Program: ICORRECT

Karen Assatourians and Gail Atkinson

Engineering Seismology Toolbox (**www.seismotoolbox.ca**): June 2008

Introduction:

Program "ICORRECT" was developed to process seismic data files in SEED format (Geological Survey of Canada convention) to obtain instrument-corrected acceleration time series and corresponding Pseudo Spectral Acceleration (PSAs).

ICORRECT is written in FORTRAN and compiled using Compaq Visual Fortran. The application works in Windows and DOS environments. The code carries some of the subroutines (some are modified) developed by G. Atkinson for the program AGRAM, as well as subroutines and functions written by D. Boore for the program SMSIM (Boore, 1996). To run this program two other applications "rdseed.exe" and "dirf.exe" should be in the path and accessible.

ICORRECT reads a SEED data file and creates a data structure in the form of folders, with a folder for the event file and sub-folders for every channel of data available in the SEED file. In the next step it puts uncorrected time series and instrument response information (in a set of formats: SEED, SAC, GSE2, and SAC) of each channel in the corresponding folder. Then, based on a set of user defined control parameters it applies a sequence of processing stages including: deglitching, de-trending, windowing, filtering, and instrument correction; to obtain ground motion acceleration time series. Finally it calculates the response spectra of the instrument-corrected acceleration time series and applies a simple smoothing algorithm for better visualization of the obtained spectra. All these information, including time series, instrument response parameters in various formats, products of data processing in each stage, and response spectral information of each channel are stored in the corresponding subdirectory of the channel in the directory of the event under study.

ICORRECT can be modified (or improved) for handling any kind of SEED data files provided by any agency around the world. The specific implementation of ICORRECT given here is for data downloaded from the Geological Survey of Canada

(GSC). This limitation to GSC data is because of the convention that GSC is following for treating instrument response information in SEED volumes; note that different agencies follow different conventions, with some providing poles and zeroes for displacement response, and some for velocity response, for example.

The different elements of the ICORRECT program were validated against some well-accepted applications' outputs; however this program, as any other, should be tested by users to make sure all possible problems and bugs are captured and fixed. The instrument response calculation subroutine output shows a perfect match with outputs from "PITSA" (Programmable Interactive Toolbox for Seismological Analysis", "Resp" (instrument response calculation program in SEISAN package) and the IRIS program "JPlotResp". The time series after instrument correction show a very close match with the time series obtained after instrument correction by "SAC" and "SEISAN". Also, the response spectrum outputs of ICORRECT show a good match with response spectra obtained by the "SPECEQ" program from the NISEE software library. The intermediate processing stages are tested visually after development of useful graphs for visual inspection. The current version of the ICORRECT program should be considered the "beta version"; there may be future improvements.

Program structure:

The ICORRECT program performs a sequence of processes to obtain ground motion acceleration time series and corresponding response spectra from SEED files archived by GSC (Geological Survey of Canada). This program is written in FORTRAN compatible with "Compaq Visual Fortran" and compiled for use in Windows and DOS environments. The structure of the program is shown in Figure 1. In the first stage, the program reads the SEED data file as well as a program parameter file (parameter.dat) and runs the IRIS program RDSEED to get the information of all channels. Based on the number of the channels of available data, the program creates a directory structure and copies the SAC-ASCII data file as well as the SEED response file of each channel in the corresponding subdirectory (Figure 2 demonstrates a test sample). In the next stage the program goes into the subdirectory of each individual channel and reads data and response files. The program applies the following to the SEED response file: reformatting to SAC, reformatting to GSE2, reformatting to PITSA, calculating the instrument response function, and deriving frequency/amplitude/phase response curve. It also applies the following modifications to the data file:

1-Removing glitches (by subroutine: deglitcher)

2-Removing offset/trend (by subroutine: detrender)

3-Applying a taper window (by subroutine: windower)

4-Zero padding of signal (by subroutine: padder)

5-Transformation to frequency domain (by subroutine: fork)

6-Band-pass filtering (by subroutine: filter)

7-Applying instrument correction (for displacement, velocity, and acceleration) based on calculated response function (by subroutine: respremover)

8-Reverse transformation of the signals to time domain (by subroutine: tracemaker)

9-Calculating PSAs of corrected acceleration time series (by subroutine: make_PSA)

10-Smoothing calculated PSAs (by subroutine: PSA_smoother)

The subroutines for reading SAC-ASCII data file, glitch removal, tapering, Fourier transforms, filtering, and PSA calculations include subroutines by G. Atkinson (AGRAM) and D. Boore (SMSIM), modified to fit this program.

Information for running ICORRECT:

The ICORRECT program calls two other executables from the "SEISAN" package, a number of system commands of the operating system, a parameter file, and the SEED data file downloaded from GSC. To be able to use this program the two open-source executables: "RDSEED.EXE" and "DIRF.EXE" should be in the path. These two files can be copied in a directory and addressed in "autoexec.bat" or the path can be set in environmental variables in Windows XP. If one has already installed the free program "SEISAN" on one's system the abovementioned two files are already on the path. To make sure that these two programs are on the path open a console window and type rdseed to see if it is recognized (also type dirf *.*). "RDSEED.EXE" in the SEISAN package is an executable file for Windows (and DOS) that has the capability to extract data from SEED files and write them in SAC-ASCII format. "DIRF.EXE" in the SEISAN package is a program that performs the same job as the "dir" command in DOS, and outputs a list of the file names as another file.

The parameter file provides the controlling values for running various processing stages in the program and it should be set by the user prior to running the program. A sample of this file, named parameter.dat, is provided in the program compressed file. The name of the parameter file MUST BE "parameter.dat" and should be in the same directory as the input SEED data file. A sample of the "parameter.dat" file is shown in Figure 3. The details of "parameter.dat" are as follows:

- Line1 contains four flags: iglit, itrend, ifilt, and iresp. These flags control four processes: deglitching, de-trending, filtering, and removing instrument response respectively. If the value of any of these flags is set equal to zero the corresponding stage will be skipped.
- Line2 contains the Cosine tapering window width as a fraction relative to trace length. This number should be less than or equal to 0.5. A small value (say 0.05 to 0.1) is recommended to avoid removing signal energy.
- Line 3 contains filter parameters. The program applies Butterworth filters to allow a pass band in the signal. The three parameters in line 3 (norder, flcut, and fhcut) correspond to the order, low frequency limit and high frequency limits of the Butterworth filter applied to the signal, respectively. A default recommendation for broadband data is 4, 0.1, 50.
- Line 4 contains parameters for the calculation of pseudo spectral acceleration (PSA). The first three parameters (freq1, freq2, nfreq) are lowest frequency, highest frequency, and number of frequency points for the calculation of PSA from the acceleration time series. The nfreq points are distributed logarithmically-evenly in frequency space. The fourth parameter (damp) is the ratio of the critical damping of the single degree of freedom system for which the response is calculated. Default recommendation is 0.1,50, 200, 5.
- Line 5 contains PSA smoothing parameters. The first parameter (ismooth) is a flag; if set equal to zero no smoothing will apply. If "ismooth" is set to 1, a running flat box with half width "fbox" (in Hertz) will smooth the PSA values for calculation of a running average. An improvement to the program could include

defining other options for ismooth, so that other types of running windows could apply for smoothing.

The ICORRECT program works on SEED data files provided by GSC. In the convention implemented by GSC, the sensor generator constant does not appear in the SEED response file; instead its value is considered in the stage1 sensitivity constant. ICORRECT read the values of zeros, poles, the normalization factor, and normalization frequency of the sensor in stage 1 as well as the total sensitivity in stage0 (last stage), and constructs the instrument response function based on these parameters. A sample of the SEED response file is presented in Appendix A and the parameters used by the program are highlighted.

Outputs after running ICORRECT:

ICORRECT does correction and signal processing separately for each channel of data and writes all the information of a channel in its directory. Figure 2 demonstrates the types of files that are saved in a channel's directory after ICORRECT processes its data. After running "RDSEED" and extracting time history and instrument response information of a channel from the SEED file, they are stored in the corresponding directory. In the example of Figure 2, "2008.140.00.00.0000.CN.A16.HHE.SAC_ASC" is the SAC-ASCII raw time history data file and "RESP.CN.A16.HHE" is the instrument response information file in SEED format.

To be able to process the data using other software, the response file is reformatted in GSE2, SAC, and PITSA formats, for the convenience of users. In Figure 2 they are "A16__HH_E.2008-05-19-0000_GSE", "SAC.PAZ", and "PITSA.PAZ" for abovementioned formats. Note that the GSE2 response file naming is following the convention expected by the "SEISAN" program to be usable for it. Also there is an intentional format flaw in the CAL2 line of GSE2 response file to make it understandable for "SEISAN". To have the correct GSE2 format, one needs to go to the "make_GSE_resp" subroutine and switch the commented format line with the non-commented one and recompile the program. This way of naming GSE2 calibration files makes it possible for "SEISAN" to keep track of changes of instrument by time and apply the correct one to a signal if it is implemented in signal processing. There is also an

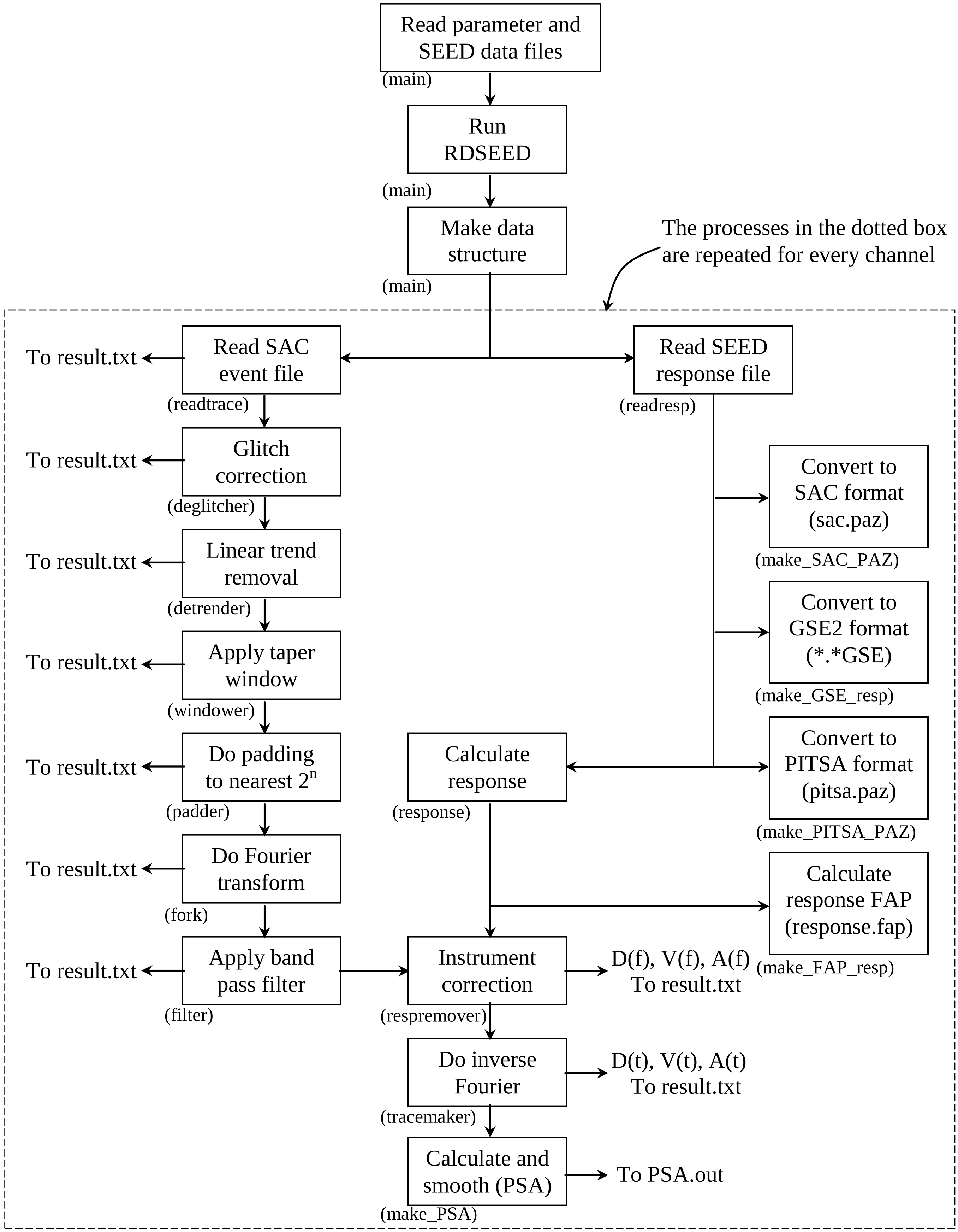

Figure 1. Flow chart of program ICORRECT with subroutine names in parentheses

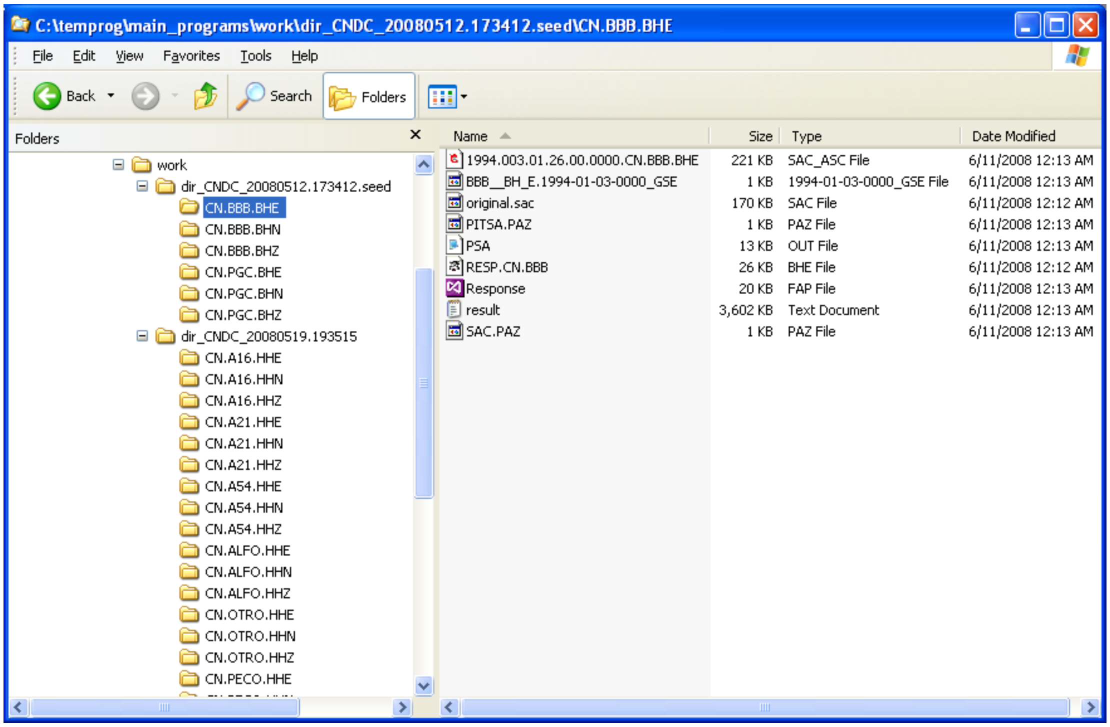

Figure 2. Data structure developed by ICORRECT for a SEED data file (CNDC_20080512.173412.seed) downloaded from GSC AutoDRM. The Left panel shows the data structure and the right panel shows content of one channel directory after running ICORRECT. The file "1994.003.01.26.0000.CN.BBB.BHE.SAC_ASC" is the SAC-ASCII data file ("original.sac" is the original SAC data file generated by RDSEED which in some cases needs a correction before use). The files: "RESP.CN.BBB.BHE", "BBB__BH_E.1994-01-03-0000_GSE", "SAC.PAZ", and "PITSA.PAZ" are instrument response files in SEED, GSE2, SAC, and PITSA formats. "Response.FAP" contains frequency/amplitude/phase curves for instrument response. "result.txt" contains parameter values, time series information, instrument response information, and process results after each stage. "result.txt" carries the most comprehensive information about processes. "PSA.OUT" contains PSA and its smoothed values as a function of both frequency and period.

```
1, 1, 1, 1                         iglit, itrend, ifilt, iresp
0.02                               taper fraction (should be less than 0.5)
1, 0.1, 50.                        norder, flcut, fhcut
0.1, 50., 200, 5.                  freq1, freq2, nfreq, damp
1, 1.5                             ismooth, fbox
```

Figure 3. Sample of a "parameter.dat" file. On the right side the name of parameters are given. First line carries four flags for handling deglitching, de-trending, filtering, and instrument response correction. The second line carries the fraction of tapering width relative to the whole length of window (Cosine window is applied). The third line carries the order, low frequency, and high frequency of Butterworth band-pass filter. The forth line caries frequency range, number of frequency points and percent of critical damping for calculation of PSA. The fifth line carries a flag for PSA curve smoothing and frequency box half-width for smoothing.

important point if one wants to use SAC. Since the time history data file and SAC response file are created in Windows (or DOS) environment, the “end of line characters” should be changed to the Unix or Linux environment equivalents. This is easily done by the “dos2unix” application in the Unix or Linux environments. The program also converts the information in SEED response file to frequency/amplitude/phase displacement, velocity, and acceleration response values and saves them in “Response.FAP” file (see Figure 4 for a portion of a sample file).

The file “result.txt” in Figure 2 contains the most comprehensive information. The header of this file contains parameter values, time series information, instrument response information, and details about processing stages. The body of the file contains processing outputs of each stage up to (and including) corrected acceleration time histories separated in columns (see Figure 5 for a portion of a sample file). The results of all stages are given and separated in columns to make outputs up to a certain processing stage usable. The results in this file are presented in the units of counts, and after instrument correction in meters except for the last column which is the acceleration time series in units of cm/s^2. “PSA.OUT” contains PSA and its smoothed values as a function of both frequency and period. Smoothing is performed by means of a running box with “fbox” halfwidth frequency (see Figure 6 for a portion of a sample file). In this file all PSAs are presented in cm/s^2.

Comparing ICORRECT against other programs:

Instrument response curve, corrected traces, and response spectra calculated by the ICORRECT program were compared against results from other widely used programs as a performance test.

The instrument response curve of a system calculated by ICORRECT is compared against similar calculations for the same system by “PITSA” (Programmable Interactive Toolbox for Seismological Analysis), “JplotResp” (a program for instrument response curve derivation from SEED response files provided by IRIS), and “Resp” (instrument response curve derivation program from instrument parameters, provided in SEISAN package). Figure 7 shows sample calculations performed for station “BBB” using the abovementioned programs. The recorded digital signal was sampled with the rate of

40Hz, so the instrument correction is meaningful (theoretically) up to 20Hz. It is clear from Figure 7 that all results from the four programs show a perfect match up to this frequency. JPlotResp uses all the information in the SEED response file, including all FIR filters, for constructing the instrument response function and consequently generates a different curve for the high frequency response (above 20 Hz). To plot a response file in PITSA one has to generate a series of zeros with one spike in a point and filter it with instrument response file. The sampling frequency of the synthetic time series is assumed to be 100Hz, and 50Hz is the maximum frequency recoverable from such signal. So the response function curve of PITSA shows a break at 50Hz in Figure 7.

Freq.(Hz)	Disp.Ampl.	Disp.Phase	Vel.Ampl.	Vel.Phase	Acc.Ampl.	Acc.Phase
0.10000E-01	0.10119E+08	-170.20	0.16105E+09	99.805	0.25632E+10	9.8048
0.10471E-01	0.11150E+08	-171.98	0.16947E+09	98.022	0.25757E+10	8.0221
0.10965E-01	0.12273E+08	-173.76	0.17815E+09	96.240	0.25858E+10	6.2405
0.11482E-01	0.13497E+08	-175.54	0.18709E+09	94.460	0.25934E+10	4.4598
0.12023E-01	0.14828E+08	-177.32	0.19629E+09	92.680	0.25984E+10	2.6800
0.12589E-01	0.16274E+08	-179.10	0.20574E+09	90.901	0.26010E+10	0.90059
0.13183E-01	0.17844E+08	179.12	0.21544E+09	89.121	0.26010E+10	-0.87867
0.13804E-01	0.19547E+08	177.34	0.22538E+09	87.342	0.25986E+10	-2.6581
0.14454E-01	0.21392E+08	175.56	0.23555E+09	85.562	0.25936E+10	-4.4382
0.15136E-01	0.23389E+08	173.78	0.24594E+09	83.781	0.25861E+10	-6.2190
0.15849E-01	0.25546E+08	172.00	0.25654E+09	81.999	0.25761E+10	-8.0009
0.16596E-01	0.27875E+08	170.22	0.26733E+09	80.216	0.25637E+10	-9.7841
0.17378E-01	0.30387E+08	168.43	0.27829E+09	78.431	0.25487E+10	-11.569
0.18197E-01	0.33090E+08	166.65	0.28941E+09	76.646	0.25313E+10	-13.354
0.19055E-01	0.35997E+08	164.86	0.30067E+09	74.859	0.25114E+10	-15.141
0.19953E-01	0.39119E+08	163.07	0.31204E+09	73.072	0.24890E+10	-16.928
0.20893E-01	0.42466E+08	161.28	0.32349E+09	71.285	0.24643E+10	-18.715
0.21878E-01	0.46050E+08	159.50	0.33501E+09	69.498	0.24371E+10	-20.502
0.22909E-01	0.49883E+08	157.71	0.34655E+09	67.714	0.24076E+10	-22.286
0.23988E-01	0.53973E+08	155.93	0.35810E+09	65.931	0.23759E+10	-24.069
0.25119E-01	0.58334E+08	154.15	0.36961E+09	64.153	0.23419E+10	-25.847
0.26303E-01	0.62976E+08	152.38	0.38106E+09	62.380	0.23058E+10	-27.620
0.27542E-01	0.67909E+08	150.61	0.39242E+09	60.615	0.22676E+10	-29.385
0.28840E-01	0.73144E+08	147.11	0.41471E+09	57.111	0.21856E+10	-32.889
0.31623E-01	0.84562E+08	145.38	0.42559E+09	55.378	0.21420E+10	-34.622
0.33113E-01	0.90764E+08	143.66	0.43625E+09	53.658	0.20968E+10	-36.342
0.34674E-01	0.97309E+08	141.96	0.44666E+09	51.956	0.20502E+10	-38.044
0.36308E-01	0.10421E+09	140.27	0.45679E+09	50.273	0.20024E+10	-39.727
0.38019E-01	0.11147E+09	138.61	0.46663E+09	48.610	0.19534E+10	-41.390
0.39811E-01	0.11910E+09	136.97	0.47615E+09	46.970	0.19036E+10	-43.030
0.41687E-01	0.12712E+09	135.36	0.48534E+09	45.356	0.18529E+10	-44.644
0.43652E-01	0.13554E+09	133.77	0.49417E+09	43.768	0.18018E+10	-46.232
0.45709E-01	0.14436E+09	132.21	0.50264E+09	42.210	0.17502E+10	-47.790
0.47863E-01	0.15360E+09	130.68	0.51074E+09	40.681	0.16983E+10	-49.319
0.50119E-01	0.16327E+09	129.18	0.51847E+09	39.185	0.16464E+10	-50.815
0.52481E-01	0.17339E+09	127.72	0.52582E+09	37.722	0.15946E+10	-52.278
0.54954E-01	0.18396E+09	126.29	0.53278E+09	36.293	0.15430E+10	-53.707
0.57544E-01	0.19502E+09	124.90	0.53938E+09	34.900	0.14918E+10	-55.100
0.60256E-01	0.20656E+09	123.54	0.54560E+09	33.544	0.14411E+10	-56.456
0.63096E-01	0.21862E+09	122.22	0.55146E+09	32.224	0.13910E+10	-57.776
0.66069E-01	0.23121E+09	120.94	0.55697E+09	30.942	0.13417E+10	-59.058
0.69183E-01	0.24435E+09	119.70	0.56213E+09	29.698	0.12932E+10	-60.302
0.72444E-01	0.25807E+09	118.49	0.56696E+09	28.492	0.12456E+10	-61.508
0.75858E-01	0.27238E+09	117.32	0.57147E+09	27.324	0.11990E+10	-62.676

Figure 4. Sample of an instrument response function ("Response.FAP") file. The first column shows the values of frequencies, and the successive pairs of columns correspond to displacement, velocity, and acceleration amplitudes and phases.

```
RECORD INFORMATION:
Beginning of record:  1994/01/03 01:26:00.000
Sampling rate:          40.0000
# of samples:          10900
# after padding:       16384
STATION INFORMATION:
Station name:         BBB
Component:            BHE
Station latitude:      52.1847
Station longitude:    -128.1133
Station height:          14.00
GLITCHES:
Glitches removed:     Yes
TREND AND OFFSET:
Trend/offset removed: Yes
A and B values in equation: y=Ax+B
A and B values:         -2.9674         36578.
TAPERING:
Tapering window type: Cosine
Tapering window ratio: .02000
FILTER:
Filter applied:       Yes
Filter type:          Butterworth
Filter order:          1
Frequency range:        0.100 to  50.000 Hertz
INSTRUMENT RESPONSE INFORMATION:
Instrument correction: Applied
Normalization factor:    98752.
Normalization freq.:     1.0000
Number of zeros:       2 in:
     0.0000         0.0000
     0.0000         0.0000
Number of poles:       4 in:
   -0.31400E-01     0.0000
   -0.20900         0.0000
    -222.11        -222.18
    -222.11         222.18
Overall sensitivity:    0.62500E+09
Normalization factor: Matches
Calibration files in SAC, SEED, GSE2, and PITSA formats created in channel folder
RESPONSE SPECTRUM INFORMATION:
PSA information file: PSA.out
Response spectrum:    Smoothed
Smoothing box width:   1.50000Hz in each side
END_HEADER
   Sample #   |   Original   |  Deglitched  |  Detrended    |   Windowed   |<--   Fourier transform   -->| Fourier(Amp) |
                  (Count)        (Count)        (Count)        (Count)        Re(Count*s)   Im(Count*s)     (Count*s)
           1   -78080.0       -78080.0       -114655.      -0.547676E-10    380786.        0.00000        380786.
           2   -80128.0       -80128.0       -116700.       -6.11479       -410147.        22072.2        410741.
           3   -81408.0       -81408.0       -117977.       -24.7256        284482.       -120438.        308926.
           4   -84480.0       -84480.0       -121046.       -57.0749       -324449.        372726.        494158.
           5   -87040.0       -87040.0       -123603.       -103.597        357831.       -12731.2        358058.
           6   -90112.0       -90112.0       -126672.       -165.864        500153.       -454718.        675960.
           7   -93184.0       -93184.0       -129741.       -244.584       -982958.       -499318.       0.110251E+07
           8   -96256.0       -96256.0       -132810.       -340.703       -978098.        774548.       0.124764E+07
           9   -99840.0       -99840.0       -136391.       -456.879       0.125407E+07   0.182151E+07   0.221147E+07
          10   -102144.       -102144.       -138692.       -587.818       0.217682E+07  -0.107459E+07   0.242760E+07

| Fourier(Amp) |   Filtered    | Instr. corr. Disp., Vel., Acc. Amp. Spect. | Instr. corr. Disp., Vel., Acc. time history|
   (Count*s)      (Count*s)      (Meter*s)       (Meter)       (Meter/s)        (Meter)       (Meter/s)       (M/s^2)
  380786.        0.00000        0.00000        0.00000        0.00000       0.832143E-05   0.407617E-05   0.296487E-05
  410741.        244.675       0.793822E-03   0.121771E-04   0.186794E-06   0.842405E-05   0.412134E-05   0.109186E-05
  308926.        734.786       0.377333E-03   0.115764E-04   0.355160E-06   0.852766E-05   0.415579E-05   0.126491E-05
  494158.        2636.73       0.515806E-03   0.237371E-04   0.109237E-05   0.863115E-05   0.416045E-05  -0.642147E-06
  358058.        3382.45       0.351523E-03   0.215692E-04   0.132347E-05   0.873464E-05   0.415230E-05  -0.462285E-06
  675960.        9924.70       0.648996E-03   0.497773E-04   0.381787E-05   0.883848E-05   0.410737E-05  -0.261725E-05
 0.110251E+07    23160.3       0.105492E-02   0.970938E-04   0.893641E-05   0.894075E-05   0.405009E-05  -0.253459E-05
 0.124764E+07    35404.8       0.120208E-02   0.129077E-03   0.138601E-04   0.904046E-05   0.395207E-05  -0.497792E-05
 0.221147E+07    81260.9       0.215862E-02   0.264903E-03   0.325085E-04   0.913767E-05   0.383146E-05  -0.490109E-05
 0.242760E+07    111806.       0.240959E-02   0.332664E-03   0.459270E-04   0.923162E-05   0.367907E-05  -0.681728E-05
```

Figure 5. Sample of a “result.txt” file. The header of this file contains parameter values, time series information, and instrument response information. The body of the file contains processing outputs of each stage up to (and including) corrected acceleration time histories separated in columns.

```
 Calculated PSA for the frequency range:
  0.1000000      Hz and    50.00000      Hz
         200 frequencies distributed logarithmically
 Oscillator damping is    5.000000     % of critical
  Freq.(Hz)        Period(s)       PSA(cm/s^2)  Smooth PSA(cm/s^2)
   50.00000      1.9999998E-02   65.90721       65.90721
   48.46267      2.0634437E-02   66.00179       66.15620
   46.97261      2.1289000E-02   66.31063       66.32871
   45.52837      2.1964328E-02   66.67371       66.60388
   44.12852      2.2661081E-02   66.82729       66.64972
   42.77172      2.3379933E-02   66.44816       66.35059
   41.45663      2.4121592E-02   65.77632       65.93240
   40.18199      2.4886772E-02   65.57272       65.59551
   38.94653      2.5676232E-02   65.43749       65.25227
   37.74905      2.6490729E-02   64.74658       64.72079
```

Figure 6. Sample of a “PSA.out” file. The header of this file contains the frequency range of PSA calculations and number of frequency points. The body of the file contains Frequency (and period), calculated PSA, and smoothed PSA curve values.

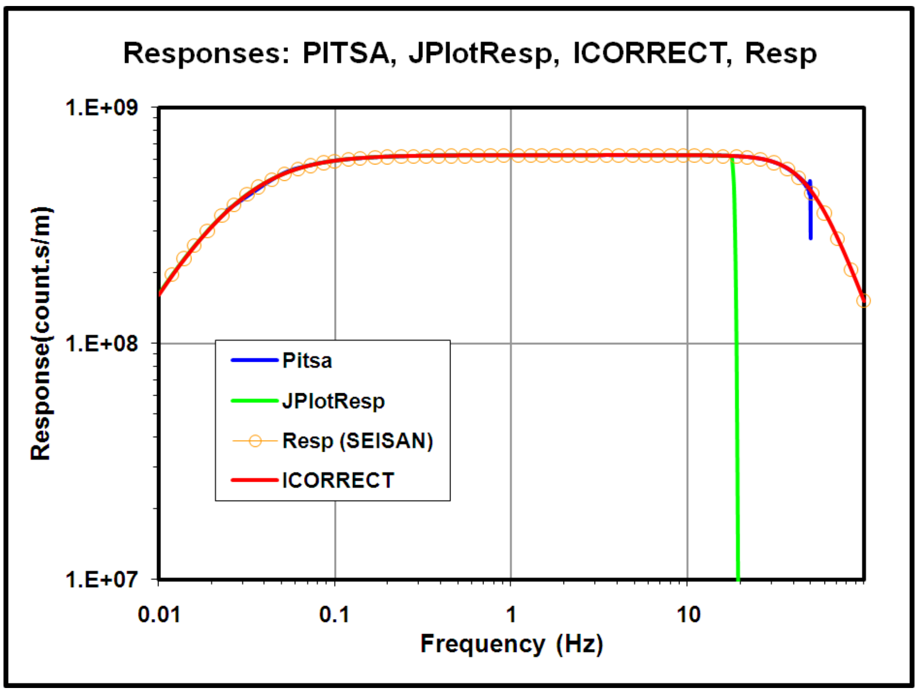

Figure 7. Instrument response curve of “BBB” station derived by four programs. This comparison demonstrates performance of the instrument response calculation of ICORRECT.

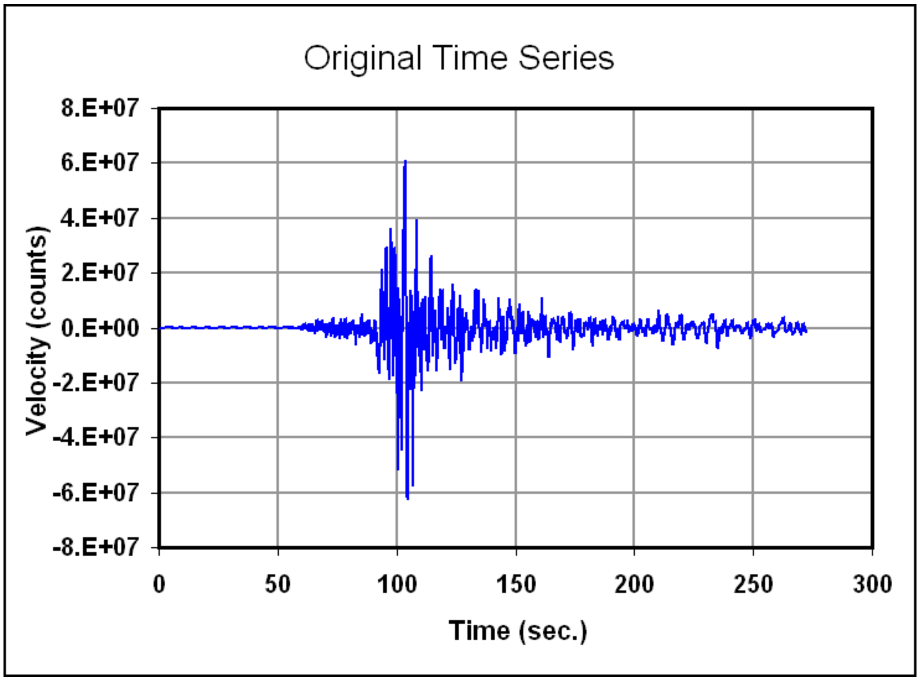

Figure 8. Digitized time series of an M5.6 earthquake recorded by station "BBB" at a distance of 293 km.

Now lets look at the time series obtained after instrument correction. Figure 8 shows the digitized signal of an **M**5.6 earthquake recorded by station "BBB" at a distance of 293 km before any processing. The unit of measurement in the raw signal is counts.

Instrument correction is applied to the signal using three programs: ICORRECT, "SAC", and "SEISAN". Figure 9 is an overlay of corrected signals by these three programs. The 3 programs didn't adopt exactly the same processing stages/parameters before instrument correction, but the three signals show an excellent match after instrument correction. This was also verified by examining "zooms" of the overlay in detail. This test further demonstrates that ICORRECT program is doing a reasonable job in seismic signal processing and it is reliable for processing SEED data files obtained from GSC.

As the last test, the PSAs calculated by ICORRECT are compared with those calculated by SPECEQ (a program for calculation of response spectra available to download from NISEE, Berkeley). The two PSAs are calculated for the corrected signal of Figure 9. Both programs follow the Nigam and Jennings (1969) algorithm but in two separate programs. We use the subroutine written by D. Boore (for SMSIM). Figure 10 shows an overlay of PSAs derived from two programs; there is a near- perfect match.

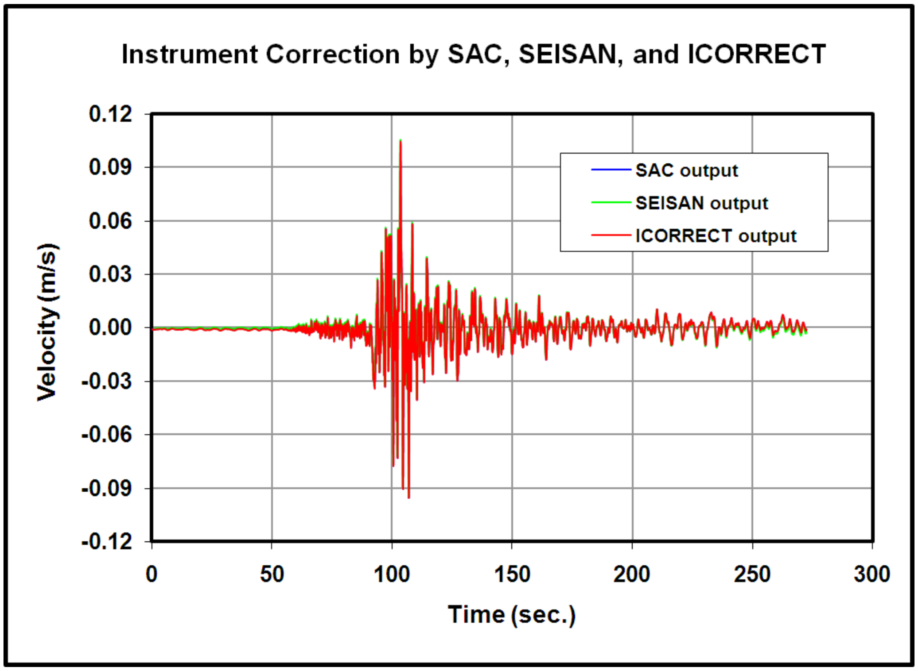

Figure 9. Corrected signal (Figure 8) after instrument response removal by three programs. The close match between the three traces demonstrates the reasonable performance of the program ICORRECT.

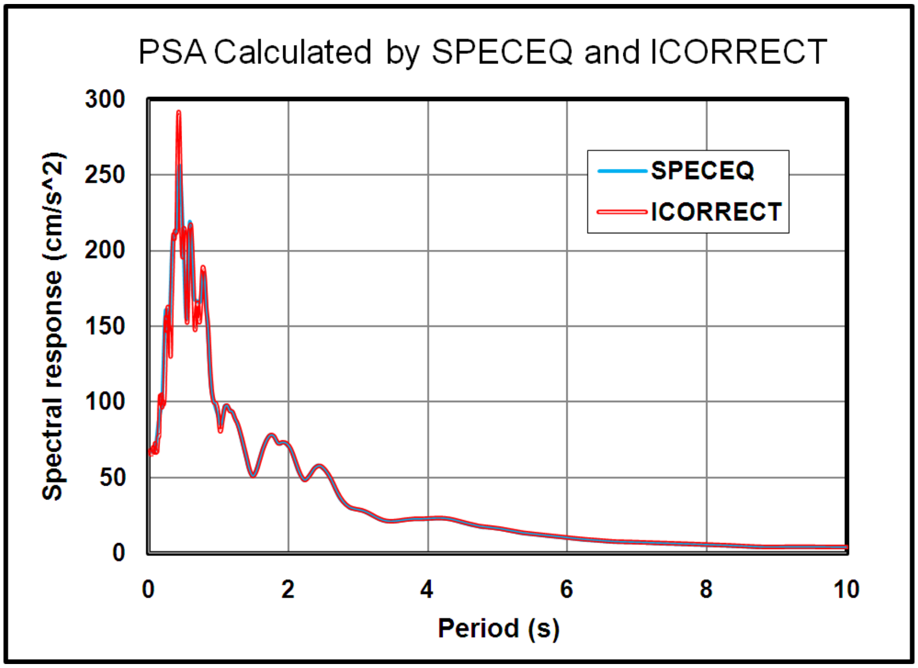

Figure 10. PSA of corrected signal in Figure 9 (after converting to acceleration) calculated by ICORRECT and "SPECEQ" programs. The two spectra derived from two different programs show near-perfect match and confirm performance of ICORRECT program.

Acknowledements:

Some of the subroutines implemented in this program are modified or exact version of those developed by D. Boore for the software package SMSIM. We benefitted from discussions with and comments by H. Ghofrani during the proof testing of the code and application. N. Kraeva provided insight into instrument correction using SAC. A. Darlington found a problem in the SAC-ASCII files.

Appendices:

Appendix A: Instrument response file of station “BBB” in SEED format

(The highlighted parameters are used in response calculation)

```
#           << IRIS SEED Reader, Release 4.15 >>
#
#           ======== CHANNEL RESPONSE DATA ========
B050F03     Station:     BBB
B050F16     Network:     CN
B052F04     Channel:     BHE
B052F22     Start date:  1993,302,14:00:00
B052F23     End date:    No Ending Time
#           =======================================
#           +            +---------------------------------------------------+              +
#           +            |   Response Reference Information,   BBB ch BHE    |              +
#           +            +---------------------------------------------------+              +
#
B060F03     Number of Stages:                       7
B060F04     Stage number:                           1
B060F05     Number of Responses:                    2
#
#           +               +---------------------------------------------+                 +
#           +               |   Response (Poles & Zeros),   BBB ch BHE    |                 +
#           +               +---------------------------------------------+                 +
#
B043F05     Response type:                          A [Laplace Transform (Rad/sec)]
B043F06     Response in units lookup:               M/S - (null)
B043F07     Response out units lookup:              V - (null)
B043F08     A0 normalization factor:                98752.1
B043F09     Normalization frequency:                1
B043F10     Number of zeroes:                       2
B043F15     Number of poles:                        4
#           Complex zeroes:
#             i  real           imag           real_error    imag_error
B043F11-14    0  0.000000E+000  0.000000E+000  0.000000E+000  0.000000E+000
B043F11-14    1  0.000000E+000  0.000000E+000  0.000000E+000  0.000000E+000
#           Complex poles:
#             i  real           imag           real_error    imag_error
B043F16-19    0 -3.140000E-002  0.000000E+000  0.000000E+000  0.000000E+000
B043F16-19    1 -2.090000E-001  0.000000E+000  0.000000E+000  0.000000E+000
B043F16-19    2 -2.221110E+002 -2.221780E+002  0.000000E+000  0.000000E+000
B043F16-19    3 -2.221110E+002  2.221780E+002  0.000000E+000  0.000000E+000
#
#           +                  +---------------------------------------+                  +
#           +                  |   Channel Sensitivity,   BBB ch BHE   |                  +
#           +                  +---------------------------------------+                  +
#
B048F05     Sensitivity:                            1.504000E+003
B048F06     Frequency of sensitivity:               1.000000E+000
B048F07     Number of calibrations:                 0
#
#           +            +---------------------------------------------------+              +
#           +            |   Response Reference Information,   BBB ch BHE    |              +
#           +            +---------------------------------------------------+              +
#
B060F03     Number of Stages:                       7
B060F04     Stage number:                           2
B060F05     Number of Responses:                    2
#
#           +               +---------------------------------------------+                 +
#           +               |   Response (Poles & Zeros),   BBB ch BHE    |                 +
#           +               +---------------------------------------------+                 +
#
B043F05     Response type:                          A [Laplace Transform (Rad/sec)]
B043F06     Response in units lookup:               V - (null)
B043F07     Response out units lookup:              COUNTS - (null)
B043F08     A0 normalization factor:                1
B043F09     Normalization frequency:                1
B043F10     Number of zeroes:                       0
B043F15     Number of poles:                        0
#           Complex zeroes:
#             i  real           imag           real_error    imag_error
#           Complex poles:
#             i  real           imag           real_error    imag_error
```

```
#
#          +                  +---------------------------------------+                  +
#          +                  |   Channel Sensitivity,   BBB ch BHE   |                  +
#          +                  +---------------------------------------+                  +
#
B048F05     Sensitivity:                         1.063830E+008
B048F06     Frequency of sensitivity:            1.000000E+000
B048F07     Number of calibrations:              0
#
#          +            +---------------------------------------------------+            +
#          +            |   Response Reference Information,   BBB ch BHE   |            +
#          +            +---------------------------------------------------+            +
#
B060F03     Number of Stages:                    7
B060F04     Stage number:                        3
B060F05     Number of Responses:                 2
#
#          +               +---------------------------------------------+               +
#          +               |   Response (Poles & Zeros),   BBB ch BHE   |               +
#          +               +---------------------------------------------+               +
#
B043F05     Response type:                       A [Laplace Transform (Rad/sec)]
B043F06     Response in units lookup:            COUNTS - (null)
B043F07     Response out units lookup:           COUNTS - (null)
B043F08     A0 normalization factor:             1
B043F09     Normalization frequency:             1
B043F10     Number of zeroes:                    0
B043F15     Number of poles:                     0
#           Complex zeroes:
#             i  real          imag          real_error    imag_error
#           Complex poles:
#             i  real          imag          real_error    imag_error
#
#          +                  +---------------------------------------+                  +
#          +                  |   Channel Sensitivity,   BBB ch BHE   |                  +
#          +                  +---------------------------------------+                  +
#
B048F05     Sensitivity:                         3.906250E-003
B048F06     Frequency of sensitivity:            1.000000E+000
B048F07     Number of calibrations:              0
#
#          +            +---------------------------------------------------+            +
#          +            |   Response Reference Information,   BBB ch BHE   |            +
#          +            +---------------------------------------------------+            +
#
B060F03     Number of Stages:                    7
B060F04     Stage number:                        4
B060F05     Number of Responses:                 3
#
#          +                     +---------------------------------+                     +
#          +                     |   FIR response,   BBB ch BHE   |                     +
#          +                     +---------------------------------+                     +
#
B041F05     Symmetry type:                       B
B041F06     Response in units lookup:            COUNTS - (null)
B041F07     Response out units lookup:           COUNTS - (null)
B041F08     Number of numerators:                149
#           Numerator coefficients:
#             i, coefficient
B041F09       0 -3.059151E-008
B041F09       1 -1.754854E-008
B041F09       2 -1.528537E-008
B041F09       3 -8.036346E-009
B041F09       4  6.287972E-009
B041F09       5  3.016296E-008
B041F09       6  6.643942E-008
B041F09       7  1.183109E-007
B041F09       8  1.892547E-007
B041F09       9  2.829424E-007
B041F09      10  4.031165E-007
B041F09      11  5.534309E-007
B041F09      12  7.372518E-007
B041F09      13  9.574200E-007
B041F09      14  1.215973E-006
B041F09      15  1.513832E-006
B041F09      16  1.850453E-006
B041F09      17  2.223456E-006
B041F09      18  2.628228E-006
B041F09      19  3.057527E-006
B041F09      20  3.501086E-006
B041F09      21  3.945234E-006
```

B041F09	22	4.372561E-006
B041F09	23	4.761638E-006
B041F09	24	5.086809E-006
B041F09	25	5.318094E-006
B041F09	26	5.421203E-006
B041F09	27	5.357704E-006
B041F09	28	5.085359E-006
B041F09	29	4.558641E-006
B041F09	30	3.729468E-006
B041F09	31	2.548155E-006
B041F09	32	9.645956E-007
B041F09	33	-1.070308E-006
B041F09	34	-3.602989E-006
B041F09	35	-6.675321E-006
B041F09	36	-1.032252E-005
B041F09	37	-1.457083E-005
B041F09	38	-1.943511E-005
B041F09	39	-2.491624E-005
B041F09	40	-3.099852E-005
B041F09	41	-3.764707E-005
B041F09	42	-4.480524E-005
B041F09	43	-5.239233E-005
B041F09	44	-6.030137E-005
B041F09	45	-6.839735E-005
B041F09	46	-7.651592E-005
B041F09	47	-8.446250E-005
B041F09	48	-9.201217E-005
B041F09	49	-9.891017E-005
B041F09	50	-1.048733E-004
B041F09	51	-1.095923E-004
B041F09	52	-1.127350E-004
B041F09	53	-1.139507E-004
B041F09	54	-1.128754E-004
B041F09	55	-1.091385E-004
B041F09	56	-1.023701E-004
B041F09	57	-9.220990E-005
B041F09	58	-7.831622E-005
B041F09	59	-6.037684E-005
B041F09	60	-3.811999E-005
B041F09	61	-1.132605E-005
B041F09	62	2.016026E-005
B041F09	63	5.641772E-005
B041F09	64	9.743634E-005
B041F09	65	1.431057E-004
B041F09	66	1.932037E-004
B041F09	67	2.473866E-004
B041F09	68	3.051802E-004
B041F09	69	3.659731E-004
B041F09	70	4.290111E-004
B041F09	71	4.933949E-004
B041F09	72	5.580803E-004
B041F09	73	6.218807E-004
B041F09	74	6.834736E-004
B041F09	75	7.414104E-004
B041F09	76	7.941291E-004
B041F09	77	8.399717E-004
B041F09	78	8.772049E-004
B041F09	79	9.040441E-004
B041F09	80	9.186818E-004
B041F09	81	9.193185E-004
B041F09	82	9.041980E-004
B041F09	83	8.716437E-004
B041F09	84	8.200988E-004
B041F09	85	7.481676E-004
B041F09	86	6.546576E-004
B041F09	87	5.386230E-004
B041F09	88	3.994069E-004
B041F09	89	2.366834E-004
B041F09	90	5.049688E-005
B041F09	91	-1.587007E-004
B041F09	92	-3.900165E-004
B041F09	93	-6.420886E-004
B041F09	94	-9.130618E-004
B041F09	95	-1.200571E-003
B041F09	96	-1.501731E-003
B041F09	97	-1.813134E-003
B041F09	98	-2.130855E-003
B041F09	99	-2.450466E-003
B041F09	100	-2.767059E-003
B041F09	101	-3.075282E-003
B041F09	102	-3.369376E-003

```
B041F09     103 -3.643231E-003
B041F09     104 -3.890442E-003
B041F09     105 -4.104382E-003
B041F09     106 -4.278275E-003
B041F09     107 -4.405284E-003
B041F09     108 -4.478596E-003
B041F09     109 -4.491521E-003
B041F09     110 -4.437591E-003
B041F09     111 -4.310660E-003
B041F09     112 -4.105004E-003
B041F09     113 -3.815429E-003
B041F09     114 -3.437368E-003
B041F09     115 -2.966972E-003
B041F09     116 -2.401209E-003
B041F09     117 -1.737940E-003
B041F09     118 -9.759976E-004
B041F09     119 -1.152480E-004
B041F09     120  8.433559E-004
B041F09     121  1.897735E-003
B041F09     122  3.044658E-003
B041F09     123  4.279733E-003
B041F09     124  5.597408E-003
B041F09     125  6.990998E-003
B041F09     126  8.452717E-003
B041F09     127  9.973734E-003
B041F09     128  1.154424E-002
B041F09     129  1.315354E-002
B041F09     130  1.479015E-002
B041F09     131  1.644189E-002
B041F09     132  1.809604E-002
B041F09     133  1.973945E-002
B041F09     134  2.135871E-002
B041F09     135  2.294028E-002
B041F09     136  2.447064E-002
B041F09     137  2.593646E-002
B041F09     138  2.732478E-002
B041F09     139  2.862312E-002
B041F09     140  2.981968E-002
B041F09     141  3.090343E-002
B041F09     142  3.186431E-002
B041F09     143  3.269332E-002
B041F09     144  3.338264E-002
B041F09     145  3.392572E-002
B041F09     146  3.431739E-002
B041F09     147  3.455391E-002
B041F09     148  3.463300E-002
#
#           +                     +------------------------------+                       +
#           +                     |   Decimation,   BBB ch BHE   |                       +
#           +                     +------------------------------+                       +
#
B047F05     Response input sample rate:           1.500000E+004
B047F06     Response decimation factor:           25
B047F07     Response decimation offset:           0
B047F08     Response delay:                       0.000000E+000
B047F09     Response correction:                  0.000000E+000
#
#           +                 +---------------------------------------+                  +
#           +                 |   Channel Sensitivity,   BBB ch BHE   |                  +
#           +                 +---------------------------------------+                  +
#
B048F05     Sensitivity:                          1.000000E+000
B048F06     Frequency of sensitivity:             0.000000E+000
B048F07     Number of calibrations:               0
#
#           +           +---------------------------------------------------+            +
#           +           |   Response Reference Information,   BBB ch BHE   |            +
#           +           +---------------------------------------------------+            +
#
B060F03     Number of Stages:                     7
B060F04     Stage number:                         5
B060F05     Number of Responses:                  3
#
#           +                    +--------------------------------+                      +
#           +                    |   FIR response,   BBB ch BHE   |                      +
#           +                    +--------------------------------+                      +
#
B041F05     Symmetry type:                        B
B041F06     Response in units lookup:             COUNTS - (null)
B041F07     Response out units lookup:            COUNTS - (null)
B041F08     Number of numerators:                 37
```

```
#           Numerator coefficients:
#             i, coefficient
B041F09       0 -1.068933E-008
B041F09       1  3.042043E-007
B041F09       2  1.426381E-006
B041F09       3  3.482656E-006
B041F09       4  5.028638E-006
B041F09       5  1.982701E-006
B041F09       6 -1.153722E-005
B041F09       7 -3.865957E-005
B041F09       8 -7.175441E-005
B041F09       9 -8.543766E-005
B041F09      10 -3.904887E-005
B041F09      11  1.033548E-004
B041F09      12  3.359234E-004
B041F09      13  5.729440E-004
B041F09      14  6.407028E-004
B041F09      15  3.310730E-004
B041F09      16 -4.752729E-004
B041F09      17 -1.654912E-003
B041F09      18 -2.744849E-003
B041F09      19 -3.007405E-003
B041F09      20 -1.715471E-003
B041F09      21  1.394095E-003
B041F09      22  5.686433E-003
B041F09      23  9.485824E-003
B041F09      24  1.045387E-002
B041F09      25  6.523235E-003
B041F09      26 -2.871479E-003
B041F09      27 -1.578225E-002
B041F09      28 -2.754034E-002
B041F09      29 -3.172284E-002
B041F09      30 -2.220007E-002
B041F09      31  4.360541E-003
B041F09      32  4.650803E-002
B041F09      33  9.747236E-002
B041F09      34  1.465384E-001
B041F09      35  1.820423E-001
B041F09      36  1.950000E-001
#
#           +                      +------------------------------+                      +
#           +                      |   Decimation,   BBB ch BHE   |                      +
#           +                      +------------------------------+                      +
#
B047F05     Response input sample rate:          6.000000E+002
B047F06     Response decimation factor:          5
B047F07     Response decimation offset:          0
B047F08     Response delay:                      0.000000E+000
B047F09     Response correction:                 0.000000E+000
#
#           +                  +---------------------------------------+                  +
#           +                  |   Channel Sensitivity,   BBB ch BHE   |                  +
#           +                  +---------------------------------------+                  +
#
B048F05     Sensitivity:                         1.000000E+000
B048F06     Frequency of sensitivity:            0.000000E+000
B048F07     Number of calibrations:              0
#
#           +            +---------------------------------------------------+            +
#           +            |   Response Reference Information,   BBB ch BHE   |            +
#           +            +---------------------------------------------------+            +
#
B060F03     Number of Stages:                    7
B060F04     Stage number:                        6
B060F05     Number of Responses:                 3
#
#           +                     +---------------------------------+                     +
#           +                     |   FIR response,   BBB ch BHE   |                     +
#           +                     +---------------------------------+                     +
#
B041F05     Symmetry type:                       B
B041F06     Response in units lookup:            COUNTS - (null)
B041F07     Response out units lookup:           COUNTS - (null)
B041F08     Number of numerators:                196
#           Numerator coefficients:
#             i, coefficient
B041F09       0  3.756967E-008
B041F09       1 -1.457860E-008
B041F09       2 -5.080732E-008
B041F09       3 -4.919095E-008
B041F09       4  1.172643E-008
```

B041F09	5	9.347280E-008
B041F09	6	1.120729E-007
B041F09	7	1.441347E-008
B041F09	8	-1.451895E-007
B041F09	9	-2.177803E-007
B041F09	10	-8.604914E-008
B041F09	11	1.918969E-007
B041F09	12	3.755745E-007
B041F09	13	2.329767E-007
B041F09	14	-2.059310E-007
B041F09	15	-5.861272E-007
B041F09	16	-4.902566E-007
B041F09	17	1.423920E-007
B041F09	18	8.340515E-007
B041F09	19	8.928371E-007
B041F09	20	6.272560E-008
B041F09	21	-1.079149E-006
B041F09	22	-1.466479E-006
B041F09	23	-4.911403E-007
B041F09	24	1.247564E-006
B041F09	25	2.214784E-006
B041F09	26	1.236015E-006
B041F09	27	-1.224667E-006
B041F09	28	-3.102822E-006
B041F09	29	-2.389361E-006
B041F09	30	8.520866E-007
B041F09	31	4.038764E-006
B041F09	32	4.022165E-006
B041F09	33	6.831091E-008
B041F09	34	-4.856010E-006
B041F09	35	-6.157212E-006
B041F09	36	-1.760656E-006
B041F09	37	5.299345E-006
B041F09	38	8.735865E-006
B041F09	39	4.449121E-006
B041F09	40	-5.019578E-006
B041F09	41	-1.158170E-005
B041F09	42	-8.318419E-006
B041F09	43	3.581625E-006
B041F09	44	1.436567E-005
B041F09	45	1.346108E-005
B041F09	46	-4.908430E-007
B041F09	47	-1.657926E-005
B041F09	48	-1.981415E-005
B041F09	49	-4.758459E-006
B041F09	50	1.752331E-005
B041F09	51	2.709054E-005
B041F09	52	1.261038E-005
B041F09	53	-1.632097E-005
B041F09	54	-3.471348E-005
B041F09	55	-2.335159E-005
B041F09	56	1.196278E-005
B041F09	57	4.176480E-005
B041F09	58	3.699691E-005
B041F09	59	-3.390085E-006
B041F09	60	-4.696009E-005
B041F09	61	-5.316562E-005
B041F09	62	-1.038009E-005
B041F09	63	4.866455E-005
B041F09	64	7.096189E-005
B041F09	65	3.009053E-005
B041F09	66	-4.496233E-005
B041F09	67	-8.887732E-005
B041F09	68	-5.604729E-005
B041F09	69	3.378952E-005
B041F09	70	1.047363E-004
B041F09	71	8.790563E-005
B041F09	72	-1.313563E-005
B041F09	73	-1.157058E-004
B041F09	74	-1.244599E-004
B041F09	75	-1.868886E-005
B041F09	76	1.183909E-004
B041F09	77	1.634650E-004
B041F09	78	6.273027E-005
B041F09	79	-1.090305E-004
B041F09	80	-2.015216E-004
B041F09	81	-1.190367E-004
B041F09	82	8.380331E-005
B041F09	83	2.340585E-004
B041F09	84	1.863023E-004
B041F09	85	-3.924064E-005

```
B041F09      86 -2.554442E-004
B041F09      87 -2.615514E-004
B041F09      88 -2.726986E-005
B041F09      89  2.592530E-004
B041F09      90  3.399115E-004
B041F09      91  1.169195E-004
B041F09      92 -2.386999E-004
B041F09      93 -4.145242E-004
B041F09      94 -2.288992E-004
B041F09      95  1.872427E-004
B041F09      96  4.766431E-004
B041F09      97  3.598632E-004
B041F09      98 -9.933377E-005
B041F09      99 -5.159571E-004
B041F09     100 -5.035059E-004
B041F09     101 -2.872143E-005
B041F09     102  5.211647E-004
B041F09     103  6.503229E-004
B041F09     104  1.978585E-004
B041F09     105 -4.808023E-004
B041F09     106 -7.876287E-004
B041F09     107 -4.053822E-004
B041F09     108  3.843069E-004
B041F09     109  8.998887E-004
B041F09     110  6.442223E-004
B041F09     111 -2.232614E-004
B041F09     112 -9.694080E-004
B041F09     113 -9.024186E-004
B041F09     114 -7.252935E-006
B041F09     115  9.773908E-004
B041F09     116  1.162936E-003
B041F09     117  3.073024E-004
B041F09     118 -9.053523E-004
B041F09     119 -1.403893E-003
B041F09     120 -6.708679E-004
B041F09     121  7.368267E-004
B041F09     122  1.599276E-003
B041F09     123  1.084868E-003
B041F09     124 -4.592810E-004
B041F09     125 -1.720160E-003
B041F09     126 -1.528574E-003
B041F09     127  6.610657E-005
B041F09     128  1.736442E-003
B041F09     129  1.973531E-003
B041F09     130  4.414634E-004
B041F09     131 -1.619007E-003
B041F09     132 -2.384092E-003
B041F09     133 -1.052680E-003
B041F09     134  1.342260E-003
B041F09     135  2.718607E-003
B041F09     136  1.746016E-003
B041F09     137 -8.868437E-004
B041F09     138 -2.931276E-003
B041F09     139 -2.488475E-003
B041F09     140  2.423846E-004
B041F09     141  2.974610E-003
B041F09     142  3.235578E-003
B041F09     143  5.899338E-004
B041F09     144 -2.802396E-003
B041F09     145 -3.932119E-003
B041F09     146 -1.595177E-003
B041F09     147  2.372982E-003
B041F09     148  4.513672E-003
B041F09     149  2.743019E-003
B041F09     150 -1.652680E-003
B041F09     151 -4.908776E-003
B041F09     152 -3.986781E-003
B041F09     153  6.190280E-004
B041F09     154  5.041622E-003
B041F09     155  5.263205E-003
B041F09     156  7.363502E-004
B041F09     157 -4.834962E-003
B041F09     158 -6.492899E-003
B041F09     159 -2.405545E-003
B041F09     160  4.212805E-003
B041F09     161  7.581216E-003
B041F09     162  4.363045E-003
B041F09     163 -3.102319E-003
B041F09     164 -8.419085E-003
B041F09     165 -6.565489E-003
B041F09     166  1.434040E-003
```

```
B041F09     167  8.882906E-003
B041F09     168  8.952579E-003
B041F09     169  8.610203E-004
B041F09     170 -8.831848E-003
B041F09     171 -1.144919E-002
B041F09     172 -3.859935E-003
B041F09     173  8.099306E-003
B041F09     174  1.396857E-002
B041F09     175  7.667336E-003
B041F09     176 -6.471534E-003
B041F09     177 -1.641655E-002
B041F09     178 -1.245805E-002
B041F09     179  3.636455E-003
B041F09     180  1.869640E-002
B041F09     181  1.857586E-002
B041F09     182  9.450267E-004
B041F09     183 -2.071411E-002
B041F09     184 -2.679285E-002
B041F09     185 -8.406543E-003
B041F09     186  2.238377E-002
B041F09     187  3.914758E-002
B041F09     188  2.179603E-002
B041F09     189 -2.363252E-002
B041F09     190 -6.278789E-002
B041F09     191 -5.334038E-002
B041F09     192  2.440493E-002
B041F09     193  1.483565E-001
B041F09     194  2.624688E-001
B041F09     195  3.086670E-001
#
#           +                      +------------------------------+                       +
#           +                      |   Decimation,   BBB ch BHE   |                       +
#           +                      +------------------------------+                       +
#
B047F05     Response input sample rate:            1.200000E+002
B047F06     Response decimation factor:            3
B047F07     Response decimation offset:            0
B047F08     Response delay:                        0.000000E+000
B047F09     Response correction:                   0.000000E+000
#
#           +                  +---------------------------------------+                  +
#           +                  |   Channel Sensitivity,   BBB ch BHE   |                  +
#           +                  +---------------------------------------+                  +
#
B048F05     Sensitivity:                           1.000000E+000
B048F06     Frequency of sensitivity:              0.000000E+000
B048F07     Number of calibrations:                0
#
#           +            +-------------------------------------------------+              +
#           +            |   Response Reference Information,   BBB ch BHE   |              +
#           +            +-------------------------------------------------+              +
#
B060F03     Number of Stages:                      7
B060F04     Stage number:                          0
B060F05     Number of Responses:                   1
#
#           +                  +---------------------------------------+                  +
#           +                  |   Channel Sensitivity,   BBB ch BHE   |                  +
#           +                  +---------------------------------------+                  +
#
B048F05     Sensitivity:                           6.250000E+008
B048F06     Frequency of sensitivity:              1.000000E+000
B048F07     Number of calibrations:                0
#
#
```

Appendix B: Source code of ICORRECT program

```
      USE DFLIB
      integer(2) n_of_args
        dimension trace(100000),detrended(100000),windowed(100000)
        dimension deglitched(100000),padded(100000)
        dimension respremoved_tr(100000,3)
        complex fourier(100000),filtered(100000)
        complex p(100),z(100),accresp, velresp, dspresp
        complex respremoved_sp(100000,3)
        character input*150,root*150,current*150,currdir*150
        character reslis*80(1000),tralis*80(1000),resdir*80,tradir*80
        character sta_path*150(1000),sta_nam*20(1000)
        character event*12,stime*12,staname*4,comp*3
        common /par/igmtn,iglit,itrend,ifilt, iresp,taper,norder,
     *  flcut, fhcut, ismooth, fbox, freq1, freq2, nfreq, damp
        common /respinfo/A0,f0,nzero,npole,p,z,sensitivity,A0ok
        common /traceinfo/sr,nsamp,trace,event,stime,staname,comp
        common /stainfo/stalat,stalon,stah
        common /detrend/a,b,detrended

        n_of_args=nargs()
        if(n_of_args.gt.1)then
        call getarg(1,input)
        else
c       Read data file (in seed format):
        write(*,*)'Enter input file name: '
        read(*,*)input
        endif

c       Read parameter file:
        call readpar()

c       Create event directory, copy event file in it, change to event directory,
c       run rdseed, and clean the copied seed data file:
        root=currdir(0,1)
        call mkdir('dir_'//input(1:index(input,'  ')-1),0,1)
        call choose_copy_path(root,root(1:index(root,'  ')-1)//'\'//'dir_'
     *  //input(1:index(input,'  ')-1),input)
        call changedir(root(1:index(root,'  ')-1)//'\'//'dir_'//input,0,1)
        call runner
     *          ('rdseed -R -d -o 6 -f '//input(1:index(input,' ')-1),0,1)
        call runner('del '//input,0,1)

c       Make list of SAC data and response files, open and read them:
        call runner('dirf *sac_asc',0,1)
        call runner('ren filenr.lis trace.lis',0,1)
        call runner('dirf resp.*',0,1)
        call runner('ren filenr.lis resps.lis',0,1)
        open(unit=1,file='resps.lis',status='old')
        open(unit=2,file='trace.lis',status='old')
        do 10 i=1,1000,1
        read(1,'(a80)',end=11,err=11)reslis(i)
10      continue
11      n_resp=i-2
        close(unit=1)
        do 20 i=1,1000,1
        read(2,'(a80)',end=21,err=21)tralis(i)
20      continue
21      n_tra=i-2
        close(unit=2)

c       Check if every trace has a response file (SIGNAL IF NOT):
        if(n_resp.ne.n_tra)write(*,*)'Error, event and trace mismatch!!!'

c       Generate station/component directories, and copy corresponding event and
c       response files in that directory.
        current=currdir(0,1)
```

```
        do 50 i=1,n_resp,1
        read(reslis(i)(8:),'(a)')resdir
        call mkdir(resdir(6:index(resdir,' ')-1),0,1)
c       call choose_copy_path(current,current(1:index(current,' ')-1)//'\'
c     * //resdir(6:index(resdir,' ')-1),'*'//resdir(6:index(resdir,' ')-1)
c     * //'*')
        call runner('copy '//current(1:index(current,' ')-1)//'\*'//resdir
     *  (6:index(resdir,' ')-1)//'* '//current(1:index(current,' ')-1)//'\
     *  '//resdir(6:index(resdir,' ')-1)//'\*.*',0,1)

c       USEFUL AND IMPORTANT LINES                      These two lines provide directory names
        and paths
c       VVVVVVVVVVVVVVVVVVVVVVVVVV                      to subfolders of recorded traces and
        spectra
        sta_nam(i)=resdir(6:index(resdir,' ')-1)
        sta_path(i)=current(1:index(current,' ')-1)//'\'//resdir(6:index(r
     *  esdir,' ')-1)
c       ^^^^^^^^^^^^^^^^^^^^^^^^^^
c       USEFUL AND IMPORTANT LINES
        call runner('del '//'*'//resdir(6:index(resdir,' ')-1)//'*',0,1)
50      continue

c       Start processing data of each station/component:
        do 100 i=1,n_resp,1
        call changedir(sta_path(i),0,1)
        call readresp('resp.'//sta_nam(i))
        call repair_trace(tralis(i)(8:))
        call readtrace(tralis(i)(8:))
        deglitched=trace
                !KA20080508 Order of steps changed after discussion with Hadi
        call deglitcher(deglitched,nsamp,iglit)
        !KA20080508 Order of steps changed after discussion with Hadi
        call detrender(deglitched,nsamp,itrend)
        !KA20080508 Order of steps changed after discussion with Hadi
        call windower(detrended,nsamp,taper,windowed)
        !KA20080508 Order of steps changed after discussion with Hadi
        call padder(windowed,nsamp,padded,npoints)
        !KA20080508 Order of steps changed after discussion with Hadi
c       call detrender(trace,nsamp)
                !KA20080508 Order of steps changed after discussion with Hadi
c       call windower(detrended,nsamp,taper,windowed)
        !KA20080508 Order of steps changed after discussion with Hadi
c       deglitched=windowed
                        !KA20080508 Order of steps changed after discussion with Hadi
c       call deglitcher(deglitched,nsamp,iglit)
        !KA20080508 Order of steps changed after discussion with Hadi
c       call padder(deglitched,nsamp,padded,npoints)
        !KA20080508 Order of steps changed after discussion with Hadi

        fourier=cmplx(padded,0.)
        call fork(npoints,fourier,-1.)
        call filter(fourier,sr,npoints,ifilt,norder, flcut,fhcut,filtered)
        call respremover(filtered,sr,npoints,iresp,respremoved_sp)
        call tracemaker(respremoved_sp,npoints,respremoved_tr)
        call make_GSE_resp()
        call make_PITSA_PAZ()
        call make_SAC_PAZ()
        call make_FAP_resp()
        call make_PSA(sr,npoints,respremoved_tr(:,3))
        call writer(npoints,deglitched,windowed,fourier,filtered,
     *                  respremoved_sp,respremoved_tr)
c       call writer(npoints,trace,deglitched,detrended,windowed,
c    *                  fourier,filtered,respremoved_sp,respremoved_tr)
100     continue
        end
```

```
        subroutine writer(npoints,deglitched,windowed,fourier,filtered,
     *                  respremoved_sp,respremoved_tr)
c       subroutine writer(nsamp,trace,deglitched,detrended,windowed,
c    *                  fourier,filtered,respremoved_sp,respremoved_tr)
        dimension trace(100000),detrended(100000),windowed(100000)
        dimension deglitched(100000),respremoved_tr(100000,3)
        complex fourier(100000),filtered(100000)
        complex respremoved_sp(100000,3)
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
        complex p(100),z(100)
        character respname*30,event*12,stime*12,staname*4,comp*3,tsr*9
        common /par/igmtn,iglit,itrend,ifilt, iresp,taper,norder,
     *  flcut, fhcut, ismooth, fbox, freq1, freq2, nfreq, damp
        common /respinfo/A0,f0,nzero,npole,p,z,sensitivity,A0ok
        common /traceinfo/sr,nsamp,trace,event,stime,staname,comp
        common /stainfo/stalat,stalon,stah
        common /detrend/a,b,detrended
        common /writeinfo/iyr,idoy,mon,iday,respname
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
        open(unit=3,file='result.txt',status='unknown')
        write(3,'(a19)')'RECORD INFORMATION:'
        write(3,20)iyr,mon,iday,stime
20      format('Beginning of record:  ',i4,2('/',i2.2),' ',a12)
        write(3,21)sr
21      format('Sampling rate:        ',f9.4)
        write(3,22)nsamp,npoints
22      format('# of samples:         ',i6,/,'# after padding:      ',i6)
        write(3,'(a20)')'STATION INFORMATION:'
        write(3,23)staname,comp
23      format('Station name:         ',a4,/,'Component:            ',a3)
        write(3,231)stalat,stalon,stah
231     format('Station latitude:     ',f8.4,/,'Station longitude:    ',f9
     *  .4,/,'Station height:       ',f8.2)
                if(iglit.eq.0)then
                write(3,24)
24              format('GLITCHES:',/,'Glitches removed:     No')
                else
                write(3,25)
25              format('GLITCHES:',/,'Glitches removed:     Yes')
                endif
        if(itrend.eq.0)then
        write(3,26)
26      format('TREND AND OFFSET:',/,'Trend/offset removed: No')
        else
        write(3,27)
27      format('TREND AND OFFSET:',/,'Trend/offset removed: Yes')
        write(3,'(a)')'A and B values in equation: y=Ax+B'
        write(3,28)a,b
28      format('A and B values:       ',2(g13.5,2x))
        endif
        write(3,29)taper
29      format('TAPERING:',/,'Tapering window type: Cosine',/,'Tapering wi
     *  ndow ratio: ',f6.5)
                if(ifilt.eq.0)then
                write(3,30)
30              format('FILTER:',/,'Filter applied:       No')
                else
                write(3,31)
31              format('FILTER:',/,'Filter applied:       Yes')
                write(3,32)
32              format('Filter type:          Butterworth')
                write(3,33)norder
33              format('Filter order:         ',i2)
                write(3,34)flcut, fhcut
34              format('Frequency range:      ',f7.3,' to ',f7.3,' Hertz')
                endif
        write(3,'(a32)')'INSTRUMENT RESPONSE INFORMATION:'
        if(iresp.eq.0)then
        write(3,'(a)')'Instrument correction: Not applied'
```

```
        else
        write(3,'(a)')'Instrument correction: Applied'
        write(3,35)A0
35      format('Normalization factor: ',g13.5)
        write(3,36)f0
36      format('Normalization freq.:  ',g13.5)
        write(3,37)nzero
37      format('Number of zeros:      ',i2,' in:')
        do i=1,nzero
        write(3,'(2(2x,g13.5))')real(z(i)),imag(z(i))
        enddo
        write(3,38)npole
38      format('Number of poles:      ',i2,' in:')
        do i=1,npole
        write(3,'(2(2x,g13.5))')real(p(i)),imag(p(i))
        enddo
        write(3,39)sensitivity
39      format('Overall sensitivity:  ',g13.5)
        if(A0ok.eq.1)then
        write(3,'(a)')'Normalization factor: Matches'
        else
        write(3,'(a)')'Normalization factor: Does not match'
        endif
        endif
        write(3,40)
40      format('Calibration files in SAC, SEED, GSE2, and PITSA formats cr
     *  eated in channel folder')
                write(3,'(a30)')'RESPONSE SPECTRUM INFORMATION:'
                write(3,'(a)')'PSA information file: PSA.out'
                if(ismooth.eq.0)then
                write(3,'(a)')'Response spectrum:    Not smoothed'
                else
                write(3,'(a)')'Response spectrum:    Smoothed'
                write(3,41)fbox
41              format('Smoothing box width:  ',f8.5,'Hz in each side')
                endif
        Write(3,'(a)')'END_HEADER'
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
        write(3,61)
61      format('   Sample #   |   Original   |  Deglitched  |  Detrended
     * |   Windowed   |<--   Fourier transform   -->| Fourier(Amp) |   F
     *  iltered   | Instr. corr. Disp., Vel., Acc. Amp. Spect. | Instr. co
     *  rr. Disp., Vel., Acc. time history|',/,'                  (Count)
     *       (Count)        (Count)        (Count)        Re(Count*s)
     *  Im(Count*s)     (Count*s)      (Count*s)      (Meter*s)       (Met
     *  er)       (Meter/s)       (Meter)       (Meter/s)       (M/s^2)')
        do 10 i=1,npoints,1
        write(3,'(15(g13.6,2x))')i,trace(i),deglitched(i),detrended(i),
     *  windowed(i),fourier(i),cabs(fourier(i)),cabs(filtered(i)),
     *  (cabs(respremoved_sp(i,j)),j=1,3),(respremoved_tr(i,j),j=1,3)
10      continue
        close(unit=3)
        return
        end



        subroutine make_PSA(sr,npoints,a)
        dimension a(100000),b(100000),freq(1000),psa(1000),smooth(1000)
        common /par/igmtn,iglit,itrend,ifilt, iresp,taper,norder,
     *  flcut, fhcut, ismooth, fbox, freq1, freq2, nfreq, damp
        open(unit=15,file='PSA.out',status='unknown')
        do 5 i=1,npoints,1
        b(i)=a(i)*100
5       continue
        call rscomp(sr,npoints,b,freq,psa)
        call PSA_smoother(nfreq,freq,psa,ismooth,fbox,smooth)
        write(15,*)'Calculated PSA for the frequency range: '
```

```
        write(15,*)freq1,' Hz and ',freq2,' Hz'
        write(15,*)nfreq,'frequencies distributed logarithmically'
        write(15,*)'Oscillator damping is',damp,'% of critical'
        write(15,'(a64)')'  Freq.(Hz)       Period(s)      PSA(cm/s^2)  Sm
     *  ooth PSA(cm/s^2)'
        do 10 i=nfreq,1,-1
        write(15,*)freq(i),1/freq(i),psa(i),smooth(i)
10      continue
        close(unit=15)
        return
        end



        subroutine PSA_smoother(nfreq,freq,psa,ismooth,fbox,smooth)
        dimension freq(1000),psa(1000),smooth(1000)
        if(ismooth.eq.0)then
        smooth=psa
        return
        endif
        do 10 i=1,nfreq,1
                n=0
                sum=0.
                fl=freq(i)-fbox
                fh=freq(i)+fbox
                do 20 j=1,nfreq,1
                        if(freq(j).ge.fl.and.freq(j).le.fh)then
                        sum=sum+psa(j)
                        n=n+1
                        endif
20              continue
        smooth(i)=sum/float(n)
10      continue
        return
        end



        subroutine rscomp(sr,npoints,a,freq,psa)
c       calculates response spectrum from acceleration time history
        dimension freq(1000),Omega(1000),psa(1000),a(*)
c       damp is % critical damping. freq2 is maximum freq. to consider.
        common /par/igmtn,iglit,itrend,ifilt, iresp,taper,norder,
     *  flcut, fhcut, ismooth, fbox, freq1, freq2, nfreq, damp
        pi=3.141593
        if (iabs(nfreq).gt.1000) then
                write (15,*)
                write (15,*) ' Nfreq cannot exceed 1000'
                return
        endif
c       Convert % damping to fraction
        beta=damp*0.01
c       Generate some logarithmically-spaced frequencies
        freq(1)=freq1
        omega(1)=2.*pi*freq(1)
        if (nfreq.ge.2) then
                frinc=alog(freq2/freq1)/(nfreq-1)
                do 120 k=2,nfreq
                        freq(k)=freq1*exp((k-1)*frinc)
                        omega(k)=2.*pi*freq(k)
120             continue
        endif
c       Call new response spectrum routine for each desired freq
        do 200 k=1,nfreq
                call sdcomp(a,npoints,omega(k),beta,1/sr,sd)
                psa(k)=sd*omega(k)*omega(k)
200     continue
```

```
      return
      end



      subroutine sdcomp(accg,na,omn,beta,dt,sd)
c     This is a modified version of "Quake.For", written by
c     Stavros A. Anagnostopoulos, Oct. 1986.  The formulation is that of
c     Nigam and Jennings (BSSA, v. 59, 909-922, 1969).  This modification
c     eliminates the computation of the relative velocity and absolute
c     acceleration; it returns only the relative displacement.
c     Dates: 05/06/95 - Modified by David M. Boore
c     This subroutine was implemented in agram developed by Gail Atkinson
      dimension accg(*)
      omt=omn*dt
      d2=1-beta*beta
      d2=sqrt(d2)
      bom=beta*omn
      d3=2.*bom
      omd=omn*d2
      om2=omn*omn
      omdt=omd*dt
      c1=1./om2
      c2=2.*beta/(om2*omt)
      c3=c1+c2
      c4=1./(omn*omt)
      ss=sin(omdt)
      cc=cos(omdt)
      bomt=beta*omt
      ee=exp(-bomt)
      ss=ss*ee
      cc=cc*ee
      s1=ss/omd
      s2=s1*bom
      s3=s2+cc
      a11=s3
      a12=s1
      a21=-om2*s1
      a22=cc-s2
      s4=c4*(1.-s3)
      s5=s1*c4+c2
      b11=s3*c3-s5
      b12=-c2*s3+s5-c1
      b21=-s1+s4
      b22=-s4
      sd=0.
      n1=na-1
      y=0.
      ydot=0.
      do 1 i=1,n1
      y1=a11*y+a12*ydot+b11*accg(i)+b12*accg(i+1)
      ydot=a21*y+a22*ydot+b21*accg(i)+b22*accg(i+1)
c     next two lines have been added for hardware portability
      if (y1.lt.1.e-30.and.y1.gt.0.) y1=0.
      if (ydot.lt.1.e-30.and.ydot.gt.0.) ydot=0.
      y=y1
      z=abs(y)
      z1=abs(ydot)
      if (z.gt.sd) sd=z
1     continue
      return
      end



      subroutine make_FAP_resp()
      complex p(100),z(100),accresp,velresp,dspresp
```

```
      common /respinfo/A0,f0,nzero,npole,p,z,sensitivity,A0ok
      open(unit=12,file='Response.FAP',status='unknown')
      write(12,20)
20    format(' Freq.(Hz)     Disp.Ampl.    Disp.Phase     Vel.Ampl.
     *  Vel.Phase     Acc.Ampl.    Acc.Phase')
      do 10 i=0,200,1
      f=10.**(-2.+i*0.02)
      call response(A0,f,nzero,npole,z,p,sensitivity,accresp,
     *                  velresp,dspresp)
      write(12,21)f,cabs(dspresp),phase(dspresp),cabs(velresp),
     *  phase(velresp),cabs(accresp),phase(accresp)
21    format(7(1x,g12.5,1x))
10    continue
      close(unit=12)
      return
      end




      subroutine make_SAC_PAZ()
      complex p(100),z(100)
      common /respinfo/A0,f0,nzero,npole,p,z,sensitivity,A0ok
      open(unit=14,file='SAC.PAZ',status='unknown')
              if(nzero.lt.10)then
              write(14,141)nzero
141           format('ZEROS ',i1)
              else
              write(14,142)nzero
142           format('ZEROS ',i2)
              endif
      if(nzero.ge.1)then
      do 10 i=1,nzero,1
      write(14,'(2(f11.4,2x))')real(z(i)),imag(z(i))
10    continue
      endif
              if(npole.lt.10)then
              write(14,143)npole
143           format('POLES ',i1)
              else
              write(14,144)npole
144           format('POLES ',i2)
              endif
      if(npole.ge.1)then
      do 20 i=1,npole,1
      write(14,'(2(f11.4,2x))')real(p(i)),imag(p(i))
20    continue
      endif
              write(14,145)sensitivity*A0
145           format('CONSTANT ',g13.5)
              close(unit=14)
      return
      end




      subroutine make_PITSA_PAZ()
      complex p(100),z(100)
      common /respinfo/A0,f0,nzero,npole,p,z,sensitivity,A0ok
      open(unit=13,file='PITSA.PAZ',status='unknown')
      write(13,'(a34)')'CAL1                            PAZ'
              if(npole.lt.10)then
              write(13,'(i1)')npole
              else
              write(13,'(i2)')npole
              endif
      if(npole.ge.1)then
      do 10 i=1,npole,1
```

```
      write(13,'(2(g8.3E1))')real(p(i)),imag(p(i))
10    continue
      endif
              if(nzero.lt.10)then
              write(13,'(i1)')nzero
              else
              write(13,'(i2)')nzero
              endif
      if(nzero.ge.1)then
      do 20 i=1,nzero,1
      write(13,'(2(g8.3E1))')real(z(i)),imag(z(i))
20    continue
      endif
              write(13,'(g8.3)')sensitivity*A0
              close(unit=13)
      return
      end



      function phase(x)
      complex x
      if(real(x).gt.0.)then
      phase=atan(imag(x)/real(x))
      elseif(real(x).eq.0.)then
              if(imag(x).gt.0.)phase= 2*atan(1.)
              if(imag(x).lt.0.)phase=-2*atan(1.)
      elseif(real(x).lt.0.)then
              if(imag(x).gt.0.)phase=atan(imag(x)/real(x))+4*atan(1.)
              if(imag(x).eq.0.)phase=4*atan(1.)
              if(imag(x).lt.0.)phase=atan(imag(x)/real(x))-4*atan(1.)
      endif
      phase=phase*45./atan(1.)
      return
      end



      subroutine make_GSE_resp()
      dimension trace(100000)
      complex p(100),z(100)
      character event*12,stime*12,staname*4,comp*3,respname*30
      common /respinfo/A0,f0,nzero,npole,p,z,sensitivity,A0ok
      common /traceinfo/sr,nsamp,trace,event,stime,staname,comp
      common /writeinfo/iyr,idoy,mon,iday,respname
      PI=4*atan(1.)
      read(event(1:4),'(i4)')iyr
      read(event(6:8),'(i3)')idoy
      call doy2md(iyr,idoy,mon,iday)
      respname(1:4)=staname
      do 10 i=1,4,1
      if(respname(i:i).eq.' ')respname(i:i)='_'
10    continue
      respname(5:5)='_'
      respname(6:7)=comp(1:2)
      respname(8:8)='_'
      respname(9:9)=comp(3:3)
      respname(10:10)='.'
      respname(11:14)=event(1:4)
      write(respname(15:29),15)mon,iday
15    format('-',i2.2,'-',i2.2,'-0000_GSE')
      sensGSE=10.**9/(sensitivity*2*PI*f0)
      sfactor=A0/10.**9
      open(unit=11,file=respname,status='unknown')
      write(11,21)staname,comp,sensGSE,1/f0,iyr,mon,iday
21    format('CAL2 ',a5,1x,a3,13x,e10.2,1x,f7.3,18x,i4,'/',i2.2,'/',i2.2
     *  ,' 00:00')
```

```
c21     format('CAL2 ',a5,1x,a3,13x,e15.8,1x,f7.3,13x,i4,'/',i2.2,'/',i2.2
c     * ,' 00:00')
        write(11,22)sfactor,npole,nzero+1
22      format('PAZ2  1 V ',e15.8,15x,i3,1x,i3,' Laplace transform')
        do 100 i=1,npole,1
        write(11,23)real(p(i)),imag(p(i))
100     continue
        do 110 i=1,nzero,1
        write(11,23)real(z(i)),imag(z(i))
110     continue
        write(11,23)0.,0.
23      format(2(1x,e15.8))
        write(11,24)sensitivity,sr
24      format('DIG2  2 ',e15.8,1x,f11.5)
        close(unit=11)
        return
        end



        subroutine doy2md(iyr,idoy,mon,iday)
        dimension imonth(12)
        data (imonth(i),i=1,12)/31,28,31,30,31,30,31,31,30,31,30,31/
        if(mod(iyr,4).ne.0) goto 10
        imonth(2)=29
        if(mod(iyr,100).eq.0)then
        imonth(2)=28
                if(mod(iyr,400).eq.0)then
                imonth(2)=29
                endif
        endif
10      iday=idoy
        do 20 i=1,12,1
                if(iday-imonth(i).le.0)then
                mon=i
                goto 30
                endif
        iday=iday-imonth(i)
20      continue
30      return
        end



        subroutine tracemaker(respremoved_sp,npoints,respremoved_tr)
        dimension respremoved_tr(100000,3)
        complex respremoved_sp(100000,3),temp(100000)
        do 10 i=1,3,1
        temp=respremoved_sp(:,i)
        call fork(npoints,temp,+1.)
        respremoved_tr(:,i)=real(temp)
10      continue
        return
        end



        subroutine respremover(filtered,sr,npoints,iresp,respremoved_sp)
        complex filtered(100000),respremoved_sp(100000,3),S
        complex p(100),z(100),accresp, velresp, dspresp
        common /respinfo/A0,f0,nzero,npole,p,z,sensitivity,A0ok
        PI=4.*atan(1.)
        ncalc=npoints/2+1
        W0=2.*PI*sr/npoints
        if(iresp.eq.0)then
        do 10 i=1,ncalc,1
```

```
      W=(i-1)*W0
      S=cmplx(0.,W)
              if(i.ne.1)then
                      respremoved_sp(i,1)=filtered(i)/S
                      respremoved_sp(npoints+2-i,1)=respremoved_sp(i,1)
              else
                      respremoved_sp(i,1)=filtered(i)/cmplx(0.,W0/100)
              endif
                      respremoved_sp(i,2)=filtered(i)
                      respremoved_sp(i,3)=filtered(i)*S
              if(i.gt.1)then
                      respremoved_sp(npoints+2-i,1)=conjg(respremoved_sp(i,1))
                      respremoved_sp(npoints+2-i,2)=conjg(respremoved_sp(i,2))
                      respremoved_sp(npoints+2-i,3)=conjg(respremoved_sp(i,3))
              endif
10    continue
      else
      do 20 i=1,ncalc,1
      f=(i-1)*sr/npoints
      call response(A0,f,nzero,npole,z,p,sensitivity,accresp,velresp,
     *                  dspresp)
              respremoved_sp(i,1)=filtered(i)/dspresp
              respremoved_sp(i,2)=filtered(i)/velresp
              respremoved_sp(i,3)=filtered(i)/accresp
              if(i.gt.1)then
              respremoved_sp(npoints+2-i,1)=conjg(respremoved_sp(i,1))
              respremoved_sp(npoints+2-i,2)=conjg(respremoved_sp(i,2))
              respremoved_sp(npoints+2-i,3)=conjg(respremoved_sp(i,3))
              endif
20    continue
      endif
      return
      end



      subroutine filter(fourier,sr,npoints,ifilt,
     *                norder, flcut,fhcut,filtered)
      complex fourier(100000),filtered(100000)
      if(ifilt.eq.0)then
      filtered=fourier
      return
      endif
      ncalc=npoints/2+1
      do 10 i=1,ncalc,1
      f=(i-1)*sr/npoints
      filtered(i) = cmplx(buttrlcf(f, flcut, norder),0)*fourier(i)
      filtered(i) = cmplx(buttrhcf(f, fhcut, norder),0)*filtered(i)
              if(i.eq.1)goto 10
      filtered(npoints+2-i) = conjg(filtered(i))
10    continue
      return
      end

      function buttrlcf(f, fcut, norder)
c**   norder Butterworth low-cut filter response from AGRAM
c**   taken from Boore, 1996
      buttrlcf = 1.
      if (fcut .eq. 0.) return
      buttrlcf = 0.
      if (f .eq. 0.) return
      buttrlcf = 1./ (1. + (fcut/f)**(2*norder))
      return
      end

      function buttrhcf(f, fcut, norder)
c**   norder Butterworth high-cut filter response from AGRAM
c**   taken from Boore, 1996
```

```
      real buttrhcf
      buttrhcf = 1.
      if (fcut .eq. 0.) return
      buttrhcf = 0.
      if (f .eq. 0.) return
      buttrhcf = 1./ (1. + (f/fcut)**(2*norder))
      return
      end
```

```
      subroutine fork(lx,cx,signi)
c     fast fourier transform routine from Dave Boore.
c     result of sequence from time to freq to time requires no scaling.
      complex cx,carg,cexp,cw,ctemp
      dimension cx(lx)
      j=1
      sc=sqrt(1./lx)
c     write(*,*)'***** FAST FOURIER TRANSFORM CALLED *****'
      do 5 i=1,lx
        if(i .gt. j) go to 2
        ctemp = cx(j)*sc
        cx(j) = cx(i)*sc
        cx(i) = ctemp
2       m = lx/2
3       if(j .le. m) go to 5
          j = j - m
          m = m / 2
          if(m .ge. 1) go to 3
5     j = j + m
      l = 1
c     write(*,*)'First step completed'
6     istep = 2 * l
      temp = 3.14159265 * signi/l
      do 8 m = 1, l
        carg=(0., 1.) * temp * (m-1)
        cw=cexp(carg)
        do 8 i = m, lx, istep
             ctemp = cw *cx(i+l)
             cx(i+l) = cx(i) - ctemp
8     cx(i) = cx(i) + ctemp
      l = istep
      if(l .lt. lx) go to 6
c     write(*,*)'***** FAST FOURIER TRANSFORM DONE *****'
9     return
      end
```

```
      subroutine padder(windowed,nsamp,padded,npoints)
      dimension windowed(100000),padded(100000)
      padded=0.
      do 10 i=1,nsamp,1
      padded(i)=windowed(i)
10    continue
      do 20 i=1,16,1
      if(2**i.ge.nsamp)then
      npoints=2**i
      goto 50
      endif
20    continue
50    return
      end
```

```
        subroutine deglitcher(x,npts,iglit)
c
c       Removes glitches from array x and makes corresponding
c       correction to valmax.
c       Anything >10* the (log) avg of a running 20-pt. amplitude
c       average is assumed to be a glitch.  Replaced with avg of
c       neighbour values.
c       Constant amplitude steps (more than 20 identical values
c       in a row) are assumed to be glitches.  Replaced with 0
c
c       G.M. Atkinson, Oct. 1990
c       Revised Aug. 1991
c
        dimension x(100000)
c       valmax = 0.
c
c       First remove constant amplitude steps.
c       If next 20 values are exactly the same as this one,
c       we assume its an erroneous step.
c
        if(iglit.eq.0)goto 105

        jstop = 0
        do 500 j=1,npts-20
        iflag = 1
        do 400 jj = 1,20
          if(x(j+jj) .ne. x(j)) iflag = 0
400     continue
                        if (iflag .eq. 1) then
c                       This is a step.
                                jstop = j + 20
                        endif
        if (jstop .ge. j) x(j) = x(j+20)
500     continue
c       write(*,*)'Finished looking for a Long Glitch'
c
        ntotal = 0
49      continue
c        write(*,*)'Starting glitch removal'
        runavg = 0.
        nspike = 0
c        write(*,*)'Calculating Run Average'
        do 50 jj = 1,21
          if(x(jj) .eq. 0.) x(jj)=1.
          runavg = runavg + alog10(abs(x(jj)))
50      continue
        runavg=runavg/21.
c       write(*,*)'Looking for Spikes'
        do 100 j=11,npts-11
           if(x(j-10) .eq. 0.) x(j-10)=1.
           if(x(j+11) .eq. 0.) x(j+11)=1.
           runavg =runavg - alog10(abs(x(j-10)))/21.
     *            + alog10(abs(x(j+11)))/21.
           spike = 10.* 10.**runavg
           if(abs(x(j)) .gt. spike) then
             nspike = nspike + 1
             x(j) = (x(j-1) + x(j+1))/2.
             if(abs(x(j)) .gt. spike)then
               x(j) = (x(j-2) + x(j+2))/2.
               if(abs(x(j)) .gt. spike) x(j) = 1.
             endif
           endif
c        if(abs(x(j)) .gt. valmax) valmax = abs(x(j))
100     continue
        ntotal = ntotal + nspike
        if (nspike .gt. 1) go to 49
105     return
        end
```

```
      subroutine windower(trace,nsamp,taper,windowed)
      dimension trace(100000),windowed(100000)
      do 10 i=1,nsamp,1
      call win(i,1,nsamp,int4(nsamp*taper),wind)
      windowed(i)=trace(i)*wind
10    continue
      return
      end


      subroutine win(i, nstart, nstop, ntaper, wind)
c     applies cosine tapered window.
c     unit amplitude assumed
c     written by D. M. Boore
c     latest revision: 9/26/95
      wind = 0.0
      if ( i .lt. nstart .or. i. gt. nstop) return
      wind = 1.0
      if ( i .ge. nstart+ntaper .and. i .le. nstop-ntaper ) return
      pi = 4.0 * atan(1.0)
      dum1 = (nstop+nstart)/2.0
      dum2 = (nstop-nstart-ntaper)/2.0
      wind = 0.5 * (1.0 - sin( pi*
     * ( abs(float(i)-dum1) - dum2 ) /float(ntaper) ) )
      return
      end


      subroutine detrender(deglitched,nsamp,itrend)
      dimension deglitched(100000),detrended(100000)
      doubleprecision x11,x12,x21,x22,y11,y21
      common /detrend/a,b,detrended
      if(itrend.eq.0)then
      detrended=deglitched
      return
      endif
      x11=0.
      x12=real(nsamp)
      x21=0.
      x22=0.
      y11=0.
      y21=0.
      do 10 i=1,nsamp,1
      x11=x11+i
      x21=x21+i**2
      y11=y11+deglitched(i)
      y21=y21+i*deglitched(i)
10    continue
      x22=x11
      a=(x22*y11-x12*y21)/(x11*x22-x12*x21)
      b=(x11*y21-x21*y11)/(x11*x22-x12*x21)
      do 20 i=1,nsamp,1
      detrended(i)=deglitched(i)-(i*a+b)
20    continue
      return
      end


      subroutine response(A0,f,nzero,npole,z,p,sensitivity,
     *                                 accresp, velresp, dspresp)
      complex z(100), p(100),S
```

```
      complex accresp, velresp, dspresp, Xnum, Xdenom
      PI=4.*atan(1.)
      if(f.eq.0.)f=0.01
      W=2.*PI*f
      S=CMPLX(0.0,W)
      Xnum = CMPLX(A0*sensitivity,0.)
      do iz=1, nzero
        Xnum = Xnum*(S-z(iz))
      enddo
      Xdenom = CMPLX(1.,0.)
      do ip = 1, npole
        Xdenom = Xdenom*(S-p(ip))
      enddo
      velresp= Xnum/Xdenom
      dspresp=VELRESP*S
      accresp=VELRESP/S
      return
      end



      subroutine readresp(fname)
      complex p(100),z(100),accresp, velresp, dspresp
      character fname*(*),respread*80(3000)
      common /respinfo/A0,f0,nzero,npole,p,z,sensitivity,A0ok
      open(unit=4,file=fname,status='unknown')
      do 10 i=1,3000,1
      read(4,'(a80)',end=11,err=11)respread(i)
10    continue
11    nline=i-1
      close(unit=4)
      istage1=0
      istageL=0
      iZeroCount=0
      iPoleCount=0
      A0ok=1.
      do 20 i=1,nline,1
      if(respread(i)(1:7).eq.'B060F03')then
      read(respread(i)(52:),'(i6)')nstage
      endif
      if(respread(i)(1:7).eq.'B060F04')then
      read(respread(i)(52:),'(i6)')n_curr_stage
      endif
      if(n_curr_stage.eq.1)istage1=1
      if(n_curr_stage.ne.1)istage1=0
      if(n_curr_stage.eq.0)istageL=1
      if(istage1.eq.1.and.respread(i)(1:7).eq.'B043F08')
     * read(respread(i)(52:),*)A0
      if(istage1.eq.1.and.respread(i)(1:7).eq.'B043F09')
     * read(respread(i)(52:),*)f0
      if(istage1.eq.1.and.respread(i)(1:7).eq.'B043F10')
     * read(respread(i)(52:),*)nzero
      if(istage1.eq.1.and.respread(i)(1:7).eq.'B043F15')
     * read(respread(i)(52:),*)npole
      if(istage1.eq.1.and.respread(i)(1:7).eq.'B043F11')then
      iZeroCount=iZeroCount+1
      read(respread(i)(17:),*)z1,z2
      z(iZeroCount)=cmplx(z1,z2)
      endif
      if(istage1.eq.1.and.respread(i)(1:7).eq.'B043F16')then
      iPoleCount=iPoleCount+1
      read(respread(i)(17:),*)p1,p2
      p(iPoleCount)=cmplx(p1,p2)
      endif
      if(istageL.eq.1.and.respread(i)(1:7).eq.'B048F05')
     * read(respread(i)(52:),*)sensitivity
20    continue
      call response(A0,f0,nzero,npole,z,p,sensitivity,
```

```
     *              accresp, velresp, dspresp)
      if(abs(cabs(velresp/sensitivity)-1).gt.0.0001)A0ok=0.
      return
      end




      subroutine readtrace(fname)
      dimension trace(100000)
      character fname*(*),event*12,stime*12,staname*4,comp*3
      common /traceinfo/sr,nsamp,trace,event,stime,staname,comp
      common /stainfo/stalat,stalon,stah
      open(unit=5,file=fname,status='old')
      read(5,*)sr1
      sr=1/sr1
      do 9 ii=1,5,1
      read(5,*)
9     continue
      read(5,*)x,stalat,stalon,stah
      do 10 ii=1,7,1
      read(5,*)
10    continue
      read(5,*)yr,doy,hour,min,sec
      write(event,50)int(yr),int(doy)
50    format(i4.4,'/',i3.3,'    ')
      write(stime,51)int(hour),int(min),int(sec)
51    format(2(i2.2,':'),i2.2,'.000')
      read(5,'(44x,i6)')nsamp
      do 11 ii1=1,6,1
      read(5,*)
11    continue
      read(5,'(a4)')staname
      do 12 ii2=1,5,1
      read(5,*)
12    continue
      read(5,'(16x,a3)')comp
      read(5,*)
      read(5,*) (trace(i),i=1,nsamp)
      close(unit=5)
      return
      end




      subroutine repair_trace(fname)
      character fname*(*),line*100(100000)
      call runner('copy '//fname//' original.sac',0,1)
      open(unit=21,file=fname,status='unknown')
      do 10 i=1,100000,1
      read(21,'(a100)',end=11,err=11)line(i)
10    continue
11    nline=i-1
      do 20 i=31,nline,1
              do 25 j=15,75,15
26                    if(line(i)(j:j+1).eq.'00')then
                              do 30 k=j,99,1
                              line(i)(k:k)=line(i)(k+1:k+1)
30                            continue
                      goto 26
                      elseif(line(i)(j:j).eq.'0')then
                      line(i)(j:j)=' '
                      endif
25            continue
20    continue
      rewind(unit=21)
      do 50 i=1,nline,1
      write(21,'(a)')line(i)(1:len(line(i)))
```

```
50      continue
        close(unit=21)
        return
        end



        subroutine readpar()
        common /par/igmtn,iglit,itrend,ifilt, iresp,taper,norder,
     *  flcut, fhcut, ismooth, fbox, freq1, freq2, nfreq, damp
        open(unit=6,file='parameter.dat',status='old')
        open(unit=61,file='parameter.out',status='unknown')
        read(6,*)iglit,itrend,ifilt,iresp
        read(6,*)taper
        read(6,*)norder, flcut, fhcut
        read(6,*) freq1, freq2, nfreq, damp
        read(6,*)ismooth, fbox
        write(61,*)iglit,itrend,ifilt,iresp
        write(61,*)taper
        write(61,*)norder, flcut, fhcut
        write(61,*) freq1, freq2, nfreq, damp
        write(61,*)ismooth, fbox
        if (taper .gt. .50)write(*,*)' ERROR. Use taper<.5'
        close(unit=6)
        close(unit=61)
        return
        end



        function rounder(x,n)
        y=x*10.**n
        k=nint(y)
        rounder=real(k/10.**n)
        return
        end



        subroutine choose_copy_path(from_dir,to_dir,filename)
c       This subroutine copies the file "filename" from directory "from_dir" to
c       directory "to_dir" and for doing this it calls subroutine "runner".
        character from_dir*(*),to_dir*(*),from_file*160,to_file*160
        character temp*256,filename*(*)
        length1=index(from_dir,'  ')-1
        if(length1.eq.-1)length1=len(from_dir)
        from_file=from_dir(1:length1)//'\'//filename
        length2=index(to_dir,'  ')-1
        if(length2.eq.-1)length2=len(to_dir)
        to_file=to_dir(1:length2)//'\'//filename
        temp= 'copy '//from_file(1:index(from_file,'  ')-1)//' '//to_file
     *  (1:index(to_file,'  ')-1)
        call runner(temp(1:index(temp,'  ')-1),0,1)
        end



        subroutine runner(argument,ireport,iolog)
c       This subroutine puts the argument in the parenthesis in the system shell
c       command. If the argument is something like "copy file1 file2" it will do
c       exactly a task of copying. This subroutine is called by many other ones.
c       It reports the success of this task to unit "iolog" (will not if ireport=0).
        USE PORTLIB
        character argument*(*)
```

```
      integer(4) i
      i=system(argument)
      if(ireport.eq.0)return
      if(i.eq.-1)then
      write(iolog,*)argument,' was not successful.'
      else
      write(iolog,*)argument,' was successful.'
      endif
      end



      subroutine mkdir(dirname,ireport,iolog)
c     Function currdir finds the current working directory and assigns the name
c     to currdir, then it reports the success of this task to unit "iolog" (will
c     not if ireport=0).
      USE MSFLIB
      character dirname*(*)
      logical(4) imakedirqq
      imakedirqq=makedirqq(dirname)
      if(ireport.eq.0)return
      if (imakedirqq.gt.0) then
      write (iolog,1) dirname
1     format(' The directory "',a,'" was generated successfully.')
      else
      write (iolog,2) dirname
2     format(' The directory "',a,'" was not generated successfully.')
      endif
      return
      end



      function currdir(ireport,iolog)
c     Function currdir finds the current working directory and assigns the name
c     to currdir, then it reports the success of this task to unit "iolog" (will
c     not if ireport=0).
      USE MSFLIB
      character currdir*(*)
      integer(4) igetdrivedirqq
      currdir=FILE$CURDRIVE
      igetdrivedirqq=getdrivedirqq(currdir)
      if(ireport.eq.0)return
      if (igetdrivedirqq.gt.0) then
      write (iolog,1) currdir(1:index(currdir,' ')-1)
1     format(' The current directory is: "',a,'".')
      else
      write (iolog,2) currdir(1:index(currdir,' ')-1)
2     format(' The cuurent directory lenght too long.')
      endif
      return
      end



      subroutine changedir(dir,ireport,iolog)
c     Subroutine changedir changes the current directory to the directory "dir"
c     and reports the (un)successful change of directory to unit "iolog" (will
c     not if ireport=0).
      USE MSFLIB
      character dir*(*)
      logical(4) ichangedirqq
      ichangedirqq=changedirqq(dir)
      if(ireport.eq.0)return
      if (ichangedirqq) then
      write (iolog,1) dir
```

```
1       format(' The change directory "',a,'" was successful.')
        else
        write (iolog,2) dir
2       format(' The change directory "',a,'" was not successful.')
        endif
        return
        end
```