ELECTRONIC SEISMOLOGIST

Computer Programs in Seismology: An Evolving Tool for Instruction and Research

by Robert B. Herrmann

INTRODUCTION

Earthquake seismology, like most areas of geophysics, is a fascinating mix of theory, computation, and observation. The past 50–60 years of earthquake seismology can be described as a synergistic interaction between the expanding quantity and improving quality of seismic data and important advances in practical wave-propagation physics and computation. An important impetus for many of these developments was the need to monitor and characterize nuclear explosions in the atmosphere, oceans, and underground, which stimulated a major investment in seismology during the 1960s that, combined with the plate tectonics revolution, initiated large growth in the field. Substantial effort was invested to produce standard analysis packages and software tools for the processing and analysis of seismic observations. As a result, high-quality seismic data are readily accessible and sufficient computational power to routinely analyze these observations is available to almost everyone. Turn-key seismic networks are available from manufacturers who provide sensors, data acquisition and transmission, event location, and archival functions in a manner that requires training, but not a detailed understanding of the of the hardware or software packages.

As we advance toward more sophisticated analysis of ever-larger data sets, including the effects of 3D structure, the practical and theoretical challenges facing students are substantial. Students must adapt to research-quality software and use it to develop the deep insight into seismic-wave excitation and propagation before they can confidently apply the concepts to more sophisticated earth and earthquake models. Although beginning students have the skills to use some tools such as word processors and have some scripting experience, many are less comfortable extending old or developing new analysis tools or even using existing codes to tackle realistic 1D problems. The challenge facing educators is to enable students to quickly become proficient in basic seismic data processing so that they can move on to research that is more meaningful. The *Computer Programs in Seismology* (CPS) package provides a set of tools and includes a number of tutorials that may help the young researchers develop essential skills needed for the study of the Earth and seismic sources.

The purpose of this article is to review the history of the package, to show how it can be used, and to highlight a few of the programs that are heavily used in data analysis. The history of the software development is one that has probably been repeated in most larger research groups, and we review it partially to provide some context for the younger readers, who work in a very different computational environment than existed even ten years ago.

HISTORY

CPS Version 1

The first version of CPS (Herrmann, 1978) was developed by Herrmann between 1971 and 1974 as part of his Ph.D. studies at Saint Louis University, which focused on determining earthquake focal mechanisms, depths, and seismic moment from the analysis of surface waves. The codes included general-purpose utility, spectral analysis, and surface-wave tools for data analysis and formal geophysical inversion. All codes were developed on punched cards with processing at the single central computer of the university. These codes were then used by other students for their research.

CPS Version 2

In 1981 the Department of Earth and Atmospheric Sciences received a Digital Equipment Corporation (DEC) PDP 11/34 minicomputer for seismic data acquisition and a DEC PDP 11/70 for data analysis to support the United States Geological Survey (USGS)-funded regional seismic network. The PDP 11/70 ran under a *UNIX V7* license (http://en.wikipedia.org/wiki/Version_7_Unix, last accessed September 2013). Much software development was done by faculty and graduate students during this time. The software was distributed on magnetic tape or punched cards, often with printed documentation. The *UNIX* experience was beneficial because of the underlying philosophy of creating compatible, flexible tools to perform complicated tasks (Kernighan and Plauger, 1974). Scientific research relies heavily on graphics for insight and communication through publication and reports and, before the advent of modern 24 bit graphics monitors and laser and inkjet printers, the use of mechanical plotters, such as *CALCOMP* (http://en.wikipedia.org/wiki/Calcomp, last accessed September 2013), electrostatic plotters, for example, Versatec, interactive terminals, such as the Tektronix 4014 (http://terminals.classiccmp.org/wiki/index.php/Tektronix_4014, last accessed September 2013), or even dot matrix printers, for example, EPSON (http://

doi: 10.1785/0220110096

en.wikipedia.org/wiki/Dot_matrix_printing, last accessed September 2013) or Printronix (http://en.wikipedia.org/wiki/Printronix, last accessed September 2013) facilitated research. At the time, a major problem was that each output device required a unique set of software to produce a graphic. Fortunately, a logical framework of device-independent graphics was provided with *UNIX V7* through its plot filters. To simplify addressing all devices, a library that converted *CALCOMP* calls to plot calls was developed. The user program then created a binary file of basic plot calls, for example, pen up/down, pen move, which could then be directed to a specific device driver to produce an output similar to that of the mechanical plotter. Subsequent development of these *CALPLOT* libraries in CPS added interactive capabilities (mostly in 2D) with useful features, for example, a choice of cursors including crosshairs and hyperbolas.

CPS Version 3

Although the CPS *Version 2* software development focused on synthetic seismogram computation including the adaptation of the growing community software used and required for research, students continued to develop and extend the package necessary to perform research. As the package grew and assimilated more tools, a problem developed: each program had different input and output specifications. Successful use of the software often required a deep understanding of the seismology and a solid background in numerical analysis to insure that the program output was reasonable and reliable.

In 1996, most of the codes were rewritten, which is why most of the current programs have 96 as part of their names, for example, *hspec96*. The codes were and continue to be written with the following guidelines:

- All code is open source and extensively commented to ease maintenance.
- The package is complete in that all graphics libraries are provided. Only a *UNIX* or *LINUX*-like operating system environment and compilers are required to use the programs.
- A clearly defined seismic velocity-model format is used to drive multiple methods for creating synthetic seismograms.
- Synthetic seismogram codes output waveforms in a single-file format.
- A command-line help specification is built into each tool to reduce the need for external documentation. For example, the command-line controls are displayed by entering *program –h*.
- Program modifications are documented within the source code and on the package website.
- Tutorials for running the codes are available as separate pdf documents or in HTML.
- Graphics drivers support X11 and PostScript. Output to printers or conversion to other formats, for example, PNG, can be accomplished using other freely available programs such as *GraphicsMagick*, *ImageMagick*, or *ghostscript*. Limiting the graphics focus permits the package to focus on solving seismological problems rather than supporting the many graphics devices and formats.
- All codes execute on *Solaris*, *LINUX*, *MacOS-X*, and *Cygwin* (a Windows environment) on 32- and, later, 64-bit hardware architectures.
- When possible, all codes are tested using different compilers, with preference to the free open-source GNU C and FORTRAN compiler suite.

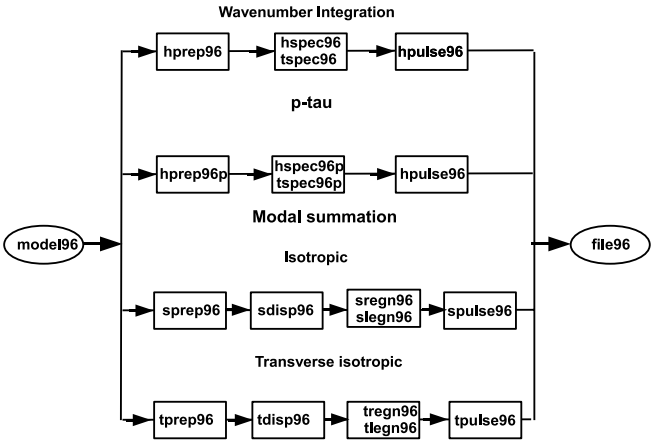
To illustrate the impact of these guidelines, consider the velocity model file in *model96* format (Table 1 for a simple crustal model). The velocity model format is simple ASCII with several fields available for researchers to use to document the model in a manner that does not require reading documentation to understand the velocity model.

Figure 1 illustrates the uniform sequence for making synthetic seismograms using four different methods, as well as the naming convention for the CPS programs. Generating

Table 1
SCM.mod

MODEL.01
Simple Crustal Model
ISOTROPIC
KGS
FLAT EARTH
1-D
CONSTANT VELOCITY
LINE08
LINE09
LINE10
LINE11

H (km)	V_P(km/s)	V_S(km/s)	RHO (GM/CC)	Q_P	Q_S	ETAP	ETAS	FREFP	FREFS
40.	6.0	3.5	2.8	0	0	0	0	1	1
00.	8.0	4.7	3.3	0	0	0	0	1	1

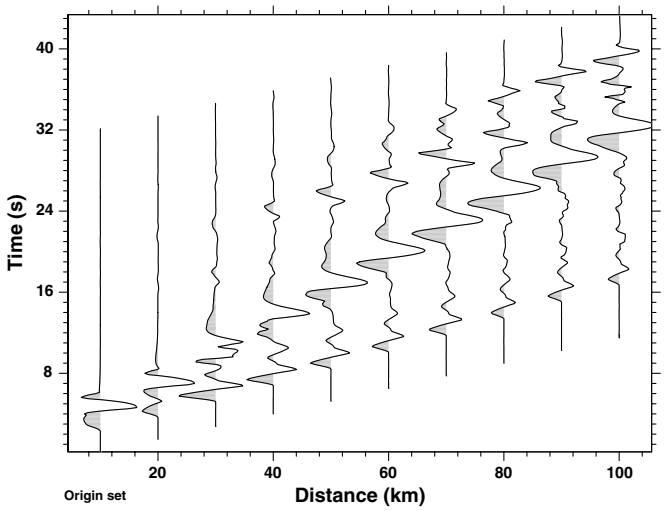


▲ **Figure 1.** Flowchart for making synthetic seismograms. The paradigm is that the velocity model drives the synthetic. The sequence of programs to be run depends on the particular technique.

a synthetic seismogram starts with a velocity model and leads to one or more time series stored in the *file96* ASCII format. A separate file-format conversion utility, *f96tosac*, is provided to convert the traces in this file to individual *SAC* files. Creation of a synthetic seismogram requires control files unique to each algorithm, here handled by the separate preparation programs *hprep96*, *hprep96p*, *tprep96*, and *sprep96*. After the computation sequence is completed, a source time function is applied by *hpulse96*, *tpulse96*, or *spulse96*. As an aid in remembering the program name, the first character of each program name usually indicates the function of the program, for example, *s* for surface wave, *h* for Haskell–Thomson, *p* for plotting, *f* for *file96* file manipulation, *r* for receiver function inversion, *j* for joint receiver function surface-wave dispersion inversion, *t* for transverse isotropic media, and so on. The graphics files created by some of the commands usually appear on the file system with uppercase names with a name similar to the program used to create them, for example, the binary file SHWMOD96.PLT is created by the program *shwmod96* that plots a velocity model. All of these conventions are designed to make the use of the programs intuitive.

The final item required to compute synthetic seismograms is a file giving the epicentral distance, sample rate, number of time samples in the signal, start time specified with an offset and a reduction velocity, for example, in a *dfile*, which consists of lines as

10.0	0.125	256	−1.0	8.0
20.0	0.125	256	−1.0	8.0
30.0	0.125	256	−1.0	8.0
40.0	0.125	256	−1.0	8.0
50.0	0.125	256	−1.0	8.0
60.0	0.125	256	−1.0	8.0
70.0	0.125	256	−1.0	8.0
80.0	0.125	256	−1.0	8.0
90.0	0.125	256	−1.0	8.0
100.	0.125	256	−1.0	8.0



▲ **Figure 2.** Vertical component Green's functions for a vertical strike-slip source. Positive amplitudes are shaded.

The format of this *dfile* is available from the command line help option in all of the *prep96* commands, for example, *hprep96 -h*. Creating wavenumber-integration synthetics for a source depth of 10 km and a receiver depth of 0 km just requires a few commands to create *SAC* files for the Green's functions for moment tensor sources and point forces:

```
hprep96 -M SCM.mod -d dfile -HS 10 -HR 0 -ALL
hspec96 > hspec96.out
hpulse96 −V −p −l  4 > h.vel
f96tosac −G < h.vel
```

Figure 2 presents a record section of the vertical-component Green's functions for a vertical strike-slip source. The plot was made using *gsac* and the *SAC* files (Goldstein *et al.*, 2003) created by *f96tosac*. This simple sequence is possible because *hprep96* intelligently provided the proper wavenumber sampling parameters to the wavenumber-integration program, *hspec96*.

CPS Version 3.30

The initial version of the *Version 3* software package was denoted as CPS 3.00 with the source and executables stored in a directory named PROGRAMS.300. Subsequent updates used different version numbers. However, once *Version 3.30* was reached, the package name was not changed in order to facilitate installation. The current distribution consists of 150 programs and executable shell scripts that permits the computation of synthetic seismograms in flat and spherical (using an Earth-flattening approximation) structures, inversion of surface-wave dispersion and receiver functions for lithospheric structure, the inversion of seismograms for source parameters, and for processing digital seismic data that are in the *SAC* file format. The distribution also contains pdf tutorials with titles "An Overview of Synthetic Seismogram Computation, Source Inversion, Surface Waves, Receiver Functions and Crustal Structure," "*CALPLOT* Graphics" which discusses

the C and FORTRAN graphics interface, and *GSAC* which discusses the use of the *gsac* program and routines to work with *SAC* files.

The package also includes codes for synthetic seismogram generation using generalized rays (*genray81* of C. Langston, personal comm., 1985) and asymptotic wave theory (Červený and Psenčik, 1981) that were modified to run within the CPS framework. In addition to a useful SHELL-based scripting program dialog, we also distribute versions of two Incorporated Research Institutions for Seismology (IRIS) tools, *rdseed* and *evalresp*. These two IRIS programs (http://www.iris.edu/dms/nodes/dmc/software/downloads/, last accessed September 2013) are included so that CPS 3.30 is a self-contained complete distribution for the processing and analysis of modern earthquake data sets. We have ensured that these codes compile on the target operating-system environments.

Because the programs *rdseed*, *evalresp*, and *gsac* are heavily used for processing earthquake recordings, it is useful to discuss these programs that are distributed in CPS and also indicate how they are used.

rdseed

CPS330 includes modifications of the IRIS *rdseedv5.3*. The first class of changes address the syntax required to overcome subtle differences in installed C compilers and support libraries to permit compilation on different platforms, although the differences are becoming fewer with newer versions of the *SEED* reader. The other class of changes modifies textual output and adds comments to the pole-zero files used by *SAC* or *gsac*. To illustrate the changes, *rdseed* is preferentially invoked from the command line, because this operation can be performed within a shell script:

```
rdseed -f 20020618.5710 -R -d -o 1 -p
```

The syntax of the command, from left to right, is to open the *SEED* volume 20020618.5710 obtained from the IRIS DMC, extract the response files (-R), the waveforms in SAC format (-d -o 1) and the responses in a pole-zero format (-p). All output of the command is redirected to the file *out*. For this example the waveform, response file, and pole-zero files created are, respectively:

```
2002.169.17.30.17.1480.IU.WCI..BHZ.D.SAC
RESP.IU.WCI..BHZ
SAC_PZs_IU_WCI_BHZ__1999.316.00.00.00.0000
   _2002.340.24.60.60.99999
```

The CPS 3.30 modification of *rdseed* also provides extra output to the screen (stderr) as shown in Table 2. This consists of one line for each unique network, station, component, and location entry in the *SEED* volume. The output is actually too long to display in Table 2 without line wrapping. The column entries are the network code, station name, location code, component name, on- and off-dates for the response period, station latitude, longitude, elevation, component azimuth, and dip, sample rate, name of response file, and name of the pole-zero file for this time period. This verbose output was introduced to permit scripted quality control of the metadata that describes the Saint Louis University seismic-station data channels by listing station information and plotting the history of channel responses. In doing so, we have been able to correct data-entry errors. An example of the use of the information is given on the following web page and its links: http://www.eas.slu.edu/eqc/eqc_netinfo/SLM/SLM.R/SLMindex.html (last accessed September 2013).

The other significant change made was to introduce comments into the pole-zero file. *SAC* versions have long permitted comments in this file by ignoring a line starting with an initial *. After our original modification in 2007 of the *Version 4.5* of *rdseed*, additional comments were added to be consistent with the pole-zero files provided by the USGS National Earthquake Information Center (NEIC) Continuous Wave Buffer (CWB) requests. The only differences between the annotation from the current IRIS and SLU versions of *rdseedv5.3* are now in the physical-unit annotation for some fields and the addition of the Site Name and Owner fields. Table 3 shows the annotated pole-zero file for the WCI BHZ trace. Although *rdseed* will create a displacement sensitivity pole-zero file giving response in counts/meter from a proper *SEED* volume, this is only apparent by reading the source code. The comments in the pole-zero file displayed in Table 2 explicitly show the input and output units. Some creators of pole-zero response files may use nanometers for the input units. There is nothing inherently wrong with this, because the pole-zero file just describes a filter and the user is responsible for tracking the units. Neither *SAC2000* nor *gsac* parse the comments to use the physical units of the response.

evalresp

This program provides tabular values of the instrument amplitude and phase response as a function of frequency from the *RESP* file created by *rdseed*. The IRIS source code was slightly modified to include files that are available to all target systems.

Table 2
Output of *rdseed*

```
IU WCI ** BHE      1999,316,00:00:00      2002,340,00:00:00 38.229000 −86.294000 506.0    0.0    90.0\      20 RESP.IU.WCI..BHE
SAC_PZs_IU_WCI_BHE__1999.316.00.00.00.0000_2002.340.24.60.60.99999
IU WCI ** BHN      1999,316,00:00:00      2002,340,00:00:00 38.229000 −86.294000 506.0    0.0    0.0\       20 RESP.IU.WCI..BHN
SAC_PZs_IU_WCI_BHN__1999.316.00.00.00.0000_2002.340.24.60.60.9999
IU WCI ** BHZ      1999,316,00:00:00      2002,340,00:00:00 38.229000 −86.294000   506.0    −90.0    0.0 \
20 RESP.IU.WCI..BHZ SAC_PZs_IU_WCI_BHZ__1999.316.00.00.00.0000_2002.340.24.60.60.99999
```

Table 3
Annotated Pole-Zero File

```
* **********************************
* NETWORK (KNETWK): IU
* STATION (KSTNM): WCI
* LOCATION (KHOLE):
* CHANNEL (KCMPNM): BHZ
* CREATED :2013-05-03T16:20:01
* START :1999-11-12T00:00:00
* END :2002-12-06T24:60:60
* DESCRIPTION :Wyandotte Cave, Indiana, USA
* LATITUDE (°) : 38.229000
* LONGITUDE (°) : −86.294000
* ELEVATION (m) : 506.0
* DEPTH (m) : 132.0
* DIP (°) : 0.0
* AZIMUTH (°) : 0.0
* SAMPLE RATE (Hz) : 20.0
* INPUT UNIT : M
* OUTPUT UNIT : COUNTS
* INSTTYPE : Streckeisen STS-1V/VBB Seismometer
* INSTGAIN : 2.398000e + 03 (m/s)
* COMMENT : N/A
* SENSITIVITY : 1.006000e + 09 (m/s)
* A0 : 3.948580e + 03
* Site Name : Wyandotte Cave, Indiana, USA
* Owner : (GSN) IRIS/USNSN, St. Louis University, USGS
* **********************************
ZEROS 3
        +0.000000e + 00   + 0.000000e + 00
        +0.000000e + 00   + 0.000000e + 00
        +0.000000e + 00   + 0.000000e + 00
POLES 4
        −1.234000e − 02   + 1.234000e − 02
        −1.234000e − 02   − 1.234000e − 02
        −3.918000e + 01   + 4.912000e + 01
        −3.918000e + 01   − 4.912000e + 01
CONSTANT      +3.972272e + 12
```

evalresp is included because of the senior author's preference in using it to create files for the removal of instrument response and, more importantly, because *evalresp* ensures that the physical units for input are meters, meters/s or meters/s^2 if displacement, velocity, or acceleration response, respectively, are requested from the command-line invocation. More importantly, *evalresp* includes the effects of the digital filters that are inherent in modern digitizers, thus giving the proper response near the Nyquist frequency. Validating the response metadata in the data-less *SEED* involves checking the responses in both the *RESP* and pole-zero representations because there are ways in which the two may be incompatible even though the underlying metadata is correct.

GSAC

The largest single program in the package is *GSAC*. This program implements many *SAC* commands and was written during the spring of 2004 before the *SAC2000* code was transitioned from Lawrence Livermore National Laboratory to IRIS members and affiliates (http://www.iris.edu/dms/newsletter/vol7/no1/sac-availability-for-the-iris-community/, last accessed September 2013). *GSAC* was written in C and was relatively easy to write because the command syntax and file format were patterned after those of *SAC*. *GSAC* is an open-source, stand-alone program that does not require any environment parameters to operate, although a graphics environment parameter can used to improve access by the visually impaired. *GSAC* is documented with tutorials, online help, and often in separate CPS web-based tutorials. *GSAC* graphics are based on the distributed CPS device-independent *CALPLOT* graphics package. *GSAC* only uses *SAC* files in the binary format in the local machine architecture, but conversion programs *saccvt*, *asctosac*, and *asctosac* are distributed with CPS to facilitate exchange with other machine architectures. *GSAC* was not designed to replace *SAC*, but to have a *SAC*-like tool that is more closely integrated with CPS. As such, *GSAC* usually includes only a subset of the many command options in *SAC*. Although *GSAC* does not support a macro language, much preprocessing can be completed within shell scripts utilizing the CPS program *saclhdr* to extract header parameters from the *SAC* file. To allow the convenient commenting of shell scripts that invoke *GSAC*, *GSAC* recognizes the SHELL comment delimiter number or pound sign.

Some of the *GSAC* commands behave slightly differently. For example, the *SAC rotate* command required that the horizontal instruments be exactly 90° apart and that the traces have the same start times and lengths. *GSAC* only requires that the horizontal orientations be different and adjusts for time differences. An early user suggested a command *rotate3* (*rot3*) that would rotate three independent components, not necessarily orthogonal, to form vertical, radial, and transverse components. The *rotate* and *rotate3* commands identify the common absolute time window for all traces and trim the seismograms to the common time window. A simple example of the use of the command is as follows (note GSAC> is the command prompt):

```
GSAC> r HRV*
HRVBHE.IU..sac HRVBHN.IU..sac HRVBHZ.IU..sac
GSAC> rot3 to gc
Executing rot3
Rotating to great circle to form R, T and Z in this order
GSAC> w
overwriting traces:
HRVBHR HRVBHT HRVBHZ
GSAC>
```

Those who rotate components for receiver function and source studies will appreciate the simplicity of this sequence because neither macros nor interactive picking are required. Because it was required to understand seismic sensor performance, the command was modified to allow rotation to a

UVW coordinate system for the evaluation of performance of individual internal sensors of some modern seismometers with a Galperin triaxial suspension, for example, STS-2's and Trilliums.

Another example of a difference/extension of *SAC* syntax is the setting of header time variables. Setting the origin time in *SAC* requires the specification of the day of year, which often means an inconvenient computation or lookup of the day of year from the calendar date. *GSAC* permits the use of calendar dates; the following two commands are equivalent in *GSAC*:

```
GSAC > ch o gmt 1982 123 13 37 10 103
GSAC > ch o cal 1982 5 3 13 17 10 103
```

GSAC graphics uses CPS graphics. Graphics output is either to an X window or to the device independent plot file. CPS graphics support seismic trace shading. By default, color is implemented for X11 graphics, but an environment parameter *PLOTXVIG* can be set to work in grayscale or in black-and-white to assist those with insensitivity to color. Basic seismogram plotting is also simplified; the *SAC2000 plot*, *plot*1, and plot2 commands are merged in one plot command using different command options:

```
# implement single trace plot
GSAC > p perplot 1
# implement multi trace plot1
GSAC > p
# implement overlay plot2
GSAC > p overlay on
```

Plotting record sections is streamlined, requiring no subsystem call:

```
GSAC > prs *.ZDS gcarc
```

with the result show in Figure 3. Options exist for trace shading of positive/negative amplitudes, reduced travel time, presentation, for example, refraction or reflection plots, and for the use of other header variables besides arc distance, for example, ray parameter stored in a header position and back azimuth for receiver function studies.

Writing a program like *GSAC* from scratch benefited from some design decisions that can help with long-term maintenance of the software. With the exception of graphics and filter routines, *GSAC* commands are implemented in a single, independent source file, for example, *gsac_sort.c*, that consists of two procedures: one to parse the command line, for example, *gsac_set_param_sort*, and the other to execute the command, for example, and *gsac_exec_sort*. This modularity promotes code maintenance (new commands can be easily prototyped) and permits the development of GUIs. Figure 4 shows the interactive screen resulting from the command:

```
GSAC > ppk regional perplot 3,
```

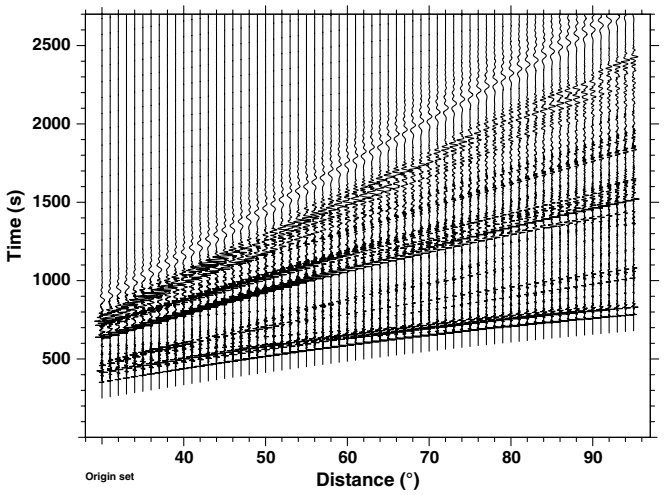
which is used for picking arrival times. The purpose of the buttons is to form the phase string in a manner that is not restricted by the four characters used in *HYPO71* (http://pubs.er.usgs.gov/publication/ofr75311, last accessed September 2013) but in a manner that permits easy parsing for event location and first motion plots, for example, *eP_C* or *ePKP*, if *ppk teleseism* is the command. Of course, *elocate*, the simple location program of CPS can parse this syntax.

For analyses such as receiver function studies, one is not interested in the phase quality or polarity, and the *SAC2000* or *GSAC* command:

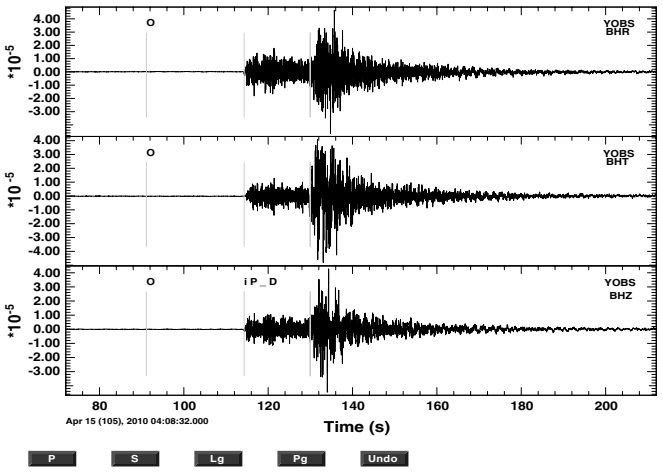
```
GSAC > ppk perplot 3 markall
```

suffices to define the *P*-wave arrival time pick by entering a *P* from the keyboard.

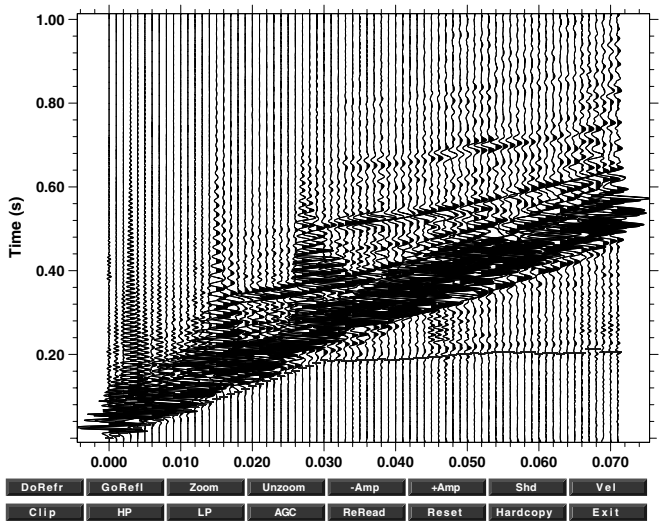
To permit rapid movement along the trace, the *xx* pair of the *SAC2000 ppk* command need not be given in a left-to-right order; *GSAC* internally properly orders the entries. Keyboard



▲ **Figure 3.** Record section plot of the vertical component Green's functions for a vertical dip-slip source at a depth of 200 km in the AK135 earth model (Kennett *et al.*, 1995).



▲ **Figure 4.** The *ppk regional perplot 3* screen. These are used to create a phase identification that can be easily parsed.



▲ **Figure 5.** The *refr* screen for a walkaway experiment using a transverse hammer source and transverse geophones. The *buttons* permit different displays of the waveforms.

commands adopted from Steve Malone's 1982 *ping* program for the PDP 11/70, "+" and "−" commands to expand and contract the times scale, "space" to move the trace along, and "*" and "/" to zoom in and contract the amplitudes, respectively, are implemented. These are very useful keyboard shortcuts for picking arrival times and for scanning long traces. In addition, a recent change of about six lines in the source code permits the use of a mouse wheel to change the trace amplitude scaling.

Figure 5 shows another GUI that was trivially built into *GSAC*. Given a record section, in this case a walk-away experiment using a horizontal hammer source and horizontal geophones, the command *refr* permits the determination of refraction times. The menu option *DoRefl* permits an analysis of reflection through the implementation of an interactive cursor in the shape of a hyperbola.

GSAC also has some unique, but useful, commands: the already-mentioned *rotate3* to quickly compute vertical, radial and transverse component seismograms without macros and user-defined cutting, *sgn*, and *whiten* for use with noise cross correlation, *boxcar*, *triangle*, and *trapezoid* for emulating simple earthquake source time functions when used with precomputed high-frequency Green's functions with short-duration sources, *shift* to lag synthetics for finite fault modeling, *outcsv* to output a waveform in a simple format that can be used with spreadsheet tools, and *mt* to combine Green's functions to compute a seismogram predicted by a particular moment tensor.

Some additions, such as the *rotate3* command, were added to speed processing of data sets. To quickly perform data-quality review of large numbers of waveforms for source inversion or receiver function analysis, a quality control option was added to the *plotpk* (*ppk*) command:

```
GSAC > ppk q
```

which permits the user to indicate that a trace is to be used for inversion with just a simple mouse click (indicated on the graphics display by a red plus sign). Internally, the code sets a value in the *SAC* header integer value position 20, IHDR20. When the user is finished looking at the traces, a *writeheader* (*wh*) saves the header values in each *SAC* file. Then a set of commands, usually in a shell script, using the CPS 3.30 program *saclhdr*, is run to examine the header values. If IHDR20 is set, the waveform is copied to the processing area. The following shell script shows how the QC is invoked and then how the selected waveforms are moved to a processing area for source inversion:

```
#/bin/bash
gsac ≪ EOF
#####
#   perform QC on all traces using the same
#   frequency band as for final source inversion
#####
fileid name
markt on
xlim a -10 a 180
r *
sort up dist
rtr
hp c 0.02 n 3
lp c 0.10 n 3
qdp 10
ppk q relative perplot 3
wh
q
EOF
#####
#   now move selected traces to processing area
#####
for i in *[ZRT]
do
IHDR20='saclhdr -IHDR20 $i'
ANS='echo $IHDR20 | awk '{if($1 < 1)print "NO";else if
($1 > 1)print "NO" ; else print "YES" }' '
if [ $ANS = "YES" ]
then
echo $i $IHDR20 $ANS
cp $i ../../DAT.REG
fi
done
```

This shell script also illustrates use of comments for documentation, the method of providing input to a program from within the shell, and the use of the program *saclhdr* to pass trace header values to shell variables. This script will run on all of the targeted systems: *UNIX*, *LINUX*, *MacOS-X*, and *Cygwin* (Windows). The shell script also indicates the value of a uniform approach to naming the *SAC* data files. In this case, the filenames end in Z, R, or T.

Tutorials

Most recently, some effort has been expended to develop tutorials targeted at beginning users, such as graduate students in seismology. The tutorials are available on the web and include sample data and processing scripts. Initially, the tutorials arose as part of the software validation process. Other tutorials were developed in response to user requests. The tutorials illustrate the use of the codes, and hopefully, the value of well-documented adaptable processing scripts. Some topics currently discussed are

- Determination of receiver functions
- Inversion of dispersion and receiver functions for structure
- Determination of surface-wave group and phase velocity
- Unpacking *SEED* volumes, deconvolving the instrument response, and rotating to vertical, radial, and transverse components
- Cross correlation of ground noise for interstation Green's functions
- Instrument response characterization by pole-zero and *RESP* files
- Simple programs to read and write *SAC* files
- Moment tensor inversion

CONCLUSION

The package and tutorials are available from the web page: http://www.eas.slu.edu/eqc/eqccps.html (last accessed September 2013). The availability of free operating systems and compilers, and the open-access distribution of seismic data have enabled the broad use of tools such as CPS. Open and free distribution often leads to user questions that result in bug fixes that produce a more robust package for seismic analysis. Of course, no package is perfect, and users are encouraged to explore and to improve the code. Contributed codes will be properly acknowledged, commented, and modified to fit within the design of the package.

This brief introduction to the package has focused on its ease of use, examples of documented scripting for processing large data sets, and the promotion of an open, supported, documented environment in support of research. This package has become fun to use as the individual components are combined to address interesting problems.

ACKNOWLEDGMENTS

The software development benefited from many years of external support by the USGS, AFOSR, NSF, and the USNRC, to mention a few. The significant contribution of students must be mentioned: Chien-Ying Wang wrote the stable surface-wave code and David Russell built on Wang's work to create the surface-wave-dispersion inversion package. Recently, the incorporation of comments in the pole-zero file was benefited by interaction of David Ketchum at the USGS NEIC. Finally, Charles Ammon has provided source code and a vision for processing earthquake data that has guided this code development.

REFERENCES

Červený, V., and I. Psenčik (1981). *EIS81, a 2D Seismic Ray Package*, Charles University, Prague, Czechoslovakia.

Goldstein, P., D. Dodge, M. Firpo, and L. Minner (2003). SAC2000: Signal processing and analysis tools for seismologists and engineers, Invited contribution to *The IASPEI International Handbook of Earthquake and Engineering Seismology*, W. H. K. Lee, H. Kanamori, P. C. Jennings, and C. Kisslinger (Editors), Academic Press, London, UK.

Herrmann, R. B. (1978). *Computer Programs in Earthquake Seismology, Volume 1: General Programs*, R. B. Herrmann (Editor), Department of Earth and Atmospheric Sciences, Saint Louis University, November 1978 (NTIS PB\ 292\ 462).

Kennett, B. L. N., E. R. Engdahl, and R. Buland (1995). Constraints on seismic velocities in the Earth from travel times, *Geophys. J. Int.* **122,** 108–124.

Kernighan, B. W., and P. J. Plauger (1974). *The Elements of Programming Style*, McGraw-Hill, New York, ISBN 0-07-034199-0.

Robert B. Herrmann
Department of Earth and Atmospheric Sciences
Saint Louis University
O'Neil Hall, Room 203
3642 Lindell Boulevard
St. Louis, Missouri 63108 U.S.A.
rbh@eas.slu.edu