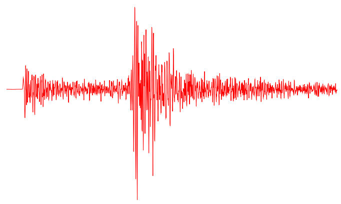
COMPUTER PROGRAMS IN SEISMOLOGY



GSAC

Generic Seismic Application Coding

Robert B. Herrmann

Professor of Geophysics
Department of Earth and
Atmospheric Sciences
Saint Louis University

Charles. J. Ammon

Associate Professor
Department of Geosciences

Penn State University

Version 3.30
2004

Contents

PREFACE

SAC, the Seismic Analysis Code, was created by researchers at the Lawrence Livermore National Laboratory in the early 1980's. Initially distributed as a FORTRAN program with low level routines in C, SAC became widely used by the earthquake research community. The current SAC2000 is written in C, and is distributed as an execuable binary form for several common platforms.

SAC and SAC2000 actually permit more than simple manipulation of seismic traces. The macro script language and signal processing features make it a processing tool that has only recently been supplanted in capabilites by commercial packages such at MATLAB, MATHCAD and Mathematica.

The other contribution of SAC was the definition of a seismic trace file. The concept of this file is similar to that use in seismic exploration for which the trace consists of a trace header and the binary trace itself. Many programs have been written to use the SAC trace files. This was encouraged in the original SAC distribution by ready access to a library of input/output routines for the FORTRAN and C languages.

Unfortunately SAC/SAC2000 has become dated because of its monilithic structure, the previously closed source distribution, and advances in computer platforms. The signal processing capabilities have been supplanted by MATLAB and Mathematica, the support of 24 bit color displays under X11 is lacking, and the assumptions about the underlying X11 support engine have become dated.

With this in mind, we decided to write a program to permit necessary seismic trace manipulation from scratch. Starting, March 27, 2004, we created a functional GSAC by June 1, 2004 without much effort. GSAC takes its name from the free gcc and g77 compilers used, with the corresponding commitment to open sources.

SAC is a group effort to provide documented tools for manipulating seismic traces which happen to be stored in a SAC file format. GSAC thus emphasizes waveform processing rather than a specific implementation. Thus GSAC is meant to be all inclusive which means that the concept will encompass different underlying operating systems (UNIX, LINUX, MacOS-X, Windows), different hardware architectures (IEEE big-endian and little-endian), and different development environments (gcc/g77, MATLAB, Maple, etc).

The design goals of the GSAC project are simple:

- Platform independent seismic waveform calculator core routines, with front ends that permit command line operation, especially within shell scripts,

- Complete documentation of all internal signal processing routines,
- Tutorials addressing the needs of beginner, intermediate and advanced users,
- Platform independence of wavefrom data, and
- A design that permits GUI operation.

CHAPTER 1
INTRODUCTION TO GSAC

1. Introduction

SAC and SAC2000 implement many commands. The mechanism that relates names on the SAC command line to the actual internal function is accomplished through the files *clspe*, *clsss* and *clstd* in the directory *${SACAUX}*. The first step in the design of *GSAC* was to tablulate all commands used in SAC and SAC2000 and then to prioritize these commands for implementation into *GSAC*. The criteria for this choice was that *GSAC* function as a seismic waveform manipulation tool for basis analysis of earthquake waveforms. Any complicated analysis of a waveform or group of waveforms is best left to stand-alone programs. This tabulation is given in §1.3.

1.2 GSAC Design

To implement the goals of platform independence in both the operation of the program and the interactive graphics, GSAC is developed using the free *gcc/g77* compilers. The code is written in C in a manner that is hopefully understandable and supportable. The current version of *GSAC* consists of about 14000 lines of code, and has the command parsing and online help build in. Thus there is no need for the SAC *aux* directory or for the f2${SACAUX} environment parameter to be set. As a comparison, 1990 FORTRAN version of SAC consisted of over 110,000 line of source code in C and FORTRAN to support the SAC operations and underlying graphics. *gsac* uses the *CALPLOT* package of Computer Programs in Seismology for its graphics support. *gsac* is not written in an object-oriented manner, which may make the implementation of certain commands, such as `mulf`, slightly more difficult. On the other hand, it is trivial to prototype and implement new commands.

In the process of writing *gsac*, certain limitations of SAC/SAC2000 were noted that should guide the evolution of not only *gsac* but also SAC2000.

- The SAC header must define the binary data type - e.g., IEEE big/little endian.IP • Physical UNITS matter. The IRIS evalresp outputs response tables in in units of M M/S or M/S/S. POLEZERO FILES MUST HAVE A FIELD FOR INPUT_UNIT and OUTPUT_UNIT.
- Keep GSAC lean and mean - everyone can get GMT and MATLAB. However, Provide tools and examples for use of SAC traces with GMT and MATLAB

- SAC2000 has an option of loading an external object from a shared library. This is too complicated for casual user. Perhaps just spawn off an external process from with in *gsac*. The current libsac.a has lot's of routines that might conflict with user use of library. Do not share the library of IO routines used by *gsac* with user programs. Provide a separate set of interfaces.
- GSAC MUST work with majority of existing shell scripts (perhaps in place of macros)
- Picks - define more options to ppk, also have external routines to handle information from external databases. PPK need the ability to enter Phase names, such as Lg Pn Sn etc
- Do not permit PRINT. All output is either a SAC file or one of a few graphics files. However a GUI wrapper can always print a current window. But that is a function of the GUI and not GSAC

1.3 GSAC and SAC commands

SAC	SAC2000	GSAC	ESSENTIAL	USEFUL	SPECIAL	NOT USEFUL	DESCRIPTION
1.01	1.01	**YES**	QUIT				Terminate program
1.02	1.02				PRODUCTION		Controls Abort
1.03	1.03					NEWS	Ever implemented?
1.04	1.04	**YES**	HELP				Essential aid
1.05	1.05				REPORT		Current status
1.06	1.06	built in	SYSTEMCOMMAND				Execute shell command
1.07	1.07				INICM		Reinitializes
1.08	1.08	**YES**	FUNCGEN				Generate simple trace
1.09	1.09			MESSAGE			Write message to term
	1.1					PRINTHELP	Print help
1.11	1.11			COMCOR			command correction option
1.12	1.12			SYNTAX			concise help
1.13	1.13	**YES**		PAUSE			Pause/message to term
1.14	1.14	**YES**	ECHO				echoing of input and output
1.15	1.15				EVALUATE		In line calculator
1.16	1.16				SETBB		Set blackboard
1.17	1.17				GETBB		Get blackboard

1.18	1.18				READBBF		Reads black-board file
1.19	1.19				WRITEBBF		Writes black-board var
1.2	1.2				MACRO		Executes macro
1.21	1.21				SET-MACRO		Define directo-ries for search
1.22	1.22					INSTALL-MACRO	Install in global direc
1.23	1.23				UNSETBB		Unset BB vari-able
1.24	1.24				TRAN-SCRIPT		Save history of command
1.25	1.25				TRACE		Track BB vari-ables
1.26	1.26					LOAD	Load external shared object routines
	1.27	**YES**	CD				Change work-ing directory
	1.28				ABOUT		Version number This is impor-tant since LLNL uses this number to decide of big or little endian (replace with VERSION)
2.01	2.01	**YES**	READ				Read binary SAC file
2.02	2.02		READERR				Controls errors on READ
2.03	2.03	**YES**	WRITE				Write binary SAC file
2.04	2.04		CONVERT				Converts data file format
2.05	2.05	**YES**	CUT				Trace cut
2.06	2.06	**YES**	CUTERR				how to handle cut errors
2.07	2.07	**YES**	LISTHDR				list header
2.08	2.08	**YES**	CHNHDR				change header
2.09	2.09	**YES**	READHDR				read header
2.1	2.1	**YES**	WRITEHDR				write header
2.11	2.11	**YES**	SYNCHRO-NIZE				Synchronize ref times

2.12	2.12				WILD		It this necessary? Everythin now had BASH shell, even WIN32
2.13	2.13				READALPHA		Read ASCII
2.13	2.13				READTABLE		Read ASCII
2.14	2.14		COPYHDR				Copies variables from one file in memory to all
2.15	2.15				DATAGEN		Test seismograms
2.16	2.16					READSDD	Read SDD
2.17	2.17					WRITESDD	
2.18	2.18					READCSS	Read CSS
2.19	2.19				HEADERWINDOW		
	2.2				CUTIM		Cuts files in memory (IO is fast enough now so why do this)
	2.21					WRITETABLE	
	2.23					PICKPREFS	
	2.24					WRITEGSE	Write CSE
	2.25					WRITECSS	Write CSS
	2.27			DELETECHANNEL			Clean memory traces
	2.28					PICKAUTHOR	Pick authority
	2.29				PICKPHASE		Read pickphase info
	2.3	**YES**	SORT				Sort by header value
	2.31					READSUDS	Read SUDS
	2.32					READGSE	Read GSE
3.01	3.01	**YES**	BEGINDEVICES				Start graphics (note drop SUN, KEEP TEK, SGF, add PS, EPS)
3.02	3.02		ENDDEVICES				End graphics
3.03	3.03		ERASE				Erase graphics
3.04	3.04		VSPACE				Control plot
3.05	3.05		HCD				???

3.02	3.05		SGF				Control SGF device
4.01	4.01	**YES**	XLIN				Control plot
4.02	4.02	**YES**	XLOG				Control plot
4.03	4.03	**YES**	YLIN				Control plot
4.04	4.04	**YES**	YLOG				Control plot
4.05	4.05	**YES**	LINLIN				Control plot
4.06	4.06	**YES**	LINLOG				Control plot
4.07	4.07	**YES**. Only a place holder. Not imple-mented.	LOGLIN				Control plot
4.08	4.08	**YES**. Only a place holder. Not imple-mented.	LOGLOG				Control plot
4.09	4.09		XFULL				Control plot
4.1	4.1		YFULL				Control plot
4.13	4.13		XVPORT				Control plot
4.13	4.13		XWIND				Control plot
4.14	4.14		YVPORT				Control plot
4.14	4.14		YWIND				Control plot
4.15	4.15		XDIV				Control plot
4.16	4.16		YDIV				Control plot
4.17	4.17	**YES**	GRID				Control plot
4.18	4.18		BORDER				Plot border
4.19	4.19		AXES				Control axes
4.19	4.19		AXIS				Control plot
4.2	4.2		TICKS				Control plot
4.21	4.21		LOGLAB				Control plot
4.22	4.22			XFUDGE			Control plot
4.23	4.23			YFUDGE			Control plot
4.24	4.24	YES	TITLE				Control plot
4.25	4.25		XLABEL			Control plot	
4.26	4.26		YLABEL				Control plot
4.27	4.27	**YES**			QDP		Defautl should be OFF
4.28	4.28		FLOOR				Control log plots
4.29	4.29				WAIT		Plot wait
4.3	4.3		LINE				Plot style
4.32	4.32		SYMBOL				Control plot
4.33	4.33				BEGIN-FRAME		Turns off auto new frame

4.34	4.34				ENDFRAME		Resume new frame between plots
4.35	4.35		GTEXT				Control plot
4.36	4.36	**YES**	COLOR				Control color selection
4.37	4.37	**YES**	XGRID				Control plot
4.39	4.38	**YES**	YGRID				Control plot
4.3	4.39		PLABEL				Control plot
4.4	4.4		TSIZE				Control plot
4.41	4.41		WINDOW				Control plot
4.42	4.42			BEGINWIN-DOW			Start new window (easy to plot in a new window but return to old may be hard)
4.45	4.45		NULL				Control plot
4.46	4.46		WIDTH				Control plot
4.47					LOAD-CTABLE		new color table for 2D
5.01	5.01	*Yes*				PLOT	Direct to printer
5.02	5.02	**YES**	PLOT1				Plot single, group trace
5.03	5.03	**YES**	PLOT2	*Yes*			superimpose trace
5.04	5.04	**YES**	PLOTPK				Interactive PICK
5.05	5.05					PLOTC	interactive annotation of plots
5.06	5.06	**YES 050323**	FILEID				Trace annotation
5.07	5.07		PICKS				Control plot
5.08	5.08				PLOTPM		Plot particle motion
5.09	5.09				SETDEVICE		Reset default device
5.1	5.1	**YES**	XLIM				Control plot
5.11	5.11	**YES**	YLIM				Control plot
5.12	5.12				PLOTXY		GMT does better
5.14	5.14				PLOTDY		
5.15	5.15					PLOTALPHA	use READALPHA then plot
	5.16				FILENUM-BER		trace annotation

	5.17					PRINT	Print recent SGF
6.01	6.01				DFT		True name for FFT
6.01	6.01	**YES**	FFT				Discrete FT
6.02	6.02				IDFT		True name for IFFT
6.02	6.02		IFFT				Inverse Discrete FT
6.03	6.03	**YES**	PLOTSP				Plot spectra
6.04	6.04	**YES**	WRITESP				Write spectra file
6.05	6.05		READSP				Read spectra file
6.06	6.06	**YES**	LOWPASS				IIR Lowpass filter
6.07	6.07	**YES**	HIGHPASS				IIR Highpass filter
6.08	6.08	**YES**	BANDPASS				IIR Bandpass filter
6.09	6.09	**YES**	BANDREJ				IIR Band reject filt
6.1	6.1			WIENER			Wiener filter
6.11	6.11		FIR				Apply FIR filter
6.12	6.12	(see taper)	HANNING				Window trace
6.13	6.13				UNWRAP		Tribolet - non-trivial
6.14	6.14	*YES*	CORRE-LATE				cross-,auto-cor-relation
6.15	6.15					KHRONHITE	This is older than FOR-TRAN
6.16	6.16				BENIOFF		Benioff filter
6.17	6.17		DIVOMEGA				Integrate Freq domain on spectra file
6.18	6.18		MULOMEGA				Differentiate freq domain on spectra file
6.19	6.19	**YES**	HILBERT				TD Hilbert transform
6.2	6.2	**YES**	ENVELOPE				Envelope using hilbert
6.21	6.21					FILTERDE-SIGN	design filter
6.22	6.22				AM		

6.22	6.22				KEEPAM		Keep amplitude as 1-D so ordinary SAC commands work
	6.23	YES	CONVOLVE				Convolve traces
7.01	7.01	**YES**	ADD				ADD constant
7.02	7.02	**YES**	SUB				Subtract constant
7.03	7.03	**YES**	MUL				Multiply const
7.04	7.04	**YES**	DIV				Divide by const
7.05	7.05	**YES**	SQR				Square trace
7.06	7.06	**YES**	SQRT				Square root trace
7.07	7.07	**YES**	INT				Integrate
7.08	7.08	**YES**	ABS				ABS trace
7.09	7.09	**YES**		LOG			Log trace
7.1	7.1	**YES**		LOG10	**YES**		Log trace
7.11	7.11	**YES**		EXP			Exp trace
7.12	7.12	**YES**		EXP10	**YES**		Exp trace
7.13	7.13	**YES**	DIF				Differentiate
8.01	8.01	**YES**		MERGE			Concatenate files
8.02	8.02	**YES**	ADDF				ADD file
8.03	8.03	**YES**	SUBF				Subtract file
8.04	8.04	**YES**	MULF				Multiply files
8.05	8.05	**YES**	DIVF				Divide file
8.06	8.06		BINOPERR				Controls errors that can occur during binary file operations.
8.06	8.06				BOEC		
9.01	9.01					OHPF	Open hypo71 pick file
9.02	9.02					CHPF	close hypo71 pick file
9.03	9.03					WHPF	write hypo71 pick file
9.04	9.04					OAPF	Open alpha pick file
9.04	9.04				OCIPF		
9.05	9.05					CAPF	Close Alpha pick file
9.05	9.05				CCIPF		
9.01	9.06				APK		Auto pick
10.01	10.01					RQ	Remove Q effect from data
10.02	10.02				RIR		
10.03	10.03				RGLITCHES		Remove Glitch

10.04	10.04	**YES**	RTREND				
10.05	10.05	**YES**	RMEAN				Remove mean
10.06	10.06	*YES*	TAPER				Taper trace
10.07	10.07	**YES**	**ROTATE**				Rotate horizontals (FIX so it works if horizontals not orthogonal)
10.08	10.08	YES	**INTERPOLATE**				Resample
10.09	10.09				QUANTIZE		Quantize - good for signal analysis
10.10	10.10				STRETCH		Resample trace
10.11	10.11	**YES**	REVERSE				Reverse time series
10.12	10.12	**YES**	SMOOTH				Smooth trace
10.13	10.13	Interpolate	DECIMATE				Down sample. *Note that there is no FIR filter applied. There is no constraint on the decimation factor.*
10.14						LINEFIT	straight line to data
11.01	11.01					SPE	Spectral estimation subsystem
11.02	11.02					COR	
11.03	11.03					PLOTCOR	Plot correlation
11.04	11.04	YES	**READ**				
11.05	11.05					PLOTPE	plot prediction error
11.05	11.05					PPE	
11.06	11.06					PDS	Power Spectra Density Spectra
11.07	11.07					MLM	Max likelihood spectra
11.08	11.08					MEM	Max entropy spectum
11.09	11.09					PLOTSPE	Plot spectral estimate
11.1	11.1					SPEID	???
11.11	11.11					WRITECOR	
11.12	11.12					WRITESPE	
11.13	11.13					READCOR	

11.14	11.14					QUITSUB	Not needed if no SSS/SPE subsystem support
	11.15			PREWHITEN			Flattens the spectrum
12.01	12.01					SSS	Some of these are useful
12.02	12.02				ZEROSTACK		
12.03	12.03				GLOBALSTACK		
12.04	12.04				ADDSTACK		
12.05	12.05				DELETESTACK		
12.06	12.06				CHANGESTACK		
12.07	12.07				LISTSTACK		
12.08	12.08				INCREMENTSTACK		
12.09	12.09				SUMSTACK		
12.1	12.1				TIMEWINDOW		
12.11	12.11				PLOTSTACK		
12.12	12.12				DELTACHECK		
12.13	12.13				DISTANCEAXIS		
12.14	12.14				TIMEAXIS		
12.15	12.15				DISTANCEWINDOW		
12.16	12.16				VELOCITYROSETTE		
12.17	12.17	**YES**			PLOTRECORDSECTION		
12.18	12.18				VELOCITYMODEL		
12.19	12.19				WRITESTACK		
12.2	12.2				QUITSUB		
12.21	12.21				TRAVELTIME		
12.22	12.22				PHASE		
13.01	13.01					MTW	Measurement time window
13.02	13.02				MARKVALUE		Searches values
13.03	13.03		**YES**	MARKTIMES			Set time in header for later XLIM/CUT
13.04	13.04				MARKPTP		Measure PP amp

13.05	13.05				RMS		Compute rms in window
14.01	14.01	**YES**	TRANSFER				Apply/remove filter Perhaps we should get rid of the many predesigned filters
	14.02				PREWHITEN		
	14.02				PW		
	14.02	**YES**			WHITEN		
15.01	15.01				IF		
15.02	15.02				ELSEIF		
15.03	15.03				ELSE		
15.04	15.04				ENDIF		
15.05	15.05				DO		
15.06	15.06				WHILE		
15.07							
15.07							
T}				ENDDO			
15.08	15.08				BREAK		
16.01	16.01				WRITENN		
17.01	17.01				SPECTRO-GRAM		spectrogram
17.02	17.02				GRAYSCALE		grayscale images
17.03	17.03				CONTOUR		contour plots of data
	17.04				ZLEVELS		levels for contour
	17.05				ZLINES		line style for contour
	17.06				ZTICKS		ticks for contour
	17.07				ZLABELS		labels for contour
17.08	17.08				ZCOLORS		color for contour
	17.09				IMAGE		color image plots
	17.1				SONOGRAM		spectrogram
18.01	18.01				ARRAYMAP		Plots array loc
18.02	18.02				BBFK		F-K analysis
18.03	18.03			BEAMFORM			From beam
18.04	18.04	YES	MAP			GMTMAP	Generate GMT map
	19.01					3C	Launch MATLAB

	19.01					MAT3C	Launch MATLAB
	19.02					MAT	Matlab
	19.03					SETMAT	Set MATLAB directory
	19.04					CLOSEMAT	Close MATLAB
	19.05					MATPRS	???
	19.05				RECORDSECTION		
	19.06					MATDEPMEC	MATLAB for Depth determination
	20.01					CODA	???
	20.01					MCODA	???
02.19						CURRENTDATASET	
02.20						RENAME	
02.21							DELETE
02.22						COPY	
02.23						LISTDATASETS	
		NEW	HOLD				Permit plot overlay
		NEW	PCTL				Set axis control for PLOT1 and PLOTPK
		NEW	FILT				Filter traces - similar to TRANSFER
		NEW	AGC				AGC trace filter
		NEW	STACK				Stack traces in absolute or relative time
		NEW	SGN				Apply one big sign operator to trace
		NEW	WHITEN				Whiten trace by adjusting Fourier amplitude spectrum
		NEW	OUTCSV				output traces in comma separated variable format for use with srpeadsheet

		NEW	BACK-GROUND				Define back-ground color for plots
		NEW	BOXCAR				Convolve trace with a boxcar
		NEW	TRIANGLE				Convolve trace with a triangle
		NEW	TRAPEZOID				Convolve trace with a trapezoid
		NEW	VERSION				Give gsac ver-sion number
		NEW	MT				Compute syn-thetic from pre-computed Greens func-tions
		NEW	SHIFT				Shift signal a fixed numebr of seconds chang-ing pick times but not origin time
		NEW	ROTATE3				Rotate three components to make ZRT etc
		NEW	HISTORY				List all com-mands used in the current ses-sion.
		NEW	REFRAC-TION				Enter refrac-tion/reflection analysis mod-ule.
		NEW	RICKER				Convolve with Ricker wavelet

CHAPTER 2
INTRODUCTION TO GSAC

1. Introduction

This chapter is an introduction to seismic signal procession using GSAC. Since GSAC implements many concepts used in SAC2000, this will also act as a needed tutorial on using SAC2000. However, the GSAC syntax may significantly from that of SAC2000. So be careful.

2. The SAC File

The unifying feature of many seismological applications is the use of waveform data in the SAC format. The following sections introduce the data format and attempt to clarify some confusion with respec to timing.

The SAC data file is illustrated in the next figure. The data file cand be viewed as a linear stream of numbers separated into an initial header section followed by the trace data. The header consists of 70 4-byte real numbers, 40 4-byte integers and 24 8-byte characters. The binary data that follows can be interpreted as a single set of y-values, a set of x-y pairs, or a set of x-y-z pairs.

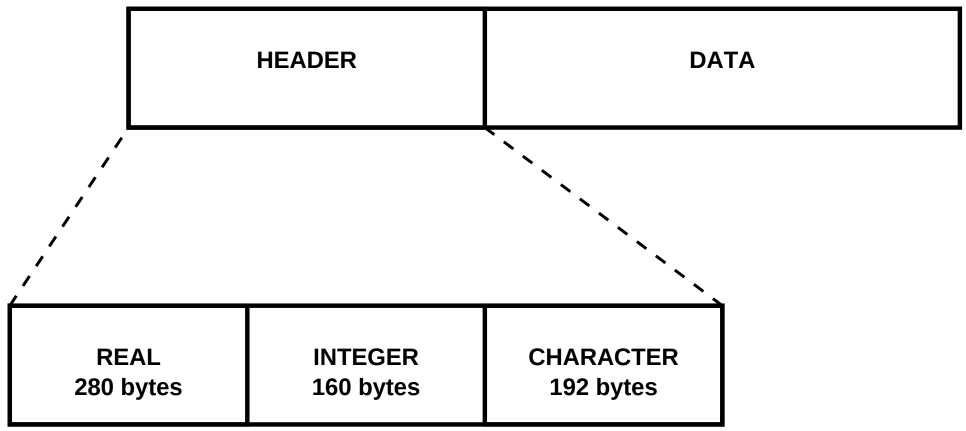
`Currently GSAC only considers the set of equally spaced y-values`.

2.1 Headers

2.2 Time

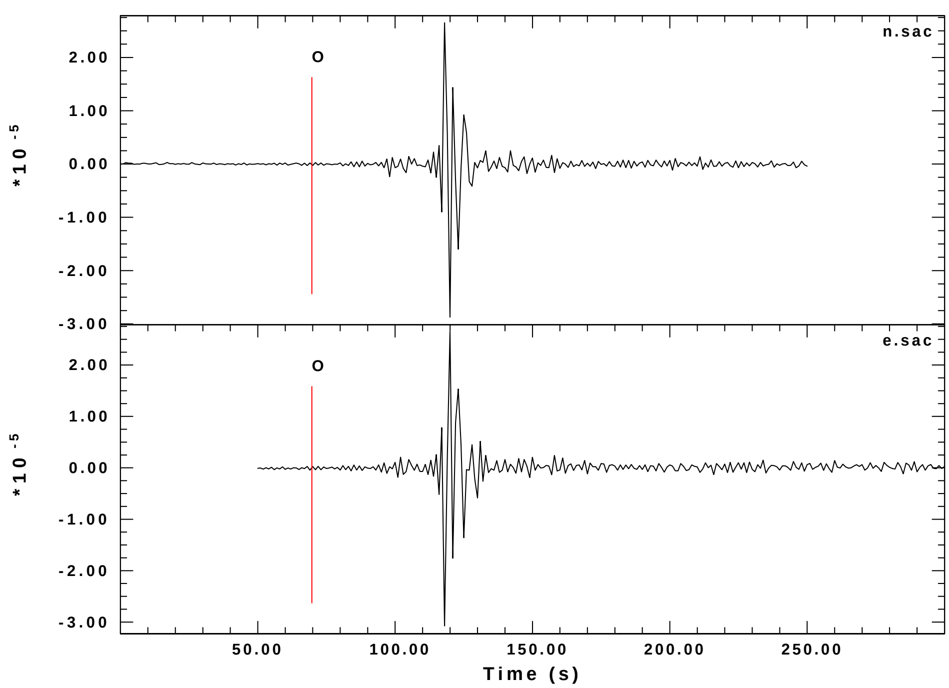
When the SAC file describes a time series, certain header values can be used to describe characteristics of the seismic source or or the seismic arrivals. Rather than defining each of this as an 8-byte floating point number, SAC chooses to define a reference time and offsets with respect to the reference time. This approach uses fewer bytes than the other alternative.

If the trace is defined as a set of y-values, the trace in memory is just a sequence of numbers without any time tag. However, the first and last point of the trace do possess an



SAC data file consists of a linear array of numbers, the first 632 byte of which form a header, which is then followed by the binary dating. The header is separated into three sections.

absolute time stamp. SAC defines a reference time through the integer headers *NZYEAR*, *NZJDAY*, *NZHOUR*, *NZMIN*, *NZSEC* and *NZMSEC*. The offset of the first point on the trace with respect to this reference time is given by the header value *B*. *E*, *A*, *O*, *T0* ... *T9* are other header values that relate to this reference time. To illustrate this consider the following figure that shows the north-south and east-west components at the station BLO for the June 18, 2002 southwestern Indiana earthquakes:



Plot of BLO horizontal components in absolute time.

These traces are plotted in absolute time. Note that the start and end times of each trace differ. If we use the following GSAC commands we can look at the header values in more detail:

```
GSAC> r n.sac e.sac
n.sac e.sac
GSAC> lh b o kzdate kztime columns 1
n.sac (0):
            B             50.32334
       KZDATE   Jun 18 (169), 2002
       KZTIME         17:35:16.000
            O                  120
e.sac (1):
            B                    0
       KZDATE   Jun 18 (169), 2002
       KZTIME         17:36:56.323
            O             19.67686
GSAC>
```

We see that the reference time denoted by the combination of *KZDATE* and *KZTIME* differs by 100 seconds. The *B* header value, for the time of the first sample, and the *)* header value, for the origin time, also differ. Since the origin time is the same, the following must be true that the the *KZDATE* + *KZTIME* + *O* be the same, which we can see since

```
      n.sac               e.sac
  17:35:16.000        17:36:56.323
      2:00.000             :19.677
  ------------        ------------
  17:37:16:000        17:37:16.000
```

There is no restriction on the offsets. They can be negative or positive. If necessary, one can synchronize the traces with the following command:

```
GSAC> synchronize
GSAC> lh
n.sac (0):
            B                    0
       KZDATE   Jun 18 (169), 2002
       KZTIME         17:36:06.323
            O             69.67666
e.sac (1):
            B             49.99966
       KZDATE   Jun 18 (169), 2002
       KZTIME         17:36:06.323
            O             69.67651
GSAC>
```

2.3 Machine Data Format

The SAC trace file consists of a header followed by the data. The header has three parts: floats (or reals), integers and characters. The first two sections are in binary while the characters are in ASCII. The trace data is also in binary format. The binary format compacts data stream but also makes the trace specific to the computer architecture that created the SAC trace file. Although there could be a unique data format for each processor, standards exist in modern computers because of the acceptance of just a few architectures. Today one would use an IEEE floating point representation on INTEL, SPARC and PowerPC platforms, but would find that the byte order differs between the INTEL processor used for MS Windows or LINUX and the SPARC processor, running Solaris, and the PowerPC running MacOS-X. To illustrate this we apply the UNIX/LINUX command *od -a* on a seismic trace file.

If the file was created on an INTEL (or AMD) based PC, we would see the following in the header:

```
0000000   M   L   L   = nul nul nul nul nul nul  sp   A nul   d   @   F
0000020 nul   d   @   F   M   L   L   B   M   L   L   B nul   d   @   F
0000040 nul   d   @   F nul   d   @   F nul   d   @   F nul   d   @   F
*
0000340 nul nul  sp   ; nul   d   @   F nul   d   @   F nul   d   @   F
0000360 nul   d   @   F nul   d   @   F nul   d   @   F nul   d   @   F
*
0000420 nul   d   @   F nul   d   @   F   2 bel nul nul soh nul nul nul
0000440 nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul
0000460 ack nul nul nul   G   O del del   G   O del del nul dle nul nul
0000500   G   O del del   G   O del del   G   O del del   G   O del del
0000520   G   O del del soh nul nul nul   G   O del del   G   O del del
0000540   G   O del del   G   O del del   G   O del del   G   O del del
*
0000640   G   O del del soh nul nul nul nul nul nul nul soh nul nul nul
0000660 soh nul nul nul nul nul nul nul   -   1   2   3   4   5  sp  sp
0000700   -   1   2   3   4   5  sp  sp   -   1   2   3   4   5  sp  sp
*
0001160   -   1   2   3   4   5  sp  sp nul nul nul nul nul nul nul nul
0001200 nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul
*
0021160 nul nul nul nul nul nul nul nul nul nul  sp   A nul nul nul nul
0021200 nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul
*
0041160 nul nul nul nul nul nul nul nul
0041170
```

and the following if it was created on a SPARC or Mac:

```
0000000   =   L   L   M nul nul nul nul   A  sp nul nul   F   @   d nul
0000020   F   @   d nul   B   L   L   M   B   L   L   M   F   @   d nul
0000040   F   @   d nul   F   @   d nul   F   @   d nul   F   @   d nul
*
0000340   ;  sp nul nul   F   @   d nul   F   @   d nul   F   @   d nul
0000360   F   @   d nul   F   @   d nul   F   @   d nul   F   @   d nul
*
0000420   F   @   d nul   F   @   d nul nul nul bel   2 nul nul nul soh
0000440 nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul
0000460 nul nul nul ack del del   O   G del del   O   G nul nul dle nul
0000500 del del   O   G del del   O   G del del   O   G del del   O   G
0000520 del del   O   G nul nul nul soh del del   O   G del del   O   G
0000540 del del   O   G del del   O   G del del   O   G del del   O   G
```

```
*
0000640 del del   O   G nul nul nul soh nul nul nul nul nul nul nul soh
0000660 nul nul nul soh nul nul nul nul   -   1   2   3   4   5  sp  sp
0000700   -   1   2   3   4   5  sp  sp   -   1   2   3   4   5  sp  sp
*
0001160   -   1   2   3   4   5  sp  sp nul nul nul nul nul nul nul nul
0001200 nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul
*
0021160 nul nul nul nul nul nul nul nul   A  sp nul nul nul nul nul nul
0021200 nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul
*
0041160 nul nul nul nul nul nul nul nul
0041170
```

The point to note is that the 4-byte patterns of the reals and integers are reversed in place, e.g., 'M L L =' becomes '= L L M' . The character strings '- 1 2 3 4 5 sp sp', where 'sp' is a space, are not changed.

Although it is possible to make GSAC automatically handle the conversion, there are conceptual problems in how to handle the WRITEHEADER command since this over-writes only part of the trace file - the header and not the data. There is also the question of whether to convert everything into the native format or whetheer to preserve the original format always.

To keep GSAC simple, we assume that the SAC file is ALWAYS in the format for the local architecture. This means that it is necessary to convert the file format, if necessary. To do this, we provide the utility `saccvt` which does this. This program functions by looking for the ubiquitous number -12345 in the header. To undeerstand the function of this program, just enter the command `saccvt -h`. We do the following whenever we import data from another architecture:

```
(using sh or bash)

            for i in *.sac
            do
            saccvt -I < w$]i > tmp
            mv tmp w$]i
            done
```

or

```
(using csh )

            foreach i ( *.sac)
            saccvt -I < w$]i > tmp
            mv tmp w$]i
            end
```

3.0 Gsac Help

GSAC has a built in HELP command. This is inherent to the program and does not require path to an external file or directory. If you do not know the command, just enter

```
GSAC> HELP
```

This page will look like this:

```
GSAC Command Reference Manual                                HELP

SUMMARY:
     List syntax for GSAC command

     Help [command]

     where command is one or more of the following:
      ABS                         Get absolute value of trace
      AGC                         AGC traces
      ADD                         Add constant to each trace
      BD        BEGINDEVICES      Begin graphics
      BG        BEGINGRAPHICS     Begin graphics
      BP        BANDPASS          Bandpass filter
      BR        BANDREJ           Bandreject filter
                    ...
```

The listing gives the abbreviated and full command names together with a short description of the command. For detailed information on any one command, just enter:

```
GSAC> help pctl



GSAC Command Reference Manual                                PCTL

SUMMARY:
     Control time-domain plots

     PCTL [options]

INPUT:
     X0    x0       : X-position of lower left corner of plot
     Y0    y0       : Y-position of lower left corner of plot
     XLEn   xlen    : Length of x-axis YLen  ylen     : Length of y-
                      axis
     XLAb  x-label  : Label for X-axis
     YLAb  y-label  : Label for Y-axis
     Grid [ ON|OFF] : Turn postioning grid on/off. This is for subplot
               alignment
     Default        : Reset to X0 1.5 Y0 1.0 XLEN 8.0 YLEN 6.0

DESCRIPTION:
     This  set  controls for the current plot. When used with the HOLD
     command, multiple figures can be displayed on a frame,

EXAMPLES:
Default:
     X0 1.5 Y0 1.0 XLEN 8.0 YLEN 6.0 XLAB "Time (s)" YLAB ""

SEE ALSO
GSAC>
```

Carefully note the options. Only the upper case letters are required to define the option. This *PCTL D* and *PCTL DEFAULT* are equivalent.

4.0 Trace Rotation

Is is often necessary and desirable to rotate traces to form radial and transverse components of ground motion, for example. This is possible because the SAC header has two variable that define the sense of positive ground motion for an instrument. A *CMPINC = 0* indicates motion in an upward direction and a *CMPINC = 90* indicates motion in a horizontal direction. The *CMPAZ = 0* indicates north and CMPAZ = 90 indicates east. There is no reason that the horizontal components be oriented in such that positive motion on the horizontal component is in the north or east direction. This is often true with borehole instruments.

To illustrace the process of rotation, we consider the same two horizontal components used in the timing example: `e.sac` and *n.sac*. These have different reference times and different absolute start and end times.

To rotate these components to the great circle, and to rename the component header names *KCMPNM* under SAC (also works with GSAC), the following commands are required:

```
sac2000 << EOF
r e.sac n.sac
synchronize
w e.sac n.sac
cut o 0 o 100
r e.sac n.sac
rot to gc
w r t
cut off
r r
ch kcmpnm LHR
wh
r t
ch kcmpnm LHT
wh
quit
EOF
```

This is complicated. First we must define the same reference time and correctly change header values associated with time offsets. These resultant traces are overwritten. Next these are re-read, cutting them to the same length, rotated and saves as the files *t* and *r*. The cut operation requires that the traces have the same reference time. These in turn are read in separately to change the component names,

GSAC does this all internally:

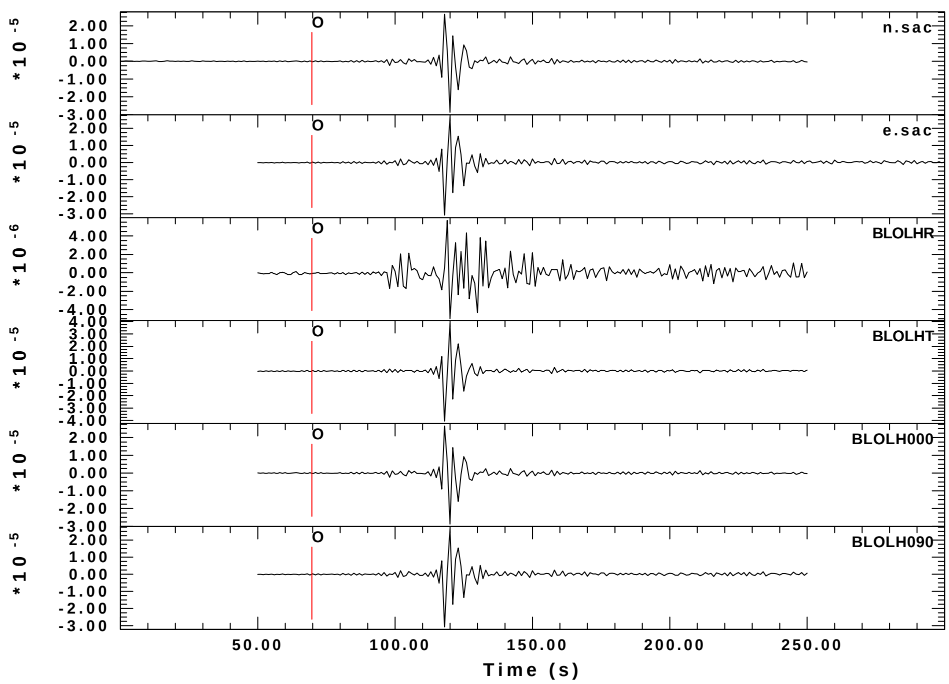
```
GSAC> r [en].sac
GSAC> rot to gc
GSAC> w
```

Note that in this example, no files names are specified in the `w` write command, since GSAC automatically redefines the component name based on the rotation. Note the use of the *glob* shorthand for reading in the files: the *[en].sac* will be expanded to *e.sac* and *n.sac*.

Upon rotation the *CMPAZ* of the horizontals are changed to reflect the new component azimuths. This means that one can rate to the great circle and then rotate back to the north and east components as in the next sequence of commands:

```
GSAC> r [en].sac
GSAC> rot to gc
GSAC> w
GSAC> rot to 0
GSAC>
```

The first write creates (or overwrites - be careful) the files BLOLHR and BLOLHT. The second write creates the files BLOLH000 (which is the same north) and BLOLH090. Plots of the traces are shown in the next figure.



n.sac and e.sac are the initial traces which have different absolute begin and end times as well as reference times. BLOLHR and BLOLHT are the radial and transverse components, respectively. The have the same reference time and absolute start and end times. The BLOLH000 and BLOLH090 are formed from the radial and transverse components and have the same time stamp.

5.0 Trace Merge

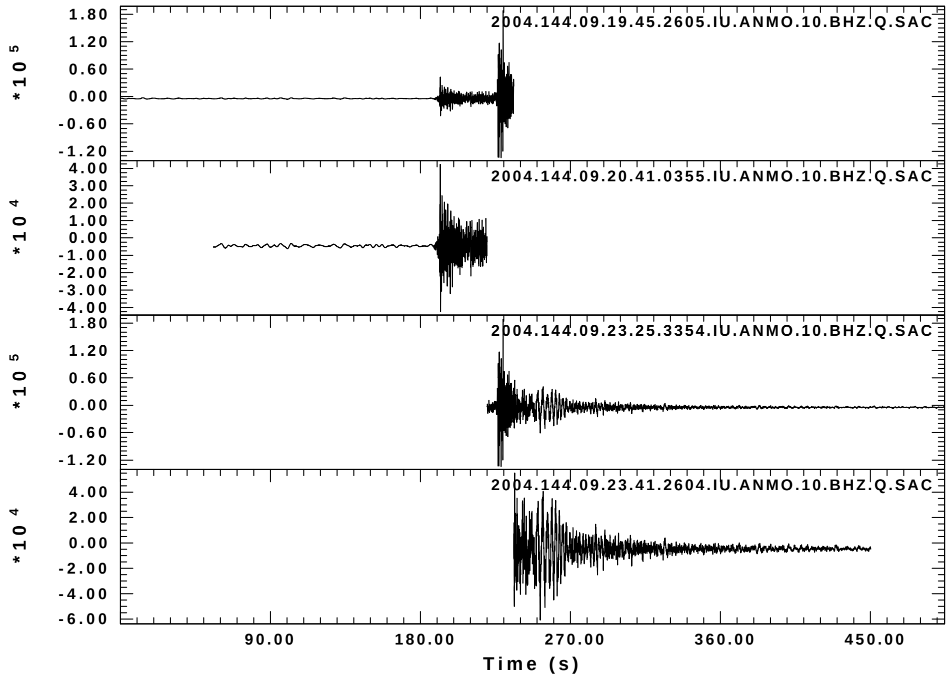
When requesting a data set from the IRIS DMC, on often finds that several overlapping segments are provided rather than the desired continuous trace. If the traces actually overlap, then the GSAC command `merge` can be used. Consider the following traces files obtained by using `rdseed` with a returned IRIS SEED volume:

```
2004.144.09.19.45.2605.IU.ANMO.10.BHZ.Q.SAC
2004.144.09.20.41.0355.IU.ANMO.10.BHZ.Q.SAC
2004.144.09.23.25.3354.IU.ANMO.10.BHZ.Q.SAC
2004.144.09.23.41.2604.IU.ANMO.10.BHZ.Q.SAC
```

GSAC is used to read the traces, display them, merge them into a single trace file and then to save that file:

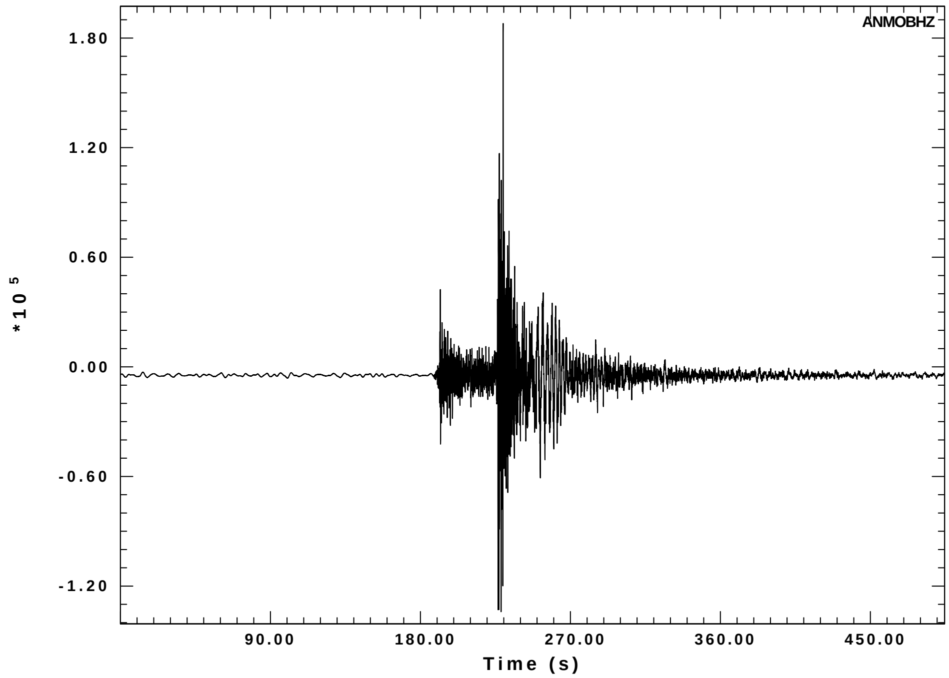
```
rbh@rudy 20040523]w$] gsac
GSAC - Computer Programs in Seismology [V0.1 12 APR 2004]
   Copyright 2004 R. B. Herrmann
GSAC> r *.ANMO.10.BHZ*
2004.144.09.19.45.2605.IU.ANMO.10.BHZ.Q.SAC
 2004.144.09.20.41.0355.IU.ANMO.10.BHZ.Q.SAC
  2004.144.09.23.25.3354.IU.ANMO.10.BHZ.Q.SAC
   2004.144.09.23.41.2604.IU.ANMO.10.BHZ.Q.SAC
GSAC> p1
GSAC> merge
May 23 (144), 2004 09:19:45.259 May 23 (144), 2004 09:23:41.235 (0)
May 23 (144), 2004 09:20:41.035 May 23 (144), 2004 09:23:25.309 (1)
May 23 (144), 2004 09:23:25.335 May 23 (144), 2004 09:27:59.710 (2)
May 23 (144), 2004 09:23:41.259 May 23 (144), 2004 09:27:15.560 (3)

May 23 (144), 2004 09:19:45.259 May 23 (144), 2004 09:27:59.710 (Merge window)
New time series length:  19779
k 0 j 0 - 9439  npts 9440
k 1 j 2231 - 8802  npts 6572
k 2 j 8803 - 19778  npts 10976
k 3 j 9440 - 18012  npts 8573
New default output filename: ANMOBHZ
GSAC> p1
GSAC> w
```

The first plot shows the traces that were read in.



The GSAC `merge` command examines the individual traces to define the time window that would encompass all the data. The number of points in the new time series is indicated (19779) and then the merge occurs. The resultant trace is given the name `ANMOBHZ`. The write operation saves this file.

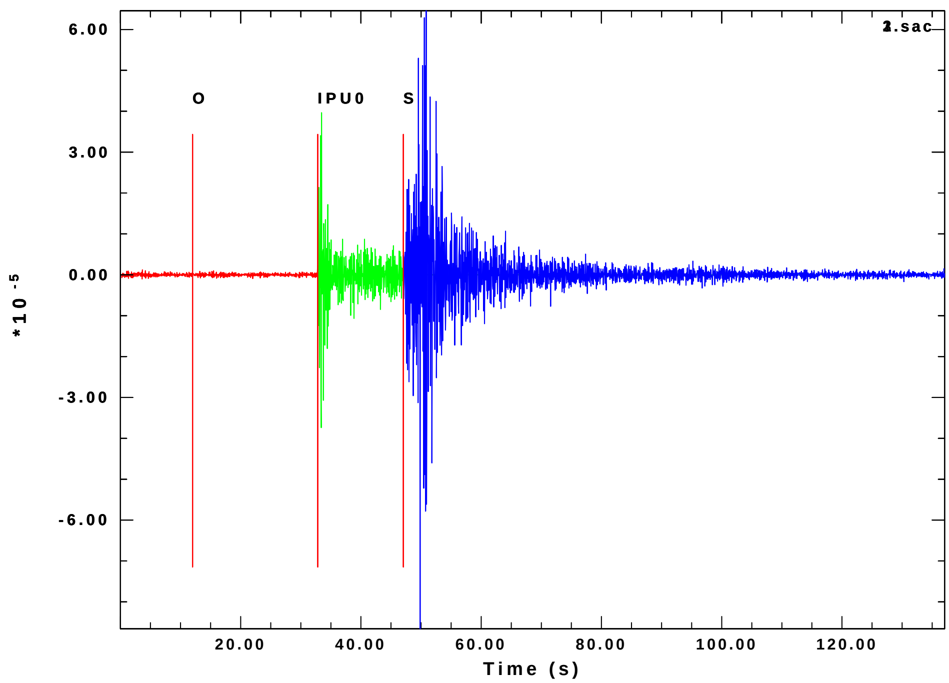
The resultant time series is seen in the second plot:



6.0 Color

GSAC permits the use of color for traces. The following example indicates how one may use color to highlight different portions of a trace. The waveform, from the Korean Meteorological Administration station KAN, has the P-arrival time set in the *A* header value and the S-arrival time set in the *T0* header value. The objective is to plot the sections before P, between P and S, and after S with different colors. To accomplish this, the trace is cut into three parts and saved as three different SAC files. Finally these three are read into memory and plotted.

```
GSAC> cut b 0 a 0
GSAC> r KANHHZ.Sac
GSAC> w 1.sac
GSAC> cut a 0 t0 0
GSAC> r
GSAC> w 2.sac
GSAC> cut t0 0 e 0
GSAC> r
GSAC> w 3.sac
GSAC> cut b 0 b 90
GSAC> r 1.sac 2.sac 3.sac
GSAC> bg plt
GSAC> color on  list red green blue
GSAC> p1 overlay on
```



Plot of signal before P (red), between P and S (green) and after S (blue).

7. GSAC Graphics

GSAC is built upon the CALPLOT graphics package. More importantly is permits interactive window graphics and also the creation of an external CALPLOT binary metafile. This metafile, named P001.PLT, P002.PLT, ... , etc., can be converted to other formats using the programs *plotnps*, *plotgif* or *plotpng*. The *plotnps* can convert to either a PostScript or an Encapsulated PostScript file for printing or inclusion into a document. For good quality bit map graphics, I convert the CALPLOT metafile to Encapsulated PostScript, and then use the UNIX/LINUX/CYGWIN ImageMagick package to convert to GIF, JPEG or PNG.

The CALPLOT printed plot area is very large, but the screen displays are best imagined as being on a piece of paper that is 10 units wide and 8 units high, which is roughly the aspect ratio of a computer screen. When printed this will correspond to a 10 inch x 8 inch region or 25 cm by 20 cm region, which easily fits on a piece of US Letter or A4 paper.

7.1 Window characteristics

In screen mode, the first plotting command will create a window of dimension 800x640 pixels. This default size can be overcome in two ways: specification of the PLOTXVIG environment parameter, or through the GSAC command *bg x* or bg win. If the *bg* command is used, then the *bg* must be invoked *before* the first graphics window is created using the *PLOT*, *PLOTSP* or *PLOTRECORDSECTION* is invoked. The following are some possibilities:

BG X GEOM 1000 800 *(create a larger window with 1000x800 pixels)*
BG X GEOM 1000 800 GRAY *(Use gray scale for colors)*
BG X GEOM 1000 800 COLOR *(Use color)*
BG X GEOM 1000 800 COLOR REVERSE *(Use color and make background BLACK)*

These choices are then used for all screen plots for the current session.

7.2 Plot positioning

By default any plot command places the image on a clean background and uses as much of the 10 x 8 plot space as possible. It is possible to change the length of the axes and the physical position of the plot on the plot space by invoking the *pctl* command before the plot. To assist is positioning, *pctl* can be used to plot the 10 x 8 grid on the screen. *pctl* can also be used to replace the title on the time axis. The default of this command is

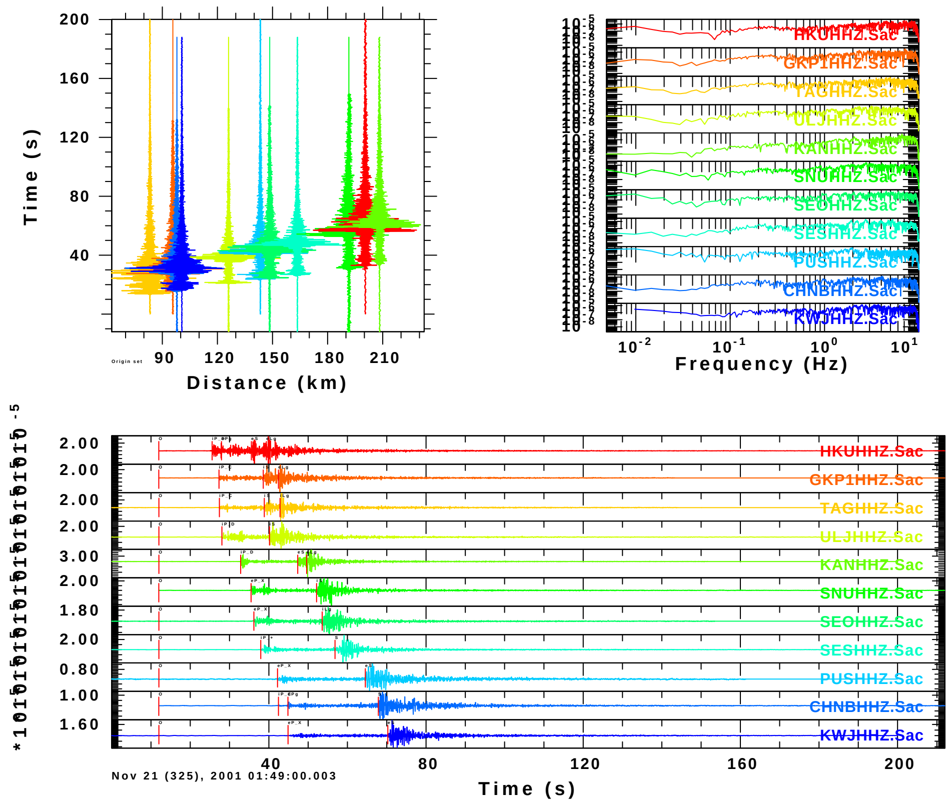
PCTL X0 1.5 Y0 1.0 XLEN 8.0 YLEN 6.0 XLAB "Time (s)" YLAB "" GRID OFF

7.3 Plot HOLD

The real reason for the *PCTL* command is to permit interesting plot overlays. Consider the following example which is run as a SHELL script:

```
!/bin/sh

gsac << EOF
bg plt
r *Z.Sac
sort dist
color rainbow
pctl ylen 3 y0 0.5
hold on
p1
pctl xlen 3 y0 4.5
fft
prs
pctl x0 6.0
psp overlay off
hold off
pctl default
quit
EOF
plotnps -F7 -W10 -EPS -K < P001.PLT >../../DOC/GSAC.TRF/DAT/Chap2/hold.eps
```

The resulting plot is



Result of using the the overlay command. Note that GSAC remembers the current plot state, so that we do not have to define the *ylen* more than once. To reset to original settings, invoke *pctl default*. Note the sequence here: HOLD ON, the plot figures, then HOLD OFF.

8. Data preparation

8.1 saccvt

8.2 Receiver Function Studies

8.3 Surface-Wave Studies

8.3 Source Studies

CHAPTER 3
GSAC SIGNAL PROCESSING

1. Introduction

Transformations

Since all real filters, with corner frequency at $\omega = 1$, will consist of a single pole state of the form

$$H(s) = \frac{1}{s+1}$$

or second order filters of the form

$$H(s) = \frac{1}{s^2 + 2\zeta s + 1}$$

these can be transformed from lowpass to lowpass, highpass, band-bass and band-reject through the following transformations by replacing the Laplcae transfrom variable s with the corresponding values:

Lowpass $\rightarrow$ *lowpass*

$$s \rightarrow \frac{s}{\omega_c}$$

Lowpass $\rightarrow$ *highpass*

$$s \rightarrow \frac{\omega_c}{s}$$

Lowpass $\rightarrow$ *band-pass*

$$s \rightarrow \frac{s^2 + \omega_l \omega_h}{s(\omega_h - \omega_l)}$$

Lowpass $\rightarrow$ *band-reject*

$$\mathbf{s} \rightarrow \frac{\mathbf{s}(\omega_\mathbf{h} - \omega_\mathbf{l})}{\mathbf{s}^2 + \omega_\mathbf{l}\omega_\mathbf{h}}$$

There is one more added complication to consider. Some digital filters, such as the Bessel, are designed to have a lowpass response of 0.707 at the corner frequency, but the filter form to accomplish this is

$$H(s) = \frac{a}{s + a}$$

$$H(s) = \frac{a^2}{s^2 + 2\zeta as + a^2}$$

To accomplish the above transformations we must now do the following:

Lowpass → lowpass

$$s \rightarrow \frac{s}{a\omega_c}$$

Lowpass → highpass

$$s \rightarrow \frac{\omega_c / a}{s}$$

Lowpass → band-pass

$$s \rightarrow \frac{s^2 + \omega_l \omega_h}{s(a\omega_h - a\omega_l)}$$

Lowpass → band-reject

$$s \rightarrow \frac{s(\omega_h / a - \omega_l / a)}{s^2 + \omega_l \omega_h}$$

The lowpass → bandpass or lowpass → bandreject filter transformation will double the number of poles. This will be a problem when we apply a digital transformation since the bilinear transformation also doubles the number of poles in the z^{-1} domain. For this reason it is appropriate to consider the lowpass → bandpass transformation carefully for the 2-pole lowpass filter stage to see if it is possible to express the resultant 4-pole filter as the product of two 2-pole stages

There are two ways to consider this. The first involves complex arithmetic, which is simple under FORTRAN or C++ and the second a numerical root search, which is not unreasonable, since the computer obtains a square-root through a numerical iteration.

1. Complex arithmetic

The second order stage, can be written as

$$\mathbf{H(s)} = \frac{\mathbf{1}}{\mathbf{s^2 + 2\zeta s + 1}} = \frac{\mathbf{1}}{\mathbf{(s-P)(s-P^*)}}$$

where $P = -\zeta + i\sqrt{1-\zeta^2} = e^{i}\theta$. Applying the lowpass $\rightarrow$ band-pass transformation leads to the need to find the roots of the new equation

$$\left(\frac{s^2 + \omega_l\omega_h}{s(\omega_h - \omega_l)}\right) = P$$

or

$$s^2 - sP(\omega_h - \omega_l) + \omega_l\omega_h = 0$$

By completing the square we find that the 4'th order low pass filter will be of the form

$$H(s) = \frac{sB}{(s-P_+)(s-P_+{}^*)}\ \frac{sB}{(s-P_-)(s-P_-{}^*)}$$

$$= \frac{sB}{s^2 + 2\zeta_+\omega_+ s + \omega_+^2}\ \frac{sB}{s^2 + 2\zeta_-\omega_- s + \omega_-^2}$$

where

$$P_\pm = \frac{1}{2}\left[PB \pm \sqrt{(PB)^2 - 4A^2}\right]$$

$2\zeta_\pm\omega_\pm = -2.\,\mathrm{Re}(P_\pm)$ and $\omega_\pm^2 = P_\pm P_\pm{}^*$ and where we let $A^2 = \omega_l\omega^h$ and $B = (\omega_h - \omega_l)$

2. Algebraic

If we apply the lowpass $\rightarrow$ bandpass transformation we obtain the following 4'th order which we attempt to factorize into two 2'nd order filters.

$$H(s) = \frac{s^2b^2}{s^4 + 2\zeta Bs^3 + (2A^2 + B^2)s^2 + 2\zeta BA^2 s + A^4}$$

$$= \frac{sB}{s^2 + 2\zeta_1\omega_1 s + \omega_1^2}\ \frac{sB}{s^2 + 2\zeta_2\omega_2 s + \omega_2^2}$$

where we have used the definition of A^2 and B given above.

Comparing similar powers of s we have

$$2\zeta B = 2\zeta_1\omega_1 + 2\zeta_2\omega_2$$

$$2A^2 + B^2 = \omega_1^2 + \omega_2^2 + 4\zeta_1\zeta_2\omega_1\omega_2$$

$$2\zeta BA^2 = 2\zeta_1\omega_2\omega_2^2 + 2\zeta_2\omega_2\omega_1^2$$

$$A^4 = \omega_1^2\omega_2^2$$

From these we see that $\omega_2 = A^2/\omega_1$ and that $\zeta_1 = \zeta_2$. Using these, the equations reduce to

$$\zeta_2 = \zeta_1 = \frac{\zeta B}{\omega_1 + \frac{A^2}{\omega_1}}$$

and

$$\mathbf{2A^2 + B^2} = \left(\omega_1^2 + \frac{\mathbf{A^4}}{\omega_1^2}\right) + \frac{\mathbf{4A^2}\zeta^2\mathbf{B^2}}{\left(\omega_1 + \frac{\mathbf{A^2}}{\omega_1}\right)^2}$$

This last equation can be solved niumerically using a Newton-Raphson iteration ttech-nique.

CHAPTER 4
GSAC - ELOCATE

1. Introduction

There are many earthquake location programs, each with different degrees of complexity. The program described here, *elocate* was written over 10 years ago with the purpose of being a general piece of code that can be extended. This program also incorporates many of the ideas of the Center for Seismic Studies 3.0 data format which might permit an easy connection of this program with a seismological data base.

elocate has a number of interesting characteristics which were incorporated into its design. These features are as follow:

1. Permit location of events at local, regional and teleseismic distances to 108° through the use of built-in Jeffreys-Bullen P and S-wave tables.
2. Locate teleseisms with a local or regional network array by using the azimuth of approach and phase-velocity of the teleseismic P-wave across the network.
3. Locate local or regional events using one or more local 1-dimensional velocity models and multiple phases.
4. Operate in either batch (non-interactive) or interactive modes.

The output of the program is on the terminal and in a set of temporary summary files that could fit into a CSS 3.0 schema.

The important adjunct to the location program is *sac2eloc* which reads the header information in the SAC traces files for the event, so that transcription of arrival time information is not required. In its simplest form, the following two steps are all that are required to locate an event:

```
rbh> sac2eloc *.sac
rbh> elocate -M 4 -BATCH -D 10
```

The *sac2eloc* command read the SAC traces files and created the file *elocate.dat* which has the station name, the phase identification and time, and the station latitude and longitude. This file is then used by the program *elocate* together with the local velocity model file *VEL.MOD* to locate the event. The arguments of the *elocate* command say to use velocity model 4, the run this in a non-interactive mode, and to use 10 km as the starting depth. Since this was not run in the quiet mode (-*Q* flag), the following output appears on the screen:

```
STA   IWT       ARRIVAL TIME   PhIDQL PHASE  FM CHAN
KAN     0 20011121014932.819    1 0 E P        o  Z
KAN     0 20011121014947.060    2 0 E S        o  Z
KWJ     0 20011121014945.259    1 0 E P        o  Z
KWJ     0 20011121015010.188    3 0 E Lg       o  Z
PUS     0 20011121014942.289    1 0 E P        o  Z
PUS     0 20011121015004.698    3 0 E Lg       o  Z
SEO     0 20011121014936.251    1 0 E P        o  Z
SEO     0 20011121014953.308    2 0 E S        o  Z
SES     0 20011121014937.909    1 0 E P        o  Z
SES     0 20011121014957.597    2 0 E S        o  Z
TAG     0 20011121014927.396    1 0 E P        o  Z
TAG     0 20011121014938.794    2 0 E S        o  Z
ULJ     0 20011121014927.984    1 0 E P        o  Z
ULJ     0 20011121014939.826    2 0 E S        o  Z
    35.9760  128.7194     10.00 20011121014912.585     2436.42
    36.7006  128.3225     13.65 20011121014911.967       11.29
    36.6976  128.3448     13.38 20011121014911.962        9.90
    36.6985  128.3434     13.13 20011121014911.940        9.77
    36.7060  128.3358     11.09 20011121014911.679        3.33
    36.6966  128.3297     11.35 20011121014911.624        2.34
    36.6936  128.3266     11.31 20011121014911.610        2.08
    36.6930  128.3251     11.17 20011121014911.603        2.04
    36.6930  128.3242     11.00 20011121014911.596        2.02
    36.6930  128.3237     10.90 20011121014911.593        2.01
    36.6930  128.3235     10.84 20011121014911.591        2.00
    36.6930  128.3234     10.80 20011121014911.590        2.00
    36.6930  128.3233     10.77 20011121014911.590        2.00
    36.6930  128.3233     10.76 20011121014911.589        1.99
 14 phases used

 STA  COMP DIS(K) AZM  AIN    ARR TIME             RES(SEC) WT QFM PHASE    WGT
 TAG    Z   94.60 164.  97. 20011121014927.396       -0.06  0 Eo P        0.47
 TAG    Z   94.60 164.  97. 20011121014938.794       -0.31  0 Eo S        0.31
 ULJ    Z   96.83  89.  97. 20011121014927.984        0.16  0 Eo P        0.39
 ULJ    Z   96.83  89.  97. 20011121014939.826        0.08  0 Eo S        0.44
 KAN    Z  126.95  23.  47. 20011121014947.060       -1.17  0 Eo S        0.08
 KAN    Z  126.95  23.  95. 20011121014932.819        0.00  0 Eo P        0.39
 SEO    Z  152.79 306.  47. 20011121014953.308       -0.42  0 Eo S        0.16
 SEO    Z  152.79 306.  48. 20011121014936.251        0.13  0 Eo P        0.25
 SES    Z  167.12 274.  47. 20011121014957.597        0.82  0 Eo S        0.09
 SES    Z  167.12 274.  48. 20011121014937.909        0.00  0 Eo P        0.30
 PUS    Z  188.14 160.  94. 20011121015004.698        0.02  0 Eo Lg       0.25
 PUS    Z  188.14 160.  48. 20011121014942.289        1.75  0 Eo P        0.04
 KWJ    Z  208.41 216.  48. 20011121014945.259        2.19  0 Eo P        0.02
 KWJ    Z  208.41 216.  93. 20011121015010.188       -0.18  0 Eo Lg       0.17

 Error Ellipse  X=   0.8015 km  Y= 1.3435 km  Theta =  30.7825 deg

 RMS Error        :              0.181             sec
 Travel_Time_Table:          SCM
 Latitude         :            36.6931 +-    0.0087 N         0.9731 km
 Longitude        :           128.3232 +-    0.0138 E         1.2250 km
 Depth            :              10.75 +-      2.58 km
 Epoch Time       :     1006307351.589 +-      0.18 sec
 Event Time       :  20011121014911.589 +-      0.18 sec
```

this listing has 4 components. The first part shows the phase arrival times at each station. The second follows the iterative location procedure, showing the latitude, longitude, depth, origin time and squared time difference error. The third part shows the arrival time, distance and azimuth for each phase as well as the weight assigned to each phase. The final part gives the error ellipse and summary location information.

This Chapter will describe the use of the current version of these two programs. As these programs are used, the input and output formats will be adjusted to provide better interaction with other programs of the Computer Programs in Seismology.

2. sac2eloc

As will be seen *sac2eloc* runs silently to create a data file for the program *elocate*. However, the SAC file must have certain header values set.

2.1 Sac file preparation

Since the location programs knows nothing about data bases, station coordinate information is obtained from the individual SAC files. This means that some data preparation must be peerformed. It may be necessary to convert a field data format to SAC. Software to do this may be available from the PASSCAL data center for certain data-loggers. If this is not true, then a conversion program can be written using the SAC IO libaries described in Appendix C.

Once the binary SAC files are available, it will be necessary to add the station coordinates to the sac files. This can be done with a simple shell script:

```
#!/bin/sh

if [ $# -eq 0 ]
then
        echo 'Usage: SACIT ascfiles'
        echo '       such as SACIT *.NM'
        exit 0
fi

#####
#       create the coordinate file
#####

cat > coord << EOF
BLO     39.1718   -86.5222
BVIL    38.5137   -89.9238
CBMO    37.3037   -89.5237
CCM     38.0557   -91.2445
CGM1    37.3918   -89.5907
CGM2    37.2897   -89.3693
EOF

for TRC in $*
do

#####
```

```
#       make sure that we have the correct binary format for the SAC file
#####

saccvt -I < $TRC > tmp ; mv tmp $TRC

#####
#       get the station name
#####
KSTNM=‘saclhdr -KSTNM $TRC‘
echo Processing $KSTNM
#####
#       now we get the latitude and longitude for the station
#####
STLA=‘grep ${KSTNM} coord  | awk ’{print $2}’ ‘
STLO=‘grep ${KSTNM} coord  | awk ’{print $3}’ ‘

gsac > /dev/null << EOF
r ${TRC}
ch STLA $STLA STLO $STLO
wh
quit
EOF

done

#####
#       clean up
#####
rm -f coord
```

This shell script runs under UNIX/LINUX/MacOS-X/CYGWIN as follows. A coordinate file is created. This should be the only thing for you to change. Then, we ensure that the SAC file is in the proper binary format for the machine architecture. (Note that the programs *saccvt* does not check to see if the file is actually a SAC file). Then the station name, *KSTNM* is extraced from the trace header. The UNIX/LINUX/POSIX pattern search program *awk* is used to select the station latitude and longitude from the coordinate list. These values are then placed into the SAC header using *gsac*'s *ch* command. The SAC files are now have the necessary information.

2.2 Running sac2eloc

sac2eloc operates by using the arguments on the command line as in the following example:

```
sac2eloc ../DATA/*Z.Sac ../DATA.KIGAM/*Z.sac
```

which says to use all files in the two directories DATA and DAT.KIGAM which end with the pattern *Z.sac*, which are the vertical component traces. *sac2eloc* opens each file, determines if the file is a valid, SAC file

Using the SAC definition of an entry of *-12345.* to indicate an undefined parameter, search through the SAC header fields *A, T0, ..., T9* for fields that are defined. A valid *A* entry is designated `P` - the first arriving P, a valid *T0* entry is designated at `S` - the first arriving S-arrival. The other fields use the corresponding name from the SAC character header, e.g., the phase name associated with the *T8* arrival is the character string *KT8*.

The program also gets the latitude and longitude of the station. The result is the text file *elocate.dat* which has the following form:

```
KAN    Z  2001 11 21 01 49 32.825       1 i P        0 D        1   37.7425  128.8893       25.        0  9999999
SEO    Z  2001 11 21 01 49 36.245       7 e P        2 -        7   37.4879  126.9188       33.        0  9999999
SEO    Z  2001 11 21 01 49 53.697       8 e Lg       2 X        8   37.4879  126.9188       33.        0  9999999
TAG    Z  2001 11 21 01 49 27.396      11 i P        0 C       11   35.8760  128.6194       58.        0  9999999
ULJ    Z  2001 11 21 01 49 27.982      13 i P        0 D       13   36.7021  129.4084       77.        0  9999999
GKP1   Z  2001 11 21 01 49 27.303      17 i P        0 C       17   35.8892  128.5890        0.        0  9999999
HKU    Z  2001 11 21 01 49 25.517      19 i P        0 D       19   36.6102  127.3602        0.        0  9999999
SNU    Z  2001 11 21 01 49 35.226      21 e P        2 X       21   37.4509  126.9566        0.        0  9999999
```

The output format used here is very specific with respect the placement of information with each column. The contents of the 14 columns are as follow:

Column	Description
1.	Station name
2.	Station component name
3.	Year
4.	Month
5.	Day
6.	Hour
7.	Minute
8.	Second
9.	Arrival ID (ARID) from data base - here incremented by 1
10.	Phase quality (I = sharp, E= emergent)
11.	Phase name (P, S or perhaps Pg or Lg)
12.	Phase weight (0=full, 4 = no-weight HYPO71 convention)
13.	First motion
14.	Association ID (ASID)
15.	Station latitude (degrees, N=+, S=-)
16.	Station longitude (degrees,E=+, W=-)
17.	Elevation (meters)
18.	On-date for station
19.	Off-date for station

The concept of the CSS 3.0 data base is that signals are first timed and given a unique Arrival ID. A subsequent step in processing is to identify the phases and then to associate then with an event (Association ID). The arrival times in time in seconds after the 1970/01/01:00:00:00.000 epoch is accomplished through simple routines to convert between human and epoch time.

The reason for using epoch time and for obtaining the station coordinates from the SAC file is that the location program does not need to know anything about local station information databases nor have to worry about calendar dates. The function of *elocate* is just to locate the events, not to make bulletins

3. elocate

3.2 VEL.MOD

The local velocity models are provided in the file named *VEL.MOD*. This file consists of a number of models from which one may be selected for use. The file format follows that of HYPO71 and is very simple. The extension of *elocate* is that more than P and S wave arrivals can be considered. To see some representative models, the *VEL.MOD* can be automatically created by *elocate* in the current working directory by executing the command:

```
elocate -VELMOD
```

A sample velocity model file is created in the current directory with the name *VEL.MOD*. Modify this for your local model. Prior of (22 JAN 2005, this option listed the model to the screen).

```
elocate -VELMOD
```

This VEL.MOD is

```
HALF
1       2
     'P'  'S'
00.0 6.00 3.46
CUS
5       2
     'P'  'S'
00.0 5.00 2.89
01.0 6.10 3.52
10.0 6.40 3.70
20.0 6.70 3.87
40.0 8.15 4.70
UPL
5       3
     'P'  'S'  'Lg'
00.0 5.60 3.23 3.55
02.0 6.15 3.55 3.55
20.0 6.70 3.87 3.55
40.0 8.18 4.72 3.55
97.0 8.37 4.83 3.55
EMBN
6       4
     'P'  'Ps'  'Sp'  'S'
 0.0 1.80 0.40 1.80 0.40
 0.6 5.10 5.10 2.94 2.94
 2.0 6.15 6.15 3.55 3.55
20.0 6.70 6.70 3.87 3.87
40.0 8.18 8.18 4.72 4.72
97.0 8.37 8.37 4.83 4.83
```

This example consists of 4 models, which will be identified internally by the program as models 3 - 6, Model 1 being reserved for the J-B table teleseismic location and Model 2 for the teleseismic beam program. the format for each model is simple:

```
MODEL_NAME
NUMBER_LAYERS   NUMBER_PHASES
(phase strings)
Depth_to_top_of_layer Vel Vel
```

The `CUS` model consists of 4 layers over a halfspace and supports the first arrival `P` and `S` phases. The `UPL` model permits an `Lg` phase which is defined as an arrival with a velocity of 3.55 km/sec. The `EMBN` model supports converted phases at a rock sediment boundary. All of the arrivals are essentially first arrival times, and this this program cannot use phases such as `PmP`, a reflection from the Moho.

3.3 elocate.dat

As described in the section on *sac2eloc*, the arrival time information as weel as station location is given in the fixed format file *elocate.dat*.

3.4 Examples

To demonstrate the capabilities of *elocate*, some representative data sets are in the directory PROGRAMS.XXX/DOC/GSAC.TRF/Chap4, where XXX is the current version number of Computer Programs in Seismology. To run these tests, go to this directory and copy the data sets into the file *elocate.dat* by an operation such as `cp slu.dat elocate.dat`.

Item	Columns	Format	Description
1.	1-6	a6	Station name
2.	8-9	i2	Station component name
3.	11-30	a20	Phase arrival time in seconds past 1970/01/01:00:00:00.000
4.	32-33	i2	Arrival ID (ARID) from data base - here incremented by 1
5.	36	a1	Phase quality (I = sharp, E= emergent)
6.	38-45	a8	Phase name (P, S or perhaps Pg or Lg)
7.	47-48	i2	Phase weight (0=full, 4 = no-weight HYPO71 convention)
8.	50-51	a2	First motion
9.	53-60	i8	Association ID (ASID)
10.	62-70	f9.4	Station latitude (degrees, N=+, S=-)
11.	72-80	f9.4	Station longitude (degrees,E=+, W=-)
12.	82-90	f9.4	Elevation (meters)
13.	92-99	i8	On-date for station
14.	101-108	i8	Off-date for station

3.4.1 Teleseism - local network

The data set here is panda.dat from a University of Memphis deployment in the Mississippi Embayment of the central United States. A unique feature of this region is the thick deposit of surficial sediments with such low velocities that conversion from incident P and S waves to P and S at the rock/sediment interface are very prominent. The sequence of arrivals on the seismogram are P, Ps, Sp and S in order of increasing time. In the *VEL.MOD* in the test directory, the *EMBN* model permits these arrivals by just changing the velocity of the top part of the model. On can locate this event using other models, but the Sp and Ps arrival times will not be used.

3.4.2 Teleseism - local network

The data set *slu.dat* is from a network run by Saint Louis University in the New Madrid Seismic Zone. The earthquake is from the east coast of Russia. The event was located by treating the regional network as an array, using the beam to define the azimuth of approach and phase velocity to determine the location. Because of this data set is limited, essentially sampling one teleseismic ray parameter, the source depth is fixed at 10 km in our example. The command lines to run the case is

```
rbh> elocate  -DEPTH -10 -BATCH -M 2
```

The comparison of the solution with the PDE is

Source	PDE	BEAM
Origin Time	1990 04 21 22 56 55.1	1990 04 21 22 56 02.14 +- 0.11
Latitude	47.46 N	48.08 +- 0.33
Longitude	138.96 E	138.36 +- 0.37
Depth	503 km	10 fixed

The agreement is remarkable given the fact that only 9 arrival times are used.

3.4.3 Teleseism - regional network

Using the data file *peru.dat* and the command

```
rbh> elocate  -DEPTH 100 -BATCH -M 2
```

Source	PDE	BEAM
Origin Time	1992 07 13 18 11 33.7	1992 07 13 18 11 19.266
Latitude 3.919 S	-5.4647 +- 1.41	
Longitude	76.602 W	-75.4435 +- 0.39
Depth 97 km G	100 km F	

The event in peru is located well by a 13 station local network. There is a tradeoff between epicentral distance and source depth when the BEAM method is used.

3.4.4 Teleseism - teleseismic network

Using the data file *arg.dat* and the command

```
rbh> elocate  -LAT 0 -LON -90 -DEPTH 100 -BATCH -M 1
```

Source	ISC	TELE
Origin Time	1988 01 26 18 05 23.0 +- 1.2	1988 01 26 18 05 29.83 +- 0.67
Latitude	27.5 S +- 0.12	28.11 S +- 0.08
Longitude	68.5 W +- 0.19	67.13 W +- 0.13
Depth	300 km	254 +- 13 km

This location agrees well with the ISC location because 5 stations are within a source depth and because there is good azimuthal control.

4. First motion plots

The program *fmplot* will plot the first motion data assembled by *elocate* in the file *fmplot.tmp*. To see the data, just execute the command to create the plot file *FMPLOT.PLT*.

```
rbh> fmplot -F fmplot.tmp
```

To define the focal mechanism, you will require a stereonet, which is generated using the command

```
rbh> stereo
```

which creates the plot *STEREO.PLT*.

The syntax for these commands is as follows:

fmplot *[flags]*, where the command flags are

-eq Equal area projection (default)
-st Stereographic projection
-XX Mxx (1,1) component of moment tensor
-XY Mxy (1,2) component of moment tensor
-XZ Mxz (1,3) component of moment tensor
-YY Myy (2,2) component of moment tensor
-YZ Myz (2,3) component of moment tensor
-ZZ Mzz (3,3) component of moment tensor
-P P-wave display
-SV SV-wave display
-SH SH-wave display
-pol S-wave polarization angle
-RAD rad Radius of circle (default 2.0 in)
-X0 x0 x-coordinate of center of circle (default 4.0 in)
-Y0 y0 y-coordinate of center of circle (default 4.0 in)
-S Strike of fault plane
-D Dip of fault plane
-R Rake or rake angle on plane
-MOM Mom Seismic moment in dyne-cm)default 1.0)
-MW Mw Moment Magnitude
-FMFILL Solid Fill region of positive amplitude (default = .false.)
-FMPLMN Fill region with +- signs related to amplitude (default = false)
-FMAMP Display amplitude contour (default = .false.)
-F file file contains P-wave first motion data
Trend Takeoff-angle ID

where ID =+-1 > Circle/triangle, +-2 -> + or - sign

-ANN Annotate plot with type: P, SV, SH, S or pol
-Z Clears background - useful for Overlays (default=false)
-TT title Title above plot
-TB subtitle Title below plot
-TS titlesize (inches. Default=0.28*rad)
-NM No mechanism only circle and first motion data
-K For amplitude plots use red for zero line
-UP Upper hemisphere projection (default lower)
-F1 f1 -F2 f2 -F3 f3 (default 0.0) point force
-? Usage query, but no execution
-h Usage query, but no execution

stereo *[flags]*, where the command flags are
-DD *dip_inc* Equal area projection (default)
-DD *dip_inc* (default 5) Dip increment in degrees
-DA *az_inc* (default 5) Azimuth increment in degrees
-DS *slip_inc* (default 5) Slip increment in degrees

CHAPTER 5
SAC FILE TOOLS

1. Introduction

A number of tools have been written that work with the SAC file format: **sactoasc**, **asctosac**, **shwsac**, **sacdecon**, **saciterd**, **sacevalr**, **saclhdr** , **sacfilt**, **saccvt**, **sacspc96** and **pltsac**. Of these **saccvt** and **saclhdr** are used the most, followed by **sacevalr**, **pltsac**, and **saciterd**. Programs described in the structure inversion tutorial are **do_mft** which calls **sacmat96** and **sacmft96** and **do_pom** which calls **sacpom96**. The programs **wvfmch96**, **wvfgrd96**, **wvfmt96** and **wvfmtd96**.

2. sactoasc

This program converts a SAC binary trace file to a SAC ascii file. The command is run directly from the command line as:

```
sactoasc SAC_BINARY_FILE SAC_ASCII_FILE
```

The order of arguments is important. The *SAC_ASCII_FILE* will be over-written if it already exists.

3. asctosac

This program converts a SAC ascii trace file to a SAC binary trace file. The command is run directly from the command line as:

```
asctosac SAC_ASCII_FILE SAC_BINARY_FILE
```

The order of arguments is important. The *SAC_BINARY_FILE* will be over-written if it already exists.

4. shwsac

This program reads a SAC file given on the command line, and tells everything possible about the contents of the SAC header. In addition, a simple plot is presented of the trace. The command is run with arguments on the command line:

```
shwsac [-A] [-B] SAC_FILE
```

where the flags **[-A]** and **[-B]** indicate that the SAC_FILE is ascii or binary, respectively. The user is responsible for specifying the type of SAC file, since this information is not directly obtainable from the file itself. An example of the output follows from the invocation of the command

```
shwsac -B B0101ZDD.sac
```

for which the SAC file was created by **f96osac(V)**. The screen output is

```
REAL         INDEX  NAME         INT VALUE   REAL VALUE
                 1  DELTA     0.12500E+00  0.12500E+00
                 2  DEPMIN   -0.86318E-05 -0.86318E-05
                 3  DEPMAX    0.13413E-04  0.13413E-04
                 4  SCALE    -0.12345E+05 -0.12345E+05
                 5  ODELTA   -0.12345E+05 -0.12345E+05
                 6  B         0.00000E+00  0.00000E+00
                 7  E         0.31875E+02  0.31875E+02
                 8  O         0.00000E+00  0.00000E+00
                 9  A        -0.12345E+05 -0.12345E+05
                10  FMT      -0.12345E+05 -0.12345E+05
                11  T0       -0.12345E+05 -0.12345E+05
                12  T1       -0.12345E+05 -0.12345E+05
                13  T2       -0.12345E+05 -0.12345E+05
                14  T3       -0.12345E+05 -0.12345E+05
                15  T4       -0.12345E+05 -0.12345E+05
                16  T5       -0.12345E+05 -0.12345E+05
                17  T6       -0.12345E+05 -0.12345E+05
                18  T7       -0.12345E+05 -0.12345E+05
                19  T8       -0.12345E+05 -0.12345E+05
                20  T9       -0.12345E+05 -0.12345E+05
                21  F        -0.12345E+05 -0.12345E+05
                22  RESP0    -0.12345E+05 -0.12345E+05
                23  RESP1    -0.12345E+05 -0.12345E+05
                24  RESP2    -0.12345E+05 -0.12345E+05
                25  RESP3    -0.12345E+05 -0.12345E+05
                26  RESP4    -0.12345E+05 -0.12345E+05
                27  RESP5    -0.12345E+05 -0.12345E+05
                28  RESP6    -0.12345E+05 -0.12345E+05
                29  RESP7    -0.12345E+05 -0.12345E+05
                30  RESP8    -0.12345E+05 -0.12345E+05
                31  RESP9    -0.12345E+05 -0.12345E+05
```

```
                 32  STLA      0.00000E+00  0.00000E+00
                 33  STLO      0.00000E+00  0.00000E+00
                 34  STEL      0.00000E+00  0.00000E+00
                 35  STDP     -0.12345E+05 -0.12345E+05
                 36  EVLA      0.00000E+00  0.00000E+00
                 37  EVLO      0.10000E+02  0.10000E+02
                 38  EVEL     -0.12345E+05 -0.12345E+05
                 39  EVDP      0.00000E+00  0.00000E+00
                 40  FHDR40   -0.12345E+05 -0.12345E+05
                 41  USER0    -0.12345E+05 -0.12345E+05
                 42  USER1    -0.12345E+05 -0.12345E+05
                 43  USER2    -0.12345E+05 -0.12345E+05
                 44  USER3    -0.12345E+05 -0.12345E+05
                 45  USER4    -0.12345E+05 -0.12345E+05
                 46  USER5    -0.12345E+05 -0.12345E+05
                 47  USER6    -0.12345E+05 -0.12345E+05
                 48  USER7    -0.12345E+05 -0.12345E+05
                 49  USER8    -0.12345E+05 -0.12345E+05
                 50  USER9    -0.12345E+05 -0.12345E+05
                 51  DIST      0.89930E+00  0.89930E+00
                 52  AZ        0.18000E+03  0.18000E+03
                 53  BAZ       0.00000E+00  0.00000E+00
                 54  GCARC     0.00000E+00  0.00000E+00
                 55  SB       -0.12345E+05 -0.12345E+05
                 56  SDELTA   -0.12345E+05 -0.12345E+05
                 57  DEPMEN   -0.72922E-08 -0.72922E-08
                 58  CMPAZ     0.00000E+00  0.00000E+00
                 59  CMPINC    0.00000E+00  0.00000E+00
                 60  XMINIMUM -0.12345E+05 -0.12345E+05
                 61  XMAXIMUM -0.12345E+05 -0.12345E+05
                 62  YMINIMUM -0.12345E+05 -0.12345E+05
                 63  YMAXIMUM -0.12345E+05 -0.12345E+05
                 64  ADJTM    -0.12345E+05 -0.12345E+05
                 65  FHDR65   -0.12345E+05 -0.12345E+05
                 66  FHDR66   -0.12345E+05 -0.12345E+05
                 67  FHDR67   -0.12345E+05 -0.12345E+05
                 68  FHDR68   -0.12345E+05 -0.12345E+05
                 69  FHDR69   -0.12345E+05 -0.12345E+05
                 70  FHDR70   -0.12345E+05 -0.12345E+05
INTEGER     INDEX  NAME        INT VALUE    INT VALUE
                  1  NZYEAR           1970         1970
                  2  NZJDAY              1            1
                  3  NZHOUR              0            0
                  4  NZMIN               0            0
                  5  NZSEC              15           15
                  6  NZMSEC            666          666
                  7  NVHDR               6            6
```

```
                  8  NINF               0            0
                  9  NHST               0            0
                 10  NPTS             256          256
                 11  NSNPTS        -12345       -12345
                 12  NSN           -12345       -12345
                 13  NXSIZE        -12345       -12345
                 14  NYSIZE        -12345       -12345
                 15  NHDR15        -12345       -12345
ENNUMERATED INDEX    NAME       INT VALUE    ENU VALUE
                 16  IFTYPE             1     ITIME
                 17  IDEP          -12345
                 18  IZTYPE             9     IB
                 19  IHDR4         -12345
                 20  IINST         -12345
                 21  ISTREG        -12345
                 22  IEVREG        -12345
                 23  IEVTYP        -12345
                 24  IQUAL         -12345
                 25  ISYNTH        -12345
LOGICAL     INDEX    NAME       INT VALUE    LOG VALUE
                 36  LEVEN              1            T
                 37  LPSPOL             0            F
                 38  LOVROK             0            F
                 39  LCALDA             0            F
                 40  LHDR5              0            F
CHARACTER   INDEX    NAME       INT VALUE   CHAR VALUE
                  1  KSTNM       GRN16        GRN16
                  2  KEVNM       SYNTHETI     SYNTHETI
                  3  KEVNMC      C            C
                  4  KHOLE       -12345       -12345
                  5  KO          -12345       -12345
                  6  KA          -12345       -12345
                  7  KT0         -12345       -12345
                  8  KT1         -12345       -12345
                  9  KT2         -12345       -12345
                 10  KT3         -12345       -12345
                 11  KT4         -12345       -12345
                 12  KT5         -12345       -12345
                 13  KT6         -12345       -12345
                 14  KT7         -12345       -12345
                 15  KT8         -12345       -12345
                 16  KT9         -12345       -12345
                 17  KF          -12345       -12345
                 18  KUSER0      -12345       -12345
                 19  KUSER1      -12345       -12345
                 20  KUSER2      -12345       -12345
                 21  KCMPNM      ZDD          ZDD
```

```
                22  KNETWK       -12345       -12345
                23  KDATRD       -12345       -12345
                24  KINST        -12345       -12345
1  0.
26    7.40105E-07
51    1.51866E-07
76    8.95409E-08
101    5.38562E-07
126   -8.63175E-06
151   -2.72976E-06
176   -2.86479E-07
201   -2.29318E-09
226    1.70056E-08
251    1.41848E-08
256    1.33739E-08
```

and the graphics output is shown in Figure 1. The table contains the components of the floating point, integer and character headers of the SAC file. In addition, the integer values may be interpreted as logical or to represent some ennumerated values. The second column shows the SAC internal name, or something like FHDR70 if there is not official SAC name, the actual value in the header, and the enumerated value. Finally the last 12 lines show the first, last and ten intermediate values of the time history.

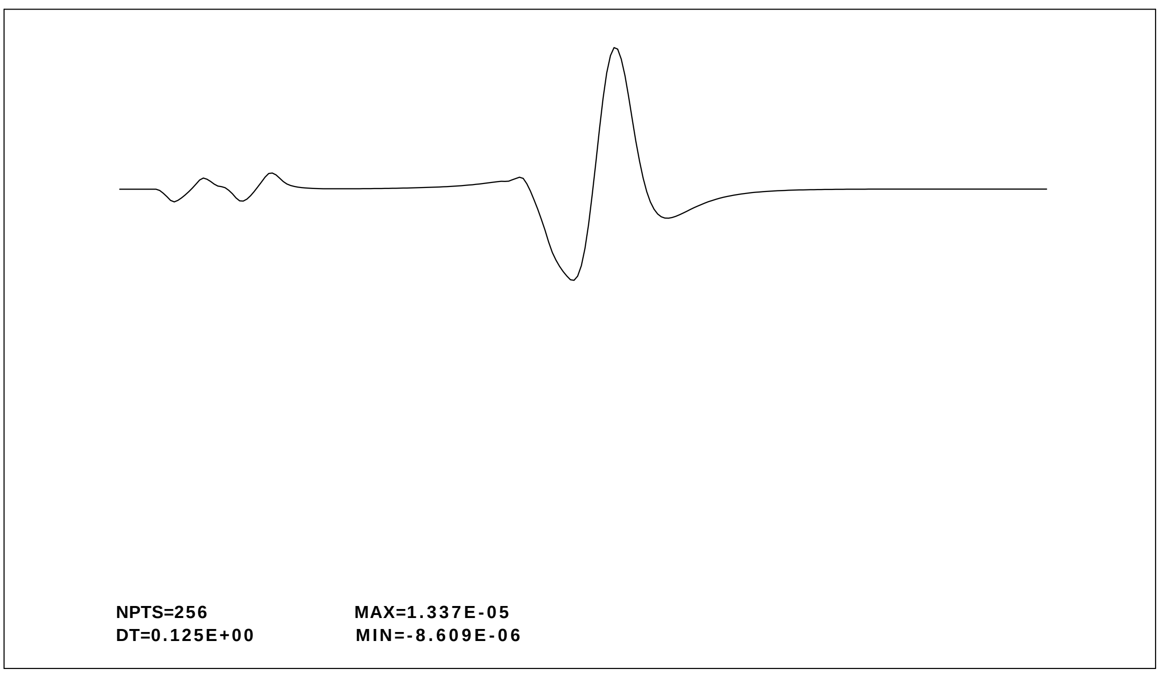


Fig. 1. Graphic output of the program **shwsac(V)**.

5. sacdecon

`sacdecon` performs a deconvolution in the frequency domain by dividing the spectra of the numerator by the spectra of the denominator. To bandlimit the results for a successful deconvolution, a cosine taper can applied. The deconvolution uses a water level set as 0.001 maximum spectral amplitude of the denominator.

The program is invoked as

sacdecon *[flags]*
- **`-FN`** *file_num* (default none) numerator
- **`-FD`** *file_den* (default none) denominator
- **`-W`** (default 0.001) water level
- **`-T`** (default false) force cosine taper in freq domain
- **`-ALP`** *alpha* (default 2.3) complex frequency parameter
- **`-D`** *delay* (0 sec) Begin output delay sec before t=0
- **`-A`** (default false) data are SAC ascii
- **`-B`** (default true) data are SAC binary
- **`-?`**
- **`-h`** Online help.

A sample invocation would be

```
sacdecon -FN numerator -FD denominator -B
```

`The output is given in the file` *sacdecon.bin* if the original files were in SAC binary and in the file *sacdecon.asc* if the original files were in SAC alpha format.

6. saciterd

`saciterd` performs a time domain deconvolution. This program was written by C. J. Ammon and is described in

> Ligorria, J. P. and C. J. Ammon (1999). Iterative deconvolution and receiver-function estimation, *Bull. Seism. Soc. Am.* **89,** 1395-1400.

An important aspect of this program is that the deconvolution is expressed as a sequence of Gaussian filtered impulses. The zero phase Gaussian filter is defined as

$$\mathbf{H(f)} = \mathbf{e}^{-\pi^2 \mathbf{f}^2/\alpha^2} \mathbf{.}$$

The parameter alpha controls the frequency content, with the e^{-1} point at a frequency α/π. Thus an $\alpha = 1.0$ give a lowpass version of the receiver function at a frequency of about 0.3 Hz.
The program is invoked as

saciterd *[flags]*
- **`-FN`** *file_num* (default none) numerator
- **`-FD`** *file_den* (default none) denominator
- **`-E`** *error* (0.001) convergence criteria
- **`-ALP`** *alpha* (default 1.0) Gaussian Filter Width

H(f) = exp(- (pi freq/alpha)**2)
Filter corner ˜ alpha/pi

-N *niter* (default 100) Number iterations/bumps
-D *delay* (5 sec) Begin output delay sec before t=0
-POS (default false) Only permit positive amplitudes
-2 (default false) use double length FFT to avoid FFT wrap around in convolution
-RAYP *rayp* (default -12345.0) Ray parameter in (sec/km) to set in SAC header for use by **rftn96** and **joint96**. This value is not used by this program. Use **udtdd** to determine this value
-?
-h Online help.

Output files:
observed: original numerator convolved with Gaussian
numerator: original numerator convolved with Gaussian
denominator: original numerator convolved with Gaussian
decon.out: Receiver function for Gaussian
predicted: Receiver function for Gaussian
SAC header values
USER0 : gwidth USER5 : fit in %
KUSER0: Rftn KUSER1: IT_DECON KEVNM : Rftn

A sample invocation would be

```
saciterd -FN file_num -FD file_den
```

where *numerator* and *denominator* are binary SAC files.

This program creates several files which are in SAC binary format:
decon.out - final Gaussian filtered deconvolution
observed - Gaussian filtered numerator
numerator - Gaussian filtered numerator
denominator - Gaussian filtered denominator
predicted - prediction of Gaussian filtered numerator obtained by convolving *decon.out* with the original denominator fil *file_den*

In addition the program sets several of the SAC header values in the file *decon.out*:

B - set to -*delay* seconds
USER0 - set to the value of the Gaussian filter parameter *alpha*
KUSER0 - set to the string *Rftn*.
KUSER1 - set to the string *IT_DECON*.
USER4 - set to the value of the ray parameter (sec/km) on the command line. If not specified on the command line, the SAC default value of -12345.0 is used.
USER5 - set to the quality of fit value, e.g., for the example below this will be *99.5* indicating that the predicted numerator accounts for 99.5% of the actual filtered numerator.

The following example uses the program **`hspec96p`** to generate vertical and radial plane-wave synthetics for a teleseismic signal incident from a halfspace onto a 1 km thick soil deposit. The vertical and radial binary SAC files are denoted as `file.Z` and `file.R`, respectively. **`saciterd`** is invoked to perform 100 iterations with $\alpha = \mathbf{5}$.

Figure 2 presents the input traces and the traces generated by the script:

```
#!/bin/sh

cat > dfile << EOF
100.0 0.01 4096  5.0 0.0
EOF

#####
#              define the simple soil model
#####

cat > soil.mod << EOF
MODEL
simple soil model
ISOTROPIC
KGS
FLAT EARTH
1-D
CONSTANT VELOCITY
LINE08
LINE09
LINE10
LINE11
HR      VP      VS  RHO QP  QS  ETAP ETAS FREFP FREFS
1.00     1.80   0.80 1.6 100 100 0.0 0.0 1.0 1.0
10.0     6.1000 3.55 2.7 100 100 0.0 0.0 1.0 1.0
EOF

#####
#              Make synthetic plan wave response
#              This should be a good approximation for teleseisms
#              especially since we are interested in the receiver function
#
#              We only consider upgoing P saves form the source in hspec96p
#####

hprep96p -M soil.mod -d dfile -HS 10 -HR 0 -TF -BH -EQEX -PMIN 0.07 -PMAX 0.07 -DF 0.0
hspec96p -SPUP
hpulse96 -p -V -l 2 | f96tosac -B
mv B0109ZEX.sac file.Z
mv B0110REX.sac file.R
rm B*sac

#####
#              now run saciterd
#####

saciterd  -FN file.R -FD file.Z -N 100 -D 10.0 -E 0.00001 -ALP 5.0 -RAYP 0.07
```

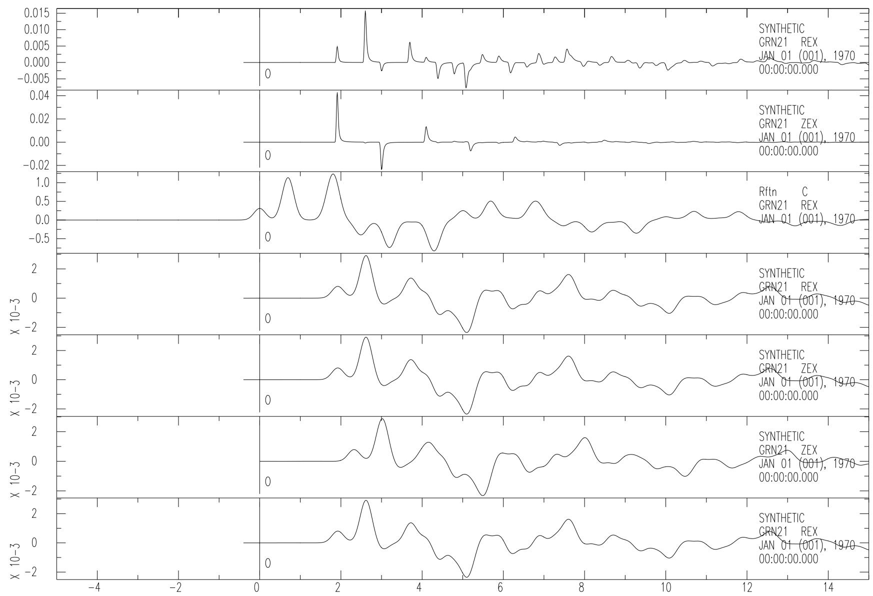


Fig. 2. Traces in order from top to bottom: `file.R`, radial time series; `file.Z`, vertical time series; `decon.out`, receiver function; `numerator`; `denominator`; `predicted` and `observed`. The file `decon.out` would be input to a program to invert the receiver funciton for earth structure.

If the **`-V`** flag is invoked, SAC files are created at each stage of the iteration with names `rXXX`, `dXXX` and `fXXX`, where XXX is the three digit iteration number. For example, `p001` is the predicted Gaussian filtered numerator at iteration 1, `d001` is the receiver function as of iteration 1, and `r001` is the residual receiver function to be fit by further iterations.

7. sacevalr

`The command syntax changed in Version 3.30`

Using the amplitude and phase ascii files output from the IRIS program **`evalresp`**, this program convolves or deconvolves the instrument response.

`sacevalr` *[flags]*

- **`-DEMEAN`** (default false) remove mean before filter
- **`-TAPER`** (default false) apply taper before filter
- **`-FREQLIMITS f1 f2 f3 f4`** apply a taper for deconvolution (-R) This cubic taper ensures that the deconvoled spectrum passband is [f2,f3] and that the spectum is zero for **`f < f1`** and **`f > f4`** with a cubic taper int the region [f1,f2] and [f3,f4].

-A (default true) apply filter
-R (default false) remove filter
-AMP *amp_file* (none) evalresp amp file
-PHA *phase_file* (none) evalresp phase file
-SACIN *binary_sac_input* file (none)
-SACOUT *binary_sac_output* file (none)
-?
-h
Online help

In addition the program sets several of the SAC header values in the file *binary_sac.out* if the instrument response is removed:

USER1 - minimum period in the passband
USER2 - maximum period in the passband
KUSER1 - set to the string *PER_MIN*.
KUSER2 - set to the string *PER_MAX*.
These fields are set when removing the instrument response (**-R**) so that other programs, such as **sacmft96** and **sacpom96**, only use frequencies within the passband of deconvolution. The use of frequency limits ensures a stable deconvolution by a zero-phase bandbass.

8. saclhdr

This program examines the header of a SAC file to return a specified header value. The purpose of the program is to return the header value in a way that permits is to be assigned to a SHELL variable. The following shell script illustrates its use. Note that a different syntax is required in *csh*.

```
#!/bin/sh
#####
#       test sac file using the new saclhdr
#####

ISSAC=‘saclhdr -ISSAC $1‘

if test $ISSAC = 1
then
        echo ISSAC FILE
fi
DIST=‘saclhdr -DIST $1‘
echo $DIST
```

The command is run directly from the command line as:

saclhdr [-?] [-h] -Cmd[s] -NL SAC_ALPHA_OR_BINARY_FILE

The -Cmd is one of the SAC header items listed in the description of **shwsac**. The SAC file can either be machine dependent binary or in the SAC alpha format. The program attempts to determine the data type.

In addition to the SAC header values, we recently added the *NZMON* and *NZDAY* to permit output of month and day.

If the **-NL** flag is not used, then only the first command is output using a C printf statment, such as *printf("%f",fval)* - no newline is placed in the output, which permits it use as a SHELL variable.

If the **-NL** flat is invoked, more than one command can be evaluated, and a space is output in between the fields using a C printf statement, such as *printf("%f ",fval)*, with the entire stream terminated by a newline.

The output is free format. If the SAC header is set to the default no-value or *-12345.*, *-12345* or *"-12345"* for real, integer or string values, then these appear on the output.

Special formatting is used for the *NZYEAR*, *NZJDAY*, *NZMON*, *NZDAY NZHOUR*, *NZMIN* and *NZMSEC* fields to ensure that the complete field widths of 4, 3, 2, 2, 2, 2, and 3, respectively are used. Thus successive queries can lead to the time stamp:

2001 002 05 10 03 010

This is useful for defining unique trace names or wavefrom directories directly from the header:

```
#!/bin/sh

#####
#              query the SAC header to define a name for storing the receiver function
#              for this station
#####
ALP=1.0
for STA in *BHZ*
do
               BASE=‘basename ${STA} .BHZ‘
               saciterd -ALP ${ALP} -P 0.10 -N ${BASE}.BHR -D ${BASE}.BHZ -D 10
               KSTNM=‘saclhdr -KSTNM ${STA}‘
               NZYEAR=‘saclhdr -NZYEAR ${STA}‘
               NZJDAY=‘saclhdr -NZJDAY ${STA}‘
               NZHOUR=‘saclhdr -NZHOUR ${STA}‘
#               be careful here in that the trace time may not be the origin time
#               does the directory for this stations’ receiver functions exist
               if [ -d ../${STA}RFTN ]
               then
               echo exists
               else
               mkdir ../${STA}RFTN
               fi
#####
#              rename the decon.out file and move it to the station directory
#####
               mv decon.out ../${STA}RFTN/${NZYEAR}${NZJDAY}${NZHOUR}${STA}${ALP}
done
```

The *-NL* flag was introduced to permit an quick evaluation of the contents of the SAC header without a lot of SHELL programming. This example examines the header values of some receiver functions.

```
#!/bin/sh
for i in */*.1.0
do
               saclhdr -NL -KSTNM -KCMPNM -USER0 -USER4 -USER6 $i
done
```

9. sacfilt

`The command syntax changed in Version 3.30`

This program applies or removes a general instrument/filter response define in terms of poles and zeros. For compatibility with routines that convert SEED or GSE3.0 to SAC, the pole-zero response is defined in SAC format.

Program control is through command line flags:

sacfilt *[flags]*, where the command flags are

- **`-DEMEAN`** (default false) remove mean before filter
- **`-TAPER`** (default false) apply taper before filter
- **`-FREQLIMITS f1 f2 f3 f4`** apply a taper for deconvolution (-R) This cubic taper ensures that the deconvoled spectrum passband is [f2,f3] and that the spectum is zero for **`f < f1`** and **`f > f4`** with a cubic taper int the region [f1,f2] and [f3,f4].
- **`-A`** (default true) apply filter
- **`-R`** (default false) remove filter
- **`-PZ pole_zero_file`** (none) SAC response file
- **`-SACIN`** *binary_sac_input* file (none)
- **`-SACOUT`** *binary_sac_output* file (none)
- **`-h`**
- **`-?`** Online help concerning program usage

In addition the program sets several of the SAC header values in the file *binary_sac.out* if the instrument response is removed:

USER1 - minimum period in the passband
USER2 - maximum period in the passband
KUSER1 - set to the string *PER_MIN*.
KUSER2 - set to the string *PER_MAX*.
These fields are set when removing the instrument response (**`-R`**) so that other programs, such as **`sacmft96`** and **`sacpom96`**, only use frequencies within the passband of deconvolution. The use of frequency limits ensures a stable deconvolution by a zero-phase bandbass.

10. saccvt

This program addresses the problem of transporting SAC binary files between a SPARC, or other machine using IEEE big-endian INTEL little-endian representations of numbers. This utility thus performs the necessary byte swaps to accomplish this.

Program control is through command line flags:

saccvt *[flags]*, where the command flags are

-I (default none) intelligently guess whether to convert
-h
-? Online help concerning program usage

The use of the program is illustrated by executing **saccvt -h** or **saccvt -?**, which gives:

```
 Convert SAC binary  IEEE to INTEL format
  Convert SAC binary  INTEL to IEEE format
  All 4 byte integers and floats (a,b,c,d) are
  transposed to (d,c,b,a)
Example: saccvt < SAC_BINARY > tmp ; mv tmp SAC_BINARY
-I   (default none)  intelligently guess whether to convert
-h   (default none)  this help message
-?   (default none)  this help message
```

Ultimately this program will be modified to permit the files to be converted to be given on the command line. The reason for the two stage process outlined above is to ensure that the conversion is a conscious act. The other alternative is to use the **sactosac** and **asctosac** routines to convert the native binary SAC file to ASCII, transfer the ASCII between the machines, and then convert the ASCII to the other machine's binary SAC file format. The use of **saccvt** is the preferred mechanism.

The **-I** flag is the latest addition to the code. If this flag is invoked, then the file is examined by looking for the pattern -12345. in the floating point header values or the integer -12345 in the integer header values. if this is NOT seen then the file is converted. Otherwise if is not. I usually run this program in a shell script:

```
for i in *sac
do
            saccvt -I < i > tmp; mvtmpi
done
```

This will ensure that all SAC files in the current directory are in the format for the local architecture.

11. sacspc96

This program computes the Fourier amplitude spectra of the binary SAC file and plots it. This purpose was written to permit the overlay of theoretical spectra from surface-wave synthetic programs onto the spectral amplitudes estimated by the multiple filter analysis programs, `sacmft96`.

Program control is through command line flags:

sacspc96 *[flags]*, where the command flags are

-FREQ	(default true) X-Axis is frequency
-PER	(default false)X-Axis is period
-XMIN xmin	(default 0.0) minimum value of X-Axis
-XMAX xmax	(default) maximum value of X-Axis
-YMIN ymin	(default 0.0) minimum value of Y-Axis
-YMAX ymax	(default 0.0) maximum value of Y-Axis
-X0 x0	(default 2.0) lower left corner of plot
-Y0 y0	(default 1.0) bottom left corner of plot
-XLEN xlen	(default 6.0) length of X-Axis
-YLEN ylen	(default 6.0) length of Y-Axis
-K kolor	(default 1) color for curves
-NOBOX	(default false) do not plot axes
-XLOG	(default linear) X axis is logarithmic
-YLOG	(default linear) Y axis is logarithmic
-f sacfilename	(default none) Binary SAC file name
-h	(default false) online help
-?	(default false) online help

12. pltsac

This program computes the Fourier amplitude spectra of the binary SAC file and plots it. This purpose was written to permit the overlay of theoretical spectra from surface-wave synthetic programs onto the spectral amplitudes estimated by the multiple filter analysis programs, `sacmft96`.

Program control is through command line flags:

pltsac *[flags]*, where the command flags are

-XLEN xlen	(default 6.0) Length X-axis
-YLEN ylen	(default 6.0) Length Y-axis
-NMIN nmin	(default 1) First point to plot
-NMAX nmax	(default 64000) last point to plot
-PCY pcy	(default 0.6) Fraction of dy for trace amplitude
-YBOT ybot	(default 0.0) if ymax < ybot trace is zero
-ABS	(default false) plot absolute amplitudes
-X0 x0	(default 2.0) x-position of lower left corner
-Y0 y0	(default 7.0) y-position of lower left corner
-K PEN	(default 1) Use color for

if kolor < 0 use red->blue progression

-DOAMP	(default false)	Annotate with amplitude value
-DOPLUS	(default false)	Shade positive values
-DOMINUS	(default false)	Shade negative values
-O	(default false)	Overlay traces no y space
-OS	(default false)	Overlay traces and shade
-TSCT	(default false)	Put time scale/label at top
-TSCB	(default false)	Put time scale/label at bottom
-TTMT	(default false)	Put time scale grid at top
-TTMB	(default false)	Put time scale grid at bottom
-USER9	(default false)	Show wvfgrd96 time shift
sac_files		Sac binary trace files
-h	(default false)	online help
-?	(default false)	online help

The placement of traces on a page is a little complicated because of the many command options. After some trial an error, the following shell script is used:

```
#!/bin/sh
set -x

Y0=8
$T=‘echo $Y0 | gawk ’{print $1 - 0.5 }’‘
rm -f CMP1.plt
calplt << EOF
NEWPEN
1
CENTER
1.25 ${YT} 0.2 ’Z’ 0.0
CENTER
3.50 ${YT} 0.2 ’R’ 0.0
CENTER
5.75 ${YT} 0.2 ’T’ 0.0
PEND
EOF
cat CALPLT.PLT > CMP1.plt
rm -f CALPLT.PLT CALPLT.cmd

#####
#               make up the list of stations
#####
rm -f dlist

for i in *[ZRT]
do
KSTNM=‘saclhdr -KSTNM $i‘
DIST=‘saclhdr -DIST $i‘
echo $DIST $KSTNM >> dlist
done
cat dlist | sort -n | uniq | awk ’{print $2}’ > sdlist
edlist=‘tail -1 sdlist‘
rm -f ssdlist

for STA in ‘cat sdlist‘
do
Y0=‘echo $Y0 | gawk ’{print $1 - 1.0 }’‘

case $STA in
$edlist)
```

```
pltsac -USER9 -TSCB -K -1 -DOAMP  -XLEN 2.0 -X0 0.25 -Y0 ${Y0} -ABS -YLEN 1.0 ${STA}Z*
cat PLTSAC.PLT >> CMP1.plt
pltsac -USER9 -TSCB -K -1 -DOAMP  -XLEN 2.0 -X0 2.50 -Y0 ${Y0} -ABS -YLEN 1.0 ${STA}R*
cat PLTSAC.PLT >> CMP1.plt
pltsac -USER9 -TSCB -K -1 -DOAMP  -XLEN 2.0 -X0 4.75 -Y0 ${Y0} -ABS -YLEN 1.0 ${STA}T*
cat PLTSAC.PLT >> CMP1.plt
               ;;
*)
pltsac -USER9 -K -1 -DOAMP  -XLEN 2.0 -X0 0.25 -Y0 ${Y0} -ABS -YLEN 1.0 ${STA}Z*
cat PLTSAC.PLT >> CMP1.plt
pltsac -USER9 -K -1 -DOAMP  -XLEN 2.0 -X0 2.50 -Y0 ${Y0} -ABS -YLEN 1.0 ${STA}R*
cat PLTSAC.PLT >> CMP1.plt
pltsac -USER9 -K -1 -DOAMP  -XLEN 2.0 -X0 4.75 -Y0 ${Y0} -ABS -YLEN 1.0 ${STA}T*
cat PLTSAC.PLT >> CMP1.plt
               ;;
esac
calplt << EOF
NEWPEN
1
LEFT
7.00 ${Y0} 0.10 '${STA}' 0.0
PEND
EOF
cat CALPLT.PLT >> CMP1.plt



done
plotnps -K -EPS -F7 -W10 < CMP1.plt > cmp1.eps


rm CALPLT.cmd
rm CALPLT.PLT
```

Figure 3 shows the result of applying this script. The purpose of this script is to plot the observed traces in red and the predicted traces in blue. The traces are ordered according to the vertical, radial and transverse components. The stations are plotted in order of increasing epicentral distance. Each pair of traces is plotted using the same amplitude scale, and the peak amplitudes are annotated to the left of each trace. Since the predicted traces were generated using the programs **wvfgrd96** with the **-n 20** time shift flag, the header variable *USER9* indicated the optimal time shift between the observed and predicted traces.

To create this figure the program **gawk** (or **nawk** on Solaris) is used. The program **CALPLT** is used to define the figure annotation. *saclhdr* is used to determine the unique station - distance listing. Then the plotting begins for each station in the *for* loop of the script. Special care is taken if the station is the last on the list - if it is, then we also annotate with a time scale. The script used the property of the CALPLOT binary file that permits the individual plots to be concatenated. Finally the station names are placed to the right of each set of traces.

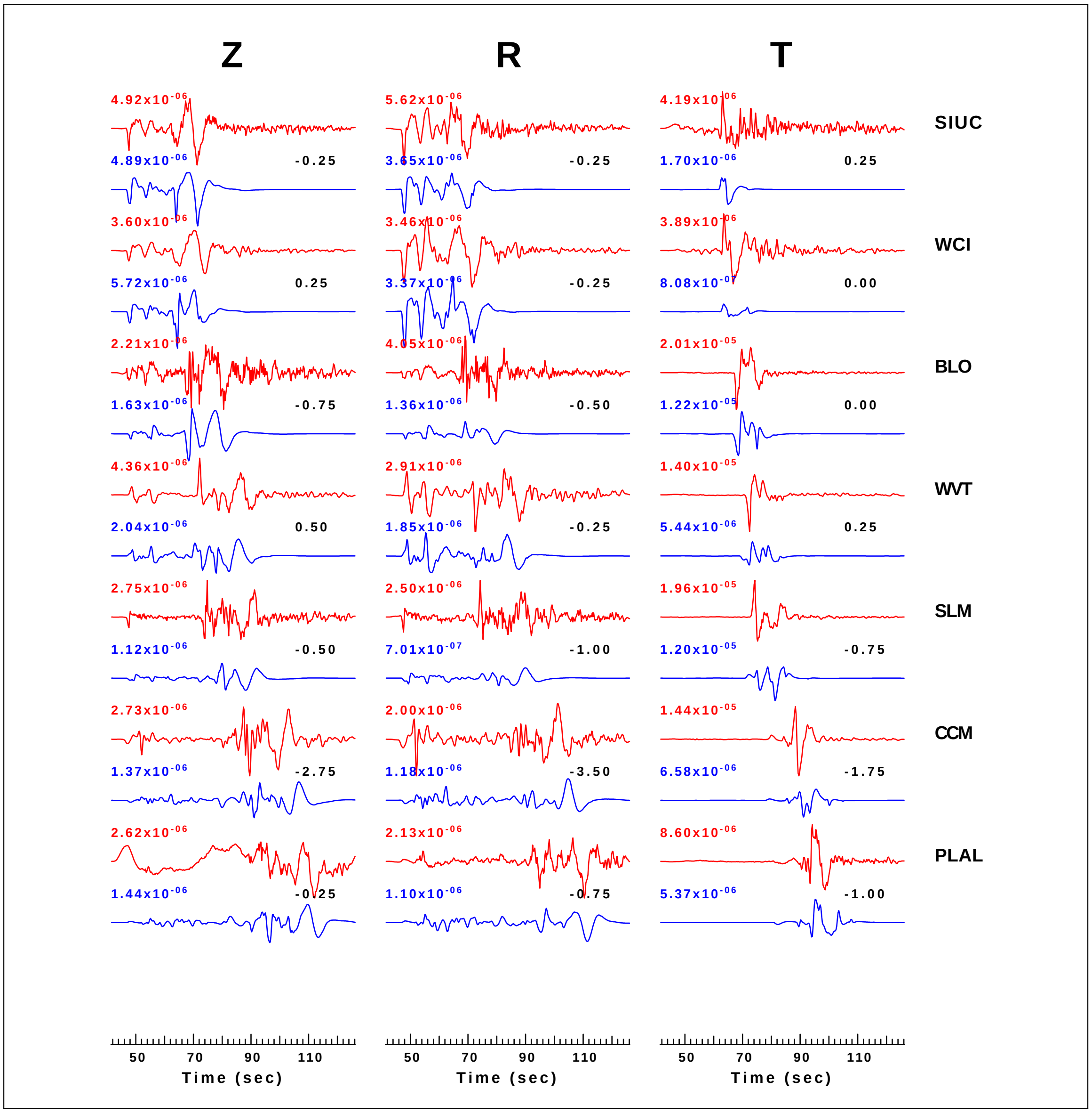
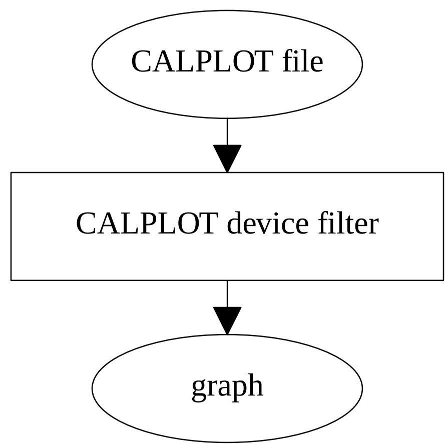


Fig. 3. Graphic output of the program **shwsac(V)**.

APPENDIX A
CALPLOT GRAPHICS

1. Introduction

Computer Programs in Seismology is distributed with its own graphics package to make the installation of distributed software easier. Each non-interactive graphics program will create a binary, device-independent metafile of plotting commands, which must be converted for use by a specific hardware device. At the simplest level, the low level plotting commands are a sequence of pen up, pen move and pen down commands. Some of the early plotting devices supported were Calcomp mechanical plotters, Versatec electrostatic printers and Tektronix graphics terminals. Today graphic output is supported for X-Windows, PC Windows displays, and PostScript printers. Only a small subset of output devices are currently supported, primarily because of the existence of excellent conversion software; one example of which is *ghostscript* which converts PostScript to many devices.



If a CALPLOT device filter is named **plotdev**, one uses the program as follows:

`plotdev [options] < PLOTFILE` *for a screen device*

`plotdev [options] < PLOTFILE > temp_file` *(create temp file)*
`print temp_file` *(output to the actual printer)*

Some common options are

`-S`*scaling_factor*
Multiply all plot moves by the *scaling_factor* (default 1.0)
`-R`
Rotate the plot by 90°
`-F`*font*

Change the default fault to number *font*. The default is Times Roman. A **-F7** will invoke bold Helvetica in PostScript.

Other commands are specific to the hardware device. A complete description of all supported devices is given in **CALPLOT(I)** of *Computer Programs in Seismology*.

2. PostScript Output

The program **plotnps** converts the binary CALPLOT file to PostScript. This program supports 128 unique colors in its palette. The output can also be in the form of Encapsulated PostScript, which is used to provide all graphics in this document.

Program control is through the command line:

plotnps *[flags]*, where the command flags are

-S*scalefac*
Scale all plot motions by this factor.

-P*pipe_process*
On UNIX/LINUX pipe the PostScript output through this process instead of sending through *stdout*

-R
Rotate the plot on the printed page. In effect the plot region is 8.5" wide and 11.0" high instead of 11.0" wide and 8.5" high.

-N
Turn off shading options for smaller PostScript file

-F*font*
Make the default font equal *font*

font	Font Used
0	Times-Roman
1	Times-Roman
2	Times-Italic
3	Times-Bold
4	Symbol (Greek)
5	Helvetica
6	Helvetica-Oblique
7	Helvetica-Bold
8	Symbol (Greek)
9	Courier
10	Courier-Oblique
11	Courier-Bold
12	Symbol (Greek)

-H30
-H60
Use a halftone density of 30 or 60 (default) dots per inch. The density of 30 produces larger dots, and may be of use when a figure must be reduced for publication.
-K
Show colors with a red -> green -> blue palette
-KR
Show colors with a red -> white -> blue pallette
-KB
Show colors with a blue -> white -> red pallette
-KW
Show colors, but whiten the spectrum.
-G
Show colors in grayscale.

The default action when neither **-G**, **-K** nor **-KW** are used is that shading is in gray, but all colored lines and text are black.
-B
assume the paper is 11 x 14 instead of 8.5 x 11
-L
assume the paper is 8.5 x 14 instead of 8.5 x 11
-A3
assume the paper is A3 instead of 8.5 x 11
-A4
assume the paper is A4 instead of 8.5 x 11
-W*min_linewidth*
Reset the minimum line width.
-EPS
Make the output an Encapsulated PostScript file.
-T*title*
Place the title string *title* in the lower left corner. Do not use spaces, or under UNIX/LINUX place string between quotes, e.g., **-T***"a test case"*
-h
-?
Online help

Standards

To be compatible with PostScript display software and with word processing software that permits inclusion of PostScript files, PostScript Document Structure Convention 3.0 (DSC 3.0) is followed.

Plotspace Mapping

The CALPLOT definition of axes is such that the X-axis is horizontal and the Y-axis is vertical. This is then mapped onto a printed page of dimension 8.5" x 11". In the default case the X-axis is mapped onto the long dimension of the paper. The plot space on the paper is demonstrated in Figure 1.

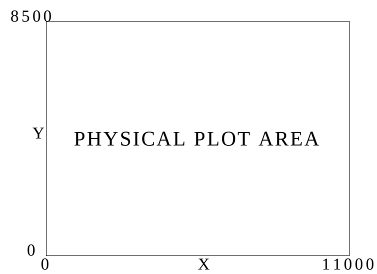


Fig. 1. Default mapping of CALPLOT plot space onto physical page.

The **-R** option rotates the mapping, such that the Y-axis is mapped onto the long dimension of the paper. The plot space on the paper is demonstrated in Figure 2.

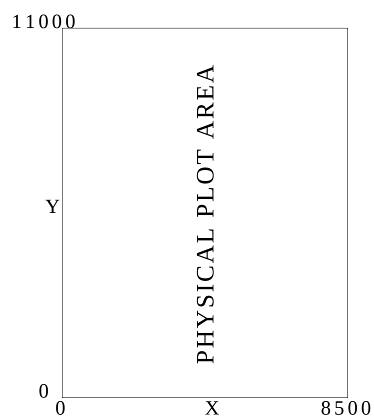


Fig. 2. Mapping of CALPLOT plot space onto physical page using *-R*.

Note that the CALPLOT plot space is mapped onto a rectangular page with no distortion of the unit lengths of the X- or Y-axes.

The PostScript plot space is assumed to be that the X-axis is horizontal with a length of 8.5" and the Y-axis is vertical with a length of 11.0". The use of the **-EPS** or **-LEPS** options permits a plot to be able t0 fit within these limits. For the **-EPS** option, the CALPLOT X-axis will still be horizontal. The default and **-R** option changes the lengths of the plotted axes in the manner consistent with Figures 1 and 2. The **-LEPS** option make the X-axis parallel to the long direction of the page.

The following examples use the second page of the example file *PLTTST* to illustrate the result of using this program with different options.

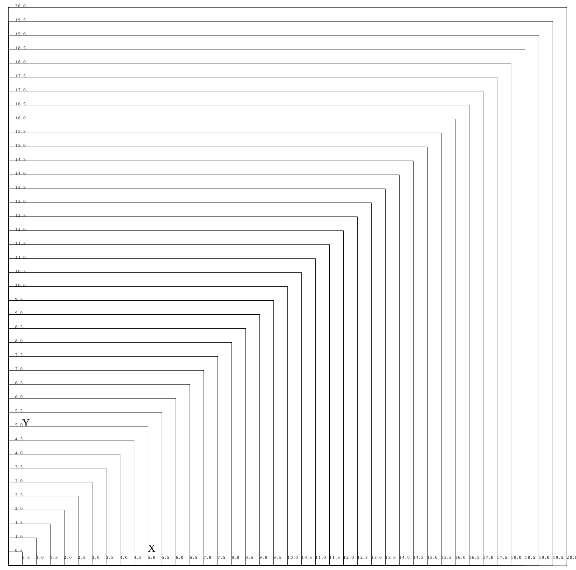


Fig. 3. This is the result of using *plotnps -EPS < plt > plttst.eps*

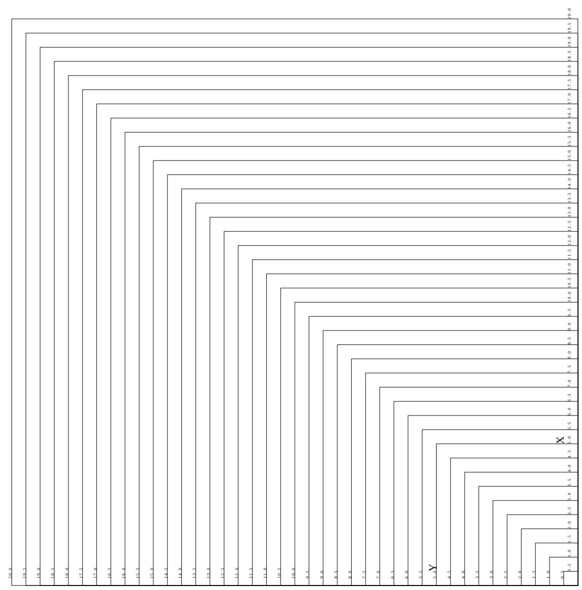


Fig. 4. This is the result of using *plotnps -EPS -R < plt > plttst.rps*

3. Windows Screen Output

The program **plotmsw** is a native WIN32 program for use under the MS Windows (WIN 95/98/NT etc) windowing system. It is built on the *graphapp* package.

Program control is through the command line:

`plotmsw` *[flags]*

`-S`*scalefac*

Scale all motions by this factor.

`-F`*font*

Change the default font.

`-R`

Rotate the plot by 90°

`-N`

No shading

`-I`

Invert the background. The background will be black instead of white. This is done by interchanging the black and white color map entries

`-K`

(default) Show colors with a red -> green -> blue pallette

`-KR`

Show colors with a red -> white -> blue pallette

`-KB`

Show colors with a blue -> white -> red pallette

`-G`

Grayscale *(color is default)*

`-W`*width*

Minimum linewidth in units of 0.001" or 0.0025cm

`-geometry width`xheight+-xoff+-yoff

set geometry in manner of X11. The *xoff* and *yoff* are optional. *(Default width=800, height=640).*

`-p`

`-p2`

`-p4`

`-p10`

Put up background positioning grid every 1000 CALPLOT units. every 500, every 250 or every 100, respectively

To provide additional user control, the command line arguments can be placed in the environment by separating them by colons (:) with no intervening spaces. **This is the only way to change display options when using the graphics libraries are used for interactive plots**. To force a scale factor of 0.5, and the images size of 800x600 one would set the environment parameter **PLOTMSW**

```
set PLOTMSW=:-S0.5:-g:800x600:
```

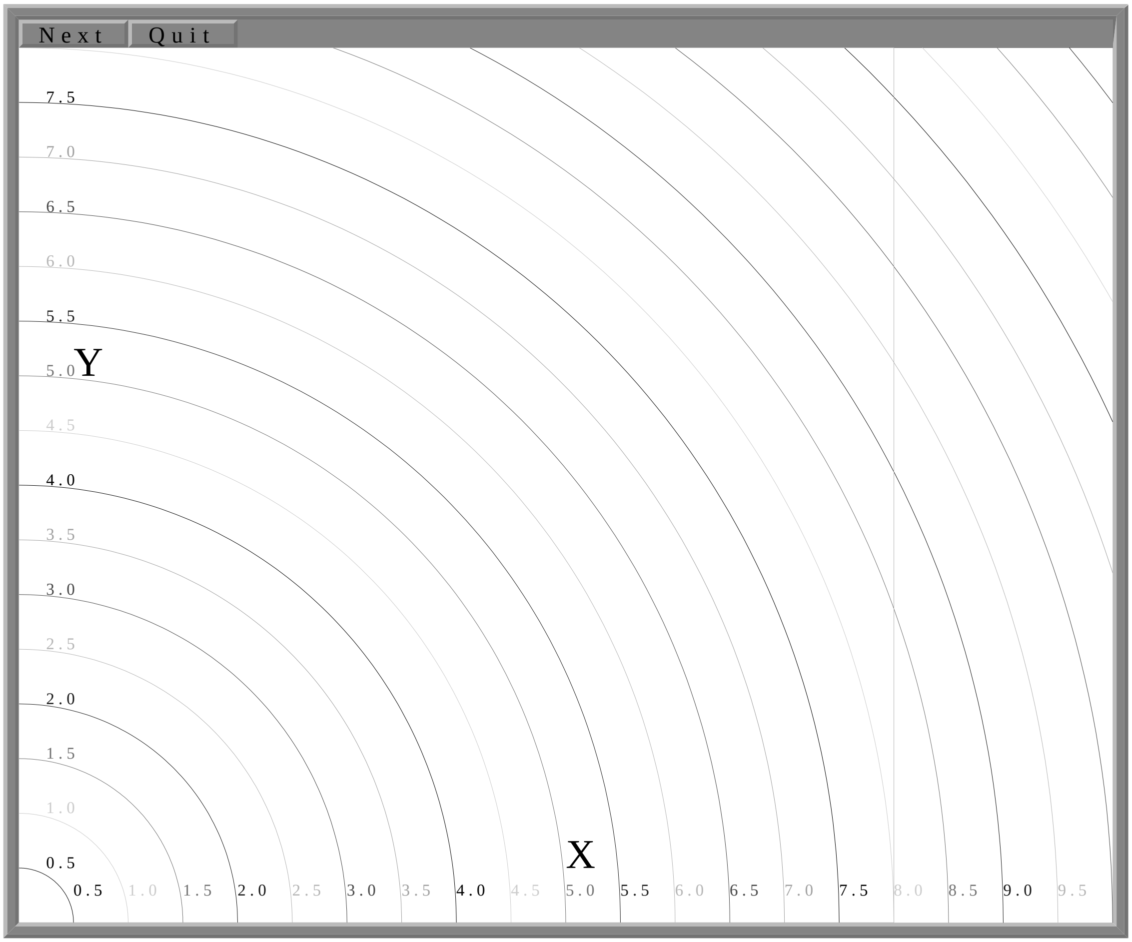
`-h`

`-?`

Online help

The Windows screen is viewed as a piece of paper exactly 10.0" wide and 8.0" high (approximately 25.4 cm wide by 20.32 cm high). The default screen has dimensions of

800 x 640 pixels.



When the page is completely drawn, a cursor will appear. One can use this to point to a feature of interest. The following actions can be performed:

- Pressing and releasing the *Left Mouse Button* will advance the page.
- Pressing the menu button *Next* will advance the page.
- Pressing the menu button *Quit* terminates the plot.

In order of importance, an entry such as this overrides a command line or environment option. For the other options, the command line overrides the PLOTMSW environment control.

Last Modified 05 NOV 2001

4. Tektronix Output

Although few will possess an actual Tektronix 4010 or 4014 graphics terminal, **kermit (3.3)** (for dial up modems) and **teraterm** (for use under Windows 3.1 and Winsock for PPP or SLIP), support this protocol and also have added some color capability. These capabilities permit remote examination of graphics output.

Program control is through the command line:

plot4014 *[flags]* , where the command flags are

-S*scalefac*
: Scale all plot motions by this factor.

-R
: Rotate plot by 90°

-N
: Turn off shading for speed.

-T4025
: Plotting device is a Tektronix 4025 terminal.

-K
: Plotting device is KERMIT 3.0. Color shading is not permitted even though color lines are. Also switch KERMIT from VT emulation to TEK emulation

-W*sleeptime*
: Set a delay between plot frame. This delay is set at the computer, and will have no effect for modem or internet connections.

-D*reduc*
: Reduce the resolution. The original 4014 had a 4096 x 3120 resolution. Emulator displays on PC's may have something like 500 x 400. Thus any attempt to plot greater detail will not be resolvable. If the plot driver does not send these coordinates, the result is significantly less data transmission and faster plotting. A **-D4** is useful.

-F*font*
: Change the default font.

-TT
: Plotting device is TERATERM. Color shading is not permitted even though color lines are is now permitted.

-h

-?
: Online help

5. X11 Output

The program **plotxvig** is a native X11 program for use under the X11 windowing system. It is based on the XviG Version 1.1 package (Antoon Demarrée, IMEC, © 1993). This program supports 35 unique colors in its palette. If these colors are not available, dithering is used to create the apparent set. This package is also the basis of interactive X11 software.

Program control is through the command line:

plotxvig *[flags]*

-S*scalefac*

Scale all motions by this factor.

`-F`*font*

Change the default font.

`-R`

Rotate the plot by 90°

`-N`

No shading

`-I`

Invert the background. The background will be black instead of white. This is done by interchanging the black and white color map entries

`-K`

(default) Show colors with a red -> green -> blue pallette

`-KR`

Show colors with a red -> white -> blue pallette

`-KB`

Show colors with a blue -> white -> red pallette

`-G`

Grayscale *(color is default)*

`-W`*width*

Minimum linewidth in units of 0.001" or 0.0025cm

`-geometry width`*x*height+-xoff+-yoff

set geometry in manner of X11. The *xoff* and *yoff* are optional. *(Default width=800, height=640).*

`-p`

`-p2`

`-p4`

`-p10`

Put up background positioning grid every 1000 CALPLOT units. every 500, every 250 or every 100, respectively

To provide additional user control, the command line arguments can be placed in the environment by separating them by colons (:) with no intervening spaces. **This is the only way to change display options when using the graphics libraries are used for interactive plots**. To force a scale factor of 0.5, and the images size of 800x600 one would set the environment parameter **PLOTXVIG**

`set PLOTXVIG=:-S0.5:-g:800x600:`
`export PLOTXVIG` *(under sh or ksh)*

or

`setenv PLOTXVIG=:-S0.5:-g:800x600:` *(under csh)*

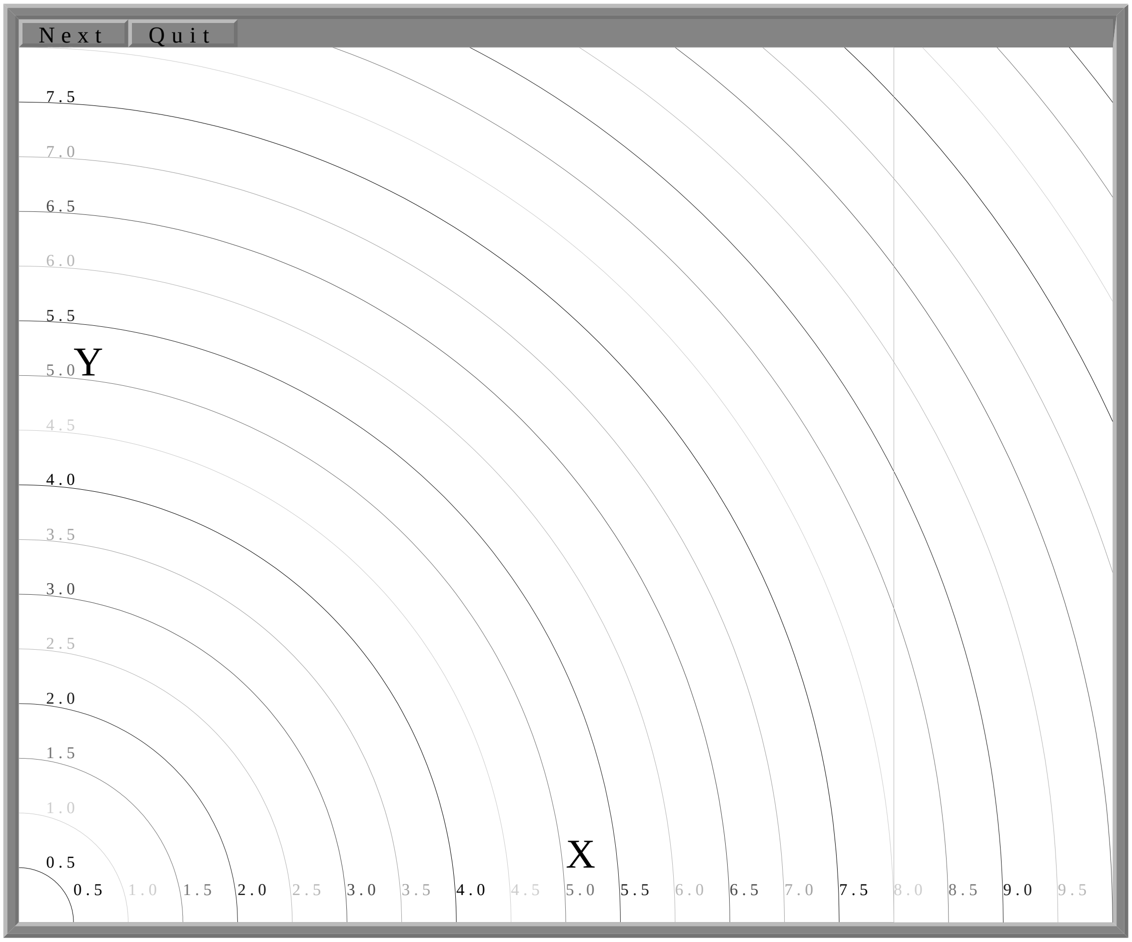
`-h`

`-?`

Online help

The X11 screen is viewed as a piece of paper exactly 10.0" wide and 8.0" high (

approximately 25.4 cm wide by 20.32 cm high). The default screen has dimensions of 800 x 640 pixels, which can be changed through the window manager when the program begins. The following screen would appear:



When the page is completely drawn, a cursor will appear. One can use this to point to a feature of interest. The following actions can be performed:

- Pressing and releasing the *Left Mouse Button* will advance the page.
- Pressing the menu button *Next* will advance the page.
- Pressing the menu button *Quit* terminates the plot.

To be consistent with X11, the geometry of the plot window can be specified by an entry in the .Xdefaults file:

```
plotxvig.calxvig.plotxvig.geometry: 1000x800+100+50
```

In order of importance, an entry such as this overrides a command line or environment option. For the other options, the command line overrides the PLOTXVIG environment control.

Problems:

Since X11 programming is a new experience, here are some annoyances.

Resizing a window after plotting begins will truncate the plot, if the window is smaller, or will have unused areas. Because of the size of the binary plotfiles, there is no way to rewind and redraw a plot. Instead a backup image is used.

Finally, **`plotxvig`** works by setting up two UNIX processes: one to do do the drawing, the other to handle events and to place the drawing on the screen - an interesting use of interprocess communication. You may find yourself with a display that is not responsive - this usually happens because one, but not both processes have terminated. Use **ps** to list the processes, and then **kill `PID`** to get rid of **calxvig** and **plotxvig**.

6. PNG Output

This is a CALPLOT driver to create Portable Network Graphics (PNG) files. This CALPLOT driver was built upon the sample programs wpng.c and *writepng.c* available at *http://www.libpng.org*. This driver supports 256 unique colors in its palette which is more than the 103 colors available in CALPLOT.

Program control is through the command line:

`plotpng` *[flags] < CALPLOT_FILE > FILE.png*

`-V`
Program Version

`-S`*scalefac*
(default=1.0) Plot magnifier

`-R`
(default off) Rotate plot 90 degrees

`-N`*num*
(default 1) Convery page num

`-I`
(default off) Invert the background. The background will be black instead of white. This is done by interchanging the black and white color map entries

`-F`*font*
(default 0)
0 Roman
1 Roman
2 Italic
3 Bold
4 Symbol (Greek)

`-X`*numx*
(default 640) X-pixels one of 640,800,400,1000,2000

`-Y`*numy*
(default 480)

Y-pixels one of 480,600,320,800,1600

-K
(default color) Color output

-KW
(default -K) Color output, whitened spectrum

-KR
(default -K) Color output Red->White->Blue

-KB
(default -K) Color output Blue->White->Red

-G
(default -K) Gray output

-C*colors*
(default 256) Size of Colormap 2, 4, 16 or 256

-h
Do not execute, show options

-?
Do not execute, show options

The PNG image is viewed as a piece of paper exactly 10.0" wide and 8.0" high (approximately 25.4 cm wide by 20.32 cm high). The default image has dimensions of 640 x 480 pixels. To maintain the design aspect ratio, choose the dumension combinations 320 x 200 640 x 480, 800 x 600, 1000 x 800 or 2000 x 1600.

The program output is to *stdin* and consists of the binary PNG graphics file. It is necessary to redirect the output to a file. It is also necessary to ensure that the file name terminates with a *.png* so that other programs can recognize the file type. You can view the PNG file using a recent Web Broswer or some graphics manipulation program. An example of a command would be

```
plotpng -C16 -K -X800 -Y640 < GRAYSC.PLT > graysc.png
```

7. Figure Manipulation

The program **reframe** permits manipulation of a CALPLOT figure, either by changing the position on the page, by imposing a primitive clipping. Options exist to select one figure of a multipage plot file, and to merge plot files. The output of this program is another plot file. The program imput is from the last argument on the command line, if that argument is not a command flag, or the standard input

Program control is through the command line:

reframe *[flags]*, where the command flags are

-O
Redirect the output to the standard output. Otherwise a **plotXXXXX** file will be created, where **XXXXX** is a unique identification number.

`-P`

Force the output to be a plot file. This the default.

`-M`*mergefile*

This is the file that will be superimposed onto the original file.

`-XL`*x_low_clip* (default = -100000000)

`-XH`*x_high_clip* (default = 100000000)

`-YL`*y_low_clip* (default = -100000000)

`-YH`*y_high_clip* (default = 100000000)

A selected position of the input figure can be passed through to the output. The selected region is bounded by these coordinates.

`-X0`*x_origin*

`-Y0`*y_origin*

These values are added to the (x,y) coordinates of all input values within the clipping window to shift the resulting figure on the page.

The sequence of operations is that first the image is clipped, and then the origin is shifted.

To illustrate the usage of the program, consider the following two examples:

To merge the second frames of two plot files, one need only do

```
reframe -N2 -O -MPLOTrhwvint < PLOTrefplt > PLOTrefplt2
```

Note that the page number flag applies to both input files. It may be necessary to run the program three times to select the desired pages, first two runs, and then to merge the output using the temporary files.

To select the first three pages of a multipage plotfile, and then to combine them on to a single page,

```
reframe -V  -XH8500 -YH11000  -N1 -O < TABL > hunk1
```

(retrieve page 1 and save in the file hunk1)

```
reframe -V  -XH8500 -YH11000  -X0+8750  -N2 -O < TABL > hunk2
```

(retrieve page 2, move plot 8750 units to right and save in the file hunk2)

```
reframe -V  -XH8500 -YH11000  -X0+17500 -N3 -O < TABL > hunk3
```

(retrieve page 3, move plot 11000 units to right and save in the file hunk3)

```
reframe -V -N1 -O -Mhunk2 hunk1 > munk1
```

(merge the files hunk2 and hunk1 into the file munk1)

```
reframe -V -X0+1000 -Y0+1000 -N1 -O -Mhunk3 munk1 > PLOTreframe2
```

(merge files hunk3 and munk1, and shift the origin 1000 units to the right and upward)

All units are in the device independent plot units. When the CALPLOT programs are used, 1000 units correspond to 1.000 inches on the hardcopy plot.

The results of another example are shown in Figures 3 and 4. The object is to cut Figure 3 into four quadrants centered at (4.0,4.0) and to exchange the upper right with the lower left quadrant and the upper left with the lower right quadrant. The commands used are as follow:

```
reframe -N1 -O < PLTTST > p
reframe -N1 -O -X0+4000 -Y0+4000 -XL0000 -XH4000 -YL0000 -YH4000 < p1 > g1
reframe -N1 -O -X0-4000 -Y0+4000 -XL4000 -XH8000 -YL0000 -YH4000 < p1 > g2
reframe -N1 -O -X0+4000 -Y0-4000 -XL0000 -XH4000 -YL4000 -YH8000 < p1 > g3
reframe -N1 -O -X0-4000 -Y0-4000 -XL4000 -XH8000 -YL4000 -YH8000 < p1 > g4
reframe -N1 -O -Mg1 < g2 > g5
reframe -N1 -O -Mg3 < g4 > g6
reframe -N1 -O -Mg5 -X0+1000 -Y0+1000 < g6 > g7
plotnps -F7 -W10 -G -EPS < g7 > g7.eps
rm p1 g1 g2 g3 g4 g5 g6 g7
```

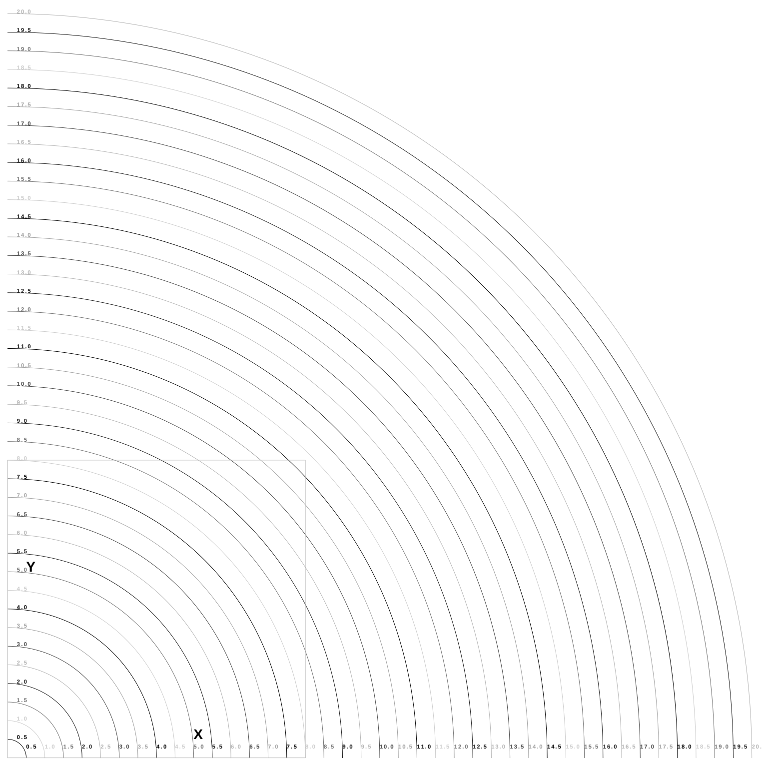


Fig. 3. Initial plot to be sectioned

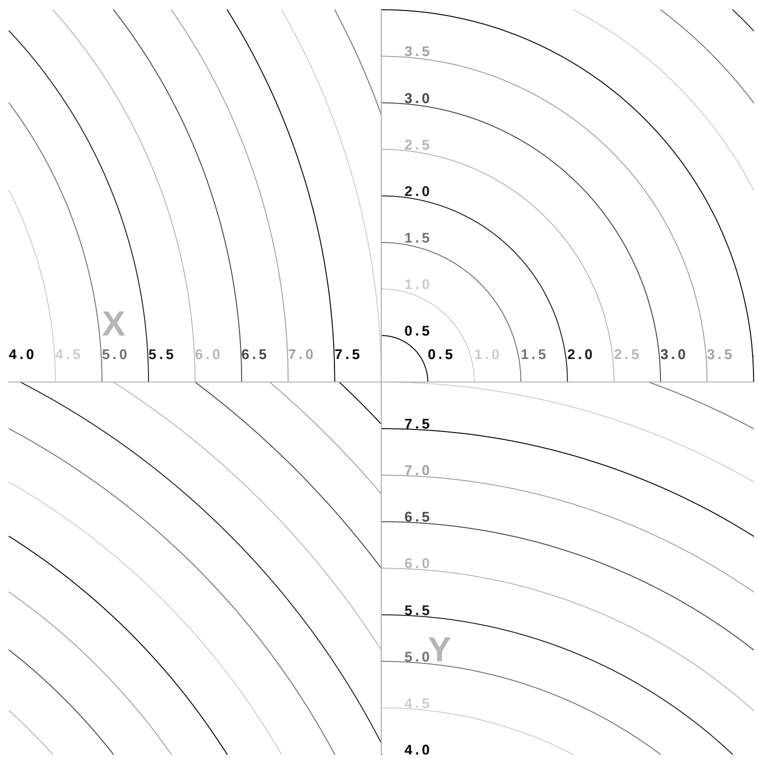


Fig. 4. Result of clipping and shifting

8. CALPLOT Colors

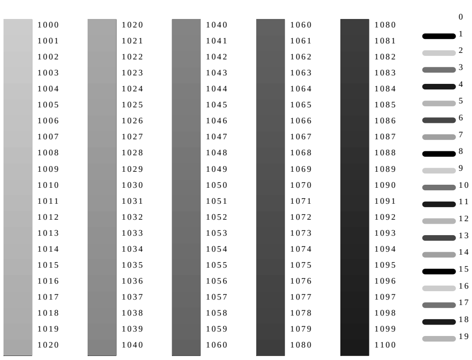
Many programs permit the user definition of colors for curves. These are invoked using

-K `kolor`

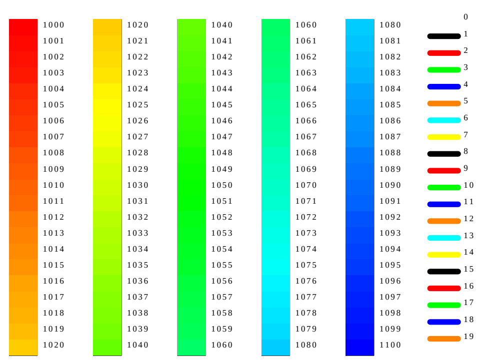
The CALPLOT graphics uses a set of predefined colors that take on slightly different meanings depending upon whether the plot program (**`plotxvig, plotmsw, plotnps, plotgif`**), is invoked with the **`-G`**, **`-K`**, **`-KR`** or **`-KB`** flags. Values of `kolor` between **`0`** and **`999`** are mapped into a specified sequence of 7 colors. Values in the range **`1000`** - **`1100`** select a palette of continuous color tones selected by the use of these flags. The table below defines some of these values as do the figures, which are best viewed on a color terminal screen using GhostView or Acroread.

Kolor	-G	-K	-KR	-KB
0	Background	Background	Background	Background
1	Foreground	Foreground	Foreground	Foreground
2	(see below)	Red	(see below)	(see below)
3		Green		
4		Blue		
5		Orange		
6		Blue-Green		
7		Yellow		
8	Foreground	Foreground	Foreground	Foreground
9		Red		
999				
1000	Lt. Gray	Red	Red	Blue
1025		Orange	Light Red	Light Blue
1050	Med.Gray	Green	White	White
1075		BlueGreen	Light Blue	Light Red
1100	Dk. Gray	Blue	Blue	Red

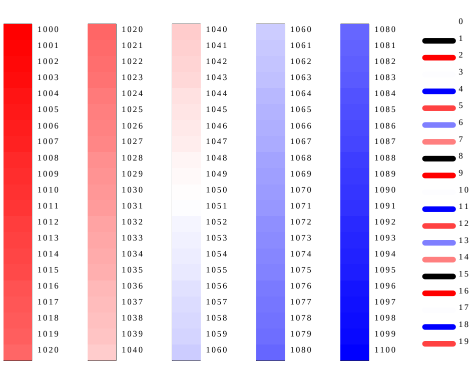
Normal plotting uses the **`-G`** and **`-K`** flags. Displays of continuous color maps can use the **`-KR`** and **`-KB`** modes if the color indices are programed to represent a range of negative - positive values with white representing a median value. The following figures show the resulting colors for a given choice of the `kolor` index.



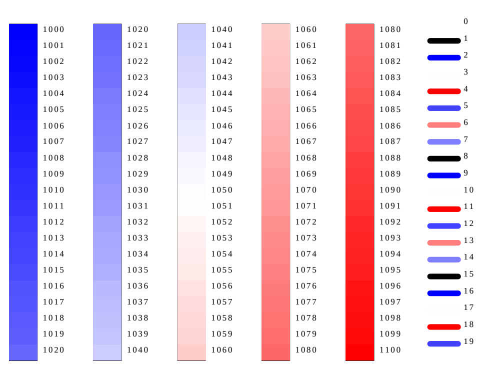
plotnps -G < GRAYSC



plotnps -K < GRAYSC



plotnps -KR < GRAYSC



plotnps -KB < GRAYSC

APPENDIX B
GSAC COMMANDS

1. Introduction

This section prints the on-line help text available under *gsac*. The documentation is not complete but is current.

2. GSAC Commands

HELP

SUMMARY:

List syntax for GSAC command

Help [command]

where command is one or more of the following:

ABS		Get absolute value of trace
AGC		AGC traces
ADD		Add constant to each trace
ADDF		Add Files in memory
BACKGROUND		Set trace background color
BD	BEGINDEVICES	Begin graphics
BG	BEGINGRAPHICS	Begin graphics
BOXCAR		Convolve with unit area boxcar
BP	BANDPASS	Bandpass filter
BR	BANDREJ	Bandreject filter
CD		change working directory
CH	CHNHDR	Change header
COLOR		Control plotting colors
CON	CONVOLVE	Convolve traces

COR	CORRELATE	Correlate traces
CUT		Cut trace as it is read
CUTERR		Control CUT
DEC	DECIMATE	Decimate trace
DIF		Differentiate traces numerically
DIV		Divide each trace by constant
DIVF		Divide Files in memory
ECHO		Echo a line of text to the terminal
ENV	ENVELOPE	Get trace envelope
EXP		Exponentiate traces
EXP10		Raise traces to power of 10
FFT	DFT	Take Fourier Transform
FG	FUNCGEN	Generate an impulse function
FILEID		Controls the plot legend
FILT	FILTER	Apply or remove instrument response
GRID		Control plot grid for x[en] and y[en]axes
H	HELP	Print this help list
HILBERT		Get Hilbert transform of a trace
HIS	HISTORY	Display readline command history
HOLD		Permit plot overlay in current frame
HP	HIGHPASS	Highpass filter
INTERP	INTERPOLATE	Resample traces
INT		Integrate traces
LH	LISTHDR	List header
LINLIN		lin[en]lin plot for plot, plotpk
LINLOG		lin[en]log plot for plot, plotpk
LOG		Take natural logarithm of trace
LOG10		Take base 10 logarithm of trace
LP	LOWPASS	Lowpass filter
MAP		Produce GMT map
MARKT	MARKTIMES	Marks velocity times in plot
MT	MOMENTTENSOR	Generate seismogram for moment tensor
MUL		Multiply each trace by constant
MULF		Multiply Files in memory
OUTCSV		Output timeseries as spreadsheet CSV
P1	PLOT1	Plot trace (all part of PLOT)
P	PLOT	Plot trace
P2	PLOT2	Plot trace
PAUSE		Pause a specified number of seconds
PCTL		Control time[en]domain plots
PPK	PLOTPK	Interactive trace plot
PSP	PLOTSP	Plot trace spectra
PRS	PLOTRECORDSECTION	Plot record section
QDP		Decimate factor for screen plot
Q	QUIT	Exit GSAC
R	READ	Read SAC files

REFR	REFRACTION	Enter Refraction Mode
REV	REVERSE	Reverse Time Series
RH	READHDR	Read header
RMEAN		Remove mean
ROT	ROTATE	Rotate horizontal components
ROT3	ROTATE3	Rotate three components to form R T Z
RTR	RTREND	Remove trend
SGN		1BIT digitization
SHIFT		Shift trace in time
SM	SMOOTH	Apply a smoothing operator
SORT		Sort traces
SQR		Square traces
SQRT		Square root of traces
STACK		Stack traces
SUB		Subtract constant from each trace
SUBF		Subtract Files in memory
SYNC	SYNCHRONIZE	Synchronize trace timing
TAPER		Apply a symmetric taper to traces
TITLE		Plot title
TRANS	TRANSFER	Apply or remove instrument response
TRAPEZOID		Convolve with unit area trapezoid
TRIANGLE		Convolve with unit area triangle
VERSION		Print version number
W	WRITE	Read SAC files
RI	RICKER	RICKER
WSP	WRITESPEC	Write spectra
WH	WRITEHDR	Write header
WHITEN		Whiten signal
XGRID		Control x[en]axis grid
XLIM		Set time axis limits for trace plot
XLIN		Linear x[en]axis for plot, plotpk
XLOG		Logarithmic x[en]axis for plot, plotpk
YGRID		Control y[en]axis grid
YLIM		Set y–axis scaling for plot1
YLIN		Linear y[en]axis for plot, plotpk
YLOG		Logarithmic y[en]axis for plot, plotpk

ABS

SUMMARY:

Take the absolute value of each data point

ABS

INPUT:

DESCRIPTION:

This determines the absolute value of each point of the trace.

SEE ALSO:

SQR, SQRT

ADD

SUMMARY:

Add a constant to all SAC data files in memory.

ADD [v]

INPUT:

[v] : constant to be added to all files

SAC COMPATIBILITY:

SAC permits an extended syntax that permits applying different constants to respective files in memory. We have not implemented this complexity.

DEFAULT

ADD 0

SEE ALSO

SUB, MUL, DIV

ADDF

SUMMARY:

Add Files in memory

ADDF [Master n] [Suffix suffix] [Default]

INPUT:

Master : Trace uses as master trace. Default is 0, which is the first in memory.
Suffix siffix : The traces are renamed and the original traces in memory are overwritten to be of the form
[STA2][CMP2]_[STA1][CMP1].suffix. The default value ofthe suffix is '.add'

DESCRIPTION:

This adds the master trace to all traces in memory. After the addition operation, the files are named as follow: [STA2][CMP2]_[STA1][CMP1].suffix Beware that nothing is done to the original header other than to reset the start and end times since the output trace is only for the common overlapping absolute time window.

HEADER CHANGES

DEPMAX, DEPMIN, DEPMEN, NPTS, B, O

EXAMPLES:

DEFAULT:

ADDF MASTER 0 Suffix .add

SEE ALSO

SUBF, MULF, DIVF

AGC

SUMMARY:

AGC traces

AGC W window

INPUT:

W window : Define the averaging window in seconds

DESCRIPTION:

This routine applies an AGC operator to the trace such that the mean signal amplitude is near unity. The operator is obtained by using a running average of the absolute value of the trace.

EXAMPLES:

HEADER VALUES SET:

DEPMAX, DEPMIN and DEPMEN are updated.

SEE ALSO

BACKGROUND

SUMMARY:

Set trace background color

BACKGROUND [ON | OFF] [Color color_int] [DEFAULT]

INPUT:

ON : turn background color on
OFF : turn background color off
Color Color_int : set the color [en] default value = 0; An integer between 0 and 1100. 0 is always background of screen, 1 is the screen foreground. for color displays 2 = Red, 3= Green, 4 = Blue etc. 1000 to 1100 is a transition from red to blue.

DESCRIPTION:

Set the background display for the plots. Note this does not change the fram background.

EXAMPLES:

DEFAULT

BACKGROUND OFF

SEE ALSO

BANDPASS

SUMMARY:

BandPass filter traces

BandPass [options]

where options is one or more of the following:

[Butter | BEssel | C1] [Corner fc] [Npoles npoles] [Passes npass] [Tranbw tranbw] [Atten atten]

INPUT:

Butter : Butterworth filter (default)
BEssel : Bessel filter
C1 : Chebyshev Type I filter
Corner : Corner frequencies (R) range 0 – Nyquist
Npoles : Number of poles (I) range 1 – 10
Passes : Number of passes (I) range 1 – 2
Tranbw : Chebyshev transition bandwidth fraction (0.3 default)

Atten : Chebyshev stop band attenuation (30 default)

DESCRIPTION:

Highpass filter using a BI–LINEAR Z–transformation implementation of a highpass filter. A bi–linear method is chosen since this is easily implemented algebraically. Passes = 1 gives a causal filter while Passes = 2 gives a zero–phase filter with a 6db point at the corner frequency.

The lowpass filter design of the Chebyshev Type 1 filter is based on the information in Hamming (1997) equation (13.5.4) and Figure 13.6.1, This filter attempts to approximate a sharp lowpass filter. In reality a transition band is defined by Fp and Fs, where Fp is the lowpass corner frequency and Fs is the stopband, where Fs = (1 + tranbw)*Fp. The amplitude level of the stop band (f > Fs) is 1/atten. Actual implementation requires a parameter epsilon, eps, which defines the lowpass ripple, which varies between 1 and 1/sqrt(1 + eps*eps).

Hamming, Richard W. (1997). Digital Filters (3rd edition), Dover Publications, 296 pp, ISBN 048665088X

Given npoles and eps, the poles and zeros are given by

http://www.answers.com/topic/chebyshev[en]filter

The correct normalization amplitude together with poles and zeros is given in

Digital Filter Designers Handbook with C++ algorithms, C. Britton Rorabaugh 2nd Edition, McGraw Hill, New York, 479 pp 1997 Chapter 5

EXAMPLES:

Bandpass with corner frequencies at 1 and 10 Hz, zero phase, 2–pole

BP BUTTER C 1 10 P 2 NP 2

HEADER VALUES SET

USER1 = permin, USER2=permax, where permin=1.0/(filt_fh), and permax= 1.0/(filt_fl) for use by sacmft96 adn sacpom96

SEE ALSO

LOWPASS, HIGHPASS, BANDREJECT

BANDREJECT

SUMMARY:

BandReject filter traces

BandReject [options]

where options is one or more of the following:

[Butter | BEssel | C1] [Corner fc] [Npoles npoles] [Passes npass] [Tranbw tranbw] [Atten atten]

INPUT:

Butter : Butterworth filter
BEssel : Bessel filter
C1 : Chebyshev Type I filter
Corner : Corner frequencies (R) range 0 – Nyquist
Npoles : Number of poles (I) range 1 – 10
Passes : Number of passes (I) range 1 – 2
Tranbw : Chebyshev transition bandwidth fraction (0.3 default)
Atten : Chebyshev stop band attenuation (30 default)

DESCRIPTION:

Highpass filter using a BI–LINEAR Z–transformation implementation of a highpass filter. A bi–linear method is chosen since this is easily implemented algebraically. Passes = 1 gives a causal filter while Passes = 2 gives a zero–phase filter with a 6db point at the corner frequency.

The lowpass filter design of the Chebyshev Type 1 filter is based on the information in Hamming (1997) equation (13.5.4) and Figure 13.6.1, This filter attempts to approximate a sharp lowpass filter. In reality a transition band is defined by Fp and Fs, where Fp is the lowpass corner frequency and Fs is the stopband, where Fs = (1 + tranbw)*Fp. The amplitude level of the stop band (f > Fs) is 1/atten. Actual implementation requires a parameter epsilon, eps, which defines the lowpass ripple, which varies between 1 and 1/sqrt(1 + eps*eps).

Hamming, Richard W. (1997). Digital Filters (3rd edition), Dover Publications, 296 pp, ISBN 048665088X

Given npoles and eps, the poles and zeros are given by
http://www.answers.com/topic/chebyshev[en]filter

The correct normalization amplitude together with poles and zeros is given in
Digital Filter Designers Handbook with C++ algorithms, C. Britton Rorabaugh 2nd Edition, McGraw Hill, New York, 479 pp 1997 Chapter 5

EXAMPLES:

Bandreject with corner frequencies at 1 and 10 Hz, zero phase, 2–pole
BR BUTTER C 1 10 P 2 NP 2

SEE ALSO

LOWPASS, HIGHPASS, BANDPASS

BEGINGRAPHICS

SUMMARY:

Initialize the graphics device

BeginGraphics [options] BeginDevices [options]

where options is one or more of the following:

[X | W | Plt] [GEOM width height] [REVERSE] [GRAY] [COLOR]

INPUT:

X or W : Interactive windows display
Plt : Generate external Calplot PLT file
[These are only useful if nothing has been plotted on screen in window mode]
GEOM height width : set screen window height and width
GRAY : use gray scale insteadk of color
COLOR : use color scale insteadk of gray
REVERSE : invert the foreground and background

DESCRIPTION:

This initializes the graphics. For an interactive display under X11 or Windows, the graphics display is initialized at the time of the first plot command (Plot Plot1 PlotPK).

Invocation of the PLT option causes the creation of a CALPLOT plot file of the form Pxxxx.PLT, where the xxxx is a unique number incremented each time the BG PLT is invoked

For windows plot, the screen geometry and color map is defined by the external environment parameter PLOTXVIG. One may define this variable or override the shell version through this command. However, because of low level problems of color maps and malloc s, the window definition can only per performed ONCE BEFORE an actual screen plot is made.

SAC COMPATIBILITY:

SUNWINDOWS SGF and Tektronix are not supported. A separate Pnnn.PLT CALPLOT binary file is created for each plot made. P0001.PLT is always the first plot file. To convert to PostScript use plotnps < Pxxxx.PLT ; to convert to an EPS file, use

cat Pxxxx.PLT | reframe –Npage_number –O | plotnps –EPS –F7 –W10 > epsfile

SEE ALSO

BOXCAR

SUMMARY:

Convolve with unit area boxcar

BOXCAR Width width

INPUT:

Width width : the length of the boxcar function rounded to the next sample interval.

DESCRIPTION:

his routine convolves all traces in memory with a unit area isoceles trianglar pulse. This command is equivalent to

TRAPEZOID WIDTH 0.0 width 0.0

This acts as a lowpass filter.

If Width < DELTA, nothing is done.

EXAMPLES:

SEE ALSO

TRAPEZOID, TRIANGLE, RICKER

CD

SUMMARY:

Change the current working directory

CD path

INPUT:

path : path to the next directory

DESCRIPTION:

This changes the working directory. If there are already traces in memory and one decides to do a write, the operation may not work if the file path is no longer valid from the new directory

SEE ALSO

CHANGEHEADER

SUMMARY:

Change header values in memory

ChangeHeader [name value]

INPUT:

name : SAC header name
value : New value for that header variable

DESCRIPTION:

This commands lets one change many of the header values associated with a trace. If more than one trace is in memory, then all header variables are updated. The following listing gives the header names that can be changed:

Reals: DELTA, DEPMIN, DEPMAX, SCALE, ODELTA, B, E, O, A, FMT, T0, T1, T2, T3, T4, T5, T6, T7, T8, T9, F, RESP0, RESP1, RESP2, RESP3, RESP4, RESP5, RESP6, RESP7, RESP8, RESP9, STLA, STLO, STEL, STDP, EVLA, EVLO, EVEL, EVDP, USER0, USER1, USER2, USER3, USER4, USER5, USER6, USER7, USER8, USER9, DIST, AZ, BAZ, GCARC, SB, SDELTA, DEPMEN, CMPAZ, CMPINC, XMINIMUM, XMAXIMUM, YMINIMUM, YMAXIMUM, ADJTM

Integers: NZYEAR, NZJDAY, NZHOUR, NZMIN, NZSEC, NZMSEC, N NINF, NHST, NPTS, NSNPTS, NSN, NXSIZE, NYSIZE, IFTYPE, IDEP, IZTYPE, IINST, ISTREG, IEVREG, IEVTYP, IQUAL, ISYNTH, IDHR12, IDHR13, IDHR14, IDHR15, IDHR16, IDHR17, IDHR18, IDHR19, IDHR20, LPSPOL, LOVROK, LCALDA,

Strings: KSTNM, KEVNM, KHOLE, KO, KA, KT0, KT1, KT2, KT3, KT4, KT5, KT6, KT7, KT8, KT9, KF, KUSER0, KUSER1, KUSER2, KCMPNM, KNETWK, KDATRD, KINST

NOTE THAT ANY ATTEMPT TO CHANGE NVHDR or NPTS IS IGNORED

Because SAC permits something like

ch o 10

ch o gmt 2004 123 10 20 30 500

The command parsing is complicated. GSAC handles this by internally concatenating the o and gmt to form the new GSAC ogmt. GMT introduces the following time commands:

Setting GMT Time: OGMT, AGMT, T0GMT, T1GMT, T2GMT, T3GMT, T4GMT, T5GMT, T6GMT, T7GMT, T8GMT, T9GMT

Also since it is often inconvenient to determine the day of the year, GSAC permits entries of the form

ch OCAL 2004 05 02 10 20 30 500

ch o cal 2004 05 02 10 20 30 500

Setting Calendar Time: OCAL, ACAL, T0CAL, T1CAL, T2CAL, T3CAL, T4CAL, T5CAL, T6CAL, T7CAL, T8CAL, T9CAL

NOTE You can assign values to USER1 and USER2 if followed by a WRITEHEADER. However if you follow by a WRITE, then the USER1 and USER2 fileds will be the minimum and maximum filter periods. THIS WAS A DESIGN DECISION.

SEE ALSO

WriteHeader, ListHeader

COLOR

SUMMARY:

Controls color of trace plots

COLOR [ON|OFf] [options] where options is on of the following:
DEFAULT RAINBOW LIST list_entries

INPUT:

ON : Enable color output
OFf : Disable color output – traces will be CALPLOT foreground
DEFAULT : Use the default color sequence
LIST : A user specificed list for the alternating colors is used which are one of the following keywords: BLAck, REd, Green BLUe, ORange, Cyan, Yellow
RAINBOW : A selection of colors from RED to BLUE (or light to dark in gray scale), with colors chosen according to the trace number. Two traces will plot as RED and BLUE.

DESCRIPTION:

This permits different traces to be plotted with different colors. color BLACK is actually the foreground color. By default this is BLACK in CALPLOT, but if the interactive disply environment is set by the SHELL command ’export PLOTXVIG=:–K:–I:’ under the ’bash shell’, the background will be black and the foreground white.

The RAINBOW option indicates that the traces will be plotted in a progression from RED to BLUE, as in a rainbow.

SEE ALSO

CONVOLVE

SUMMARY:

Convolve traces

CONvolve Zero [B|O|A|T0|T1|T2|T3|T4|T5|T6|T7|T8|T9] File external_sac_file [Suffix suffix]

INPUT:

Zero : Specify marker to define the zero time for the external pulse. Since the external pulse may have been delayed, one marker must be set. The default is the O (oh) marker
File FILE : Name of external file containing the impulse response of the filter
Suffix suffix : Suffix to be appended to the file name in memory after convolution.The default value ofthe suffix is '.con' After the convolution operation, the files are named as follow: original_name{suffix}, e.g., SLMNMBHZ.con.

DESCRIPTION:

All traces in memory are convolved with the external trace specificed on the command line. with the master trace is computed. If the trace in memory is called x, and the external trace is called h, the convolution is defined as

INT x(tau) h(t [en] tau) dtau

where dtau is the sample interval.

The input trace is linearly interpolated to match the sampling of the traces in memory. The convolution is implemented in the time domain to avoid discrete Fourier transform (DFT) wrap[en]around problems.

The zero lag time for the external trace, h(t), is specified on the command lineas one of the B, O, A, T0, ..., T9 markers, which must be set in the external file. Otherwise the convolution will not occur.

Note the only the DEPMAX, DEPMIN and DEPMEN header values are changed for the traces in memory. The zero marker from the external trace is not written on the the headers of the traces in memory.

DEFAULT

Align O Suffix .con

SEE ALSO

CORRELATE

SUMMARY:

Compute auto[en] and cross correlation of traces

CORrelate [Master n] [Number n] [Length ON|OFF|Window] [Default] [Suffix suffix] [Reverse] [2]

INPUT:

Master : Trace uses as master trace. Default if first in memory

Number : Number of correlation windows to be used. The original trace is split into Number segments; the cross[en]correlation is determined for each segment and then summed.

Length : The original time series is broken into segments of Window seconds; the cross[en]correlation is determined for each segment and then summed.

Suffix suffix : The traces are renamed and the original traces in memory are overwritten to be of the form [STA2][CMP2]_[STA1][CMP1].suffix. The default value ofthe suffix is '.cor'

2 : Use a double length time series to prevent FFT periodicity. Output length is not changed

Reverse : Instead of computing INT x(tau)y(t + tau) dtau compute INT y(tau)x(t + tau) dtau, where x is the master trace.

DESCRIPTION:

A cross[en]correlation of all traces in memory with the master trace is computed. If the master (first) trace is called x, and the other trace is called y, the cross[en]correlation is defined as

INT x(tau)y(t + tau) dtau

When the Number or Length options are used, the original time series are cut into segments of equal length. The correlations are computed and then stacked. The average of the stack is then output.

After the correlation operation, the files are named as follow: [STA1][CMP1]_[STA2][CMP2]. The EVLA and EVLO are that of STA1 and the STLA and STLO are those of STA2.

CAUTION

Note that correlation is done in absolute time. If the x(t) and y(t) have vastly different reference times, the values of the B and E may not be correct because of the problem of stuffing a double into a float.

HEADER CHANGES:

DEPMIN, DEPMAX, DEPMEN, EVLA, EVLO, STLA, STLO, DIST, GCARC. The time markers O, B and E are set. The others are reset to [en]12345. T9 is set to the offset time of the maximum of the cross[en]correlation

DEFAULT:

MASTER 1 NUMBER 1 LENGTH OFF

SEE ALSO:

CUT

SUMMARY:

Cut a trace as it is read.

CUT [ON|OFF] [ref offset | GMT beg | CAL beg] [ref offset | GMT end | CAL end]

INPUT:

ON : Permit cutting on read
OFF : Turn off cutting on read
ref : A header reference value for the cut which is one of B|E|O|A|Tn where n=0,...,9
offset : Number of seconds relative to the reference value. refbeg offset refers to the start point refend offset refers to the end point
CAL : Calendar time in YEAR MONTH DAY HOUR MINUTE SECOND MILLISECOND
GMT : GMT time in YEAR DAYOFYEAR HOUR MINUTE SECOND MILLISECOND

DESCRIPTION:

This routine cuts a trace on the next READ according to specified header values. This is useful for focusing on one part of the trace.

The option for CAL or GMT times was introduced 11 JAN 2005 to permit selection of time windows from very long time segments. The following are equivalent:
CUT GMT 2005 001 01 02 03 456 GMT 2005 032 06 05 04 321
CUT GMT 2005 001 01 02 03 456 CAL 2005 02 01 06 05 04 321
CUT CAL 2005 01 01 01 02 03 456 GMT 2005 032 06 05 04 321
CUT CAL 2005 01 01 02 03 456 CAL 2005 02 01 06 05 04 321
which cuts from January 1, 2005 01:02:03.456 to February 1, 2006 06:05:04.321

SAC COMPATIBILITY:

SEE ALSO

CUTERR

Note

The CUTERR FILLZ is done by default

CUTERR

SUMMARY:

INPUT:

DESCRIPTION:

SAC COMPATIBILITY:

SEE ALSO

DECIMATE

SUMMARY:

Decimate trace by an integer

DECimate [n]

INPUT:

[n]   : decimation factor [en] an integer

DESCRIPTION:

This routine decimates a trace by outputting the first and then every n points. The resultant time series changes length.

NOTE THE DECIMATION FACTOR IS NEVER REMEMBERED [en] THE DEFAULT IS ALWAYS 1

SAC COMPATIBILITY:

Sac limits the decimation interval to 2 <= n <= 7. This is because Sac permits an antialiasing filter option which meant that those filter coefficients must be built into the program.

In the GSAC implementation, I suggest a zero ppahse low pass filter at a frequency 0.5 the Nyquist frequency, or 0.25/DELTA, perhaps a

    lp c corner_frequency n 4 p 2

EXAMPLES:

dec 2

HEADER VALUES SET:

NPTS, DELTA, B, E, DEPMIN, DEPMAX, DEPMEN

DEFAULT

decimate 1

DELETE

SUMMARY:

Delete a file in memory from any trace processing

DELETE trace_number

INPUT:

DESCRIPTION:

This is useful if one finds a bad trace in a large input and does not appear in any trace processing. It still stays in memory but will never be output.

SEE ALSO

UNDELETE

DIF

SUMMARY:

Differentiate all SAC data files in memory.

DIF

SAC COMPATIBILITY:

SAC permits a choice of TWO, THREE and FIVE point oeprators. A simple two–point rule is used here.

SEE ALSO

INT

DIV

SUMMARY:

Divide all SAC data files in memory by a constant.

DIV [v],

INPUT:

[v] : constant which all files are divided

SAC COMPATIBILITY:

SAC permits an extended syntax that permits applying different constants to respective files in memory. We have not implemented this complexity.

DEFAULT:

DIV 1

SEE ALSO

ADD, SUB, MUL

DIVF

SUMMARY:

Multiply Files in memory

DIVF [Master n] [Suffix suffix] [Water water_level] [Default]

INPUT:

Master : Trace uses as master trace. Default is 0, which is the first in memory.
Suffix siffix : The traces are renamed and the original traces in memory are overwritten to be of the form
[STA2][CMP2]_[STA1][CMP1].suffix. The default value ofthe suffix is '.div' ,BR Water water_level : By default this is 0.0001 of the maximum amplitude. This is used to avaoid dividing by zero. Thus instead of X/Y, X/MAX(water_level*ymax,Y) is computed.

DESCRIPTION:

This divides all traces in memory by the master trace. After the division operation the files are named as follow: [STA2][CMP2]_[STA1][CMP1].suffix Beware that nothing is done to the original header other than to reset the start and end times since the output trace is only for the common overlapping absolute time window.

The result on the master trace is to square it, and perhaps to change the begin and end of the trace.

This is designed to permit computation of spectral ratios, using the amplitude spectra generated using writesp.

HEADER CHANGES

DEPMAX, DEPMIN, DEPMEN, NPTS, B, O

EXAMPLES:

DEFAULT:

DIVF MASTER 0 Suffix .div

SEE ALSO

ADDF, SUBF, MULF

ECHO

SUMMARY:

Echo a line of text to the terminal

ECHO [message]

INPUT:

message : text to be output to terminal.

DESCRIPTION:

EXAMPLES:

SEE ALSO

PAUSE

ENVELOPE

SUMMARY:

Get the envolope of a trace.

ENVELOPE

INPUT:

DESCRIPTION:

This determines the envelope by first computing the Hilbert transform, q(t), of the trace, h(t), and then performing the operation sqrt[h(t)*h(t) + q(t)*q(t)].

SEE ALSO

ENVELOPE

EXP

SUMMARY:

Exponentiate trace.

EXP

INPUT:

DESCRIPTION:

This exponentiates a trace. However NOTHING will be done if the trace value is greater than 85 since this would yield a real number > 1.0E+37.

SEE ALSO

SQRT, SQR, LOG, ABS

EXP10

SUMMARY:

Raise trace to power of 10 .

EXP10

INPUT:

DESCRIPTION:

This exponentiates a trace raising it to a power of 10. However NOTHING will be done if the trace value is greater than 37 since this would yield a real number > 1.0E+37.

SEE ALSO

SQRT, SQR, LOG, ABS

FFT

SUMMARY:

Obtain the discrete Fourier transform of traces in memory

FFT [Length 1|2|4|8] [Default]

INPUT

Length Nvalue [en] pad the time series with zeros to increase the FFT by a factor or 1, 2, 4 or 8. The purpose of this is to ensure greater frequency resolution.

Default [en] resent Length to 1

Note that the use must ensure there is no offset between the time series and the appended zeros by some combination of RMEAN, RTR or TAPER. The default is Length = 1.

Note also that zeros are always added since the FFT requires a power of 2 length.

DESCRIPTION:

This estimates the discrete Fourier transform of the traces in by using a version of Brenner's original FOUR1 routine. The definition of the DFT is

$$H(n\, DF) = DT \sum_{k=0}^{N-1} h(k)\, e^{j \{2\, pi\, n\, k/N\}}$$

where DF = 1/(N DT), DT = sampling interval, DELTA, and N is a power of 2. Note that this is an extension to the definition of a DFT in that physical dimension are introduced

DEFAULT

FFT L 1

SEE ALSO

PLOTSP, WRITESP, RMEAN, RTR, TAPER

FILEID

SUMMARY:

Controls the plot legend

FILEID [ON|OFf] [TYpe DEfault|Name|LIst hdrlist] LOcation[UR|UL|LR|LL|UC|LC] Format [EQuals|COlons|NOnames]

INPUT:

ON : Turn on file id option. Does not change file id type or location.

OFf : Turn off file id option.

TYpe DEfault : Change to the default file label. This is KSTNM, KCMPNM

TYpe Name : Use the name of the file as the file label.

TYpe LIst hdrlist : Define a list of header fields to display in the fileid.

The header fields permitted are the exact words KZDATE, KZTIME, KCMPNM, KSTNM, KEVNM, GCARC, DIST, AZ, BAZ, STLA, STLO, STEL, EVLA, EVLO, EVDP, The programs permits up to ten (10) such labels. However, because of limited space the appearance will be messy.
LOcation UR : Place file id in upper right hand corner.
LOcation UL : Place file id in upper left hand corner.
LOcation LR : Place file id in lower right hand corner.
LOcation LL : Place file id in lower left hand corner.
Format EQuals : Format consists of header field name, an equals sign, and the header field value.
FOrmat COlon : Format consists of header field name, a colon, and the value.
FOrmat NOnames : Format consists of header field value only with no label.

DESCRIPTION:

DEFAULT

FILEID ON TYPE DEFAULT LOCATION UR FORMAT NONAMES

SEE ALSO

FILTER

SUMMARY:

Apply or remove an instrument response/filter from the data

FILTER [FROM|TO] [APPLY|REMOVE] [POLEZERO pzfile] [FAPFILE fapfile] [EVAL afile pfile]
[FREQLIMITS f1 f2 f3 f4] [NOFREQLIMITS]

INPUT:

APPLY or TO : The wave form is passed through the filter
REMOVE or FROM : The instrument filter is deconvolved from the trace
POLEZERO pzfile : Use the SAC Pole–zero format
EVAL afile pfile : Use the output of the IRIS evalresp which gives two files, one of gain(frequency) and the other phase(freq)
FAPFILE fapfile : Use the Frequency/amplitude/phase file
FREQLIMITS f1 f2 f3 f4i : Apply a cubic taper to the response such the

response is 0 for f < f1 and for f > f4, the response is 1 for f > f2 and for f < f3, and tapers cubically from 0 to 1 for f1 < f < f2 and f4 > f > f3. Note the only way to turn this off is to reset the limits as in FREQLIMITS –2 –1 1.0e5 1.0e6 FREQLIMITS is only used in the FROM process. This is essential for a clean deconvolution

DESCRIPTION:

SAC COMPATIBILITY:

This is similar in concept to the the SAC TRANSFER, which where the pazfile is the displacement sensitivity of the instrument. This command is suitable to the forward transform, but must not be used for a deconvolution that involves a second transformation, e.g., do not try

FILTER FROM POLEZERO pzfile FREQLIMITS f1 f2 f3 f4

FILTER TO VEL because this actually creates two time series and used the Fast Fourier transform to accomplish the filtering. Taking a derivative of the second series to yield velocity, may introduce a glitch at the last point This artifact could be eliminated if a frequency domain bilinear Z–transform differentiator were implemented. Instead use the syntax

TRANSFER FROM POLEZERO SUBTYPE pzfile FREQLIMITS f1 f2 f3 f4 TO VEL

NOFREQLIMITS is not in SAC

HEADER VALUES SET

USER1 = permin, USER2=permax, where
permin=MAX[1.0/filt_f3,old permin] and
permax=MIN[1./filt_f2,old permax]. This feature is used
by sacmft96 and sacpom96

SEE ALSO

TRANSFER

FUNCGEN

SUMMARY:

Generate a synthetic time series for testing.

FuncGen [Impulse | Triangle | Box] Delta delta Npts npts [Length length] [Comp ncomb delay]

INPUT:

Impulse : Generate a time series with a single point with amplitude equal to 1.0/delta, where delta is the sampling interval in seconds. The impulse is centered at npts/2, and the B header value is set as – (npts/2)*delta. The default output file is called impulse.sac This will have a unit Foureir amplitude spectrum.

Triangle : Generate a time series with three points with amplitudes equal to (0.25/deta, 0.50/delta, 0.25/delta) where delta is the sampling interval in seconds. The triangle is centered at npts/2, and the B header value is set as – (npts/2)*delta. The default output file is called triangle.sac This will have a unit spectral maplitude at zero frequency and a spectral zero at the Nyquist frequency, 0.5/delta.

Box : Create a boxcar with duration of 'length' seconds, starting at center of the trace. The minimum length is internally set to 10*delta!

Delta : Sample interval in seconds

Npts : Number of points in the time series

Comb : repeat the chosen pulse 'ncomb' times with a separation of 'delay' seconds. The total area of this function is 1.0. This is designed to create compulciated pulses.

DESCRIPTION:

This generates a synthetic time series. At present only the impulse and triangle pulses are supported. The internal file name is either 'impulse.sac' or 'triangle.sac'.

SAC COMPATIBILITY:

SAC creates an impulse to have unit amplitude int the time domain, which means that the spectral amplitude will be 'delta'. GSAC creates an impulse with unit spectral amplitude at zero frequency. This is done to permit an easy view of the instrument response:

```
funcgen impulse deltaa 0.05 npts 4096
transfer from none to polezero subtype resp.paz
fft
psp am
```

DEFAULT:

FUNCGEN IMPULSE DELTA 1.0 NPTS 1024

SEE ALSO

GRID

SUMMARY:

Control plot grid for x[en] and y[en]axes

GRID [ON | OFF] [Solid | Dotted] [Color int_value] [Minor ON | OFF]

INPUT:

ON : turn grid on
OFF : turn grid off
Solid : Use solid line
Dashed : Use dotted line
Minor : connect minor tics too if ON
Color int_value : Define the color for the grid. The figure frame will continue to be in black. Be careful to select a color not used for the trace.

DESCRIPTION:

This annotates the plots with a grid. Note that the Dotted option is takes more time to plot.

EXAMPLES:

DEFAULT

GRID OFF DOTTED COLOR 1030 MINOR OFF
[Note COLOR 1 is black, 2 red, 3 green, 4 blue, 1030 pale yellow]

SEE ALSO

XGRID, YGRID, COLOR

HIGHPASS

SUMMARY:

HighPass filter traces

HighPass [options]

where options is one or more of the following:

[Butter | BEssel | C1] [Corner fc] [Npoles npoles] [Passes npass] [Tranbw tranbw] [Atten atten]

INPUT:

Butter : Butterworth filter
BEssel : Bessel filter
C1 : Chebyshev Type I filter
Corner : Corner frequency (R) range 0 – Nyquist
Npoles : Number of poles (I) range 1 – 10
Passes : Number of passes (I) range 1 – 2
Tranbw : Chebyshev transition bandwidth fraction (0.3 default)
Atten : Chebyshev stop band attenuation (30 default)

DESCRIPTION:

Highpass filter using a BI–LINEAR Z–transformation implementation of a highpass filter. A bi–linear method is chosen since this is easily implemented algebraically. Passes = 1 gives a causal filter while Passes = 2 gives a zero–phase filter with a 6db point at the corner frequency.

The lowpass filter design of the Chebyshev Type 1 filter is based on the information in Hamming (1997) equation (13.5.4) and Figure 13.6.1, This filter attempts to approximate a sharp lowpass filter. In reality a transition band is defined by Fp and Fs, where Fp is the lowpass corner frequency and Fs is the stopband, where Fs = (1 + tranbw)*Fp. The amplitude level of the stop band (f > Fs) is 1/atten. Actual implementation requires a parameter epsilon, eps, which defines the lowpass ripple, which varies between 1 and 1/sqrt(1 + eps*eps).

Hamming, Richard W. (1997). Digital Filters (3rd edition), Dover Publications, 296 pp, ISBN 048665088X

Given npoles and eps, the poles and zeros are given by

http://www.answers.com/topic/chebyshev[en]filter

The correct normalization amplitude together with poles and zeros is given in

Digital Filter Designers Handbook with C++ algorithms, C. Britton Rorabaugh 2nd Edition, McGraw Hill, New York, 479 pp 1997 Chapter 5

HEADER VALUES SET

USER1 = permin, USER2=permax, where permin=1.0/(2*dt), and permax= 1.0/(filt_fl) for use by sacmft96 adn sacpom96

SEE ALSO

LOWPASS, BANDPASS, BANDREJECT

HILBERT

SUMMARY:

Obtain the Hilbert transform of a trace

HILBERT

DESCRIPTION:

The Hilbert transform of each trace is obtained. This is implemented using Discrete Fourier Transforms. Since the Hilbert transform of an impulse is infinately long, there may be a wrap around problem, which can be alleviated by using a longer tome series. By definition the Hilbert transform does not change the amplitude spectrum but does change the spectrum by pi/2 radians. A double invocation of Hilbert will yield an inverted version of the original trace.

SEE ALSO

HISTORY

SUMMARY:

Display readline command history

HISTORY [Default | List n]

INPUT:

List n : List the last n commands
Default : List all commands since GSAC started.

DESCRIPTION:

When compiled with the GNU readline library, a history of the commands entered is saved. One can use a combbination of the arrow keys and the users editor commands to move around the current command line and previous command lines. When the ENTER key is hit, the command line will be interpreted by GSAC.

Often it may be useful to review the history, and then cut and paste to repeat previous commands.

Unlike the history mechanism of the CSH or BASH shells, one cannot use the !n sequence to edit previous lines.

DEFAULT:

HISTORY DEFAULT

SEE ALSO

HOLD

SUMMARY:

Permit plot overlay in current frame

HOLD [ON | OFF]

INPUT:

ON : Set the HOLD for the next plot
OFF : Unset the HOLD option. The next instance of PRS PSP or P1 will start on a new page.

DESCRIPTION:

To permit an overlay of different graphics on a current plot frame, one must not perform an erase. HOLD is similar to the MATLAB HOLD command which has the same purpose.

EXAMPLES:

SEE ALSO

PCTL, P1, PSP, PRS

INT

SUMMARY:

Integrate all SAC data files in memory.

INT

SAC COMPATIBILITY:

SAC permits a choice of TRAPEZOIDAL or RECTANGULAR rules. A simple one–point running summation is used. The first point of the output is set to zero to avoid a linear trend in the output. DEFAULT

SEE ALSO

DIF

INTERPOLATE

SUMMARY:

Resample the current traces in memory

INTERPOLATE [Delta new_delta}

INPUT:

Delta new_delta : Define the new sample interval. This may be greater or less than the current DELTA of the trace.

DESCRIPTION:

When comparing observed and synthetics quantitatively, the traces sample intervals must be identical. This routine permits a resampling at an arbitrary new DELTA. The start time of the trace is maintained, but the number of points and end time are changed. When the new_delta > the original DELTA, it is best to lowpass filter the trace before interpolating

SEE ALSO

LINLIN

SUMMARY:

lin[en]lin plot for plot, plotpk

LINLIN

INPUT:

DESCRIPTION:

DEFAULT:

Linear x[en]axis and linear y[en]axis

SEE ALSO

XLIN, YLIN, XLOG, YLOG, LOGLIN, LOGLIN, LOGLOG

LINLOG

SUMMARY:

lin[en]log plot for plot, plotpk

LINLOG

INPUT:

DESCRIPTION:

DEFAULT:

Linear x[en]axis and linear y[en]axis

SEE ALSO

XLIN, YLIN, XLOG, YLOG, LINLIN, LOGLIN, LOGLOG

LISTHEADER

SUMMARY:

List trace header values

ListHeader [options] where options is one or more of the following:

[Default] [list]

INPUT:

Default : Output all defined header fields
list : a listing of header fields to output
Columns 1 : 1 column output
Columns 2 : 2 column output

DESCRIPTION:

Output header values. See the documentation for ChangeHeader to learn the header values. However one cannot use OGMT or OCAL since these are used only in ChangeHeader. No unset header value, incidated by the [en]12345 sequence, will be displayed.

EXAMPLES: List distance, azimuth and P and S picks
LH DIST AZ A T0

SAC COMPATIBILITY

SAC defaults to a single column listing and has a built in more command.

SEE ALSO

LOG

SUMMARY:

Take natural logarithm of trace

LOG

INPUT:

DESCRIPTION:

This takes the natural logarithm of a trace. However NOTHING will be done if the trace value is negative or zero.

SEE ALSO

SQRT, SQR, EXP, ABS

LOG10

SUMMARY:

Take base 10 log of trace

LOG10

INPUT:

DESCRIPTION:

This takes the base 10 logarithm of a trace. However NOTHING will be done if the trace value is negative or zero.

SEE ALSO

SQRT, SQR, EXP, ABS

LOWPASS

SUMMARY:

LowPass filter traces

LowPass [options]

where options is one or more of the following: [Butter | BEssel | C1] [Corner fc] [Npoles npoles] [Passes npass] [Tranbw tranbw] [Atten atten]

INPUT:

Butter : Butterworth filter
BEssel : BEssel filter
C1 : Chebyshev Type I filter
Corner : Corner frequency (R) range 0 – Nyquist
Npoles : Number of poles (I) range 1 – 10
Passes : Number of passes (I) range 1 – 2
Tranbw : Chebyshev transition bandwidth fraction (0.3 default)
Atten : Chebyshev stop band attenuation (30 default)

DESCRIPTION:

Lowpass filter using a BI–LINEAR Z–transformation implementation of a lowpass filter. A bi–linear method is chosen since this is easily implemented algebraically. Passes = 1 gives a causal filter while Passes = 2 gives a zero–phase filter with a 6db point at the corner frequency.

The lowpass filter design of the Chebyshev Type 1 filter is based on the information in Hamming (1997) equation (13.5.4) and Figure 13.6.1, This filter attempts to approximate a sharp lowpass filter. In reality a transition band is defined by Fp and Fs, where Fp is the lowpass corner frequency and Fs is the stopband, where Fs = (1 + tranbw)*Fp. The amplitude level of the stop band (f > Fs) is 1/atten. Actual implementation requires a parameter epsilon, eps, which defines the lowpass ripple, which varies between 1 and 1/sqrt(1 + eps*eps).

Hamming, Richard W. (1997). Digital Filters (3rd edition), Dover Publications, 296 pp, ISBN 048665088X

Given npoles and eps, the poles and zeros are given by

http://www.answers.com/topic/chebyshev[en]filter
The correct normalization amplitude together with poles and zeros is given in
Digital Filter Designers Handbook with C++ algorithms, C. Britton Rorabaugh 2nd Edition, McGraw Hill, New York, 479 pp 1997 Chapter 5

HEADER VALUES SET:

USER1 = permin, USER2=permax, where permin=1.0/filt_fh, and permax= 0.01/(npts * dt) for use by sacmft96 adn sacpom96

SEE ALSO:

HIGHPASS, BANDPASS, BANDREJECT

MAP

SUMMARY:

Produce GMT map

MAP [options]

where options may include

[North maxlat South minlat East maxlon West minlon] [Topography ON | OFF] [STation ON | OFF] [EPicenter ON | OFF] [Global ON | OFF] [Raypath ON | OFF] [KStnm ON | OFF] [Default]

INPUT:

North maxlat : maximum latitude
South minlat : minimum latitude
East maxlon : maximum longitude
West minlon : minimum longitude
STation : Do or do not plot station locations
EPicenter : Do or do not plot epicenter locations
Global : If ON, use a global linear projection from LATLON=[en]157/203/[en]80/80
Raypath : IF ON draw great circle path between epicenter and station
KStnm : IF ON draw station names
Default : returns to default parameters
Topography : if ON plots from global topography data base, otherwise just goasl is plotted.

DESCRIPTION:

This routine examines the sac headers and plots station locations as a filled sircle and the event locations as a star.

By default the SAC file headers are used to define the latitude and longitude limits.

If the latitude and longitude limits are not specified, the plot is based on the event and station latitude and longitudes.

Unless the GLOBAL is ON, a Mercator projection is used and the map LATLON variable in script never includes the poles.

The output of this routine is a shell script of GMT commands. To create the map.eps, you must enter the command: sh map.sh. The map.sh is annotated, so that you can easily change projections, symbol sizes and colors, raster databases with very little editing and some knowledge of GMT. The shell script also includes a topographic resampling so permit leass grainy plots for small regions as well as a crude image for global maps. The size of the image if kept small and should look good when converted to a PNG, GIF or JPG file.

The encapsulated PostScript file is called map.eps. This can be included in groff, LaTeX dociments, or converted to PNG, GIF, JPG etc using ImageMagick dispay or convert for including in Word documents or PowerPoint presentations.

EXAMPLES:

DEFAULT:

map topography off station on epicenter on global off raypath off kstnm off

SEE ALSO

MARKTIMES

SUMMARY:

Marks velocity times in plot

MARKTIMES [DEFault] [Distance Header|dist] [Origin   Header|GMT time | CAL time] [Velocities v ...] [ON|OFF]

INPUT:

DEFault : reset to default values

Distance Header : Use the distance in the trace header if set

Distance dist : Use dist as the distance

Origin Header : Use the origin time in the trace headers if set

Origin GMT time : Use time of the form YEAR JDAY HOUR MINUTE SECOND MILLISECOND

Origin CAL time : Use time of the form YEAR MONTH DAY HOUR MINUTE SECOND MILLISECOND

Velocities v ... : Set of velocities (km/s) to be used. No more than 10 are permitted

ON : Mark the times. This is automatically the condition when called unless turned off explicitly by the OFF command.

OFF : Turn off marking

DESCRIPTION:

This routine marks the velocity arrival times as a function of the origin time, distance and velocity set. These are indicated by BLUE colored tics with the command plot. The arrival times are determined from the simple equation

arrival_time = origin_time + distance/velocity

SAC COMPATIBILITY:

Nothing is set in the header. The purpose is to indicate approximate arrival times for phase identification. The default values also include 1, 7 and 8 km/sec.

EXAMPLES:

DEFAULT

MARKTIMES VELOCITIES 1. 2. 3. 4. 5. 6. 7. 8. DISTANCE HEADER ORIGIN HEADER

SEE ALSO

PLOT

MERGE

SUMMARY:

Merge all files in memory to form a single trace.

MERGE [GAP gap_value]

INPUT:

GAP gap_value : Insert gap_value when filling in gaps. The default is to use a value of 0.0

DESCRIPTION

Data requests from the IRIS DMC often yield segmented traces when the SEED volume is read using rdseed to create SAC files. This segmentation may reflect actual data gaps, or just different submissions to the database. The merge command will combine all files and provide disgnostic information on the absolute time/date of the beginning and end of each trace, whether there are data gaps, and whether there are data inconsistencies in the individual amplitude values.

SAC COMPATIBILITY:

sac2000 does not permit changing the default fill of the gap.

DEFAULT

SEE ALSO

MOMENTTENSOR

SUMMARY:

SUMMARY: Generate 3 component seismogram for moment tensor

MomentTensor TO [ZRT|ZNE|UZ|UR|UT|UN|UE|Z|R|T|N|E] MW mw Az az [Baz baz] [STK stk DIP dip RAKE rake | ISO | FN fn Fe fe FD fd | MXX mxx MYY myy MZZ mzz MXY mxy MXZ mxz MYZ myz] FILE fileproto

INPUT:

ZRT : generate ZRT components named T.Z T.R T.T
ZNE : generate ZNE components named T.Z T.N T.E (requires back azimuth)
UZ or Z : generate Z component named T.Z
UR or R : generate R component named T.R
UT or T : generate T component named T.T
UN or N : generate N component named T.N (requires back azimuth baz)
UE or E : generate E component named T.E (requires back azimuth baz)
MW mw : Moment magnitude (default 2.60 (for log Mo=20)).This is converted to Moment using Mw = 2/3(log Mo [en] 16.1) Mo in dyne[en]cm. This applies to double couple and isotropic sources.
MXX mxx : moment tensor element in units of dyne[en]cm, e.g., MXX 1.0e+20
MYY myy
MZZ mzz
MXY mxy
MXZ mxz
MYZ myz
Az az : source to receiver azimuth. (default 0)
Baz baz : back azimuth (e.g., from receiver to source) (default mod(az+180,360). Note for teleseisms use the true baz
ISO : point source isotropic source (explosion)
STK stk : strike of double couple model
DIP dip : dip of double couple model
RAKE rake : rake off double couple model
FE fe : point force directed east in multiples of 1.0E+15 dynes
FD fd : point force directed down in multiples of 1.0E+15 dynes
FN fn : point force directed north in multiples of 1.0E+15 dynes

FILE fileproto : full path of Green’s function prototype

DESCRIPTION:

This program accesses the Green s functions and generates a three component ground velocity seismogram in units off m/s. The command used the fileproto, e.g., /home/rbh/PROGRAMS.310t/GREEN/nnCIA.REG/0005/011500005 and adds the suffix .ZDD etc to find the Greens functions, which are combined to make a three component seismogram for the given mechanism and moment

Note that in the future the use of TO Z etc may be removed and onlu TO UZ will be permitted.

Defaults

The strike, slip, rake is the default. DO NOT MIX types, e.g., fource plus moment tensor.

EXAMPLES:

SEE

MUL

SUMMARY:

Multiply all SAC data files in memory by a constant.

SYNTAX

MUL [v],

INPUT:

[v] : constant by which all files are multiplied

SAC COMPATIBILITY:

SAC permits an extended syntax that permits applying different constants to respective files in memory. We have not implemented this complexity.

DEFAULT:

DIV 1

SEE ALSO

ADD, SUB, DIV

MULF

SUMMARY:

Multiply Files in memory

MULF [Master n] [Suffix suffix] [Default]

INPUT:

Master : Trace uses as master trace. Default is 0, which is the first in memory.
Suffix siffix : The traces are renamed and the original traces in memory are overwritten to be of the form
[STA2][CMP2]_[STA1][CMP1].suffix. The default value ofthe suffix is ’.mul’

DESCRIPTION:

This multiplies all traces in memory by the master trace. After the multiplication operation the files are named as follow: [STA2][CMP2]_[STA1][CMP1].suffix Beware that nothing is done to the original header other than to reset the start and end times since the output trace is only for the common overlapping absolute time window.

The result on the master trace is to square it, and perhaps to change the begin and end of the trace.

HEADER CHANGES

DEPMAX, DEPMIN, DEPMEN, NPTS, B, O

EXAMPLES:

DEFAULT:

MULF MASTER 0 Suffix .mul

SEE ALSO

ADDF, SUBF, DIVF

OUTCSV

SUMMARY:

Output time series as CSV for spreadsheet

INPUT:

DESCRIPTION:

This writes all times series in memory in ASCII to a file f001.csv, which consists of N+1 columns, where N is the number of traces in memory. Column 1 is the time with respect tot he reference time, and the remaining columns are the samples.

This program handles differing time windows by outputing the common time segment, as is done by the commands rotate, rotate3, addf, subf, mulf and divf.

The number of data points (rows) that can be read into a spreadsheet is limited. For example OpenOffice only permits 65536 rows.

This command was created to facilitate student manipulation of traces using EXCEL or OpenOffice.

EXAMPLES:

SEE ALSO

PAUSE

SUMMARY:

Pause a specified number of seconds

PAUSE Period delay

INPUT:

Period delay : number of seconds to wait

DESCRIPTION:

Termination processing until the time delay is met. It was originally intended to await an ENTER/RETURN but this would not work using stdio in shell scripts. On could of course open /dev/console in UNIX/LINUX but this may not be transportable.

The purpose of this routine lies in using a shell script to run gsac. The combination of an ECHO followed by a PAUSE permits text to be read before proceding with processing.

SAC COMPATIBILITY

SAC also permits a y/n response for continuation.

EXAMPLES:

SEE ALSO

ECHO

PCTL

SUMMARY:

Control time[en]domain plots

PCTL [options]

INPUT:

X0 x0 : X[en]position of lower left corner of plot
Y0 y0 : Y[en]position of lower left corner of plot
XLEn xlen : Length of x[en]axis YLen ylen : Length of y[en]axis
XLAb x[en]label : Label for X[en]axis
YLAb y[en]label : Label for Y[en]axis
Grid [ON|OFF] : Turn positioning grid on/off. This is for subplot alignment
Default : Reset to X0 1.5 Y0 1.0 XLEN 8.0 YLEN 6.0

DESCRIPTION:

This set controls for the current plot. When used with the HOLD command, multiple figures can be displayed on a frame,

EXAMPLES:

Default:

X0 1.25 Y0 1.0 XLEN 8.0 YLEN 6.0 XLAB \"Time (s)\" YLAB \"\"

SEE ALSO

PLOT1

SUMMARY:

Plot traces

Plot1 [options]

INPUT:

Perplot [n | OFF] :Plot n traces per plot frame
Absolute : Plot in absolute time
Relative : Plot with all traces starting at the first sample
Overlay [ON | OFF] : Overlay all traces in current frame

DESCRIPTION:

When plotting the traces in more than one window, e.g., when using the perplot option, the prompt asks is ’More? y/n/b’ [en] A return or ’y’ moves to the next plot, an ’n’ terminates the display, and the ’b’ returns to the previous set of displayed traces. This was instroduced 02/27/2006 to facilitate the review of many traces.

DEFAULT:

PERPLOT OFF ABSOLUTE OVERLAY OFF

SEE ALSO

PLOTPK

PLOTPK

SUMMARY:

Interactively work with traces

PlotPK

The options are

Perplot [n | off] : Plot n traces per frame or plot all on one frame (off)
MARKALL : Change headers for all traces within the plot frame. This is convenient for assigiing the same P–pick time to all 3–components recorded at

a station
MARKALLOFF : Turn off Markall – this is because we do not require a MARKALL ON
Relative : Plot traces accroding to time from the first sample
Absolute : Plot all traces in absolute time
REGional : Put up a simple regional phase menu
Teleseism : Put up a simple teleseism phase menu
Quality : Put up a simple quality control menu
PQuality : Put up a simple quality control menu and repick P
Default : Do not put up a phase menu, turn off marking

DESCRIPTION:

The cursor responds to the following commands:
[en] : compress time scale by factor of 2, recenter trace
_ : compress time scale by factor of 2, recenter trace
+ : expand time scale by factor of 2, recenter trace
= : expand time scale by factor of 2, recenter trace
(space) : recenter trace
* : increase trace amplitude by factor of 2
/ : decrease trace amplitude by factor of 2
A : accept (put +1 in IHDR20 for trace)
R : reject (put 0 in IHDR20 for trace)
B : move to the previous page of traces
F : insert a FINI marker (end of useful signal)
L : give time and amplitude of point beneath cursor
N : move to next set of traces
O : return to original trace scaling
P : mark P time
S : mark S time
Q : end interactive trace picking
Tn : set Tn header where n is a value from 0[en]9
Note that S sets T0. (Just enter T and then an integer)
X : Define trace window by entering X two times

If the REGIONAL or TELESEISM are flagged, a small phase menu appears: P S Pg Lg for regional phases and P S and PKP for teleseismic phases. To use these menus, choose the phase from the menu, then select the time value from the trace, and classify the quality of the arrival. If the P phase is selected, then the first motion polarity must be indicated.

The Quality menu permits the use of the ’a’ and ’r’ keys to mark a trace for further use. In addition, the default action of any mouse press is to indicate accept, unless the Reject Menu button is pressed. Normally IHDR20 is set to [en]12345. An accept sets this to +1 and a reject to [en]1. If a WriteHeader is executed, then the trace headers are set to the new values, and a shell script can then select traces for further processing. The purpose this command is to use gsac graphics to speedily judge trace quality for other processing.

The Pquality menu works slightly differently from the Quality menu in that the objective is to select good traces and also to repick the P arrival. Just place the crosshair on the P arrival, click any mouse button and the P is repicked and the trace is selected (IHDR20 is set ot 1) with one click.

NOTE:

On July 10, 2010 the interactive cursor will respond to the ˜ (tilde) character by creating a screen dump with name DUMPxxx.PLT. This dump includes the menu and is introduced to assist documentation.

SEE ALSO

PLOT1

PLOTSP

SUMMARY:

Plot spectra traces

PlotSP [options]

INPUT:

AMplitude : Plot amplitude spectrum (default)
PHase : Plot phase spectrum
PErplot [n|OFF] : Plot n spectra per frame (default off)
Overlay [ON|OFF] : Overlay all spectra (default off)
XLIn : X[en]axis is linear
XLOg : X[en]axis is logarithmic (default)
YLIn : Y[en]axis is linear
YLOg : Y[en]axis is logarithmic (default)
FMIn : Minimum frequency for plot (default: DF for XLOG and 0 for XLIN)
FMAx : Maximum frequency for plot (default: Nyquist)
AMIn : Minimum spectral amplitude to plot (default: 0 for YLIN and 0.0001 Amax for YLOG)
AMAx : Maximum spectral amplitude to plot (default: maximum)
Default : Reset to default

DESCRIPTION:

After using the FFT command, the spectra are stored in memory along with the trace.

SAC Compatibility

If one wishes to look at the spectra and then the trace, SAC requires that the trace be reread. GSAC has both in memory, so that one can alternate PLOT1 and PLOTSP to see the traces and the spectra.

SEE ALSO

Notes

The plot limits for this will be independent of the plot limits of the trace plots.

PLOTRECORDSECTION

SUMMARY:

Creates a record section of traces plotted as a function of a header value.

PlotRecordSection hv [LANDSCAPE|PORTRAIT|SEASCAPE|REVERSE] and others

INPUT:

hv : Use this header value. The default is DIST. One may select from the following: EVDP, STEL, GCARC, DIST, AZ, BAZ, GCARC, USER0, ..., USER9, and MAG

PORtrait: Time increases to the right. The header value increases upward.

REVerse : Time increases to the right. The header value increases downward.

Landscape : Time increases upward. The header value increases to the right. This is a seismic refraction convention.

Seascape : Time increases downward. The header value increases to the right. This is a seismic reflection convention.

Absolute : Plot in absolute time

Relative : Plot with all traces starting at the first sample

TItle string : Title for axis. The Default titles are Dist (km/sec), Azimuth (deg), Back Azimuthy (deg, Distance (deg), USER0, ..., USER9, Depth (km), Receiver depth, and Magnitude. This option is best used to substitute for USER0, ..., USER9

P p : Make p[en]tau plot by plotting T [en] p DIST where p =sec/km.

PX p
DTDX p
PDEL dtdd : Make p[en]tau plot by plotting T [en] dtdd GCARC
DTDD dtdd
AMP amp : Change maximum amplitude of trace from 0.5 to amp units. Recall that the screen in 10 plot units wide and 8 plotunits high
SHade [POS|NEG|OFF|ALL] : Shaded area plot of trace for positive and negative amplitudes.
Color color : color for shading; Default = black (1). Red = 2, Blue = 4, 1000 (red/lt gray), 1100 (blue/dk gray).
KF first_trace_shade : shade color for first trace read usinc Color convention
KL last_trace_shade : shade color for last trace read usinc Color convention
DEfault : Reset all parameters
VLimit vl vh :
TLimit tl th :
ANnotate string : Annotate trace. Use string = STA to annotate with station name, = OFF to turn off
ScaleRelative : Each trace is plotted such that the maximum amplitude is always ’amp’
ScaleAbsolute 0.0 :
ScaleAbsolute 0.5 :
ScaleAbsolute 1.0 :
ScaleAbsolute 1.5 :
ScaleAbsolute 2.0 :
ScaleAbsolute 2.5 : Traces are scaled according as hvˆpower, where power can only be one of 0.5, 1.0, 1.5, 2.0, 2.5. The purpose is to present the difference in true amplitudes between traces

DESCRIPTION:

The purpose of this is to create a plot that goes beyond simple trace view. It is often desirable to plot traces in terms of true distance to properly understand arrivals through their moveout. In other cases one may wish to look at the variation of receiver functions.

If one uses the program saciterd to create a receiver function, the ray parameter is stored in the header variable USER4. To look a the change in the receiver function with ray parameter, one would just enter

PRS USER4 To see the variation with backazimuth from the station, enter
PRS BAZ

Note that if the Origin time is set and the Absolute mode is used, then the time axis will actually correspond to to the travel time instead of the time from the earliest time value.

Note also that if you wish to overlay observed and synthetics traces which have different reference times that you MUST use the PRS RELATIVE command. The PRS RELATIVE axis scaling is based on the B value of the first trace in memory. PRS RELATIVE may do strange things when using the P or PX commands, especially of the B value is no consistently set, which it is, for example, using the iterative deconvolution program, saciterd.

DEFAULT:

PRS DIST LANDSCAPE ABSOLUTE ScaleRelative

SEE

PLOT1, PLOTPK, SORT

QDP

SUMMARY:

Control decimation for screen plots.

QDP [ON|OFF|n]

INPUT:

ON : Turn on automatic decimation
OFF : Turn off automatic decimation
n : Define the exact decimation

DESCRIPTION:

This controls the decimation factor used for screen plots. The purpose is to speed screen displays. When OFF, all points are plotted in the current window. When ON, the increment is automatically determined to permit no more than 4000 display points. The exact number used in the decimation can be controlled by the QDP n.

To permit rapid screening of the trace and yet to be able to consider the traces at its maximum resolution, decimation is turned off if the number of points to be plotted in the current window is <= 4000. This is very useful within PLOTPK when one wishes to focus on arrival exact arrival time picks, after quickly moving the trace.

The plots resulting from a BG PLT command which creates the Pnnn.PLT CALPLOT files are not decimated.

This command was introduced since the CYGWIN plots under WINDOWS can be slower that the corresponding plot under LINUX on the same computer.

DEFAULT:

QDP OFF

SEE ALSO

PLOT1, PLOTPK

READ

SUMMARY:

Reads data from SAC data files on disk into memory.

Read [options] [filelist], or

where options is one or more of the following: MORE

INPUT:

MORE : Place the new files after the old ones. If this is omitted, new data replaces the old ones.

[filelist] : name of SAC data files. The files in the list are tested to determine if they are a binary file in either IEEE Little Endian (INTEL architecture) or Big Endian (SPARC, for example). Files are SAC files if the version number of the SAC file (NVHDR) is 6 AND at least ONE of the real header values is –12345.0 or integer header values is –12345

SAC COMPATIBILITY:

The MORE option differs in that a subsequent READ will read in all files. Each invocation of the READ MORE appends to list of SAC files in memory.

SEE ALSO

CUT

READHDR

SUMMARY:

Reads headers from SAC data files into memory.

ReadHdr [options] [filelist]

where options is one or more of the following: MORE

INPUT:

MORE : Place the new files after the old ones. If this is omitted, new data replaces the old ones. ALL HEADERS ARE READ IN AGAIN

filelist] : name of SAC data files. The files in the list are tested to determine if they are a binary file in either IEEE Little Endian (INTEL architecture) or Big Endian (SPARC, for example). Files are SAC files if the version number of the SAC file (NVHDR) is 6 AND at least ONE of the real header values is –12345.0 or integer header values is –12345

SAC COMPATIBILITY:

The MORE option differs in that a subsequent READHDR will read in all files is the aggregate list.

DESCRIPTION:

The purpose of this command is to speed making changes in the header by note reading in the complete timeseries.

SEE ALSO

READ

REFRACTION

SUMMARY:

Enter Refraction Processing Mode for Record Section

REFRaction [ON|OFF] [Ex|Reg|Tel]

INPUT:

DESCRIPTION:

This places the PLOTRECORDSECTION (prs) display into an interactive mode for refraction studies.

Ultimately we will have data structure for the arrivals, and when gsac terminates these will be written to an ascii file. Also if REFRACTION is even invoked, then the picked refraction lines will be displayed on the screen using an XOR pen [en] this is useful if one leaves PRS [en][en] or if reread then get rid of everything [en] think about all of this, perhaps we need a save button instead of a data structure

Ex [en] Exploration mode [en] Filter frequencies are 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 150, 200, 250, 500, 1000 Hz
Reg [en] Regional mode [en] Filter frequencies are 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25, 50, 100 Hz
Tel [en] Teleseism mode [en] Filter frequencies are 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.10, 0.15, 0.2, 0.25, 0.5, 1 Hz

EXAMPLES:

The terminal text output of the DoRefr command is the following if the value is accepted:
S: Refr t0 [en]0.005536 p 6.474635 (sec/km) Vel 0.154449 (km/sec) Refractor 1
S: Refr t0 0.159714 p 0.513774 (sec/km) Vel 1.946381 (km/sec) Refractor 2

which indicates that these are S[en]wave refraction measurements for the first two refraction arrivals. This information can be used in a simple program to compute layer thicknesses.

The terminal text output of the DoRefl command is the following if the value is accepted:
S: Refl t0 0.154164 p 6.384913 (sec/km) Vrms 0.156619 (km/sec) Reflector 1 Multiple 1
S: Refl t0 0.308500 p 5.856309 (sec/km) Vrms 0.170756 (km/sec) Reflector 1 Multiple 2
These values can be used to determine layer thickesses.

For each PRS001.PLT or REFR001.PLT file there is a corresponding PRS001.CTL or REFR001.CTL control file, which can be used to invoke the program refmod96 to make model predicted overlays.

In addition there is a refrpick.tmp file which will ultimately be used by an invrsion program.

NOTE:

On July 10, 2010 the interactive cursor will respond to the ˜ (tilde) character by creating a screen dump with name DUMPxxx.PLT. This dump includes the menu and is introduced to assist documentation.

SEE ALSO

REVERSE

SUMMARY:

Reverse Time Series

REVERSE [Suffix suffix]

INPUT:

Suffix : suffix for the files names. The default is .rev

DESCRIPTION:

This routine reverses the time sequence in place. The time header values with respect to timing are changed to reflect the absolute reversal in time:

EXAMPLE

HEADER VALUES SET:

B,E, A, O, T0, T1, ..., T9, NZYEAR, NZJDAY, NZHOUR, MZMIN, NZMSEC

SEE ALSO

RICKER

SUMMARY:

Convolve with Ricker wavelet

RIcker F frequency [Default]

INPUT:

F frequency : the frequency of the Ricker wavelet

DESCRIPTION:

The Ricker wavelet with frequency ’f’ is defined as follows:

f(t) = [1 [en] 0.5 * (2 pi t t)ˆ2] exp [[en] (pi f t)ˆ2]

Default:

RIcker F 25

SEE ALSO

TRIANGLE, TRAPEZOID, BOXCAR

RMEAN

SUMMARY:

Remove mean from waveform

RMEAN

DESCRIPTION:

This removes the trace mean from the trace.

SEE ALSO

RTREND

ROTATE

SUMMARY:

Rotates horizontal components through and angle

ROTate [TO GC | TO angle] [Suffix suffix]

INPUT:

TO GC : Rotate to the great circle path. This requires that the BAZ and CMPAZ header variable be set. The KCMPNM headers are changed to replace the last character, usually N and E, to R and T. In addition the file name for default changes

TO angle : Rotate to form the trace in the {angle} and {angle + 90} directions. this requires that the CMPAZ be set in the header The KCMPNM headers are changed as are the default write names.

Suffix suffix : Append the suffix to the constructed file name. This is useful when the command is followed by WRITE without any arguments.

DESCRIPTION:

In both uses of the command, the filename and KCMPNM are converted to upper case. Note that a write will be in the current directory rather in the directory of the original traces.

As an added feature, ROTATE is smart enough to handle traces that do not have equal lengths or absolute start time. The will consist of the overlapped trace window. This means that it should be possible to ROTATE without having to SYNCHRONIZE and CUT

The following dialog illustrates the naming, Recall that the LISTHEADER gives the filename for the default write.

```
GSAC> r ../020618/BLOLHN.sac ../020618/BLOLHE.sac
GSAC> lh cmpaz cmpinc az baz
      AZ        39.23643      BAZ        220.0237
   CMPAZ               0   CMPINC              90
```

```
          AZ          39.23643        BAZ          220.0237
       CMPAZ                90     CMPINC                90
GSAC> rotate to gc
Rotating to great circle to form R and T
GSAC> lh
BLOLHR (0):
          AZ          39.23643        BAZ          220.0237
       CMPAZ          40.02368     CMPINC                90
BLOLHT (1):
          AZ          39.23643        BAZ          220.0237
       CMPAZ          130.0237     CMPINC                90
GSAC> rotate to 40
Rotating to angle 40.000000 to form 040 and 130
GSAC> lh
BLOLH040 (0):
          AZ          39.23643        BAZ          220.0237
       CMPAZ                40     CMPINC                90
BLOLH130 (1):
          AZ          39.23643        BAZ          220.0237
       CMPAZ               130     CMPINC                90
```

HEADER VALUES SET:

The CMPAZ is properly set to each rotated component. If the components are rotated to the great circle (GC), then CMPAZ for the resulting radial component is BAZ+180 and for the transverse component is BAZ + 270.

If the angle is given, then the positive motion will be in the direction of the angle given in the file name.

After the rotate command the order of traces in memory is radial and transverse, or AZ and AZ + 90.

SEE ALSO

ROTATE3

SUMMARY:

Rotate three components to form Z and horizontals or UVW

ROTate3 [TO GC | TO angle | TO UVWSTS2 | UVWTRIL | ZNE] [Suffix suffix]

INPUT:

TO GC : Rotate to the great circle path. This requires that the BAZ and CMPAZ header variable be set. The KCMPNM headers are changed to replace the last character, usually N and E, to R and T. In addition the file name for default changes

TO angle : Rotate to form the trace in the {angle} and {angle + 90} directions. this requires that the CMPAZ be set in the header The KCMPNM headers are changed as are the default write names.

TO UVWSTS2 : Rotate x (E) y (N) z (up) to UVW for STS[en]2

TO UVWTRIL : Rotate x (E) y (N) z (up) to UVW for Trillium

The difference between the two conversions to UVW is that
U(STS2) == U(TRIL) rotated horizontally 180 degrees
V(STS2) == W(TRIL) rotated horizontally 180 degrees
W(STS2) == V(TRIL) rotated horizontally 180 degrees

TO ZNE : Rotate to ZNE. This is useful if one as UVW or event ZRT and wants a nice clean naming of rht output.

Suffix suffix : Append the suffix to the constructed file name. This is useful when the command is followed by WRITE without any arguments.

DESCRIPTION:

This is an extension of the ROTATE command that uses all three components to form the three rotated components. Although it may seem redundant to include the vertical component in a rotation, this is useful if the CMPINC of the vertical component in 180, meaning that the positive trace value is down rathern than the desired up. At the same time, the use of ROTATE3 (ROT3) ensures that the three component have the same default naming convention, e.g., SLMBHR, SLMBHT and SLMBHZ.

In both uses of the command, the filename and KCMPNM are converted to upper case. Note that a write will be in the current working directory rather in the directory of the original traces.

As an added feature, ROTATE3 is smart enough to handle traces that do not have equal lengths or absolute start time. The will consist of the overlapped trace window. This means that it should be possible to ROTATE without having to SYNCHRONIZE and CUT

HEADER VALUES SET:

The CMPAZ and CMPINC are properly set to each rotated component. If the components are rotated to the great circle (GC), then CMPAZ for the resulting radial component is BAZ+180 and for the transverse component is BAZ + 270.

If the angle is given, then the positive motion will be in the direction of the angle given in the file name.

The default order of the traces in memory is radial (R), transverse (T) and vertical (Z), AZ, AZ+90 and vertical (Z). Thus the default WRITE would identical to the command

w SLMBHR SLMBHT SLMBHZ

for example. When the conversion to UVW is requested, the default write would be the same as the command

w SLMBHY SLMBHV SLMBHW

SEE ALSO

ROTATE

RTREND

SUMMARY:

Remove linear trend from waveform

RTRend

INPUT:

None.

DESCRIPTION:

This fits a straight line through the data and removes the trend by a simple least squares algorithm.

SEE ALSO

RMEAN

SGN

SUMMARY:

SGN

INPUT:

DESCRIPTION:

This implements a 1[en]bit digitization according to the rule

b(t) = +1 for s (t) >=0

b(t) = [en]1 for s (t) < 0

The DEPMAX and DEPMIN are reset. Note for this to work one should perform a rmean prior to invocation.

References

Derode, A., A. Tourin and M. Fink (1999). Ultrasonic pulse compression with one[en]bit time reversal through multiple scattering, J. Appl. Phys. 85, No 9, 6343[en]6352.

EXAMPLES:

SEE ALSO

SHIFT

SUMMARY:

Shift trace in time

SHIFT Fixed sec

INPUT:

Fixed amount : Shift the trace by amount seconds.

DESCRIPTION:

The purpose of this is to adjust synthetics for a source delay. If the amount is positive, then the trace is shifted to be later in time. This is accomplished by changing the B and E parameters in the header. In addition all arrival time picks, such as A, T0 ... T9 are adjusted also. The origintime offset is NOT changed.

EXAMPLES:

SEE ALSO

SMOOTH

SUMMARY:

Apply a smoothing operator

SMOOTH [MEAn|MEDian] [Halfwidth n] [Pass p] [Default}

INPUT:

MEAN (default) : Apply an averaging operator
MEDIAN : Apply a median filter
Halfwidth n : Smoothing operator consists of 2n+1 points (default 1)
Pass p : Apply the operator p times
Default : Reset to MEAN Halfwidth 1 Pass 1

DESCRIPTION:

SAC COMPATIBILITY:

The Pass option is new.

DEFAULT:

SMOOTH MEAN HALFWIDTH 1 PASS 1

SEE ALSO

ABS, ENV

SORT

SUMMARY:

Sort all displays by the key

SORT [OFF|DEFAULT] [UP|DOWN] [FORWARD|REVERSE] [ASCEND|DESCEND] header_variable

INPUT:

header_variable : The header value to be sorted, e.g, O, DIST, GCARC, etc.

UP, ASCEND, FORWARD : Sort in order of increasing value so that a LISTHEADER header_variable gives the smallest value first

DOWN, DESCEND, REVERSE: Sort in order of decreasing value so that a LISTHEADER header_variable gives the largest value first

OFF, DEFAULT : Do not sort

DESCRIPTION:

This sorts the trace display according to the value of key. To sort on distance,

sort dist

If USER4 has a ray parameter from saciterd, then

sort user4 plot1

will display the traces in order of the ray parameter.

At present only a sort on the integer or floating point header values is implemented.

SEE ALSO

LISTHEADER

SQR

SUMMARY:

Square each data point of the trace

SQR

INPUT:

DESCRIPTION:

SEE ALSO

SQRT, ABS

SQRT

SUMMARY:

Take the square root of each data point in the trace.

SQRT

INPUT:

DESCRIPTION:

Take the square root of a trace. Note this fails if a trace value is < 0.0.

SEE ALSO

SQR, ABS

STACK

SUMMARY:

Stack traces

STACK [Relative|Absolute] [Norm On | Off] [Suffix suffix]

INPUT:

Relative : stack according to first sample. Output length is controlled by smallest time window

Absolute : Stack in absolute time. Output window is based on latest begin time and earliest end time.

Norm On | Off : If on divide the stack by the number of traces used

Suffix : suffix for the files names. The default is .stk

DESCRIPTION:

EXAMPLES:

DEFAULT:

STACK ABSOLUTE NORM OFF

HEADER CHANGES:

DEPMIN, DEPMAX, DEPMEN. IHDR11 is set to the number of traces actually stacked.

SEE ALSO

CORRELATE

SUB

SUMMARY:

Subtract a constant to all SAC data files in memory.

SUB [v]

INPUT:

[v] : constant to be subtracted to all files

SAC COMPATIBILITY:

SAC permits an extended syntax that permits applying different constants to respective files in memory. We have not implemented this complexity.

DEFAULT

SUB 0

SEE ALSO

ADD, MUL, DIV

SUBF

SUMMARY:

Add Files in memory

SUBF [Master n] [Suffix suffix] [Default]

INPUT:

Master : Trace uses as master trace. Default is 0, which is the first in memory.
Suffix siffix : The traces are renamed and the original traces in memory are overwritten to be of the form
[STA2][CMP2]_[STA1][CMP1].suffix. The default value ofthe suffix is '.sub'

DESCRIPTION:

This subtracts the master trace from all traces in memory. After the subraction operation, the files are named as follow: [STA2][CMP2]_[STA1][CMP1].suffix Beware that nothing is done to the original header other than to reset the start and end times since the output trace is only for the common overlapping absolute time window.

HEADER CHANGES

DEPMAX, DEPMIN, DEPMEN, NPTS, B, O

EXAMPLES:

DEFAULT:

SUBF MASTER 0 Suffix .sub

SEE ALSO

ADDF, MULF, DIVF

SYNCHRONIZE

SUMMARY:

Synchronize reference times

SYNChronize [o | O] [a | A]

INPUT:

o : set the reference time as the origin time
a : set the reference time as the P arrival time

DESCRIPTION:

The purpose of this command is to make the reference times the same for all traces in memory. It determines the earliest absolute starting time of all files, and sets the B time fo that file to 0.0. The B times of other files are then always >0 =. Although SAC used the latest start time, ours ensures that B is never set to the magic –12345.

As a result of this operation, any marked times, e.g., B, E, O, A, Tn are result so that the absolute time of these markers are not changed.

A previous use for this was to ensure that MARKALL and ROTATE will work properly. GSAC works with absolute time.

The reason for the O (oh [en] origin time) option is to set the origin time as the reference time. This is useful when documenting events [en] just use saclhdr with the [en]KZDATE [en]KZTIME options.

The reason for the A option is to align waveforms on the P arrival in a two step process. First this option changes the reference times so that the A header value is zero. Then one can use a

ch NZYEAR year NZJDAY jday NZHOUR hour NZMIN min NZSEC sec NZMSEC msec

to change the reference time. This is one way to overcome bad timing on the individual channels.

SEE ALSO

TAPER

SUMMARY:

Apply a symmetric taper to ends of traces

TAPER [Cosine|HANning|HAMming] [WIDTH w]

INPUT:

Cosine : apply a cosine taper to each end of the trace.
HANning : apply a Hanning taper
HAMming : apply a Hamming taper
WIDTH w : Taper width of as a function of the entire trace. This is a value between 0.0 and 0.5. If the first and last 25% of the trace are to be tapered, w = 0.25. w = 0 implies no taper. If a value of w is > 0.5, 0.5 is used; if w < 0.0, 0.0 is used.

DESCRIPTION:

The taper function taper(x) varies from [0,1] as t varies in the range [0,1]. The mathematical definitions of the function is:

taper(x) = A + B FUN (C x)

where

Taper	A	B	FUN	C
COSINE	0.0	1.0	sin	PI/2
HAMMING	0.54	[en]0.46	cos	PI [note taper(0) = 0.08]
HANNING	0.5	[en]0.5	cos	PI

HEADER

DEPMAX, DEPMIN, DEPMEN

DEFAULT:

TAPER HANNING W 0.05

SEE ALSO

TITLE

SUMMARY:

This command defines the title for a plot

Title [ON|OFF] [Location Top|Bottom|Right|Left] [Size Tiny|Small|Medium|Large] [Default] Text text

INPUT:

Location Top|Bottom|Right|Left : Position of title with respect to plot frame

Size Tiny|Small|Medium|Large : Size of characters [en] these are scaled to a fraction of the plot dimensions

Default : Reset all parameters

ON|OFF : turn the title on or off

Text text : the title text. Note that if the text constains spaces it must be in within quotes.

DESCRIPTION:

If this option is on, a title can be placed on the plot. Of course, if the CALPLOT graphic file is converted to Encapsulated PostScript, then the title can be added using the programs xfig of Illustrator.

SAC COMPATIBILITY:

Note that sac2000 does not require the text to be defined by a keyword. gsac requires that the title be keyed using the Text keyword.

DEFAULT:

Title OFF Location Top Size Small Text None

SEE ALSO

TRANSFER

SUMMARY:

Apply or remove an instrument response/filter from the data

TRANSfer [FROM|TO] [Polezero SUBTYPE pzfile] [FApfile SUBTYPE fapfile] [evaL SUBTYPE afile pfile] [ACC | VEL | DISP]
[FREQlimits f1 f2 f3 f4]

INPUT:

FROM : Deconvolve the filter from the trace
TO : Convolve the filter with the trace
Polezero SUBTYPE pzfile : Use SAC pole–zero format
Eval SUBTYPE afile pfile : Use output of IRIS evalresp program which creates two files, each of two columns. The afile has columns of frequency and amplitude in order of increasing frequency. The pfile has colums of frequancy and phase (degrees) in order of increasing frequency.
FApfile SUBTYPE fapfile : a GSE frequency amplitude period file
ACC | VEL | DISP : an internal type [en]
basically ACC means multiply/divide by (i omega)ˆ2 for TO/FROM
basically VEL means multiply/divide by (i omega) for TO/FROM
basically DISP means multiply/divide by (1) for TO/FROM
FREQlimits f1 f2 f3 f4 : Apply a cubic taper to the response such the response is 0 for f < f1 and for f > f4, the response is 1 for f > f2 and for f < f3, and tapers cubically from 0 to 1 for f1 < f < f2 and f4 > f > f3. Note the only way to turn this off is to reset the limits as in FREQLIMITS –2 –1 1.0e5 1.0e6 FREQLIMITS is only used in the FROM process. This is essential for a clean deconvolution

DESCRIPTION:

SAC COMPATIBILITY:

This does not support the many built[en]in instrument responses of SAC. Instead the user must define the corresponding pole–zero or response file.

The EVAL option is different than SAC. Instead of working with a seed database, GSAC expects the user to have already run 'evalresp' independently to create two files with names such as AMP.NM.SLM..BHZ, and PHASE.NM.SLM..BHZ

The FAP file can have a line start with a # to indicate a comment line. The lines following the # signs are essential, however the count of the number of entries is ignored by GSAC. An example of this format is

```
 # Velocity response for INCN BHZ
 #
 #
 #
 # Phase unwrapped
 #
  theoretical  0   instrument    fap Organization
 40
  0.100000E[en]02  0.141401E+09  0.149665E+03  0.000000E+00  0.000000E+00
  0.100926E[en]02  0.143988E+09  0.149365E+03  0.000000E+00  0.000000E+00
```

```
 0.101861E[en]02  0.146621E+09  0.149061E+03  0.000000E+00  0.000000E+00
 0.102804E[en]02  0.149300E+09  0.148754E+03  0.000000E+00  0.000000E+00
 0.103757E[en]02  0.152025E+09  0.148443E+03  0.000000E+00  0.000000E+00
```

For compatibility with SAC, the SUBTYPE field MUST be used.

Note that the use of DISP, VEL or ACC with any operation requires that the user know what the original filter relates, e.g., counts/meter

HEADER VALUES SET:

USER1 = permin, USER2=permax, where permin=MAX[1.0/filt_f3,old permin] and permax=MIN[1./filt_f2,old permax]. This feature is used by sacmft96 and sacpom96

SEE ALSO

FILTER

TRAPEZOID

SUMMARY:

Convolve with unit area trapezoid

TRAPEZOID Width L1 L2 L3

INPUT:

Width L1 L2 L3 : L1, L2 and L3 define the trapezoidal pulse. These values are adjusted to lie on a sample with.

DESCRIPTION:

Convolve the time series in memory with a unit area trapezoidal pulse.

This acts as a lowpass filter.

If L1 + L2 + L2 < 2*DELTA, nothing is done

EXAMPLES:

SEE ALSO

TRIANGLE, BOXCAR, RICKER

TRIANGLE

SUMMARY:

Convolve with unit area triangle

TRIANGLE [Half half_width] [Width width]

INPUT:

Half half[en]width : the half[en]width of the isoceles triangle function rounded to the next sample interval.

Width width : the base width of the isoceles triangle function rounded to the next sample interval.

DESCRIPTION:

This routine convolves all traces in memory with a unit area isoceles trianglar pulse. This command is equivalent to

TRAPEZOID WIDTH half_width 0.0 half_width

TRAPEZOID WIDTH width/2 0.0 width/2

This acts as a lowpass filter.

If half_width < DELTA, no filtering is done

EXAMPLES:

SEE ALSO

TRAPEZOID, BOXCAR, RICKER

VERSION

SUMMARY:

Print GSAC version number

Version

INPUT:

DESCRIPTION:

EXAMPLES:

SEE ALSO

WHITEN

SUMMARY:

Whiten signal

WHITEN [DEFAULT] [FREQlimits f1 f2 f3 f4] [Absolute]

INPUT:

DEFAULT : initialize frequency limits

FREQlimits f1 f2 f3 f4 : Apply a cubic taper to the response such the response is 0 for f < f1 and for f > f4, the response is 1 for f > f2 and for f < f3, and tapers cubically from 0 to 1 for f1 < f < f2 and f4 > f > f3. Note the only way to turn this off is to reset the limits as in FREQLIMITS –2 –1 1.0e5 1.0e6 This may eliminate the need for a bandpass following the whitening.

Absolute : make spectrum absolute flat.

DESCRIPTION:

This routine determines a smooth amplitude spectrum of the signal, and then normalizes the signal by the spectrum. Because of the possibility of zeros in the amplitude spectrum, a water[en]level deconvolution is used. The purpose of this routine is to use it prior to bandpass and correlation.

If Absolute, the spectrum is not smoothed, which leads to a nominally flat amplitude spectrum with some character. Instead the signal has an absolutely flat amplitude spectrum.

EXAMPLES:

HEADER VALUES SET:

USER1 = permin, USER2=permax, where permin=MAX[1.0/filt_f3,old permin] and permax=MIN[1./filt_f2,old permax]. This feature is used by sacmft96 and sacpom96

DEFAULT:

WHITEN FREQlimits –2 –1 1.0e5 1.0e6

SEE ALSO

WRITE

SUMMARY:

Writes trace files in memory to disk.

Write [options] [filelist], where

where options is one or more of the following:

[APPEND text] [PREPEND text]

INPUT:

APPEND : Append the text to the beginning of all file names. However the leading directory information will be stripped and the file written in the current working directory.

PREPEND : Add the text to the end of all file names. However the leading directory information will be stripped and the file written in the current working directory.

SAC COMPATIBILITY:

DESCRIPTION: With no filelist, the original data files are overwritten by the current versikons in memory. The output order is that in which they were read in and not the way that they are sorted for display.

If the filelist is given, there must be a one–to–one correspondence between the number of traces in memory and the number of file names in filelist.

As of August 15, 2007 the APPEND and PREPEND options will write the files in the current directory only.

As of July 22, 2009, the APPEND and PREPEND options have no short cuts and must be written exactly.

SEE ALSO

WRITEHEADER

WRITEHEADER

SUMMARY:

WriteHeader

INPUT:

DESCRIPTION:

This overwrites the header information of the corresponding trace files in memory. The trace information is not overwritten. This is useful if one wishes to filter traces prior to picking arrival times, but only want to save the arrival times and not the filtered traces.

SEE ALSO

WRITESPEC

SUMMARY:

Write spectra

WriteSPec [options]

where options are [AM] [Append text] [Prepend text]

INPUT:

AMplitude : output the amplitude spectrum

Append : Append the text to the beginning of all file names. However the leading directory information will be stripped and the file written in the current working directory.

Prepend : Add the text to the end of all file names. However the leading directory information will be stripped and the file written in the current working directory.

DESCRIPTION:

If the traces have had the command FFT applied, the amplitude spectrum can be written as a trace file with this command. The filename of the outfile will be the same as the trace file with an ’.am’ appended.

To distinguish this file from a time series, the following header values are set in the ’.am’ file: LEVEN = true, IFTYPE = IXY, B = 0, NPTS = N/2 + 1 where N is the power of two used in the FFT, DELTA = DF where DF = 1/N*DELTA is the frequency sampling. Only the positive frequencies are output. The station and component names are preserved.

All time markers are reset to an uninitialized value of [en]12345. The reference time and date are preserved.

SAC COMPATIBILITY:

DESCRIPTION:

sac2000 ([8/8/2001 (Version 00.59.44)]) does not have the APPEND and PREPEND options for the spectra.
As of August 15, 2007 the APPEND and PREPEND options will write the files in the current directory only.

EXAMPLES:

HEADER VALUES SET:

IFTYPE is set to IXY which means that this is a general xy plot [en] we use this since sac2000 does not define a frequency series. The IXY flag checked by the PLOT command to set the horizontal axis as frequency

LEVEN = true

DEFAULT

WRITESPEC AM

SEE ALSO

XGRID

SUMMARY:

Control x[en]axis grid

XGRID [ON | OFF] [Solid | Dotted] [Color int_value] [Minor ON | OFF]

INPUT:

ON : turn grid on
OFF : turn grid off
Solid : Use solid line
Dashed : Use dotted line
Minor : connect minor tics too if ON
Color int_value : Define the color for the grid. The figure frame will continue to be in black. Be careful to select a color not used for the trace.

DESCRIPTION:

This annotates the plots with a grid. Note that the Dotted option is takes more time to plot.

EXAMPLES:

DEFAULT

GRID OFF DOTTED COLOR 1030 MINOR OFF
[Note COLOR 1 is black, 2 red, 3 green, 4 blue, 1030 pale yellow]

SEE ALSO

GRID, YGRID, COLOR

XLIM

SUMMARY:

Set time axis limits for trace plot

XLIM [ON|OFF] [ref offset | GMT beg | CAL beg] [ref offset | GMT end | CAL end]

INPUT:

ON : Turn on plot limits returning to previous value
OFF : Turn off plot limits
ref : A header reference value for the cut which is one of B|E|O|A|Tn where n=0,...,9
offset : Number of seconds relative to the reference value. refbeg offset refers to the start point refend offset refers to the end point
CAL : Calendar time in YEAR MONTH DAY HOUR MINUTE SECOND MILLISECOND
GMT : GMT time in YEAR DAYOFYEAR HOUR MINUTE SECOND MILLISECOND

DESCRIPTION:

This permits user modification of the trace display in PLOT1 and PLOTPK. One must be careful about the window because PLOT1 or PLOTPK can have either ABSOLUTE or RELATIVE plot modes.

In the ABSOLUTE display mode, origin time is one marker that could be In the RELATIVE mode, trace alignment is permitted. For example, to look at all marked P[en]wave first arrivals in relative time, one may try a

xlim A [en]10 A 10
plot1 RELATIVE

Note that if the ABSOLUTE plot mode had been used, then the display would have the P arrivals in absolute time and they would not be aligned.

Note that some combinations, such as xlim A [en]10 T0 +20, will not have the desired outcome with traces at different distances since the windows are different for each distance. The program will use the common window.

The option for CAL or GMT timesd was introduced 11 JAN 2005 to permit selection of time windows from very long time segments. The following are equivalent:

XLIM GMT 2005 001 01 02 03 456 GMT 2005 032 06 05 04 321
XLIM GMT 2005 001 01 02 03 456 CAL 2005 02 01 06 05 04 321
XLIM CAL 2005 01 01 01 02 03 456 GMT 2005 032 06 05 04 321
XLIM CAL 2005 01 01 02 03 456 CAL 2005 02 01 06 05 04 321

which cuts from January 1, 2005 01:02:03.456 to February 1, 2006 06:05:04.321

SAC COMPATIBILITY:

SEE ALSO

CUT, PLOT1, PLOTPK

XLIN

SUMMARY:

Linear x[en]axis for plot, plotpk

XLIN

INPUT:

DESCRIPTION:

DEFAULT:

Linear X[en]axis

SEE ALSO

YLIN, XLOG, YLOG, LINLIN, LINLOG

XLOG

SUMMARY:

Logarithmic x[en]axis for plot, plotpk

XLOG

INPUT:

DESCRIPTION:

DEFAULT:

Linear x[en]axis

SEE ALSO

XLIN, YLIN, YLOG, LINLIN

YGRID

SUMMARY:

Control y[en]axis grid

YGRID [ON | OFF] [Solid | Dotted] [Color int_value] [Minor ON | OFF]

INPUT:

ON : turn grid on
OFF : turn grid off
Solid : Use solid line
Dashed : Use dotted line
Minor : connect minor tics too if ON
Color int_value : Define the color for the grid. The figure frame will continue to be in black. Be careful to select a color not used for the trace.

DESCRIPTION:

This annotates the plots with a grid. Note that the Dotted option is takes more time to plot.

EXAMPLES:

DEFAULT

GRID OFF DOTTED COLOR 1030 MINOR OFF
[Note COLOR 1 is black, 2 red, 3 green, 4 blue, 1030 pale yellow]

SEE ALSO

GRID, XGRID, COLOR

YLIM

SUMMARY:

Define plot limits for y[en]axis

YLIM [ALL | OFF | Scale min max]

INPUT:

ALL : Plot all traces in the current window to the same scale
OFF : Each trace is autoscaled
Scale min max : User specified minimum and maximum values

DESCRIPTION:

This option permits all traces on a screen to be plotted using the same scale so that the relative differences in amplitude are obvious. Otherwise eash trace is plotted with its own scale.

The min max in scale can be in scientific notation, e.g.,
YLIM SCALE 1.0e[en]6 2.0e[en]5

EXAMPLES:

SAC COMPATIBILITY:

Sac permits a PM v to set +[en] v. It also permits setting the scaling of individual traces in a multitrace plot. GSAC does not permit either.

Note this command only affects the time series plots using the command plot1. It does not affect the spectra plot using the command plotsp or the record section plot prs.

DEFAULT:

YLIM OFF

SEE ALSO

YLIN

SUMMARY:

Linear x[en]yxis for plot, plotpk

YLIN

INPUT:

DESCRIPTION:

DEFAULT:

Linear y[en]axis

SEE ALSO

XLIN, XLOG, YLOG, LINLIN, LINLOG

YLOG

SUMMARY:

Logarithmic y[en]axis for plot, plotpk

YLOG

INPUT:

DESCRIPTION:

DEFAULT:

Linear y[en]axis

SEE ALSO

XLIN, YLIN, XLOG, LINLIN

APPENDIX C
GSAC LIBRARY

1. Introduction

This is a description of subroutines for manipulating SAC files. The SAC data file is either in machine dependent binary or in ASCII. Separate routines are required for input and output of these two basic types, but the structure of the files is identical.

The SAC file consists of a header, which is organized by real, integer and character groups. The routines developed are very similar in use to those described in the SAC manual.

This appendix will describe the C and FORTRAN interfaces for interacting with the SAC files. In each section the location of these programs and programs are described and an example of their use is given. The C and FORTRAN interfaces are very similar in appearance. However, unlike the original version of *sacio.a* of SAC, the C routines are true C routines and not interfaces for FORTRAN routines.

2. FORTRAN Routines

These routines are included in the file

PROGRAMSx.xx/SUBS/sacsubf.f

which also requires the support file *lgstr.f* located in

PROGRAMSx.xx/SUBS/lgstr.f

Here the *x.xx* refers to the current version of Computer Programs in Seismology, e.g., 3.30.

2.1 Routines

Low level Input/Output Routines:

call brsac (lun,maxpts,name,data,nerr) - read a binary SAC file
call brsach(lun,name,nerr) - read header of binary SAC file
call bwsac (lun,maxpts,name,data) - write a binary SAC file
call brsac2(lun,maxpts,name,x,y,npts) - read a binary SAC file
call bwsac2(lun,maxpts,name,x,y,npts) - write a binary SAC file
call arsac (lun,maxpts,name,data,nerr) - read an ASCII SAC file
call arsach(lun,name,nerr) - read header of ASCII SAC file
call awsac (lun,maxpts,name,data) - write an ASCII SAC file

brsac and **bwsac** read and write a time series consisting of equally spaced data points in the binary format appropriate for the current machine architecture. **arsac** and **awsac** read and write using an ASCII format.

bwsac2 and **brsac2** write and read, respectively, an unequally spaced time series consisting of **npts** (x,y) pairs. Note that if the dimensions are not correct, results will be in error because of the way that the two-dimension time series is stored.

lun is the FORTRAN logical unit for I/O
maxpts is the dimension of the array. On a read, no more than this value is read in.
npts is the number of points read in the time series, or the number to be written
is an alphanumeric string containing the SAC file name.
data is a real array of dimension **npts** used to store the seismic trace. On a read, if the SAC file contains more than **npts** points, then only the first **npts** points are read into memory. This ensures that array dimensions will not be exceeded.
nerr is used to indicate an error condition. **nerr = 0** indicates a successful operation.

High level Input/Output Routines:

call rsac1(infile,y,npts,btime,dt,maxpts,nerr)
call rsac2(ofile,y,npts,x,maxpts,nerr)
call wsac0(ofile,x,y,nerr)
call wsac1(ofile,y,npts,btime,dt,nerr) - write an equally spaced time series.
call wsac2(ofile,y,npts,x,nerr)

These routines set header values and then call the low level routines for binary I/O. Specifically, the routines accomplish the task through the following operations:

RSAC1 - Read an evenly spaced SAC file

```
c-----
c       logical unit 1 cannot be in use
c-----
        call brsac(1,maxpts,infile,y,nerr)
        call getnhv('NPTS    ',npts,ierr)
        if(npts.gt.maxpts)then
```

```
        npts = maxpts
    endif
    call getfhv('DELTA   ',dt  ,ierr)
    call getfhv('B       ',btime  ,ierr)
```

RSAC2 - Read an unevenly spaced SPAC file

```
c-----
c       logical unit 1 cannot be in use
c-----
      call brsac2(1,maxpts,ofile,x,y,npts)
```

WSAC0 - write a SAC file using current header values. If the file is evenly spaced, only the y-values are written.

```
    call getlhv('LEVEN   ',leven,nerr)
    call getnhv('NPTS    ',npts,nerr)
    call getfhv('DELTA   ',dt,nerr)
    call getfhv('B       ',b ,nerr)
    if(leven)then
        call wsac1(ofile,y,npts,b,dt,nerr)
    else
        call wsac2(ofile,x,npts,y,nerr)
    endif
```

WSAC1

```
    call scmxmn(y,npts,depmax,depmin,depmen,indmax,indmin)
    call setfhv('DEPMAX', depmax, ierr)
    call setfhv('DEPMIN', depmin, ierr)
    call setfhv('DEPMEN', depmen, ierr)
    call setnhv('NPTS    ',npts,nerr)
    call setfhv('DELTA   ',dt  ,nerr)
    call setfhv('B       ',btime  ,nerr)
    call setihv('IFTYPE  ','ITIME   ',nerr)
    e = btime + (npts -1 )*dt
    call setfhv('E       ',e     ,nerr)
    call setlhv('LEVEN   ',.true.,nerr)
    call setlhv('LOVROK  ',.true.,nerr)
    call setlhv('LCALDA  ',.true.,nerr)
    call bwsac(1,npts,ofile,y)
```

WSAC2

```
    call setnhv('NPTS    ',npts,nerr)
    call setihv('IFTYPE  ','ITIME   ',nerr)
    call setfhv('B       ',y(1)  ,nerr)
    call setfhv('E       ',y(npts)  ,nerr)
    call bwsac2(1,npts,ofile,x,y,npts)
```

Reading SAC header values

`call getfhv(strcmd,val,nerr)` - get floating point header value (F)
`call getihv(strcmd,strval,nerr)` - get enumerated header value (I)
`call getkhv(strcmd,cval,nerr)` - get character string header value (K)
`call getlhv(strcmd,lval,nerr)` - get logical header value (L)
`call getnhv(strcmd,ival,nerr)` - get integer header value (N)

Writing SAC header values

`call setfhv(strcmd,fval,nerr)` - set floating header value (F)
`call setihv(strcmd,strval,nerr)` - set enumerated header value (I)
`call setkhv(strcmd,cval,nerr)` - set character string header value (K)
`call setlhv(strcmd,lval,nerr)` - set logical header value (L)
`call setnhv(strcmd,ival,nerr)` - set integer header value (N)

> **`character strcmd*8`** is a keyword to indicate the value set.
> **`real fval, character cval*8, logical lval`** and **`integer ival`** are the floating point, character string, logical or integer values to be read or set.
> **`integer nerr`** is used to indicate an error condition. **`nerr = 0`** indicates a successful operation. An unsuccessful attempt is indicated by **`nerr = -1`**. This negative condition occurs if the keyword is not recognized.
>
> *Examples*

```
call setfhv('DIST',10.0,nerr)
call getfhv('DIST',dist,nerr)
call setnhv('NPTS', 256,nerr)
call getnhv('NPTS',npts,nerr)
call setkhv('KSTNM','ANMO',nerr)
call getkhv('KCMPNM',cmpstr,nerr)
call setihv('IFTYPE','ITIME',nerr)
call setihv('IDEP',string,nerr)
```

Initializing header

`call newhdr()`
`call inihdr()` - initialize a new SAC header with default values. These two routines are equivalent.

> Both routines accomplish the same task. First real, integer and character strings of the SAC header are set equal to **`-12345.`**, **`-12345`** and "**`-12345`**", respectively. The following integer header value are set: *ihdr(7)=6, hdr(8)=0, ihdr(9)=0, ihdr(36)=0, ihdr(37)=0, ihdr(38)=0, ihdr(39)=0, ihdr(40)=0*. The first defines the NVHDR to the current SAC file format, the second sets NINF or NORID=0, the third sets NHST or NEVID=0, while the last five set the five logical header values

LEVEN, LPSPOL, LOVROK, LCALDA and LHDR5 to FALSE.

Note that routines do very little to the header. A user program must manually define the header values for a trace. When use in a program, the **`call newhdr()`** erases all information in the current header that may be in memory from a previous read or write.

Time Routines

`call katime(ihour,imin,isec,imsec,nctime,kktime,nerr)` - convert four integer fields representing hour, minute, second and millisecond to a string of the form HH:MM:SS.SSS
`integer ihour` - hour
`integer imin` - minute
`integer isec` - second
`integer imsec` - millisecond
`integer nctime` - number of characters in the time string. Must at least be 12 in size. The string is initially blank filled.
`character kktime*(*)` - character string that is returned.

`call kadate(iyear,ijday,ncdate,kkdate,nerr)` - convert the two integer fields representing year and day of year to a string of the form MMM DD (JJJ), YYYY
`iyear` - year
`ijday` - day of year
`integer ncdate` - number of characters in the time string. Must at least be 18 in size. The string is initially blank filled.
`character kkdate*(*)` - character string that is returned.

`call etoh(epoch,date,str,doy,year,month,day,hour,minute,second)` - convert from epoch time to human time
`real*8 epoch` - number of seconds relative to January 1, 1970 00:00:00.000.
`integer date` - date in the form 1998273, a concatenation of the year and the eday of year
`character str` - must be at least 32 characters long in size
`integer doy` - day of year
`integer year` - year
`integer month` - month as a numeral
`integer day` - day of month
`integer hour` - hour
`integer minute` - minute
`real second` - second

`call mnthdy(year,doy,month,day)` - given a YEAR and DOY, return MONTH and DAY
`integer year` - year
`integer doy` - day of year

`integer month` - month as a numeral
`integer day` - day of month

`call htoe(year,month,day,hour,minute,second,epoch)` - convert human time to epoch time
`integer date` - date in the form 1998273, a concatenation of the year and the eday of year
`integer year` - year
`integer month` - month as a numeral
`integer day` - day of month
`integer hour` - hour
`integer minute` - minute
`real second` - second
`real*8 epoch` - number of seconds relative to January 1, 1970 00:00:00.000.

Trace parameters

`call scmxmn(x,npts,depmax,depmin,depmen,indmax,indmin)` - determine the trace characteristics

`real x` - array of **`npts`** to be examined
`integer npts` - number of points in the array
`depmax` - mean value of trace
`depmin` - minimum value of trace
`depmax` - maximum value of trace
`indmax` - index of the maximum value of the trace. C notation is used. 0 (zero) is the first data point.
`indmin` - index of the minimum value of the trace. C notation is used. 0 (zero) is the first data point.

2.2 Sample program

The following is a simple subroutine to create a SAC file. The is called as
call doout(xarray,npts,'SACFILE.SAC', dt, .true.)

```
subroutine doout(out,npts,fname,dt,dobin)
 integer npts
 character fname*(*)
 real out(npts), dt
 logical  dobin

 real depmax, depmin, depmen
```

```
call scmxmn(out,NPTS,depmax,depmin,depmen,indmax,indmin)
call newhdr()
call setnhv('NPTS', npts, nerr)
call setfhv('DELTA', dt, nerr)
call setfhv('DEPMIN  ',depmin,nerr)
call setfhv('DEPMAX  ',depmax,nerr)
call setfhv('DEPMEN  ',depmen,nerr)
call setfhv('B', 0.0, nerr)
call setfhv('E', ( npts-1)*dt, nerr)
call setlhv('LEVEN   ',.true.,nerr)
call setlhv('LSPOL   ',.true.,nerr)
call setlhv('LOVROK  ',.true.,nerr)
call setlhv('LCALDA  ',.false.,nerr)
call setihv('IFTYPE   ','ITIME   ',nerr)
call setihv('IZTYPE   ','IB      ',nerr)
if(dobin)then
        call bwsac(1,NPTS,fname,out)
else
        call awsac(1,NPTS,fname,out)
endif
return
end
```

Note that the header values required to define an evenly spaced time series.

3. C Routines

These routines are included in the files

PROGRAMSx.xx/SUBS/sacsubc.c
PROGRAMSx.xx/SUBS/sacsubc.h
PROGRAMSx.xx/SUBS/csstime.c
PROGRAMSx.xx/SUBS/csstime.h

Here the *x.xx* refers to the current version of Computer Programs in Seismology, e.g., 3.30.

3.1 Sac Routines

Low level Input/Output Routines:

`void brsac (int npts,char *name,float **data,int *nerr);` -

read a binary SAC file
`void brsach(char *name,int *nerr);` - read header of binary SAC file
`void arsac (int npts,char *name,float **data,int *nerr);` - read an ASCII SAC file
`void arsach(char *name,int *nerr);` - read header of ASCII SAC file
`void bwsac (int npts,char *name,float *data);` - write a binary SAC file
`void awsac (int npts,char *name,float *data);` - write an ASCII SAC file

`npts` is the number of points read in the time series, or the number to be written
is an alphanumeric string containing the SAC file name.
`data` is a real array of dimension **`npts`** used to store the seismic trace. On a read, it is assumed that data is a pointer to NULL, and then the memory space is allocated.
`nerr` is used to indicate an error condition. **`nerr = 0`** indicates a successful operation.

See the example in §3.3 on how these are used.

High level IO routines

Routines similar to the FORTRAN **`rsac1`**, **`rsac2`**, **`wsac0`**, **`wsac1`**, **`wsac2`**, and the lower level **`brsac2`** and **`bwsac2`** had not been implemented. To do so is trivial. Just use the FORTRAN code a a guide.

Reading SAC header values

`void getfhv(char *strcmd,float *fval,int *nerr);` - get real header value (R)
`void getihv(char *strcmd,char *strval,int *nerr);` - get enumerated string header value (K)
`void getkhv(char *strcmd,char *cval,int *nerr);` - get character string header value (K)
`void getlhv(char *strcmd,int *lval,int *nerr);` - get logical header value (L)
`void getnhv(char *strcmd,int *ival,int *nerr);` - get integer header value (N)

Writing SAC header values

`void setfhv(char *strcmd,float  fval,int *nerr);` - set floating header value (R)
`void setihv(char *strcmd,char *strval,int *nerr);` - set enumerated string header value (K)
`void setkhv(char *strcmd,char *cval,int *nerr);` - set character

string header value (C)

`void setlhv(char *strcmd,int lval,int *nerr);` - set logical header value (I)

`void setnhv(char *strcmd,int  ival,int *nerr);` - set integer header value (F)

`character strcmd*8` is a keyword to indicate the value set.
`real fval, character cval*8, logical lval` and **`integer ival`** are the floating point, character string, logical or integer values to be read or set.
`integer nerr` is used to indicate an error condition. **`nerr = 0`** indicates a successful operation. An unsuccessful attempt is indicated by **`nerr = -1`**. This negative condition occurs if the keyword is not recognized.

Initializing header

`void newhdr(void)`
`void inihdr(void)` - initialize a new SAC header with default values. These two routines are equivalent.

Both routines accomplish the same task. First real, integer and character strings of the SAC header are set equal to **`-12345.`**, **`-12345`** and "**`-12345`**", respectively. The following integer header value are set: *ihdr(7)=6, hdr(8)=0, ihdr(9)=0, ihdr(36)=0, ihdr(37)=0, ihdr(38)=0, ihdr(39)=0, ihdr(40)=0*. The first defines the NVHDR to the current SAC file format, the second sets NINF or NORID=0, the third sets NHST or NEVID=0, while the last five set the five logical header values LEVEN, LPSPOL, LOVROK, LCALDA and LHDR5 to FALSE.

Note that routines do very little to the header. A user program must manually define the header values for a trace. When use in a program, the **`newhdr()`** erases all information in the current header that may be in memory from a previous read or write.

Trace parameters

`scmxmn(x,npts,depmax,depmin,depmen,indmax,indmin)` - determine the trace characteristics

`real x` - array of **`npts`** to be examined
`integer npts` - number of points in the array
`depmax` - mean value of trace
`depmin` - minimum value of trace
`depmax` - maximum value of trace
`indmax` - index of the maximum value of the trace. C notation is used. 0 (zero) is the first data point.
`indmin` - index of the minimum value of the trace. C notation is used. 0 (zero) is the first data point.

3.2 Time routines

Time Routines

#include "csstime.h"

void htoe(struct date_time *dt) ; - convert from human to epoch
void month_day(struct date_time *dt) ; - from epoch fillin monty/day
void etoh(struct date_time *dt) ; - epoch to human
void mdtodate(struct date_time *dt); - from epoch to YYYY DOY
void timestr(struct date_time *dt, char *str) ; - create an ASCII time string of the form - 1999 12 31 23:59:59.999
void timeprintstr(struct date_time *dt,char *str) ; - create an ASCII time string of the form: epoch jday mon 12,1999 23:59:59.999

The time routines use the structure *date_fime* defined as

```
struct date_time{
        double epoch;
        long date;
        int year;
        int month;
        char mname[4];
        int day;
        long doy;
        int hour;
        int minute;
        float second;
};
```

where
epoch - number of seconds relative of 1970 01 01 00 00 00.000
date - year - day of day combination of the form YYYYDDD
year -
month -
mname - 4 character string for month name, e.g. "JAN "
day - deay on the month
doy - day of year
hour
minute
second

3.3 Sample program

A sample use of the SAC IO and time routines is given here. Data are read from a SAC file, the header is interrogated, and the time routines are used.

```
#include <stdio.h>
#include <stdlib.h>
#include "csstime.h"
#include "sacsubc.h"


    int nerr, nzyear, nzjday, nzhour, nzmin, nzsec, nzmsec;
    char kstnm[9];   /* 8 character plus the null character */
    struct date_time dt_refer;
    struct date_time dt_origin;
    int nberr;
    float *data; /* brsac uses calloc to define the storage */
    char *fname = "sacinput.sac";
    char ostr[80];
    float o;

    /* open a SAC file for access to header information */
    brsac(MAXSACARR,fname,  &data, &nberr);
    getfhv("O",&o,&nerr);
    getnhv("NZYEAR",&nzyear,&nerr);
    getnhv("NZJDAY",&nzjday,&nerr);
    getnhv("NZHOUR",&nzhour,&nerr);
    getnhv("NZMIN",&nzmin,&nerr);
    getnhv("NZSEC",&nzsec,&nerr);
    getnhv("NZMSEC",&nzmsec,&nerr);
    getkhv("KSTNM",kstnm,&nerr);
    /* convert the reference to epoch time */
    dt_refer.date = 1000L*nzyear + nzjday;
    dt_refer.hour = nzhour;
    dt_refer.minute = nzmin;
    dt_refer.second = (float)nzsec + (float)nzmsec/1000.0;
    /* convert to epoch */
    htoe(&dt_refer);
    etoh(&dt_refer);
    timeprintstr(&dt_refer,ostr);
    /* create an entry for origin time */
    dt_origin.epoch = dt_refer.epoch + o;
    etoh(&dt_origin);
    ...
    free(data);     /* at end of program free allocated memory */
```

This snippet opens the SAC file *sacinput.sac* and obtains a number of header values. The human time is converted to epoch time through **`htoe`** and then other entries in the header, such as month and day of month are set through the **`etoh`** call.

Next the absolute epoch time of the origin time is set by adding the header value "O" to the epoch reference time. Finally the call to **`etoh`** sets up all other origin structure values.

APPENDIX D
SAC HEADER

1. Introduction

The SAC file consists of a header, with real, integer and character components followed by the time series. The header contains all information about the time series, such as number of points and sample interval, as well as other supplementary information, such as event station location.

The header consists of these fields in the following order:

70 4-byte floating point numbers
40 4 byte integers
24 8 byte character strings

2. Header definition

The table given here lists the header values in the order in which they appear. In the table Position indicates the postion of the header value in C notation (add 1 for the FORTRAN position). The Type symbol is F, I, L or K for a float, integer, logical or character string. In the header, integers with values or 1 or 0 are used to represent TRUE or FALSE conditions. The character string is a gouping of 8 or 16 characters, but does not have the '\0' that is typical of C strings. The Keyword is the C string that would be used in the **`setfhv()`** or **`getfhv()`** or similar call. In FORTRAN, these keywords would be defined by single quotes instead of double quotes. The Access Routine indicates the particular **`get set`** routine that is called.

Position	Type	Keyword	Access Routine	Description
0	F	**"DELTA"**	g[s]etfhv()	Sample interval in sec
1	F	**"DEPMIN"**	g[s]etfhv()	Minimum value of dependent variable
2	F	**"DEPMAX"**	g[s]etfhv()	Maximum value of dependent variable
3	F	**"SCALE"**	g[s]etfhv()	Multiplying factor (not used)
4	F	**"ODELTA"**	g[s]etfhv()	Observed sample value if not same as nominal
5	F	**"B "**	g[s]etfhv()	Initial value of independent variable
6	F	**"E "**	g[s]etfhv()	End value of independent variable
7	F	**"O "**	g[s]etfhv()	Origin time relative to reference time
8	F	**"A "**	g[s]etfhv()	First arrival time relative to reference time
9	F	**"FMT"**	g[s]etfhv()	Internal
10	F	**"T0"**	g[s]etfhv()	User defined pick time 0 relative to ref
11	F	**"T1"**	g[s]etfhv()	User defined pick time 1 relative to ref
12	F	**"T2"**	g[s]etfhv()	User defined pick time 2 relative to ref

13	F	**"T3"**	g[s]etfhv()	User defined pick time 3 relative to ref
14	F	**"T4"**	g[s]etfhv()	User defined pick time 4 relative to ref
15	F	**"T5"**	g[s]etfhv()	User defined pick time 5 relative to ref
16	F	**"T6"**	g[s]etfhv()	User defined pick time 6 relative to ref
17	F	**"T7"**	g[s]etfhv()	User defined pick time 7 relative to ref
18	F	**"T8"**	g[s]etfhv()	User defined pick time 8 relative to ref
19	F	**"T9"**	g[s]etfhv()	User defined pick time 9 relative to ref
20	F	**"F "**	g[s]etfhv()	End time of event relative to reference
21	F	**"RESP0"**	g[s]etfhv()	Instrument response parameters (not used)
22	F	**"RESP1"**	g[s]etfhv()	Instrument response parameters (not used)
23	F	**"RESP2"**	g[s]etfhv()	Instrument response parameters (not used)
24	F	**"RESP3"**	g[s]etfhv()	Instrument response parameters (not used)
25	F	**"RESP4"**	g[s]etfhv()	Instrument response parameters (not used)
26	F	**"RESP5"**	g[s]etfhv()	Instrument response parameters (not used)
27	F	**"RESP6"**	g[s]etfhv()	Instrument response parameters (not used)
28	F	**"RESP7"**	g[s]etfhv()	Instrument response parameters (not used)
29	F	**"RESP8"**	g[s]etfhv()	Instrument response parameters (not used)
30	F	**"RESP9"**	g[s]etfhv()	Instrument response parameters (not used)
31	F	**"STLA"**	g[s]etfhv()	Station latitude, °N is positive
32	F	**"STLO"**	g[s]etfhv()	Station longitude, °E is positive
33	F	**"STEL"**	g[s]etfhv()	Station elevation, meters (not used)
34	F	**"STDP"**	g[s]etfhv()	Station depth below surface, meters (not used)
35	F	**"EVLA"**	g[s]etfhv()	Event latitude, °N is positive
36	F	**"EVLO"**	g[s]etfhv()	Event longitude, °W is positive
37	F	**"EVEL"**	g[s]etfhv()	Event elevation, meters
38	F	**"EVDP"**	g[s]etfhv()	Event depth below surface, (originally meters) Computer Programs uses KILOMETERS
39	F	**"FHDR40"**	g[s]etfhv()	Internal
40	F	**"USER0"**	g[s]etfhv()	User defined storage 0
41	F	**"USER1"**	g[s]etfhv()	User defined storage 1
42	F	**"USER2"**	g[s]etfhv()	User defined storage 2
43	F	**"USER3"**	g[s]etfhv()	User defined storage 3
44	F	**"USER4"**	g[s]etfhv()	User defined storage 4
45	F	**"USER5"**	g[s]etfhv()	User defined storage 5
46	F	**"USER6"**	g[s]etfhv()	User defined storage 6
47	F	**"USER7"**	g[s]etfhv()	User defined storage 7
48	F	**"USER8"**	g[s]etfhv()	User defined storage 8
49	F	**"USER9"**	g[s]etfhv()	User defined storage 9
50	F	**"DIST"**	g[s]etfhv()	Station to event distance, km
51	F	**"AZ"**	g[s]etfhv()	Event to station azimuth, degrees
52	F	**"BAZ"**	g[s]etfhv()	Station to event azimuth, degrees
53	F	**"GCARC"**	g[s]etfhv()	Station to event distance, degrees
54	F	**"SB"**	g[s]etfhv()	Internal
55	F	**"SDELTA"**	g[s]etfhv()	Internal
56	F	**"DEPMEN"**	g[s]etfhv()	Mean value of dependent variable
57	F	**"CMPAZ"**	g[s]etfhv()	Component azimuth, degrees east of north
58	F	**"CMPINC"**	g[s]etfhv()	Component incident, degrees from vertical
59	F	**"XMINIMUM"**	g[s]etfhv()	Internal
60	F	**"XMAXIMUM"**	g[s]etfhv()	Internal
61	F	**"YMINIMUM"**	g[s]etfhv()	Internal
62	F	**"YMAXIMUM"**	g[s]etfhv()	Internal
63	F	**"ADJTM"**	g[s]etfhv()	Internal
64	F	**"FHDR65"**	g[s]etfhv()	Internal

65	F	**"FHDR66"**	g[s]etfhv()	Internal
66	F	**"FHDR67"**	g[s]etfhv()	Internal
67	F	**"FHDR68"**	g[s]etfhv()	Internal
68	F	**"FHDR69"**	g[s]etfhv()	Internal
69	F	**"FHDR70"**	g[s]etfhv()	Internal
0	I	**"NZYEAR"**	g[s]etnhv()	Year of reference time, e.g., 1970
1	I	**"NZJDAY"**	g[s]etnhv()	Day of year, e.g., 59 = February 28
2	I	**"NZHOUR"**	g[s]etnhv()	GMT hour
3	I	**"NZMIN"**	g[s]etnhv()	GMT min
4	I	**"NZSEC"**	g[s]etnhv()	GMT sec
5	I	**"NZMSEC"**	g[s]etnhv()	GMT millisecond
6	I	**"NVHDR"**	g[s]etnhv()	Header version number, always 6
7	I	**"NINF"**	g[s]etnhv()	Internal
8	I	**"NHST"**	g[s]etnhv()	Internal
9	I	**"NPTS"**	g[s]etnhv()	Number of points in time series
10	I	**"NSNPTS"**	g[s]etnhv()	Internal
11	I	**"NSN"**	g[s]etnhv()	Internal
12	I	**"NXSIZE"**	g[s]etnhv()	Internal
13	I	**"NYSIZE"**	g[s]etnhv()	Internal
14	I	**"NHDR15"**	g[s]etnhv()	Internal
15	I	**"IFTYPE"**	g[s]etnhv()	*Enumerated Value -see below*
16	I	**"IDEP"**	g[s]etnhv()	*Enumerated Value - see below*
17	I	**"IZTYPE"**	g[s]etnhv()	*Enumerated Value - see below*
18	I	**"IHDR4"**	g[s]etnhv()	Internal
19	I	**"IINST"**	g[s]etnhv()	*Enumerated Value for instrument (not used)*
20	I	**"ISTREG"**	g[s]etnhv()	*Enumerated Value for station region (not used)*
21	I	**"IEVREG"**	g[s]etnhv()	*Enumerated Value for event region (not used)*
22	I	**"IEVTYP"**	g[s]etnhv()	*Enumerated Value - see below*
23	I	**"IQUAL"**	g[s]etnhv()	*Enumerated Value for data quality (see below)*
24	I	**"ISYNTH"**	g[s]etnhv()	*Enumerated Value - see below*
25	I	**"IHDR11"**	g[s]etnhv()	Internal
26	I	**"IHDR12"**	g[s]etnhv()	Internal
27	I	**"IHDR13"**	g[s]etnhv()	Internal
28	I	**"IHDR14"**	g[s]etnhv()	Internal
29	I	**"IHDR15"**	g[s]etnhv()	Internal
30	I	**"IHDR16"**	g[s]etnhv()	Internal
31	I	**"IHDR17"**	g[s]etnhv()	Internal
32	I	**"IHDR18"**	g[s]etnhv()	Internal
33	I	**"IHDR19"**	g[s]etnhv()	Internal
34	I	**"IHDR20"**	g[s]etnhv()	Internal
35	L	**"LEVEN"**	g[s]etlhv()	TRUE (1) if evenly spaced
36	L	**"LPSPOL"**	g[s]etlhv()	TRUE if stations have positive polarity, left hand rule
37	L	**"LOVROK"**	g[s]etlhv()	TRUE if can overwrite this file
38	L	**"LCALDA"**	g[s]etlhv()	TRUE if DIST, AZ, BZ, GCARC to be calculated
39	L	**"LHDR5"**	g[s]etlhv()	Internal
0	K	**"KSTNM"**	g[s]etkhv()	Station Name - 8 characters
1	K	**"KEVNM"**	g[s]etkhv()	Event Name - first 8 characters
2	K	**"KEVNMC"**	g[s]etkhv()	Event Name - last 8 characters
3	K	**"KHOLE"**	g[s]etkhv()	Event identification
4	K	**"KO"**	g[s]etkhv()	Event origin time identification

5	K	**"KA"**	g[s]etkhv()	First arrival identification
6	K	**"KT0"**	g[s]etkhv()	User defined pick identification 0
7	K	**"KT1"**	g[s]etkhv()	User defined pick identification 1
8	K	**"KT2"**	g[s]etkhv()	User defined pick identification 2
9	K	**"KT3"**	g[s]etkhv()	User defined pick identification 3
10	K	**"KT4"**	g[s]etkhv()	User defined pick identification 4
11	K	**"KT5"**	g[s]etkhv()	User defined pick identification 5
12	K	**"KT6"**	g[s]etkhv()	User defined pick identification 6
13	K	**"KT7"**	g[s]etkhv()	User defined pick identification 7
14	K	**"KT8"**	g[s]etkhv()	User defined pick identification 8
15	K	**"KT9"**	g[s]etkhv()	User defined pick identification 9
16	K	**"KF"**	g[s]etkhv()	End identification
17	K	**"KUSER0"**	g[s]etkhv()	User comment 0
18	K	**"KUSER1"**	g[s]etkhv()	User comment 1
19	K	**"KUSER2"**	g[s]etkhv()	User comment 2
20	K	**"KCMPNM"**	g[s]etkhv()	Component name
21	K	**"KNETWK"**	g[s]etkhv()	Network Name
22	K	**"KDATRD"**	g[s]etkhv()	Date of data conversion
23	K	**"KINST"**	g[s]etkhv()	Name of instrument

The routines **getihv()** and **setihv()** are used to set the enumerated values, which are mappings of strings into integers stored in the integer header. The syntax is **getihv(strcmd,strval,nerr)**. This table gives the enumerated values and keywords for the set command:

"IDEP"		Type of dependent variable
	"IUNKN"	Unknown
	"IDISP"	Displacement in nm
	"IVEL"	Velocity in nm/s
	"IACC"	Acceleration in nm/s/s
	"IVOLTS"	Volts
"IFTYPE"		File content
	"ITIME"	Time series (ONLY THING SUPPORTED here)
	"IRLIM"	Real - imaginary spectral file
	"IAMPH"	Amplitude - phase spectral file
	"IXY"	General x versus y
"IZTYPE"		Reference time equivalence
	"IUNKN"	unknown
	"IB"	Begin time
	"IDAY"	Midnight of reference GMT day
	"IO"	Event origin time
	"IA"	First arrival time
	"ITn"	User defined time n=[0,9]
"IEVTYP"		Event type

	"IUNKN"	unknown
	"INUCL"	Nuclear
	"IPREN"	Pre -Nuclear
	"IPOSTN"	Post - Nuclear
	"IQUAKE"	Earthquake
	"IPREQ"	Foreshock
	"IPOSTQ"	Aftershock
	"ICHEM"	Chemical
	"IOTHER"	Other
"ISYNTH"		Synthetic data flag
	"IRLDATA"	Real data

APPENDIX E
ADDING ROUTINES TO GSAC

1. Introduction

The purpose of the GSAC initiative is to permit a community development of tools for earthquake seismology. The `gsac` program is a primary product. As such, the purpsoe of this Appendix is to outline the procedure for adding new functionality to the program.

The example here implements a simple trace manipulation tool commonly used in exploration geophysics for presentation graphics - an automatic gain control. Thus concept behind this tool traces its origins to electronic circuits introducted at the beginning of the radio era in the 1930’s. To overcome variable volume levels an automatic volume control (AVC) was used to enhance weak audio and to reduce strong audio. This concept was introduced in reflection seismology at the time of analog data acquisition and recording to emphasize later, low amplitude arrivals and to suppress the earlier high amplitude arrivals. When applied to digital trace data, this becomes known as a DAGC or DAVC processing procedure.

Mathemathically, given a trace $x(t)$, we wish to create a gain value, $g(t)$, at each sample point so that we can modify the original trace to form $g(t)\,x(t)$. To determine the $g(t)$ from the trace, we use a moving average of the $|x(t)|$. We start by defining an averaging window, W, in seconds. For assumed equally sampled data with interval Δt, we will define the averaging window to be M samples where M is an odd number fromed from the ratio $W/\Delta t$. After being careful about the initial and final values of the $g(t)$, we define it as

$$g(n) = \begin{cases} \dfrac{1}{M} S_1 & 0 \le n \le M/2 \\ \dfrac{1}{M} \displaystyle\sum_{k=-M/2}^{M/2} |x(n+k)| & M/2+1 \le n < N - M/2 \\ \dfrac{1}{M} S_2 & N - M/2 \le n < N \end{cases}$$

where we have used the array notation that $g(n) = g(n\Delta t)$ and $x(k) = x(k\Delta t)$ and N is the total number of data points. The beginning and end points use a taper on the following values: $S_1 = \sum_{k=0}^{M-1} |x(k)|$ and $S_2 = \sum_{k=N-M}^{N-1} |x(k)|$.

2. Defining Prototypes

The source directory, VOLVIII/src, has a shell script that will do the following: create a prototype of the routine to actually do the AGC, to set up the online help, the documentation in this tutorial, and to implement the command parsing. The shell script is called MAKEPROTO. We will use this program with the three required arguments:

```
MAKEPROTO dagc AGC "AGC traces"
```

As a result of this command the following changes were made in the src and HELP directories:

```
VOLVIII/src/gsac_docommand.h                     Modified
VOLVIII/src/gsac_command.h                       Modified
VOLVIII/src/gsac_dagc.c                          Created
VOLVIII/src/Makefile                             Modified
VOLVIII/src/Makefile.W32                         Modified
VOLVIII/src/Makefile.SOL                         Modified
VOLVIII/src/Makefile.OSX                         Modified
VOLVIII/src/Makefile.OSF                         Modified
VOLVIII/src/Makefile.LNX                         Modified
VOLVIII/src/Makefile.CYG                         Modified

VOLVIII/HELP/AGC.trf                             Created
VOLVIII/HELP/HELP.trf                            Modified
VOLVIII/HELP/MAKEDOC                             Modified
VOLVIII/HELP/MAKEINC                             Modified
```

The Makefiles are modified to include the routine gsac_dagc into the compilation of gsac.

gsac_command.h is defines the relationship between command names and the user input line. This file was changed to include the lines:

```
#define AGC 47

{AGC, "AGC", &gsac_set_param_dagc, &gsac_exec_dagc, help_dagc},
```

The second line states that when the initial word of an input string is AGC or agc, that the program will further examine the input line by using the routine *gsac_set_param_dagc* and will apply the AGC filter to traces in memory using the *gsac_exec_dagc* routine. Further, the structure *help_dagc* contains the online help.

gsac_docommand.h defines the function prototypes required by the compiler:

```
void gsac_set_param_dagc(int ncmd, char **cmdstr);
```

```
void gsac_exec_dagc(void);
```

Before compiling, we must update the online help file *gsac_help.h* in the *src* directory. This is done by moving to the *VOLVIII/HELP* help directory. Several things will be done:

1. Edit the *HELP.trf* to move the AGC command to the top so that it appears in alphabetical order.
2. Edit the *AGC.trf* to document the use of the routine. The source is GROFF. Just look at other commands for syntax.
3. Create the new header by executing the command

```
MAKEINC
```

After veryfying that gsac_help.h looks OK, copy this to the source directory by

```
cp gsac_help.h ../src
```

4 Create the command documentation for this tutorial and install in to *DOC/GSAC.TRF*.

```
MAKEDOC
```

Now return to the *src* directory and compile:

```
cd ../src
make clean
make gsac
```

At this stage the input command `AGC W 2.0` will do nothing. Since we have not implemented the algorithm yet. However, we can determine of the parsing of the input and the online help are correctly implemented:

```
[rbh@crust src]$ gsac
GSAC - Computer Programs in Seismology [V0.1 12 APR 2004]
   Copyright 2004 R. B. Herrmann
GSAC> AGC

GSAC> q

[rbh@crust src]$ gsac
GSAC - Computer Programs in Seismology [V0.1 12 APR 2004]
   Copyright 2004 R. B. Herrmann
GSAC> agc w 2.0
w 2.0
GSAC> help agc
```

```
GSAC Command Reference Manual                                AGC

SUMMARY:
     AGC traces

     AGC W window

INPUT:
     W window  : Define the averaging window in seconds

DESCRIPTION:
     This  routine  applies an AGC operator to the trace such that the
     mean signal amplitude is near unity. The operator is obtained  by
     using a running average of the absolute value of the trace.
EXAMPLES:

HEADER VALUES SET:
     DEPMAX, DEPMIN and DEPMEN are updated.

SEE ALSO
GSAC> quit

[rbh@crust src]$
```

3. Implementing the DAGC

We now focus on the file *gsac_dagc.c* in the soruce directory. This defines the place holders and currently just repeats the command line. There are two steps in the implementation - setting the control parameters based on the command line input and applying the algorithm to the traces in memory. The prototype file has the following:

```
#include    <stdio.h>
#include    "gsac_docommand.h"
#include        "gsac.h"
#include        "gsac_plot.h"
#include        "gsac_sac.h"
#include        "gsac_arg.h"
#include        "gsac_sachdr.h"

extern struct sacfile_ *sacdata;
extern int *sortptr;


void gsac_set_param_dagc(int ncmd, char **cmdstr)
```

```
{
            int i;
            for(i=1; i < ncmd; i++)
            printf("%s ",cmdstr[i]);
            printf("0);
            /* note when the testrg routine is used, if the argument is
            NO then you must use internal variables to define the
            state of the operation - if you use YES, then things are
            not changed until the input is proven correct. An exmple of
            this concept with YES is the following:
            Assume we wish aa LP filter with fc 1 np 2 p 1
            If we enter  fc 2 np2   there is a syntax error and we
            should not chnge the fc since the np2 is wrong. One way to
            do this in the code would be to do two calls

            if(testarc,ncmd, cmdstr, cmdargs, YES) is OK
            then
            testarc,ncmd, cmdstr, cmdargs, NO)
            */


}

void gsac_exec_dagc(void)
{
}
```

3.1 Parsing command line parameters

The command line consist of the command, *AGC*, the control flag, *W* for the window in seconds, and the window in seconds, which is represented as a floating point number. This window must be placed in a global area so that the command execution can see the value. In addition, we will also require that the program remember a previous value so that we can so do some thing as follows:

```
GSAC> r traces
GSAC> agc w 1.0
GSAC> r more moretraces
GSAC> agc
```

We will look at the *gsac_set_param_ylim* in *gsac_ylim.c* because it has a somewhat similar syntax. The net code added to the prototype is highlighted in the color red in the following listing: highlighted in the color red in the following listingp

```
#include    <stdio.h>
#include    "gsac_docommand.h"
#include        "gsac.h"
#include        "gsac_plot.h"
#include        "gsac_sac.h"
#include        "gsac_arg.h"
#include        "gsac_sachdr.h"

extern struct sacfile_ *sacdata;
extern int *sortptr;

#define AGCWINDOW 1

struct arghdr agcarg[] = {
{AGCWINDOW, "W" , RHDR, 0, 1, NO, "W window(sec) "},
{0      , ""      , IHDR, 0, 0, NO, ""}
};

static float agc_window = 0.0;
static float agc_real[1];

void gsac_set_param_dagc(int ncmd, char **cmdstr)
{
            int i;
            if(ncmd == 1)
            return;
            /* is the command syntax correct ? Also reset */
            if(testarg(ncmd, cmdstr, agcarg, NO))
            return;
            for(i=0 ; agcarg[i].key[0] != ' ' ; i++){
            /* check for special commands */
            if(agcarg[i].used > 0){
            if(agcarg[i].id == AGCWINDOW){
            getargr(ncmd, cmdstr, agcarg[i].key, agcarg[i].narg, i, agc_real);
            agc_window  = agc_real[0];
            }
            }
            }
}

void gsac_exec_dagc(void)
{
}
```

The line *if(ncmd == 1)* indicates that if the command line does not specify a new window, then use the previous value of the window.

The *testarg* compares everything on the input line to the syntax definition. We only look for the combination of the *W* flag and the window length, which must be a floating point or real number. If this fails, the command tries to pring a syntax message. If this test succeeds, then we finally set the processing parameters according to the input line.

We can test the operation at this point, by recompiling with `make gsac`, and then running **`gsac`**. Here we see if it accepts the command and if it rejects other input.

```
GSAC - Computer Programs in Seismology [V0.1 12 APR 2004]
   Copyright 2004 R. B. Herrmann
GSAC> agc w 3
GSAC> agc b
Error in agc: incorrect option
    agc b
        ^
GSAC> agc w ten
Error in agc: W windows
    agc w ten
          ^^^
GSAC>
```

Note the use of the "ˆ" to indicate the position of the input error.

3.2 Implementing the command

This routine will work with the traces in memory, modify them, and then update the *DEPMAX*, *DEPMIN* and *DEPMEN* header values. The simplest routine that does something like this is *gsac_exec_add* in *gsac_add.c*. This will be the building block.

The final routine is as follows (the color red will not be used here since everything changed):

```
void gsac_doagc(float *x, int n, float dt, float win );
void gsac_exec_dagc(void)
{

            int i, k, ntrc, npts;
            float depmax, depmin, depmen;
            float delta;
            /* if there are no traces return */
            ntrc = gsac_control.number_itraces;
            if(ntrc < 1)
            return;

            for ( k=0 ; k < ntrc ; k ++){
```

```
                npts  = sacdata[k].sachdr.ihdr[H_NPTS];
                delta = sacdata[k].sachdr.rhdr[H_DELTA];

                if(npts > 0){
                gsac_doagc(sacdata[k].sac_data, npts, delta,  agc_window );
                getmxmn(sacdata[k].sac_data, npts,&depmax, &depmin, &depmen);
                sacdata[k].sachdr.rhdr[H_DEPMIN] = depmin;
                sacdata[k].sachdr.rhdr[H_DEPMAX] = depmax;
                sacdata[k].sachdr.rhdr[H_DEPMEN] = depmen;
                }
                }
}

void gsac_doagc(float *x,  int n, float dt, float win )
{
                /* x - array from which to form the AGC
                 * g - gain factor from trace
                 * n - number of points in x
                 * dt - sampling interval of trace
                 * win - smoothing window for agc
                 * */
                /*
                 * m - length of smoothing window, odd number
                 * g - gain factor from trace
                 * */
                int k, m, m2, i;
                float sum;
                float S1;
                float lower_bound;
                float max;
                float *g;
                /* safety */
                if(n < 2)
                return;
                /* create the gain array */
                if((g = (float *)calloc(n, sizeof(float))) == (float *)NULL){
                printf("Cannot allocate memory to create gain array in agc0);
                return;
                }
                /* define the smoothing width in samples taking care to
                 * address extreme values. Also make the width an odd number
                 * */
                m = win/dt;
                if( m > n)
                m = n/2;
                if(m < 3 )
                m = 3;
```

```
            m += (m%2 -1 );
            m2 = m / 2;
            /* we define the operator as a running mean over the trace
             * for efficiency we note that for the next value we just have to add
             * a new endpoint and remove the first point
             *
             * For simplicity the first m points are set to the same value as are
             * the last m points
             * */
            for ( i= 0, S1=0.0 ; i < m ; i++)
            S1 += ABS(x[i]);
            /* get gain in the first section */
            for(i = 0 ; i <= m2  ; i++){
            g[i] =  S1 / m ;
            }
            /* now get everything in the middle */
            sum = S1;
            for(i = m2 +1  ; i < n - m2  ; i++ ){
            sum += ABS( x[i+m2]) - ABS( x[i-m2-1]);
            g[i] = sum / m;
            }
            /* now fill in the tail */
            for(i = n -m2 ; i < n ; i++)
            g[i] = sum / m;
            /* as of now the g[i] represents an envelope of the trace
             * We will now eliminate any zeros, and then invert it to
             * get the gain factor
             * */
            max = 0.0;
            for(i=0 ; i < n ; i++){
            if(g[i] > max) max = g[i];
            }
            /* safety */
            if(max == 0.0){
            lower_bound = 1.0 ;
            } else {
            lower_bound = 0.001 * max;
            }
            for(i = 0 ; i < n ; i++)
            g[i] = 1.0/ MAX(g[i], lower_bound);
            /* now adjust the amplitude */
            for(i = 0 ; i < n ; i++)
            x[i] *= g[i];
            /* clean up */
            free(g);
            return;
}
```

The operations performed here are as follow:

1. Do nothing is there are no traces in memory, e.g., *ntrc = 0*.
2. For each trace in memory, get the number of points, *npts*, and the *sample interval, delta*, from the trace header, noting that the first is an integer and the second is a float.
3. Process the trace with the *gsac_doagc* routine. Determine the new maximum and minimum of the trace.
4. Update the header values.

4 Example

The following example presents a data set from a walkaway experiment with a 24 channel seismograph system. The vertical sensors were distributed to a distance of 40 meters from the hammer source.

The data files are in a SAC format and all end with the characters *.z*. The following command script ensures that the traces are in the correct byte order, displays the traces, windows the traces and high pass filters, finally AGC's the traces, and convert the *CALPLOT* graphics file to Encapsulated PostScript.

```
#!/bin/sh

for i in *.z
do
saccvt -I < i > tmp; mvtmpi
done

gsac << EOF
bg plt
r *.z
prs
cut b 0 b 0.2
r *.z
hp c 100 np 3
prs
agc w 0.1
prs
quit
EOF

for i in 001 002 003
do
plotnps -F7 -W10 -EPS < Pi. PLT > AGC{i}.eps
done
```

The graphics are show in the following figures.

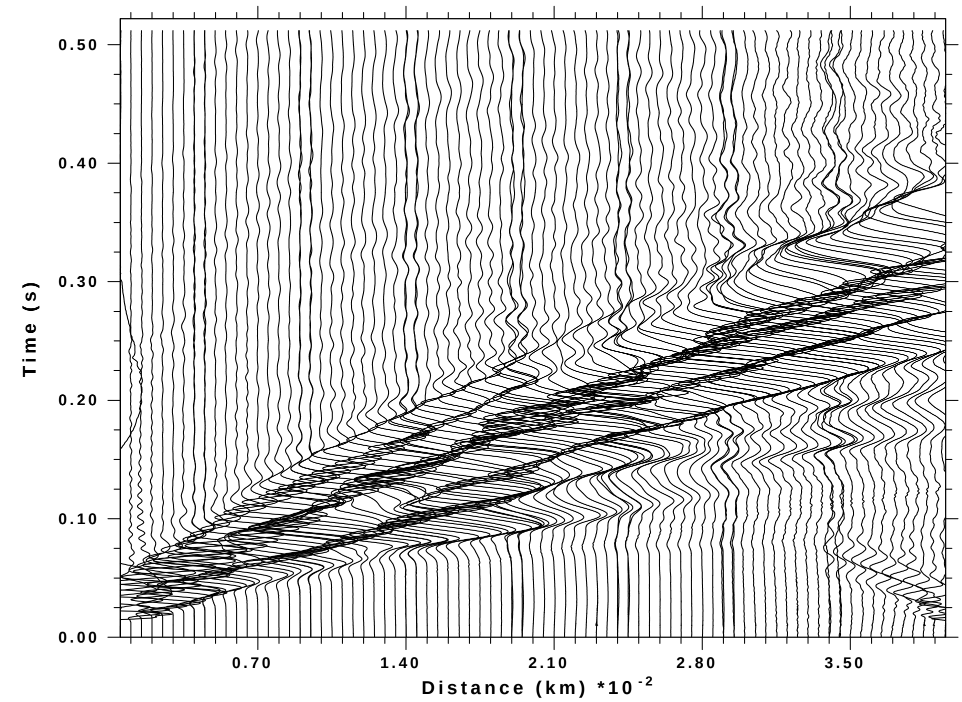


Fig. 1. Original traces recorded on a 40 meter spread. Note the very strong surface wave arrival.

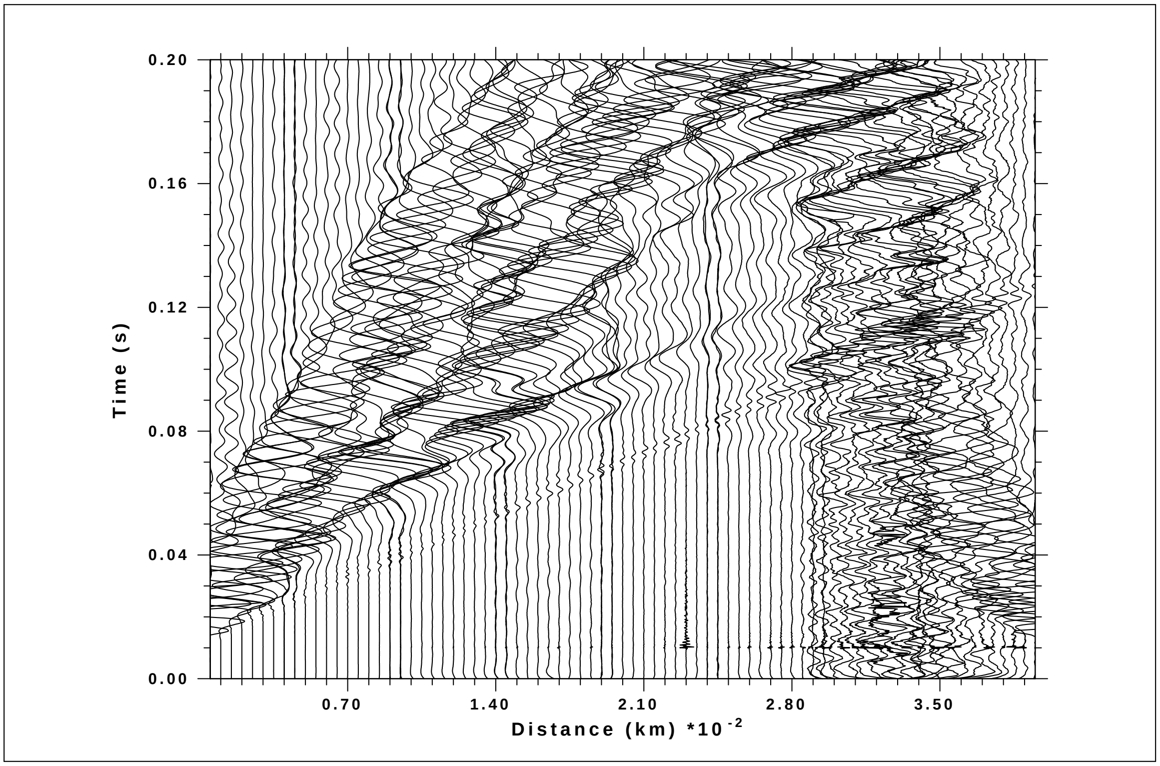


Fig. 2. Traces high pass filtered at 100 Hz.

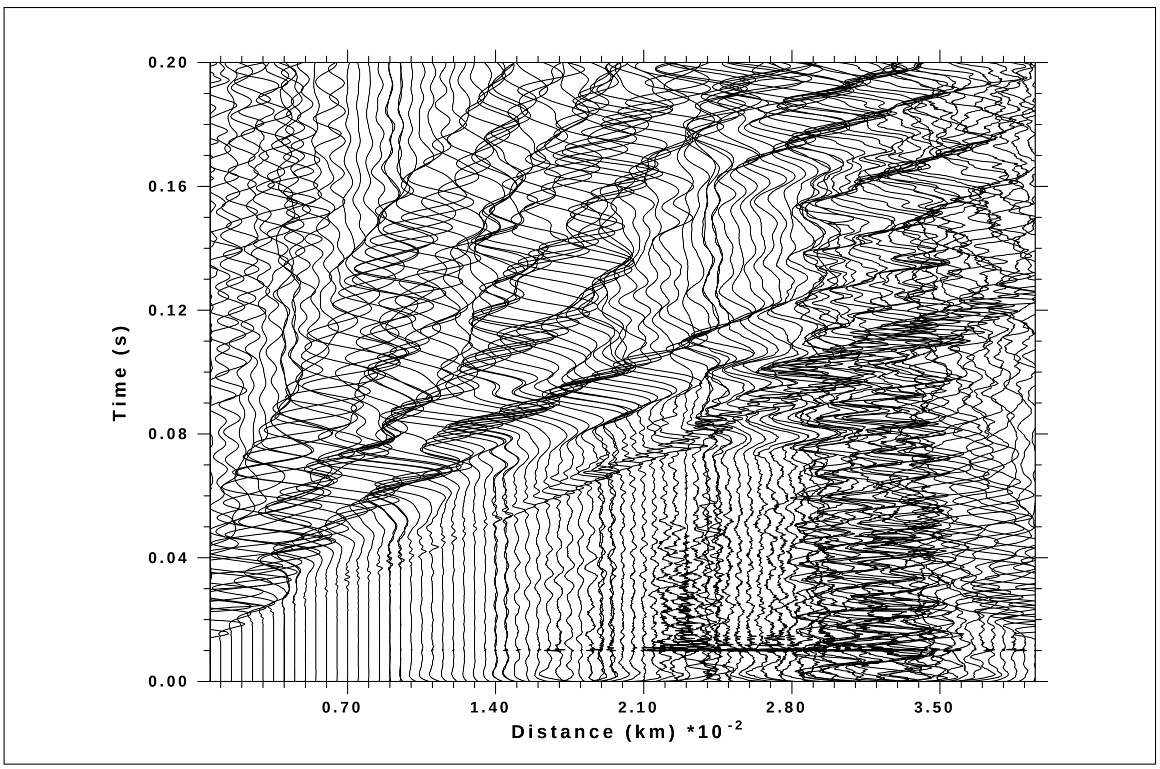


Fig. 3. Filtered traces AGC'd with a 0.10 second window. Note the prominent low frequency refraction at a time 0.08 seconds near a distance of 24 meters.

5. Discussion

This simple example demonstrates the procedure for adding new commands to **gsac**. Programming experience is required, but the use of the `MAKEPROTO` eliminates many of the errors and drudge or doing this by hand.

Look at the existing code   for examples.

APPENDIX F
IRIS TOOLS

1. Introduction

An excellent way to access digital data is to obtain a SEED volume from a data center, such as IRIS (*www.iris.edu*). The SEED volume contains the selectede digital time streams as well as the description of the data stream.

This distribution of Computer Programs in Seismology contains two programs available from IRIS, **`rdseed`** and **`evalresp`**. The first program reads the SEED volume to create SAC files and the response file. If the option is to create SAC waveforms, **`rdseed`** correctly sets the station coordinates and the component orientation in the SAC header fields.

To illustrate the procedure, we will request three component data for the station WCI for the June 18, 2002 southwestern Indiana earthquake from the IRIS data center. I used the IRIS BERE_FAST Query Form, currently at *http://www.iris.edu/SeismiQuery/breq_fast.htm*, to determine the data availability for a station. This lead to another page which permits me to view the BREQ_FAST file, which I then edited and forwarded to the IRIS DMC as an Email request for data. For my own convenience, I placed these in a shell script. Note one should always check the data collection. The shell script is used to permit me to request the same data set again in the future and to remind me of what I have previously requested.

The shell script is

```
#!/bin/sh

Mail -v  breq_fast@iris.washington.edu << EOF
WCI IU 2002 06 18 17 32 26.0 2002 06 18 17 57 25.0 1 BHE
WCI IU 2002 06 18 17 32 26.0 2002 06 18 17 57 25.0 1 BHN
WCI IU 2002 06 18 17 32 26.0 2002 06 18 17 57 25.0 1 BHZ
EOF
```

Everything within the *EOF* is the mail message sent to *breq_fast@iris.washington.edu*. Note that I will be notified by Email when my request is available in the FTP area. The seed file will be in the directory

ftp://ftp.iris.washington.edu/pub/userdata/RBHerrmann/.

To unpack this seed volume, I do the following:

```
rdseed -f xx -R -d -o 1 > out 2>&1
```

This command will create the SAC files (*-d -o 1*), and create the ASCII response files (*-R*) from the seed file. In addition information will be output to the standard error. This error output stream contains slightly more information than contained in the IRIS version of this program. For this data set we would see the following for this station:

```
IU WCI ** BHE 1997,198,00:00:00 2999,000,00:00:00.0000 38.229000 -86.294000 506.0   0.0  90.0  20 RESP.IU.WCI..BHE
IU WCI ** BHN 1997,198,00:00:00 2999,000,00:00:00.0000 38.229000 -86.294000 506.0   0.0   0.0  20 RESP.IU.WCI..BHN
IU WCI ** BHZ 1997,198,00:00:00 2999,000,00:00:00.0000 38.229000 -86.294000 506.0 -90.0   0.0  20 RESP.IU.WCI..BHZ
```

These entries are the Network, Station, Location, Component, On and Off dates, latitude, Longitude and Elevation of the Station, the component motion angle with respect to the downward vertical and azimuth from north, the sample rate and the name of the response file.

In addition the following files are created in the current directory:

```
RESP.IU.WCI..BHE
RESP.IU.WCI..BHN
RESP.IU.WCI..BHZ
2002.169.17.32.42.1481.IU.WCI..BHN.D.SAC
2002.169.17.33.35.2481.IU.WCI..BHZ.D.SAC
2002.169.17.35.16.5981.IU.WCI..BHE.D.SAC
```

The next step in data processing is to deconvolve the instrument response. Because this task is done for every earthquake studies, I put everything in a shell script:

```
#!/bin/sh

#####
#       script for processing IRIS Digital Data
#####
YEAR=2002
YR=02
MO=06
DY=18
HR=17
MN=37
SEC=16
MSEC=000
LAT=37.97
LON=-87.79
DEP=5.0
MAG=4.9
NAME=Evansville
DOY=169

####
#       No changes below here
#####

if [ -d ../GOOD ]
then
        echo GOOD exists
else
        mkdir ../GOOD
fi

for i in *.SAC
```

```
do
rm -f AMP* PHASE*
NET=‘saclhdr -KNETWK $i‘
LOC=‘saclhdr -KHOLE $i‘
if [  ${LOC} ]
then
        if [ ${LOC} = "-12345" ]
        then
                LOC=""
        fi
fi
for i in *.SAC
do
rm -f AMP* PHASE*
NET=‘saclhdr -KNETWK $i‘
LOC=‘saclhdr -KHOLE $i‘
if [  ${LOC} ]
then
        if [ ${LOC} = "-12345" ]
        then
                LOC=""
        fi
fi
KSTNM=‘saclhdr -KSTNM $i‘
KCMPNM=‘saclhdr -KCMPNM $i‘
DELTA=‘saclhdr -DELTA $i‘
echo RESP.${NET}.${KSTNM}.${LOC}.${KCMPNM} resp
cp RESP.${NET}.${KSTNM}.${LOC}.${KCMPNM} resp
evalresp ${KSTNM} ${KCMPNM} ${YEAR} ${DOY} 0.01 50.0 1000 -u ’vel’ -f resp
mv AMP* afile
mv PHASE* pfile
#####
#       set upper limits for deconvolution
#       according to sampling interval
#####
F4=‘echo $DELTA | awk ’{print 1.0/($1 * 4.0)}’‘
F3=‘echo $DELTA | awk ’{print 1.0/($1 * 8.0)}’‘
echo $KSTNM $KCMPNM $F3 $F4

gsac << EOF
r $i
ch EVLA $LAT EVLO $LON EVDP $DEP
ch O GMT $YEAR $DOY $HR $MN $SEC $MSEC
ch lovrok true
ch lcalda true
wh
rtr
w ../GOOD/${KSTNM}${KCMPNM}.S
transfer from eval subtype  afile pfile TO NONE FREQLIMITS 0.005 0.01 ${F3} ${F4}
w ../GOOD/${KSTNM}${KCMPNM}.sac
quit
EOF

done
EOF
```

This operation does not change the original SAC files in any way. Instead the event coordinates are placed in the header and the original trace file is copied to a parallel directory *../GOOD* and the ground velocities in *meters/sec* are also placed in that directory. The files ending with the *.S* is in counts and those ending if *.sac* are the ground velocity.

2. Documentation

The manual pages for **`rdseed`** and **`evalresp`** are available with this distribution in the directory

PROGRAMS.XXX/DOC/IRIS.pdf