TSPP---A Collection of FORTRAN Programs for Processing and Manipulating Time Series

By David M. Boore

U.S. Geological Survey Open-File Report 2008-1111

Revised 10 January 2020

U.S. Department of the Interior
U.S. Geological Survey

U.S. Department of the Interior
David Bernhardt, Secretary

U.S. Geological Survey
Jim Reilly, Director

U.S. Geological Survey, Reston, Virginia 2009
Revised: 10 January 2020

For product and ordering information:
World Wide Web: http://www.usgs.gov/pubprod
Telephone: 1-888-ASK-USGS

For more information on the USGS—the Federal source for science about the Earth,
its natural and living resources, natural hazards, and the environment:
World Wide Web: http://www.usgs.gov
Telephone: 1-888-ASK-USGS

Suggested citation:
Boore, D.M., 2020, TSPP---A collection of FORTRAN programs for processing and manipulating time series, U.S. Geological Survey Open-File Report 2008-1111 (Revised 10 January 2020).

Contents

Figures

Tables

TSPP---A Collection of FORTRAN Programs for Processing and Manipulating Time Series

By David M. Boore

Introduction

This report lists a number of FORTRAN programs that I have developed over the years for processing and manipulating strong-motion accelerograms. The collection is titled **TSPP**, which stands for **T**ime **S**eries **P**rocessing **P**rograms. I have excluded "strong-motion accelerograms" from the title, however, as the boundary between "strong" and "weak" motion has become blurred with the advent of broadband sensors and high-dynamic-range dataloggers, and many of the programs can be used with any evenly spaced time series, not just acceleration time series.

This version of the report is relatively brief, consisting primarily of an annotated list of the programs, with four examples of processing, and a few comments on usage. I do not include a parameter-by-parameter guide to the programs.

Although these programs have been used by the U.S. Geological Survey, no warranty, expressed or implied, is made by the USGS as to the accuracy or functioning of the programs and related program material, nor shall the fact of distribution constitute any such warranty, and no responsibility is assumed by the USGS in connection therewith.

The programs are distributed on an "as is" basis, with no warranty of support from me. These programs were written for my use and are being distributed in the hope that others might find them as useful as I have. I would, however, appreciate being informed about bugs, and I always welcome suggestions for improvements to the codes. Please note that I have made little effort to optimize the coding of the programs or to include a user-friendly interface. Many of the programs in this collection have been included in the software usdp (Utility Software for Data Processing), being developed by Akkar et al. (personal communication, 2008); usdp includes a graphical user interface. Unfortunately, as of this writing the program no longer seems to be available. Speed of execution has been sacrificed in favor of a code that is intended to be easy to understand, although on modern computers speed of execution is rarely a problem.

I will be pleased if users incorporate portions of my programs into their own applications; I only ask that reference be made to this report as the source of the programs.

Revisions

This manual has been revised and a new example has been added. A number of the programs have been revised slightly, but no or few new programs have been added. A few programs are no longer supported (and have been removed from the distribution files) because they

have been superseded by other programs (such as **smc2psa_rot_gmrot_interp_acc_rot_osc_ts**, which is a more general version of **smc2psa_rot_gmrot**). I draw attention to these useful programs: **filt_plot_lc.for** and **filt_plot_gen.for**. Both of these programs write a one page postscript file with plots of an acceleration time series and a series of filtered versions of that time series. In **filt_plot_lc.for** the low-cut filter corner frequencies are computed from a specified low and high frequency and a number of frequencies, such that the frequencies are log-spaced; **filt_plot_gen.for** is more general in that filters can be either low-cut, high-cut, or band-pass, with arbitrarily specified corner frequencies. Both of these programs are useful for determining the filter corners to be used in processing data. A sample session (Session 3) shows how **filt_plot_lc.for** can be helpful in choosing low-cut filter corner frequencies.

All examples have been recomputed and new figures were prepared, to guarantee consistency between the most recent versions of the programs and the figures in this report.

WARNING: On 02 February 2012 I added the computation of epicentral distance to **smcwrite.for** if both the station and earthquake coordinates are available. This required a call to the subroutine **distaz.for**. Although the set of subroutines in TSPP includes this subroutine, any main program that calls **smcwrite** must add an include statement that specifies the location of **distaz.for**. I have done so for the main programs that I have been using since 02 February 2012, but I have made no attempt to add the include statement to all main programs that call **smcwrite**. Here is the include statement that I use (substitute the proper path for your application):

include '\forprogs\distaz.for'

Some Abbreviations

I occasionally use abbreviations, particularly in the descriptions of the programs. For the convenience of the reader I define the abbreviations here:

Table 1. Abbreviations.

Abbreviation	Meaning
FAS	Fourier amplitude spectrum, usually of acceleration
FFT	Fast Fourier Transform
FS	Fourier amplitude spectrum
PGA	peak ground acceleration
PGD	peak ground displacement
PGV	peak ground velocity
PSA	pseudo-absolute response spectral acceleration ($\omega^2 SD$)
PSV	pseudo-relative velocity response spectrum (ωSD)
RS	response spectrum (type unspecified)
SA	absolute acceleration response spectrum

SD	relative displacement response spectrum
SV	relative velocity response spectrum
sps	samples per second
zoc	Zeroth-order corrected time series (in which the only baseline-correction consists of a mean being removed from the time series, the mean usually determined from the pre-event portion if available or the whole record if not available).

Program Acquisition and Use

Acquiring: The programs can be obtained from the online-software link on my web site, http://www.daveboore.com. The programs are contained in several compressed binary files ("zip" files). This report and the file containing the SMC format specifications (discussed below) are in TSPP_DOCUMENTATION.ddMmmyy.ZIP. The source code is in five files: TSPP_CONVERT_FOR.ddMmmyy.ZIP, TSPP_PROCESSING_FOR.ddMmmyy.ZIP, TSPP_UTILITIES_FOR.ddMmmyy.ZIP, TSPP_SUBROUTINES_FOR.ddMmmyy.ZIP, and TSPP_SUBROUTINE_COLLECTIONS.ddMmmyy.ZIP. The control files are in TSPP_CONTROL_FILES.ddMmmyy.ZIP. The executables, compiled to be run on a PC, are in TSPP_CONVERT_EXE.ddMmmyy.ZIP, TSPP_PROCESSING_EXE.ddMmmyy.ZIP, and TSPP_UTILITIES_EXE.ddMmmyy.ZIP. The data used in the sample sessions are in TSPP_FILES_FOR_SAMPLE_SESSIONS.ddMmmyy.ZIP. In the filenames, "ddMmmyy" is an abbreviated date for the revision (e.g., "10jan20" is a version created on 10 January 2020). I no longer use version numbers.

Compiling and linking: The programs use "include" statements to bring in the subroutines. These include statements assume that the subroutines are in a folder titled "\forprogs", located in the same letter drive as the programs. The programs are written in FORTRAN 77, with some Fortran 90 extensions. I compiled and linked the programs using Lahey/Fujitsu LF95.

Using: All programs are used from a command-prompt window. Most of the programs use a control file so that processing can be done in a batch mode, working with a list of files. It is convenient to make the file list using the command "dir/on/b [*specify file, using wildcard character * if necessary*] > file.list" in a Command Prompt window. Note that the "b" switch produces a brief listing, without file sizes, creation dates, and so on; the /on switch is optional and will order the files alphabetically. Using /o-d will order the files by date and time, with the most recent being first; this is often useful if there are many files in a folder and only the most recent are to be used. For example, to make a list of all files in the current folder (named "working_folder" in this example) with the extension "smc" and containing the string "_u" in the file name, use the following command in the Command Prompt window:

```
C:\working_folder> dir/on/b *_u*smc > file.list
```

The file *`file.list`* will contain a list of files that can be pasted into the appropriate control file.

This is not a detailed user's guide to the programs. For details of program capabilities and use, start by reading the brief annotations in the following tables, then read the comments at the

beginning of each program, and also read the comments in the corresponding control file. It may also be useful to read the dated list of program modifications included at the bottom of the header comments at the beginning of each program file. I have included several examples of processing, from which the user can obtain some information that will help in using the programs. The zip file TSPP_FILES_FOR_SAMPLE_SESSIONS.ddMmmyy.ZIP contains almost all of the input and output files.

Some comments and examples regarding processing can be found in a few of my papers, including those in the references at the end of this report.

Data Format

The processing programs read and write files that are in the standard SMC format used for the accelerogram data distributed by the Strong-Motion Program of the U.S. Geological Survey (USGS) (http://escweb.wr.usgs.gov/nsmp-data/smcfmt.html, last accessed 11 January 2020) except that I have utilized one of the undefined integer header values (47) to allow an option for the higher precision now available with modern dataloggers (see the file SMCFMT_DMB(with version number).PDF in TSPP_DOCUMENTATION.ddMmmyy.ZIP for a description of the modified format). The SMC-format files are in ASCII, with text, integer, and real headers, followed by a block of comments, and then the data. The headers and comments allow many of the metadata for a time series, including details about the processing, to be carried along with the time series values. The data are stored in line-oriented format (a number of consecutive data values per line, wrapped to following lines as needed).

Although other formats for strong-motion data are in use, such as COSMOS V1 (https://www.strongmotion.org/archive/publications/reports/format_1_20.pdf, last accessed 11 January 2020), I have not added conversions to the more recent formats. Note, however, that TSPP includes a collection of programs to convert data files in various formats into SMC format. These reformatting programs are grouped into three sets: those dealing with generic data formats, those with agency-specific formats, and one program for creating simple time series (such as a box, a spike, or a portion of a sinusoid) in SMC format.

Programs

This collection contains many programs that are designed to operate on SMC-format files as objects (not to be confused with object-oriented programming, but as opposed to a user having to explicitly open each file and read the contents to extract the data.). Not all of these programs are as thoroughly vetted as others. Some of the programs were written for a special use and have been used rarely, while others make up the workhorses of the collection---these include, but are not limited to, **asc2smc**, **blpadflt**, **smc2fs2**, **smc2rs**, **smc2vd**, **smc2asc**, and **smctsplt**. In the tables below I have highlighted in yellow those programs that I use often (lack of highlighting does NOT mean that the other programs are not useful, only that I use them less often). A warning: I don't always update all programs that perform similar functions, so when an alternative exists to perform a certain operation, use the more recent program (when in doubt, consult the dated list of program changes in the headers to the program file). I include all of the programs in the collection more as a convenience for me than for the user. I hope that the yellow highlighting will help the user sort through the collection.

I have organized the programs into five sets:

1. Programs that convert data in various ASCII formats into SMC-formatted files; also included in this set is a program for generating files with simple waveforms for testing purposes.

2. Programs that process the SMC files to create new SMC files with the modified time series, or to compute other measures of ground shaking. Some of the programs work with a single time series, and others work with a pair of time series.

3. Post-processing and utility programs to accomplish a variety of tasks, such as changing header values in the SMC files (e.g., **smcnuhdr**), or combining a mix of time series, response spectra, and Fourier spectral SMC files into a single file with columns containing time, amplitude, period, response spectra, and/or frequency, and Fourier amplitude spectra (**smc2asc**). The output file written by **smc2asc** can be imported easily into a variety of programs for further analysis or to make plots. Some of the programs in this group could logically be placed into the first group. Examples are the program **smc_rot_1pair_per_azm.for** that rotates two time series to generate a new time series, or the program **smc_snip** that snips out a segment of specified duration and starting time from a longer time series.

4. Subroutines and collections of subroutines. The subroutines used by the previous programs are collected into this group. As before, some are used often, some rarely. The subroutines are merged into the main program's source code at compile time using Fortran "include" statements. For some programs I used a batch file to create a collection of subroutines to be included, replacing a long list of include statements with a single include statement. These collections are listed at the end of the table of subroutines.

Because it is understood that the output of the programs will be a file or files, I rarely mention this in the tables below. The programs produce output in either SMC format or in column-oriented format (some programs have the option of saving in either format). Column-oriented format arranges consecutive values of the dependent variable in columns, unlike the line-oriented format of SMC files. The collection of post-processing and utility programs contains one very useful program---**smc2asc**---that converts a series of SMC files into a single ASCII file with columns of time (or frequency or period, depending on type of data) and dependent values, a pair of columns for each SMC file. These column-oriented ASCII files can then be easily imported into standard graphics programs for plotting or subsequent manipulation.

Time Series Preparation (Formatting and Conversion)

The programs for reformatting data written with generic formats are:

Table 2. Programs for generic formats.

Program Name	Program Description
asc2smc.for	Create an SMC file from an ASCII file in which the data are in two columns: time, data
onecol2smc	Create an SMC file from an ASCII file in which the data are in one column, with the samples per second specified in the control file
wrapped2smc.for	Create an SMC file from an ASCII file in which the data are in a

	block with a known format (e.g., (10f8.2)).

The programs for reformatting data written with agency-specific formats are in the following table. I have specified the agency for some of these programs.

Table 3. Programs for agency-specific formats.

Program Name	Agency Name
bdsn2smc.for	Berkeley Digital Seismic Network
cgs2smc.for	California Geological Survey
cr2gs.for	USGS-Central Region
cwb2smc.for	Central Weather Bureau (Taiwan)
esmd2smc.for	European Strong-Motion Database
estb_acc2smc.for	Engineering Seismology Toolbox
estb_tra2smc.for	Engineering Seismology Toolbox
evt2smc.for	Kinemetrics EVT format files
f96_2smc.for	
impc2smc.for	Imperial College (England)
iran2smc.for	
knet2smc.for	K-Net (Japan)
nga2smc.for	
pea2smc.for	Pacific Engineering Associates
sac2smc.for	SAC format
sce2smc2.for	S. Cal. Edison
sxv2smc.for	Spudich and Xu Compsyn
taiwan_one_component2smc.for	
uca2smc.for	UCA (San Salvador)
upsar2smc.for	
usc2asc.for	University of S. California
uscv1gs.for	University of S. California
uscv2gs.for	University of S. California
uscv3gs.for	University of S. California
uw2gs.for	University of Washington

The program for making "test" time series consisting of simple waveforms is listed below:

Table 4. Program to make "test" time series.

Program Name	Program Description
smc_make.for	Make a spike, pulse, step, ramp, n cycles of sine, or noise in SMC format

Processing Time Series

Table 5. Processing programs.

Program Name	Program Description
General Purpose	
blpadflt.for	The main processing program. It can do baseline corrections, filtering, integration to velocity and displacement, and computation of response spectra.
filt_plot_lc.for	Applies a suite of filters and plots the unfiltered acceleration, zeroth-order-corrected (zoc) velocity, and filtered displacements for a suite of filter corners, on a single page. In combination with smc2fs2, this program is particularly useful in helping decide what filter corner frequencies to choose in processing the data. [NOTE: This version replaces filt_plot.for; see below for another version
filt_plot_gen.for	This is a more general version of filt_plot in that the filtering is done for a list of filter corners rather than the filter corners being derived from flow, fhigh, number of filters, with the ratio of the filter frequencies being a constant (log spacing).
smc_periods_from_zero_crossings_and_extrema.for	Computes measures of periods for time series in smc format. This is useful in comparisons with random-vibration theory results and in computing earthquake magnitudes.
Filter	
smc_hicut_fd_cosine_taper.for	Simulates an anti-aliasing filter by applying a raised cosine taper between two frequencies in the frequency domain ("FD"). Used in the Boore and Goulet (2014) paper.
smc2cav	Compute the Cumulative Absolute Velocity (CAV)
smc2instr_response	Computes the time series response of a specified instrument to an input acceleration. As of now two instruments are supported: Wood-Anderson and WWSSN-SP.
Integrate and Differentiate	
smc2vd.for	Integrate an acceleration time series to velocity and displacement.
smc_d2va.for	Calculate acceleration and velocity from displacement.

smc_v2a.for	Calculate acceleration from velocity.
smc_v2ad.for	Calculate acceleration and displacement from velocity file.
smc2husid.for	Calculate the cumulative integral of square acceleration from acceleration file (use to make Husid plots).
smc2jerk.for	Calculate jerk (the first derivative of acceleration).
Resample Time Series	
smc_interpolate_time_series_using_fork.for	Resample a time series to a sampling interval given by the original sample interval divided by a power of 2, by extending the FAS from a FFT with zeros from the original Nyquist frequency to the new (and higher) Nyquist frequency. This is equivalent to convolution in the time domain with a sinc function and is the operation required by the sampling theorem.
smc_interpolate_time_series_linear.for	Interpolates a smc-format time-series file to finer time spacing, using linear interpolation
Envelope of Time Series	
smc2env.for	Use the FFT to compute the envelope of a time series. Store the result in a column-oriented file.
Fourier Spectra	
smc2fs2.for	Compute the Fourier amplitude spectra for a specified time series. Smoothing options and different ways of specifying frequencies are included. Output can be in SMC or column-oriented format.
smc2fas.for	Compute the Fourier amplitude spectra for a specified time series. Less general than smc2fs2.
smc2fs2_complex.for	Compute the Fourier complex spectra for a specified time series. A special purpose program; not well tested.
smc2phs.for	Compute the Fourier amplitude, phase, and phase derivative (see Boore, 2003b).
Response Spectra	
smc2rs.for	Compute SD, PSV, pseudo-absolute PSA, SV, and SA response spectra. The periods can be specified in various ways. Options include resampling using straightline interpolation when there are fewer than 10 samples per oscillator period and sinc interpolation, as discussed by Boore and Goulet (2014).

smc2rs_ts.for	A revision of smc2rdts that allows for computation of rd or rv oscillator response, specification of the portion of the acceleration record to use in computing the oscillator response, and the specification of the oscillator initial conditions [see Boore, D. M., A. Azari Sisi, and S. Akkar (2012). Using pad-stripped acausally filtered strong-motion data, *Bull. Seismol. Soc. Am.* **102**, 751-760, for an example of using nonzero initial conditions].
smc2rs_sel_per.for	Compute RS for selected periods, writes the result to a single-column file, one record per line, along with the latitude and & longitude of the station, and the distance from a specified location (such as an epicenter).
smc2psa_sd_pgv_pga.for	Compute PSA, SD, PGV and PGA for a list of files.
Operations Using Pairs of Time Series	
smc_acc2psagm_vs_r.for	Similar to smc_psagm_vs_r, but allows filtering of the input time series before computing the PSA.
smc_psagm_vs_r.for	Compute geometric mean of PSA from two components; write into a file along with distance from a multi-segment fault. No rotations are performed.
gmeanrot.for	Compute the geometric mean of the two horizontal components for a series of rotation angles.
smc2psa_rot_gmrot_interp_acc_rot_osc_ts.for	Combine two horizontal components for a range of rotation angles in order to compute several measures of seismic spectral intensity. This program is an updated and expanded version of smc2psagmrot and smc2psa_rot_gmrot, which are no longer supported. See Boore et al. (2007) and Boore (2010) for definitions of the intensity measures. This program has two output options: 1) separate files for each pair of horizontal-component ground motions, with each line being the ground-motion intensity measures (GMIMs) for a single period, and 2) a single file with separate rows for each pair of horizontal-component ground motions, with the seismic intensities for each period given by entries in individual columns; this format of output is convenient for subsequent analysis of multiple recordings. The post-processing program split_smc2psa_rot_gmrot_output.for splits the single-file output of smc2psa_rot_gmrot_interp_acc_rot_osc_ts into

	two files, one with the RotD50 GMIM and the other with the RotD100 GMIM. This is a revision of smc2psa_rot_gmrot that is many times faster than the previous program. Also included is the option to resample the input time series, using the theoretical correct interpolation operator. See notes_on_revisions_to_smc2psa_rot_gmrot_v1.0.pdf on the dave-s notes page of www.daveboore.com for details. smc2psa_rot_gmrot is no longer provided in the TSPP package.
smc_psa_vs_rotation_angle.for	Output is PSA for a series of rotation angles, with pairs of horizontal component time series as input. The PSAs are computed for a set of periods. The program is useful for checking the results of smc2psa_rot_gmrot.
smc2gm_ar.for	Compute the geometric mean of two as-recorded (no rotations) horizontal components, as well as the response spectra of each individual component, and the larger of the two components.
smc2gmn.for	Compute the geometric mean of the two horizontal components for a series of rotation angles. Output minimum, maximum, and fractile values of the geometric mean for specified periods. Used in preparing Boore et al. (2006)
smccoher.for	Compute the coherence between two SMC files.
smccorrl.for	Cross correlate two SMC files using a specified range of lag times.
smcxcorr.for	Cross correlate up to 10 pairs of files and write results as columns in a single file.
smcinty1y2.for	Produce a file of the cumulative integral square of two time series
Miscellaneous (Mostly Older) Programs	
blftsegs.for	Integrate to velocity, fit a series of line segments to the velocity, subtract the slopes of the line segments from the acceleration and write a new, baseline-corrected file.
fasratio.for	Compute the ratio of Fourier amplitude spectra for pairs of time series in SMC format.
smc2subd.for	Read an SMC filename from a control file, compute PSV, and write it to a file in the same format as Gail Atkinson's SUBALL.DAT.
samaxmin.for	Loop over a series of rotation angles, for each forming the time series for the rotation angle and computing PSA. The maximum and minimum

	over the set of rotation angles is output (an older program).
smc_periods_from_zero_crossings_and_extrema.for	Computes measures of periods for time series in smc format. This program is useful in comparing random-vibration calculations with time-domain calculations and in computing some types of earthquake magnitude.
smc2spectral_moments.for	Computes the spectral moments used in random-vibration theory.
smc_apply_site_response	Computes the time domain response of an input motion after applying site amplifications from the program Nrattle (part of the SMSIM suite of programs, downloadable from the online software page of www.daveboore.com).

Post-Processing and Other Utility Programs

Table 6. Programs for post-processing and other tasks.

Program Name	Program Description
Plot SMC time series	
smctsplt.for	Plot up to 31 time series on a single page. A very useful program.
Convert from SMC to ascii	
smc2asc.for	Combine up to 72 SMC files into a formatted, single-column file. The SMC files can be a mix of time series, Fourier spectra, and response spectra. Time-shifting, amplitude scaling, and decimation can be included.
smc2col	Convert a list of smc files into column files, one file per smc file (unlike smc2asc, which creates one column file from a list of smc files).
Edit SMC headers	
smcnuhdr.for	Replace integer and real headers in SMC files with specified values.

smcaddeq.for	Read earthquake name from control file and write into the appropriate SMC text header (4, columns 27:80).
smcaddsn.for	Read station number from appropriate SMC text header (3, col 1:4), verify that it is non-blank and an integer in the range 0-9999, and if so, then write into int_header(30).
smcfxhdr.for	Add header information to SMC files (primarily text headers).
smc_add.for	Read station name from control file and write it in text header(6)(11:40), obtain orientation from text header(6)(53:57) and write the appropriate number in the integer headers 13, 14.
smc_long.for	Change sign of station longitude and earthquake longitude to negative (if positive) (this program was written to overcome a provincialism in some older accelerogram data from areas of west longitude, for which positive values were stored for the longitude).
add_vs30_2smc	Add vs30 to smc file real_head(40)
Obtain information about SMC file	
smc_info.for	Read a list of SMC or RS2 files, and for each, extract station information and print it to two files: 1) a summary file; and 2) a file in comma-delimited format for import into data base programs.
smc_hdrs.for	Read a list of SMC file names, and for each entry, open the data file and read the header information. The extracted information is then written to a file. This eliminates the need to manually list the contents of each file to check components, station coordinates, and so on.
smc_sta.for	Read a list of SMC files, and for each, extract station information and print it to two files: 1) a summary file; and 2) a file in comma-delimited format for import into data base programs.
smc_y1ym.for	Get initial value (y(1)) and peak value of time series
Make new SMC files from existing SMC files	Does not include filtering, integration, and differentiation operations---see the section on processing programs for these operations
smc_snip.for	Snip out a section of an SMC file and write the results as a new SMC file.
smc_rot_1pair_per_azm.for	Rotate pairs of files, where each pair, the azimuth, and the output file names are specified separately.
smc_detrend	Removes a straight line fit to the first and last points of a time series and writes the results in a new smc file. This program is

	intended as a quick way of doing a baseline correction for those very infrequent cases where the time series has an approximately linear baseline drift over the complete length of the record.
smc_pad.for	Add leading zeros to SMC file and write as a new file.
smc_std.for	Snip out a section of an SMC file and compute the mean and standard deviation.
smcadd2.for	Read two SMC files and adds one to another, time step by time step, to create a third file.
smcoffst.for	Read in a SMC file, apply a specified offset, and write as a new SMC file. The intent is to mimic a digitizer that does not have a perfect 0.0 baseline.
smcrmvsn.for	Remove a single cycle of a sine from acceleration. This is to study Norm Abrahamson's "fling-step" (personal communication, 2003).
smcrvrse.for	One-time-use program to time-reverse an SMC file (to confirm that acausal filtering is blind to the time direction).
smcxtend.for	Extends a SMC file by either adding zeros to front and back or by reflecting the time series around the beginning and ending points (a special purpose program, written while preparing Boore, 2005b).
smc_dig.for	Simulate analog-to-digital conversion, and write as a new SMC file. Used for Boore (2003a).
Post-process results of main processing programs	
band_avg.for	Compute averages of spectra between specified bands.
env2avg.for	Read a series of files containing envelopes (made by env2phs) and compute the average of the envelopes.
env2hist.for	Read a series of files containing envelopes (made by env2phs) and compute an average histogram.
fas_avg.for	Read a column-oriented file made by smc2fs2 and compute and output the average and standard average of the mean, using linear averages.
fs2_avg.for	Read up to 72 Fourier amplitude spectra in SMC format and reformat to produce a column-oriented file, including the average of the spectra.
lfhf_avg.for	Read a column-oriented file made by blpadflt and compute and output the average and standard average of the mean, using both linear and log averages.
colmerge.for	Merge entries in a series of single column files, such as those produced by BlPadFlt, into one single column file.
col_rat.for	Make a file with ratios of columns from specified column-formatted files.

colspect2gm_ar.for	Reads a pair of filenames from a control file, for which each file is a column file containing a column of frequency and another of spectra (they could be Fourier or response spectra, such as produced by smc2fs2 or smc2rs). Compute the as-recorded geometric mean, the sqrt sum of squares of the spectra (note: the meaning of this is questionable for response spectra), and writes to a file.
decimate_file_rows.for	Remove rows in a file. This is very useful when plotting Fourier spectra, as the output of, for example, smc2fs2 may have a frequency spacing that is smaller than necessary for the plot, and thus the resulting graph may be very large and may take a long time to load in a Word or pdf file.
smc2fasrat.for	Compute the ratio of Fourier amplitude spectra for specified time series. The program is unfinished; see the code for a work-around.
psv_avg.for	Read a column-oriented file made by blpadflt and compute and print out the average and standard average of the mean, using both linear and log averages.
smc2rs_col_output_transposed.for	Transpose output of smc2rs so that the spectra for each period are given in columns rather than rows; each row contains the response spectra for a single record. It is assumed that the periods are the same for each record.
split_smc2psa_rot_gmrot_output.for	Splits the single-file output of smc2psa_rot_gmrot_interp_acc_rot_osc_ts into two files, one with the RotD50 GMIM and the other with the RotD100 GMIM.
transpose_smc2rs_output.for	Transpose output of smc2rs so that the spectra for each period are given in columns rather than rows. This is a more specialized version of smc2rs_col_output_transposed, in which each record produces two records in the output file, one with the periods, followed by the response spectral values. It was written to allow importing into the spreadsheet in the e-supplement to Atkinson and Boore (2006).
Miscellaneous	
normal_probability_plot_prep.for	Sorts and ranks a setoff input values and computes the expected normal probability for the ranked set; write the output into a file for use in making a quantile-quantile plot (usually used to see if the input values have a normal distribution).
smc_sort.for	Reads a list of SMC files made with the command "dir/on/b *.smc > smc.lst" and rewrites the list, sorting so that the uncorrected, acceleration, velocity, and displacement files are in order. Use in building a control file for smctsplt.
smcwinno.for	Extracts a subset of files from a list based on a specified character string in the file name. This program can be used in combination with a dos dir/on/b command to generate a list of file names to be used as input into programs for which only the subset is needed.

smooth_x_y_data.for	Does what the name implies
tabsxpnd.for	Read a list of files from a control file and replace tab characters with a specified number of spaces.
unix2pc.for	Reformats a unix ASCII file to a pc ASCII file
a09_a32.for	Reformats a unix ASCII file to a pc ASCII file by replacing ASCII character 09 with ASCII character 32 (blank).
add_cr.for	Adds an extra carriage return to each line, resulting in double spaced text when printed.
mac2pc.for	Reformats a Mac ASCII file to a pc ASCII file.

Subroutines and Collections of Subroutines

The subroutines are listed alphabetically in the table below rather than by function or frequency of use. As noted previously, some of the programs are more widely used than others. For example, **smcread** and **smcwrite** are the main subroutines for reading and writing SMC files, and **fork** and **rdrvaa** are the main subroutines for computing Fourier Amplitude and response spectra, respectively. The subroutines **get_lun** (to return the largest unused logical unit number), **trim_c** (to remove leading and trailing blanks from a character string), **upstr** (to convert a string to uppercase), **skip** (to skip a specified number of lines when reading a file), and **skipcmnt** (skips over comment lines, which start with "!", when reading a control file) are used in almost every main program (note that comments in SMC-formatted files are preceded by "|", not "!"). **smooth_interpolate** is a useful subroutine that allows various smoothing operators. **filter** calls the time-domain filtering programs.

The subprograms taken from Numerical Recipes (Press et al., 1992) have been collected into a separate table; although these subroutines work well, they are have been classified as obsolete by the authors of the subroutines and are no longer supported by Numerical Recipes (see http://numerical.recipes/ for more information, including the downloadable book from which these subroutines were taken--Numerical Recipes in Fortran 77, Second Edition (1992)).

Table 7. Subroutines and Collections of Subroutines (except for those from Numerical Recipes).

Subroutine Name	Subroutine Description
Subroutines	
abs_mnmaxidx.for	Return minimum and maximum values of an array, along with the indices of the array corresponding to these values.
abs_spectra.for, absspect.for	Apply a tapered window to the front and back of a time series, pad with zeros to next power of 2, and compute the Fourier spectral amplitude. These are the same subroutines; I prefer to use the name "abs_spectra" for stylistic reasons, but some main programs still include "absspect.for" via an include statement.
abs_spectra_dp.for	A double precision version of abs_spectra.
absspect_complex.for	Apply a tapered window to the front and back of a time series, pad with zeros to next power of 2, and compute the complex Fourier spectrum.
acc2seismo_response.for	Computes the time domain response of an instrument to an input acceleration (see smc2instr_response for a driver). At this time two instruments are supported: Wood-Anderson and WWSSN-SP.
acc2vd.for	Integrate acceleration to obtain velocity and displacement.
accsqint.for	Integrate the square of the acceleration (in order to compute Arias intensity). The integration uses a formula derived under the assumption the acceleration is represented by straight lines connecting the digitized values.
accsqint_trapezoidal.for	Integrate the square of the acceleration (in order to compute Arias intensity). The integration uses the standard trapezoidal integration formula, with no assumptions about the form of the time series between sampled points.
ampphphd.for	Compute the amplitude, phase/2π, and if desired, the derivative of phase with respect to angular frequency.
band.for	Bandpass time-domain Butterworth filter (can be causal or acausal).
bjf94v30amp.for	Compute site amplifications using Boore et al. (1994) results.

bjf97.for	Return ground-motion values from the Boore et al. (1997) ground-motion prediction equations (GMPEs).
blpadflt_util_subs.for	This collection of subroutines available individually in this table is used by blpadflt.for.
byte_swap.f90	Used in sac2smc (written by C. Stephens).
cann.for	A character string subroutine from C. Mueller.
cav_compute.for	Compute the cumulative absolute velocity (CAV)
cmplxspc.for	Apply a tapered window to the front and back of the time series, pad with zeros to next power of 2, and compute the complex Fourier spectrum.
conditn.for	Apply a tapered window to the front and back of the time series and pad with zeros to the next power of two if needed.
construct_filename_extension.for	Constructs a character string containing information about the blpadflt option and filtering parameters. This subroutine is used in filt_plot to construct the names of the time series that are plotted.
correl_dmb.for	Compute correlation between two time series (used in smccorrl.for).
csr123.for	Parse next field in a comma-separated character string (from C. Mueller).
csrc.for	Extract a character string from a larger comma-separated string (from C. Mueller).
csrf.for	Extract a real number from a character comma-separated string (from C. Mueller).
csri.for	Extract an integer from a character comma-separated string (from C. Mueller).
d2va.for	Compute velocity and acceleration from displacement, assuming the reverse of the formulas in acc2vd (which assumes straight-line segments between acceleration values).
datetime.for	Obtain date and time character strings using a system call.
dcdt.for	Fit mean or trend between indices indx1 and indx2, then remove mean or trend from whole trace.
deg2km_f.for	Convert lat, long into km north and east from a reference point.
digitize.for	Simulate analog-to-digital conversion, used in Boore (2003a).
dis2va_wang.for	Compute velocity and acceleration from displacement using finite difference operators. [adapted from G.-Q. Wang]
dist_3df.for	Compute various distance measures from a point on the Earth's surface to a rectangle with arbitrary orientation and location in space (used

	to obtain distance from a station to a finite fault).
dist3dmf.for	Call dist_3df for each of a number of rectangles and return the minimum distances.
distaz.for	Compute great circle distances between two points on the surface of a sphere.
downsample.for	Downsample a time series
doy.for	Convert 4-digit year, calendar month and calendar day to day of year.
downsample_wang.for	Downsample a time series (a modification of the routine downsample by G.-Q. Wang).
envelope.for	Compute the Hilbert tranform of y and use it to compute the envelope and instantaneous frequency.
evt2smc_subs.for	Various subroutines used by the evt2smc converter program
evt2smc_headers.txt	Statements brought into evt2smc via include statements.
fbctpr.for	Apply cosine tapers to the front and back ends of time series.
filter.for	Interface to filter subroutines band, locut, hicut.
fork.for	Compute complex-to-complex Fourier spectrum using FFT algorithm.
forkdp.for	Double precision version of fork.for
fs2read.for	Read Fourier spectrum from a file in FS2 format (a form of the SMC format for Fourier spectra).
fs2write.for	Write Fourier amplitude spectra into SMC format
get_abs_fas.for	Returns Fourier amplitude spectrum, smoothed and interpolated to specified frequencies, if desired.
get_abs_fas_dp.for	Double precision version of get_abs_fas.for.
get_avg.for	Compute the mean of array entries between two indices.
get_date.for	Calls the Fortran 90 internal routine DATE_AND_TIME and formats the date into a string “mm/dd/yyyy”.
get_extrema.for	Returns an array of extrema for a time series (adjacent points equal in amplitude or a change in the sign of the slope on either side of a point).
get_index_for_character.for	
get_lun.for	Get highest available logical unit number.
get_npw2.for	Find nearest power of two.
get_path_from_file_name.for	Extract path from a file name
get_path_from_system_call.for	Get path using a system call

get_path_from_system_call.4linux_os.for	Get path using a system call, for Linux operating system
get_time.for	Calls the Fortran 90 internal routine DATE_AND_TIME and formats the time into a string “hh:mm:ss.sss”.
getampph.for	Use a FFT to obtain the amplitude and phase spectrum.
getf_out.for	Construct output file name.
hicut.for	High-cut (low-pass) time-domain Butterworth filter (can be causal or acausal).
hilbert.for	Compute the Hilbert tranform of y and use it to compute the envelope and instantaneous frequency.
histfreq.for	Place data values into bins, to use in plotting a histogram.
icmpmx.for	Used in one of the response spectral programs
imnmax.for	Find the minimum and maximum of an integer array.
indxlast.for	Find the last occurrence in string c_string of the single character c_char.
integrate_y.for	Compute cumulative integral of y assuming that y is represented by straight lines connecting the digitized values.
interpolate.for	Used in interpolating tabulated data.
interpolate_1d.for	Used in interpolating tabulated data.
interpolate_2d.for	Used in interpolating tabulated data.
interpolate_time_series_linear.for	Interpolates an smc file to finer spacing, using ! linear interpolation.
interpolate_time_series_using_fork.for	The subroutine to resample a time series to a sampling interval given by the original sample interval divided by a power of 2, used by smc_interpolate_time_series_using_fork and smc2psa_rot_gmrot_interp_acc_rot_osc_ts
intrp_ts.for	Linearly interpolate unevenly t_in, y_in to evenly sampled time series with sps samples per second.
invnorm.for	the z value corresponding to a specified area of the cumulative normal distribution between negative infinity and z.
len_trim.for	Return length of TEXT without trailing blanks.
lin_interp.for	Compute linearly interpolated value of y.
locut.for	Low-cut (high-pass) time-domain Butterworth filter (can be causal or acausal).
mak_stem.for	Construct a stem name (stem) by appending up to 4 characters of tag with a 4-character string giving the period T.

mean_std.for	Computes mean, standard deviation, standard error of the mean, and 70% and 95% confidence intervals.
mean_std_cl.for	Computes mean, standard deviation, and confidence limits of the array entries.
mnmax.for	Find the minimum and maximum of a real array.
mnmaxidx.for	Find the minimum and maximum of a real array, along with the array index of the minimum and maximum.
mnmxixdp.for	mnmaxidx for a double precision array.
newmnmax.for	Same functionailty as mnmaxidx.
notch.for	Notch time-domain Butterworth filter (can be causal or acausal).
notnumrc.for	Check character string to see if it has any non-numeric characters.
pea_read.for	Used in program to convert pea format to smc format time series.
peak_trough_max_amp_period.for	Computes the maximum peak and trough amplitude of a time series, as well as the approximate period associated with the maximum absolute amplitude.
period_from_zero_crossings.for	Measure period from zero crossings for a portion of a time series.
pole_zero_response.for	Returns the amplitude and phase of an instrument response, given the poles and zeros of the instrument.
poles_zeros_values.for	Returns poles and zeros for common instrument types.
putspace.for	Used in sac2smc (written by K. Assatourians).
rc_subs.for	Contains include statements for RCC, RCF, and RCI
rcc.for	Extract a character string from a larger character string (from C. Mueller).
rcf.for	Extract a real number from a character string (from C. Mueller).
rci.for	Extract an integer from a character string (from C. Mueller).
rdrvaa_rd_rv_ts.for	The same as rdrvaa, but it also returns the time series of the relative displacement and velocity oscillator response.
rdrvaa.for	Calculate relative displacement, relative velocity, and absolute acceleration response spectra.
read_csv.for	Read Comma Separated Variable (csv) file.
readhdrs.for	Read headers of SMC and RS2 files.
rmv_crlf.for	Remove carriage return/line feed characters and replace with a blank.
rmv_mean.for	Determine mean from portion of time series "a" between t4mean_strt and t4mean_stop, and

	remove this mean from the whole time series.
rmvtrend.for	Remove a straightline fit to first and last points, replacing the input array with the detrended array.
rotate.for	Rotates z1, z2 into azmr (stored in z1 on return) and azmr+90.0 (stored in z2 on return). No assumptions are made about the relation between the azimuths of the two components (other than that they are orthogonal). This subroutine should be replaced rotate_separate_in_out_variables.for, but it still might be called by some programs so it is included here.
rotate_separate_in_out_variables.for	Rotate z1, z2 into the azimuth azmr. This subroutine replaces rotate.for, which overwrote the input time series and azimuths with the rotated time series and azimuths. This sometimes led to problems when I forgot that the arrays had changed after the call. The new subroutine uses separate variables for the input and output time series and azimuths.
rs2rdhdr.for	Read headers of response spectra stored in rs2 format.
rs2read.for	Read response spectra stored in rs2 format.
rs2write.for	Write response spectra in SMC rs2 format.
rs_calc.for	Use the Nigam and Jennings algorithm, resampling the input time-series to a finer time spacing, if specified in the control file. The resampling is straightline interpolation when there are fewer than 10 samples per oscillator period. The more appropriate sinc interpolation, discussed in Boore and Goulet, 2014, can be implemented by using a resampled time series created by smc_interpolate_time_series_using_fork as the input to rs_calc. The program smc2rs combines the two subroutines. rs_calc uses subroutines icmpmx and rdrvaa.
select_subsets_for_rotations.for	Selects subset of original time series for computation of intensity measures in smc2psa_rot_gmrot_interp_acc_rot_osc_ts.for
skip.for	Skip a specified number of lines whe reading a file.
skipcmnt.for	Skip comments while reading a file.
smc_npts.for	Obtain length of time series stored in an SMC file.
smc_nsps.for	Obtain npts, sps of time series stored in an SMC file.

smcpadf.for	Pad a time series with leading and trailing zeros.
smcpadf_detrend.for	Pad a time series with leading and trailing zeros, with an option for removing a straight line fit to the first and last points.
smcread.for	Read SMC files.
smcwrite.for	Write SMC files.
smooth_interpolate.for	A general subroutine for smoothing the elements in an array. It is used by get_abs_fas.for, which in turn is called by smc2fs2.for. The smoothing can be over linear and log abscissa values. The subroutine can do smoothing or interpolation or both. The abscissa values at which the smoothed values are returned can be a different set than that corresponding to the input array. This subroutine includes the functionality of the individual smoothing operators in the "smooth" subroutines below.
smooth_konno.for	Smooth over log-spaced abscissa values, using Konno and Ohmachi weighting function (see BSSA 88, 228-241). This subroutine is no longer maintained. See smooth_interpolate for the most recent version.
smooth_logbox.for	Non-weighted smoothing over log-spaced abscissa values. This subroutine is no longer maintained. See smooth_interpolate for the most recent version.
smooth_logtriangle.for	Triangular-weighted smoothing over log-spaced abscissa values. This subroutine is no longer maintained. See smooth_interpolate for the most recent version.
smooth_n.for	Smooth y over nsmooth points (an odd number!) using a triangular weighting function. This subroutine is no longer maintained. See smooth_interpolate for the most recent version.
smths.for	Smooth amplitude spectrum with a triangle window (an old program, use smooth_n instead). This subroutine is no longer maintained. See smooth_interpolate for the most recent version.
tab_xpnd.for	Replace tabs in a character string with a specified number of blanks.
tabl_a_6.txt	Table A-6 in Boore et al. (1997); used by bjf97.for.
tabs_rmv.for	Replace tabs in a character string with a blank.
time_diff.for	Returns the difference in time between two times obtained using get_time, which in turn calls the internal Fortran 90 routine DATE_AND_TIME. See the program listing

	for the proper use of DATE_AND_TIME.
timediff.for	Compute the time difference in seconds between times returned from two calls to the system clock. This subroutine used a character string for the time that was returned by an earlier version of get_time. For consistency with the new version of get_time (which uses a standard Fortran 90 intrinsic routine to obtain the system time), the subroutine time_diff.for should be used. Not all programs in this version of TSPP have been modified appropriately.
tprfrctn.for	Apply cosine tapers to beginning and end of an array.
trapezoidal_integration.for	Does what the name implies.
trim_c.for	Remove leading and trailing blanks fro a character string.
tsplot_sub.for	Includes the functionality of smctsplt.for, used in filt_plot.for to plot a unfiltered acceleration, zeroth-order-corrected velocity, and filtered displacements for a suite of filter corners on a single page.
unwrap.for	Attempt to unwrap the phase of a complex Fourier spectrum.
upstr.for	Convert a character string to upper case.
v2a.for	Compute acceleration from velocity, using the reverse process of going from acceleration to velocity.
v2ad.for	Compute acceleration and displacement from velocity.
wind_box.for	Apply weighting using a box window with leading and trailing cosine tapers.
yintrf.for	Linear interpolation (old, use lin_interp).
zeropad.for	Add zeros to end of time series; used when computing FFT (smcpadf is a more sophisticated version, used when filtering data).
zeropad2.for	Determine npw2 and add zeros to end of time series; used when computing FFT.
Subroutine Collections	
blpadflt_util_subs.for	
evt2smc_subs.for	
filt_plot_util_subs.for	
hicut_fd_cosine_taper_fft_subroutines.for	

psa_rot_gmrot_util_subs.for	
rc_subs.for	
smc_hicut_fd_cosine_taper_subroutines.for	
smc2psa_util_subs.for	
tsplot_sub.for	

Table 8. Numerical Recipes subroutines.

Subroutine Name	Subroutine Description
avevar.for	Numerical Recipes subroutine
dcovsrt.for	Numerical Recipes subroutine
dddpoly.for	Numerical Recipes subroutine
dfpoly.for	Numerical Recipes subroutine
dgaussj.for	Numerical Recipes subroutine
dlfit.for	Numerical Recipes subroutine
four1.for	Numerical Recipes subroutine
fpoly.for	Numerical Recipes subroutine
gasdev.for	Numerical Recipes subroutine
indexx.for	Numerical Recipes subroutine
interp.for	Numerical Recipes subroutine
julday.for	Numerical Recipes subroutine
locate.for	Numerical Recipes subroutine
moment.for	Numerical Recipes subroutine
moment_dmb_n_lt_2.for	Slightly modified version of moment.for
momntdmb.for	Subroutine moment, modified to compute the moment for array entries between specified indices.
ran1.for	Numerical Recipes subroutine
realft.for	Numerical Recipes subroutine
select.for	Numerical Recipes subroutine
sort.for	Numerical Recipes subroutine
twofft.for	Numerical Recipes subroutine

Sample Sessions

Many examples of processing are contained in the references below, and guidelines for usage are given in some of the control files. More examples illustrating the consequences of various processing parameters are included in some of my informal notes on a variety of topics—see http://www.daveboore.com/daves_notes.html, last accessed 11 January 2020). I include here four sample sessions, containing results not included in the references below. The first sample session illustrates many of the processing steps, including formatting data from a data agency into SMC format, zoc processing, low-cut filtering (with and without tapers where the pads adjoin the data), and computation of Fourier and response spectra. The second session illustrates the results of different smoothing options in the computation of Fourier amplitude spectra. The third sample session illustrates processing that can help in choosing the low-cut filter frequency. The fourth session is new to this (10Jan20) version of TSPP and gives an example of a simple baseline correction.

Unless inserted in the text, all control files are contained in TSPP_FILES_FOR_SAMPLE_SESSIONS.ddMmmyy.ZIP. I have preserved the subfolders used in developing the examples for the sake of simplicity, even though this results in some duplication of files. These subfolders contain all of the files I used and created in doing the examples. Because I repeatedly use programs such as **smctsplt** and **blpadflt** in these sessions, to eliminate confusion I have added the session and step numbers to the control file names. Most of the processing programs prompt the user for a control file name, with a default name chosen by pressing "Enter" at the prompt. Although the user can override the default name by entering the actual control file name, it might be easier to rename the step-specific control file to the default name for the step (e.g., *blpadflt.ctl.session_1_step4* would be renamed *blpadflt.ctl*).

Session 1: Basic Processing

The records are from the 21 November 2004, 13:37 UTC, **M** 5.3 earthquake recorded on the island of Guadalupe (Jousset and Douglas, 2007). Here are some processing steps used in analyzing these data.

I have set the comments in the control files shown below in italics, although the ASCII files for the control files in the distribution package are in plain text.

1. Convert from European Strong-Motion Database Format to SMC Format

I used program **esmd2smc** to do the conversion, using this control file:

```
Control file for ESMD2SMC  ! first line
Name of summary file:
 esmd2smc.sum
Spc?
 N
Xfactr
 100.0
Input file name ("stop" in any column to quit):
```

```
501053xa.raw
501053za.raw
 STOP
```

(I actually converted the format of many more time series in one run of **esmd2smc**; to simplify this example, I show only the two horizontal components of record 501053 being processed.)

2. Plot the Time Series

I then used **smctsplt** to plot the time series in order to preview how much pre-event motion was available, and to check for obvious problems with the data. Here is the plot:

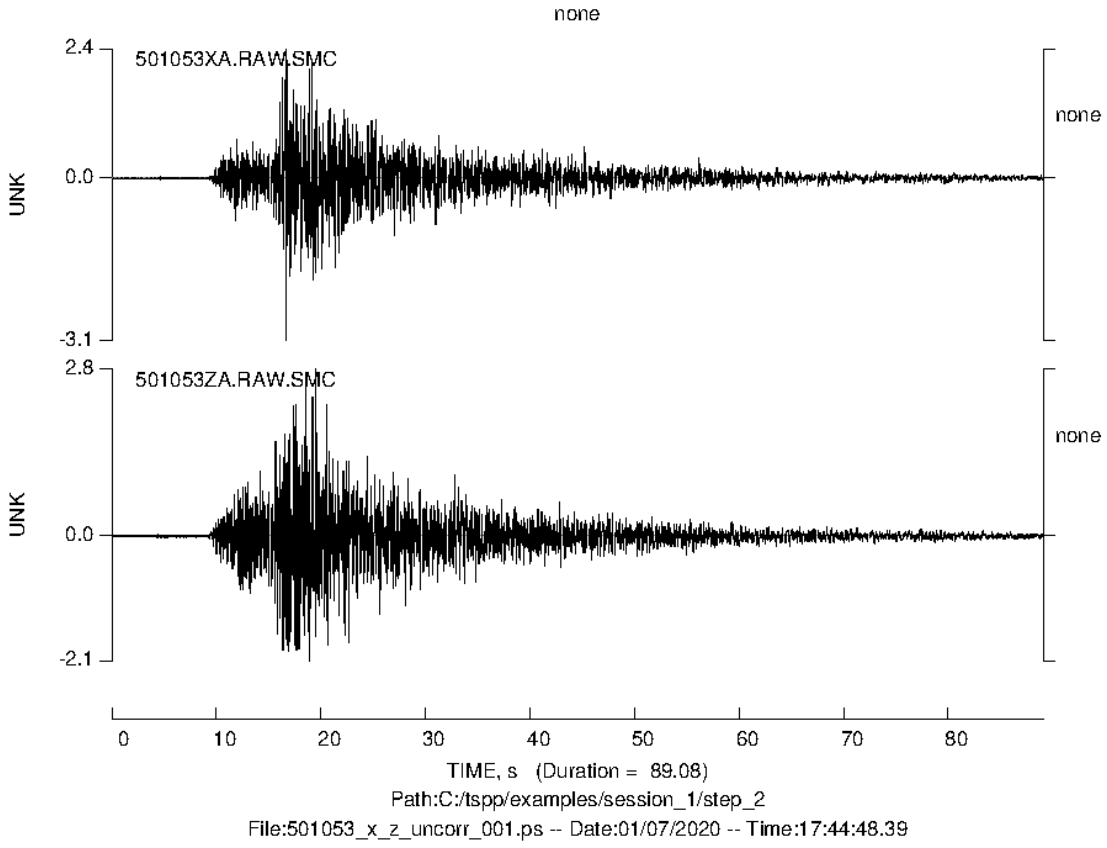

Figure 1. Plot of unprocessed acceleration time series. Note that the grainy appearance is a result of the time series values being replaced by an average value per pixel, thus greatly speeding up the plotting and reducing the file size.

Note that the plot above is an Portable Network Graphics (PNG) file, although the output of **smctsplt** is a PostScript (PS) file. I used **GSView** (http://www.ghostgum.com.au/software/gsview.htm/; last accessed on 07 January 2020) to make the conversion from PS to PNG (to be able to insert the plot in a Word document).

3. Do ZOC (Zero-Order-Corrected) Processing

Using the plot made in Step 2, I chose the time range of 0 to 8 s for determining the mean to be removed from the whole record. I used **blpadflt** to do what I call the zoc or zero-order-corrected processing, and I used **smctsplt** to make plots of the resulting acceleration, velocity and displacement time series.

Here is the plot made using **smctsplt**:

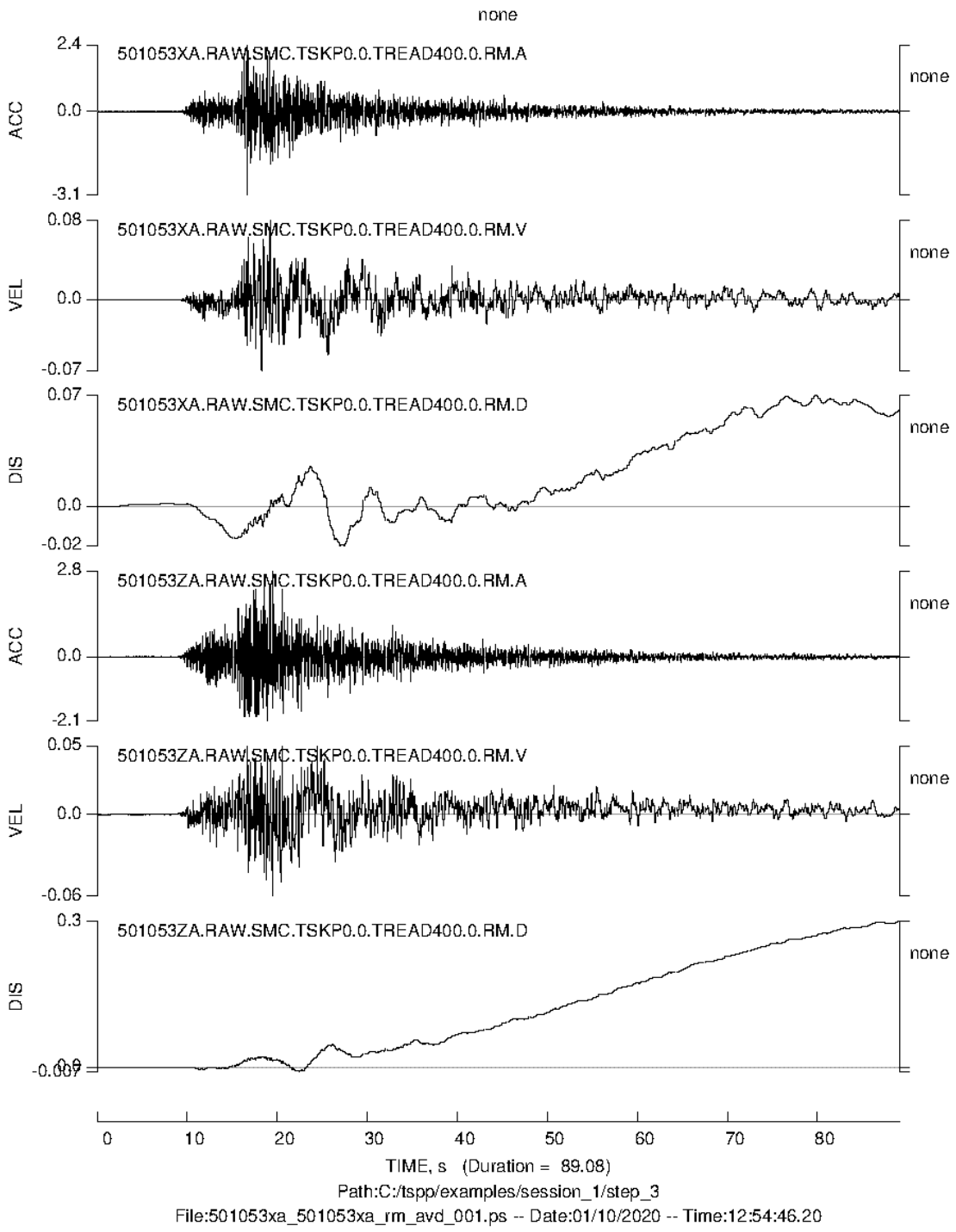

Figure 2. Plot of acceleration, velocity, and displacement time series from zoc processing. Note that the grainy appearance is a result of the time series values being replaced by an average value per pixel, thus greatly speeding up the plotting and reducing the file size.

4. Filter, Integrate, Compute Response Spectra

The two horizontal components (components x and z in the plots below) were each processed with acausal, low-cut filters for two different frequency corners, one at 0.02 Hz and the other at 0.04 Hz (selected in part by visual inspection of the plot of the zoc velocity). In all cases the filter order is such that the filter decays as f^8 at low frequencies, and tapers of 0.0 and 20.0 s were applied at the start and end, respectively, of each record before adding zeros and filtering. All of the processing was done using **blpadflt**. The control file for the 0.02 Hz and 0.04 Hz filters with 00-s and 20-s taper processing is given in *blpadflt.ctl.session_1_step4*. This control file produced the following files for the x-component, thee 20-s taper, and the 0.02 Hz filter (other sets of files were also produced, for the x-component filtered at 0.04 Hz, for the 00-s taper, and the same sets for the z component):

Table 9. Files produced by **blpadflt**, using the control file above.

File Name	File Description
501053xa.raw.smc.tskp0.0.tread400.0.alc00.020ns08.taper__0.0_20.0.a	Pad-stripped acceleration for component x, 20 s taper, and 0.02 Hz filter.
501053xa.raw.smc.tskp0.0.tread400.0.alc00.020ns08.taper__0.0_20.0.a.pad	Acceleration for component x, 20 s taper, and 0.02 Hz filter.
501053xa.raw.smc.tskp0.0.tread400.0.alc00.020ns08.taper__0.0_20.0.v	Pad-stripped velocity for component x, 20 s taper, and 0.02 Hz filter.
501053xa.raw.smc.tskp0.0.tread400.0.alc00.020ns08.taper__0.0_20.0.v.pad	Velocity for component x, 20 s taper, and 0.02 Hz filter.
501053xa.raw.smc.tskp0.0.tread400.0.alc00.020ns08.taper__0.0_20.0.d	Pad-stripped displacement for component x, 20 s taper, and 0.02 Hz filter.
501053xa.raw.smc.tskp0.0.tread400.0.alc00.020ns08.taper__0.0_20.0.d.pad	Displacement for component x, 20 s taper, and 0.02 Hz filter.
501053xa.raw.smc.tskp0.0.tread400.0.alc00.020ns08.taper__0.0_20.0.r.col	Response spectrum (not in SMC format), with columns containing period, relative displacement, pseudo-relative velocity, pseudo-

	absolute acceleration, relative velocity, and absolute acceleration response spectra.

5. Plot and analyze results

After filtering and integration using **blpadflt**, I used **smctsplt** to make plots of the displacements, which are shown below for the x-component:

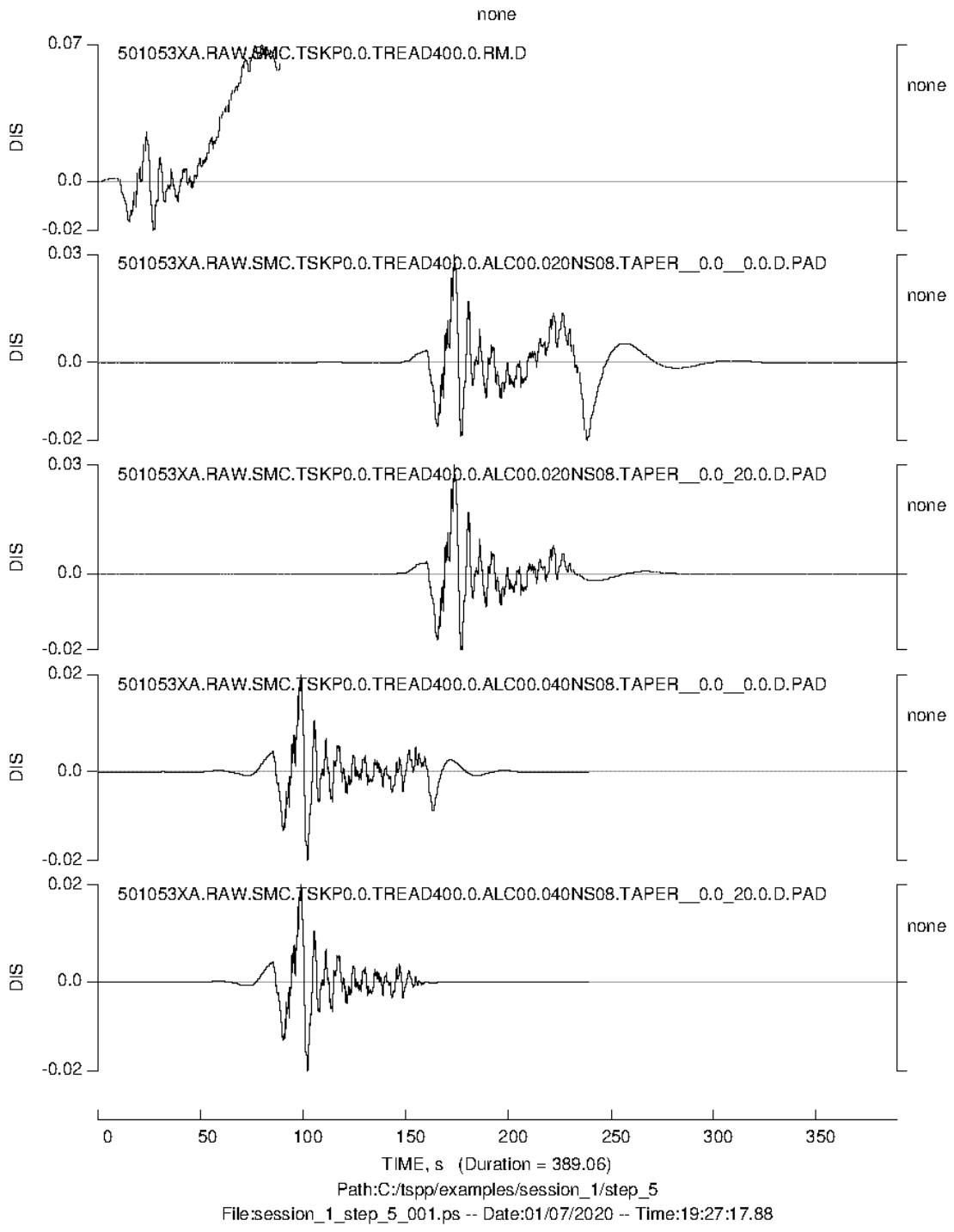

Figure 3. Plot of displacement time series from zoc (top) and filtered processing. The 2nd and 3rd traces are for a 0.02 Hz acausal low-cut filter, with no taper and a taper of 20 s where the training zeros are added to the data; the bottom two traces show the same thing for a 0.04 Hz filter.

Also shown above are the results of using the 0.0 s taper (the z component is omitted for clarity). Note that the waveforms are shifted in time relative to one another because they are aligned by the

first sample and not by the time of the first sample, which in each case varies according to the duration of the pre-signal pad that is applied.

Plots from **smctsplt**, like that above, are rough and intended to provide a quick overview of many time series on a single page. In order to make more precise plots of these particular time series I used **smc2asc** to merge the SMC files into a single column-oriented file, decimated the displacement time series by a factor of 5 (to save disk space and to generate smaller plot files; the original time series were sampled at 125 samples-per-sec (sps), and decimating by 5 leads to a sample rate of 25 sps, which is more than enough to display variations in the displacement waveforms) and time-shifted the displacements to allow the waveforms to line up; for each trace the shift was determined by using a text editor to look at the comments of the SMC files resulting from the **blpadflt** operation. The **smc2asc** control file is *smc2asc.ctl.session_1_fig_4*. I then opened the output file produced by **smc2asc** in the graphics program CoPlot (http://www.cohort.com/coplot.html; last accessed 11 January 2020) and made the following plot:

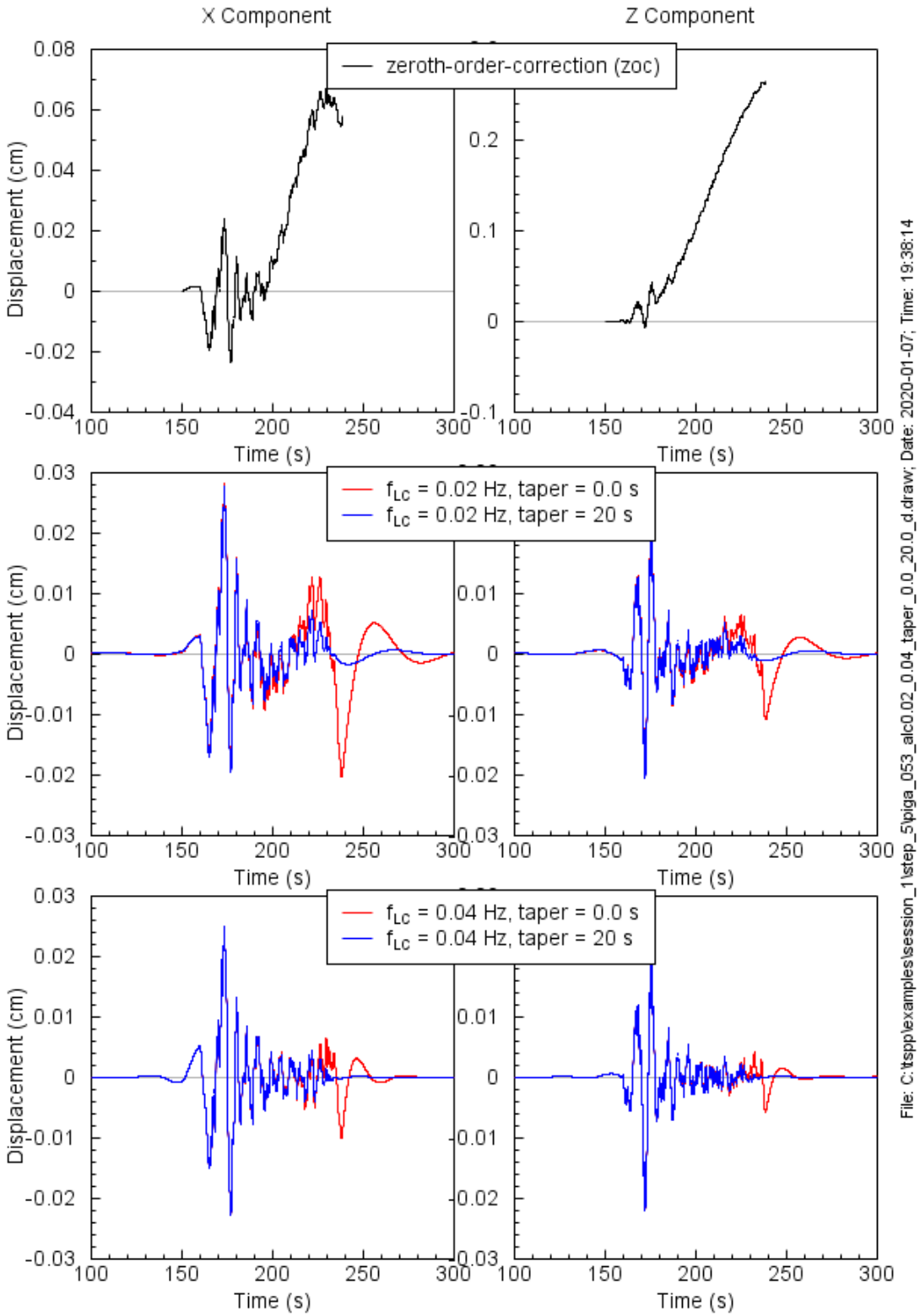

Figure 4. Plot of displacement time series from zoc (top) and filtered processing, with and without tapers where the training zeros are added to the data.

The left and right columns show the results for the x- and the z-components, respectively. Note that the transient associated with adding trailing pads is worse for the x-component than for the z-component and is reduced for the higher-frequency filter. Using a taper reduces the size of the transient significantly.

Generating plots of displacement waveforms for various filters can help in the choice of the appropriate filter corner frequency, as shown in the Session 3 examples. This selection process can also be aided by plotting the Fourier amplitude spectra (FAS) of the acceleration traces. For the traces being considered here I computed the FAS with **smc2fs2**, using the control file *smc2fs2.ctl.session_1_fig5*. I then used **smc2asc** with the control file *smc2asc.ctl.session_1_fig5* to produce a column-oriented file containing the Fourier spectra. (I could also have specified in **smc2fs2** that the output be in column-formatted files rather than smc-formatted files, and I could have opened these files in CoPlot, or I could have used **colmerge** to combine the column-formatted files into a single column-formatted file, which could then be opened in the graphics program). From the output of **smc2asc** the following plot was generated using CoPlot:

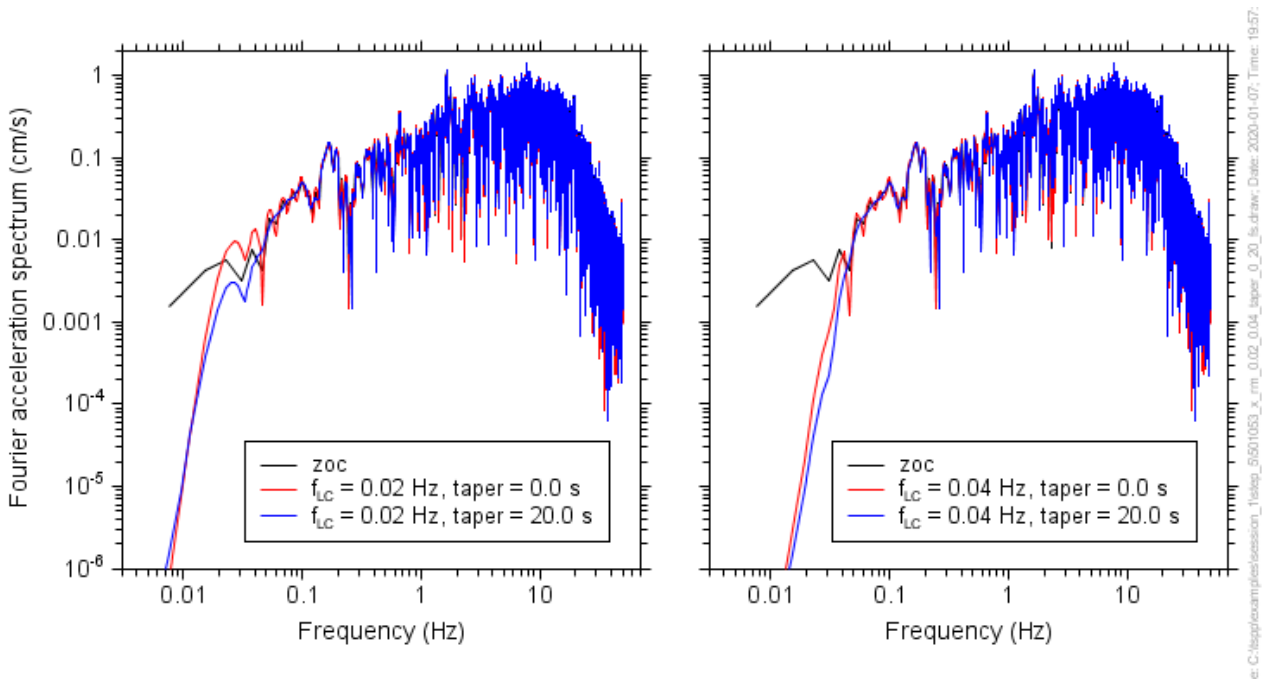

Figure 5. FAS of the acceleration time series corresponding to the processing used for the previous figures.

Although the choice of filter corner is subjective, I would choose a corner closer to 0.04 Hz than to 0.02 Hz, based on the decreasing slope with decreasing frequency starting around 0.03 Hz for the zoc FAS, and based on the general appearance of the displacement waveforms.

I also made plots of the relative displacement response spectra, using **colmerge** to combine the response spectral files made by **blpadflt**. The control file for **colmerge** used in this example is *colmerge.ctl.session_1_fig6*. Here is the plot of the response spectra (starting at T=1 s to show more details at long periods, where the differences due to the various processing will appear):

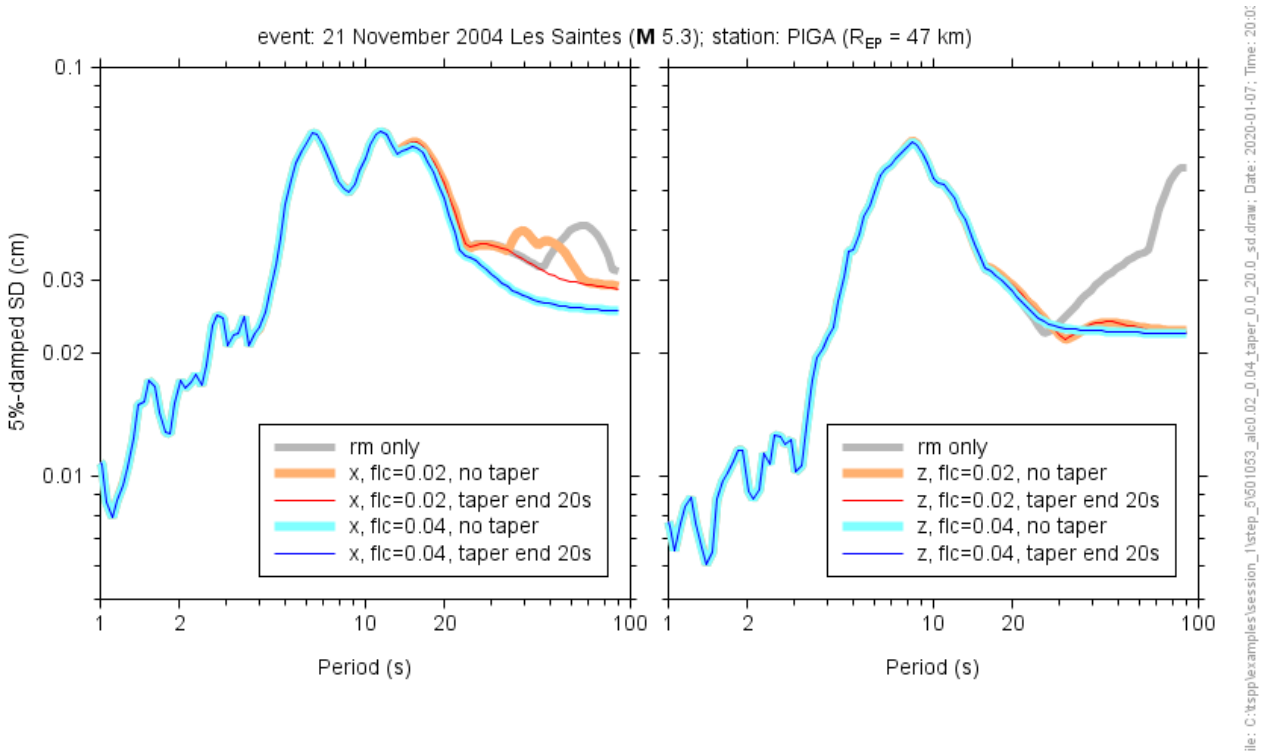

Figure 6. Response spectra (SD) of the acceleration time series corresponding to the processing used for the previous figures.

Note that for the 0.04 Hz filter corner the response spectra for both tapers are indistinguishable, but this is not true for the 0.02 Hz filter and the x-component of motion (the time series plot above shows that the transient is particularly large for this case).

Session 2: Smoothing of FAS

This is an abbreviated session intended to illustrate some of the smoothing options in **smc2fs2**. In seismological applications Fourier amplitude spectra (FAS) are commonly plotted using a log scale for the frequency axis (and also for the ordinate axis). It is often desirable to work with a smoothed version of the FAS. FAS from FFTs are defined at equally spaced frequencies, and therefore smoothing using a weighting function that is symmetric when plotted using linear frequency may give what appears to be too much smoothing at low frequencies and not enough smoothing at high frequencies. There may be good technical reasons for choosing a smoothing convolutional operator whose width is constant in log-spaced or in linear-spaced frequency (e.g., Konno and Ohmachi, 1998, make such an argument), but I suspect that usually it comes down to subjective judgment: what looks best in the eye of the researcher. The type of smoothing should depend on why smoothing is being used in the first place. Consider the site response of a single uniform layer over a halfspace: the peaks and troughs are equally spaced in frequency, not log frequency, and in this case smoothing using a function with constant width in linear frequency may be appropriate. On the other hand, if the purpose of the smoothing is to reduce "noise" in a FAS

that will be used to fit a spectral model, it might be more desirable to use a convolutional operator that has a constant width in log frequency.

The **smc2fs2** program allows five options for smoothing, two based on smoothing over a frequency-independent interval and three based on a smoothing operator whose width varies with frequency such that the width divided by the center frequency is constant (this corresponds to a constant width in log frequency). Box- and triangular-smoothing operators are available both for the frequency-independent and frequency-dependent smoothing operators. Another operator for the frequency-dependent case was proposed by Konno and Ohmachi (1998). Their operator uses the portion between the zero points on either side of the center frequency of a $\sin \chi / \chi$ function as the smoothing operator---see their equation (4). I've modified the smoothing to be between frequencies for which the weighting function exceeds 0.043 (see program comments). Here is the relevant portion of the **smc2fs2** control files:

```
! Meaning of smoothing input parameters
!
! NO SMOOTHING
! itype = 0
! SMOOTHING OVER EQUALLY SPACED FREQUENCIES
! itype = 1: box weighting function
! smooth_param = width of box weighting function (Hz)
! itype = 2: triangular weighting function
! smooth_param = width of triangular weighting function (Hz)
! SMOOTHING OVER LOGARITHMICALLY SPACED FREQUENCIES
! itype = 3: box weighting function
! smooth_param = xi, which is the fraction of a decade for the
!  box weighting function
! itype = 4: triangular weighting function
! smooth_param = xi, which is the fraction of a decade for the
!  triangular weighting function
! itype = 5: Konno and Ohmachi weighting function (see BSSA 88, 228-241)
! smooth_param = xi, which is the fraction of a decade for which
!  the Konno and Ohmachi weighting function is greater
!  than 0.043.(it is related to
!  their smoothing parameter b by the equation
!  b = 4.0/smooth_param, so we have this correspondence between
!  b and smooth_param
!   b smooth_param
!    10 0.40
!    20 0.20
!    40 0.10
!
!  b = 40 seems to be commonly used, but I do not think that it
!  gives enough smoothing; I PREFER SMOOTH_PARAM = 0.2,
!  corresponding to b = 20.
!
! ipow = power of FAS to be smoothed (2 = smoothing energy spectrum)
!
```

```
! df_smooth: Note: need df_smooth for linearly-spaced smoothers,
! and generally it should be the df from the fft.  For general x data, it is
! the spacing between x values, assumed to be constant,  The reason for
! including it as an input parameter is to "fool" the
! program to do smoothing over a specified number of points by
! setting df_smooth = 1 and smooth_param = number of points (including
! points with zero weight at ends; e.g., smooth_param = 5 will
! give a smoother with weights 0, 1/4, 2/4, 1/4, 0; smooth_param
! should be odd).
!
```

For smoothing using frequency-independent intervals, the smoothing parameter is the width of the smoothing operator. The program can be "fooled" to smooth over a specified number of points. The frequency spacing df is needed for the frequency-independent smoothing operators (usually given by the frequency spacing used in the FFT calculation), but by setting df = 1 and smooth_param = number of points (including points with zero weight at ends; there should be an odd number of points), the smoothing will be done over the desired number of points. For example, df = 1 and smooth_param = 5 will give a smoothing operator with weights of 0, 1/4, 2/4, 1/4, 0. For intervals that depend on frequency, the smoothing parameter is expressed as the fraction of a decade spanned by the operator (this is more understandable than the parameter used by Konno and Ohmachi).

I show in the figure below the results of smoothing using the frequency-independent triangular smoothing operator, using frequency intervals of 1, 2, and 4 Hz (the data are from the 08 January, 2006, **M** 6.7 Kythera, Greece, earthquake, recorded at station KRN1 at an epicentral distance of 146 km). The **smc2fs2** control file, deleting the comments portion shown above, for these three spectral computations is given in *smc2fs2.ctl.session_2_figs_7_8*, and the **smc2asc** file *smc2asc.ctl.session_2_figs_7_8* was used to combine the output files into a single column-oriented file, and CoPlot was used to make the following plot. This plot uses linear x-axis scaling to be consistent with the smoothing operator.

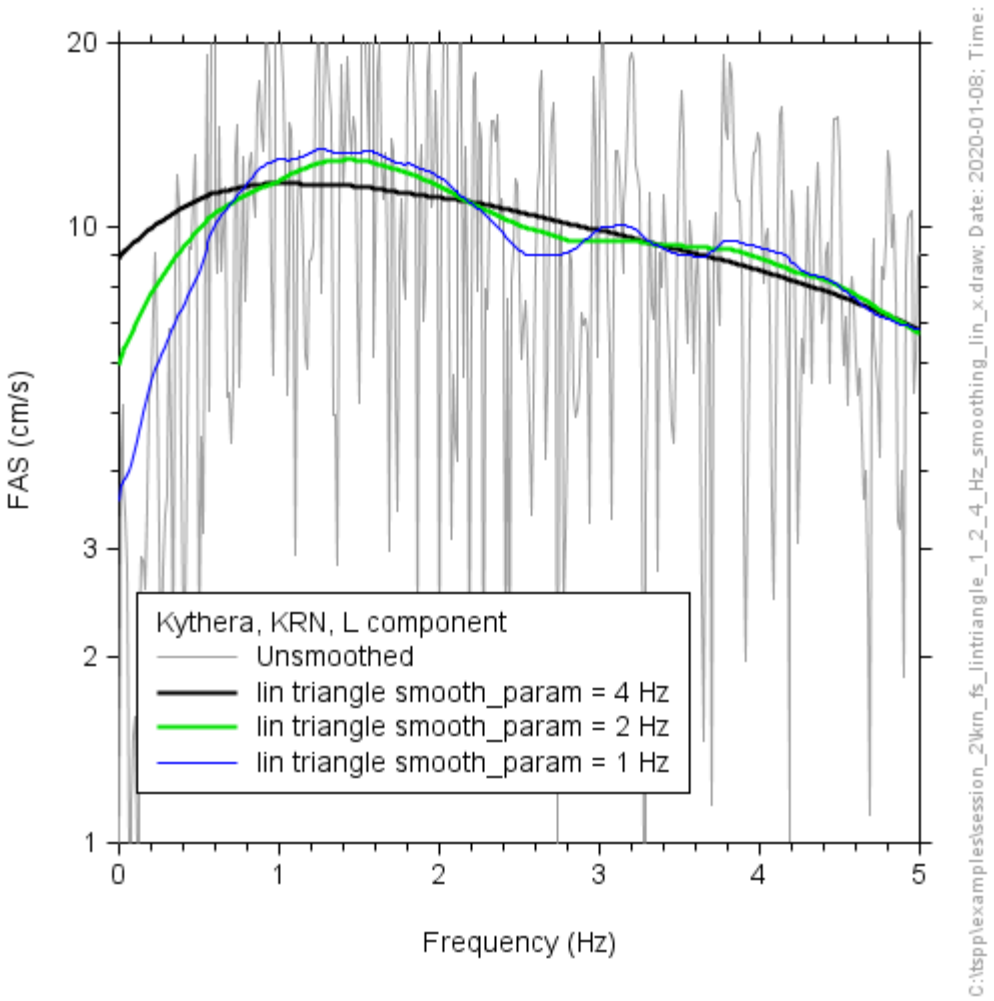

Figure 7. FAS without smoothing and with frequency-independent triangular smoothing over frequency intervals of 1, 2, and 4 Hz, plotted using a linear frequency scale, plotted with expanded x and y scales to show details.

When the figure is plotted using log-x axis scaling the excessive smoothing at low frequencies mentioned above is clearly seen, as shown in the following figure:

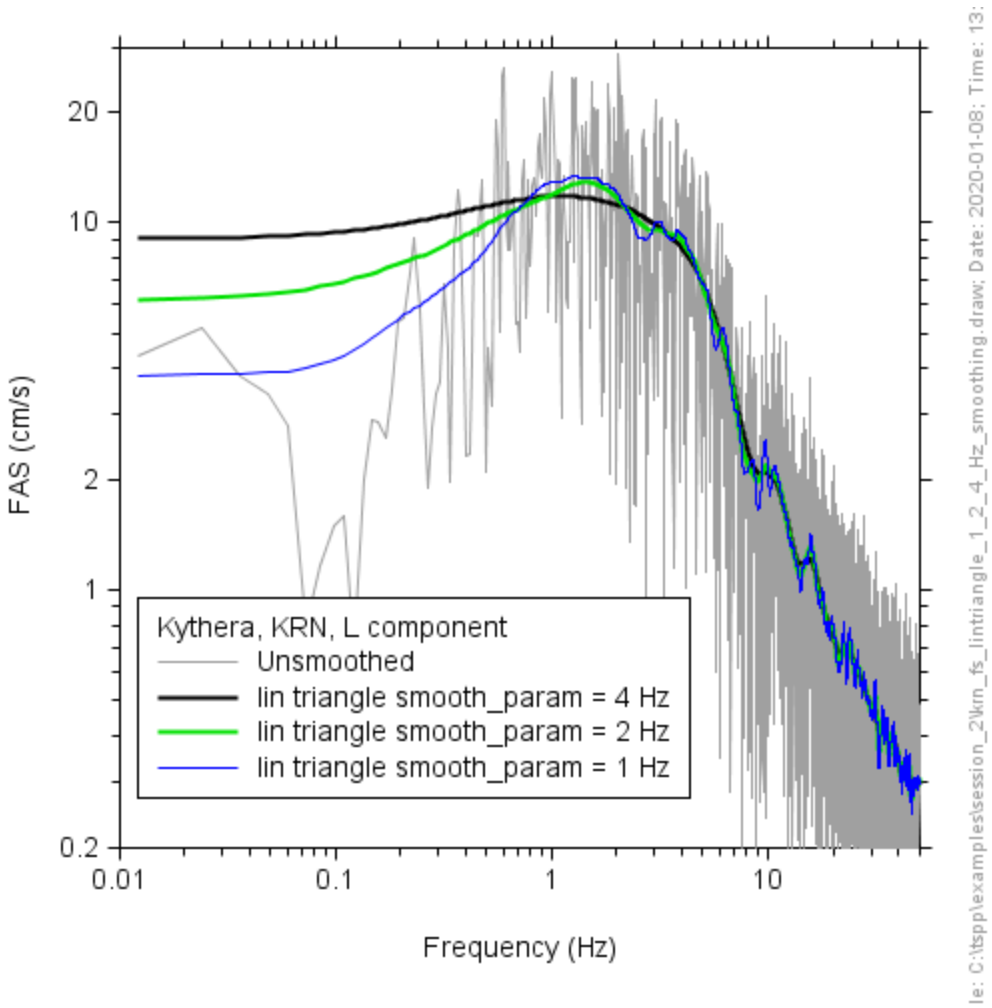

Figure 8. FAS without smoothing and with frequency-independent triangular smoothing over frequency intervals of 1, 2, and 4 Hz, plotted using log-scaling for the x-axis, to emphasize the low-frequency portion of the FAS.

This excessive smoothing can be overcome by using one of the logarithmically spaced operators. The plots below show the results of four such operators, using the following control file *smc2fs2.ctl.session_2_figs_9_10*.

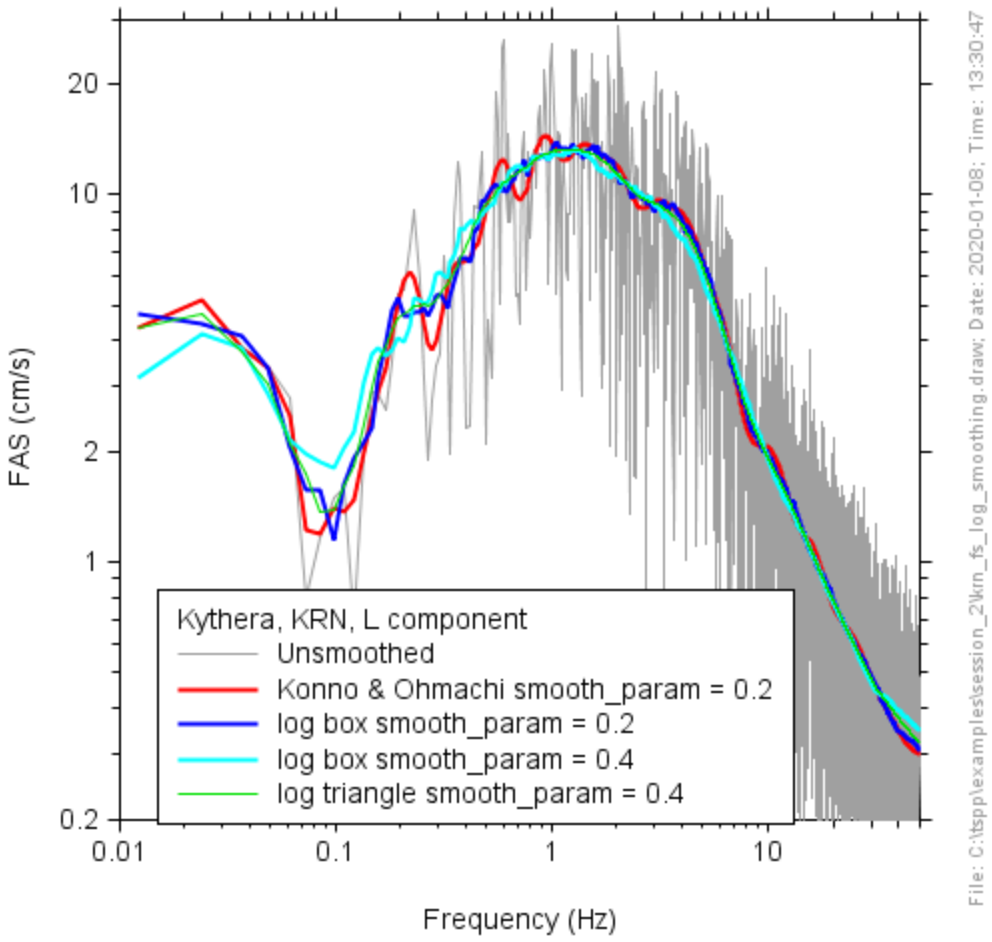

Figure 9. FAS without smoothing and with frequency-dependent smoothing.

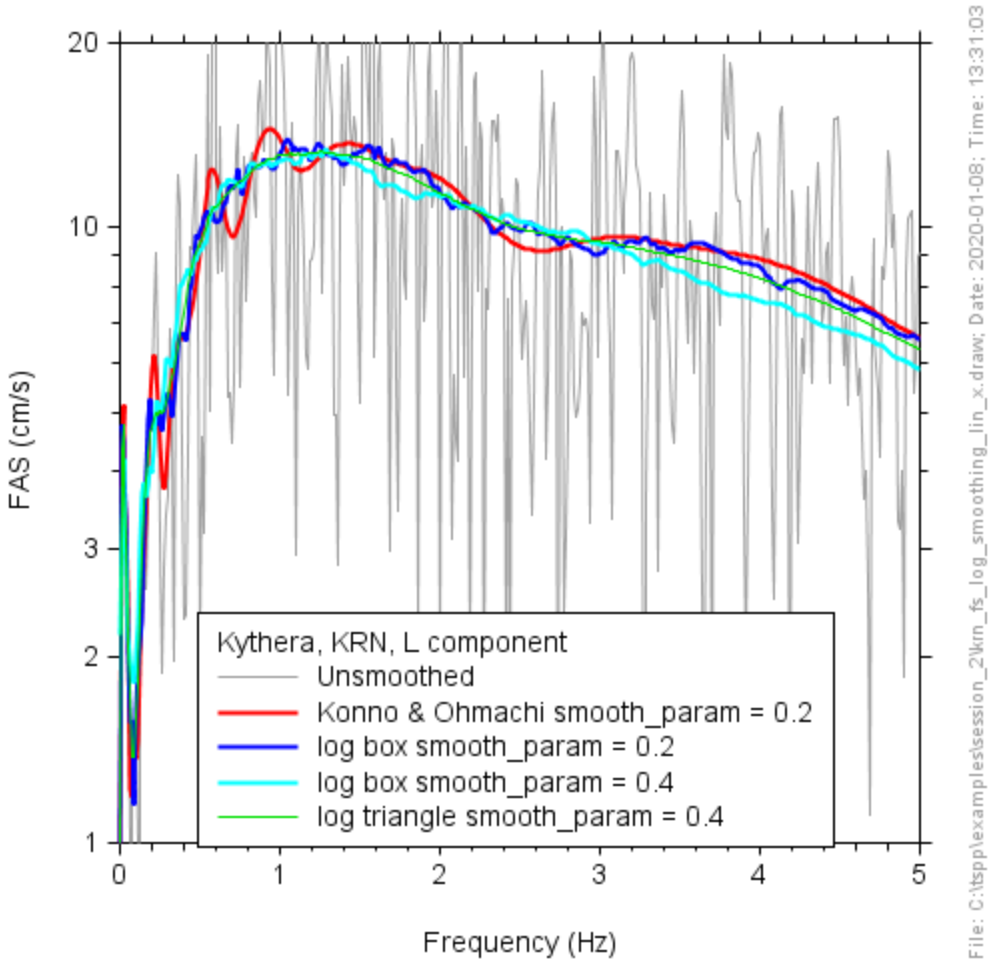

Figure 10. FAS without smoothing and with frequency-dependent smoothing, plotted using a linear frequency scale with a maximum frequency of 5 Hz.

All of the smoothing operators work well at low frequency, but the box function has more chatter than the other two (use the zoom button in Adobe Acrobat or Adobe Acrobat Reader to see this more clearly) and thus is the least desirable smoother (although in practical applications the difference between the three functions may not be important). The Konno and Ohmachi smoothing operator seems to be the preferable operator, although the program execution time is slower than the other operators because of the sine and log function evaluations; this is probably not relevant for most applications.

Session 3: Runs to help in choosing the low-cut filter corner frequency

It is always a good idea initially to compute and graph the Fourier amplitude spectrum (FAS) and to compute unfiltered velocity and displacements from zero-order-corrected (zoc) acceleration data (acceleration data from which the mean of either the pre-event portion of the record, if available, or the complete record is removed from the whole record). Decisions about further processing, including the need for higher-order baseline corrections or filtering, should be based on both the FAS, and on a review of the zoc velocities and displacements for physical

plausibility. Usually a low-cut filter is all the processing that is required. I show here two examples of choosing a low-cut filter. This section illustrates the use of the program **filt_plot_lc.for,** specifically written for use in deciding on the filter corner. The program does a series of filters with filter corners that are spaced equally in log frequency. The results (either velocity or displacement or both) are plotted on a single page, along with the original acceleration and the zoc velocity. Only two to four files are saved per run (counting the summary file and a file of response spectra, if requested), unlike **blpadflt**, which produces seven or eight files per run.

A time series with a relatively simple Fourier acceleration spectrum

The first example used the record KAV18501.L.dat.smc. This was recorded at an epicentral distance of 45 km from an **M** 5.2 earthquake at 4 km depth in Greece; the earthquake occurred on 09 November 1985. Figure 11 shows the FAS for the uncorrected record.

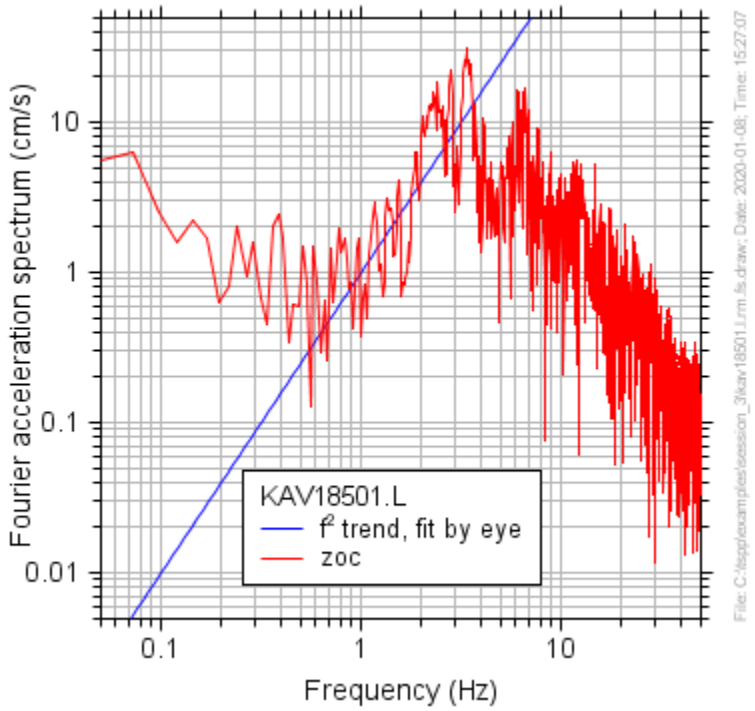

Figure 11. FAS for the sample record. The blue line has a slope of 2, as expected from simple source theory; its location was determined by eye.

This is a fairly typical spectrum, with a decrease in amplitudes close to the theoretically expected slope of 2 (in a log-log plot) and an increase in amplitude at low frequencies, presumably due to noise. (What is unusual is the relatively high frequency at which the FAS starts to level out; simple source theory would have this starting near the corner frequency of the earthquake, which is about 1 Hz for a generic magnitude 5 earthquake.) Just based on the shape of the FAS, a filter corner near 1 Hz is suggested. To see the consequence of the filter frequency on the displacement waveforms, the program **filt_plot_lc.for** was used to make plots of the filtered displacements for a series of log-spaced frequencies ranging from 0.2 Hz to 2 Hz. The control file is given by

filt_plot_lc.ctl.session_3_fig12. The program can create figures of the suite of filtered velocities, displacements, or both. In addition, it can compute response spectra for each filter and write to an output file. The control file above specifies that only displacements are plotted, and that 5%-damped response spectra are computed for 200 periods from 0.01 s to 90 s. The program produced the following plot:

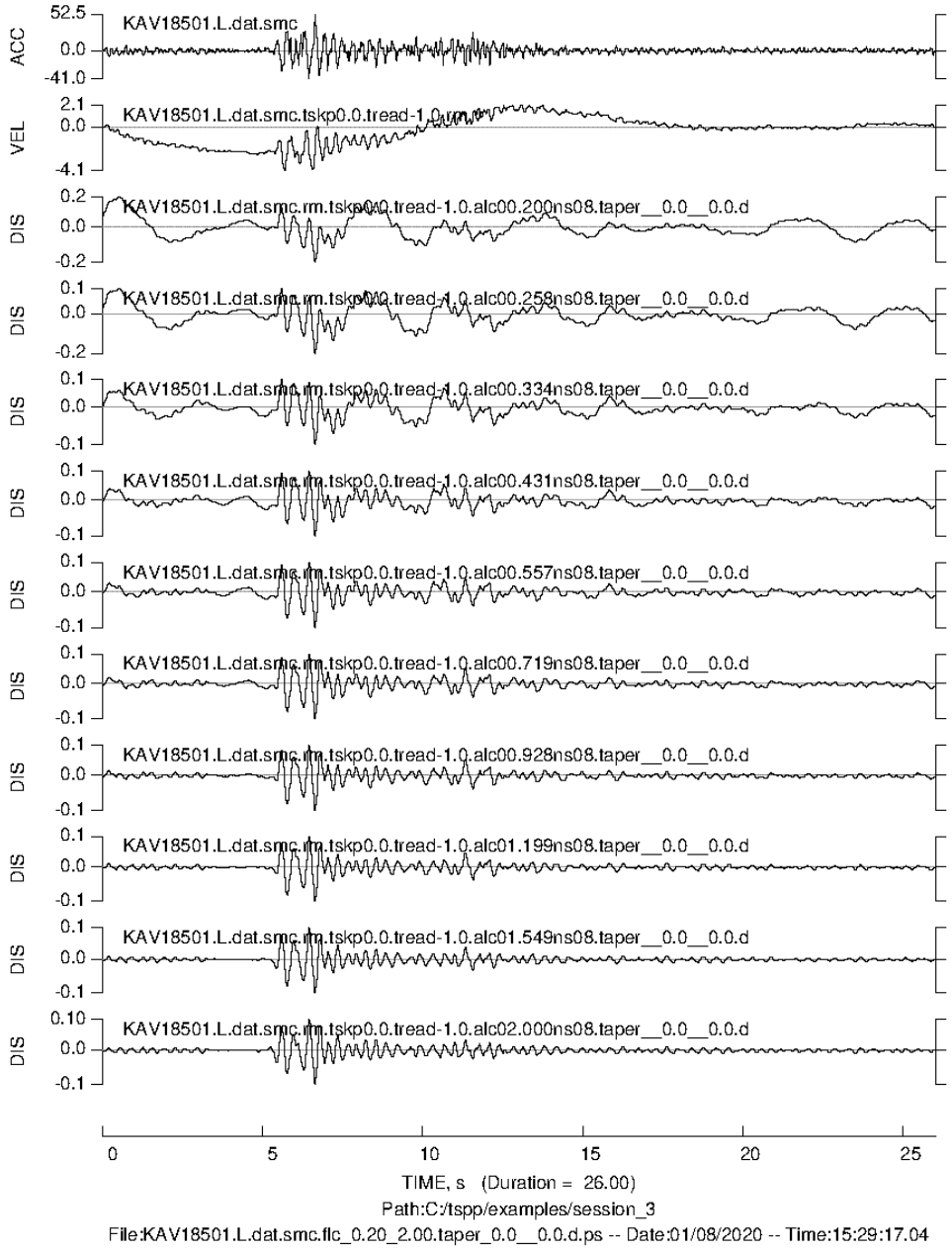

Figure 12. A suite of displacements from filtered accelerations. The filter frequencies are part of the time series name above each trace (e.g., "alc00.431ns08" indicates that an acausal low-cut filter with corner frequency of 0.431 Hz and an order such that the filter goes as f^8 at low frequencies was used). For ease of comparison, only pad-stripped time series are shown.

The small transients at the beginning and end of the time series (particularly evident for the lower-frequency filters) can be reduced by applying tapers, as shown in Figure 13, made using the control file *filt_plot_lc.ctl.session_3_fig13*.

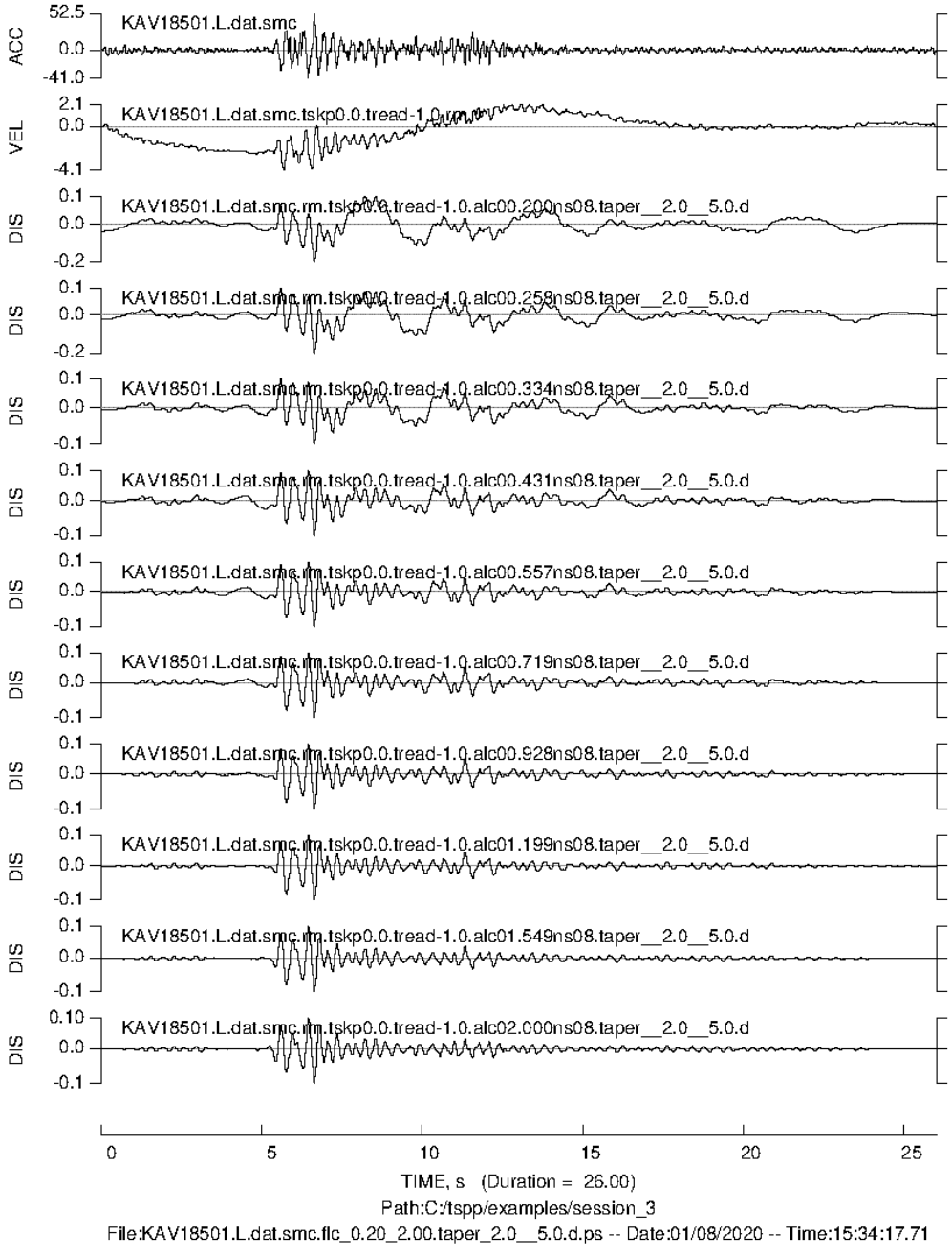

Figure 13. A suite of displacements from filtered accelerations. The filter frequencies are part of the time series name above each trace (e.g., "alc00.431ns08" indicates that an acausal low-cut filter with corner frequency of 0.431 Hz and an order such that the filter goes as f^8 at low frequencies was used). Tapers of 2 s and 5 s were applied to the beginning and end of the original record, respectively, before adding zero pads and filtering. For ease of comparison, only pad-stripped time series are shown.

A plot of response spectra can also be useful, in combination with the FAS and time series plots, in determining the filter corner frequency. Figure 14 shows both PSA and SD, the latter to emphasize the sensitivity of the low-frequency portion of the spectrum to the filter corner. For ease in comparing with the FAS plot, I've used frequency rather than period on the abscissa. The plot was made using the output of **filt_plot_lc.for** for all but one of the response spectra; for the response spectra corresponding to the zoc acceleration I used **smc2rs.for** (see *smc2rs.ctl.session_3_figs_14_18*).

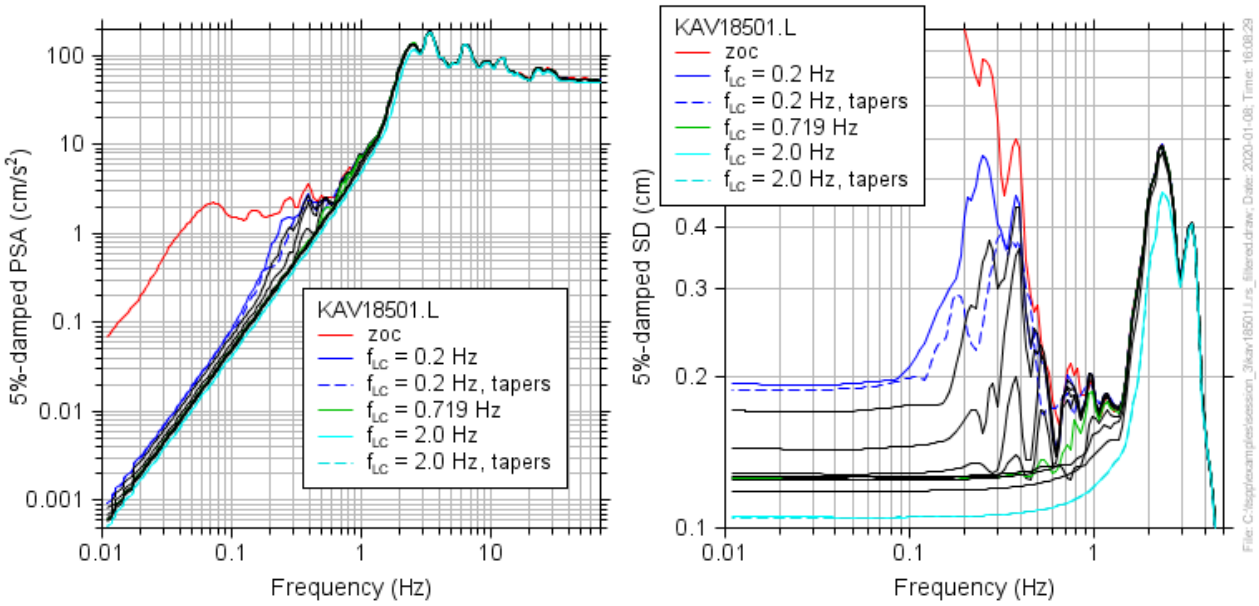

Figure 14. Response spectra for the zoc acceleration and for ten filters (only the response specta for three of the filters are distinguished by color). Also shown are the spectra for the tapered and filtered time series---clearly tapering makes little difference in the response spectra.

It is not the purpose of this report to provide a detailed discussion of how to choose filter corner frequencies, but a few comments are in order. Because ground-motion records from moderate events usually have distinct P- and S-wave arrivals, it is not expected that long period oscillations will occur before the S-arrival (unless close to the fault, when near- and intermediate-field terms can produce such motions). Thus the pre-S oscillations seen in Figures 12 and 13 for filters less than about 0.719 are not expected on physical grounds (although acausal filter transients can produce pre-event oscillations, as discussed in Boore, 2005b). Note also that there is an increase in the FAS around 0.4 Hz which seems to produce the large SD at frequencies around 0.2—0.4 Hz for the records filtered with a fairly low filter frequency. As seen in the figure above, this peak in SD is very sensitive to the filter frequencies (not surprising, given that the filters start at 0.2 Hz and thus quickly eliminate the motions leading the low-frequency peak in SD). Based on all of the figures above, I would choose a filter corner between 0.719 and 0.928 Hz.

A time series with a somewhat complicated Fourier acceleration spectrum

The second example used the record HER19401.L.dat.smc. This was recorded at an epicentral distance of 35 km from an **M** 6.1 earthquake at 90 km depth in Greece; the earthquake occurred on 23 May 1994. Figure 14 shows the FAS for the uncorrected record.

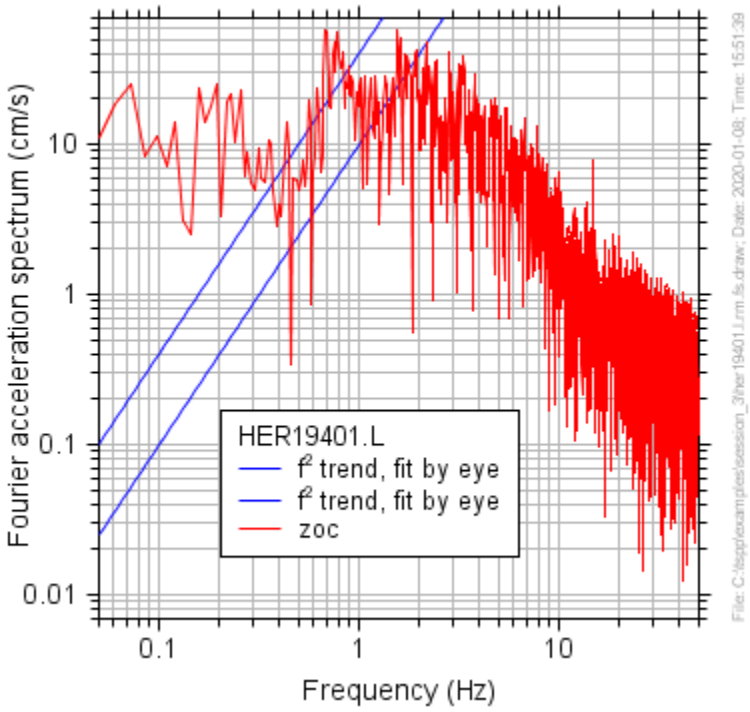

Figure 15. The FAS for the HER19401.L record (see the caption to Figure 11).

The spectrum for this event is considerably more complicated than for the previous event. In particular, there are two relative peaks in the FAS, one near 0.8 Hz and one near 1.5 Hz. Is the first one real? If so, the filter corner should be chosen less than 0.8 Hz. The displacement waveforms, with and without tapers, are shown in Figures 16 and 17.

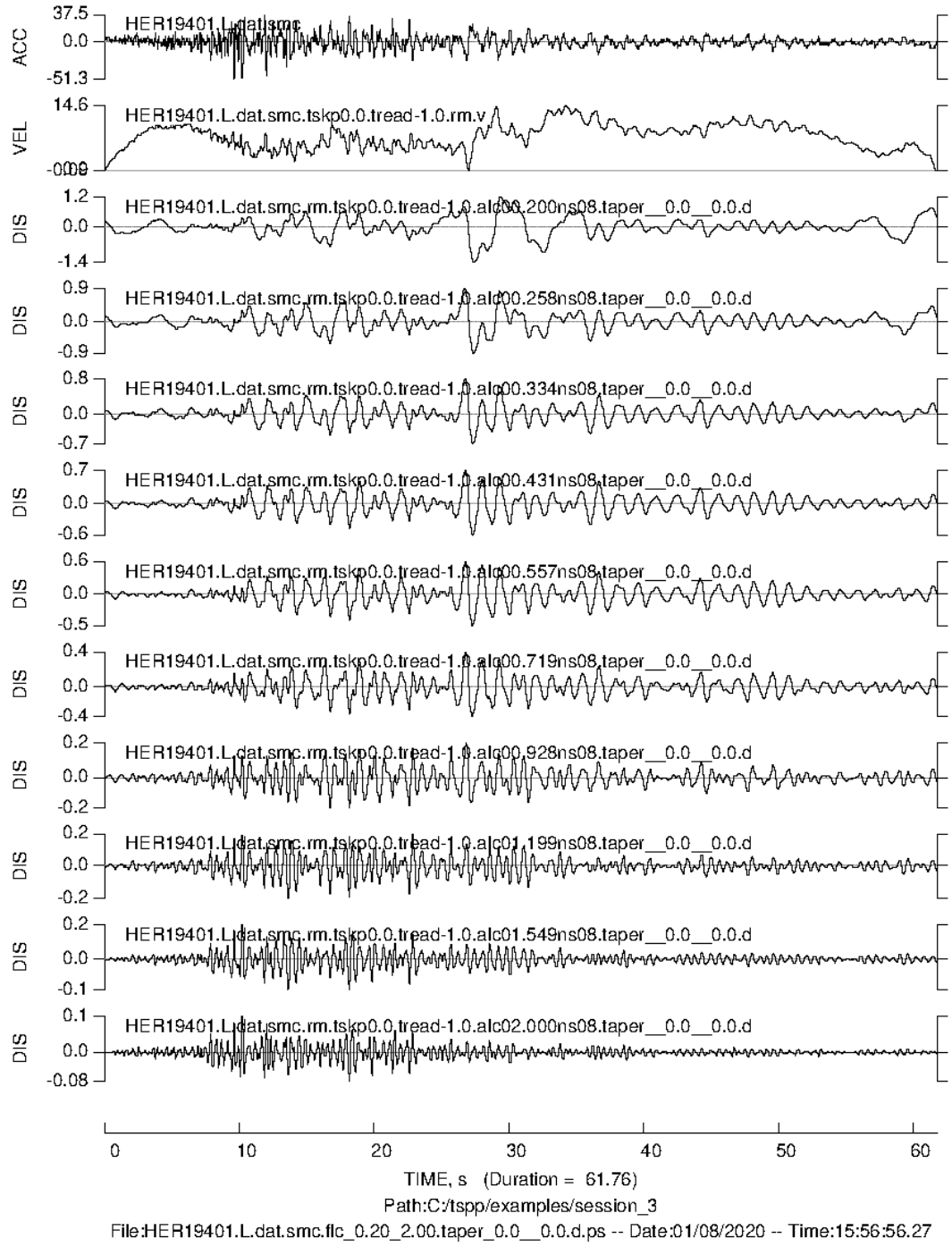

Figure 16. Filtered waveforms for the HER1940.L record (see Figure 12 caption for details).

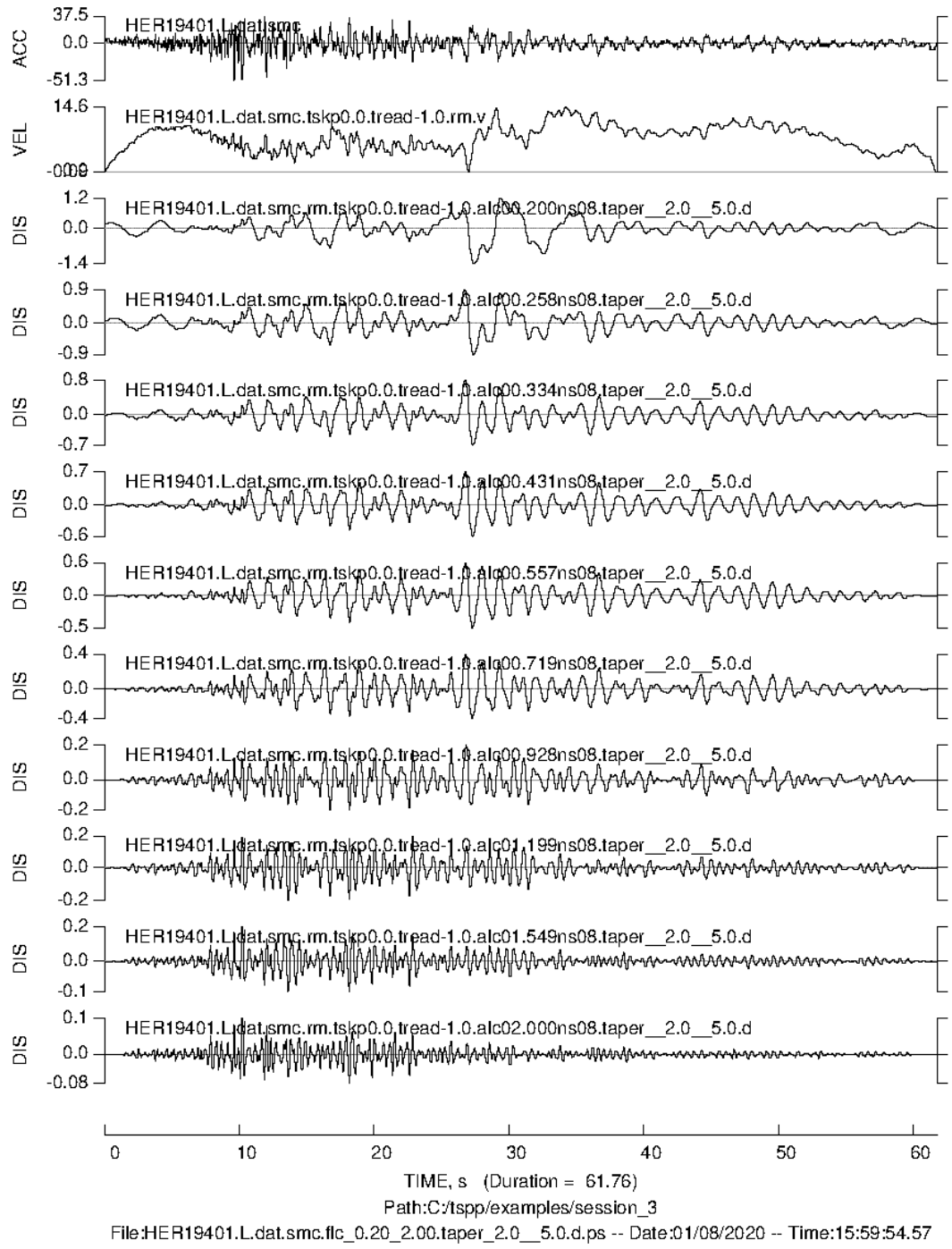

Figure 17. Filtered waveforms for the HER1940.L record, when tapers have been applied (see Figure 13 caption for details).

Note the fairly long-period phase at about 27 s. This phase probably contributes to the relative increase in the FAS around 0.2 Hz (Figure 14) and to the strong peak in SD near 0.2 Hz (Figure 18, below).

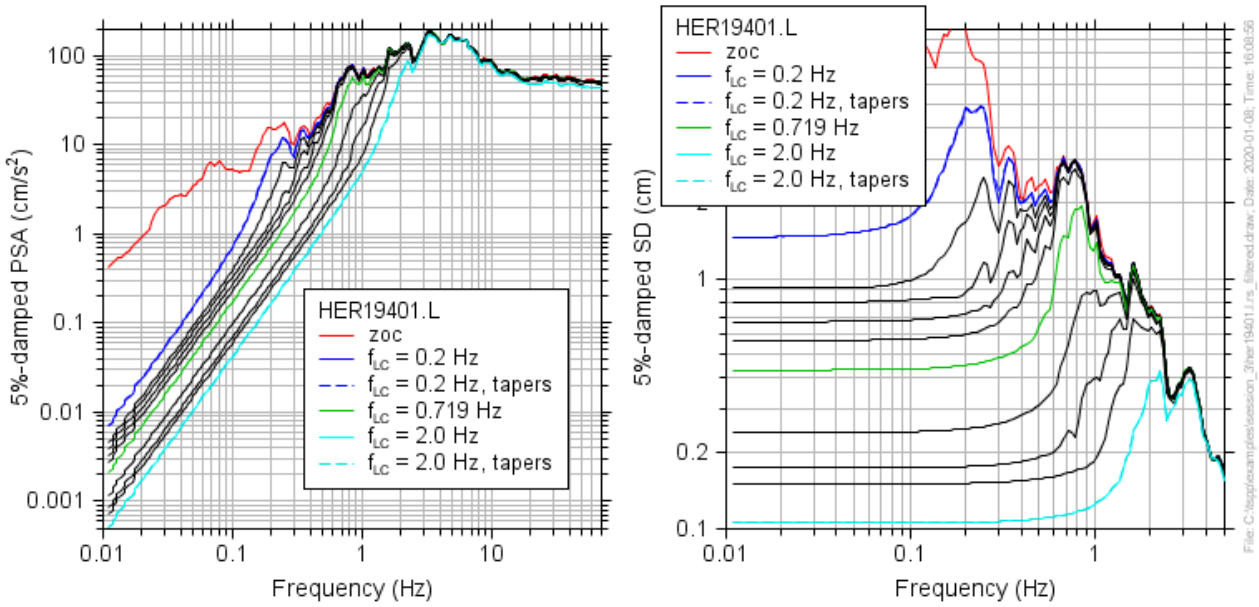

Figure 18. PSA and SD for the HER19401.L record (see the caption to Figure 14 for details).

Is the phase near 27 s real motion, or noise? It is hard to tell. It occurs well after the S-arrival and has a finite duration, as would be expected of a real phase. Also, as the event is deep and the recording station is on the island of Crete, there might be wave propagation effects associated with the subducting slab that will produce complexity in the recorded waveforms. On the other hand, there are other long-period oscillations throughout the record that might be due to noise. For this reason, I would choose a filter corner near 0.5 Hz. This will preserve the peak in SD around 0.8 Hz (note that filter corners of 0.719 Hz and higher strongly decrease the amplitude of the response spectrum around 0.8 Hz) but will eliminate the longer-period oscillations.

Session 4: Example of a simple baseline correction

In general, I find that low-cut filtering is sufficient to eliminate or reduce the long-period noise that is present in most digital ground-motion recordings. A consequence of this is that "static" coseismic displacements can not be determined from low-cut filtered records. To overcome this limitation, a number of papers have been published with algorithms for determining and removing long-period noise. These papers usually (always?) assume that the long-period noise is due to shifts in the acceleration baseline. When integrating to velocity and displacement these shifts show up as long period trends. In particular, a constant offset in acceleration produces linear and quadratic trends in the velocity and displacement time series, respectively. Many of the records of the 1999 Chi-Chi, Taiwan mainshock seem to be affected by a particularly simple baseline problem: a single offset occurring at a time that can be estimated by projecting the fit of a line to the velocity trace after the strong shaking part of the time series to its intersection with the zero line. I call this the "v0" baseline correction (Boore, 2001), and it is demonstrated in this session.

Figure 19 shows a map of stations that recorded the mainshock ground motions, as well as stations that provided coseismic static offsets. Of particular interest, of course, are situations where both kinds of stations are close to one another. A number of such station pairs can be seen in Figure 19, such as TCU102-G103 and TCU076-AF11 (on the footwall side of the fault) and

TCU074-HTZS, TCU052-M324, and TCU068-G104 (above the hanging wall of the fault). Because the hanging wall motions are larger than the footwall motions, I use the TCU052-M324 pair in this example. The distance between the stations is 2.7 km; a closer pair is TCU074-HTZS, the distance between them being 2.0 km. But TCU074 is one of the very rare examples where double integration of the acceleration produces a stable estimate of the static offset, with no baseline correction. For this reason I obviously cannot use the recording at TCU074 to demonstrate the baseline correction procedure, as no correction is needed!

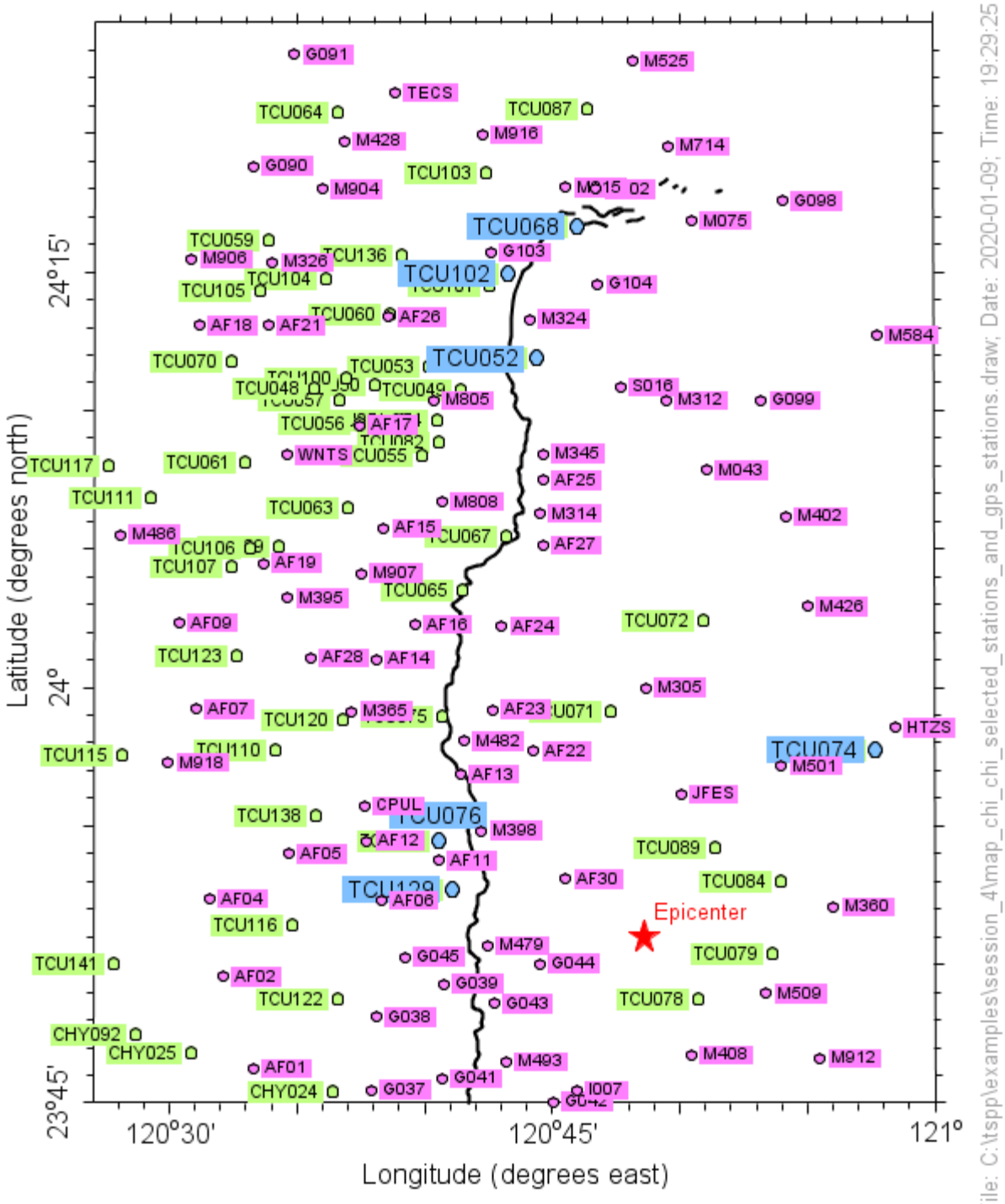

Figure 19. Map of GPS stations used on Yu et al. (2001) (magenta) and strong-motion recording stations (light green, light blue), along with the surface trace of the 1999 Chi-Chi mainshock.

As in the previous sessions, the first step is to plot the results of the zoc procedure. The resulting velocity time series for the three components of motion are shown in Figure 20.

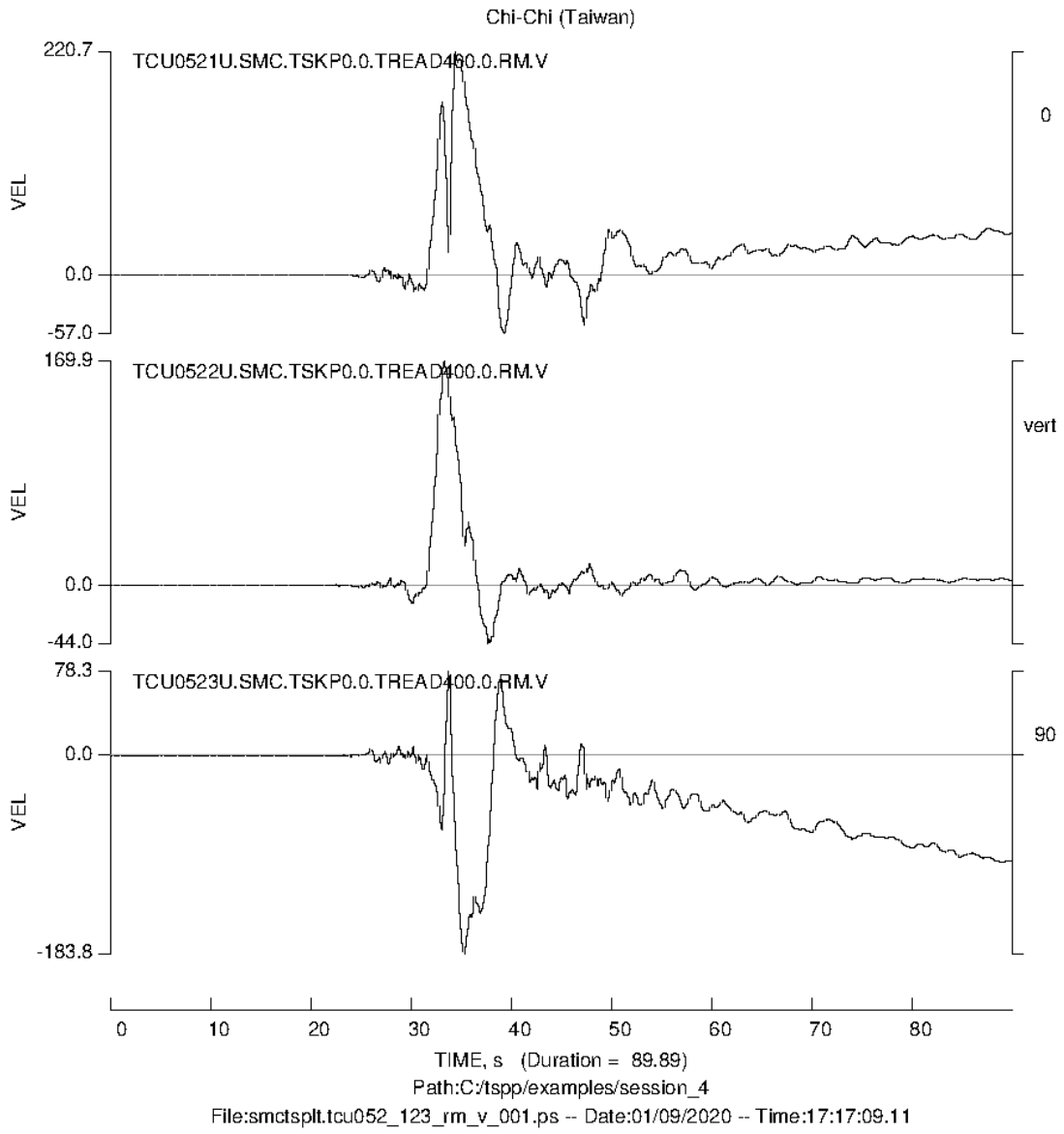

Figure 20. Velocity time series resulting from the zeroth-order correction (zoc) applied to the acceleration time series (a mean determined from 0 s to 15 s was removed from the whole acceleration time series, and the result was integrated to velocity).

The linear trends in velocity are clearly visible, suggesting that there was a single offset in the acceleration time series. A line was fit to the segment of each velocity time series starting after the strong shaking. I used baseline correction option 3 in **blpadflt.ctl**, with the line fit to the velocity time series between 65.0 s and 89.895 s (the coefficients of the regression can depend on the portion being fit, and in practice this should be studied by trying different start and stop times). The line fit to the EW component is shown in Figure 21. The line intersects the zero-velocity line

at a time of 34.985 s; a step adjustment in the acceleration time series was applied from this time to the end of the record, with an amplitude equal to the negative of the slope of the fitted velocity line (-1.81743 cm/s/s). In terms of the Boore (2001) terminology, t1 = t2 = 34.985 s.

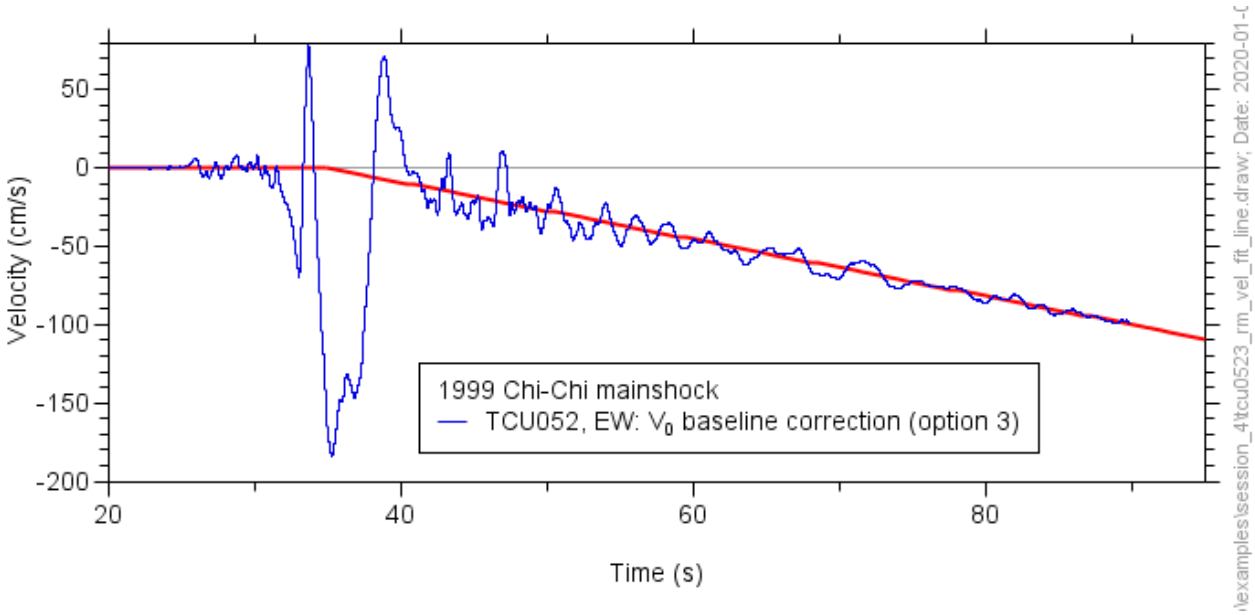

Figure 21. The velocity time series from the zoc acceleration and the line fit to the velocity time series between 34.985 s and 89.895 s.

The resulting displacement time series for the three components at TCU052 after applying the option 3 baseline correction are shown in Figure 22. Also shown are the GPS displacements from Yu et al. (2001) for the closest GPS station to TCU052 (2.7 km). The displacement time series from the baseline-corrected accelerations are essentially constant after the main shaking portion of the ground motion. The residuals displacements have the same sense as the GPS displacements with similar, if not the same, amplitudes. It is not my purpose here to investigate the reasons for any discrepancies between the residual displacements derived from the two types of instruments (e.g., there could be spatial variations in the ground motions), but I will point out that a map of the residual displacement horizontal vectors shows close similarities in the displacement fields from both estimates of the residual displacements (e.g., Oglesby and Day, 2001).

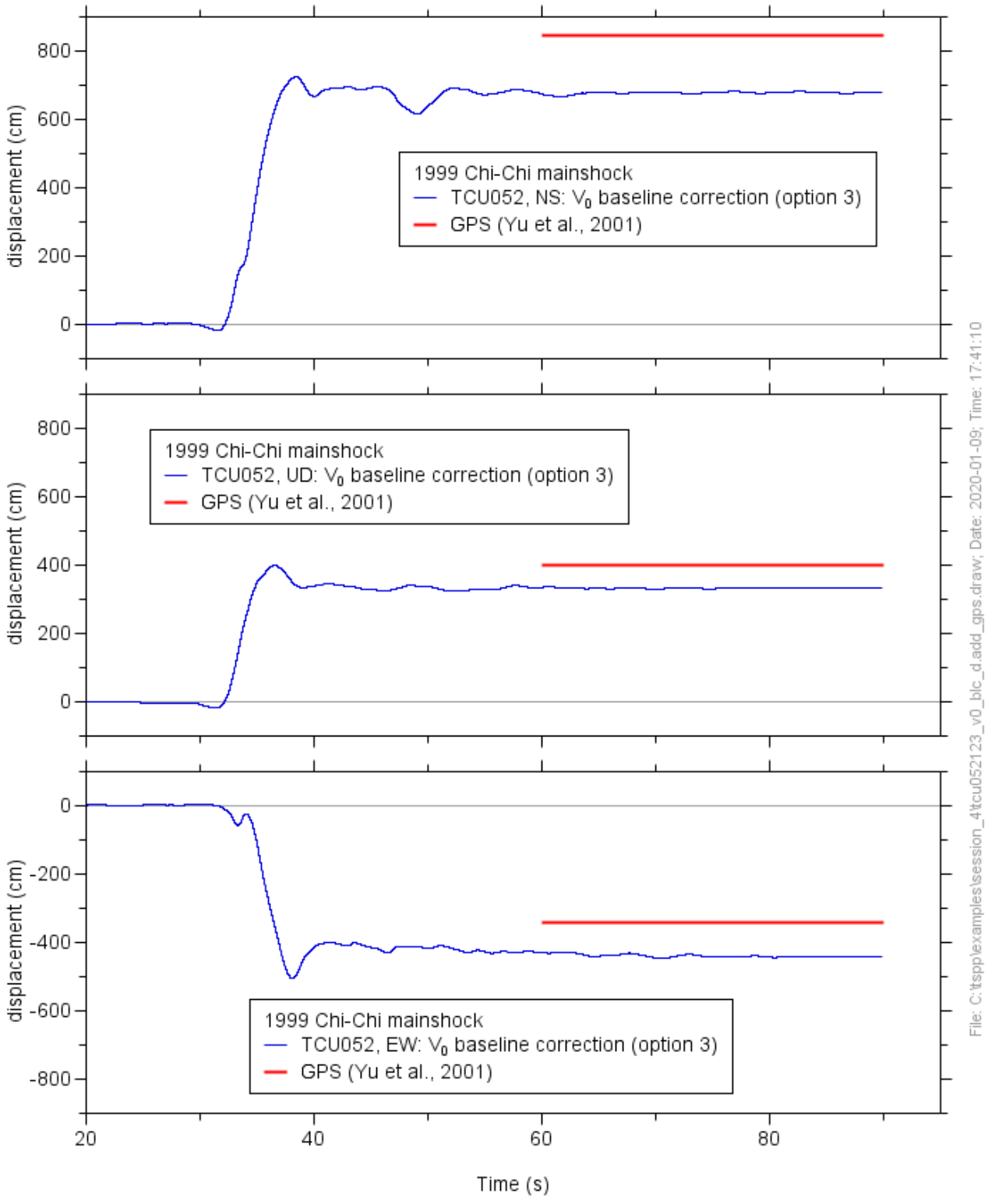

Figure 22. (blue) Velocity time series from option 3 baseline-corrected acceleration time series (blue) and (red) coseismic displacements from a nearby GPS station (M324, 2.7 km from TCU052).

A General Comment about using Padded Time Series

Fourier spectra, response spectra, and velocity and displacements of filtered data should always be computed from the complete, padded time series, not the pad-stripped data (see Boore, 2005b). Pad-stripped time series are often provided by data agencies, but ground-motion intensity measures such as peak velocity and response spectral amplitudes provided by the agencies are (or should be) obtained from the complete padded and filtered acceleration time series. It is inconsistent for the user to compute these ground-motion intensity measures from the pad-stripped acceleration time series (this also implies that using the pad-stripped acceleration time series for any analysis, such as the response of a nonlinear structure, is formally an inconsistent use of the time series). (See Boore *et al.*, 2012, for more discussion.)

Acknowledgments

I thank Charles Mueller for providing the Fortran source code for the original versions of several subroutines and Chris Stephens and Sinan Akkar for reviews of the report. John Douglas and Basil Margaris supplied the data used in the sample sessions. A number of these programs benefitted greatly from suggestions made by Chris Stephens; I give him credit for his suggestions in the modification logs contained at the end of the headers in each program, just before the beginning of the source code.

References

Boore, D. M. (2001). Effect of baseline corrections on displacements and response spectra for several recordings of the 1999 Chi-Chi, Taiwan, earthquake, *Bull. Seismol. Soc. Am.* **91**, 1199–1211.

Boore, D.M., 2003a, Analog-to-digital conversion as a source of drifts in displacements derived from digital recordings of ground acceleration: *Bulletin of the Seismological Society of America*, v. 93, p. 2017-2024.

Boore, D.M., 2003b, Some notes on phase derivatives and simulating strong ground motions: *Bulletin of the Seismological Society of America*, v. 93, p. 1132-1143.

Boore, D.M., 2005a, Long-period ground motions from digital acceleration recordings; a new era in engineering seismology, in *Directions in Strong Motion Instrumentation*, P. Gülkan and J. G. Anderson, Editors, Springer, Dordrecht, The Netherlands, p. 41-54.

Boore, D.M., 2005b, On pads and filters: Processing strong-motion data: *Bulletin of the Seismological Society of America*, v. 95, p. 745-750.

Boore, D.M. (2010). Orientation-independent, non geometric-mean measures of seismic intensity from two horizontal components of motion, *Bull. Seismol. Soc. Am.* **100**, 1830–1835.

Boore, D.M. and Akkar, S., 2003, Effect of causal and acausal filters on elastic and inelastic response spectra: *Earthquake Engineering and Structural Dynamics*, v. 32, p. 1729-1748.

Boore, D.M. and Bommer, J.J., 2005, Processing of strong-motion accelerograms: Needs, options and consequences, *Soil Dynamics and Earthquake Engineering*, v. 25, p. 93-115.

Boore, D.M., Joyner, W.B., and Fumal, T.E., 1994, Estimation of response spectra and peak accelerations from western North American earthquakes: An interim report, Part 2, *U. S. Geological Survey Open-File Report* 94-127, 40 p.

Boore, D.M., Joyner, W.B., and Fumal, T.E., 1997, Equations for estimating horizontal response spectra and peak acceleration from western North American earthquakes: A summary of recent work, *Seismological Research Letters*, v. 68, p. 128–153.

Boore, D.M., Watson-Lamprey, J., and Abrahamson, N.A., 2006, Orientation-independent measures of ground motion: *Bulletin of the Seismological Society of America*, v. 96, p. 1502-1511.

Boore, D. M., A. Azari Sisi, and S. Akkar (2012). Using pad-stripped acausally filtered strong-motion data, *Bull. Seismol. Soc. Am.* **102**, 751-760.

Jousset, P. and Douglas, J., 2007, Long-period earthquake ground displacements recorded on Guadaloupe (French Antilles), *Earthquake Engineering and Structural Dynamics*, v. 36, p. 949-963.

Konno, K. and Ohmachi, T., 1998, Ground-motion characteristics estimated from spectral ratio between horizontal and vertical components of microtremor: *Bulletin of the Seismological Society of America*, v. 88, p. 228-241.

Oglesby, D. D., and S. M. Day (2001). Fault geometry and the dynamics of the 1999 Chi-Chi (Taiwan) earthquake, *Bull. Seismol. Soc. Am.* **91,** 1099–1111.

Press, W.H., Teukolsky, S.A., Vetterling, W.T., and Flannery, B.P., 1992, *Numerical Recipes in FORTRAN: The Art of Scientific Computing, Second Edition*, Cambridge University Press, Cambridge, England, 963 pp.

Yu, S.-B., L.-C. Kuo, Y.-J. Hsu, H.-H. Su, C.-C. Liu, C.-S. Hou, J.-F. Lee, T.-C. Lai, C.-C. Liu, C.-L. Liu, T.-F. Tseng, C.-S. Tsai, and T.-C. Shin (2001). Preseismic deformation and coseismic displacements associated with the 1999 Chi-Chi, Taiwan, earthquake, *Bull. Seismol. Soc. Am.* **91**, 995–1020.