1.2 GENREF

NOTE: This program is only for **BISON** data. For **SEG-2** formatted data, consider GENSETG 10.1.6 and SETGEOM 10.1.7. This is an interactive program that generates bash scripts for setting geometry when doing conventional "Roll-a-long" shooting, but no survey data are available. The program generates an *.nez file and scripts:

- **geom** run this first, it runs program TOPCON to generate *.xyz files.
- **geom2** run this second, it calls script **go1** which reads the *.xyz files
- **go1** called by geom2

TIP: Make sure to chmod +x the scripts to make them executable. The following is an example log of a small run.
From a terminal, type: `genref`

```
 |--------------------------------|
 | Copyright (C) 2017  P. Michaels |
 |       All rights reserved       |
 |see GNU General Public License   |
 |--------------------------------|
  CDP Roll-a-long Pattern Generator
  Bison Format Data
 -----------SOURCES----------------------------
   Enter 6 char. name for nez file (ex. STP001)
ABC001
   Enter 4 char. LINEID
0001
   Enter Z-Datum: Elevation
100
   Enter number of shots
5
   Enter Shot Record Names 8char: First
FAC10001
   Enter Shot Record Names 8char: Last
FAC10005
  Enter First Shot Station Number
1
   Enter First Source: x, y, z
0,0,100
   Enter Last Source: x, y, z
5,0,105
  Enter number of receivers in a shot gather
24
  Enter TOTAL NUMBER of stations on line
48
   Enter First Geophone Station: x, y, z
0,0,100
   Enter Last Geophone Station: x, y, z
48,0,109
  Enter first shot NEAR GEOPHONE station
0,0,100
  Enter first shot FAR GEOPHONE station
24,0,105
```

The ABC001.nez (index, Northing, Easting, Elev) file looks like this:

```
0001          0.0000     0.0000    200.0000  SP001
0002          0.0000     1.2500    201.2500  SP002
0003          0.0000     2.5000    202.5000  SP003
0004          0.0000     3.7500    203.7500  SP004
0005          0.0000     5.0000    205.0000  SP005
0001          0.0000     0.0000    200.0000  VP001
0002          0.0000     1.0213    200.1915  VP002
0003          0.0000     2.0426    200.3830  VP003
0004          0.0000     3.0638    200.5745  VP004
0005          0.0000     4.0851    200.7660  VP005
0006          0.0000     5.1064    200.9574  VP006
.
. etc...
.
0041          0.0000    40.8511    207.6596  VP041
0042          0.0000    41.8723    207.8511  VP042
0043          0.0000    42.8936    208.0426  VP043
0044          0.0000    43.9149    208.2340  VP044
0045          0.0000    44.9362    208.4255  VP045
0046          0.0000    45.9574    208.6170  VP046
0047          0.0000    46.9787    208.8085  VP047
0048          0.0000    48.0000    209.0000  VP048
```

The labels “SP” are shot locations, the labels “VP” are voltage points (geophone) locations.

The **geom** file calls **topcon** 10.1.3 for the **Bison** files FAC10001 etc., and looks like this:

```
topcon ABC001.nez FAC10001 0001 0.0  1 24 000 023   1  0.   0   0   0   0
topcon ABC001.nez FAC10002 0001 0.0  2 24 001 024   2  0.   0   0   0   0
topcon ABC001.nez FAC10003 0001 0.0  3 24 002 025   3  0.   0   0   0   0
topcon ABC001.nez FAC10004 0001 0.0  4 24 003 026   4  0.   0   0   0   0
topcon ABC001.nez FAC10005 0001 0.0  5 24 004 027   5  0.   0   0   0   0
```

The **geom2** script looks like this:

```
 go1 001
 go1 002
 go1 003
 go1 004
 go1 005
```

The **go1** script looks like this:

```
bis2seg FAC10$1
bhed FAC10$1.seg FAC10$1.xyz 0
mv bhedFAC1.seg F$1.seg
rm FAC10$1.seg
```

10.1.3 TOPCON

For **BISON** data. The program combines *.nez survey file (Northing, Easting, Elevation) and Bison files from a Bison seismograph to produce *.xyz header files. The program BHED 10.1.4 then reads the *.xyz files and uploads them into the *.seg files. The command line arguments are:

```
topcon  topf bisf lid shdp is nch vp1 vpn ir esh isa isv ira ita

 topf     =file name of topcon .nez file
 bisf     =file name of Bison file with data
 lid      =line ID
 shdp     =shot depth
 is       =shot location number
 nch      =number of channels (<48)
 vp1      =geophone location number channel 1
 vpn      =geophone location number channel nchanl
 ir       =shot record number
 esh      =elevation adjustment to be added
 isa      =source polarization azimuth (deg.)
 isv      =source polarization vertical (deg.)
 ira      =reference polarization R-axis (deg.)
 ita      =reference polarization T-axis (deg.)
```

10.1.4 BHED

BHED either uploads or downloads header data into/from *.seg files. The command line arguments are:

```
  bhed infile  header_file  iupdn

 infile       =input file name
 header_file  =file with selected header info
 iupdn        =1 download headers to header_file
              =0 upload headers to BSEGY data set
```

Aside from initial upload of headers, one can also use this program to edit existing headers. Just download to a header file from an existing *.seg headers, open the header file and edit. **HINT: Watch out for zeros. In particular, note that a binary zero is used to terminate character strings. Depending on how initial headers were set, it is possible that a header string might have a binary zero, often shown as a @ symbol in an editor like VI**.

As a sample, the top of a header file looks like this:

```
&BHED
 LOWCUT=8     ,
 HIGHCT=500   ,
 LINE="4N__",
 YEAR=1994  ,
 DAY=1117  ,
 HOUR=11    ,
 MINUTE=46    ,
 PHONE="VERT",
 SDEPTH= 0.400000006    ,
 UPHOLE=  0.00000000    ,
 CHARGE=0     ,
 SREC=8     ,
 /
   1 0.0000    003  9668.130 10131.190   818.700    001  9670.780 10125.040   818.840   0 000 180 000 000
   2 0.0000    003  9668.130 10131.190   818.700    002  9671.670 10123.330   818.860   0 000 180 000 000
   3 0.0000    003  9668.130 10131.190   818.700    003  9671.120 10120.710   818.840  20 000 180 000 000
   4 0.0000    003  9668.130 10131.190   818.700    004  9673.280 10119.480   818.830  20 000 180 000 000
   5 0.0000    003  9668.130 10131.190   818.700    005  9674.080 10117.840   818.760  20 000 180 000 000
   6 0.0000    003  9668.130 10131.190   818.700    006  9674.990 10115.940   818.670  40 000 180 000 000
   7 0.0000    003  9668.130 10131.190   818.700    007  9675.950 10114.170   818.710  40 000 180 000 000
   8 0.0000    003  9668.130 10131.190   818.700    008  9676.910 10112.280   818.790  40 000 180 000 000
   9 0.0000    003  9668.130 10131.190   818.700    009  9677.660 10110.530   818.720  40 000 180 000 000
  10 0.0000    003  9668.130 10131.190   818.700    010  9678.490 10108.690   818.720  40 000 180 000 000
  11 0.0000    003  9668.130 10131.190   818.700    011  9679.280 10106.840   818.720  40 000 180 000 000
etc .....
  42 0.0000    003  9668.130 10131.190   818.700    042  9705.330 10050.830   819.400  60 000 180 000 000
  43 0.0000    003  9668.130 10131.190   818.700    043  9706.300 10049.040   819.400  60 000 180 000 000
  44 0.0000    003  9668.130 10131.190   818.700    044  9707.120 10047.260   819.390  60 000 180 000 000
  45 0.0000    003  9668.130 10131.190   818.700    045  9708.000 10045.380   819.420  60 000 180 000 000
  46 0.0000    003  9668.130 10131.190   818.700    046  9708.910 10043.660   819.410  60 000 180 000 000
  47 0.0000    003  9668.130 10131.190   818.700    047  9709.710 10041.910   819.380  60 000 180 000 000
  48 0.0000    003  9668.130 10131.190   818.700    048  9710.460 10039.950   819.480  60 000 180 000 000
```

These are read as namelist files by BHED. The above was created by the command:
`bhed c008.seg header.txt 1`
and the header file was created with the name header.txt.

10.1.5 TOPCON2

For **SEG-2** data. The program combines *.nez survey file data with the SEG-2 seismic file data to produce a BSEGY format file, *.seg. One difference between this and the Bison data **TOPCON** 10.1.3 procedure is that there is no need to run **BHED** with an intermediate *.xyz file. This goes directly to *.seg. The command line arguments are:

```
 topcon2  topf seg2f lid shdp is nch vpl vpn ir esh isa isv ira ita

   topf   =  topcon file name
   seg2f  =  seg-2 file name
   lid    =  line ID
   shdp   =  shot depth
   is     =  shot location number
   nch    =  number of channels (nch<66)
   vp1    =  geophone station channel 1
   vpn    =  geophone station channel n
   ir     =  shot record number
   esh    =  elevation adjustment to be added
   isa    =  source polarization azimuth (deg.)
   isv    =  source polarization vertical (deg.)
   ira    =  reference phone polarization R-axis (deg.)
   ita    =  reference phone polarization T-axis (deg.)
```

An example of issuing the command for a down-hole surve:

```
topcon2 stp001.nez 1051.DAT 00X5 0.0 1 6 0156 0151 1051 0.  270 135 0 270
```

This combines survey file stp001.nez with SEG-2 data 1051.DAT. In the case of down-hole data, the only orientation of horizontal components is known for the reference phone. To determine the orientation of a down-hole phone see **GENBHOD** 10.1.9 and **BHOD** 10.1.11. A partial listing of the resulting header dump by program **BDUMP** 4.0.1 follows:

```
                    |------------------------------|
                    |   PARTIAL SEGY HEADER DUMP   |
                    |                              |
                    |           1051.seg           |
                    |                              |
                    |------------------------------|

 ----------------------------------------------------------------------------------
   Length =  2000 samples              |  Shot Elevation =      849.2
   Sample Interval =    0.00025 sec.   |  Shot Depth =        0.0
   Delay Time =    0 msec.             |  Up Hole Time =     0 msec
   Low Cut Filter  =    0 Hz.          |  Shot X-COORD =    9963.09
   High Cut Filter = 1000 Hz.          |  Shot Y-COORD =   10022.70
   Line ID: 00X5                       |  Shot Date (year.moday) = 1999.1102
   Shot Orientation:                   |  Shot Time (hr:min)   = 14:25
   Azimuth=270 Deg.  Vertical=135 Deg.|  Charge Size (grams)=     0
 ----------------------------------------------------------------------------------
 TRACE|SHOT| STATION | OFFSET|        RECEIVER          |VERT|1STBRK|K-GAIN|AZI|VER|
   #  |REC.|SHOT  REC|       |  ELEV.  X-COORD   Y-COORD|FOLD|(SEC.)| (dB) |   |   |
 -----|----|---------|-------|-------------------------|--|------|------|---|---|
    1 |   0|0001 0156|   0.73|  848.96    9963.09  10022.00| 3|0.0000|    0 |270| 90|
    2 |   0|0001 0155|   0.73|  848.96    9963.09  10022.00| 3|0.0000|    0 |  0| 90|
    3 |   0|0001 0154|   0.73|  848.96    9963.09  10022.00| 3|0.0000|   24 |  0|  0|
    4 |   0|0001 0153|  14.49|  834.68    9963.09  10023.25| 3|0.0000|   24 |  0| 90|
    5 |   0|0001 0152|  14.49|  834.68    9963.09  10023.25| 3|0.0000|   24 |  0| 90|
    6 |   0|0001 0151|  14.49|  834.68    9963.09  10023.25| 3|0.0000|   24 |  0|  0|
```

10.1.6 GENSETG

This program sets up files for a second program **SETGEOM** 10.1.7 which does the actual setting of geometry for SEG2 data. A primary application is reciprocal refraction shooting where blocks of geophones are irregularly placed on banks of a river. Given the flexible nature of this pair of programs, it can be useful for other applications as well. This is an interactive program, and produces two text files, one for shots, one for geophones. An example log of a run is shown below. **From a terminal, type:** `gensetg`

```
SHOTS: ---------------------------
 Enter first shot file NAME number
1001
 Enter last shot file NAME number
1004
 Enter first SP label NUMBER
01
 Enter increment for SP label NUMBER
01
PHONES: ---------------------------
 Enter number of BLOCKS to define channels
2
BLOCK Number------- 1
Channels (1) through (?)
 Enter last channel for this block
12
 Enter first label VP NUMBER for this block
01
 Enter label VP increment for this block
01
BLOCK Number------- 2
Channels (13) through (?)
 Enter last channel for this block
24
```

```
 Enter first label VP NUMBER for this block
50
 Enter label VP increment for this block
1
```

The two files output are **shots.txt** and **phones.txt**. The shots.txt file contains the following:

```
1001.seg  SP001
1002.seg  SP002
1003.seg  SP003
1004.seg  SP004
```

The phones.txt file contains the following:

```
01  VP001
02  VP002
03  VP003
04  VP004
05  VP005
06  VP006
07  VP007
08  VP008
09  VP009
10  VP010
11  VP011
12  VP012
13  VP050
14  VP051
15  VP052
16  VP053
17  VP054
18  VP055
19  VP056
20  VP057
21  VP058
22  VP059
23  VP060
24  VP061
```

These SP and VP labels would correspond to those in an *.nez file produced by a surveying instrument. This example might correspond to channels 1–12 being on one bank of a river, then a jumper cable might cross the river and connect to channels 13–24 with geophones on the other bank of the river. The 4 shots might then be taken with an airgun deployed from the bridge. In reality, there would likely be more shots than 4, but this illustrates the concept.

10.1.7 SETGEOM

After running **GENSETG** 10.1.6, one needs to also have a survey *.nez file before proceeding. Continuing the example started in 10.1.6, this migh look like this:

```
 1    0.000000    0.000000   100.000000 SP001
 2    0.000000    2.000000   101.000000 SP002
 3    0.000000    4.000000   102.000000 SP003
 4    0.000000    6.000000   103.000000 SP004
 5    2.000000    2.000000   100.000000 VP001
 6    3.000000    2.000000   100.000000 VP002
 7    4.000000    2.000000   100.000000 VP003
 8    5.000000    2.000000   100.000000 VP004
 9    6.000000    2.000000   100.000000 VP005
10    7.000000    2.000000   100.000000 VP006
11    8.000000    2.000000   100.000000 VP007
12    9.000000    2.000000   100.000000 VP008
13   10.000000    2.000000   100.000000 VP009
14   11.000000    2.000000   100.000000 VP010
15   12.000000    2.000000   100.000000 VP011
16   13.000000    2.000000   100.000000 VP012
17    2.000000    8.000000   125.000000 VP050
18    3.000000    8.000000   125.000000 VP051
```

```
19    4.000000    8.000000   125.000000 VP052
20    5.000000    8.000000   125.000000 VP053
21    6.000000    8.000000   125.000000 VP054
22    7.000000    8.000000   125.000000 VP055
23    8.000000    8.000000   125.000000 VP056
24    9.000000    8.000000   125.000000 VP057
25   10.000000    8.000000   125.000000 VP058
26   11.000000    8.000000   125.000000 VP059
27   12.000000    8.000000   125.000000 VP060
28   13.000000    8.000000   125.000000 VP061
```

The *.nez file contains the (N,E,Z) coordinates and must include the SP and VP labels that match the shots.txt and phones.txt files. If the SEG2 data files were converted to BSEGY format with **EGG2SEG** 3.1.6 we might have files 1001.seg through 1004.seg in our directory. We would then run setgeom with the following command:

```
setgeom shots.txt phones.txt samp0000.nez
```

where it is assumed that the *.nez file is as shown here. The output BSEGY files would be setg1001.seg through setg1004.seg. The header dump using BDUMP 4.0.1 of file setg1001.seg would look like this:

```
 ----------------------------------------------------------------------------------
  Length =  1000 samples              |  Shot Elevation =      100.0
  Sample Interval =     0.00050 sec.  |  Shot Depth =        0.0
  Delay Time =    0 msec.             |  Up Hole Time =      0 msec
  Low Cut Filter  =     8 Hz.         |  Shot X-COORD =       0.00
  High Cut Filter =  500 Hz.          |  Shot Y-COORD =       0.00
  Line ID: 001^@                       |  Shot Date (year.moday) = 1994.1117
  Shot Orientation:                   |  Shot Time (hr:min)   = 11:46
  Azimuth=  0 Deg.  Vertical=180 Deg.|  Charge Size (grams)=     0
----------------------------------------------------------------------------------
TRACE|SHOT| STATION  | OFFSET|         RECEIVER          |VERT|1STBRK|K-GAIN|AZI|VER|
  #  |REC.|SHOT  REC|       |  ELEV.  X-COORD   Y-COORD|FOLD|(SEC.)| (dB)  |   |   |
-----|----|---------|-------|---------------------------|--|------|------|---|---|
   1 |   1| 001  001|   2.83|  100.00       2.00       2.00|  1|0.0000|    0 |  0|180|
   2 |   1| 001  002|   3.61|  100.00       2.00       3.00|  1|0.0000|    0 |  0|180|
   3 |   1| 001  003|   4.47|  100.00       2.00       4.00|  1|0.0000|   20 |  0|180|
   4 |   1| 001  004|   5.39|  100.00       2.00       5.00|  1|0.0000|   20 |  0|180|
   5 |   1| 001  005|   6.32|  100.00       2.00       6.00|  1|0.0000|   20 |  0|180|
   6 |   1| 001  006|   7.28|  100.00       2.00       7.00|  1|0.0000|   40 |  0|180|
   7 |   1| 001  007|   8.25|  100.00       2.00       8.00|  1|0.0000|   40 |  0|180|
   8 |   1| 001  008|   9.22|  100.00       2.00       9.00|  1|0.0000|   40 |  0|180|
   9 |   1| 001  009|  10.20|  100.00       2.00      10.00|  1|0.0000|   40 |  0|180|
  10 |   1| 001  010|  11.18|  100.00       2.00      11.00|  1|0.0000|   40 |  0|180|
  11 |   1| 001  011|  12.17|  100.00       2.00      12.00|  1|0.0000|   40 |  0|180|
  12 |   1| 001  012|  13.15|  100.00       2.00      13.00|  1|0.0000|   40 |  0|180|
  13 |   1| 001  050|  26.36|  125.00       8.00       2.00|  1|0.0000|   40 |  0|180|
  14 |   1| 001  051|  26.46|  125.00       8.00       3.00|  1|0.0000|   40 |  0|180|
  15 |   1| 001  052|  26.59|  125.00       8.00       4.00|  1|0.0000|   40 |  0|180|
  16 |   1| 001  053|  26.76|  125.00       8.00       5.00|  1|0.0000|   40 |  0|180|
  17 |   1| 001  054|  26.96|  125.00       8.00       6.00|  1|0.0000|   40 |  0|180|
  18 |   1| 001  055|  27.20|  125.00       8.00       7.00|  1|0.0000|   40 |  0|180|
  19 |   1| 001  056|  27.48|  125.00       8.00       8.00|  1|0.0000|   40 |  0|180|
  20 |   1| 001  057|  27.78|  125.00       8.00       9.00|  1|0.0000|   40 |  0|180|
  21 |   1| 001  058|  28.12|  125.00       8.00      10.00|  1|0.0000|   40 |  0|180|
  22 |   1| 001  059|  28.50|  125.00       8.00      11.00|  1|0.0000|   40 |  0|180|
  23 |   1| 001  060|  28.90|  125.00       8.00      12.00|  1|0.0000|   40 |  0|180|
  24 |   1| 001  061|  29.33|  125.00       8.00      13.00|  1|0.0000|   40 |  0|180|
```

Note that the line ID has a binary zero (^@). We would fix that by dumping the headers with BHED 10.1.4, then editing that zero out, replacing it with perhaps a space or some other valid ASCII character. This would be followed by an upload of the edited header file into the *.seg data by a second run of BHED. Some renaming would be required. The flow would look like this:

```
bhed setg1001.seg 01.hed 1
(edit the file 01.hed, say with VI)
bhed setg1001.seg 01.hed 0
mv bhedsetg.seg 1001.seg
```

The final result would be over writing 1001.seg with the corrected header version.

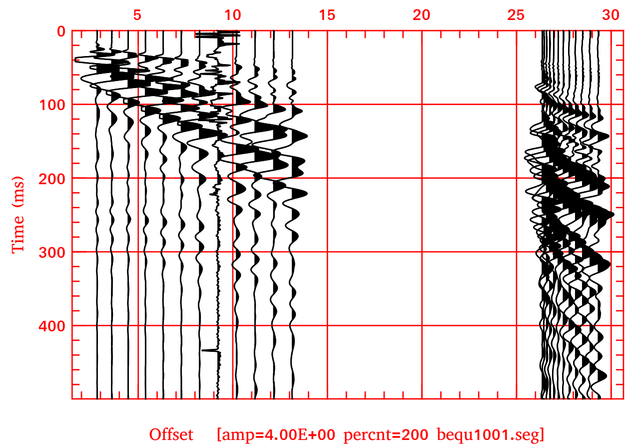


Figure 64: An example of what a plot by offset might look like, trace equalized with BEQU 12.0.9.

10.1.8 GENVSP

This is a Vertical Seismic Profile, VSP (down-hole) survey pattern generator. It is an interactive program that creates a NEZ file that can be used to assign geometry to a down-hole survey. Suitable for use with either Bison or SEG-2 file formats. The naming of the shot record files is used to determine the type of data. Bison file names are 8 character alpha numeric without a suffix, SEG-2 files are assumed to have names like 1000.DAT

The hole is assumed to be vertical, reference phone fixed. This code is hardwired for how the author acquires data. Note the initial Channel order switch assumption choice at the beginning of the program execution. See Michaels (1998).

Output files are:

- *.nez (Northing Easting Elevation file) (10.1.8.1)
- geom (calls topcon program, unites *nez survey data with the data file, producing an *.xyz header file) (10.1.8.2)
- geom2 (calls script go1 for each shot effort) (10.1.8.3)
- go1 (calls BHED program to unite *.xyz header data with seismic data into BSEGY data formated files.) (10.1.8.4)

NOTE: change the permissions on geom, geom2, and go1 files to executable. For example:

```
chmod +x geom
```

EXAMPLE RUN:

```
  Down-hole VSP  Pattern Generator
  For Setting Geometry
  Handles both Bison and SEG-2 File Formats

  Set Channel Order Switch
   1=ascending  1,2,3=downhole 4,5,6=reference
  -1=descending 6,5,4=downhole 3,2,1=reference
   2=ascending  1,2,3=down 4,5,6=ref,7=load_cell
  -2=descending 7=load_cell,6,5,4=down 3,2,1=ref

1
```

```
 -----------BOREHOLE---------------------------
   Enter 6 char. name for nez file (ex. STP001)
STP001
   Enter 4 char. LINEID
0001
   Enter Z-Datum: Casing Elevation
849.

   BOREHOLE LOCATION:
   Borehole is origin of the local coordinate system
   Source and Reference phone locations are x,y
   relative to borehole.

   Following entries will shift every x,y input to
   a final global coordinate system:
   Enter Global x-coord. of borehole
1000
   Enter Global y-coord. of borehole
1000
   Enter number of sources
1
 FOR THIS SOURCE:
   Enter Shot Record Names 8char: First
STP30001
   Enter Shot Record Names 8char: Last
STP30100
     STP30001STP30100
   Enter Source: x, y, z_sub_CE (positive down)
0,-1,-.5
   Enter Source Polarization: azi, ver
0, 180

 -----------REFERENCE RECEIVER------------------
   Enter Reference: x, y, z_sub_CE (positive down)
0,+1,-.4
   Enter Reference Polarizations: R-azi, T-azi
0,270

 -----------BOREHOLE PHONES---------------------
   Enter Bulk Shift (Added To Geophone Depth ONLY)
.3
  For Shot: SP01  AZI=  0 VER=180

   Enter Station Spacing:  dz
.25
   Enter First Station Depth:  zmax
20
   Enter Last Station Depth:  zmin
0.5
  Number of receivers =    79
---------------------------------------------
  CHECK DATA TYPE

  Files like XXXX0001 detected, ID=BISON
  Is above ID Correct, or overide needed?
  1=YES correct    0=NO incorrect
1
```

In the above example, the tool has 6 channels, so there will be 6 lines for each down-hole station. The first 3 lines are the down-hole components, 2 horizontal, 1 vertical. The next 3 lines are the reference phone (note the elevation column does not change for the reference phone since it is stationary).

10.1.8.1 nez

The NEZ files starts like this:

```
0001          999.0000   1000.0000    849.5000  SP01
0001         1000.0000   1000.0000    828.7000  VP0001
0002         1000.0000   1000.0000    828.7000  VP0002
0003         1000.0000   1000.0000    828.7000  VP0003
0004         1001.0000   1000.0000    849.4000  VP0004
0005         1001.0000   1000.0000    849.4000  VP0005
0006         1001.0000   1000.0000    849.4000  VP0006
0007         1000.0000   1000.0000    828.9500  VP0007
0008         1000.0000   1000.0000    828.9500  VP0008
0009         1000.0000   1000.0000    828.9500  VP0009
0010         1001.0000   1000.0000    849.4000  VP0010
0011         1001.0000   1000.0000    849.4000  VP0011
0012         1001.0000   1000.0000    849.4000  VP0012
.
.
.
0469         1000.0000   1000.0000    848.2000  VP0469
0470         1000.0000   1000.0000    848.2000  VP0470
0471         1000.0000   1000.0000    848.2000  VP0471
0472         1001.0000   1000.0000    849.4000  VP0472
0473         1001.0000   1000.0000    849.4000  VP0473
0474         1001.0000   1000.0000    849.4000  VP0474
```

10.1.8.2 geom

The bash script, **geom** file starts like this (for bison data in this instance, calls topcon 10.1.3):

```
topcon STP001.nez STP30001 0001 0.0  1  6 0001 0006    1  0.   0 180   0 270
topcon STP001.nez STP30002 0001 0.0  1  6 0007 0012    2  0.   0 180   0 270
topcon STP001.nez STP30003 0001 0.0  1  6 0013 0018    3  0.   0 180   0 270
topcon STP001.nez STP30004 0001 0.0  1  6 0019 0024    4  0.   0 180   0 270
topcon STP001.nez STP30005 0001 0.0  1  6 0025 0030    5  0.   0 180   0 270
topcon STP001.nez STP30006 0001 0.0  1  6 0031 0036    6  0.   0 180   0 270
.
.
.
topcon STP001.nez STP30074 0001 0.0  1  6 0439 0444   74  0.   0 180   0 270
topcon STP001.nez STP30075 0001 0.0  1  6 0445 0450   75  0.   0 180   0 270
topcon STP001.nez STP30076 0001 0.0  1  6 0451 0456   76  0.   0 180   0 270
topcon STP001.nez STP30077 0001 0.0  1  6 0457 0462   77  0.   0 180   0 270
topcon STP001.nez STP30078 0001 0.0  1  6 0463 0468   78  0.   0 180   0 270
topcon STP001.nez STP30079 0001 0.0  1  6 0469 0474   79  0.   0 180   0 270
```

10.1.8.3 geom2

The bash script **geom2** file starts like this:

```
go1 001
go1 002
go1 003
go1 004
go1 005
go1 006
.
.
.
go1 095
go1 096
go1 097
go1 098
go1 099
go1 100
```

10.1.8.4 go1

For the instance of Bison data, the **go1** file is a bash script (calls bis2seg 3.1.4):

```
bis2seg STP30$1
bhed STP30$1.seg STP30$1.xyz 0
mv bhedSTP3.seg S$1.seg
rm  STP30$1.seg
```

10.1.9 GENBHOD

This is an program that generates bash script to conduct Principle Component Analysis (PCA) on down-hole data (Michaels, 2001b). A down-hole tool will rotate as it comes up the hole, and there is a need to determine the horizontal component orientations. This is an interactive program. The following is an example log of a run for a single station (normally, the last file will reflect many stations in a survey). **From a terminal, type the command:** `genbod`

```
 |--------------------------------|
 | Copyright (C) 2009  P. Michaels |
 |       All rights reserved       |
 |see GNU General Public License   |
 |--------------------------------|

 WARNING: !!
 See Source Code, genbhod.f, or BSU documentation
 (man pages and BSU user Guide)
 before you use this program. It is hardwired for
 a specific type of acquisition.

 enter 1char_ALPHA PREFIX
c
  enter FIRST FILE NUMBER (<=3digits)
  for which source polarization is 270 deg.
009
  enter LAST FILE NUMBER (<=3digits)
  for which source polarization is 270 deg.
009
  enter UP/DOWN SWITCH
    -1= 90 Azimuth File Number 1 LESS than 270 Az
    +1= 90 Azimuth File Number 1 MORE than 270 Az
1
  enter azimuth of bowspring(R-comp)
180
  OUTPUT====> Downhole: gobhodo
  OUTPUT====> Reference: gobhodoR
  OUTPUT====> Downhole: gorunbhod
  OUTPUT====> Reference: gorunbhodR
  ----------------------------------------
 REMEMBER to change permissions on the
 above files to execute.
  ----------------------------------------
  IF examining the Down-hole Phone

  1. Run gobhodo in directory with 6 chan
  records (3 down, 3 reference phones)

  2. Run gorunbhod in directory with files
  that are named hxxxyyy.seg

  ----------------------------------------
  IF examining the Reference Phone

  1. Run gobhodoR in the directory with
  the 6 channel records.

  2. Run gorunbhodR in the directory with
  files that are named rxxxyyy.seg

  ----------------------------------------
```

After running the interactive program, change permissions of the generated scripts. For example, `chmod +x go*`. One can analyze either the fixed reference phone (scripts gobhodoR and gorunbhodR) or the down-hole phone (scripts gobhodo and gorunbhod). Because of a 180^o ambiguity in the result of the PCA analysis, one must observe and recored the tool orientation when it comes out of the hole. It is assumed that one starts logging the data with the first station at the deepest depth in the hole, pulling the tool up to the surface. For tools with a clamping bowspring, determine the orientation of the horizontal components relative to the bowspring and observe the bowspring orientation when the tool is at the last station. This PCA procedure is for horizontal sources where there are two source efforts at each subsurface station. These are of opposite polarity and recorded in separate files which can be subtracted to enhance SH-waves (Michaels, 1998).

The procedure is to run the gobhodo script which will scale and then subtract the two source efforts. For example, at a station where the *.seg files are c009.seg and c010.seg, the result will be a file, h010009.seg. For the single station example above, the **gobhodo** script is:

```
bscl c010.seg 1 1 3
bscl c009.seg 1 1 3
bsum bsclc010.seg  bsclc009.seg -1.0
mv bsumbscl.seg h010009.seg
```

Next, the script **gorunbhod** is run. It consists of **BHOD** program commands like this:

```
bhod h010009.seg 2 3 50 90.0 180.0 +90.0
```

See program **BHOD** 10.1.11 for more. The final result is a file **bhod.lst** which contains the horizontal component orientations that will be applied to data headers (**BTOR** 12.2.2) and later rotate the data as desired **BROT** 12.2.4. In addition to the bhod.lst file, the **gorunbhod** script calls to **BHOD** produces Postcript files showing the analysis results (see Figure 65).

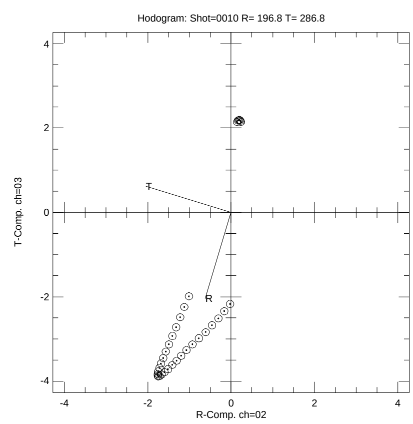


Figure 65: BHOD: plot produced showing PCA results for a geophone at about 19.39 meters depth. File bhod.lst: (00010 196.8 286.8) = (seq. R-azi, T-azi)

10.1.10 GENBHODV

Interactive program go generate bash scripts to determine down hole tool orientation by PCA with a VERTICAL IMPACT SOURCE. This is experimental, assumes Rayleigh waves generated by source. **From a terminal, type:** `genbhodV`

EXPERIMENTAL program generates 4 bash script files which can be run to determine geophone orientations based on the large particle motion. The concept is experimental. In short, one uses the horizontal motion of the Rayleigh wave in the context of the experimental setup to determine horizontal tool orientation. It works well for the reference phone, but your mileage may vary down-hole (depending on the depth of penetration of the Rayleigh wave, and on the subsurface nodal pattern of the Rayleigh wave. Two of the scripts are for a surface, reference geophone, and two are for the down-hole geophone. The Principal Component Analysis (PCA) is actually done by the program, **BHOD** 10.1.11. There are many assumptions made in this code.

10.1.10.1 Example Log The following log is for illustration, and is for a single source effort by a vertical source recorded on a file, c200.seg. In practice one would have many files, and the number of files will be set by the first and last file numbers.

```
 WARNING: !!
 See Source Code, genbhodV.f, or BSU documentation
 (man pages and BSU user Guide)
 before you use this program. It is hardwired for
 a specific type of acquisition.

 enter 1char_ALPHA PREFIX
c
  enter FIRST FILE NUMBER (<=3digits)
200
  enter LAST FILE NUMBER (<=3digits)
200
  enter azimuth of bowspring(R-comp)
180
  OUTPUT====> Downhole: gobhodo
  OUTPUT====> Reference: gobhodoR
  OUTPUT====> Downhole: gorunbhod
  OUTPUT====> Reference: gorunbhodR
  ----------------------------------------
 REMEMBER to change permissions on the
 above files to execute.
  ----------------------------------------
  IF examining the Down-hole Phone

  1. Run gobhodo in directory with 6 chan
  records (3 down, 3 reference phones)

  2. Run gorunbhod in directory with files
  that are named hxxx.seg

  ----------------------------------------
  IF examining the Reference Phone

  1. Run gobhodoR in the directory with
  the 6 channel records.

  2. Run gorunbhodR in the directory with
  files that are named rxxx.seg
    EXPERIMENTAL APPROACH on Rayleigh Wave
  ----------------------------------------
```

The **gobhodo** script scales the data (**BSCL** 12.0.10 by the maximum absolute value on traces 1 to 1 (ie. trace 1). The last option 3 sets the choice to maximum absolute value. The script is:

```
bscl c200.seg 1 1 3
mv  bsclc200.seg  h200.seg
```

The **gorunbhod** calls the **BHOD** 10.1.11 program:
`bhod h200.seg 2 3 50 0.0 180.0 +90.0` The file **bhod.lst** contains the solution:
`00200  263.0  353.0`
(file number, R-azimuth, T-azimuth) The bhod.lst is input to program **BTOR** 12.2.2 which rotates the data and updates the headers.

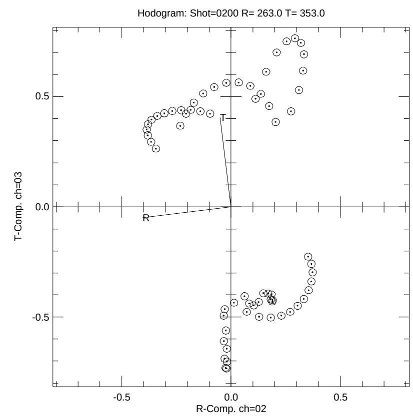


Figure 66: BHOD: plot produced showing PCA results for a geophone at about 11.68 meters depth.

10.1.11 BHOD

Hodogram analysis by Principal Component Analysis (PCA) can determine the orientation of the horizontal components of a 3-C geophone in a bore hole. Program **BHOD** does this analysis and outputs a file, **bhod.lst** that has rows of 3 numbers, (seq., R-azimuth, T-azimuth). The sequence number corresponds to the *.seg files. File, bhod.lst, is then used by **BTOR** 12.2.2 to update headers. Program **BROT** 12.2.4 is then used to rotate the data to a desired orientation.

In the case of a horizontal impulse source, two opposite polarities will be struck for each geophone depth. Depending on which source blow azimuth is first, the bhod.lst sequence number will either be the first or second blow, and the result of PCA will be applied to both source efforts at the depth being analyzed. In a typical survey there will be twice as many *.seg files as depth stations occupied. One surveys from the bottom to the top of the hole, and should make an IMPORTANT observation of the tool orientation at the surface to resolve the 180^0 ambiguity. Helper scripts are generated by **GENBHOD** 10.1.9.

In the case of a vertical impulse source, experimental helper scripts are generated by program **GENBHODV** 10.1.10. In this case, the procedure is designed to observe the large amplitude hodogram motion (which may be a Rayleigh wave). Rayleigh waves are a mix of P-SV motion. The P-motion is horizontal and may provide orientation information. Your mileage will vary depending how deep the Rayleigh waves motion penetrates.

The command line arguments to **BHOD** are:

```
bhod   infile chR chT ipct saz azctl tsw1

infile =input file name
chR    = channel with R-component (int)
chT    = channel with T-component (int)
ipct   = percent of max amplitude to include (int)
saz    = source azimuth (ie E-W, then 90 deg)
azctl  = desired direction for R- (bowspring)
         (azctl resolves 180 deg ambiguity PCA)
tsw1   = switch to set T-comp relative to R-comp
         T-comp Azimuth= R-Comp + tsw1
         Typically, tsw1= +90.(downhole) or -90.(ref)
```

EXAMPLE:

```
bhod h002001.seg 2 3 50 90.0 315.0 +90.0
```

bhod.lst file:

```
01002   258.6   348.6
```

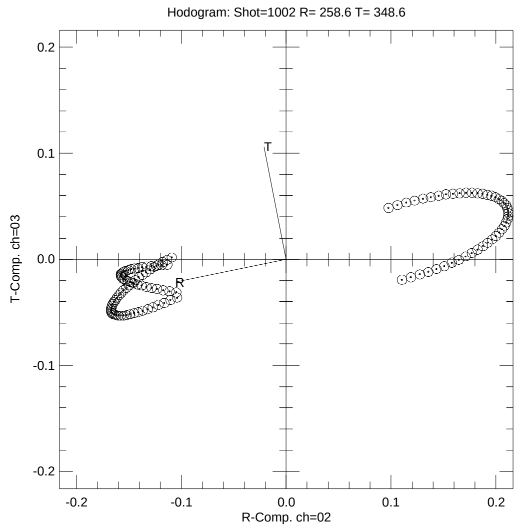


Figure 67: BHOD: plot produced showing PCA results for a geophone at about 20 meters depth.

10.1.12 BNEZ

Each row of an NEZ file provides the (Seq. Northing, Easting, Elevation, Tag). The Tag specifies either a source (SP) or geophone (VP, voltage point). One way of running the program is to run BNEZ twice to create shot.nez and phones.nez files which are then merged into a single NEZ file. Depending on if you have Bison or SEG-2 data recorded, use either program **TOPCON** 10.1.3 or **TOPCON2** 10.1.5 to generate *.xyz files that can be used by the **BHED** 10.1.4 program to set the geometry in the *.seg file headers.

The command line arguments are:

```
  bnez  outfile  n-points tag so yo xo zo ido dy dx dz did

     outfile  = output file name (ex. aaaa0001.nez

     n-points = number of survey points to generate
     tag      = 1 tag=VP
              = 2 tag=SP
     so       = first value of sequence number
     yo       = northing of first point
     xo       = easting of first point
     zo       = elevation of first point
     ido      = initial ID number
     dy       = spacing between points in north direction
     dx       = spacing between points in east direction
     dz       = spacing in elevation between points
     did      = interval in ID between points
```

10.1.12.1 Example, BNEZ

The commands for a single shot gather, Bison data, are:

```
bnez 000001.nez 1 2 1 0 0 0 01 1 1 1 1
bnez 000002.nez 12 1 2 140. 0. 0. 01 10. 0. 0. 1
cp 000001.nez LOG001.nez
cat 000002.nez >>LOG001.nez
cat LOG001.nez
```

```
     1    0.000000    0.000000    0.000000 SP001
     2  140.000000    0.000000    0.000000 VP001
     3  150.000000    0.000000    0.000000 VP002
     4  160.000000    0.000000    0.000000 VP003
     5  170.000000    0.000000    0.000000 VP004
```

```
 6  180.000000    0.000000    0.000000 VP005
 7  190.000000    0.000000    0.000000 VP006
 8  200.000000    0.000000    0.000000 VP007
 9  210.000000    0.000000    0.000000 VP008
10  220.000000    0.000000    0.000000 VP009
11  230.000000    0.000000    0.000000 VP010
12  240.000000    0.000000    0.000000 VP011
13  250.000000    0.000000    0.000000 VP012
```

The next step is to combine the Bison file headers with the NEZ and produce an output *.xyz file:

```
topcon LOG001.nez LOST0001 0001 0.0  1 12 001 012   1  0.   0   0   0   0
cat LOST001.xyz
```

```
 &bhed
    lowcut=  16, highct= 500, year=1992, day=0303,
     line='0001', hour=17, minute=07,
    sdepth= 0.0, uphole=0.000, phone='VERT', srec= 001,
&end
001 0.0000    001      0.000      0.000      0.000     001      0.000     140.000      0.000     60  000  000  000  000
002 0.0000    001      0.000      0.000      0.000     002      0.000     150.000      0.000     60  000  000  000  000
003 0.0000    001      0.000      0.000      0.000     003      0.000     160.000      0.000     60  000  000  000  000
004 0.0000    001      0.000      0.000      0.000     004      0.000     170.000      0.000     60  000  000  000  000
005 0.0000    001      0.000      0.000      0.000     005      0.000     180.000      0.000     60  000  000  000  000
006 0.0000    001      0.000      0.000      0.000     006      0.000     190.000      0.000     60  000  000  000  000
007 0.0000    001      0.000      0.000      0.000     007      0.000     200.000      0.000     60  000  000  000  000
008 0.0000    001      0.000      0.000      0.000     008      0.000     210.000      0.000     60  000  000  000  000
009 0.0000    001      0.000      0.000      0.000     009      0.000     220.000      0.000     60  000  000  000  000
010 0.0000    001      0.000      0.000      0.000     010      0.000     230.000      0.000     60  000  000  000  000
011 0.0000    001      0.000      0.000      0.000     011      0.000     240.000      0.000     60  000  000  000  000
012 0.0000    001      0.000      0.000      0.000     012      0.000     250.000      0.000     60  000  000  000  000
```

The final step is to use **BIS2SEG** 3.1.4 to convert the Bison file to *.seg, and then use **BHED** 10.1.4 to apply the headers to the *.seg file.

```
bis2seg LOST0001
bhed LOST0001.seg LOST0001.xyz 0
#plot the data by offset
bplt bhedLOST.seg 0 0 1 1 12 0 .5 1 4E-3 200
bdump bhedLOST.seg 0
cat bdump.lst
```

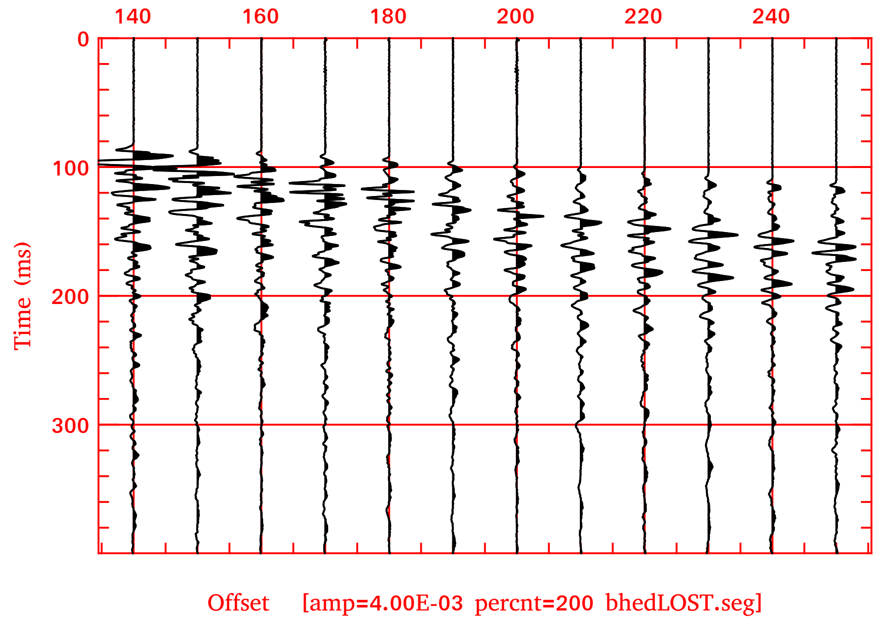


Figure 68: BNEZ: Plot of Bison file data with geometry added.

```
|-------------------------------|
|   PARTIAL SEGY HEADER DUMP    |
|                               |
|         bhedLOST.seg          |
|                               |
|-------------------------------|

----------------------------------------------------------------------------------
  Length =  2000 samples               | Shot Elevation =        0.0
  Sample Interval =    0.00020 sec.    | Shot Depth =        0.0
  Delay Time =     0 msec.             | Up Hole Time =     0 msec
  Low Cut Filter  =   16 Hz.           | Shot X-COORD =       0.00
  High Cut Filter =  500 Hz.           | Shot Y-COORD =       0.00
  Line ID: 0001                        | Shot Date (year.moday) = 1992.0303
  Shot Orientation:                    | Shot Time (hr:min)    = 17:07
  Azimuth=  0 Deg.  Vertical=  0 Deg.| Charge Size (grams)=     0
----------------------------------------------------------------------------------
```

TRACE #	SHOT REC.	STATION SHOT	STATION REC	OFFSET	RECEIVER ELEV.	RECEIVER X-COORD	RECEIVER Y-COORD	VERT FOLD	1STBRK (SEC.)	K-GAIN (dB)	AZI	VER
1	1	001	001	140.00	0.00	0.00	140.00	1	0.0000	60	0	0
2	1	001	002	150.00	0.00	0.00	150.00	1	0.0000	60	0	0
3	1	001	003	160.00	0.00	0.00	160.00	1	0.0000	60	0	0
4	1	001	004	170.00	0.00	0.00	170.00	1	0.0000	60	0	0
5	1	001	005	180.00	0.00	0.00	180.00	1	0.0000	60	0	0
6	1	001	006	190.00	0.00	0.00	190.00	1	0.0000	60	0	0
7	1	001	007	200.00	0.00	0.00	200.00	1	0.0000	60	0	0
8	1	001	008	210.00	0.00	0.00	210.00	1	0.0000	60	0	0
9	1	001	009	220.00	0.00	0.00	220.00	1	0.0000	60	0	0
10	1	001	010	230.00	0.00	0.00	230.00	1	0.0000	60	0	0
11	1	001	011	240.00	0.00	0.00	240.00	1	0.0000	60	0	0
12	1	001	012	250.00	0.00	0.00	250.00	1	0.0000	60	0	0

10.1.13 TOP2NEZ

Topcon is one of a number of of Electronic Distance Measuring (EDM) instruments. It can be controlled with an FC4 module that stores measurements in an ASCII format assuming a Microsoft file convention. For example, consider a file survey.n:

```
00001
10000.00000
10000.00000
1000.00000
BP1
00003
10000.00000
10000.00000
1000.00000
C2-48
.
.
.
00266
10318.48928
10144.12327
1002.47977
SLEDGE7
00267
10320.68105
10136.84087
995.98539
1-SP5A
00268
10205.99591
10104.81427
1002.29261
SLEDGE6
```

There are tags, sequence numbers and (y,x,z) coordinates, one item per line. This program converts the file to a NEZ file format, all items in a single line corresponding to the tag. For example, **From a terminal, type the command**:

```
top2nez survey.n
```

We will have output file **survey.n.nez:**

```
00001       10000.00000 10000.00000 1000.00000  BP1
00003       10000.00000 10000.00000 1000.00000  C2-48
.
.
.
00266       10318.48928 10144.12327 1002.47977  SLEDGE7
00267       10320.68105 10136.84087 995.98539   1-SP5A
00268       10205.99591 10104.81427 1002.29261  SLEDGE6
```

The NEZ format is read by BSU programs. Program TOP2DXF 10.1.14 can be used to create a CAD file for making base maps.

10.1.14 TOP2DXF

The command line arguments are:

```
*.nez file format(12x,f12.0,f12.0,f12.0,a)

top2dxf    infile   isw1   ilabel    txtsiz

infile  =*.nez input file name
isw1    =switch to control limits
       0=no limits header
       1=limits based on min and max values
ilabel 0=no printing of point labels
       1=print labels
txtsiz  =size of text in coord. units (float)
```

For an example, consider file **samp0000.nez**:

```
     1    0.000000    0.000000  100.000000 SP001
     2    0.000000    2.000000  101.000000 SP002
     3    0.000000    4.000000  102.000000 SP003
     4    0.000000    6.000000  103.000000 SP004
     5    2.000000    2.000000  100.000000 VP001
     6    3.000000    2.000000  100.000000 VP002
     7    4.000000    2.000000  100.000000 VP003
     8    5.000000    2.000000  100.000000 VP004
     9    6.000000    2.000000  100.000000 VP005
    10    7.000000    2.000000  100.000000 VP006
    11    8.000000    2.000000  100.000000 VP007
    12    9.000000    2.000000  100.000000 VP008
    13   10.000000    2.000000  100.000000 VP009
    14   11.000000    2.000000  100.000000 VP010
    15   12.000000    2.000000  100.000000 VP011
    16   13.000000    2.000000  100.000000 VP012
    17    2.000000    8.000000  125.000000 VP050
    18    3.000000    8.000000  125.000000 VP051
    19    4.000000    8.000000  125.000000 VP052
    20    5.000000    8.000000  125.000000 VP053
    21    6.000000    8.000000  125.000000 VP054
    22    7.000000    8.000000  125.000000 VP055
    23    8.000000    8.000000  125.000000 VP056
    24    9.000000    8.000000  125.000000 VP057
    25   10.000000    8.000000  125.000000 VP058
    26   11.000000    8.000000  125.000000 VP059
    27   12.000000    8.000000  125.000000 VP060
    28   13.000000    8.000000  125.000000 VP061
```

In a terminal, we type the command:

```
top2dxf samp0000.nez 0 1 .25
```

Figure 69 illustrates how the output file, **samp0000.dxf** can be read by a common CAD program (here Qcad). Other programs that can read Digital Exchange Format (DXF) files include Microstation and Autocad. Raw EDM files, like from Topcon FC4 controllers can be converted to the NEZ format using **TOP2NEZ** 10.1.13.

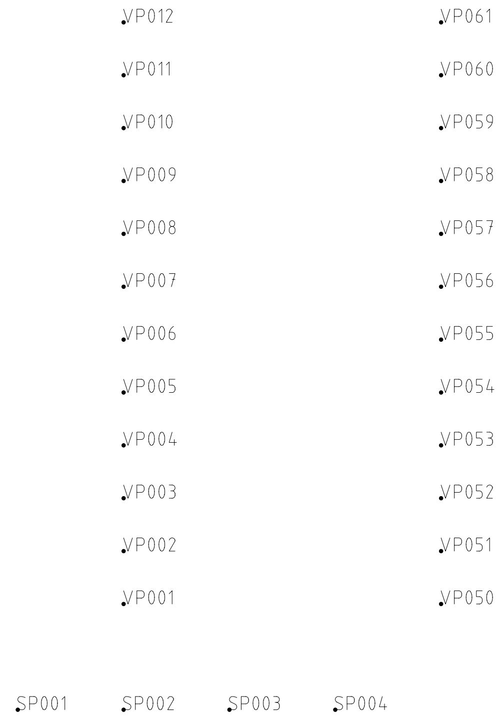


Figure 69: QCAD: Qcad used to read the file samp0000.dxf and exported to a PDF file. The point SP001 is at the origin, (0,0,0).

10.1.15 TOPBCRD

An NEZ file can be transformed by scale, shift, and rotations using this program. The program does the same thing as program **BCRD** 10.1.16 (which operates on BSEGY, *.seg, format data). The command line arguments are:

```
topbcrd   infile   theta   sfact   x0   y0   z0

 infile  =input file name
 theta   =angle (deg) from current to new x-axis
    (+)theta counterclockwise (-)theta clockwise
 sfact   =units scale factor: (old)*(sfact)=(new)
 x0      =X-offset in new coord. units
 y0      =Y-offset in new coord. units
 zo      =Y-offset in new coord. units
```

An example of modifying the **TOP2NEZ** 10.1.13 example above follows. The transform is 45 degree counter-clockwise rotation, scale factor = 1, shift of (20,20,0) meters. The point **SP001** is at the origin, (0,0,0) in the original *.nez file. After transform, the point, **SP001**, is at (20,20,0). The grid rotates counter-clockwise (or the survey points appear rotated clockwise).

```
topbcrd samp0000.nez 45. 1. 20. 20. 0.
top2dxf samp0000.mod 0 1 .25
```

Figure 70 shows the modified survey file after plotting with Qcad. The new *.nez file is samp000.mod.

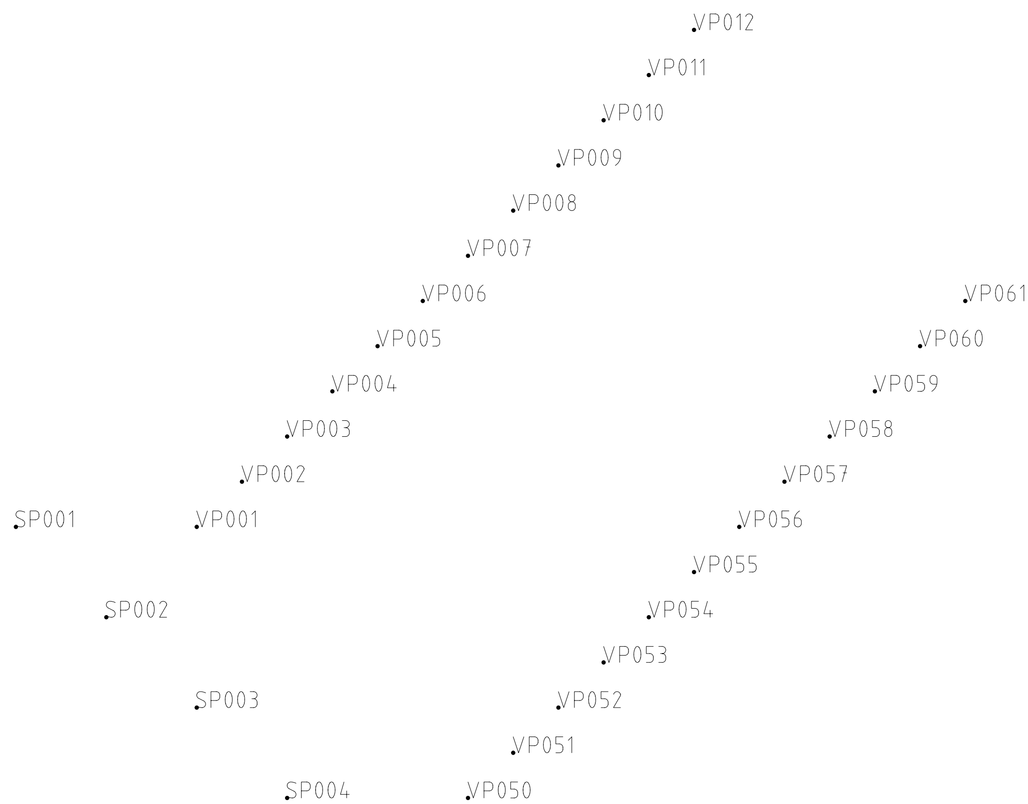


Figure 70: QCAD: Qcad plot of modified samp0000.nez file, samp0000.mod. Point SP001 is now at (20,20,0).

10.1.16 BCRD

This program does the same thing as **TOPBCRD** 10.1.15. The difference is that the input file is a BSEGY, *.seg, file instead of a NEZ survey file. The command line arguments are:

```
 bcrd infile theta sfact x0 y0 z0

infile  =input file name
theta   =angle (deg) from current to new x-axis
   (+)theta counterclockwise (-)theta clockwise
sfact   =units scale factor: (old)*(sfact)=(new)
x0      =X-offset in new coord. units
y0      =Y-offset in new coord. units
zo      =Z-offset in new coord. units
```

NOTE: When using **BCRD** or **TOPBCRD**, be aware that rotation and translation at the same time may not be the same as translation first, output a file, then rotation second on the translated file. The result of the translation and rotation operations may be viewed by running the program **BCAD** 10.1.17 which produces a DXF file from the altered headers.

10.1.17 BCAD

Similar to **TOP2DXF** 10.1.14. Rather than reading an NEZ survey file, this program takes a BSEGY (*.seg) file for input, and outputs a DXF file suitable to be read by a cad program. The command line arguments are:

```
 bcad       infile   isw1   ilabel    txtsiz

infile  =*.seg input file name
isw1    =switch to control limits
       0=no limits header
       1=limits based on min and max values
ilabel 0=no printing of point labels
       1=print labels
txtsiz  =size of text in coord. units (float)
```

BCAD Example:

```
bcad c008.seg 1 1 1.0
qcad bcadc008.dxf
```

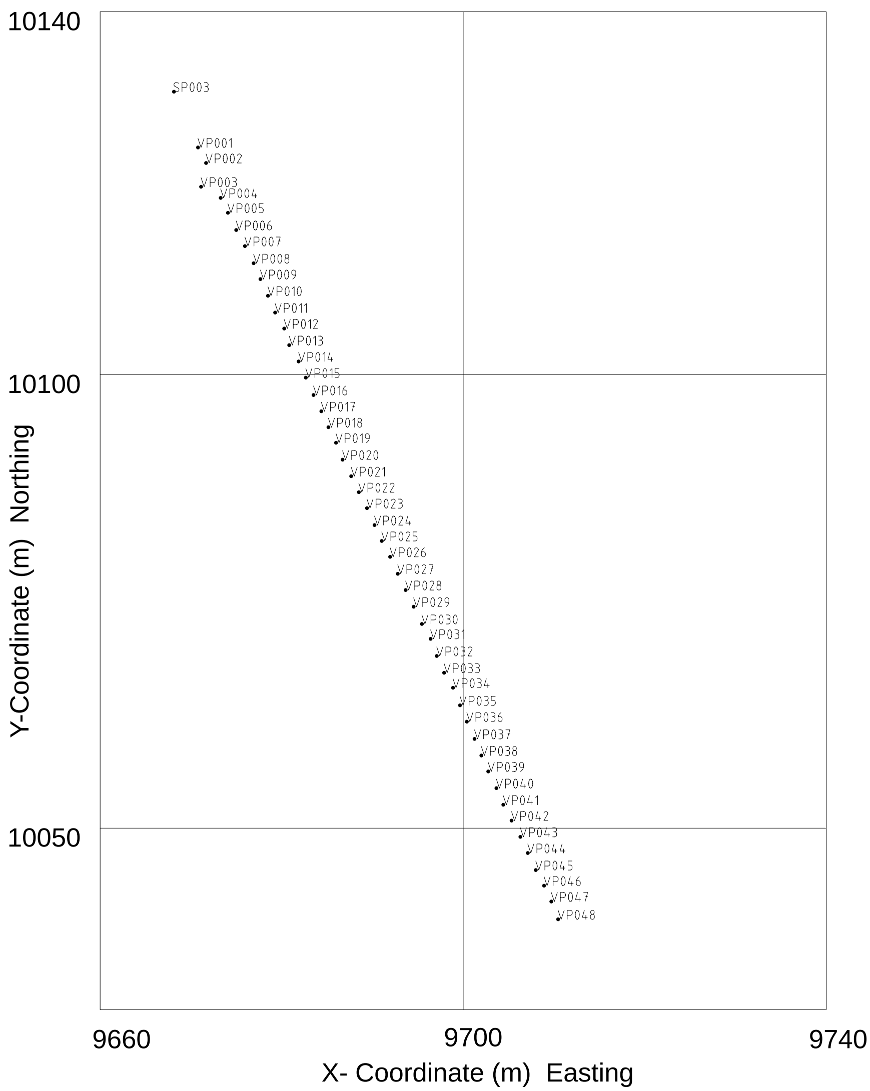


Figure 71: BCAD: DXF file edited, add some coordinates and labels. Editing DXF in QCAD, http://qcad.org/en/

10.1.18 SETSTREAM

In the case of a land streamer, assume that the coordinates of the source are known. Geophone coordinates are specified relative to the source in terms of a gap to the first phone, an interval between phones, and an azimuth of the cable from the source. The number of channels is also required as below:

```
setstream  infile  Sx Sy Sz Nch Ggap Gint Az

    infile  = input file name (4char minimum)

    Sx      = source x-coordinate
    Sy      = source y-coordinate
    Sz      = source z-coordinate
    Nch     = number of channels
    Ggap    = gap to first geophone
    Gint    = interval between phones
    Az      = azimuth of cable from source (N=0 deg)
              West=270 deg
```

10.1.18.1 Generating a script to process multiple files An example of how to create a bash script that processes a number of files using the same local coordinates is included in the GENSCRIPT example below (see /usr/local/share/scripts for the installed location). The output of GENSCRIPT is RUNSCRIPT .

Example FXYZ.txt file used by GENSCRIPT

```
FILE#     Sx         Sy          Sz
7001   424601.45   4512411     1293.8957
7002   424601.45   4512411     1293.8957
7003   424601.93   4512411     1293.902
7004   424602.34   4512410.9   1293.9074
7005   424602.7    4512410.9   1293.9122
7006   424603.07   4512410.9   1293.917
7007   424603.43   4512410.9   1293.9217
7008   424603.78   4512410.8   1293.9263
7009   424604.12   4512410.8   1293.9308
7010   424604.46   4512410.8   1293.9352
7011   424604.79   4512410.8   1293.9395
  .
  .
  .
```

GENSCRIPT

```
#!/bin/bash
# EXAMPLE genscript program to set geometry for land streamer
# Edit as needed.  Note local origin to prevent problems in BSEGY headers (integer)
# Origin: see line 11 (424600.,4512410.0)=(x,y)
# FXYZ.txt file columns: [record# Sx Sy Sz] source locations
echo "mkdir -p SEG" >runscript
echo "mkdir -p LST" >>runscript
# convert *.sgd SEGD files to *.seg BSEGY with default headers
cat FXYZ.txt | gawk '{ print "segd2seg "$1".sgd" }' >>runscript
# run setstream program to add geometry to *.seg file
cat FXYZ.txt |gawk '{print "setstream "$1".seg ",$2-424600.,$3-4512410.0,$4" 48 8.0  0.5 272.415"}'>>runscript
# move and rename the setstreamer *.seg files
cat FXYZ.txt | gawk '{ print "mv sets"$1".seg SEG/"$1".seg"}' >>runscript
# move all *.lst file to LST directory
echo "mv *.lst LST" >>runscript
# removing *.seg file that lack geometry headers
echo "rm -f *.seg" >>runscript
chmod +x runscript
```

NOTE: A local origin is applied by the gawk command above. The BSEGY header format uses integer (16 bit) combined with a scalar multiplier/divider to hold the coordinates of the shot and receivers. GPS sourced coordinates may exceed this size integer.

Example Sections of RUNSCRIPT Generated by GENSCRIPT

```
mkdir -p SEG
mkdir -p LST
segd2seg 7001.sgd
segd2seg 7002.sgd
segd2seg 7003.sgd
segd2seg 7004.sgd
segd2seg 7005.sgd
. . .
setstream 7001.seg  1.45 1 1293.8957 48 8.0  0.5 272.415
setstream 7002.seg  1.45 1 1293.8957 48 8.0  0.5 272.415
setstream 7003.seg  1.93 1 1293.902 48 8.0  0.5 272.415
setstream 7004.seg  2.34 0.9 1293.9074 48 8.0  0.5 272.415
setstream 7005.seg  2.7 0.9 1293.9122 48 8.0  0.5 272.415
. . .
mv sets7001.seg SEG/7001.seg
mv sets7002.seg SEG/7002.seg
mv sets7003.seg SEG/7003.seg
mv sets7004.seg SEG/7004.seg
mv sets7005.seg SEG/7005.seg
. . .
mv *.lst LST
rm -f *.seg
```

Sample portion of header dump (see program BDUMP 4.0.1)

```
                      |-------------------------------|
                      |   PARTIAL SEGY HEADER DUMP    |
                      |                               |
                      |          7001.seg             |
                      |                               |
                      |-------------------------------|

 ----------------------------------------------------------------------------------
   Length =  2000 samples             |  Shot Elevation =     1293.9
   Sample Interval =   0.000500 sec.  |  Shot Depth =        0.0
   Delay Time =    0 msec.            |  Up Hole Time =     0 msec
   Low Cut Filter  =    0 Hz.         |  Shot X-COORD =       1.45
   High Cut Filter = 1000 Hz.         |  Shot Y-COORD =       1.00
   Line ID: 0001                      |  Shot Date year.[moday|julian] =  22.0292
   Shot Orientation:                  |  Shot Time (hr:min)    = 17:18
   Azimuth=  0 Deg.  Vertical=180 Deg.|  Charge Size (grams)=     0
 ----------------------------------------------------------------------------------
 TRACE|SHOT| STATION | OFFSET|        RECEIVER          |VERT|1STBRK|K-GAIN|AZI|VER|
   #  |REC.|SHOT  REC|       |  ELEV.  X-COORD   Y-COORD|FOLD|(SEC.)| (dB) |   |   |
 -----|----|---------|-------|--------------------------|--|------|------|---|---|
    1 |7001|7001 6985|   8.00| 1293.90     -6.54      1.34| 1|0.0000|  180 |  0|  0|
    2 |7001|7001 6984|   8.50| 1293.90     -7.04      1.36| 1|0.0000|  180 |  0|  0|
    3 |7001|7001 6983|   9.00| 1293.90     -7.54      1.38| 1|0.0000|  180 |  0|  0|
    4 |7001|7001 6982|   9.50| 1293.90     -8.04      1.40| 1|0.0000|  180 |  0|  0|
    5 |7001|7001 6981|  10.00| 1293.90     -8.54      1.42| 1|0.0000|  180 |  0|  0|
    6 |7001|7001 6980|  10.50| 1293.90     -9.04      1.44| 1|0.0000|  180 |  0|  0|
    7 |7001|7001 6979|  11.00| 1293.90     -9.54      1.46| 1|0.0000|  180 |  0|  0|
    8 |7001|7001 6978|  11.50| 1293.90    -10.04      1.48| 1|0.0000|  180 |  0|  0|
    9 |7001|7001 6977|  12.00| 1293.90    -10.54      1.50| 1|0.0000|  180 |  0|  0|
   10 |7001|7001 6976|  12.50| 1293.90    -11.04      1.53| 1|0.0000|  180 |  0|  0|
   11 |7001|7001 6975|  13.00| 1293.90    -11.54      1.55| 1|0.0000|  180 |  0|  0|
   12 |7001|7001 6974|  13.50| 1293.90    -12.04      1.57| 1|0.0000|  180 |  0|  0|
. . .
```

11 Editing BSEGY Data

Once data are in the BSEGY format (ie. *.seg files), they can be edited in a number of ways:

- **BMRG** 11.0.1 Merge data from many files into one file
- **BEDT** 11.0.2 Edit data by traces aperture, time aperture. Edit data by interpolation or decimation of samples (includes anti-alias option for decimation).
- **BRSP** 11.0.3 Resample data. Interpolation by augmentation with zeros in frequency domain. Does not introduce any new frequencies.
- **BKIL** 11.0.4 Remove or zero out traces by range or by list.
- **BEXT** 11.0.5 Extract traces either by shot or receiver name, or by field record number.
- **BOFF** 11.0.6 Compute offset header from coordinates of shot and receiver, insert into BSEGY header.
- **BWIN** 11.0.7 Temporal window of BSEGY data. Tapers from a start time to full, extends to end time, tapers to zero.
- **BHED** 10.1.4 Extract or upload headers for BSEGY data.
- **BXOF** 11.0.8 Extract traces by offset.

11.0.1 BMRG

Often data collected in surveys results in a number of files which are numbered sequentially. For example, in down-hole surveys, each file may relate to a down-hole station for a single source effort. There may be a number of components recorded at each station. This would also be the case in walk-a-way surface data collection. **BMRG** permits one to select a sequence of files, and specific traces in each file to output into a single file.

```
bmrg pfix iffile ilfile ifinc iftrc iltrc

pfix:  =prefix for input file names
 NOTE:pfix length is no. of invarient charcters
  Ex. If file names run s001 to s090 then pfix=s0
  Ex. If file names run s001 to s132 then pfix=s
iffile =number of first file (suffix)
  EXAMPLE:  if file=s001.seg, iffile=001

ilfile =number of last file (suffix)
  EXAMPLE:  if file=s092.seg, ilfile=092

ifinc  =increment for file number (suffix)
iftrc  =first trace each file
iltrc  =last trace each file
```

For example, consider a down-hole survey with files w001.seg through w166.seg. The file order in each file is:

```
Channel  Component
   1        Vertical   (down-hole)
   2        Radial     (down-hole)
   3        Transverse (down-hole)
   4        Vertical   (ref. phone)
   5        Radial     (ref. phone)
   6        Transverse (ref. phone)
```

The reference phone is fixed at the surface and the down hole phone is logged from bottom to surface. We want to collect the transverse down-hole phone, channel 3, and output that to a single file for every other source effort. The command:

```
bmrg w 001 166 2 3 3
```

If we want every source effort, the command is:

```
bmrg w 001 166 1 3 3
```

Figure 72 shows both cases. Since the source blow is 135 degrees from the vertical, the horizontal T-component will show different polarity of source effort (every trace has the checkered look, peaks against troughs, plotted by elevation).

The data have not been rotated to a standard orientation. The T-component in this example drifts from 313 to 288 degrees azimuth as determined by PCA analysis (see **BHOD** 10.1.11). Program **BTOR** 12.2.2 applies the PCA results to data headers, and program **BROT** 12.2.4 actually does the rotation to a standard orientation (with respect to the source axis for horizontal component hammer blows).

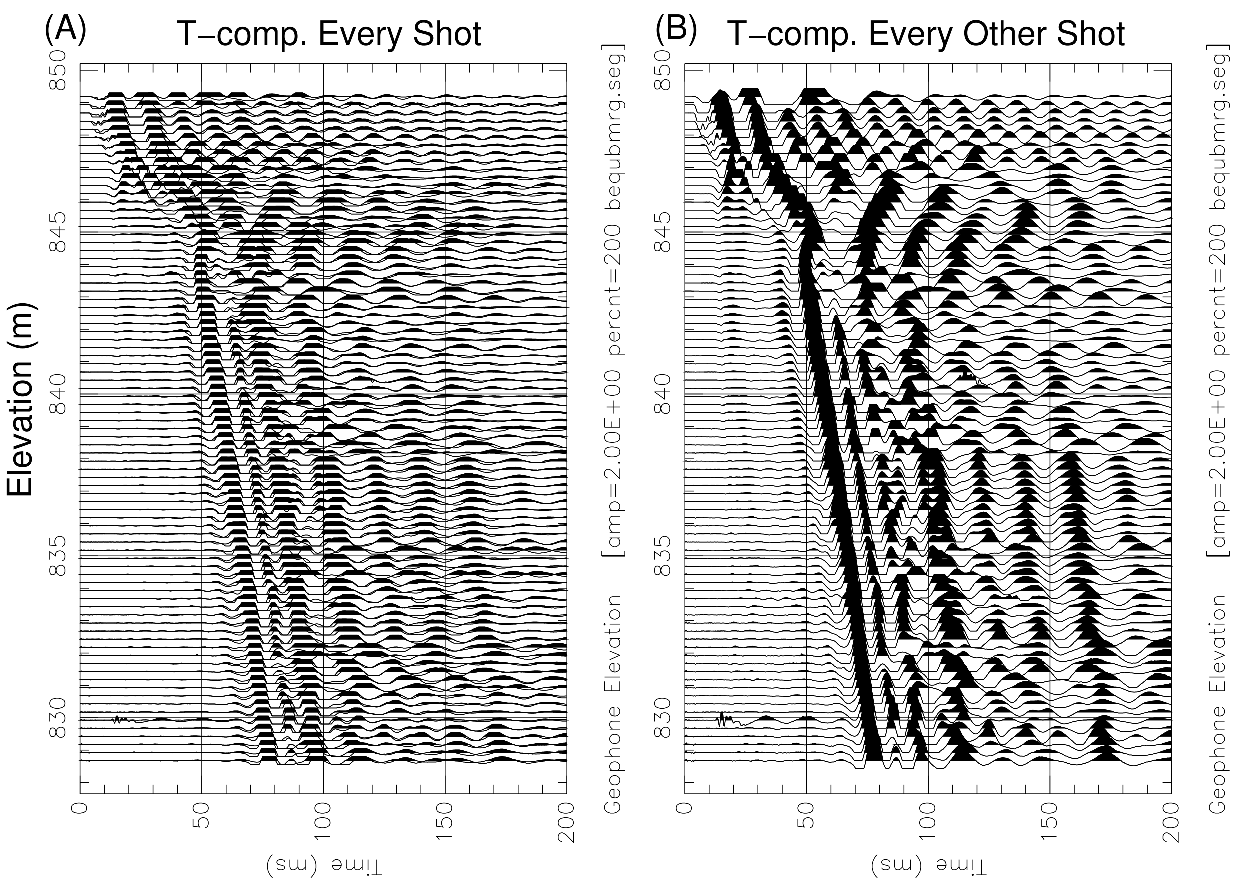


Figure 72: BMRG: A)is plot of all shot efforts (166 traces) and B) is plot of only very other shot (83 traces). NOTE: data are not rotated to a standard orientation, azimuth of T-component drifts up the hole.

11.0.2 BEDT

The command line arguments are:

```
bedt infil tmin tmax ifirst ilast idecm iantia

infil   =input file name to edit
tmin    =minimum time to extract data
tmax    =maximum time to extract data
ifirst  =first trace to extract (<0 pads left)
ilast   =last trace to extract (>ntraces pads right)
idecm   =decimation factor (idecm>0)
        =interpolation factor (idecm<0)
    EXAMPLES:
    idecm=1 keep same sample interval
    idecm=2 output every other sample
    idecm=-2 output samples between originals

iantia  =0 no anti-alias filter for resample
        =1 use anti-alias filter for resample
```

Figure 73 shows an example where a data set is resampled to include only 0 to 200 msec. of data, only first 6 traces, and interpolated to .00025 seconds per sample. The command:

```
bedt c008.seg 0 .2 1 6 -2 0
```

Sinc interpolation does not add any additional frequencies beyond the original Nyquist.

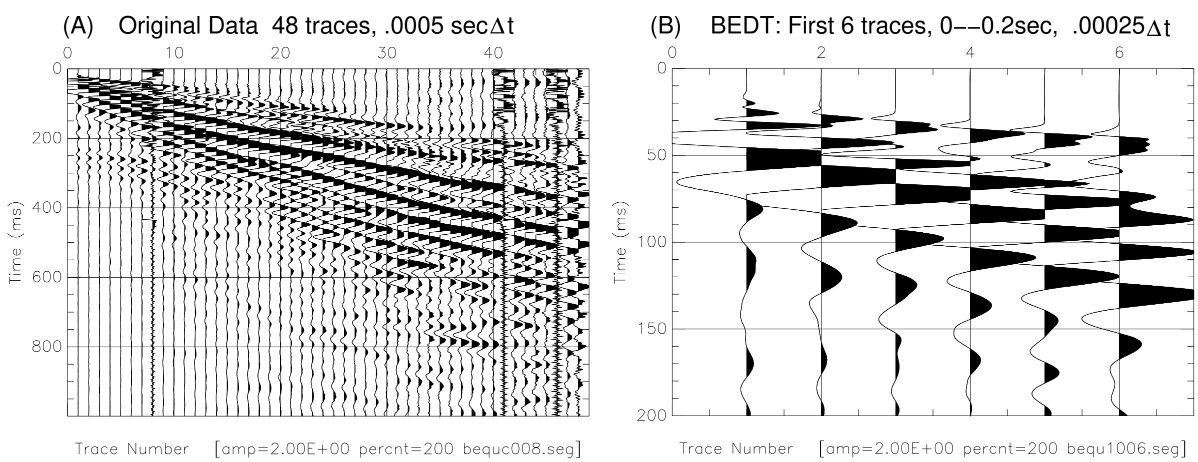


Figure 73: BEDT: (A) Original data, 48 traces, 0-1 seconds, .0005 second sample interval. (B) Edited to only first 6 traces, 0-0.2 seconds, interpolated to .00025 second sample interval.

11.0.3 BRSP

Interpolation only resampling. Done in frequency domain by augmentation with zeros. No new frquencies introduced. **Number of sample increases rapidly!**

```
brsp infile ifact tmax
infile  =input file name
ifact   = resample factor ifact>=1
        = 1 sample interval halved.
        = 2 sample interval 1/4 of original
        = n sample interval 1/(2**n) of original
    (note: trace size increases by 2**ifact)
tmax    =max time of output trace (float) sec
NOTE: This is a radix 2 algorith, so trace may be
 with zeros before computing new number of samples
   BSEGY number of samples header is 16 bit
   Maximum number of samples limited to 32,767
```

11.0.4 BKIL

The command line arguments are:

```
 INDIVIDUAL OPTION-------------
bkil infil iopt1 iopt2 ntrc itr itr ...itr

infile:    =input file name
iopt1:     =option  0=kill 1=zero traces
iopt2:     =specify traces 0=individual 1=by range
ntr        =number of traces to kill or zero
itr......  =trace numbers to kill or zero
 RANGE OPTION------------------
bkil infil iopt1 iopt2 iftr iltr

infile:    =input file name
iopt1:     =option  0=kill 1=zero traces
iopt2:     =specify traces 0=individual 1=by range
iftr       =first trace
iltr       =last trace
```

Example: Zero noisy traces 8, 41, 46 of data in Figure 73 (A).

```
bkil c008.seg 1 0 3 8 41 46
```

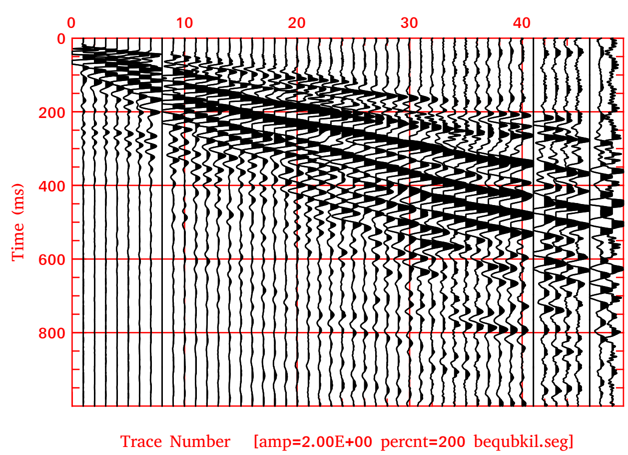


Figure 74: BKIL: Zero noisy traces 8, 41, 46 of data shown in Figure 73 (A).

11.0.5 BEXT

Traces can be extracted by either shot or receiver name in the BSEGY headers. Alternatively, field record number can also be used. This is useful when more than one shot record is in a larger file. The command line arguments are:

```
bext  infile   extsw    value

    infile   =  input file name
    extsw    = extraction switch (1 char)
            s= shot name
            r= receiver name
            f= field record number
    value    = shot or rec name (4 char)
                   or
               field record number (int)

 WARNING: leading blanks are important
 (enclose 4 char string in quotes if on command line)

  Use bdump program to find names of shots or receivers
```

For example, consider extracting the traces with receiver label 30 in a file with two shots.

`bext merged.seg r " 030"`

A partial dump of the headers for merged.seg is:

```
30 |    8|8001  030| 148.53| 1000.50   9938.79   9800.50| 1|0.0514|   40 |  0|  0|
78 |    9|9048  030|  85.15| 1000.50   9938.79   9800.50| 1|0.0386|   40 |  0|  0|
```

This shows that traces 30 and 78 are at receiver name " 030" and have the same (x,y,z) coordinates.

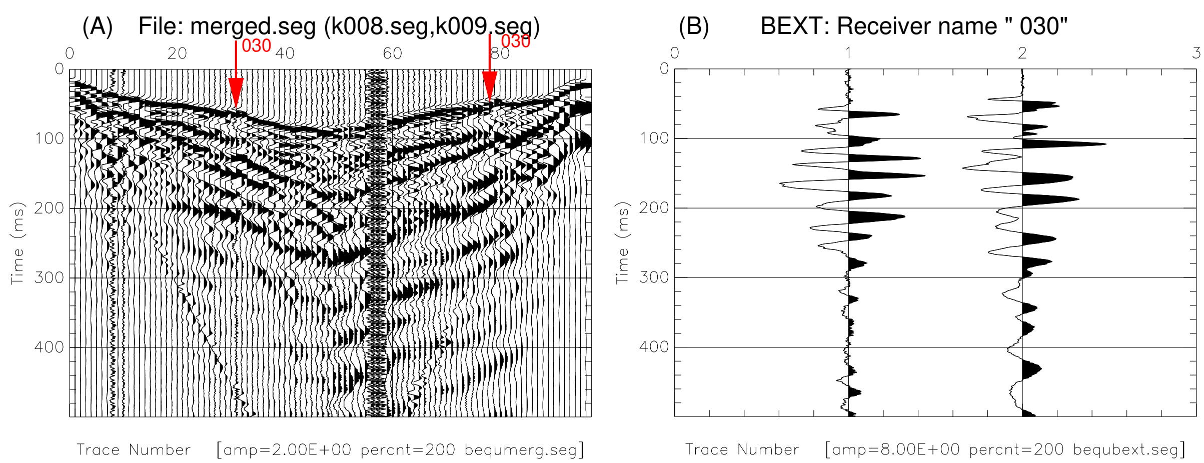


Figure 75: BEXT: Extracted traces from receiver location " 030". In the merged file (A) red arrows show receiver " 030" and these are replotted in (B). Note the receiver name is 4 characters, "blank,zero,three,zero".

11.0.6 BOFF

Some programs need a header value for the shot to receiver offset. This program computes that offset in case it is not in the headers, and then inserts the value in the header of the output file, **boff****.seg**. The only command line argument is the input file name. If the offset BSEGY header value has not been set, it will contain garbage. In the following example, we look at the first trace offset (74 m) in a test file. The commands are:

```
#!/bin/bash
# adds offset to a header value
# This header is not really needed for BSU programs,
# but is useful when converting to Seismic Unix codes
# that require it.
echo " PC linux is little endian"
echo "offset is 4 byte integer in header at hex bytes 0x24 0x25 0x26 0x27"
hexdump -C k007.seg |grep 00000020
echo "00000020               |  |  |  |  "
echo "hexdump k007.seg: starting at hexbyte 0x20, List shows 45 00 00 00"
echo "this is garbage"
# BOFF computes offset header, for trace 1 this is 74 meters
boff k007.seg >/dev/null
hexdump -C boffk007.seg |grep 00000020
echo "00000020               |  |  |  |  "
echo "hexdump boffk007.seg: shows 4a 00 00 00 "
echo "garbage replaced with 0x4a = 74, correct value  "
```

The output when run is:

```
 PC linux is little endian
offset value is 4 byte integer in header at hex bytes 0x24 0x25 0x26 0x27
00000020  00 00 01 00 45 00 00 00  61 90 01 00 ff 85 01 00  |....E...a.......|
00000020               |  |  |  |
hexdump k007.seg: starting at hexbyte 0x20, List shows 45 00 00 00
this is garbage
00000020  00 00 01 00 4a 00 00 00  61 90 01 00 ff 85 01 00  |....J...a.......|
00000020               |  |  |  |
hexdump boffk007.seg: shows 4a 00 00 00
garbage replaced with 0x4a = 74, correct value
```

11.0.7 BWIN

The command line arguments are:

```
 bwin infile tw1 tw2 tw3 tw4

 infile  =input file name
 tw1      =time of start taper on, amp=0
 tw2      =time of taper off,      amp=1
 tw3      =time of start taper off amp=1
 tw4      =time of taper off,      amp=0
```

For example,

```
bwin c008.seg .10 .15 .3 .6
```

Result is shown in Figure 76.

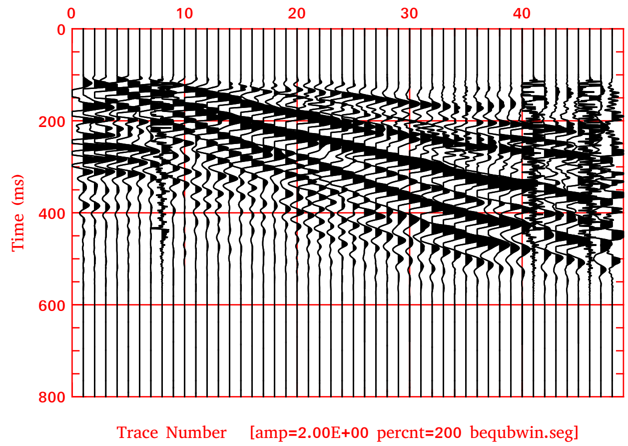


Figure 76: BWIN: Data zeroed outside of the tapered window.

11.0.8 BXOF

Selecting or removing traces can be done with **BKIL**11.0.4 or **BEDT**11.0.2. However, the identification of traces to be operated on is done by trace number. This program permits selection by source to receiver offset. The command line arguments are:

```
bxof  infile   near  far

  infile  = input file name (4char minimum)
  near    = near offset (float) to output
  far     = far  offset (float) to output
```

Offset values are $\pm$ signed to handle split spreads. Negative offsets are on one side of shot, positive on the other.

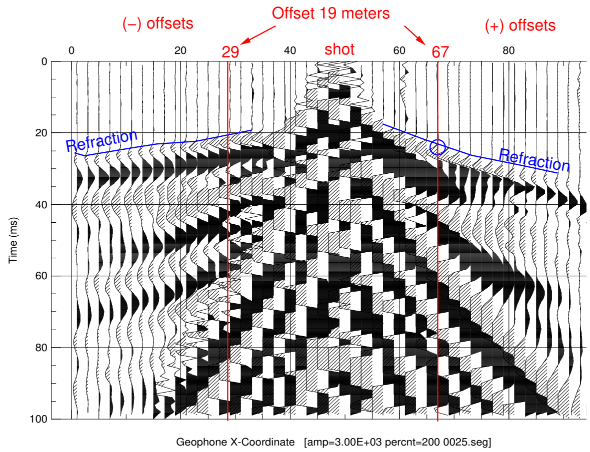
NOTE: !! When using negative offsets, near is the offset with the smallest ***absolute value****. Far is the offset with the largest absolute value. See man page for definition of offsets. To select both sides of a split spread requires 2 runs and concatenation of the two runs into a single file.*

11.0.8.1 Refraction Application One application is a *preliminary step* in understanding the near surface from a refraction point of view. The context would be soil over bedrock or perhaps the water table. Assume the refractor is parallel to the recording surface with a constant velocity in both the soil and the medium supporting the refraction. By selecting an offset beyond the critical distance (ie. always on the head wave arrival), we can plot all the traces at that one offset. This will reveal the refractor structure in time. If an estimate of the overburden velocity and refractor velocity are known, one can convert this time structure of the first arrival into depth.

$$z = \left(t(x) - \frac{x}{V_2}\right) \cdot \frac{V_1}{2cos(\theta)} \tag{16}$$

where z is the depth to the refractor, x is the source to receiver offset, $t(x)$ is the arrival time of the first arrival at offset x, V_1 is the overburden velocity, V_2 is the refracting medium velocity, and $sin(\theta) = \frac{V_1}{V_2}$. The angle θ is known as the critical angle. **This is an initial step to get a sense of the structure in depth. Alternatively, the temporal structure may be due to velocity variation in the soil for a refractor without structure. There are a number of ways to interpret a structure in time.** A later step might be to do delay time analysis (see **BREF** 8.5.6) and Michaels (1995).

An example of the this preliminary approach is shown in Figure 77. For more advanced approaches, see Yilmaz *et al.* (2022), who have provided the data shown here. As a first step, there is general agreement with this figure and the published study.



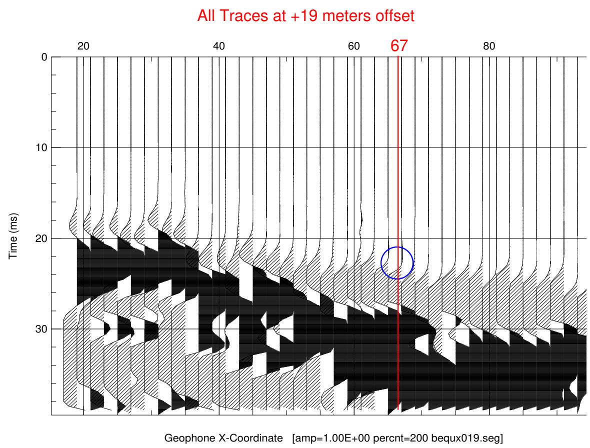


Figure 77: BXOF: A shot gather at station 25 shows a first arrival on the refraction is clear at an offset of +19 meters. Below is a collection of +19 offset traces. The overburden appears to thicken from left to right. The first arrival also seems to exhibit a measure of high frequency loss, possibly due to inelastic attenuation. Data provided by Yilmaz *et al.* (2022).

12 Signal Processing

The focus here is signal processing of BSEGY data, similar to the section on editing (sec 11), but with more emphasis on altering the data by traditional signal processing methods. Programs include:

- **BREV** 12.0.1 Reverse (a) channel order OR (b) polarity
- **BABS** 12.0.2 Rectify the data (absolute value).
- **BSDC** 12.0.3 Compute DC levels of each trace and show.
- **BRDC** 12.0.4 Remove DC levels of each trace.
- **BINT** 12.0.5 Integrate BSEGY data.
- **BSRT** 12.0.6 Sorts data by offset
- **BRPT** 12.0.7 Remove pre-trigger in header and shift data.
- **BDIF** 12.0.8 Differentiate BSEGY data.
- **BEQU** 12.0.9 Trace equalization of amplitude, BSEGY data.

- **BSCL** 12.0.10 Scale data in a profile by a determined or provided factor.
- **BGAR** 12.0.11 Exponential gain recovery by source to receiver offset.
- **BGAZ** 12.0.12 Exponential gain recover by depth gate for down-hole data.
- **BAGC** 12.0.13 Automatic Gain Control (AGC). Choice of single pole exponential envelope or zero-phase box-car envelope.
- **BBAL** 12.0.14 Balance amplitudes between two BSEGY data files such that they both have the same MAV = (MAV1 + MAV2)/2
- **BSTK** 12.0.15 Stacking data in a BSEGY file.
- **BXCR** 12.0.16 Auto- or Cross-correlation computed from a file or between two files.
- **BNOS** 12.0.17 Computes a band limited noise profile to match the aperture of a template case.
- **BTDC** 12.0.18 Decimation in time. Includes zero phase anti-alias filter, -6dB at half the new Nyquist.
- **BAGL** 12.0.19 Compute the angles between seismic traces. Can be used in comparing synthetic and field shot gathers in the context of a waveform inversion. The standard deviation can serve as an object function.
- **BPHZ** 12.0.21 Apply a phase shift to data.
- **BDEC** 12.0.22 Decimate traces in the trace number direction.
- **BSHF** 12.1.1 Static shift BSEGY data by headers or by a file of times, plus a bulk static shift.
- **BSHP** 12.2.1 Wiener Least Square Shaping filter. Can apply to data or an alternate file.
- **BTOR** 12.2.2 Apply PCA analysis (see **GENBHOD** 10.1.9) to headers of all the *.seg files in the **bhod.lst** file.
- **GENBROT** 12.2.3 Generates a bash script which will run the **BROT** program. That program will rotate the horizontal components of the down-hole data to a desired relationship to the source polarization.
- **BROT** 12.2.4 Rotates data based on horizontal component headers or a user supplied value.
- **BFXT** 12.3.1 Compute frequency-distance (FX) transform of a shot gather, or compute an inverse transform of amplitude, phase data sets.
- **BCAR** 12.3.2 Box car filter, both low- and high-pass options. Fast and specified by a moving average filter duration.
- **BFIL** 12.3.3 ARMA filter, Low-pass, Band-pass, or High-pass filters, minimum phase or zero phase, by Bilinear transform.
- **BDCN** 12.3.4 Minimum phase deconvolution.
- **BFTR** 12.3.5 Filter data with *.seg file or namelist file.
- **BWHT** 12.3.6 Non-linear whitening using AGC in overlapping band-pass filters.

12.0.1 BREV

One or more channels may be reversed in **polarity**, OR **channel order** reversed for all channels.

```
brev infil iop1 nflip ch1 ch2 ...

infil:  =name of input file
iop1            0=reverse channel order
       or
                1=reverse data polarity
nflip   =number of channels to be polarity reversed
ch1     =number of channel reverse data polarity
ch2     =number of channel reverse data polarity
ch3     =number of channel reverse data polarity
... ch_nflip.  NOTE: if nflip=number of channels,
     then all channels will be reversed in pol.  )
     (no need to input ch1, ch2,....)
```

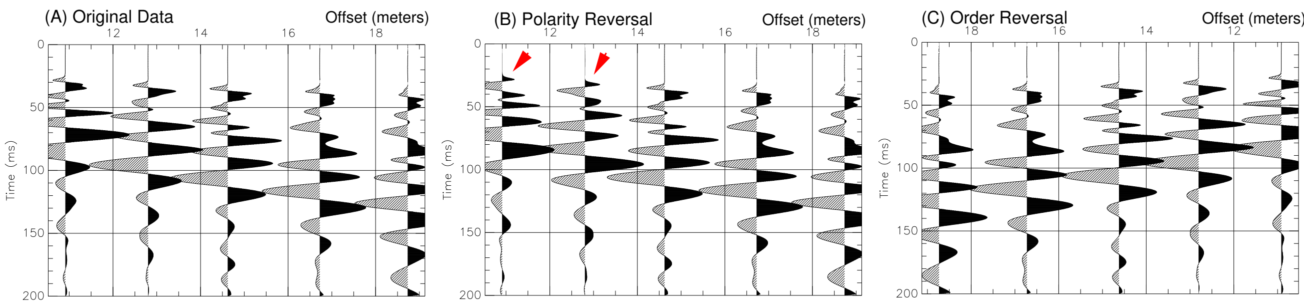


Figure 78: BREV: (A) original data, (B) reverse polarity first 2 channels, (C) reverse channel order. Data plotted by offset.

12.0.2 BABS

Takes the absolute value. The only command line argument is the input file name. Figure 79 shows an example.

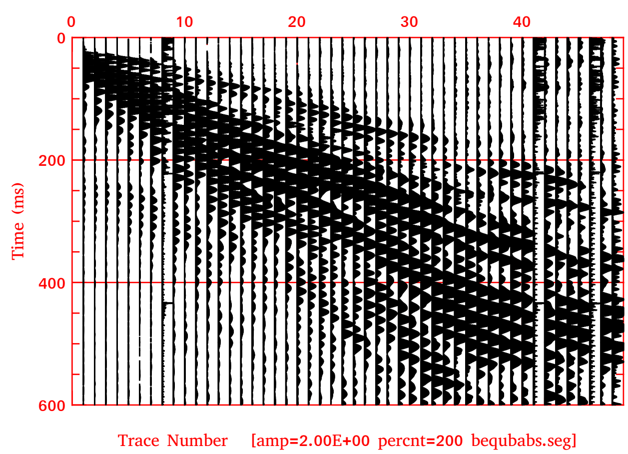


Figure 79: BABS: Rectify data (take absolute value).

12.0.3 BSDC

The only command line argument is the input BSEGY file. The output is to a file, **bsdcxxxx.lst** where xxxx.seg is the input file name. Example of partial output:

```
  Program bsdc
 Input File: c008.seg
 Output File: bsdcc008.lst
  Number of traces=   48
  Parameters:  none
     Trace     DC %(MAV)        MAV
--------------------------------------------------

         1           0.23      0.4653721E+04
         2           0.43      0.4331486E+04
         3           0.13      0.3471209E+04
         4           0.08      0.2875869E+04
         5           0.11      0.2294398E+04
         6           0.06      0.1858094E+04
         7           0.10      0.1950509E+04
         8           1.44      0.5034012E+03
         9           0.11      0.1754470E+04
        10           0.15      0.1428828E+04
        11           0.06      0.1383624E+04
        12           0.12      0.1140977E+04
```

12.0.4 BRDC

BRDC removes a DC level from each trace, or a linear trend. The command line arguments are:

```
brdc infile iswdc  usrdc

infile:  name of input file to remove DC
iswdc:   0=user supplied DC level to remove
         1=determine dc level from each trace
         2=remove linear trend measured from trace
usrdc:   = user supplied value of DC to remove
```

Example of a partial output by **BSDC** 12.0.3 after running this command:
`brdc c008.seg 1`

```
  Program bsdc
 Input File: brdcc008.seg
 Output File: bsdcbrdc.lst
  Number of traces=   48
  Parameters:  none
     Trace     DC %(MAV)        MAV
--------------------------------------------------

         1           0.00      0.4653861E+04
         2           0.00      0.4330840E+04
         3           0.00      0.3471210E+04
         4          -0.00      0.2875866E+04
         5          -0.00      0.2294333E+04
         6           0.00      0.1858083E+04
         7           0.00      0.1950462E+04
         8           0.00      0.5032268E+03
         9          -0.00      0.1754434E+04
        10           0.00      0.1428763E+04
        11          -0.00      0.1383601E+04
        12           0.00      0.1140975E+04
```

12.0.5 BINT

Integration of seismic traces. For example, if traces are in particle velocity, output will be in displacement. If traces are in acceleration units, output will be particle velocity. If data are clipped, integration will reveal a DC level by trace drift. Figure 80 shows a plot after integration with the command:

```
bint c008.seg
```

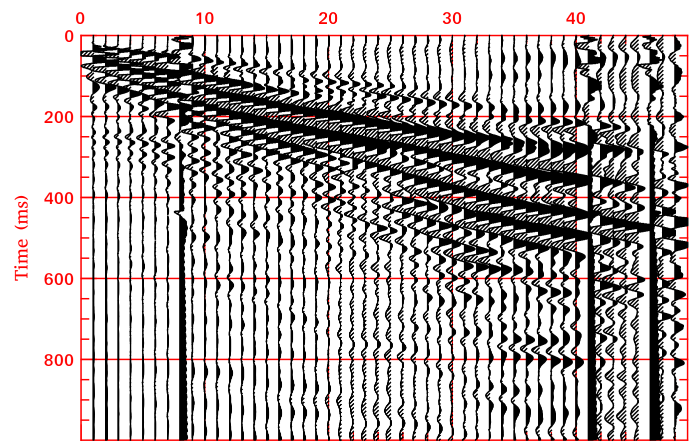


Figure 80: BINT: Integration of traces, plotted trace equalized with BEQU 12.0.9. Negative values grey, positive. DC levels are revealed by drift in either the positive or negative direction.

12.0.6 BSRT

Sorts data traces by offset.

```
bsrt  infile   isort

    infile    =  input file name
    isort   +1=  up by offset
            -1=  down by offset
```

12.0.7 BRPT

Pre-trigger refers to the time before a trigger signal is received. Engineering seismographs can retain data continuously sampled before the trigger signal. This program shifts the data and resets the delaytime header in the shot header section. The only argument is the input file. The **delay time** header value is shown in this sample bdump.lst of a file with a pre-trigger (-10 ms pre-trigger).

```
-----------------------------------------------------------------------------------
  Length =  2000 samples             |  Shot Elevation =      849.2
  Sample Interval =    0.00025 sec.  |  Shot Depth =        0.0
  Delay Time =  -10 msec.            |  Up Hole Time =     0 msec
  Low Cut Filter  =     0 Hz.        |  Shot X-COORD =    9963.19
  High Cut Filter = 1000 Hz.         |  Shot Y-COORD =   10022.41
  Line ID: 00X5                      |  Shot Date (year.moday) = 2001.0417
  Shot Orientation:                  |  Shot Time (hr:min)   = 10:32
  Azimuth=  0 Deg.  Vertical=180 Deg.|  Charge Size (grams)=     0
-----------------------------------------------------------------------------------
TRACE|SHOT| STATION | OFFSET|        RECEIVER          |VERT|1STBRK|K-GAIN|AZI|VER|
  #  |REC.|SHOT  REC|       |  ELEV.  X-COORD   Y-COORD|FOLD|(SEC.)| (dB) |   |   |
```

12.0.8 BDIF

Data are differentiated with BDIF. Thus, if the data are in units of particle velocity, then the output will be in units of acceleration. The code uses a Bi-linear Transform to compute the derivative.

```
bdif infile stab

infile:   =name of input file to be differentiated
stab      =stability factor (move pole in z-plane)
          (from nyquist, Z=-1 to Z=-(1+stab)
      For example: 0.1 <stab< 0.5 can help
      in cases with significant nyquist band noise
```

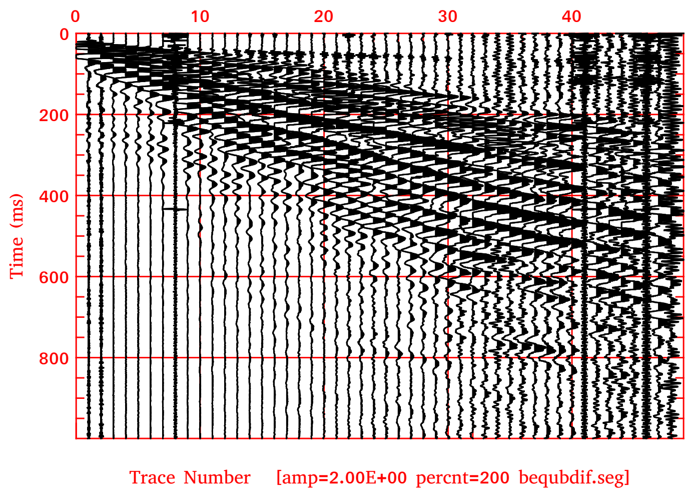


Figure 81: BDIF: Differentiation of BSEGY data, plot trace equalized with BEQU 12.0.9.

12.0.9 BEQU

Data amplitudes are rescaled by the L2 norm or the peak absolute value. Command line arguments are:

```
bequ  infile     tmin   tmax   normsel
    infile   =  input file name
    tmin     =  gate: minimum time (s)
    tmax     =  gate: maximum time (s)
    normsel  =  select normalization
            2=  L2 Norm
            0=  Peak abs(Value)

 NOTE: Default is normsel=2
 No interactive prompt for normsel
 (Must be specified on command line)
```

Figure 82 illustrates how BEQU helps compensate for the wide range of amplitudes in seismic data.

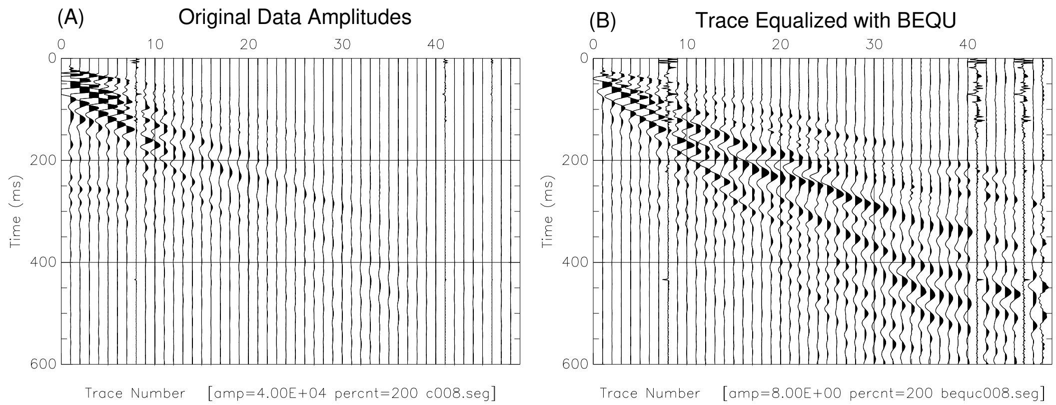


Figure 82: BEQU: (A) original scaling of data, (B) trace equalized with L2 norm. The scale factors for plotting are 40000 for (A) and 8 for (B).

12.0.10 BSCL

The program can scale a data set by a user provided value, 1/L2 norm, (trace,amplitude) pairs in a file, ampfil, or by the maximum absolute value in the file. The scale factor is found by scanning a limited number of traces defined by itr1 and itrn.

```
bscl infil itr1 itrn isw1 [ scaler | ampfil ]

 infil:   =input file name
 itr1:    =starting trace for determination window
 itrn:    =number of traces to include in window

 isw1:   0=user supplied scale factor
         1=scale factor from 1/L2 norm
         2=input file with (trace,ampfactor)
         3=scale factor from Max Abs Value
 scaler:  =user supplied scale factor (ONLY isw1=0)
 ampfil:  file name for option (ONLY isw1=2)
  -  - - - - - - - - - - - - - - - - - - - - - - - -
  | Converting microvolts to m/s particle velocity |
  |                                                |
  |                      VSP Ref.Phone 28Hz Oyo    |
  |     scalar=4.0978E-8 28Hz Oyo SMC 28-720       |
  |     scalar=5.6497E-8 14Hz Oyo Phone            |
  |     scalar=5.0761E-8 10Hz Oyo GS-20DM Phone    |
  |     scalar=3.2787E-8 10Hz Mark L10-A Phone     |
  |     scalar=3.2034E-8 08Hz Mark L10-A Phone     |
  |                                                |
  |     strain=(part.vel.)/(wave phase vel.)       |
  -  - - - - - - - - - - - - - - - - - - - - - - - -
```

Example: Use the first 5 traces closest to the source and determine the maximum absolute value, compute a scale factor so that sample with the MAV has a value of unity (1), then apply to all the traces.

```
bscl c008.seg 1 5 3
```

Figure 83 shows the first 10 traces for clarity.

```
bplt bsclc008.seg 4 0 0 1 10 0 .6 1 1 200
```

```
 Peak Absolute Value=       192440.8750000000
----------------------------------------------------------------
   Maximum Value= 0.1924E+06     Trace #=    1
                                 Sample #=  101
   Minimum Value= -.1766E+06     Trace #=    1
                                 Sample #=   80
----------------------------------------------------------------
   Scale Factor=  0.5196401E-05
```

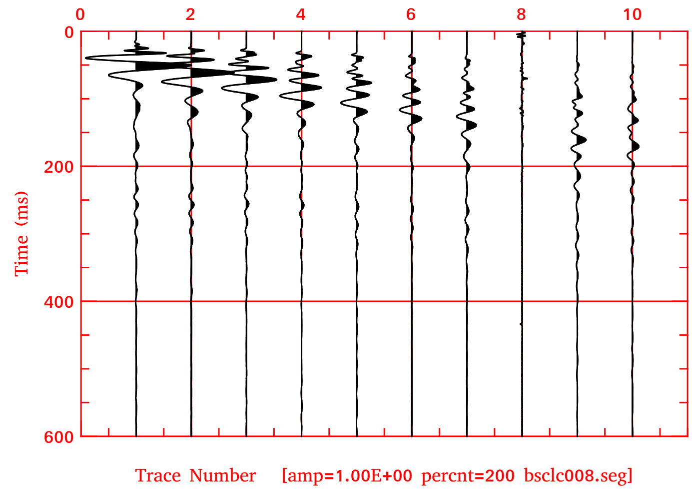


Figure 83: BSCL: Scale all traces by the maximum absolute value (MAV) found in the first 5 traces.

12.0.11 BGAR

Computes an amplitude decay envelope over a user provided **RANGE** interval. The envelope is corrected for spherical divergence, converted to decibels, and a linear fit performed. The user may then apply the recommended gain correction, or over ride it with their own choice. The spherical divergence and exponential gain corrections are applied to the entire data set, (not just the interval of analysis). The data are not filtered before hand, so the decay measurements are a single result for the entire available bandwidth. If you want to measure inelastic decay as a function of frequency, then use program BAMP 8.3.6. This program simply provides a broad-band view of amplitude decay as sensed by the summed absolute value amplitude of each trace on a survey. A PostScript plot of the linear regression is output (requires PLPLOT package be installed)

```
   bgar  infile   rmin   rmax   dbu

     infile   =  input file name
     rmin     =  min. range design gate
     rmax     =  max. range design gate
     dbu      =  gain correction to apply (dB/m)
NOTE:
    No prompt for dbu until after gain assessment.
    However, you may specify dbu on the command line
    if you already have a value you wish to use.
```

For example, see Figures 84 and 85 which show the result of the following command:

```
bgar c008.seg 6.  100.  .03
```

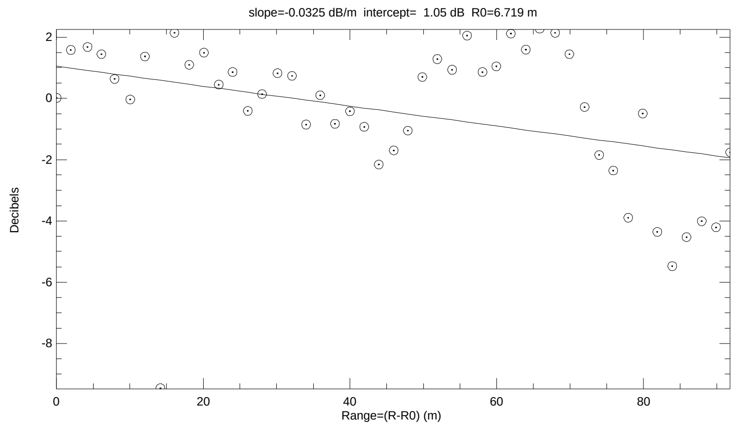


Figure 84: BGAR: Broadband scale by spherical divergence and exponential decay. Range from 6 to 100 meters.

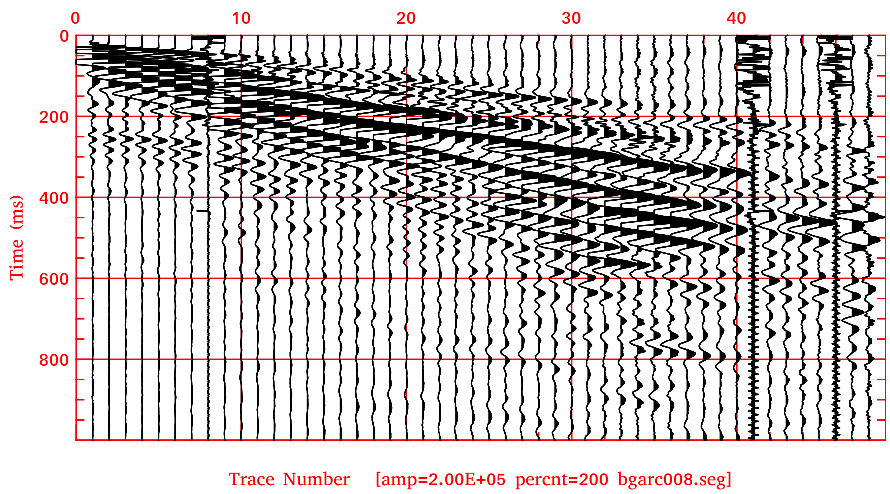


Figure 85: BGAR: Broadband scale by spherical divergence and exponential decay. Specified .03 dB/m for inelastic decay.

12.0.12 BGAZ

Computes an amplitude decay envelope over a user provided **depth interval**. The envelope is corrected for spherical divergence, converted to decibels, and a linear fit performed. The user may then apply the recommended gain correction, or over ride it with their own choice. The spherical divergence and exponential gain corrections are applied to the entire data set, (not just the interval of analysis). The data are not filtered before hand, so the decay measurements are a single result for the entire available bandwidth. If you want to measure inelastic decay as a function of frequency, then use program BAMP 8.3.6. This program simply provides a broad-band view of amplitude decay as sensed by the peak-peak amplitude of the direct arrival on a down-hole survey. A PostScript plot of the linear regression is output (requires PLPLOT package be installed).

```
  bgaz  infile   zmin   zmax   dbu

     infile   =  input file name
     zmin     =  min. depth design gate
     zmax     =  max. depth design gate
     dbu      =  gain correction to apply (dB/m)
NOTE:
    No prompt for dbu until after gain assessment.
    However, you may specify dbu on the command line
    if you already have a value you wish to use.
```

Example: See Figures 86 and 87 which show the result of the following command.

```
bgaz twave.seg 2.  20.  1.43
```

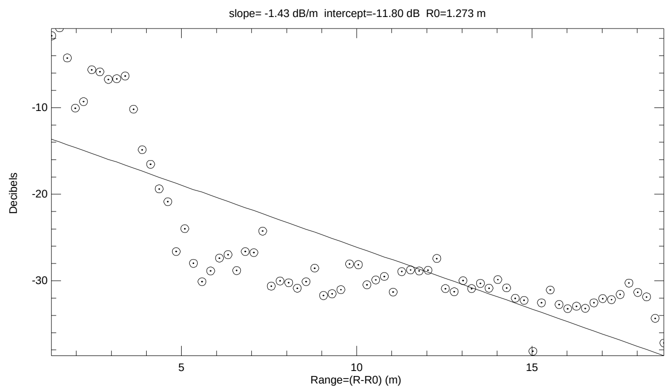


Figure 86: BGAZ: Broadband scale by spherical divergence and exponential decay. Depth range from 2 to 20 meters.

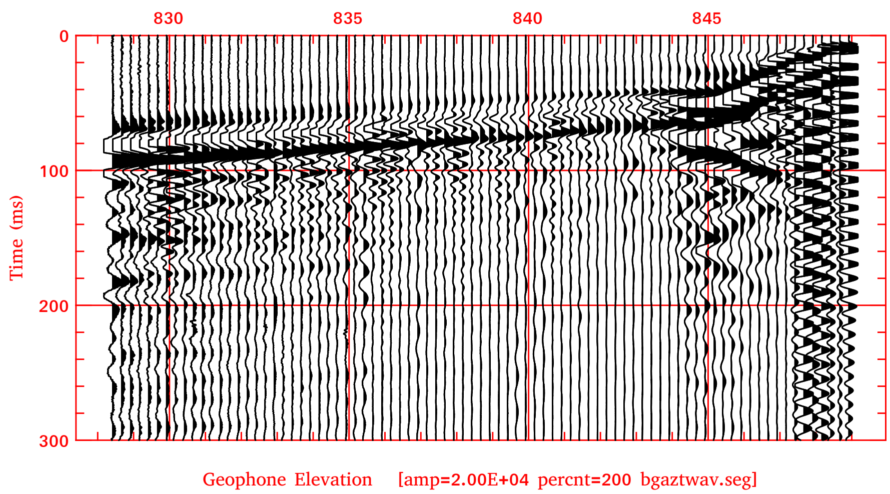


Figure 87: BGAZ: Broadband scale by spherical divergence and exponential decay. Specified 1.43 dB/m for inelastic decay. Elevations are down the bore-hole.

12.0.13 BAGC

Performs Automatic Gain recovery (both in space and time). One may choose to smooth the energy envelope with either a zero phase box car operator (which then gives an anticipatory component to the gain recovery), or one may choose to use the minimum phase (single pole on the real axis in the z-plane) filter. Output can be either the gain recovered data, or the smoothed gain recovery envelopes, sqrt(smoothed energy). First sample set to zero to avoid noise spike.

```
bagc  infile twide itype
   infile    =  input file name
   twide     =  width moving energy window (s)
   itype     =  envelope smoother and output
           0=  ARMA one-pole exp. decay
           1=  zero-phase box car
           2=  output ARMA envelope
           3=  output BOXCAR envelope
```

Example: Zero-phase box car, Figure 88.

```
bagc c008.seg .3 1
```

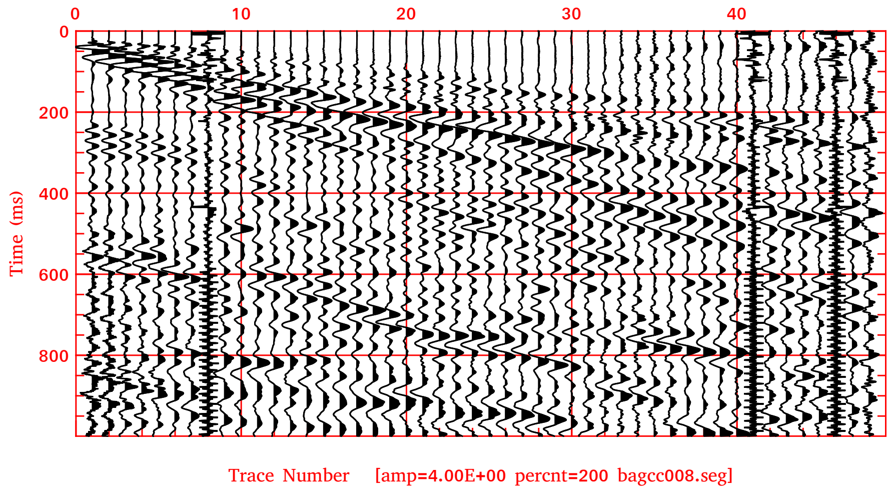


Figure 88: BAGC: Zero-phase boxcar 0.3 seconds.

Example: Single pole, Figure 89.

```
bagc c008.seg .04 0
```

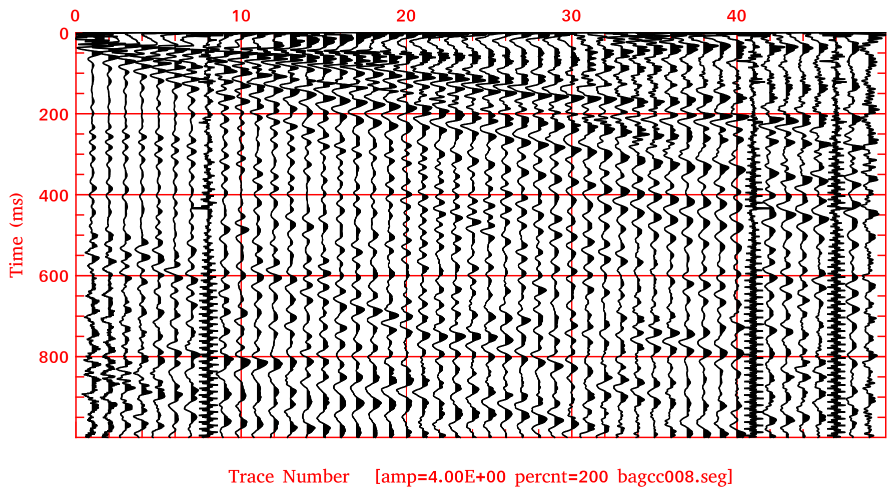


Figure 89: BAGC: Single pole AGC envelope .04 seconds.

12.0.14 BBAL

Balances two BSEGY files so that their Mean Absolute Values (MAV) are the same. Can be executed in either a trace or profile balancing mode.

```
 bbal    infile1   infile2   iopt

    infile1   =  input file_1 name
    infile2   =  input file_2 name
       iopt  0=  profile mode
       iopt  1=  trace   mode
```

An example that illustrates the concept is taken from down-hole data. There are two 3 component geophones, one at a depth of 19.39 meters, the other is a reference phone fixed at the surface elevation of the hammer.

```
--------------------------------------------------------------------------------
  Length =  2500 samples              |  Shot Elevation =      820.0
  Sample Interval =    0.00020 sec.   |  Shot Depth =        0.0
  Delay Time =    0 msec.             |  Up Hole Time =     0 msec
  Low Cut Filter  =    4 Hz.          |  Shot X-COORD =    9897.04
  High Cut Filter = 1000 Hz.          |  Shot Y-COORD =   10066.29
  Line ID: 18A_                       |  Shot Date (year.moday) = 1996.0604
  Shot Orientation:                   |  Shot Time (hr:min)   = 10:52
  Azimuth= 90 Deg.  Vertical= 90 Deg.|  Charge Size (grams)=     0
--------------------------------------------------------------------------------
TRACE|SHOT| STATION | OFFSET|         RECEIVER          |VERT|1STBRK|K-GAIN|AZI|VER|
  #  |REC.|SHOT  REC|       |  ELEV.  X-COORD   Y-COORD|FOLD|(SEC.)| (dB) |   |   |
-----|----|---------|-------|---------------------------|--|------|------|---|---|
   1 |  10| 002  517|  19.39|  800.72    9897.04  10067.79|10|0.0000|   60 |  0|  0|
   2 |  10| 002  518|  19.39|  800.72    9897.04  10067.79|10|0.0000|   60 |189| 90|
   3 |  10| 002  519|  19.39|  800.72    9897.04  10067.79|10|0.0000|   60 |279| 90|
   4 |  10| 002  520|   1.59|  819.96    9897.04  10064.70|10|0.0000|   20 |  0|  0|
   5 |  10| 002  521|   1.59|  819.96    9897.04  10064.70|10|0.0000|   20 |  0| 90|
   6 |  10| 002  522|   1.59|  819.96    9897.04  10064.70|10|0.0000|   20 |270| 90|
```

The first 3 traces are separated to a new file, as are the last 3 traces. In this example, we then do a trace balance between the down-hole and the reference phone traces and recombine as in Figure 90. The commands are:

```
bplt c010.seg 2 0 0 1 7 0 .25 1 2e4 200
mv bplt.fig c010.fig
bedt c010.seg 0 .5 1 3 1 0
mv bedtc010.seg down.seg
bedt c010.seg 0 .5 4 6 1 0
mv bedtc010.seg refn.seg
bbal down.seg refn.seg 1
cp bbaldown.seg BBAL.seg
cat bbalrefn.seg >> BBAL.seg
bplt BBAL.seg 2 0 0 1 7 0 .25 1 2e4 200
mv bplt.fig BBAL.fig
```

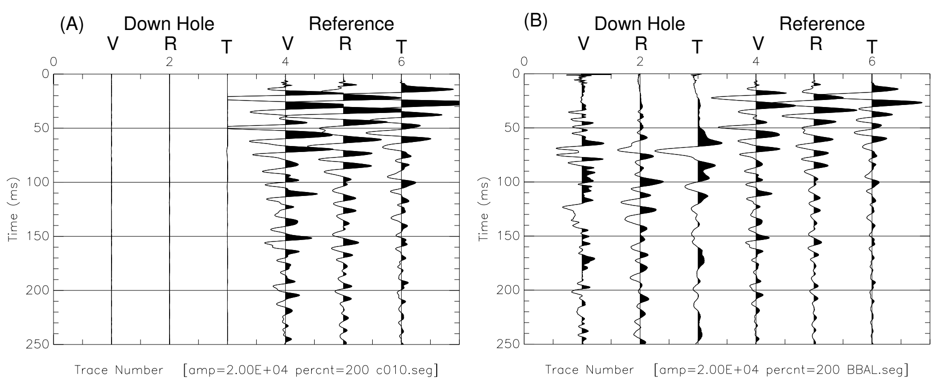


Figure 90: BBAL: (A) Original data (down-hole barely visible) (B) data after splitting the data into two files, running BBAL, then combining into a second file. This figure was created using the XFIG program and the *.fig output type in BPLT 6.0.2

12.0.15 BSTK

Stacks all the traces in a gather, outputs the same trace repeatedly, number in = number out. Command line argument is the input file name. Figure 91 (A) shows the reference phone recording for each source effort of a down-hole survey. While repeatable, there is some variation as the source compacts the ground. (B) shows the average of all the source efforts estimated by the sum of all the traces in (A). The summing is called a stack.

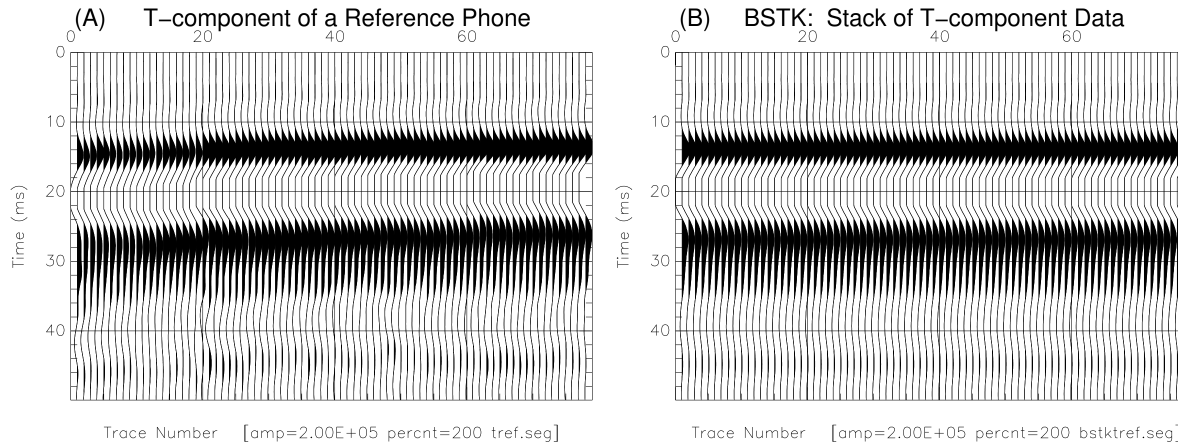


Figure 91: BSTK: (A) Original data T-component data (B) Stack of the T-component data (all traces replicas of the stack result). An application might be creating a target for wavelet processing (BSHP 12.2.1).

12.0.16 BXCR

The command line arguments are:

```
bxcr infile1 infile2 t1 t2 tlagmx

infile1: name of input file #1
infile2: name of input file #2
t1:     =start time of cross correlation gate (sec.)
t2:     =end time of cross correlation gate (sec.)
tlagmx: =maximum cross correlation lag time (sec.)
```

If the second input file is the same as the first, the result will be an auto correlation. If the input files are different, then the result is a cross-correlation between the two, and the order of the file names is important when looking at relative time shifts.

Example: Auto correlation is computed for data shown in Figure 82.

```
bxcr c008.seg c008.seg 0 1.2 .25
```

Both the auto correlation and the stack of the auto correlation are shown in Figure 92. The stack presents an average of the auto correlations at each offset. In (A) of Figure 92 we see that the near offset data (on left of the figure) present a broader bandwidth than at the further offsets. The spectral computation of the stack will provide an average spectrum, while spectral computations of the simple auto correlation in (A) will show the change in bandwidth with offset.

The zero lag sample is at the middle time. In this case, sample time of 125 msec. corresponds to zero lag (125 msec is 1/2 of 250 msec). The command above took all 1.2 seconds of data and computed the auto correlation out to $\pm$125 msec.

To compute an all pole spectrum, see **OCTAVE YULEWALKER** in section 6.0.7.

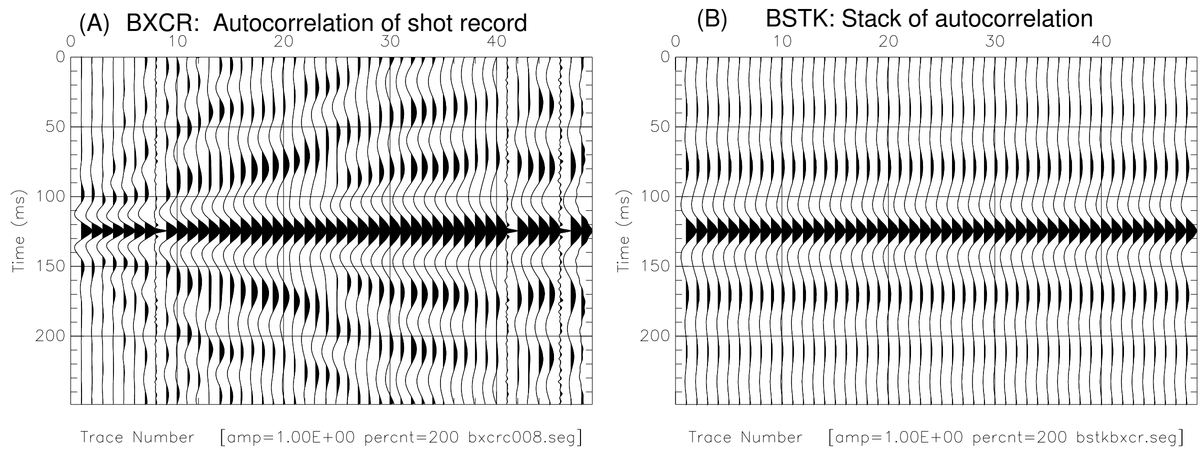


Figure 92: BXCR: (A) Auto correlation of data shown in Figure 82 (B) Stack of the auto correlation (all traces replicas of the stack result).

12.0.17 BNOS

Computes band-limited random noise which copies headers from a template *.seg file. Noise can be added back into the template file with **BSUM** 12.1.4.

```
bnos  infil  seed  F_Low  F_High  F_Roll

infil   =name of a file to match: npts,ntraces...
seed    =seed for random noise
         (positive number less than 1)
F_Low   =low-cut frequency for random noise, Hz
F_High  =high-cut frequency for random noise, Hz
F_Roll  =roll off in Hz (trapezoidal pass band)
```

Example:

```
bnos k007.seg .91827364 10.0 100.  5.0
```

See Figure 93.

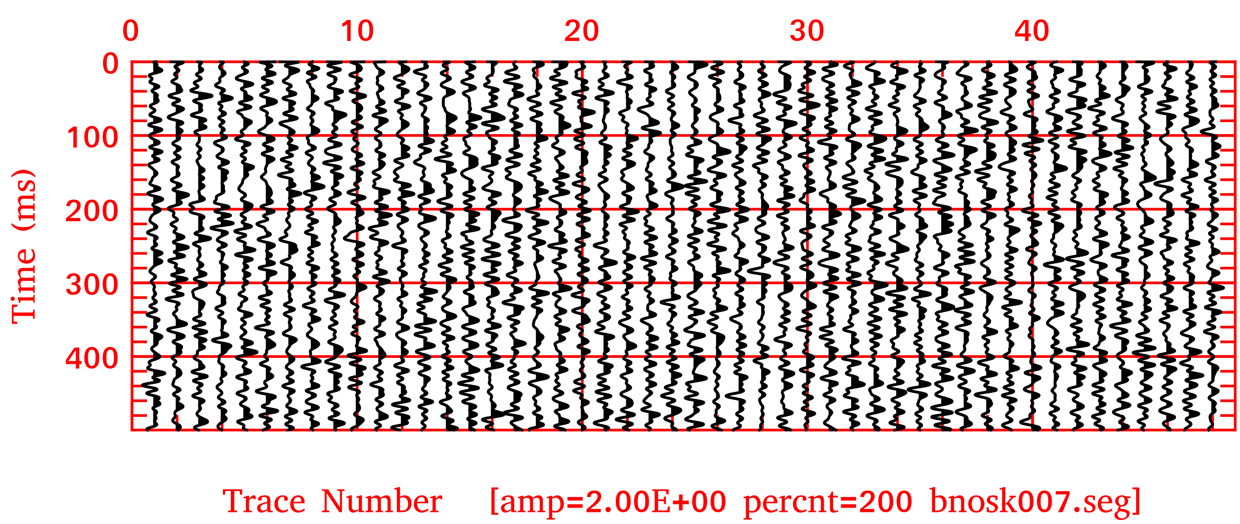


Figure 93: BNOS: Band-limited noise, 10-100 Hz.

12.0.18 BTDC

Simple resampling of data along time axis provided by **BTDC**. A zero phase anti-alias filter is applied before re-sample. The maximum sample interval is due to a limit on the integer sample interval header entry (microseconds)

```
btdc  infile  idecm

   infile  = input file name (4char minimum)

   idecm = decimation factor
         = 2 output every other sample
         = 3 output every third sample
NOTE: idecm can not produce a sample interval > .065 seconds
```

EXAMPLE: Impulse resampled every other sample

```
btdc impulse.seg 2
Sample Interval = .0002 seconds (Nyquist 2500 Hz)
     99   0.0000000E+00 |*                                                  |
    100   0.0000000E+00 |*                                                  |
    101   0.1000000E+01 |.**************************************************|
    102   0.0000000E+00 |*                                                  |
    103   0.0000000E+00 |*                                                  |
Sample Interval =.0004 seconds  (-6 dB at 625 Hz, Nyquist 1250 Hz)
     45  -0.9707062E-03 |    *                                              |
     46   0.3460892E-02 |    .*                                             |
     47   0.1248199E-02 |    *                                              |
     48  -0.2454621E-01 |****.                                              |
     49   0.9763651E-03 |    *                                              |
     50   0.1463085E+00 |    .***************************                   |
     51   0.2471394E+00 |    .**********************************************|
     52   0.1463086E+00 |    .***************************                   |
     53   0.9763851E-03 |    *                                              |
     54  -0.2454621E-01 |****.                                              |
     55   0.1248197E-02 |    *                                              |
     56   0.3460893E-02 |    .*                                             |
     57  -0.9707051E-03 |    *                                              |
     58  -0.1885937E-03 |    *                                              |
     59   0.1843894E-03 |    *                                              |
```

12.0.19 BAGL

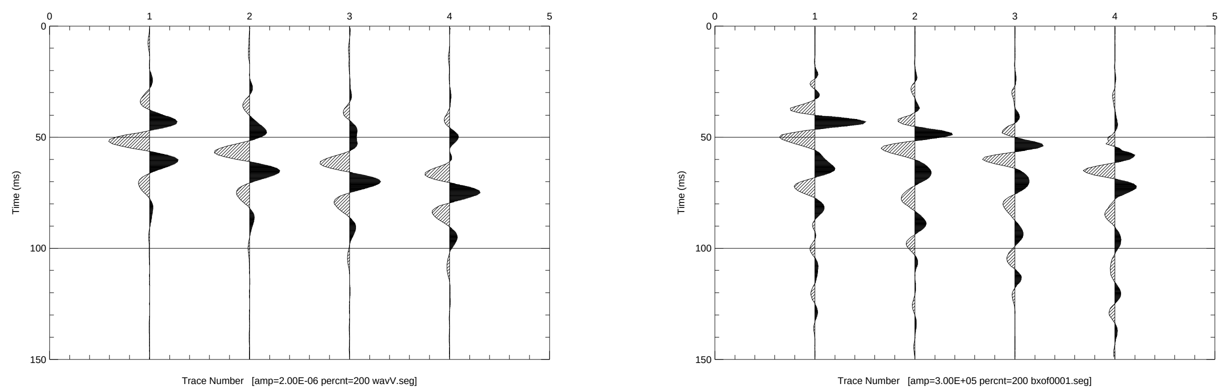
In wavform inversion, comparing the degree of match between an observed and computed shot gather is required. Program **BAGL** computes the angle between two shot gathers by using an innerproduct method. The angle is the arc-cosine of the innerproduct divided by the product of norms.

$$\Phi_k = cos^-1\left(\frac{{x_k}^T y_k}{\sqrt{{x_k}^T x_k}\cdot\sqrt{{y_k}^T y_k}}\right) \tag{17}$$

$$\overline{\Phi} = \frac{1}{N}\Sigma\left(\Phi_k\right) \qquad STD(\Phi) = \sqrt{\frac{1}{N-1}\Sigma\left(\Phi_k - \overline{\Phi}\right)^2} \tag{18}$$

where x_k and y_k are the k-th seismic trace pairs from the two shot gathers. Each pair of traces results in the k-th angle Φ_k. If traces are the same, then the angle will be zero. If they are orthogonal, the angle returned will be 90 degrees. If one data set is the negative of the other, the angle will be 180 degrees. **NOTE:** The number of samples and the sample interval must be identical between the two shot gathers.

The average and standard deviation about the mean are computed. In waveform inversions, the standard deviation could be used in formulating an objective function. That is, if all the angles are the same, then the two shot gathers only differ by a wavelet. The more this is true, the smaller the standard deviation will be. However, if the arrivals in the two shot gathers do not overlap significantly, then a false impression of a match would result (see **BOBF** 12.0.20). The code generates a gnuplot (graph.gp) and also runs gnuplot with a system command.



(a) Synthetic (b) Field

Figure 94: Two shot gathers to compare, $\overline{\Phi} = 43.3^o$, and $STD(\Phi) = 4.12^o$.

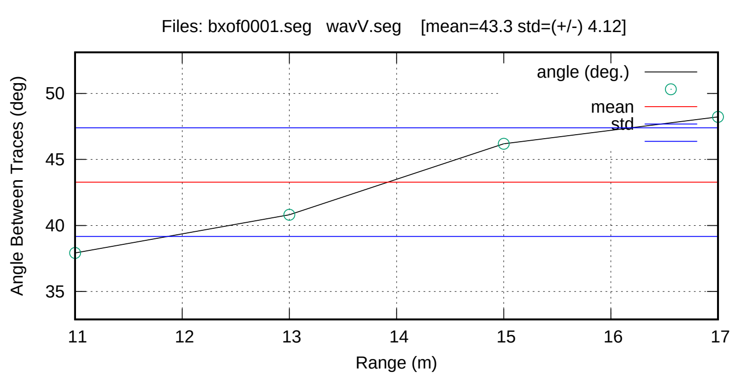


Figure 95: BAGL output figure, graph.gp

12.0.20 BOBF

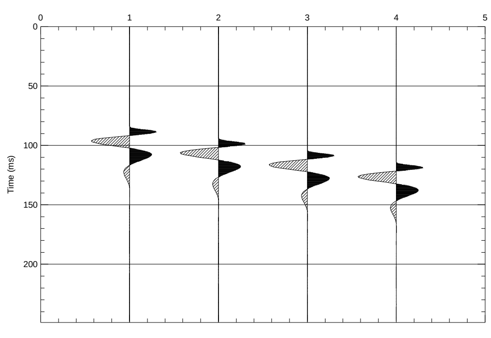
This code employs a mix of the angle between data code **BAGL** (12.0.19) with a measure of overlap. Using just BAGL in waveform inversions only will work if the data always are close to a match. If synthetic and field data are very dissimilar, and do not overlap, then the angle will be 90 degrees (ie. orthogonal). Further, there will be little variation in the 90 degree angle under non-overlap conditions when computing the standard deviation about the mean angle for all traces in the gather.

The objective function is designed to reject non-overlapping arrivals by adding a measure of non-overlap to the angle computation (equation 17, Φ_k), increasing the value of the objective function and any missleading apparent match between field and synthetic data. For each trace pair, the largest absolute value amplitude arrival is found. The corresponding sample number is noted. Let these be $J_{x^{max}}$ and $J_{y^{max}}$ for shot gathers X and Y. The absolute value of the difference in these sample numbers, α_k, are then added to the angle Φ_k to produce the value Ψ_k as below. The objective function value which measures the degree to which the shot gathers match is found by computing the standard deviation about the mean, $\overline{\Psi}$.

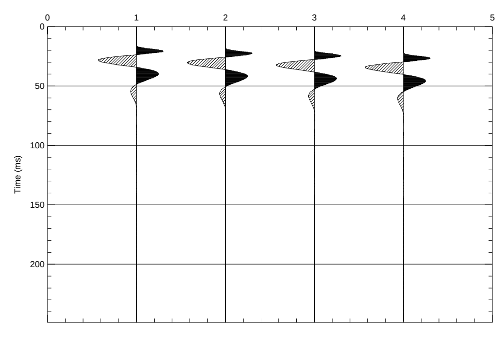
$$\alpha_k = \left| J_{x^{max}} - J_{y^{max}} \right| \tag{19}$$

$$\overline{\Psi} = \frac{1}{N}\Sigma\left(\Phi_k + \alpha_k\right) \qquad OBF = \sqrt{\frac{1}{N-1}\Sigma\left(\Psi_k - \overline{\Psi}\right)^2} \tag{20}$$

If the arrivals overlap a lot, then α_k values will be near zero. But if the data being compared are very different, then the α_k values will be very different from zero, preventing a false minima in the objective function as below where $OBF = 80$ while $STD(\Phi) = 0$.



(a) Synthetic



(b) Field

Figure 96: Two shot gathers to compare with no overlap. $STD(\Phi_k) = 0$, $\overline{\Phi} = 90^o$, and $OBF = 80$.

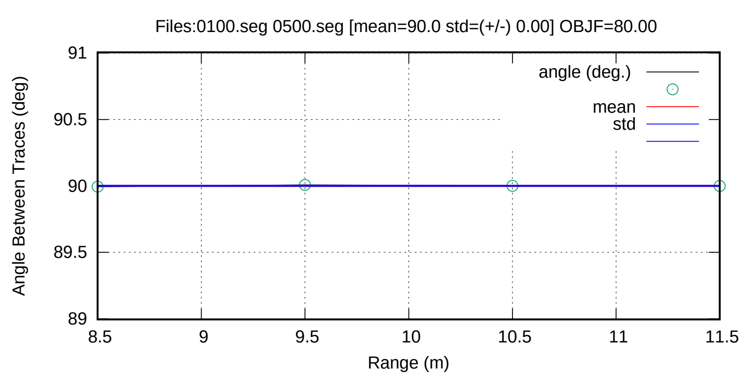


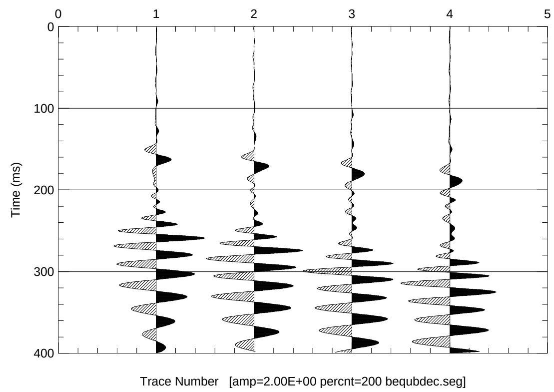
Figure 97: BOBF NOTE: objective function value = 80 despite a standard deviation of zero.

12.0.21 BPHZ

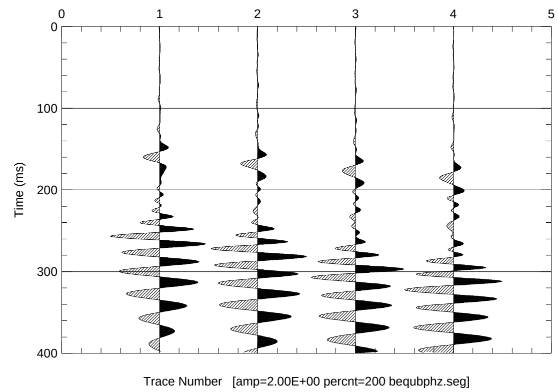
This program applies a constant phase shift to a data set. For example, if the the data are shifted 180 degrees, the polarity is reversed. Shifts may be applied from zero to 360 degrees. However, if the unshifted and shifted data are compared using either program **BAGL, 12.0.19** or **BOBF 12.0.20**, the result will be in the range of zero to 180 degrees. In other words, a 270 degree shift will be interpreted by BAGL as 90 degrees.

```
bphz   infile angle

infile  = input file name
angle   = phase angle in degrees
```



(a) Original Data



(b) BPHZ shift of 135 degrees.

Figure 98: Comparison of original and shifted data.

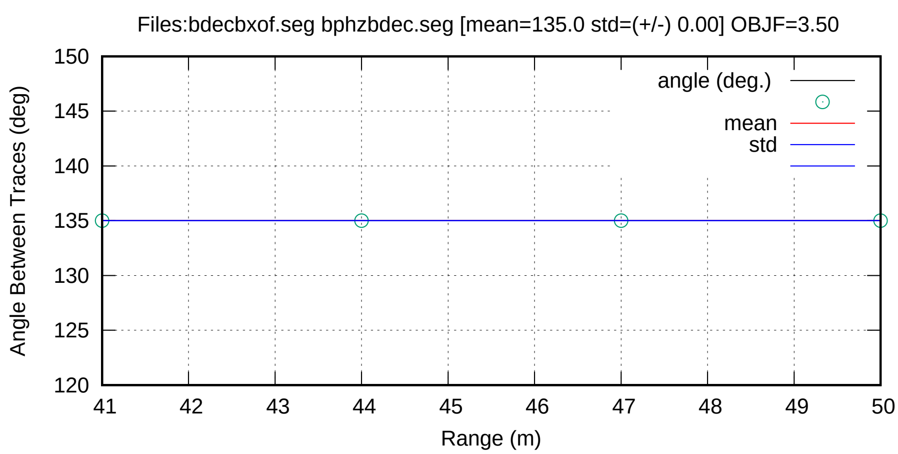


Figure 99: BOBF analysis of data (figure 98(a)) and after BPHZ 135^o phase shift (figure 98(b)). Note that the standard deviation on the angle is zero, but the maximum amplitude moves to a different sample after the shift, resulting in an objective function value of OBF=3.50 (see equation 20)

12.0.22 BDEC

Decimate traces given a start and increment. No anti-alias filter is applied in the space direction. Running **BMIX** 12.1.3 before **BDEC** can be used for anti-alias protection.

```
bdec  infile  jfirst  factor

   infile  = input file name (4char minimum)
   jfirst  = start output with this trace first
   factor = decimation factor (int)
 Example: factor = 2, output every other trace
```

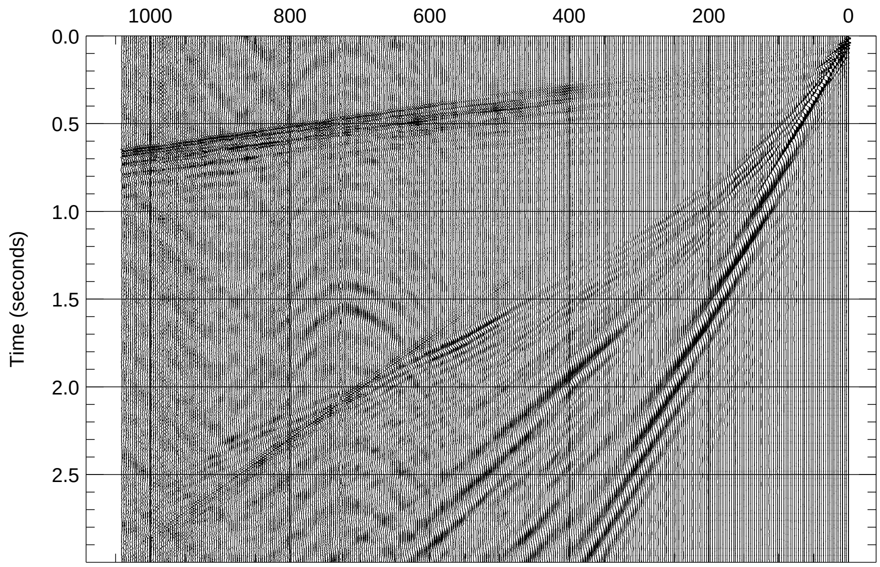


Figure 100: Original data (offset in meters).

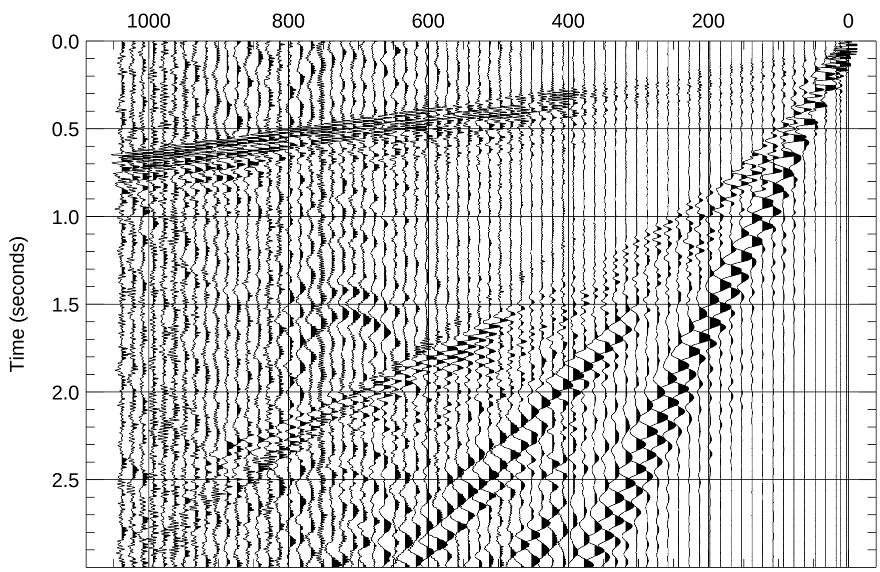


Figure 101: Decimated data, every 5th trace, (offset in meters).

12.1 Down-hole VSP Processing for Reflections

The following signal processing applications are included as a set to illustrate how a Vertical Seismic Profile (VSP) may be processed for reflections. The steps are:

1. **BGAZ** 12.0.12 gain correction:
   `bgaz twave.seg 5 20 0.6`

2. **BSHF** 12.1.1 flatten data on direct arrivals + 20 ms.
   `mv bgaztwav.seg 00X5.seg`
   `bshf 00X5.seg  0  1  .02`

3. **BMED** 12.1.2 median mix of flattened data to extract down going wave.
   `bmed bshf00X5.seg 15`

4. **BSUM** 12.1.4 subtract direct wave from total wavefield.
   `bsum bshf00X5 bmedbshf.seg -1.0`

5. **BSHF** 12.1.1 restore to 1-way time.
   `bshf bsumbshf 0  0  -.02`

6. **BSHF** 12.1.1 shift to 2-way time to flatten reflections (adding the direct arrival times again)
   `bshf bshfbsum 0 0 0.`

12.1.1 BSHF

All samples move in time by a constant shift. The shift in seconds is either in the header for first break pick, or in a separate file. See Figure 102 for the example.

```
BSHF infile ipic ishf tshift picfil

infile:   =input file name
ipic:    1=static shift data by picks pick file
         0=static shift data by picks in headers
ishf:    0=add static shifts
         1=subtract static shifts
tshift:   =bulk static to add to picks
picfil:  =file name with picks (for ipic=0)
```

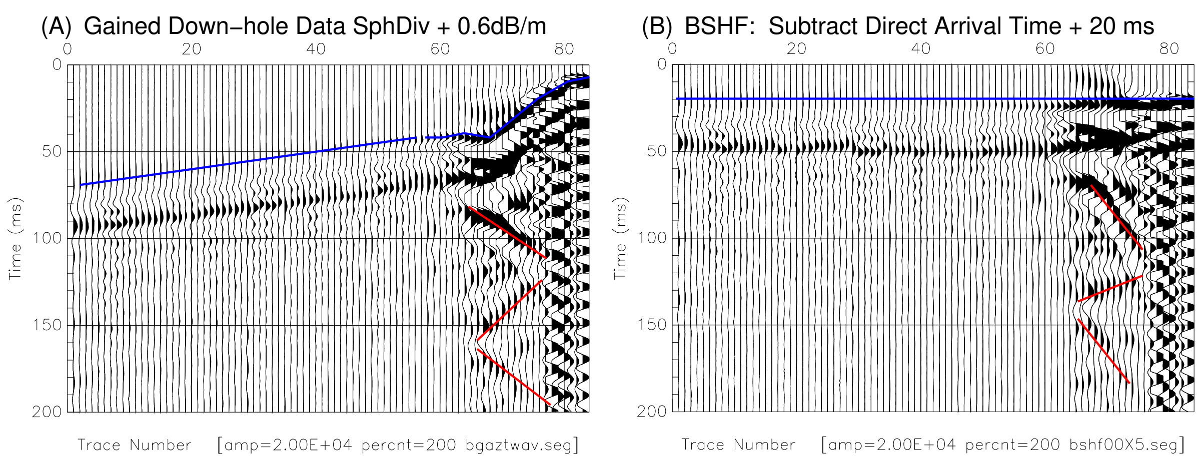


Figure 102: BGAZ: (A) Gained down-hole data, blue=direct wave, red=reverberating reflections (B) BSHF: Data flattened on down-going wave.

12.1.2 BMED

A median mix is usually preferred since it is less likely to smear large amplitude, often noise spikes. As an alternative, one could use the mean mix, **BMIX** 12.1.3 program. One should use an odd number of traces in the mix parameter.

```
bmed infile mix

infile  =input file name
mix     =mix width <21
```

12.1.3 BMIX

Mean mix, only difference between median and mean mix in wave field separation is which value (mean or median) is used in the moving average operator. The mean is not used in this example.

12.1.4 BSUM

The median mix is an estimate of the down-going wave in this example. When subtracted from the total wave-field data, the result should be up-going waves. See Figure 103 (B) .

```
 bsum    infile1   infile2   scalef
    infile1  =  first  input file name
    infile2  =  second input file name
    scalef   =  scale factor
    output = input1 + scalef*input2
```

The up-going wave field estimate is then shifted back to 1-way time. Then the data are shifted again by the direct wave down-going times (this time without any bulk shift, to adjust the data to 2-way time. The reflections should be flattened in 2-way time (see Figure 104).

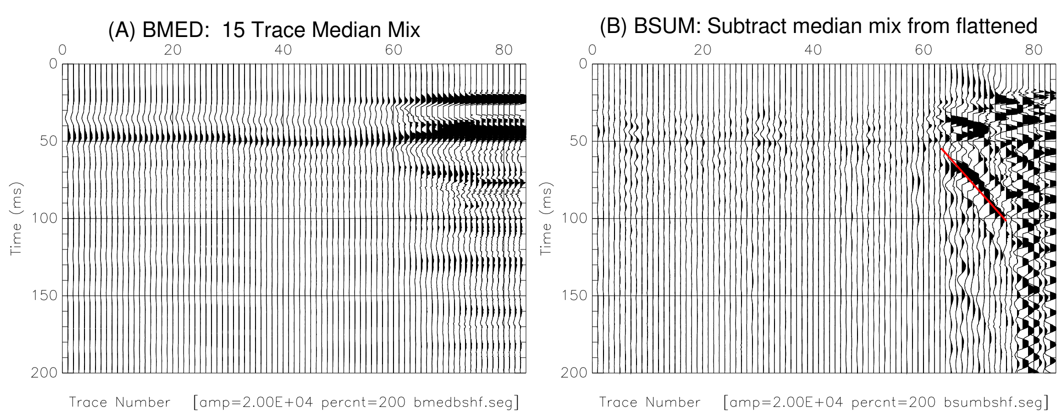


Figure 103: BMED: (A) median mix of the direct wave (see figure 102 B) (B) BSUM: direct down-going wave estimate subtracted from total wave field.

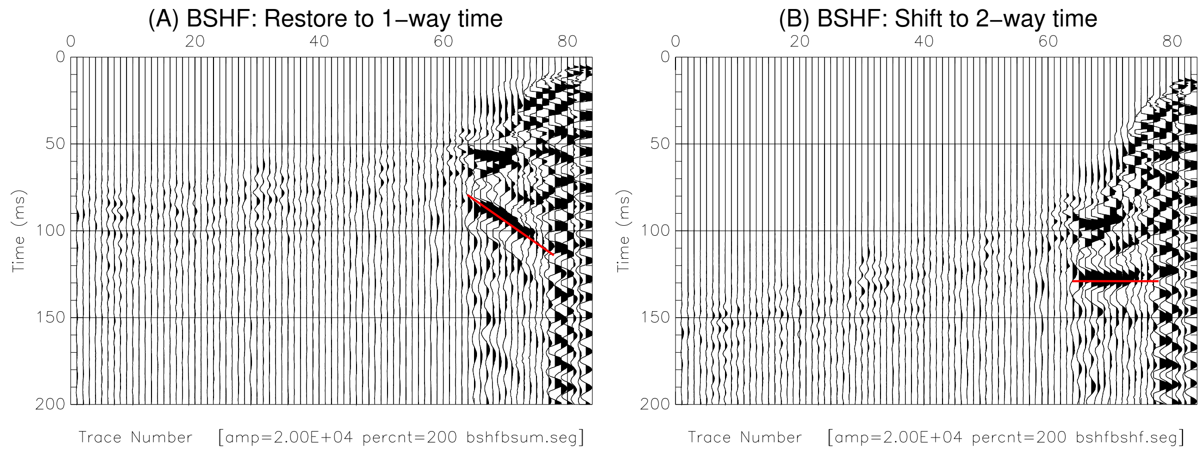


Figure 104: BSHF: (A) Restore to 1-way time, down going removed (see figure 103 B) (B) BSHF: Shifting data in (A) to 2-way time. Reflections should be horizontal.

12.2 Additional Down-hole Processing

The following signal processing programs are often used in processing down-hole data, as well as in other circumstances.

- **BSHP** 12.2.1 Weiner least square shaping filter.
- **BTOR** 12.2.2 Applies PCA analysis to headers.
- **GENBROT** 12.2.3 Generates a bash script for **BROT**.
- **BROT** 12.2.4 Actually rotates horizontal component data in a down-hole survey.

12.2.1 BSHP

Wiener least squares shaping filter design and application to standardize an embeded wavelet. When used in a down-hole survey, it can be used to remove variations in triggering and bandwidth from repeated source efforts from the down-hole data. The design is to find a filter which matches each source effort to a target trace on the reference phone. This way, we remove variations in the source effort from appearing in the down-hole data. This removal occurs when we re-apply the designed filters on the reference phone to the down-hole data, thus standardizing the embeded wavelet.

```
bshp infil  infil2 iswopt iswch tmin tmax npf sf infil3

infil :   =name of input file #1
           (file1*filter=file2)
infil2:   =name of input file #2
iswopt:  0=profile mode
         1=trace mode
iswch:   1= (file#1*filter)=output
         0= (file#3*filter)=output
tmin:     =start time of design gate in seconds
tmax:     =end time of design gate in seconds
npf:      =number of point in filter
sf:       =stability factor (.001 typical)
infil3:   =name of input file #3 (iswch=0 only)
```

Example: If **tref.seg** is the T-component of the reference phone, and if **targ.seg** is a target trace (perhaps the last of the source efforts as recorded on that same component, the a shaping filter can be found that matches each source effort to that last effort. Rational is that the last source effort is stable due to the compaction of soil below a hammer source. The command to match each source effort to the target trace might be:

```
bshp tref.seg targ.seg 1 1 0 0.1 360 .0001
```

The output file would be **bshptref.seg** and should be plotted to asses the degree of success and the chosen command line arguments. Then application of the filter designed above to the down-hole data might be done with this command:

```
bshp tref.seg targ.seg 1 0 0.  0.1 360 .0001 twav.seg
```

where **twav.seg** is the file with the original down-hole data which are contaminated by variations in trigger source timing and embedded source wavelet. What happens is that the shaping filters are recomputed with the same design input, but applied this time to the file listed as the last argument on the command line. The shaped down-hole data would be the output file, **bshptwav.seg**.

The degree to which shaping is helpful depends on how repeatable the source efforts are. With a highly stable and repeatable source, there will not be much difference in the result from shaping. However, with a source that produces variation in triggering or wavelet radiated, the result may be very helpful. Shaping will not hurt unless significantly poor choices are made in the command line parameters.

12.2.2 BTOR

Applies azimuth and vertical angles to geophone trace headers from a **bhod.lst** file. The command line arguments are:

```
btor lstfil, prfx, isw1 maxtr

lstfil  =input list file name (ex. bhod.lst)
prfx    =*.seg file prefix (one character)
isw1    =up/down switch
      -1=apply to *.lst file and one less
      +1=apply to *.lst file and one more
       0=IF VERTICAL IMPACT source
maxtr   =maximum number of traces in shot record
       6= 3 components down-hole, 3 ref-phone
       7= 3 down, 3 ref-phone, 1 load cell


EXAMPLES:
       [if file number (col. 1 of bhod.lst) is 005
       and isw1=-1, the azimuth and vertical angle
       also applied to file 004]

       [if file is 005 and isw1=+1, azi and vert
       applied to 006 also]
```

A summary of the flow is this:

- **GENBHOD** 10.1.9 creates bash scripts to run on down-hole data acquired from a horizontal component source, two blow orientations per subsurface station. The output is a file, **bhod.lst**.
- **BTOR** Reads the **bhod.lst** file and applies the determined phone orientations to the headers.
- **GENBROT** 12.2.3 creates scripts to run **BROT**.
- **BROT** 12.2.4 runs the script to actually rotate the data to a standard orientation.

12.2.2.1 Example of BTOR Consider a single depth station for illustration. There may actually be 100 or more depth stations in a single down-hole survey. There are two files in this example:

- **c009.seg** Source orientation is azimuth 270 degrees, 90 degrees from vertical (ie. horizontal blow West).
- **c010.seg** Source orientatio is azimuth 90 degrees, 90 degrees from the vertical (ie. horizontal blow East).

The steps are:

1. **gobhodo** This generates the difference between scaled versions of the two source efforts. The scaling is done on the vertical component of the down-hole phone (ch 1 on the author's wiring). File 9 is subtracted from 10. The difference file is renamed as h010009.seg
2. **gorunbhod** Program bhod is run to analyze file h010009.seg and produces files:
   h0010.plt.ps, bhod.lst
   These are the hodogram plot and a file with the determined phone orientations (R and T downhole)

   The command **in the script** for this depth is:

   ```
   bhod h010009.seg 2 3 50 90.0 180.0 +90.0
   ```

   Ch 2 is R and Ch 3 is T component downhole. 50 percent max amplitudes used in analysis 90 deg is source azimuth (ie E-W) and bowspring, R-phone observation is close to 180 degrees. The downhole phone is wired for +90 degrees between R and T components
3. **BTOR** This program inserts the orientations of the phone azimuths and vertical orientations into the headers. The command **in the script** for this depth is:

   ```
   btor bhod.lst c -1 6
   ```

   *This command will process ALL the cxxx.seg files in the directory (subject to be included in the **bhod.lst** file).*

Renaming btorxxxx.seg files to xxxx.seg

A script to rename the **BTOR** files in a directory is as follows:

```
 #!/bin/sh
#Script to rename files after btor process
#overwrite pxxx.seg files, p=prefix
# Author: P. Michaels     Date:April 2002   See GNU License

if test "$1" = ''
   then
     echo 'Enter 1 character prefix'
     echo 'Example:  w'
     echo ' for files btorw001.seg, btorw002.seg, etc...'
     read PRFX
   else
     PRFX=$1
fi

 find -name "$PRFX*.seg" |  \
 sed s/'\.\/'/' '/g |       \
 gawk '{print "mv","btor"$1,$1}'   \
  >go-rename
chmod +x go-rename
./go-rename
echo "btor files renamed$"
```

12.2.3 GENBROT

Once the *.seg files have had their headers updated with the geophone orientations, we can rotate the data so that the horizontal components face in a standard direction. In down-hole surveys, as the tool is dragged up the hole, it can slowly rotate. In some cases, the tool may become stuck, have to be unclamped and then reclamped, resulting in tool spin. This program is interactive and generates a bash script to apply a rotation of the data so that one component is parallel to the source azimuth (assuming an SH-wave source is used). An example log of a run follows:

```
Enter alpha prefix (char) of *.seg data to be rotated

EXAMPLE: if enter 1, then files 1001.seg to 1010.seg
                     would be processed if sequence
                     numbers 1 and 10 entered next


L

 ...L
 Enter first file number to process
1
 Enter last file number to process
146
 Output in file===>gobrot
```

The generated script file, **gobrot**, will then look like this:

```
  brot L001.seg 2 3 1
  brot L002.seg 2 3 1
  brot L003.seg 2 3 1
.
.
.
  brot L144.seg 2 3 1
  brot L145.seg 2 3 1
  brot L146.seg 2 3 1
```

Of course, one must then make the **gobrot** file executable:

```
chmod +x gobrot
```

12.2.4 BROT

One runs the **gobrot** bash script and this produces files **brotL001.seg** through **brotL146.seg** in this example. The command line arguments are:

```
 brot infil itrx itry iopt iangle

infil:   =input file name
itrx:    =trace number for x-axis (example, 2)
itry:    =trace number for y-axis (example, 3)
iopt:    =rotation option
        0=user supplied rotation angle
        1=Transverse Alignment: chan 3 aligned to 6
        2=Radial Alignment:     chan 2 aligned to 5
iangle:  =user supplied rotation angle from x-axis
```

One only needs to add an iangle parameter with the iopt=0 option. For each *.seg file rotated, there will also be a *.lst file output. The *.lst file shows what **iangle** value has been used based on the headers for options 1 or 2.

NOTE: BSU codes like this one assumes that the channel order in down-hole surveys matches those of the author. See section 6.7.2 of the BSU Users Guide (`bsu-user-guide3-1.pdf`) for more on this topic. Figure 105 illustrates the author's notation and wiring, and is taken from the BSU user guide. A discussion on Principal Component Analysis (PCA) is found in the literature (Michaels, 2001b).

Once the **gobrot** script is run, the rotated data will have names **brotL001.seg** through **brotL146.seg** in this example. A good practice is to create a child directory, **brot** and move the **brotxxxx.seg** files to that directory before doing further analysis. This will preserve clarity on which files have been rotated, and which files are as recorded.

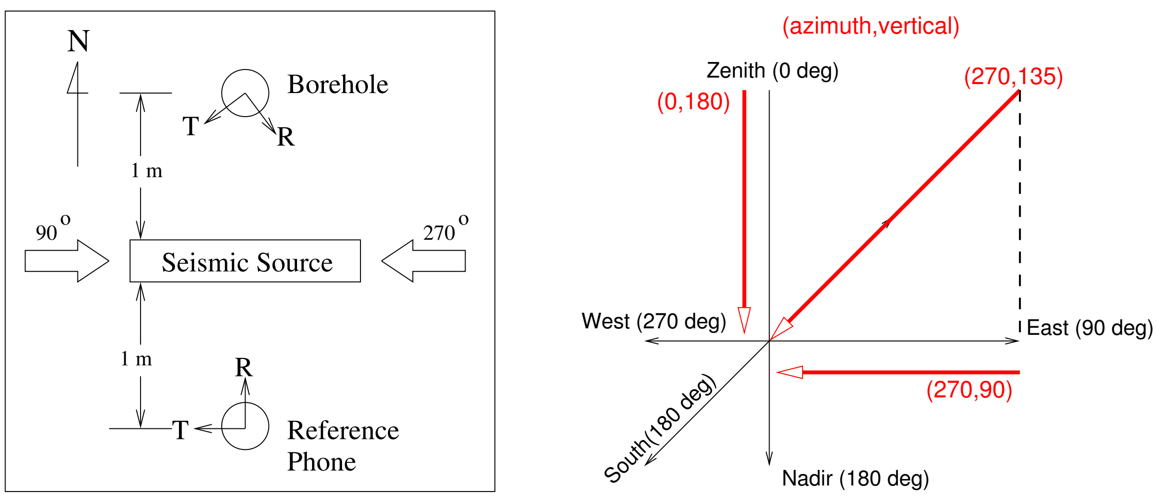


Figure 105: The author's orientations and notation for down-hole surveys. Note that the reference and bore-hole phones are wired differently (in terms of R- and T-component wiring).

12.3 FILTER Codes

12.3.1 BFXT

The Frequency-Distance (FX) transform may be computed for a shot gather. The output files are **bfxtampl.seg** and **bfxtphaz.seg** if a forward transform is computed. The command line arguments are:

```
 bfxt  infile1  iopt   infile2

infile1 =name of X-T input file (space,time)

 iopt    1=forward transform (unwrap phase)
         2=forward transform (no unwrap phase)
        -1=inverse transform (two input files)

infile1 =name of input file AMPLITUDES(iopt=-1 Only)
infile2 =name of input file PHASE(iopt=-1 Only)

NOTE: Forward transform requires only one input file
      Inverse transform requires two input files
```

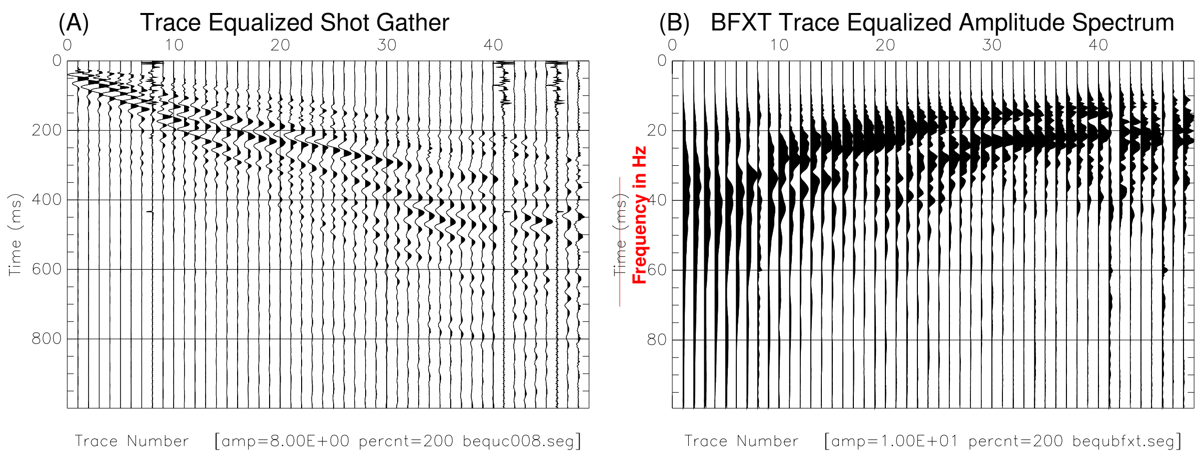


Figure 106: BFXT: (A) trace equalized shot gather using BEQU 12.0.9 (B) the amplitude spectrum after equalization with BEQU. Not shown is the phase transform.

The example shown in Figure 106A shot gather has a sample interval of $\Delta t = .0005$ *sec* and 2500 samples per trace. The code uses a Radix 2 FFT, and in this case the sample interval is modified and there are 2048 samples per trace. The Figure 106B plot has to be relabeled since we are using BSU plot program **BPLT** here, and that program is limited to assuming all data are in time. A frequency axis replaces the time axis, and frequencies run from zero to the Nyquist. It appears to the headers as if the maximum sample is at a time of 1.0. In actual fact, the maximum sample is at 1000 Hz. Some scaling is going on to make plotting easier.

So Figure 106B is plotted to a maximum of 0.1 which turns out to be 100 Hz. So what is going on? BFXT calls a subroutine, nrad2.f which computes the first power of two larger than the number of samples in the shot gather, call it N2. A frequency domain sample interval is computed on this larger number of samples (the code pads with zeros to fill it out). Thus, $\Delta f = 1/(N2 \cdot \Delta t)$. But because we are dealing with time domain codes for other things we might do, we scale the sample interval, dividing it by 1000. Thus a Nyquist of 1000 Hz (maximum sample frequency in FX domain) becomes 1.0, as if it were 1.0 seconds. When going back into the time domain (TX), all this is reversed.

12.3.2 BCAR

This is a high-speed filter based on a moving average box car operator. It can do smoothing (ie. low pass) or high-pass filtering by subtracting a low pass result from the original data. For most applications, BSU has better filters (parameters in frequency rather than time), but this is quick and dirty, and is specified in time duration. Auto-Regressive-Moving-Average (ARMA) filtering can be done with BFIL 12.3.3. The command line arguments for BCAR are:

```
bcar  infile  isign  ntimes  twide

infile    =input file name
isign   -1=low-pass box car
         0=high-pass (1-sinc^ntimes)
        +1=high-pass (1-sinc)^ntimes
ntimes    =number of times to apply boxcar
twide     =width of box car in seconds
```

An example is shown in Figure 107 where both a low and high-pass filter are demonstrated. The commands were for low- and high-pass respectivelly :

```
bcar c008.seg -1 1 .1
```

and

```
bcar c008.seg 1 1 .1
```

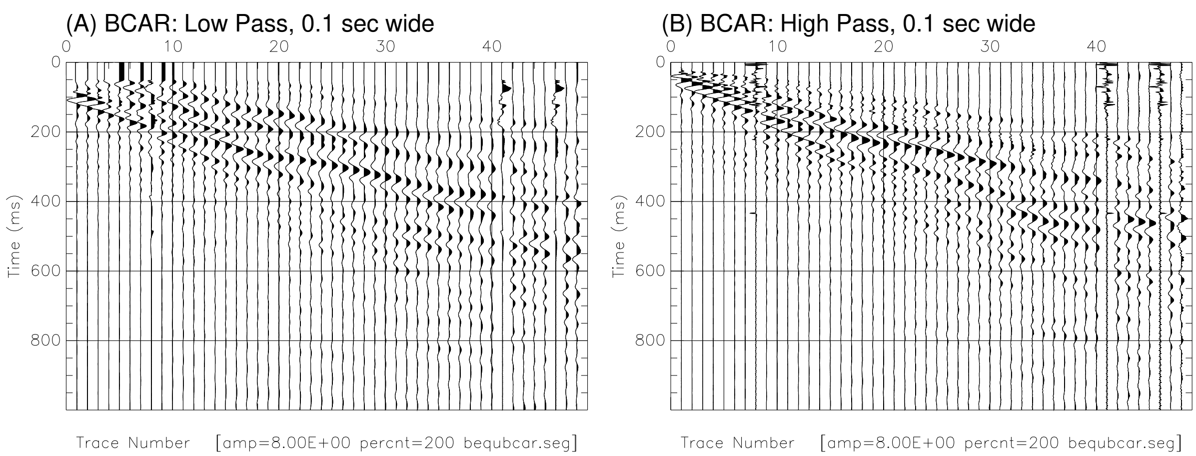


Figure 107: BCAR: (A) low-pass filter, trace equalized with BEQU 12.0.9 (B) high-pass filter by subtracting low-pass from original data, also trace equalized. Input data are same as in Figure 106A.

12.3.3 BFIL

BFIL uses a bilinear transform to perform ARMA filtering. This is the preferred filter program. Zero phase filtering is done by two passes of minimum phase filtering in opposite temporal directions. The command line parameters are:

```
bfil infile itype npoles fcenter bwidth ifaz

infile  =input file name
itype   0=low-pass filter, cut off freq= fc (-3dBv)
        1=band-pass filter, center frequency= fc
        2=high-pass filter, cut off freq= fc (-3dBv)
npoles  =number of poles in filter
         (6dB/octave)/(pair of poles)
fcenter =center frequency Hz
bwidth  =band-pass filter bandwidth (-3dB) Hz
ifaz    1=minimum phase  0=zero phase filter
```

Examples of filtering with BFIL are shown in Figure 108:

- **(A) Low-Pass** Minimum phase, 12 Hz cut-off, 4 poles

```
bfil c008.seg 0 4 12.  1
```

- **(B) High-Pass** Minimum phase, 48 Hz cut-off, 4 poles

```
bfil c008.seg 2 4 48.  1
```

- **(C) Band-Pass** Minimum phase, 24 Hz center, 24 Hz band-width, 4 poles

```
bfil c008.seg 1 4 24.  24.  1
```

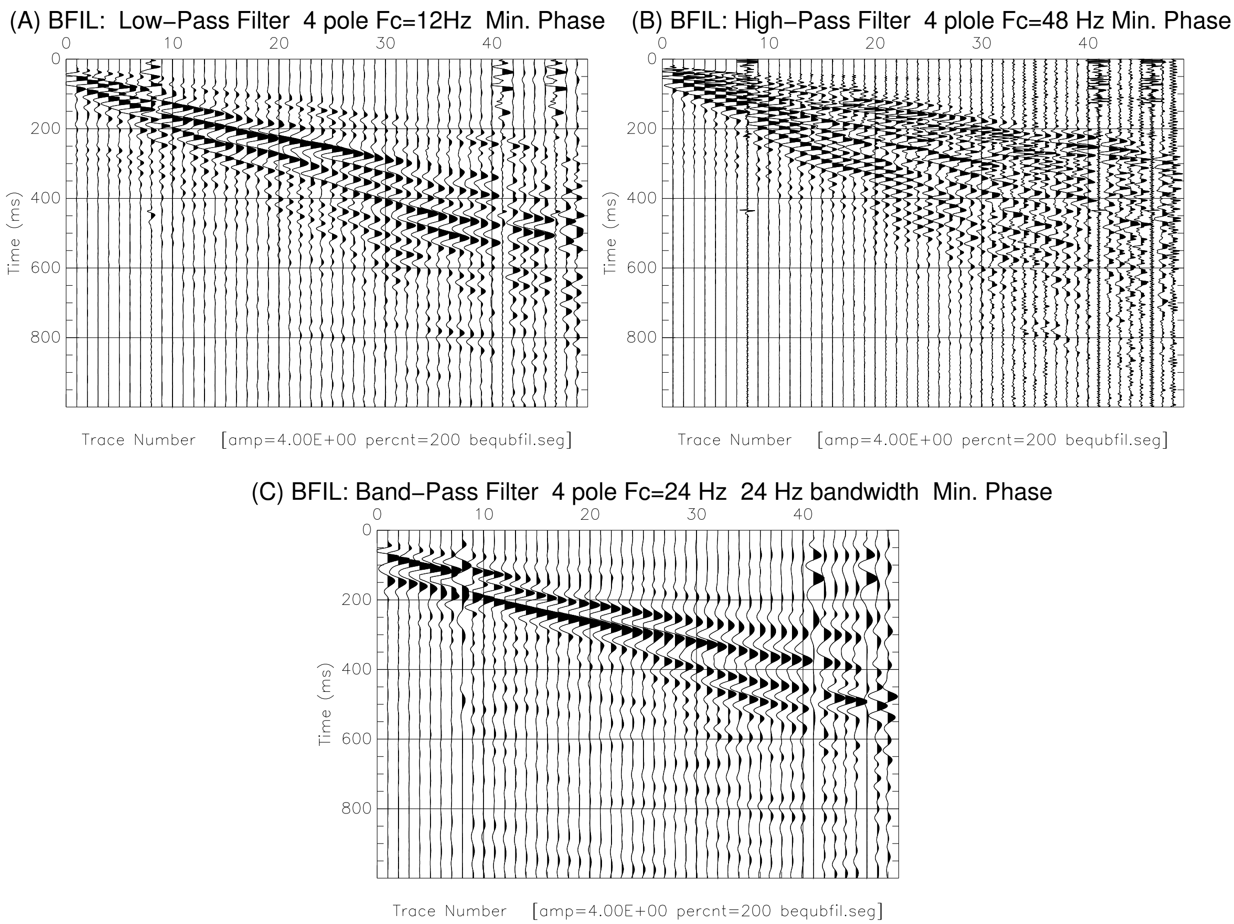


Figure 108: BFIL: Input data are same as in Figure 106A.

Another way to band-pass filter is to run the data twice, once through a low-pass, and then through a high-pass filter, choosing cut-off frequencies to produce a band-pass.

12.3.4 BDCN

Classic minimum phase spiking decon when the prediction error option is chosen. Intended for reflection data with random reflections and minimum phase wavelet, can be run in trace or profile mode. However, it can be run on other data as a whitening operator, your mileage will vary. Command line arguments are:

```
bdcn infile tmin tmax mpts stabf iprof imode
infile  =input file name
tmin    =Autocorrelation Gate: START
tmax    =Autocorrelation Gate: END
mpts    =Length of Decon Operator
stabf   =Stability Factor (ex: 0.01)
iprof   1=profile mode  0=trace mode
imode   1=Prediction 0=Prediction error
        Choose 0 for spiking decon
```

Examples of BDCN are shown in Figure 109.

- **(A) Prediction** Gate:[0-1.2 sec] 30 sample (15 ms $\Delta t = .0005\ sec$) operator, 0.1 stab factor, trace mode:
```
bdcn c008.seg 0 1.2 30 .1 0 1
```
- **(B) Prediction Error, Spiking** Gate:[0-1.2 sec] 30 sample (15 ms $\Delta t = .0005\ sec$) operator, 0.1 stab factor, trace mode:
```
bdcn c008.seg 0 1.2 30 .1 0 0
```

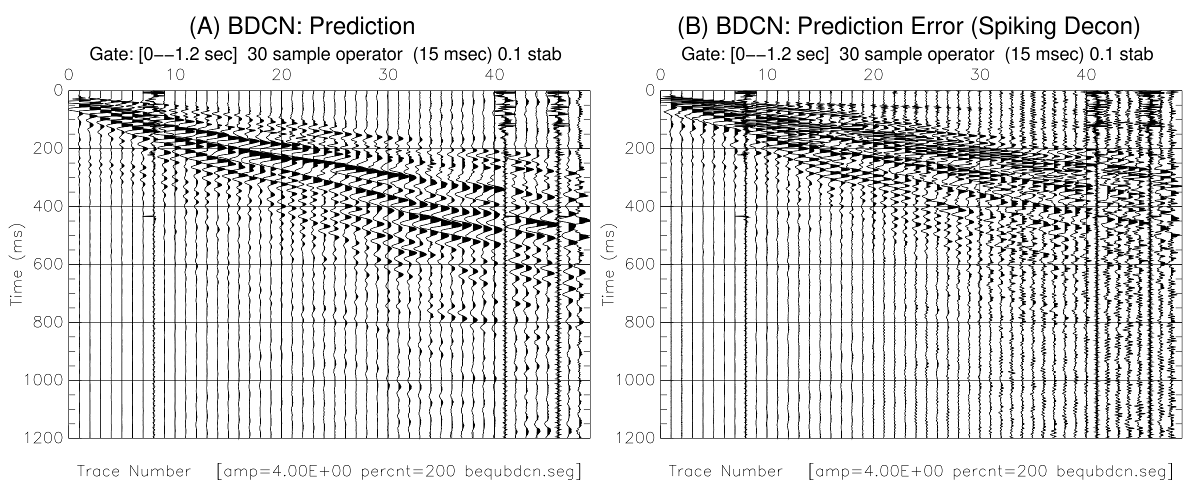


Figure 109: BDCN: Input data are same as in Figure 106A.

12.3.5 BFTR

One can either filter one data set with another, or filter using a namelist file (like the one produce with TRAPLT 6.0.1. The command line arguments are:

```
bftr   infil   iswf    filef

infil:    =input file to be filtered
iswf:     =filter source switch
         0=filters in *.SEG data set
          (one filter trace for each trace in infil)
         1=single filter specified in namelist file

filef:    =name of filter data set

------Namelist Definitions-------
&FILTER   npf=number of points in filter
          nzph=sample for zero reference
          fil=f1,f2,f3,f4,....
              (values of filter samples)
&end

NOTE: if you get a core dump, you may have
      forgotten the &end at the end of the file
```

The following example shows how to generate filter traces and apply them.

```
# filter with low pass, 4 pole 12 Hz cut off minimum phase
bdum c008.seg .10
bfil bdumc008.seg 0 4 12. 1
bplt bfilbdum.seg 2 0 0 1 100 0.0 .4 1 2E-2 200

# apply low passed by convolving bfilbdum.seg with c008.seg
bftr c008.seg 0 bfilbdum.seg
bequ bftrc008.seg 0 1.
bplt bequbftr.seg 2 0 0 1 100 0.0 1. 1 2 200
```

The procedure:

1. BDUM creates a file with an impulse at 0.1 seconds, the template is the field data file, **c008.seg**. The output is **bdumc008.seg**.
2. BFIL filter the impulse file with a low pass filter, 4 pole, 12 Hz cutoff, minimum phase. Output is **bfilbdum.seg** Figure 110A.
3. BFTR filter **c008.seg** with the file, **bfilbdum.seg** Figure 110B.

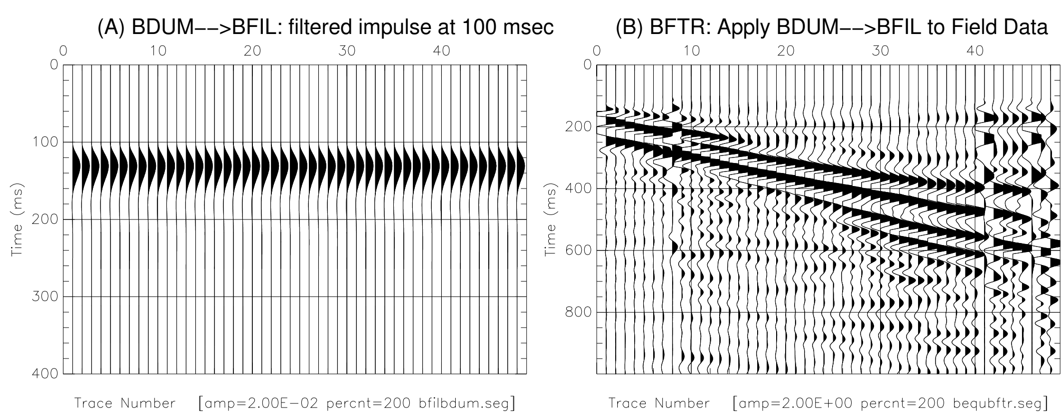


Figure 110: (A) BDUM–>BFIL: Filtered file of impulses. (B) BFTR: Filter field data with filtered impulse file. Input data **c008.seg** are same as in Figure 106A.

The other alternative would be to run **TRAPLT** on file **bfilbdum.seg** and copy the namelist section to a file, call it **filter.dat**. We must add a &end statement not provided by TRAPLT. The **filter.dat** file will look like this:

```
 &filter
  npf=  221, nzph=    1
  fil=
   0.0000,   0.0000,   0.0000,   0.0000,   0.0000,   0.0000,   0.0000,   0.0000,
   0.0000,   0.0000,   0.0000,   0.0000,   0.0000,   0.0000,   0.0000,   0.0000,
   0.0000,   0.0000,   0.0000,   0.0000,   0.0000,   0.0001,   0.0003,   0.0007,
   0.0015,   0.0027,   0.0044,   0.0067,   0.0095,   0.0130,   0.0172,   0.0220,
   0.0276,   0.0338,   0.0406,   0.0481,   0.0563,   0.0650,   0.0743,   0.0842,
   0.0945,   0.1054,   0.1166,   0.1282,   0.1401,   0.1524,   0.1648,   0.1775,
   0.1903,   0.2032,   0.2161,   0.2290,   0.2419,   0.2548,   0.2675,   0.2800,
   0.2923,   0.3044,   0.3162,   0.3278,   0.3389,   0.3498,   0.3602,   0.3702,
   0.3798,   0.3889,   0.3975,   0.4056,   0.4132,   0.4203,   0.4269,   0.4329,
   0.4384,   0.4433,   0.4477,   0.4515,   0.4547,   0.4574,   0.4596,   0.4612,
   0.4622,   0.4628,   0.4627,   0.4622,   0.4612,   0.4597,   0.4577,   0.4552,
   0.4523,   0.4489,   0.4451,   0.4409,   0.4363,   0.4313,   0.4260,   0.4203,
   0.4143,   0.4080,   0.4015,   0.3946,   0.3875,   0.3801,   0.3726,   0.3648,
   0.3569,   0.3488,   0.3405,   0.3322,   0.3237,   0.3151,   0.3064,   0.2977,
   0.2889,   0.2801,   0.2712,   0.2624,   0.2535,   0.2447,   0.2359,   0.2271,
   0.2184,   0.2098,   0.2013,   0.1928,   0.1844,   0.1761,   0.1680,   0.1599,
   0.1520,   0.1442,   0.1366,   0.1291,   0.1218,   0.1146,   0.1076,   0.1007,
   0.0940,   0.0875,   0.0811,   0.0750,   0.0690,   0.0632,   0.0575,   0.0521,
   0.0468,   0.0417,   0.0368,   0.0321,   0.0275,   0.0232,   0.0190,   0.0150,
   0.0112,   0.0075,   0.0040,   0.0007,  -0.0025,  -0.0054,  -0.0083,  -0.0109,
  -0.0135,  -0.0158,  -0.0181,  -0.0201,  -0.0221,  -0.0239,  -0.0255,  -0.0271,
  -0.0285,  -0.0298,  -0.0309,  -0.0320,  -0.0330,  -0.0338,  -0.0345,  -0.0352,
  -0.0357,  -0.0362,  -0.0366,  -0.0368,  -0.0371,  -0.0372,  -0.0372,  -0.0372,
  -0.0372,  -0.0370,  -0.0368,  -0.0366,  -0.0363,  -0.0359,  -0.0356,  -0.0351,
  -0.0347,  -0.0341,  -0.0336,  -0.0330,  -0.0324,  -0.0318,  -0.0311,  -0.0305,
  -0.0298,  -0.0291,  -0.0284,  -0.0276,  -0.0269,  -0.0261,  -0.0254,  -0.0246,
  -0.0239,  -0.0231,  -0.0223,  -0.0216,  -0.0208,  -0.0200,  -0.0193,  -0.0185,
  -0.0178,  -0.0171,  -0.0164,  -0.0156,  -0.0149,
&end
```

The commands for the alternative would be:

```
traplt bfilbdum.seg 0.09 .2 1 0 1
bftr c008.seg 1 filter.dat
bequ bftrc008.seg 0 1.
bplt bequbftr.seg 2 0 0 1 100 0.0 1. 1 2 200
```

NOTE: The TRAPLT command above does not start listing at 0.0 seconds, but at .09 seconds. This produces a namelist file with less delay, and this is evident comparing the two different approaches.

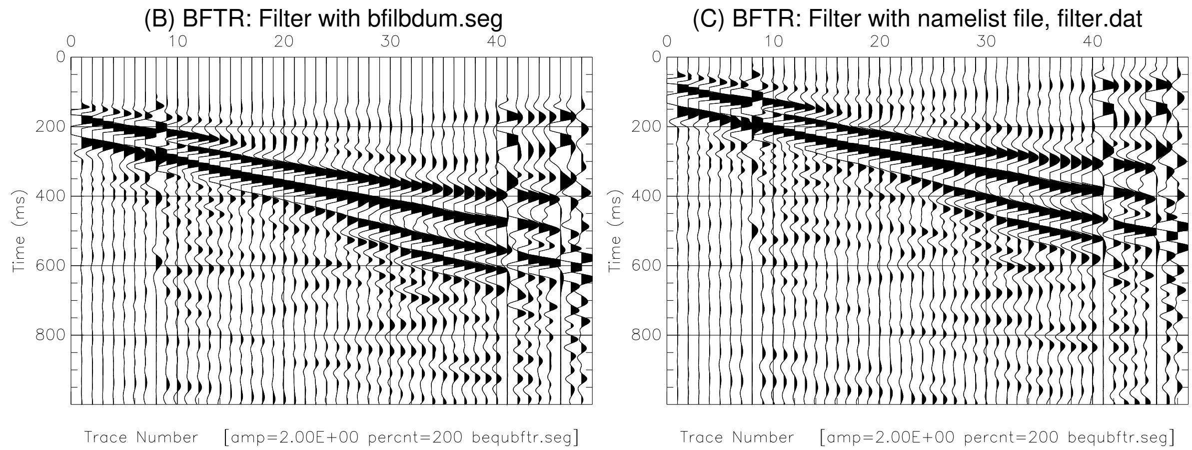


Figure 111: (B) BFTR: same as in Figure 110B. (C) BFTR: Filter field data with namelist file, **filter.dat**. Input data **c008.seg** are same as in Figure 106A. Note the different time delay due to placing an impulse at 100 ms (Figure 110A) in the filtered BDUM, compared to the start time in the TRAPLT approach (.09 sec).

12.3.6 BWHT

Data are whitened (increased bandwidth). The user defines a number of overlapping frequency bands which are individually subjected to Automatic Gain Control (AGC), and then reassembled into a whitened product. Highly nonlinear, but may reveal details in the data by overcoming dynamic range limitations in traditional plots of data. The command line arguments:

```
bwht  infil  twide  fcent  bwdth  froll

infil:    =input file to be filtered
twide:    =AGC window length in sec.
fcent:    =center frequency (Hz)
bwdth:    =bandwidth (Hz)
froll:    =roll off (Hz)
```

EXAMPLE:

```
bwht c008.seg .4 50.  80.  10.
```

We can view the **bwhtc008.lst** file to see the filter details. See Figure 112.

```
Parameters:
twide=       0.40
freqc=      50.00
bwdth=      80.00
deltf=      10.00
nfilt=     9
number of points in filter=        201
    J    F_Center (Hz)
    1          10.00
    2          20.00
    3          30.00
    4          40.00
    5          50.00
    6          60.00
    7          70.00
    8          80.00
    9          90.00
```

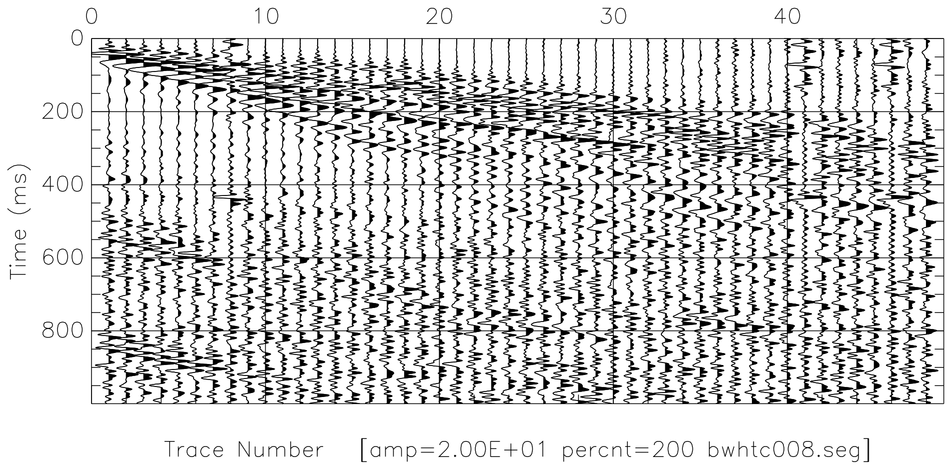


Figure 112: BWHT: 0.4 second AGC window, 50 Hz center, 80 Hz bandwidth, 10 Hz rolloff. Input data **c008.seg** are same as in Figure 106A.

13 Seismic Interferometry, Passive Sources

Cross correlation is often used to study passive sources. Typical applications will often capture signals radiated from vehicle traffic or other "noise" sources. The approach can also be used with active sources. The following codes have been added to BSU-3.0.3 to work with this type of data.

- **BCOR** 13.0.1 Cross correlate a selected trace in a shot gather with all other traces in that gather.
- **BIMG** 13.0.2 Cross correlates traces in a shot gather by relative offsets. Output is sorted by offset between pairs, near to far combinations.
- **GENBIMG** 13.0.3 Helper program that generates a BASH script, "gobimg" that calls BIMG. Correlation windows are defined over time and range.
- **BAZI** 13.0.4 Horizontal hodogram PCA analysis to locate seismic source.
- **GENBAZI** 13.0.5 Run this program to generate a bash script that will run a sliding window in time by invoking multiple calls to **bazi**.
- **BZRT** 13.0.6 Vertical hodogram PCA analysis to study major axis of polarization ellipse in vertical plane.
- **GENBZRT** 13.0.7 Run this program to generate a bash script that will run a sliding window in time by invoking multiple calls to **bzrt**.
- **HVSR** 13.0.8 Computes the ratio of vertical to horizontal (H/V) spectral ratios for multi-component data.

13.0.1 BCOR

If used, it will likely be to study a data set in terms of satisfying assumptions. The user selects one trace from a gather of traces and cross correlates that trace with all other traces.

```
bcor infile1 itrace t1 t2 tlagmx

infile1: name of input file #1
itrace:  number of trace to correlate with others
t1:     =start time of cross correlation gate (sec.)
t2:     =end time of cross correlation gate (sec.)
tlagmx: =maximum cross correlation lag time (sec.)
```

```
bcor DATA.seg 1 40.  60.  4.
```

These data (Figure 113) are sampled from the publication Zhang *et al.* (2020). They are recorded from a line of geophones along a road. Trace 1 is correlated with the other traces for a maximum lag of 4.0 seconds (actually ±2.0 seconds. Figure 114 shows the cross correlation for both positive and negative lags. That is, examine trace 1 on far left. That is an auto correlation of the first trace. The time of 2.0 seconds is essentially zero lag.

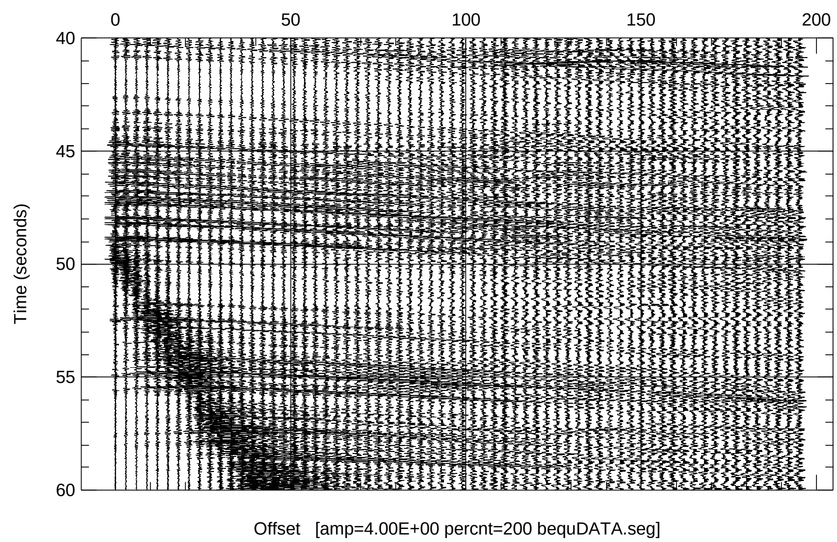


Figure 113: A section of data from 40 to 60 seconds. Note the vertical time scale is different than that in Figure 114. The large amplitude slow trend (approximately 14 km/hr) in the lower left appears to be a motor vehicle while the remaining events appear to be waves propagating in the soil (approximately 150 to 200 m/s).

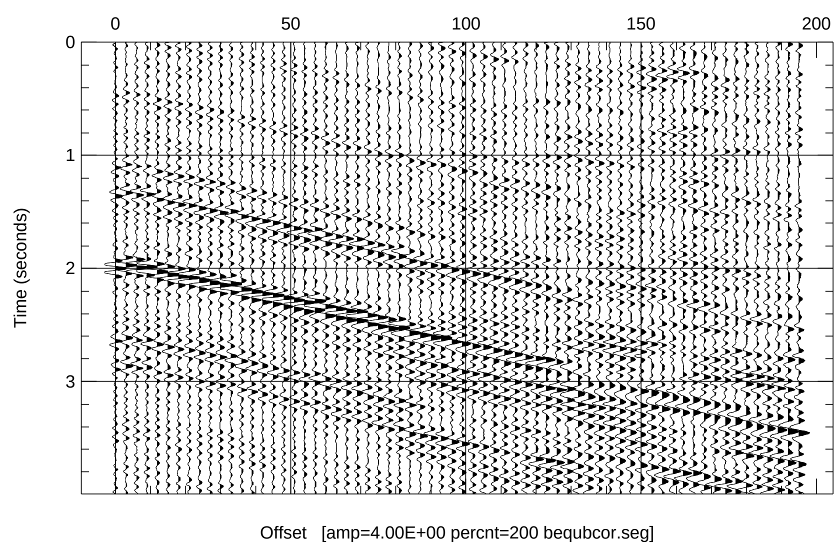


Figure 114: BCOR: Cross correlation of Figure 113 data from 40 to 60 seconds. Zero lag is at 2.0 seconds. The event starting at 2.0 seconds on the left appears to present a horizontal velocity of about 150 m/s.

13.0.2 BIMG

This program creates a pseudo shot gather by mixing cross-correlated traces by offset (measured in trace spacings). Thus, it works best if the trace spacing is uniform for the most part. The smallest offset would be 1 trace spacing, the furthest the near to far trace (but that will only give one instance). Average group interval computed, assumed all traces are equally spaced. Acausal output is for waves propagating in one direction, causal section for waves propagating in the opposite direction (see output file **bimgxxxx.seg**). The acausal and causal portions are combined without any weighting in second output file (see file **BIMGxxxx.seg**). Assumes all sources in line with geophones (no azimuthal corrections for apparent velocity). Likely use is traffic noise acquired along a road.

```
bimg infile1 t1 t2 tlagmx jnear jfar

infile1: name of input file #1
t1:     =start time of cross correlation gate (sec.)
t2:     =end time of cross correlation gate (sec.)
tlagmx: =maximum cross correlation lag time (sec.)
jnear:  = near correlation offset (trace spacings)
jfar:   = far correlation offset (trace spacings)
```

EXAMPLE:

```
bimg DATA.seg 0.  100.  4.  1 33
```

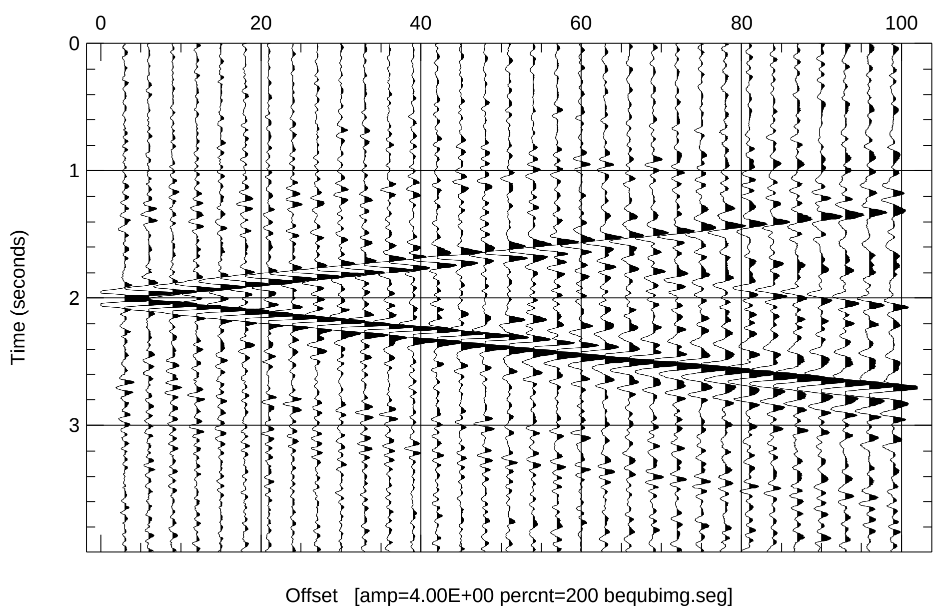


Figure 115: BIMG: Data from Figure 113, time gate 0 to 100 seconds processed for trace offsets from 1 to 33. A larger time gate improves the statistics of the stack. The average spacing is 3 meters per trace. Only half of the available offsets are used to build up the stack. Note the time scale is 0 to 4.0 seconds with zero lag at 2.0 seconds.

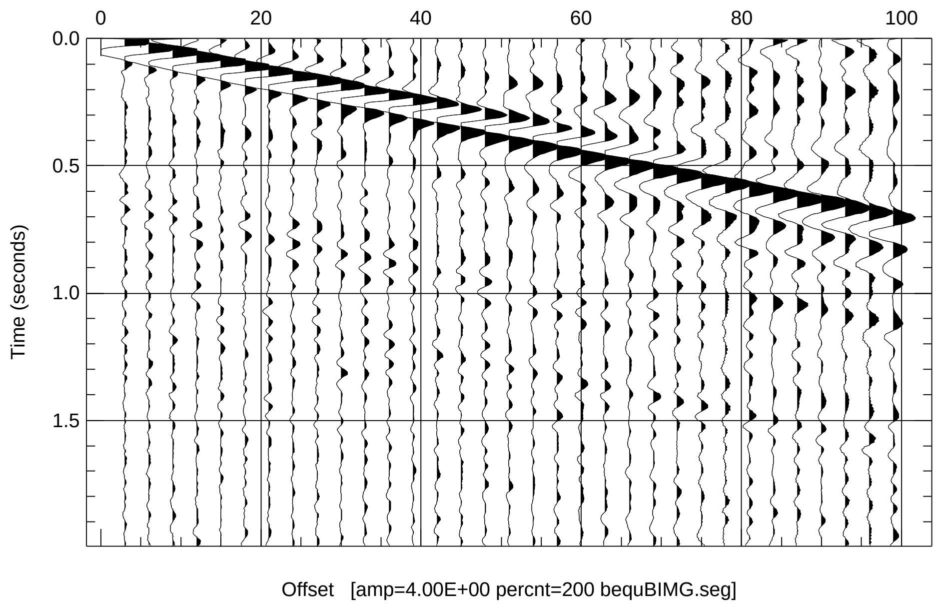


Figure 116: BIMG: Output file BIMGdata.seg mixes both causal and acausal arrivals. Note that the time scale is 0 to 2.0 seconds with zero lag at 0 seconds. The interval from 2 to 0 seconds in Figure 115 is time reversed and mixed with the 2 to 4 seconds interval. This mixes both directions of arrival.

13.0.3 GENBIMG

BENBIMG generates a bash script which makes calls to BIMG (13.0.2). Each call to BIMG is for a time-space sliding window that moves over a large data set. User specifies a window length (tgate) and time shift forward. The maximum 2 sided cross correlation lag is also required on the command line. Program BIMG is intended to process passive sourced data. It is assumed that the sources are in line with the geophone array (no azimuth corrections to correct for apparent velocities). There is a man page, but no help with the "-h" option. Rather, if only the command is issued, the following help is provided:

```
 USAGE: genbimg filename tstart tend tstep tgate tlagmx trace1 traceN jnear jfar mix

filename = input file name
tstart   = start time (float, seconds) of processing
tend     = end time (float, seconds) of processing)
tstep    = time shift forward with sliding window (float, seconds) of processing)
tgate    = length in time of sliding window (float, seconds)
tlagmx   = 2 sided maximum correlation lag (float, seconds)
trace1   = first trace to process (int)
traceN   = last trace to process (int)
jnear    = near correlation separation (int, traces)
jfar     = far  correlation separation (int, traces)
mix      = mean mix length (0=no mix)  (int, traces)
 ABORT
 you only have 1 arguments on command line
argv[0] = genbimg
```

If less than all 11 command arguments is given, then the ABORT message will be given, alerting the user to additional command line arguments are required. The mix argument is used to attenuate broadside arrivals. If it is not needed, set mix to 0. However, as will be shown in the following example, a mix of 3 helps a great deal. When a mix is used, additional calls to the program BMIX (12.1.3) are made at the end of the script.

EXAMPLE:

```
genbimg DATA.seg 0 100.  20.  20.  4.  1 66 1 30 0
```

The output is:

```
 SUCESS Now you run the output script
Output Bash Script ==> gobimg
 (type gobimg)     ==> images:  stak.pdf and STAK.pdf
                   ==> data:    stak.seg and STAK.seg
```

The output script should be executable in Linux. It will look like this:

```
#!/bin/bash
# Input File          = DATA.seg
# Start Time          = 0.00 sec.
# End Time            = 100.00 sec.
# Time Step (moveups) = 20.00 sec.
# Time Gate           = 20.00 sec.
# Max Lag             = 4.00 sec.
# First Trace Number  = 1
# Last Trace Number   = 66
# Near correlation trace spacing   = 1
# Far  correlation trace spacing   = 30
# mix  mean mix length in traces   = 0
#
echo gate: 0.0 20.0
bedt DATA.seg 0.000000 20.000000 1 66 1 0 >/dev/null
brdc bedtDATA.seg 1 >/dev/null
bimg brdcbedt.seg 0. 20.000000 4.000000 1 30 >/dev/null
mv  bimgbrdc.seg  stak.seg
mv  BIMGbrdc.seg  STAK.seg
echo gate: 20.0 40.0
 bedt DATA.seg 20.000000 40.000000 1 66 1 0 >/dev/null
brdc bedtDATA.seg 1 >/dev/null
bimg brdcbedt.seg 0. 20.000000 4.000000 1 30 >/dev/null
bsum bimgbrdc.seg  stak.seg 1.0 >/dev/null
mv bsumbimg.seg stak.seg
bsum BIMGbrdc.seg  STAK.seg 1.0 >/dev/null
mv bsumBIMG.seg STAK.seg
echo gate: 40.0 60.0
 bedt DATA.seg 40.000000 60.000000 1 66 1 0 >/dev/null
brdc bedtDATA.seg 1 >/dev/null
bimg brdcbedt.seg 0. 20.000000 4.000000 1 30 >/dev/null
bsum bimgbrdc.seg  stak.seg 1.0 >/dev/null
mv bsumbimg.seg stak.seg
bsum BIMGbrdc.seg  STAK.seg 1.0 >/dev/null
mv bsumBIMG.seg STAK.seg
echo gate: 60.0 80.0
 bedt DATA.seg 60.000000 80.000000 1 66 1 0 >/dev/null
brdc bedtDATA.seg 1 >/dev/null
bimg brdcbedt.seg 0. 20.000000 4.000000 1 30 >/dev/null
bsum bimgbrdc.seg  stak.seg 1.0 >/dev/null
mv bsumbimg.seg stak.seg
bsum BIMGbrdc.seg  STAK.seg 1.0 >/dev/null
mv bsumBIMG.seg STAK.seg
```

```
echo gate: 80.0 100.0
 bedt DATA.seg 80.000000 100.000000 1 66 1 0 >/dev/null
brdc bedtDATA.seg 1 >/dev/null
bimg brdcbedt.seg 0. 20.000000 4.000000 1 30 >/dev/null
bsum bimgbrdc.seg  stak.seg 1.0 >/dev/null
mv bsumbimg.seg stak.seg
bsum BIMGbrdc.seg  STAK.seg 1.0 >/dev/null
mv bsumBIMG.seg STAK.seg
bscl stak.seg 1 1 0 0.200000 >/dev/null
mv bsclstak.seg stak.seg
bscl STAK.seg 1 1 0 0.200000 >/dev/null
mv bsclSTAK.seg STAK.seg
bplt stak.seg 1 0 1 1 66 0. 4.000000 1 4. 200
ps2pdf bplt.ps
mv bplt.pdf stak.pdf
rm -f bplt.ps
bplt STAK.seg 1 0 1 1 66 0. 4.000000 1 4. 200
ps2pdf bplt.ps
mv bplt.pdf STAK.pdf
rm -f bplt.ps
```

Type “gobimg” on the command line, and as the program runs, it will output progress to the terminal:

```
gate: 0.0 20.0
gate: 20.0 40.0
gate: 40.0 60.0
gate: 60.0 80.0
gate: 80.0 100.0
```

The two important output files will be:

- **stak.pdf** Shows the causal and acausal parts.
- **STAK.pdf** Mixes the causal and acausal parts (see Figure 117).

WARNING: If you are running on Windows or on an Apple computer, the file system may not be case sensitive. In that case, you should manually edit the script to change the last move “mv” command before running the script. Also, be advised that unless your Windows OS can execute BASH (Bourne Again Shell), you may have a lot of editing to do.

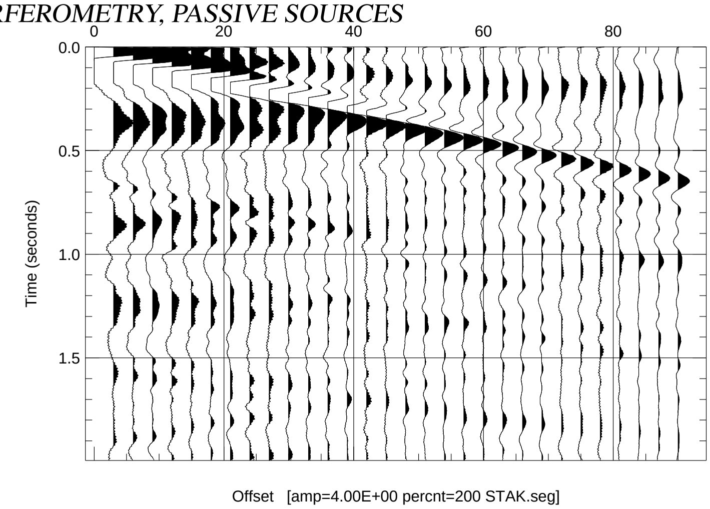


Figure 117: GENBIMG: Output file STAK.seg is the sum of the BIMGxxxx.seg files for the different time windows. Note that the time scale is 0 to 2.0 seconds with zero lag at 0 seconds. **Mix was set to zero**.

EXAMPLE MIX=3:

```
genbimg DATA.seg 0 100.  20.  20.  4.  1 66 1 30 3
```

The result of a light mix (3) is shown in Figure 118. To see the difference, examine the “gobimg” script for lines with calls to BMIX.

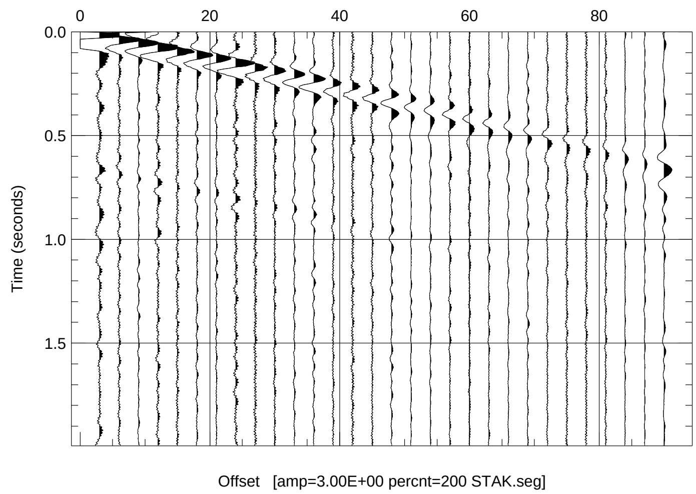


Figure 118: GENBIMG: Output file STAK.seg is the sum of the BIMGxxxx.seg files for the different time windows. Note that the time scale is 0 to 2.0 seconds with zero lag at 0 seconds. **Mix was set to 3**.

The reason a mix can help is that these data were recorded with geophones along the edge of a road (Zhang *et al.*, 2020). Traffic noise provided the signal, and when vehicles are broadside of the line of geophones, there will be broadside arrivals. Alternatively, one can address the problem of azimuthal arrivals in other ways, like applying a signal velocity window.

Pang *et al.* (2019) is an examle of this approach. It requires forming a view on what velocities are signal (ie. Rayleigh waves of interest), and which velocities are not. They also make judgements on the relative value of the causal and acausal waves fields, computing a acausal/causal ratio (ACR). In BIMG, the mix option simply removes infinite velocity, waves striking the array of geophones broadside. Such waves clearly should be excluded. However, how much slower than infinite velocity should one exclude? Such judgements would be aided by two geophone arrays, one along the road, one semi-orthogonal to the road.

If the goal is to do spectral analysis of surface waves from the resulting virtual gather, one can apply the judgement on which apparent velocities are valid by limiting the scan of velocities in the surface wave analysis. The approach in BSU-3.0.3 codes favors this line of thought.

Figure 119 illustrates how this is done using BVAX (7.0.2). Here, we only scan velocities from 100 to 800 m/s. If phase velocities are present outside that range, we don't look for them. This is an alternative to conditioning the gather in Figure 118 by velocity filtering.

The range of offsets excluded near offsets which might be contaminated by near field effects. In addition, since the source will be traffic on the road, offset orthogonally to the line of geophones, near offsets will tend to be sub-broadside, and that leads to apparent horizontal velocities contaminated by azimuthal effects. The mix of 3 only addresses the most orthogonal part of the problem, this offset range limit is additional benefit. As a last comment on this type of survey, one might restrict recordings to include only those from traffic in the near lane to the deployment of geophones.

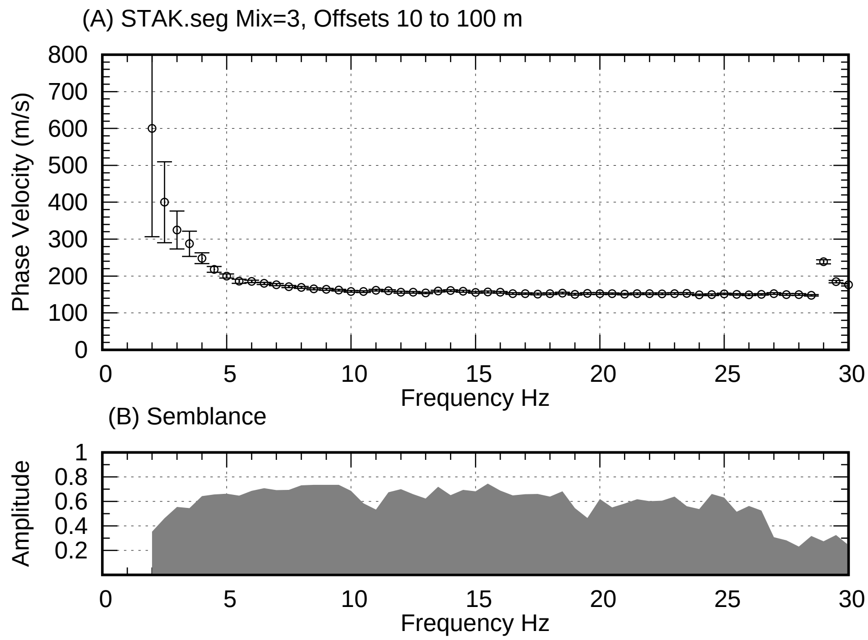


Figure 119: BVAX 7.0.2 applied to data in Figure 118. The range of offsets were 10 to 100 m, velocity search 100 to 800 m/s, frequencies 2 to 30 Hz. Error bars are for 95% confidence.

13.0.4 BAZI

With a passive source, determination of the direction of arrival of waves may be possible using 3-component geophones. Program bazi does this using PCA analyis. While multiple geophones in an array can significantly reduce the uncertainty in the direction of observed wave propogation, multi-component observations still have value. The command line arguments are:

```
bazi   infile chR chT ipct tsw1 tmin tmax

infile =input file name
chR    = channel with R-component (int)
chT    = channel with T-component (int)
ipct   = percent of max amplitude to include (int)
tsw1   = switch to set T-comp relative to R-comp
       T-comp Azimuth= R-Comp + tsw1
       Examples:  tsw1= +90. IF R=0 then T=90 East
                             IF R=90 then T=180 South
                  tsw1= -90. IF R=0 then T=270 West
                             IF R=90 then T=0 North

tmin   = start time (seconds) of data window
tmax   = end time (seconds) of data window

 Computed azimuth to source relative to R-component
 180 Degree uncertainty, angles positive clockwise
```

Illustration of the **bazi** code is shown next using a simple test. Figure 120 shows a simple layout of two 3-component geophones and two source positions. A hammer source was used and recordings made on the author's custom built seismic recorder (employs a Mayhew A/D on an Arduino processor card, SeisRecorder).

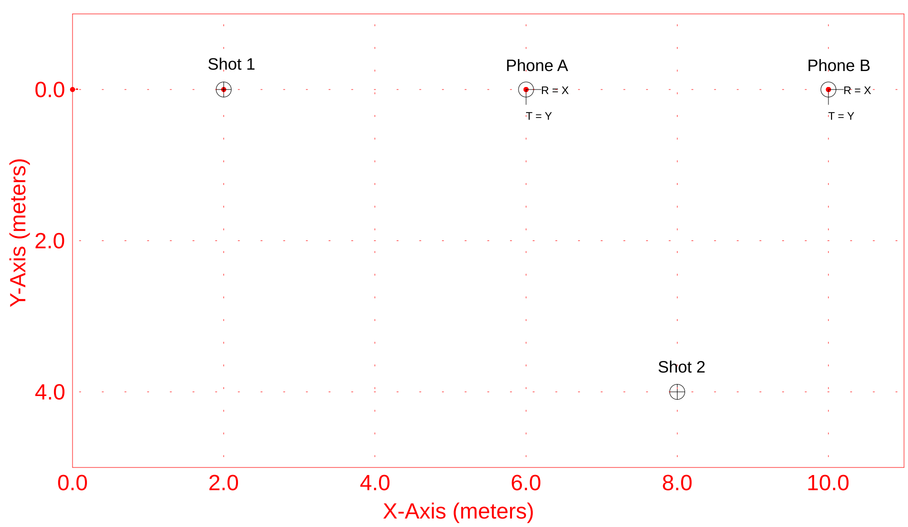


Figure 120: BAZI base map demonstrating program bazi.

Consider the source at shot 2 location. We compare the horizontal motion from the two geophones, A and B. Geophone A employs channels 1 to 3, geophone B employed channels 4 to 6. The channel order was V, R, T for each phone. The commands issued were:

Geophone A
```
bazi brsp0003.seg 2 3 50 90.  0.  .25
```

Geophone B
```
bazi brsp0003.seg 5 6 50 90.  0.  .25
```

The data were recorded at .004 *sec* sample interval. Program **brsp** was used to resample the data to a .001 *sec* sample interval.

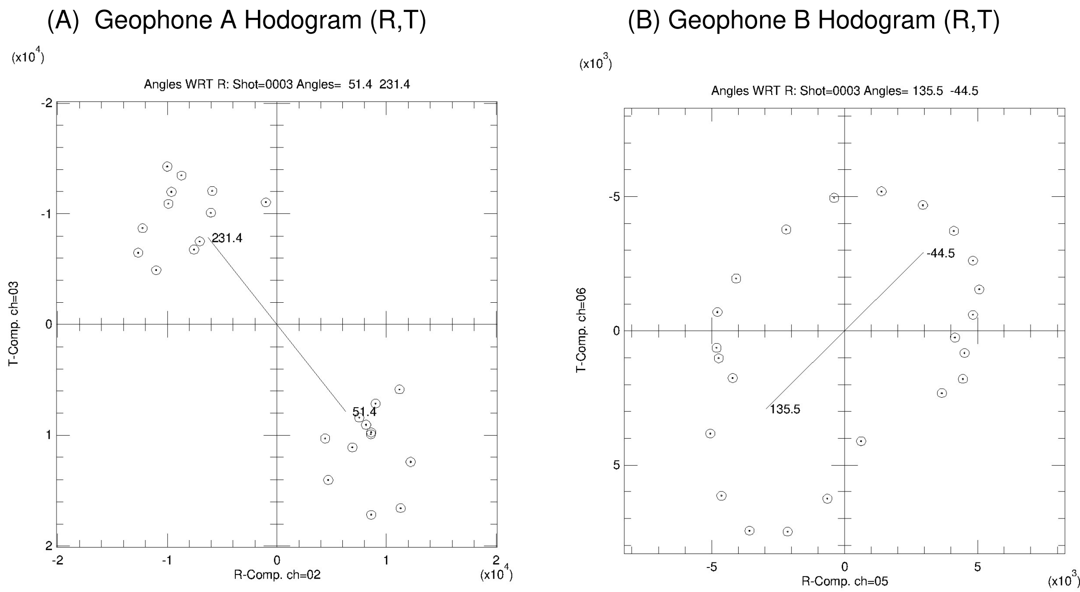


Figure 121: BAZI: Hodograms for geophones A and B, source at location shot 2. See figure 120 for locations. Note that PCA analysis determined angles relative to the R component that point toward the source. For example, the angle 51.4 degrees is measured clockwise from the R-component axis on geophone A. There is a 180 degree ambiguity with PCA analysis.

When using PCA analysis of 3-component data, one should keep in mind that the plant of the geophone is highly relevant. Even a small angular misalignment of the intended orientation will imply significant uncertainty in the source location. This uncertainty increases with the source to geophone offset.

Never-the-less, when considering the general direction to the source, PCA can be quite useful. Consider recording vehicle traffic from the shoulder of a roadway. One can determine a vehicular source location and direction of motion by computing PCA repeatedly in sliding time windows. Combined with the amplitude of the signals, this approach can be used to determine when a vehicle is closest to the geophones.

13.0.5 GENBAZI

When running in a Linux operating system, bash scripts can be used to automate the execution of codes. Program **genbazi** can be used to run a sliding window in time to capture the direction to a seismic passive source (ie. by running **bazi**13.0.4 iteratively.) All arguments must be provided on the command line, or you will get a USAGE message.

```
 USAGE: genbazi filename tstart tend tstep tgate chR chT ipct tsw1

filename = input file name
tstart   = start time (float, seconds) of processing
tend     = end time (float, seconds) of processing)
tstep    = time shift forward with sliding window (float, seconds)
tgate    = length in time of sliding window (float, seconds)
chR      = channel with R-component (int)
chT      = channel with T-component (int)
ipct     = percent of max amplitude to include in window (int)
tsw1     = switch to set T-component relative to R-component
           Example:  tsw1=+90. IF R=90 (East) and T=180 (South)
                     tsw1=-90. IF R=90 (EAST) and T=0 (North)
```

The following is an example of using **genbazi**. Shown in Figure 122 is the deployment of 2 three component geophones orthogonal to a roadway. Signals from a large box truck were captured during a recording on the 6 channels.

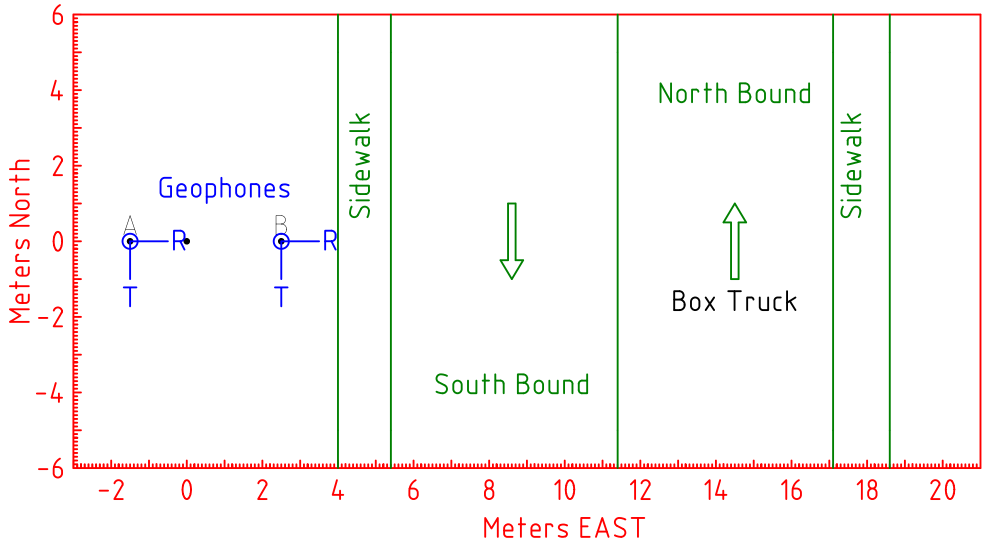


Figure 122: GENBAZI: Layout of geophones that recorded signals from a box truck traveling north.

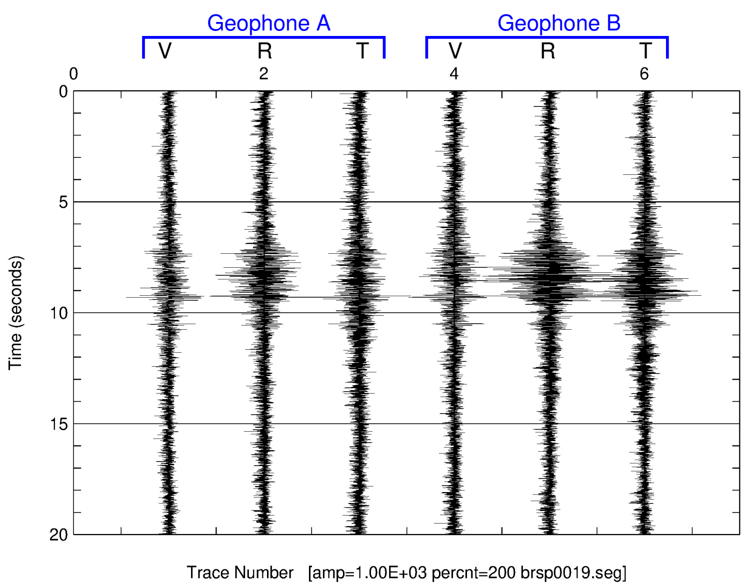


Figure 123: GENBAZI: Data recorded for layout shown in Figure 122. Data were recorded on 04 October 2021. Road surface was asphalt.

From the large amplitudes between 5 and 10 seconds, we interpret the truck is passing the orthogonal array of two 3-C geophones. We can run **genbazi** to illustrate this passage with additional clarity by combining both normalized amplitude and angle with respect to the R-component. The following command was issued using signals from geophone B:

```
genbazi brsp0019.seg 0.  20.  1.  1.  5 6 60 90
```

The data were interpolated from a .004 second sample interval to .001 seconds. The method is augmentation by zeros in the frequency domain and adds no additional frequencies. See program **BRSP** 11.0.3 for more. The file **gobazi** was generated and then run. Output files include gnuplot file **PCA.gp** which generates **PCA.ps**, a plot of both amplitude and angle with respect to R-component. The file, **bazipca.dat** contains the quadruplets that the GNUPLOT script plots. Amplitude is normalized in a way that permits plotting both angles and amplitude on the same axes. The following is the GNUPLOT script, **PCA.gp**

```
# GENERATED BY GENBAZI.C
# Channels: R=5  T=6
set grid
set terminal x11 persist
set key left
set xlabel 'Time (s)'
set title 'brsp0019.seg '
stats 'bazipca.dat' u 4
p 'bazipca.dat' u 1:3 w l lc 'red' t 'angle wrt R',\
 'bazipca.dat' u 1:($4/STATS_max*200 ) \
 w p pt 6 ps 1.5 lc 'blue'  t 'normalized amp'
set terminal postscript enhanced color
set output 'PCA.ps'
replot $
```

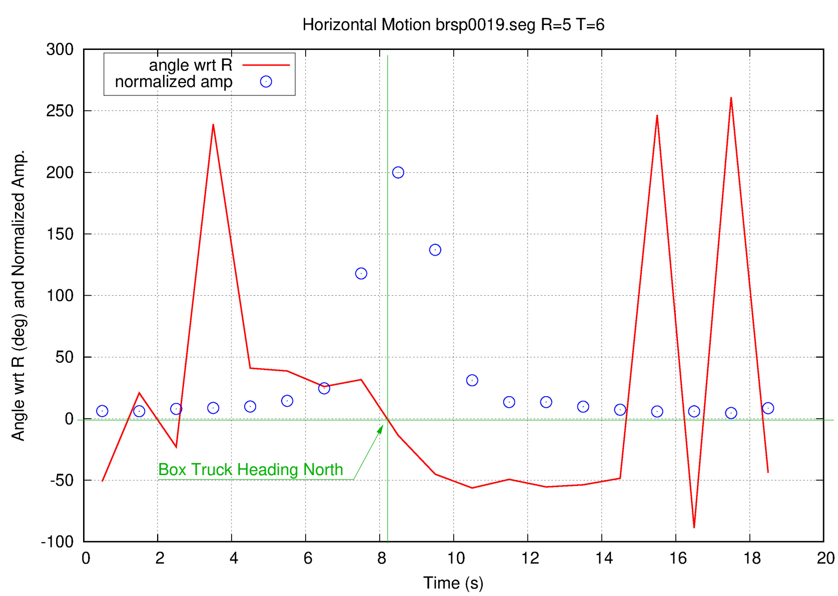


Figure 124: GENBAZI: Result of running **gobazi** bash script generated by the **genbazi** command above. Note that the angle (plotted in red) crosses zero at about 8.5 seconds, and this agrees well with the normalized amplitude on the horizontal components. Assuming a geophone to truck distance of 12 meters, the slope of the angle with time suggests a vehicle speed of about 18 mph.

13.0.6 BZRT

Programs **BAZI** 13.0.4 and **BZRT** can be used to study the particle motion recorded by 3-component geophones. Depending on the type of wave present, this may be useful in either identification of the wave or sorting out properties of the soil profile for further study. The user is responsible for understanding the wave type (surface, body, or refracted waves etc.) when drawing conclusions. Program **BZRT** is the code for studying motion in the vertical plane. The command line arguments are:

```
bzrt   infile chV chH ipct hsw1 tmin tmax

 infile =input file name
 chV    = channel with V-component (int)
 chH    = channel with R or T-component (int)
 ipct   = percent of max amplitude to include (int)
 hsw1   = switch to set horizontal component ID
          2 = Horizontal is R-component
          3 = Horizontal is T-component
    (hsw1 sets label on horizontal axis of plot)

 tmin   = start time (seconds) of data window
 tmax   = end time (seconds) of data window
```

For example, let's look at the vertical motion as the box truck passes heading North. See figures 123 (time series data) and 124 (PCA analysis in horizontal plane). We focus on the 8.0 to 9.0 time frame when the truck is passing the geophones. We issue the command:

```
bzrt brsp0019.seg 4 5 60 2 8.0 9.0
```

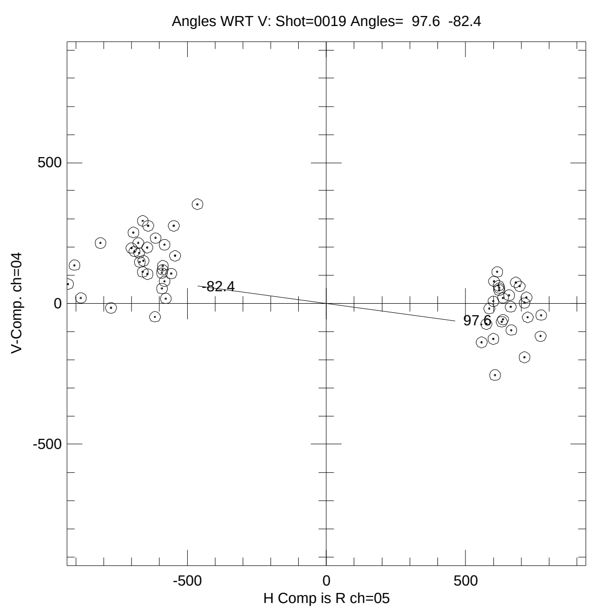


Figure 125: BZRT: Motion in the vertical plane as truck passes the geophones in the 8.0 to 9.0 time interval. Note that the polarization ellipse major axis is essentially horizontal. One does not know for sure what type of wave motion has been captured. It is possible that we are looking at more than one mode of Rayleigh wave.

13.0.7 GENBZRT

As was the case in horizontal plane motion (**GENBAZI** 13.0.5), one can also generate a bash script to explore the motion in the vertical plane through a sliding window. Program **GENBZRT** requires a complete command line list of arguments to run. Only typing the program name produces a list of the arguments and then aborts execution.

```
USAGE: genbzrt filename tstart tend tstep tgate chV chH ipct hsw1

filename = input file name
tstart   = start time (float, seconds) of processing
tend     = end time (float, seconds) of processing)
tstep    = time shift forward with sliding window (float, seconds) of processing)
tgate    = length in time of sliding window (float, seconds)
chV      = channel with V-component (int)
chH      = channel with H-component (int)
ipct     = percent of max amplitude to include in window (int)
hsw1     = switch to set Horizontal Component ID
           Example:  hsw1=2    IF Horizontal component is labeled R-component
                     hsw1=3    IF Horizontal component is labeled T-component
         IMPORTANT:  Make sure that hsw1 and chH are in agreement
```

Applying this program to the box truck data of Figure 123 we can issue the following command:

```
genbzrt brsp0019.seg 0.  20.  1.0 1.0 4 5 60 2
```

which generates a bash script, **gobzrt**. We then run the bash script, **gobzrt** to get a sense of the vertical plane motion as a function of time (Figure 126).

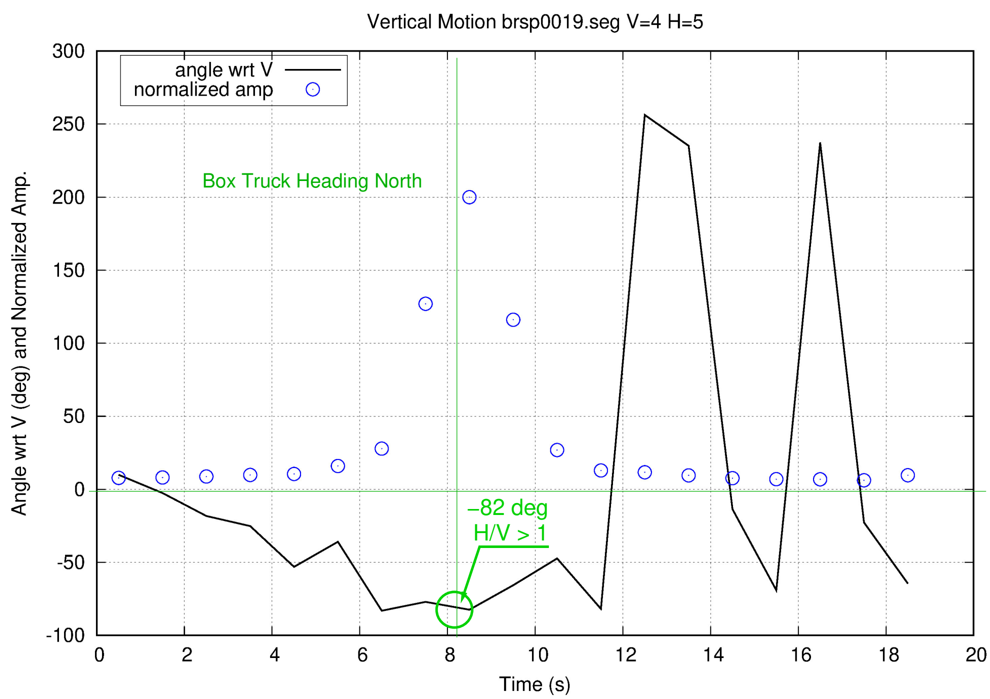


Figure 126: GENBZRT: As the truck passes the geophones, recorded waves are stronger on the horizontal rather than the vertical component. To explore this in greater detail, consider running program **HVSR**.

13.0.8 HVSR

This program can be used to plot the horizontal to vertical spectra ratio (HVSR). Computing spectra presents a number of challenges by itself, and understanding the type of waves present in the analysis is relevant to interpretation of the results. With regard to computing the spectra, one has two choices. The input data can be either of the following:

- **Seismic Signals** These are the recorded time series of the vertical and horizontal components from a 3-component geophone. These signals need to be in the same data file with the user specifying the channels.
- **Autocorrelations** These are the autocorrelations computed from the time series. The reason for this option is that autocorrelations can be stacked (summed) to improve the desired signal to noise.

The spectra are computed using an all-pole, Yule-Walker representation. The plots are generated by evaluating the Z-transform on the unit circle at a fine sampling (compared to that of the DFT). See the source code for alternative ways of setting sampling. One must decide on the order of the process as expressed by the number of samples to be included in the autocorrelations (whether they are internally computed autocorrelations of signals or those autocorrelations directly input).

13.0.8.1 Autocorrelations If one chooses to do the autocorrelations outside of **HVSR**, it would likely be to improve signal to noise. This opportunity may be when processing multi-offset recordings in a span of distance which is believed to be uniform in terms of the soil profile (ie. 1-D). Programs that may be helpful include the following:

- **BXCR**12.0.16 While this code does cross correlations between two files, if you specify the same file name twice on the input, it will do an autocorrelation.
- **BSUM**12.1.4 This will add two files together. One can combine autocorrelations from different geophones or from different sources by adding them together (stacking). A scale factor for adding can be used to create equal weighting of autocorrelations.
- **BEXT**11.0.5 In cases were a number of files have been concatenated together, one can extract traces based on header values, like shot name, receiver name, or field recored number.

The program **HVSR** can be run using command line arguments or by being prompted. The online help is:

```
hvsr   infile chV chH tmin tmax mpts idata wht

 infile  =input file name
 chV     = channel with vertical
 chH     = channel with horizontal
 tmin    = start of time window
 tmax    = end time window
 mpts    = length (lags) of autocorrelation (1-sided)
 idata   = selects type of input data
         = 0 Time series of seismic data
         = 1 Autocorrelations
 wht     = whitening to bias against notches
         = percent of zero lag auto to add to zero lag auto
```

While external autocorrelations computed by **BXCR** are two sided, the user specifies the 1-sided length (the program will sort it out with the idata switch). The computation of a ratio presents the risk of division by zero. The parameter **wht** helps avoid that possiblility. Biasing the spectra is achieved by adding a small value to the zero lag of the autocorrelation. The following example illustrates a reasonable value for this parameter.

Continuing with the our example data set, we can issue the command:

```
hvsr brsp0019.seg 4 5 8.  9.  60 0 .01
```

The parameter **mpts**=60. The sample interval is .001 seconds making this choice .06 seconds. Choosing this value (essentially the order of the process) is done by trial and error. Start by using a large value and then make a choice by deciding on a shorter length where the autocorrelations have decayed. A good choice will pick a point where two autocorrelations end at a point close to zero amplitude. Figure 127 shows the result. Note that the largest H/V ratio appears between 60 and 80 Hz.

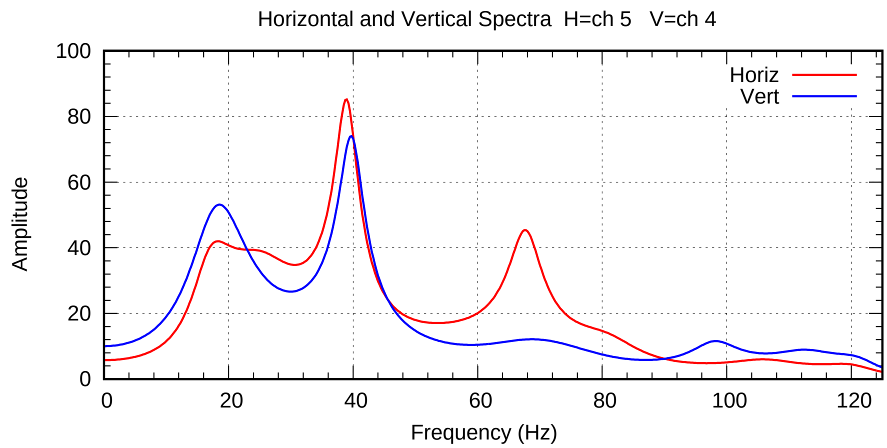


Figure 127: HVSR: We focus on the time interval from 8.0 to 9.0 seconds when the truck passes the geophones. The data are time series recorded from phone B (see figure 122). The instrument analog low pass filter has a cut-off frequency of 100 Hz.

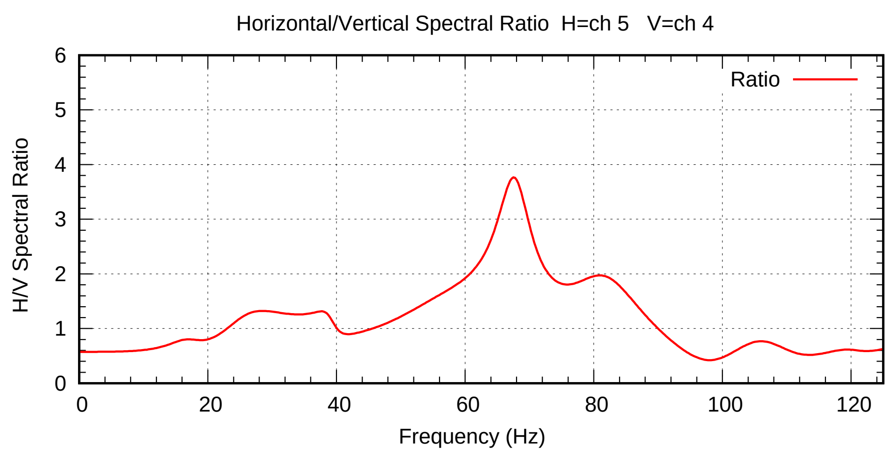


Figure 128: HVSR: The ratio between horizontal and vertical components (H/V) is computed from the spectra shown in Figure 127. Note that below about 20 Hz, the vertical motion dominates.

13.0.8.2 Caution: To get a sense of the vertical motion at selected frequencies the data were band-pass filtered with narrow band (0.4 Hz) filters. Then hodograms were plot using the **BZRT** program. The results are shown in Figure 129. When working with passive data, one should expect challenges in choosing parameters like the time interval to focus on and filter bandwidth. Comparison of figure 128 with 129 **appear** to show good agreement. However, with narrow pass-band filters, there is little temporal resolution, if any. **If multiple modes are present in the time window, then particle motion at a single frequency will be a mixture of motions.** For a discussion on multi-channel recordings and modes, see Mi *et al.* (2019).

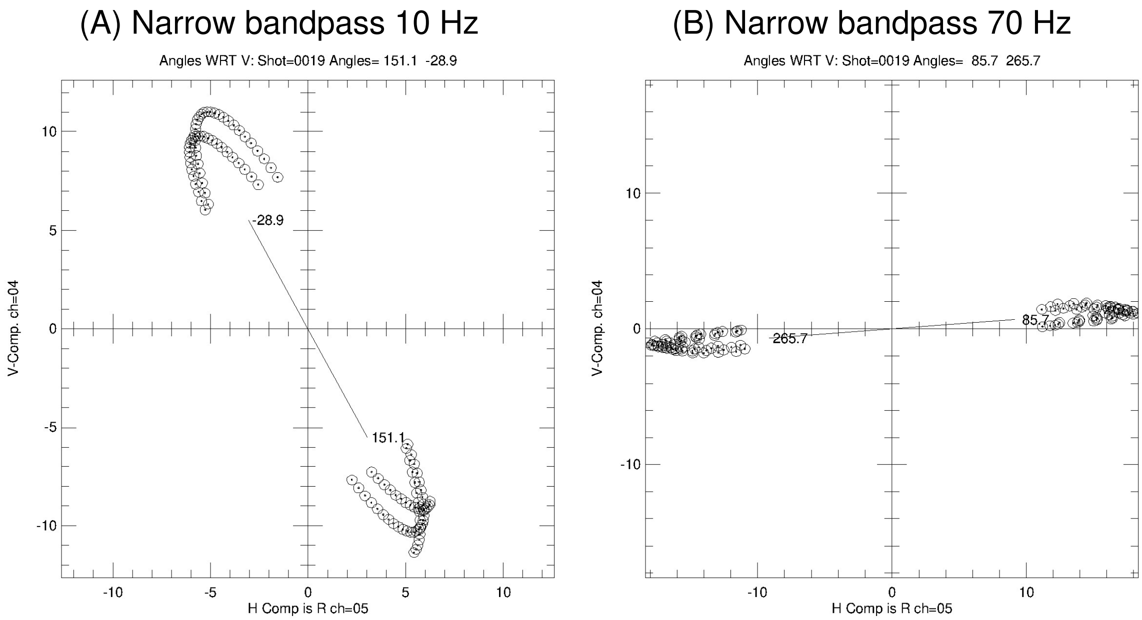


Figure 129: BZRT: Hodograms of narrow band-pass filtered data. Time interval 8.4 to 8.6 seconds. Zero-phase 18 pole filters (bandwidth 0.4 hz) with center frequencies of 10 and 70 Hz. Compare these plots with the HVSR ratio in figure 128.

References

Aki, K., & Richards, P.G. 1980. *Quantitative Seismology Vol. 1*. Vol. 1. W.H. Freeman and Co. 557p.

Lamb, H. 1904. On the propagation of tremors over the surface of an elastic solid. *Phil. Tran. Royal Society of London*, **Series A**(203), 1–42.

Menke, W. 1989. *Geophysical data analysis, discrete inverse theory*. Academic Press. San Diego 289pgs.

Mi, Binbin, Hu, Yue, Xia, Jianghai, & Socco, Laura Valentina. 2019. Estimation of horizontal-to-vertical spectral ratios (ellipticity) of Rayleigh waves from multistation active-seismic records. *Geophysics*, **84**(6), EN81–EN92.

Michaels, P. 1995. A geophysical site investigation for a bridge foundation in a narrow canyon. *Environmental & Engineering Geoscience*, **1**(2), 219–226.

Michaels, P. 1998. In situ determination of soil stiffness and damping. *Journal of Geotechnical and Geoenvironmental Engineering*, **24**(8), 709–719.

Michaels, P. 1999. Use of engineering geophysics in the design of highway passing lanes. *Proceedings of the Symposium on the Application of Geophysics to Engineering and Environmental Problems, SAGEEP99*, 179–187.

Michaels, P. 2001a. Use of Engineering Geophysics to Investigate a Site for a Bridge Foundation. *Foundations and Ground Improvement, T. L. Brandon editor GSP113, ASCE*, 715–727.

Michaels, P. 2001b. Use of principal component analysis to determine down-hole tool orientation and enhance SH-waves. *Journal of Environmental and Engineering Geophysics*, **6**(4), 175–183.

Michaels, P. 2006. Relating Damping to Soil Permeability. *International Journal of Geomechanics*, **6**(3), 158–165.

Michaels, P. 2014. Alternative in analysis of the UTexas1 Surface Wave Dataset. *Geo-Congress 2014, Technical Papers and Keynote Lectures*, **GSP 234-235**, 761–772.

Michaels, P. 2018. An alternative representation of the soil profile for MASW analysis. *Proceedings of the Symposium on the Application of Geophysics to Engineering and Environmental Problems*, **SAGEEP2018**(March), 286–289.

Mooney, H.M. 1974. Some numerical solutions for Lamb's problem. *Bulletin of the Seismological Society of America*, **64**(2), 473–491.

Pang, Jingyin, Cheng, Feng, Shen, Chao, Dai, Tianyu, Ning, Ling, & Zhang, Kai. 2019. Automatic passive data selection in time domain for imaging near-surface surface waves. *Journal of Applied Geophysics*, **162**(Dec.), 108–117.

Press, W.H., Flannery, B.P., Teukolsky, S.A., & Vetterling, W.T. 1989. *Numerical Recipes, The art of scientific computing Fortran version*. Cambridge University Press. 702p.

Pullan, S. E. 1990. Recommended standard for seismic (/radar) files in the personal computer environment. *Geophysics*, **55**(09), 1260–1271.

Robinson, E.A. 1967. *Multichannel Time Series Analysis with Digital Computer Programs*. Holden-Day. 298p.

Sheriff, R.E. 1991. *Encyclopedic dictionary of exploration geophysics*. Society of Exploration Geophysics.

Yilmaz, Oz, Gao, Kai, Delic, Milos, Xia, Jianghai, Jodeiri, Hossein, & Pugin, Andre. 2022. A reality check on full-wave inversion applied to land seismic data for near-surface modeling. *The Leading Edge*, **41**(1), 40–46.

Zhang, Kai, Li, Hongyi, Wang, Xiaojiang, & Wang, Kai. 2020. Retrieval of shallow S-wave profiles from seismic reflection surveying and traffic-induced noise. *Geophysics*, **85**(6), EN105–EN117.

Index

14 IBM LICENSE

The following license was included with the library downloaded in the archive, **libascii.tar.Z**. The only function from this library included in BUS is file, xdrfloa.c, which is used to make BUS library libibm.a.

<http://www-03.ibm.com/systems/z/os/zos/features/unix/libascii.html

15 GNU General Public License

```
                    GNU GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 Copyright (C) 2007 Free Software Foundation, Inc. <https://fsf.org/>
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                            Preamble

  The GNU General Public License is a free, copyleft license for
software and other kinds of works.

  The licenses for most software and other practical works are designed
to take away your freedom to share and change the works.  By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users.  We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors.  You can apply it to
your programs, too.

  When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

  To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights.  Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

  For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received.  You must make sure that they, too, receive
or can get the source code.  And you must show them these terms so they
know their rights.

  Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

  For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software.  For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

  Some devices are designed to deny users access to install or run
modified versions of the software inside them, although the manufacturer
can do so.  This is fundamentally incompatible with the aim of
```

protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its

```
content, constitutes a covered work.  This License acknowledges your
rights of fair use or other equivalent, as provided by copyright law.

  You may make, run and propagate covered works that you do not
convey, without conditions so long as your license otherwise remains
in force.  You may convey covered works to others for the sole purpose
of having them make modifications exclusively for you, or provide you
with facilities for running those works, provided that you comply with
the terms of this License in conveying all material for which you do
not control copyright.  Those thus making or running the covered works
for you must do so exclusively on your behalf, under your direction
and control, on terms that prohibit them from making any copies of
your copyrighted material outside their relationship with you.

  Conveying under any other circumstances is permitted solely under
the conditions stated below.  Sublicensing is not allowed; section 10
makes it unnecessary.

  3. Protecting Users' Legal Rights From Anti-Circumvention Law.

  No covered work shall be deemed part of an effective technological
measure under any applicable law fulfilling obligations under article
11 of the WIPO copyright treaty adopted on 20 December 1996, or
similar laws prohibiting or restricting circumvention of such
measures.

  When you convey a covered work, you waive any legal power to forbid
circumvention of technological measures to the extent such circumvention
is effected by exercising rights under this License with respect to
the covered work, and you disclaim any intention to limit operation or
modification of the work as a means of enforcing, against the work's
users, your or third parties' legal rights to forbid circumvention of
technological measures.

  4. Conveying Verbatim Copies.

  You may convey verbatim copies of the Program's source code as you
receive it, in any medium, provided that you conspicuously and
appropriately publish on each copy an appropriate copyright notice;
keep intact all notices stating that this License and any
non-permissive terms added in accord with section 7 apply to the code;
keep intact all notices of the absence of any warranty; and give all
recipients a copy of this License along with the Program.

  You may charge any price or no price for each copy that you convey,
and you may offer support or warranty protection for a fee.

  5. Conveying Modified Source Versions.

  You may convey a work based on the Program, or the modifications to
produce it from the Program, in the form of source code under the
terms of section 4, provided that you also meet all of these conditions:

    a) The work must carry prominent notices stating that you modified
```

```
    it, and giving a relevant date.

    b) The work must carry prominent notices stating that it is
    released under this License and any conditions added under section
    7.  This requirement modifies the requirement in section 4 to
    "keep intact all notices".

    c) You must license the entire work, as a whole, under this
    License to anyone who comes into possession of a copy.  This
    License will therefore apply, along with any applicable section 7
    additional terms, to the whole of the work, and all its parts,
    regardless of how they are packaged.  This License gives no
    permission to license the work in any other way, but it does not
    invalidate such permission if you have separately received it.

    d) If the work has interactive user interfaces, each must display
    Appropriate Legal Notices; however, if the Program has interactive
    interfaces that do not display Appropriate Legal Notices, your
    work need not make them do so.

  A compilation of a covered work with other separate and independent
works, which are not by their nature extensions of the covered work,
and which are not combined with it such as to form a larger program,
in or on a volume of a storage or distribution medium, is called an
"aggregate" if the compilation and its resulting copyright are not
used to limit the access or legal rights of the compilation's users
beyond what the individual works permit.  Inclusion of a covered work
in an aggregate does not cause this License to apply to the other
parts of the aggregate.

  6. Conveying Non-Source Forms.

  You may convey a covered work in object code form under the terms
of sections 4 and 5, provided that you also convey the
machine-readable Corresponding Source under the terms of this License,
in one of these ways:

    a) Convey the object code in, or embodied in, a physical product
    (including a physical distribution medium), accompanied by the
    Corresponding Source fixed on a durable physical medium
    customarily used for software interchange.

    b) Convey the object code in, or embodied in, a physical product
    (including a physical distribution medium), accompanied by a
    written offer, valid for at least three years and valid for as
    long as you offer spare parts or customer support for that product
    model, to give anyone who possesses the object code either (1) a
    copy of the Corresponding Source for all the software in the
    product that is covered by this License, on a durable physical
    medium customarily used for software interchange, for a price no
    more than your reasonable cost of physically performing this
    conveying of source, or (2) access to copy the
    Corresponding Source from a network server at no charge.
```

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the

```
Corresponding Source conveyed under this section must be accompanied
by the Installation Information.  But this requirement does not apply
if neither you nor any third party retains the ability to install
modified object code on the User Product (for example, the work has
been installed in ROM).

  The requirement to provide Installation Information does not include a
requirement to continue to provide support service, warranty, or updates
for a work that has been modified or installed by the recipient, or for
the User Product in which it has been modified or installed.  Access to a
network may be denied when the modification itself materially and
adversely affects the operation of the network or violates the rules and
protocols for communication across the network.

  Corresponding Source conveyed, and Installation Information provided,
in accord with this section must be in a format that is publicly
documented (and with an implementation available to the public in
source code form), and must require no special password or key for
unpacking, reading or copying.

  7. Additional Terms.

  "Additional permissions" are terms that supplement the terms of this
License by making exceptions from one or more of its conditions.
Additional permissions that are applicable to the entire Program shall
be treated as though they were included in this License, to the extent
that they are valid under applicable law.  If additional permissions
apply only to part of the Program, that part may be used separately
under those permissions, but the entire Program remains governed by
this License without regard to the additional permissions.

  When you convey a copy of a covered work, you may at your option
remove any additional permissions from that copy, or from any part of
it.  (Additional permissions may be written to require their own
removal in certain cases when you modify the work.)  You may place
additional permissions on material, added by you to a covered work,
for which you have or can give appropriate copyright permission.

  Notwithstanding any other provision of this License, for material you
add to a covered work, you may (if authorized by the copyright holders of
that material) supplement the terms of this License with terms:

    a) Disclaiming warranty or limiting liability differently from the
    terms of sections 15 and 16 of this License; or

    b) Requiring preservation of specified reasonable legal notices or
    author attributions in that material or in the Appropriate Legal
    Notices displayed by works containing it; or

    c) Prohibiting misrepresentation of the origin of that material, or
    requiring that modified versions of such material be marked in
    reasonable ways as different from the original version; or

    d) Limiting the use for publicity purposes of names of licensors or
```

authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of

```
this License.

  Each contributor grants you a non-exclusive, worldwide, royalty-free
patent license under the contributor's essential patent claims, to
make, use, sell, offer for sale, import and otherwise run, modify and
propagate the contents of its contributor version.

  In the following three paragraphs, a "patent license" is any express
agreement or commitment, however denominated, not to enforce a patent
(such as an express permission to practice a patent or covenant not to
sue for patent infringement).  To "grant" such a patent license to a
party means to make such an agreement or commitment not to enforce a
patent against the party.

  If you convey a covered work, knowingly relying on a patent license,
and the Corresponding Source of the work is not available for anyone
to copy, free of charge and under the terms of this License, through a
publicly available network server or other readily accessible means,
then you must either (1) cause the Corresponding Source to be so
available, or (2) arrange to deprive yourself of the benefit of the
patent license for this particular work, or (3) arrange, in a manner
consistent with the requirements of this License, to extend the patent
license to downstream recipients.  "Knowingly relying" means you have
actual knowledge that, but for the patent license, your conveying the
covered work in a country, or your recipient's use of the covered work
in a country, would infringe one or more identifiable patents in that
country that you have reason to believe are valid.

  If, pursuant to or in connection with a single transaction or
arrangement, you convey, or propagate by procuring conveyance of, a
covered work, and grant a patent license to some of the parties
receiving the covered work authorizing them to use, propagate, modify
or convey a specific copy of the covered work, then the patent license
you grant is automatically extended to all recipients of the covered
work and works based on it.

  A patent license is "discriminatory" if it does not include within
the scope of its coverage, prohibits the exercise of, or is
conditioned on the non-exercise of one or more of the rights that are
specifically granted under this License.  You may not convey a covered
work if you are a party to an arrangement with a third party that is
in the business of distributing software, under which you make payment
to the third party based on the extent of your activity of conveying
the work, and under which the third party grants, to any of the
parties who would receive the covered work from you, a discriminatory
patent license (a) in connection with copies of the covered work
conveyed by you (or copies made from those copies), or (b) primarily
for and in connection with specific products or compilations that
contain the covered work, unless you entered into that arrangement,
or that patent license was granted, prior to 28 March 2007.

  Nothing in this License shall be construed as excluding or limiting
any implied license or other defenses to infringement that may
otherwise be available to you under applicable patent law.
```

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program’s name and a brief idea of what it does.>
Copyright (C) <year>  <name of author>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.
```

You should have received a copy of the GNU General Public License
along with this program. If not, see <https://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type 'show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type 'show c' for details.

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <https://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <https://www.gnu.org/licenses/why-not-lgpl.html>.

16 Free Documentation License

GNU Free Documentation License
Version 1.3, 3 November 2008

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified

```
   Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for
   public access to a Transparent copy of the Document, and likewise
   the network locations given in the Document for previous versions
   it was based on.  These may be placed in the "History" section.
   You may omit a network location for a work that was published at
   least four years before the Document itself, or if the original
   publisher of the version it refers to gives permission.
K. For any section Entitled "Acknowledgements" or "Dedications",
   Preserve the Title of the section, and preserve in the section all
   the substance and tone of each of the contributor acknowledgements
   and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document,
   unaltered in their text and in their titles.  Section numbers
   or the equivalent are not considered part of the section titles.
M. Delete any section Entitled "Endorsements".  Such a section
   may not be included in the Modified Version.
N. Do not retitle any existing section to be Entitled "Endorsements"
   or to conflict in title with any Invariant Section.
O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or
appendices that qualify as Secondary Sections and contain no material
copied from the Document, you may at your option designate some or all
of these sections as invariant.  To do this, add their titles to the
list of Invariant Sections in the Modified Version's license notice.
These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains
nothing but endorsements of your Modified Version by various
parties--for example, statements of peer review or that the text has
been approved by an organization as the authoritative definition of a
standard.

You may add a passage of up to five words as a Front-Cover Text, and a
passage of up to 25 words as a Back-Cover Text, to the end of the list
of Cover Texts in the Modified Version.  Only one passage of
Front-Cover Text and one of Back-Cover Text may be added by (or
through arrangements made by) any one entity.  If the Document already
includes a cover text for the same cover, previously added by you or
by arrangement made by the same entity you are acting on behalf of,
you may not add another; but you may replace the old one, on explicit
permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License
give permission to use their names for publicity for or to assert or
imply endorsement of any Modified Version.


5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this
License, under the terms defined in section 4 above for modified
versions, provided that you include in the combination all of the
```

Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

```
8. TRANSLATION

Translation is considered a kind of modification, so you may
distribute translations of the Document under the terms of section 4.
Replacing Invariant Sections with translations requires special
permission from their copyright holders, but you may include
translations of some or all Invariant Sections in addition to the
original versions of these Invariant Sections.  You may include a
translation of this License, and all the license notices in the
Document, and any Warranty Disclaimers, provided that you also include
the original English version of this License and the original versions
of those notices and disclaimers.  In case of a disagreement between
the translation and the original version of this License or a notice
or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements",
"Dedications", or "History", the requirement (section 4) to Preserve
its Title (section 1) will typically require changing the actual
title.


9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document
except as expressly provided under this License.  Any attempt
otherwise to copy, modify, sublicense, or distribute it is void, and
will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license
from a particular copyright holder is reinstated (a) provisionally,
unless and until the copyright holder explicitly and finally
terminates your license, and (b) permanently, if the copyright holder
fails to notify you of the violation by some reasonable means prior to
60 days after the cessation.

Moreover, your license from a particular copyright holder is
reinstated permanently if the copyright holder notifies you of the
violation by some reasonable means, this is the first time you have
received notice of violation of this License (for any work) from that
copyright holder, and you cure the violation prior to 30 days after
your receipt of the notice.

Termination of your rights under this section does not terminate the
licenses of parties who have received copies or rights from you under
this License.  If your rights have been terminated and not permanently
reinstated, receipt of a copy of some or all of the same material does
not give you any rights to use it.


10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the
```

```
GNU Free Documentation License from time to time.  Such new versions
will be similar in spirit to the present version, but may differ in
detail to address new problems or concerns.  See
https://www.gnu.org/licenses/.

Each version of the License is given a distinguishing version number.
If the Document specifies that a particular numbered version of this
License "or any later version" applies to it, you have the option of
following the terms and conditions either of that specified version or
of any later version that has been published (not as a draft) by the
Free Software Foundation.  If the Document does not specify a version
number of this License, you may choose any version ever published (not
as a draft) by the Free Software Foundation.  If the Document
specifies that a proxy can decide which future versions of this
License can be used, that proxy's public statement of acceptance of a
version permanently authorizes you to choose that version for the
Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any
World Wide Web server that publishes copyrightable works and also
provides prominent facilities for anybody to edit those works.  A
public wiki that anybody can edit is an example of such a server.  A
"Massive Multiauthor Collaboration" (or "MMC") contained in the site
means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0
license published by Creative Commons Corporation, a not-for-profit
corporation with a principal place of business in San Francisco,
California, as well as future copyleft versions of that license
published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in
part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this
License, and if all works that were first published under this License
somewhere other than this MMC, and subsequently incorporated in whole or
in part into the MMC, (1) had no cover texts or invariant sections, and
(2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site
under CC-BY-SA on the same site at any time before August 1, 2009,
provided the MMC is eligible for relicensing.


ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of
the License in the document and put the following copyright and
license notices just after the title page:

    Copyright (c)  YEAR  YOUR NAME.
```

```
    Permission is granted to copy, distribute and/or modify this document
    under the terms of the GNU Free Documentation License, Version 1.3
    or any later version published by the Free Software Foundation;
    with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
    A copy of the license is included in the section entitled "GNU
    Free Documentation License".
```

```
If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts,
replace the "with...Texts." line with this:
```

```
    with the Invariant Sections being LIST THEIR TITLES, with the
    Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

```
If you have Invariant Sections without Cover Texts, or some other
combination of the three, merge those two alternatives to suit the
situation.
```

```
If your document contains nontrivial examples of program code, we
recommend releasing these examples in parallel under your choice of
free software license, such as the GNU General Public License,
to permit their use in free software.
```