User's Manual

BCSEIS

(Bispectrum Cross-correlation package for SEISmic events)

Wen-xuan (Wayne) Du and *Clifford H. Thurber*

Dept. of Geology & Geophysics
University of Wisconsin-Madison
1215 W Dayton Street
Madison WI 53706

January, 2004

1. Introduction

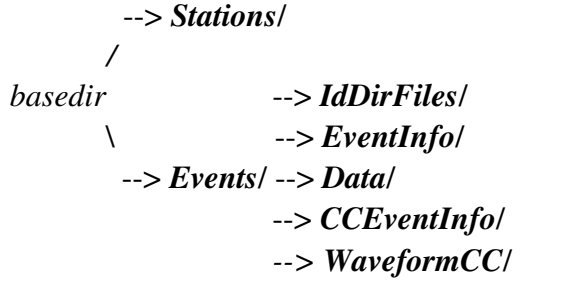
BCSEIS (**B**ispectrum **C**ross-correlation package for **SEIS**mic events) is a software package developed at the University of Wisconsin-Madison to obtain reliable waveform-based differential times for a group of earthquakes. The differential times (or equivalent time delays) are calculated for the waveforms of event pairs recorded at the same station. They can then be used to improve event relocation results. Basically, time delays are calculated with both the cross-correlation (CC) technique (using the band-pass filtered waveforms) and the bispectrum (BS) method (using both the raw and band-pass filtered waveforms). The two BS time delay estimates are used to verify (select or reject) the one computed with the CC technique.

BCSEIS consists of several core computational C++ and C programs and perl scripts. It currently works with waveform data in SAC format only. Users need to either use existing or write their own conversion program to convert their data into SAC format. This package currently runs on both Unix (Solaris) and Linux systems. It requires softwares ***sac2000*** and ***perl*** for support. The C++ and C programs can be compiled with the widely available GNU ***g++*** and ***gcc*** compiler. The ***CC*** compiler on the Solaris machine also can compile them properly.

BCSEIS is currently still under fine tuning and by no means is a final product. Researchers in the community are welcome to apply it to your dataset and send us the feedback. We have made an effort to make the package user friendly, but it may not satisfy the users thoroughly. Generally, by typing the name of a program on the command line the users will be given the usage information needed.

2. Directory Structure

The ***BCSEIS*** package requires that the users set up the following directory structure for a dataset before further processing.



Each earthquake is assigned a unique event id number <*evId*> and all of its associated waveform data are put in a unique data directory <*evDir*>.

Subdirectory ***Stations/*** contains a parameter file '***station.dat***' that gives the locations of the seismic stations providing the waveform data. Each line has a format like:

<sta> <stla> <stlo>

where <*sta*> is the station code; <*stla*> is the station latitude (degrees; positive value for N and negative values for S); <*stlo*> is the station longitude (degrees; positive value for E and negative values for W). For example:

BHW -41.4092 174.8714

Subdirectory ***IdDirFiles/*** contains a parameter file '***iddir.dat***' that associates the event id numbers with the event data directory names. Each line has two entries:

<evId> <evDir>

For example:

19900001 19900101021252

Currently the <*evDir*> for an event is named after its catalog origin time. In the above example, '1990' is the calendar year; '01' is the month; '01' is the day; '02' is the hour; '12' is the minute and '52' is the second. If the above naming convention fails to differentiate two events occurred within a very short time range, additional digits [0-9] can be appended to make each <*evDir*> unique. ***BCSEIS*** does not care how many characters <*evDir*> consists of.

Subdirectory ***EventInfo/*** contains a parameter file '***events.catalog***' that stores the catalog event information. Each line has the following format:

<evId> <yr> <mn> <dy> <hr> <mi> <sc> <lat> <lon> <dep> <mag>

For example:

19900001 1990 1 1 2 12 52.1080 -41.7505 174.2075 11.41 2.35

Subdirectory ***Data/*** contains subdirectories with names <*evDir*> that store the waveform data for each individual earthquake. Currently only SAC data format is supported. Users need to either use existing or write their own conversion program to convert their data in other format to SAC. The naming convention for raw (unfiltered) data is '<sta>.[N/E/Z].SAC', while the filtered data have 'filt.' as prefix. ***BCSEIS*** requires that certain header variables of the SAC files have values properly assigned before further processing. The list is:

a: catalog P pick;
t0: catalog S pick;
o: catalog event origin time;
evla: catalog event latitude;
evdp: catalog event depth;
user0: weight for the P pick [0 – 1];
user1: weight for the S pick [0 – 1];
delta: sampling rate;
evlo: catalog event longitude;

Subdirectory ***CCEventInfo/*** contains a parameter file '***ccEvInfo.dat***', which provides the lists of stations having catalog P and S picks for the events. Each event has a separate section. The first line has the following format:

 # <evId> <evla> <evlo> <evdp> <evDir> <pNum> <sNum>

For example:

19900001 -41.7505 174.2075 11.400 19900101021252 3 2

It is then followed by <pNum> lines, each of which contains the name of a station having

catalog P pick for the event. They are then followed by <sNum> lines of station names that have catalog S picks for the event.

3. Data Preparation

Given a certain waveform dataset, the process of data preparation can be carried out in the following way:

<1> Build a unique working directory '*basedir*' and assign an unique project code to the dataset. For example '*nz*' is used as the project code for a New Zealand dataset. Users need to modify the perl script '***chkProjCode.pl***'.

The last portion of the script '***chkProjCode.pl***' is shown below:

```
-------------------------------------------------------------------------------------------------
-
if ($projCode eq 'CTBT' || $projCode eq 'ctbt') {
        print "1 /home/dxw/CTBT\n";
} elsif ($projCode eq 'NZ' || $projCode eq 'nz') {
        print "2 /home/dxw/NewZealand\n";
} elsif ($projCode eq 'nz2') {
        print "2 /u91/dxw/NewZealand\n" ;
} elsif ($projCode eq 'PF' || $projCode eq 'pf') {
        print "3 /home/dxw/Parkfield\n";
} elsif ($projCode eq 'SD' || $projCode eq 'sd') {
        print "4 /home/dxw/Sendai\n";
} elsif ($projCode eq 'TVZ' || $projCode eq 'tvz') {
        print "5 /home/dxw/TVZ\n";
} else {
        print "-1\n";
}
-------------------------------------------------------------------------------------------------
-
```

For the two underlined lines, we set the $projCode to 'NZ' and 'nz' (thus case insensitive). '*basedir*' is set to '/home/dxw/NewZealand'. '2' is set as the project number (*projnum*) for this dataset. Users can easily modify the above script for their own datasets.

<2> Build subdirectory '***Stations/***' and construct the station file '***station.dat***'.

<3> Build subdirectory '***Events/***' and its five subdirectories.

<4> Waveform data are put under '***Data/***' on an event basis. In this step, conversion may be needed to change the waveforms from other formats to SAC. Using the phase pick files and catalog hypo files, required SAC header variables need to be assigned with proper values. Files '***iddir.dat***' and '***events.catalog***' must be generated as well. Because different dataset usually have their own formats for waveforms, phase pick files and hypo files, generally the users have to write their own program

to handle this step.

<5> run '***calcCCEvInfo3.pl***' inside directory ***CCEventInfo/*** to generate the parameter file '***ccEvInfo.dat***'. Usually a shell script like below is used:

```
-----------------------------------------------------------------------------------------------------
-
#!/bin/sh

projCode=nz
evIdList="id.dat"
staList="station.list"

echo "cut -f1 ../IdDirFiles/iddir.dat > $evIdList"
cut -f1 ../IdDirFiles/iddir.dat > $evIdList

echo "calcCCEvInfo3.pl $projCode $evIdList $staList"
calcCCEvInfo3.pl $projCode $evIdList $staList
-----------------------------------------------------------------------------------------------------
-
```

<6> Inside the '***Data/***' directory, band-pass filter the waveform data with appropriate parameters using **sac2000**.
Step 1: Generate the lists of SAC files for each individual stations with perl script 'getStaStat.pl'.
Example: *getStaStat.pl station.list dataDir.list 2> /tmp/sta.err*
where 'station.list' is a list of station names (one station per line); 'dataDir.list' is a list of event directories <evDir>.
Result: A directory 'StaStat/' is generated, which contains the lists of SAC files for each individual stations.

Step 2: Invoke perl script '**filtSacList2.pl**' to filter the SAC files for each station. Usually a perl script '**filtAll.pl**' is used to perform the filtering for all stations. (An example copy of 'filtAll.pl' like below is provided in 'scripts/' directory).

```
-----------------------------------------------------------------------------------------------------
-
#!/usr/bin/perl -w

$lf = 1.5;
$hf = 10;
$poleNum = 3;
$passNum = 2;
$stationFile = "station.list";

open(STATION,"$stationFile") or die "Can not open station file '$stationFile' for reading: $!\n";
while ($sta=<STATION>) {
        chomp $sta;
        $listFile = "${sta}.list";
```

```
        system "cp StaStat/$listFile .";
        print ">>> filtSacList2.pl $listFile $lf $hf $poleNum $passNum\n";
        system "filtSacList2.pl $listFile $lf $hf $poleNum $passNum";
        system "rm filt.${listFile} $listFile";
    }
    close(STATION);
```

---

where $lf and $hf are the low-pass and high-pass corner frequencies. They should be modified for different datasets.

4. Data Processing

After the necessary preparation work is done, users can invoke program '***bcseis***' to make the time delay calculations. Usually a shell script like the example below is used:

```
#!/bin/sh

date
baseDir='/home/dxw/TVZ'
distCutoff=5
preNum1=30
posNum1=97
preNum2=50
posNum2=141
segLen1=64
segLen2=128
lf=0
hf=10
velCode='NAN'

startNum=1
bcseis 0 $baseDir $distCutoff $preNum1 $posNum1 $preNum2 $posNum2 $segLen1
$segLen2 $lf $hf $velCode $startNum

date
```

The output from '**bcseis**' is a list of '<evId>.CC' files. <evId> is the event id of an event and '<evId>.CC' contains the time delay information calculated between <evId> and other events. Below shows a portion of the file '19950001.CC' for event '19950001'.

```
-----------------------------------------------------------------------------------------------
# 19950001 19950002
I03     0.0033  0.8993  P      -2     -2     -3      0.3306  0.0000
I04     0.0000  0.6948  P      1      0      0       0.0000  0.0000
O19     0.0000  0.7760  P      -11    -11    -12     0.0000  0.0000
PAT     0.0000  0.7801  P      1      1      1       0.0000  0.0000
I03     -0.0100 0.7965  S      -23    -23    -23     0.0000  0.0000
I04     0.0100  0.8735  S      21     21     21      -0.4313 0.6886
PAT     -0.0195 0.9157  S      -5     -5     -5      0.0254  0.2336
# 19950001 19950003
I03     -0.1560 0.8982  P      -6     -6     -5      0.4961  0.2561
O19     -0.2910 0.7536  P      -1     -1     0       0.0000  0.0000
TAZ     2.5390  0.5385  P      -10    0      -999    0.0000  0.0000
PAT     -0.9810 0.5310  S      24     10     -999    0.0000  0.0000
-----------------------------------------------------------------------------------------------
```

In the above example, '# 19950001 19950002' starts the section for event pair '19950001' and '19950002'. The subsequent lines shows the time delay information for each common station of the event pair. There are 9 entries per line:

#1: station name;
#2: CC differential times (in seconds)
#3: CC coefficient
#4: phase name (P or S)
#5: CC time delay on filtered waveforms (in lags)
#6: BS time delay on filtered waveforms (in lags)
#7: BS time delay on raw waveforms (in lags)
#8: subsample time delay (in lags)
#9: standard deviation for the estimated subsample time delay (in lags)

After the calculation finishes, users need to run '***getDtFromCC.pl***' to extract the bispectrum-verified CC differential times from the previously generated '<evId>.CC' files. The output file 'dt.cc[0-3]' is used as input file to either '***hypoDD***' or '***tomoDD***'. An example is:

```
-----------------------------------------------------------------------------------------------
#!/bin/sh
getDtFromCC.pl 3 ccdir.list 0.50 0.70 0.80 2 bcseis.id dt.cc3
-----------------------------------------------------------------------------------------------
```

5. Main Programs

calcCCEvInfo3.pl *(perl script)*
**** program to generate the information of phase availability (P&S) at various stations***

Usage: calcCCEvInfo3.pl <projCode> <evIdList> [staList]
<projCode>: project code;
<evIdList>: event id list;
[staList]: station list to check the pick availability;

This program calculates the event info for CC purpose.
'ccEvInfo.dat' stores the output
'ccEvInfo.warn' contains the warning messages.

Version: 3.0 [This version adds in the 'staList' option to confine the checking range]

bcseis (C++ program)
**** program to perform the delay time calculations for a group of earthquakes***

Usage: bcseis <option> <baseDir> <distCutOff> <preNum1> <posNum1>
<preNum2> <posNum2> <segLen1> <segLen2> <lf> <hf>
<velCode> <startNum>
Purpose: This program calculates waveform-based differential times (or time delays) using both cross-correlation and bispectrum methods.
<option>: 0, 1, 2
0 - require input file 'bcseis.id';
- perform calculations for all events in a single run;
1, 2 - require input files 'bcseis.id1' and 'bcseis.id2';
- mainly used to take advantage of the dual CPU power of denali;
1 - perform calculations for 2 groups of events consecutively;
2 - perform calculations between 2 groups of events;
<baseDir>: directory path for the dataset to be processed;
<distCutOff>: cutoff event-event distance beyond which time delay calculations are not performed;
<preNum1>: number of sample points before the P arrival (for data window selection);
<posNum1>: number of sample points after the P arrival;
<preNum2>: number of sample points before the S arrival (for data window selection);
<posNum2>: number of sample points after the S arrival;
<segLen1>: number of sample points in each segment when dividing the P data

window;
<segLen2>: number of sample points in each segment when dividing the S data window;
<segLen1> and <segLen2> must be power of 2;
<lf>: lowest frequency to perform phase spectrum fitting (for sub-sample estimation);
<hf>: highest frequency to perform phase spectrum fitting (for sub-sample estimation);
<velCode>: code of velocity model to compute theoretical S-wave travel time, 'NAN' otherwise;
<startNum>: starting event number to continue a previously unfinished run;
default value is 1; meaningless if option is set to 1;

Version: 1.02
Author: Wen-xuan(Wayne) Du -- dxw@geology.wisc.edu

getDtFromCC.pl *(perl script)*
* Program to extract the bispectrum-verified CC differential times

Usage: getDtFromCC.pl <option> <ccDirList> <lCCLim> <cCCLim> <uCCLim> <diffLim> <evIdList> <dtFile>
<option>: 0 - no limit control over CC-BS lag difference;
1 - CC-BS lag difference 1 || 2 <= diffLim;
2 - CC-BS lag difference 1 <= diffLim;
3 - CC-BS lag difference 1 && 2 <= diffLim;
<ccDirList>: list of directory names where CC results are kept;
<lCCLim>: lower CC limit
<cCCLim>: center CC limit;
<uCCLim>: upper CC limit;
<diffLim>: limit in CC-BS lag difference;
<evIdList>: event id list;
<dtFile>: output dt.cc file;

6. Other Auxiliary Programs

chkProjCode.pl *(perl script)*
Usage: chkProjCode.pl <projCode>
<projCode>: project code (currently supported are 'ctbt', 'nz' and 'pf'; case insensitive)
Output: -1 for incorrect 'projCode'; (projnum, basedir) if correct

This program check the correctness of 'projCode'.

** Attention: Users need to modify this script to put in the unique 'projCode', 'projnum' (an integer identifying the number of the dataset) and 'basedir' (directory path for the dataset) for their dataset.*

getStaStat.pl (perl script)
Usage: getStaStat.pl <staFile> <dirList>
<staFile>: list of station names to check on
<dirList>: list of data directories to check on

This program generates the lists of SAC files for each individual stations in <staFile>.

filtSacList2.pl (perl script)
Usage: filtSacList2.pl <listFile> <lf> <hf> <poleNum> <passNum>
<listFile>: file containing a list of SAC files to be filtered
<lf>: low frequency for filtering
<hf>: high frequency for filtering
<poleNum>: number of poles;
<passNum>: number of passes;

This program bandpass filters a list of SAC files and output a list of filtered SAC files.

getsachdr (C++ program)
Purpose: Get the value of a header variable in a SAC file.
Usage: getsachdr <sacFile> <hdrKey1> [hdrKey2] ...
<sacFile>: SAC file name
<hdrKey>: Header key for value checking

List of hdrKey currently supported are:
nzyear nzjday nzhour nzmin nzsec nzmsec nzdttm kzdate kztime npts
delta odelta b e o a f ko ka kf
t0 t1 t2 t3 t4 t5 t6 t7 t8 t9
kt0 kt1 kt2 kt3 kt4 kt5 kt6 kt7 kt8 kt9
knetwk kstnm stla stlo stel stdp sloc cmpaz cmpinc kcmpnm
kevnm evla evlo evel evdp eloc dist az baz gcarc
user0 user1

This program will output space separated header values for all requested header keys in sequence.

setsachdr (C++ program)
Purpose: Set the values of a set of header variables in a SAC file.
Usage: setsachdr <sacFile> <hdrKey1> <hdrVal1> [hdrKey2] [hdrVal2] ...
<sacFile>: SAC file name
<hdrKey>: Header key for value setting
<hdrVal>: Header value

List of hdrKey currently supported are:

nzyear nzjday nzhour nzmin nzsec nzmsec
o ko a ka t0 kt0
stla stlo stel stdp
kevnm evla evlo evel evdp
user0 user1 delta

Note that when 'evla' and 'evlo' are both set in command line, this program will also set
station-event info parameters (dist,az,baz,gcarc) to the header if station info is in the header.

7. Installation

(1) unzip and untar the package file (usually 'bcseis.tar.gz'). The user will obtain a directory called 'BCSEIS', which contains six subdirectories.
bin/ Doc/ linuxbin/ PerlLib/ scripts/ src/

scripts/ contains the Perl scripts;
src/ contains the source codes (in C++ and Fortran);
PerlLib/ contains two library files used by several Perl scripts;
Doc/ contains documentation files;
bin/ contains executable files for unix systems;
linuxbin/ contains executable files for linux systems***;***

(2) copy directory 'PerlLib/' to the home directory or make a symbolic link to it under the home directory.

(3) go inside 'src/' and compile the codes
In unix system, type 'make -f Makefile.sun' and the executable files are put under 'bin/';
In linux system, type 'make -f Makefile.linux' and the executable files are put under 'linuxbin/'.
Three programs are generated: bcseis, getsachdr and setsachdr

(4) add directory 'scripts/' and either 'bin/' or 'linuxbin/' to the search path.