Instruction Manual for Blacknest Data System

6th draft, September 2013, for v. 2.0.9

Sheila Peacock, AWE Blacknest, Terry Barnaby, BEAM Ltd.

The Blacknest Data System was written under contract to AWE Blacknest by BEAM Ltd.

This manual is supplemental to the manuals produced by BEAM Ltd.

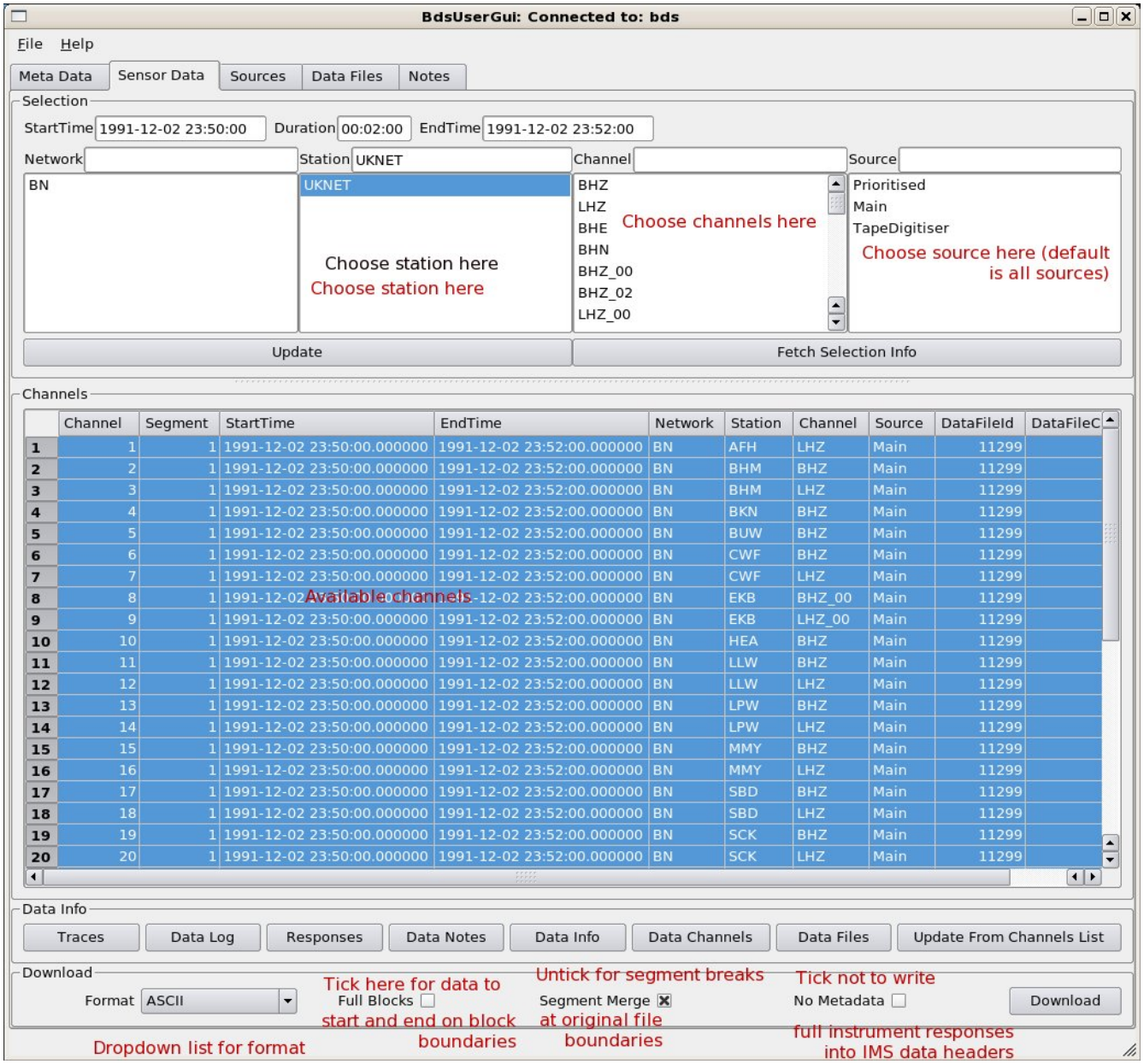
Table of Contents

1 Quick data export

1.1 GUI

The graphical user interface (GUI) is started either from an icon in a desktop menu or by the command “bdsUserGui -host bds”, after which you are prompted for your user name and password. A user with administrator privileges can use “bdsAdminGui”, which for data export is the same as the User GUI. If the environment variable “BDS_HOST” is set to the name of a default host, e.g. “bds”, then the “-host” can be omitted.

Figure 1: DataSelect window of User GUI showing broadband channels from UKNET about to be downloaded. Superimposed notes in red show points where user selection is required before download.



1. Go to “Sensor Data” tab (Figure 1);
2. In top box, set start date/time and duration or start and end dates/times;
3. Choose network (default “BN” includes all data from Blacknest archive) and station or array name (use shift or ctrl+mouse left button to choose a range of stations),
4. Choose channels or leave the upper “Channel” box blank for all available (note that the location code is appended to the channel name after an underscore – the most important ones are for WOL, “00” is the borehole seismometer and “20” is the surface seismometer) ;
5. Choose “source” - for almost all digital data this is “Main”; for most digitised

analogue data it is “TapeDigitiser”, for EKA SP data from the 40-Hz DMOD digitiser it is “EkaDig2”, and for EKA/EKB data from the backup CD-ROMS it is “EkaCD”. If you do not specify a “source” then all available “sources” will be exported, but since the output data formats lack any means to distinguish the “source”, it is sensible to export data from each different “source” separately and edit the source name into the export data filename;

6. Press the “Update” button. The list of channels to be exported, with timespans, is displayed with all the channels selected (in blue). If you want to export only some of these channels, select these with ctrl+mouse left button and shift+left mouse button to mark out a range of channels (unfortunately you cannot in this version of the GUI select them all then un-select a few);
7. When all the channels you want to export are selected (blue), choose the format of the exported data from the drop-down list at the bottom of the frame. IMS-2.0-CM6 (IDC 2004) is the most commonly used format (use this if you want to process the data with SAC). BKNAS (v. 1.0, Blacknest, 2008) will work only for fewer than 32 channels of synchronously sampled data with *no missing blocks and no digital samples exceeding the 6-digit field width.* This effectively restricts it to data of native format BDRS, WRA40, WRA64, WRA_AGSO, BERS and AD_22_YKA – and *not* TapeDigitiser. Usually (and definitely for BKNAS) you want the “Segment Merge” box TICKED and the “Full Blocks” box UNTICKED - see below;
8. Tick or untick the “No Metadata” button depending on what you want to do with the data. For BKNAS-format output data the poles-and-zeros will be written into the header regardless; for IMS-format data the instrument response will be written into the header in IMS format. When the response is in the form of a FAP file, and is repeated for many channels, this makes the data files excessively large, and you might prefer to get the response separately from the data by using the “Meta data” tab;
9. Press the “Download” button. In the File Save dialogue box, choose the directory and, if necessary, modify the default file name for the data file, then press “Save”.
10. The file is then saved, and if the timespan includes missing blocks, or there are other data warnings associated with these data, they will be displayed in a pop-up window after the file has been saved (Figure 2).

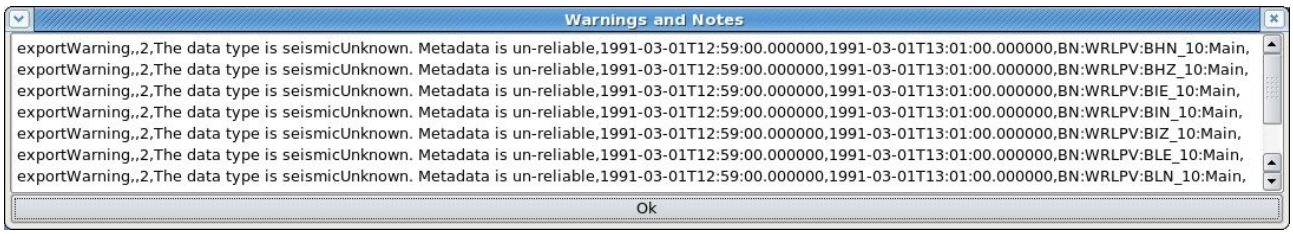


Figure 2: Export warning window showing warning of missing metadata for channels of station WRLPV (for which we lack definitive metadata).

1.2 AutoDRM

Standard messages (IDC 2004) received by the AutoDRM at bdsautodrm@blacknest.gov.uk will cause data to be emailed to you or put into the FTP

directory for your collection (anonymous FTP from bds.blacknest.gov.uk) – the return message will tell you which. A blank message with “help” on the subject line will get you the AutoDRM help file.

A standard message for the export of all EKA array data from 1980-03-01 19:00 to 19:02 is shown. This particular message will cause both digital and digitised analogue data to be returned, in IMS 2.0 CM6 compressed format – there will be 20 channels of digital data and 24 channels of digitised analogue data. The dates may have slashes or dashes as separators.

```
BEGIN GSE2.0
MSG_TYPE REQUEST
MSG_ID EKA tapedig wildcard chan slashdates
E-MAIL sheila@blacknest.gov.uk
TIME 1980/03/01 19:00:00 TO 1980/03/01 19:02:00
STA_LIST EKA
CHAN_LIST *
WAVEFORM IMS
STOP
```

You can ask for “BKNAS” or “SEED” in place of “IMS” in the “WAVEFORM” line (with the same caveat as for the AdminGUI about BKNAS data). SEED data will not be emailed but filed in the FTP directory and an email sent to direct you to the file. If you want uncompressed integer IMS-format data, use “IMS-2.0-INT” in place of “IMS”.

All AutoDRM messages are processed as if “Full Blocks” had not been set (see above).

1.3 Instrument Responses

The AutoDRM serves instrument responses if you replace “WAVEFORM” with “RESPONSE” in the example above. You can use “RESPONSE SAC” to get the seismometer response in SAC pole-zero format. “RESPONSE IMS” returns the response in IMS pole-zero format, not IMS FAP format.

The GUI provides responses for the currently selected channels if you press the “Responses” button at the bottom of the “Sensor Data” tab. The responses appear in a new window in a native format (Figure 3), and may be downloaded in SAC pole-zero, IMS pole-zero, EvalResp FAP or IMS FAP format. When more than one channel is selected, then in either IMS format they will all be saved in one file, but in SAC format they will be saved in a separate file for each channel, and the file save dialogue (Figure 4) will offer the user a generic filename in which the station and channel name are denoted “{station}” and “{channel}” and are permuted for the channels selected in the parent “Sensor Data” tab.

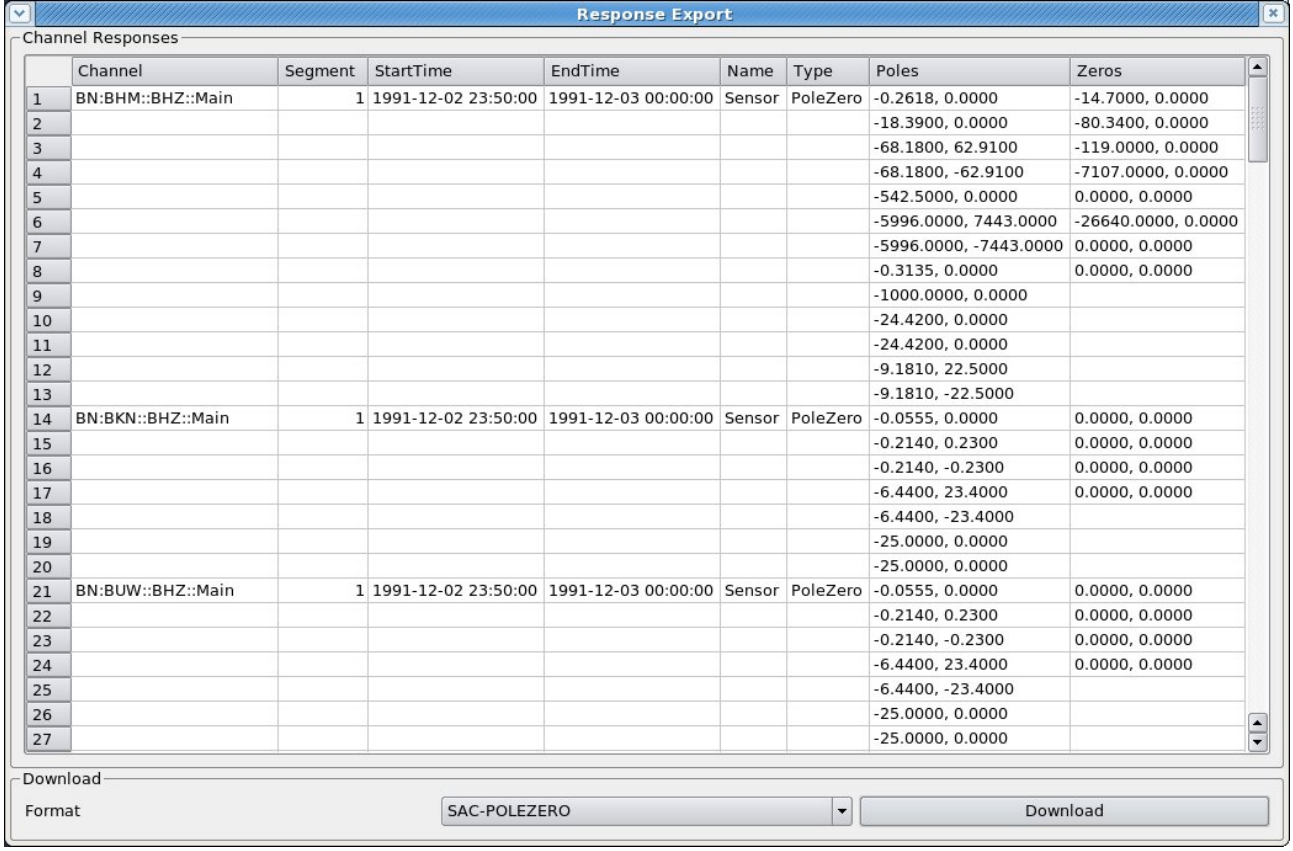


Figure 3: Response Export window showing pole-zero responses for some UKNET stations, ready to download in SAC-pole-zero format.

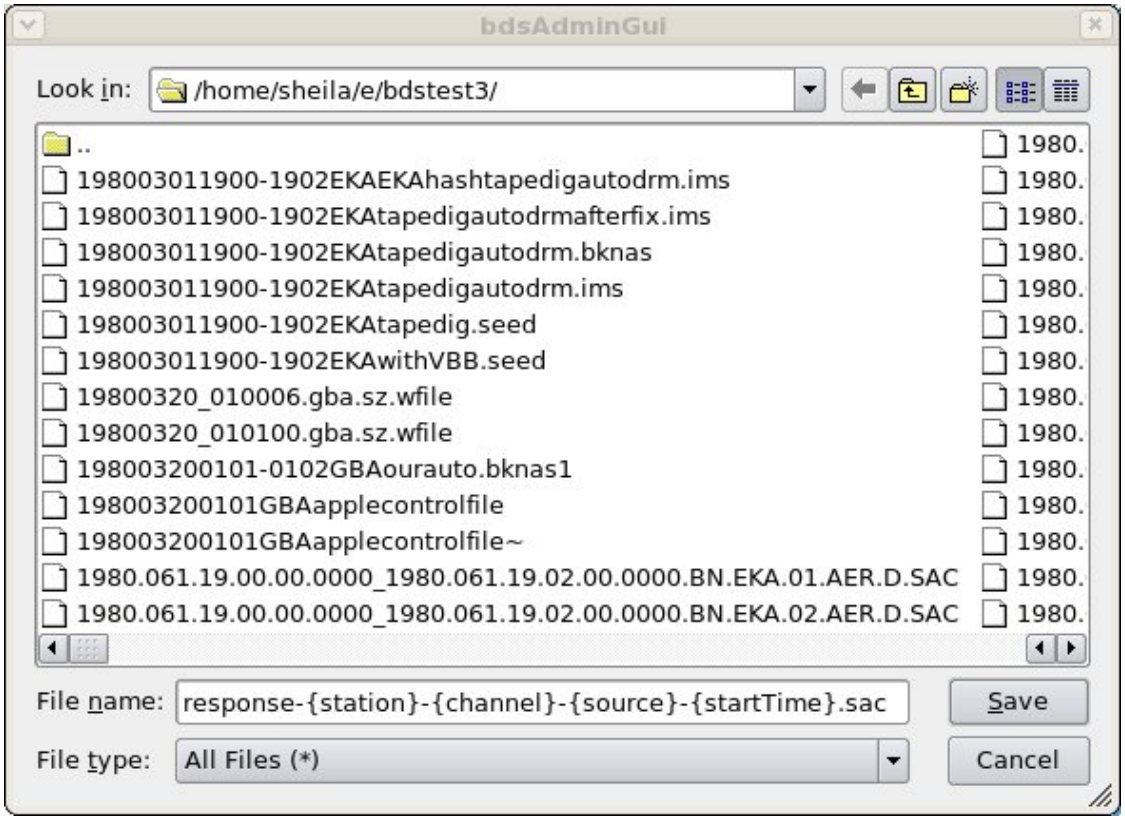


Figure 4: "Save" window for SAC pole-zero response file, showing the default filename template with permutable elements between curly braces. Only the characters outside the braces should be edited to vary the filename from the default.

1.4 Trouble?

In the AdminGUI Sensor Data tab, if no channels are displayed when you press “Update” after entering the correct timespan, station and channels, then possibly there are no data, or none for the “source” that you have selected. Go to the Data Files tab (Figure 5) and enter the day (not the time) from which you wanted data, and the name of one of the stations, and press “Update”. A list of the data files available will appear in the upper pane, and a list of the channels available from those files in the lower pane. Check particularly that the start and end times of the channels you want encompass the timespan you asked for. Data missing because a file started late or ended early are not flagged as “missing blocks”, which refer only to gaps within a file.

If no files at all appear for your day for any of the stations you want, try widening the timespan at the top of the Data Files window (press “Update” again). If files for days on either side exist, then it is probable that data for your day were not recorded, or have been corrupted and could not be imported into the BDS system (particularly older digital data).

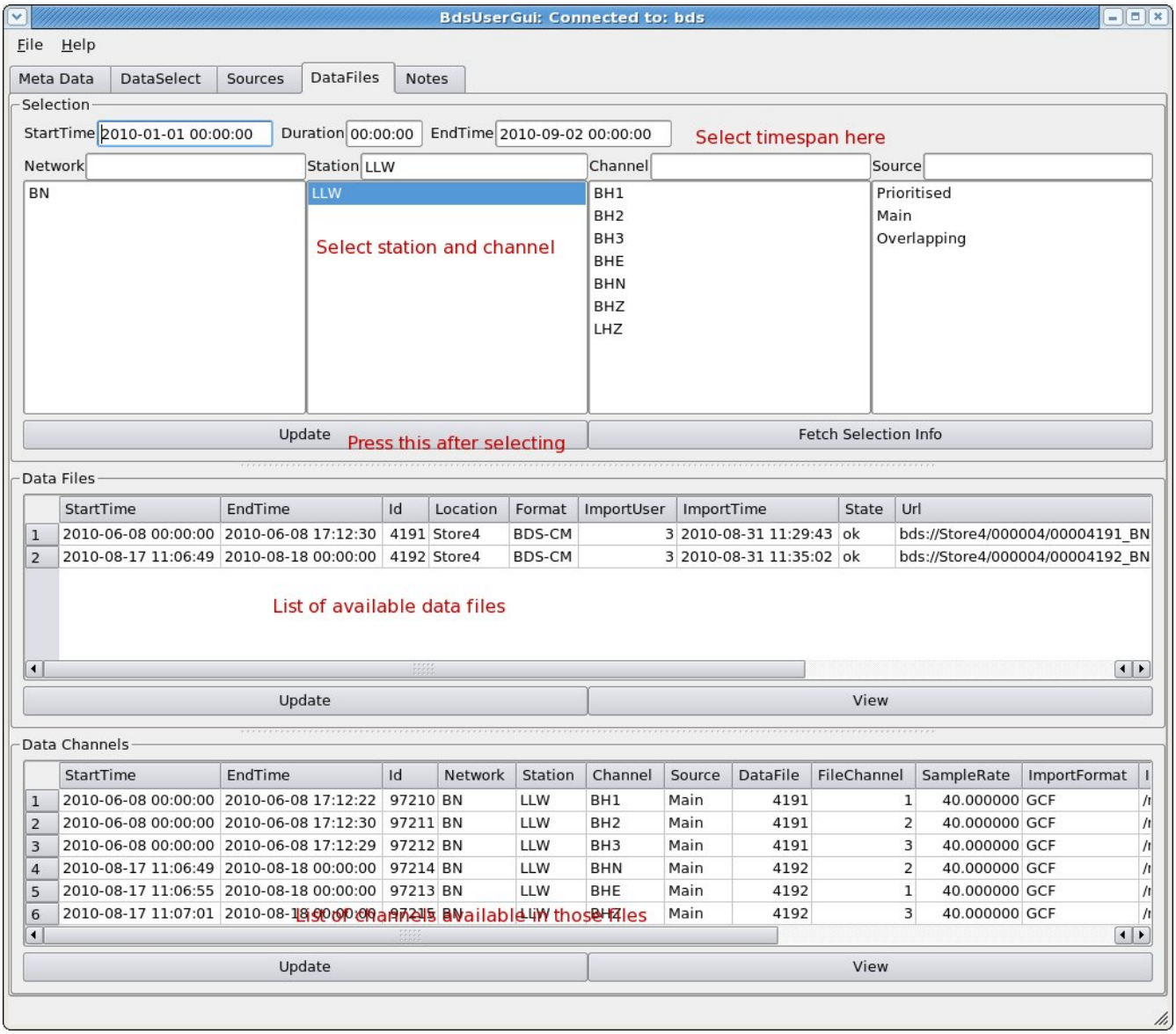


Figure 5: Data Files tab showing list of available files for station LLW for 2010-01-01 to 2010-09-02 (upper subwindow) and channels available in those files (lower subwindow). Red labels are not part of the GUI. NEEDS REPLACING WITH V1.2.11 IMAGE BUT CAN'T UNTIL LLW DATA FOR THIS TIMESPAN ARE LOADED – SP 18/1/2012

If the data channels are displayed but exporting them fails with an error message of “no metadata”, then try again with the channel(s) that lacks metadata de-selected. You can inspect the metadata by going to the “Meta Data” tab and entering the channel name and timespan. Versions from 1.2.8 on should export data from channels labelled “SeismicUnknown”, i.e. channels that we know come from a seismometer but for which we lack the instrument response parameters (e.g. WRLPV, YKC), with dummy parameters. Problems of lack of metadata for commonly-used channels should be referred to the database manager.

1.5 “Full Blocks”, “Segment Merge” and “no metadata” boxes

When the “Full Blocks” box is not ticked, your data should start and end at exactly the times that you specified. When it is ticked, the data start at the nearest block boundary before the requested start time and end at the next block boundary after the requested end time. For multichannel data this can mean that individual channels have different start times or end times. If your desired start or end time falls within a timespan of missing data on any of the channels, the data for that channel will start (or end) at the nearest time for which data are available within your timespan. If you need to use “Full Blocks” AND have all channels start and end at the same time, then this will happen automatically for all original formats except GCF, and can be made to happen for GCF by specifying the start and end times to be on the hour.

The “Segment Merge” box should be ticked to prevent the output from having a segment break (in SEED and IMS formats) at seamless boundaries between adjacent data files in the original data format, e.g. at midnight, unless there are actually data missing between the adjacent files. There is always a segment break i.e. a new header, following timespans of missing data. BKNAS data cannot be served over missing blocks, but will automatically have segments merged over seamless boundaries between original data files, whether or not this box is ticked.

The “No metadata” box should be ticked to prevent the instrument response metadata from being written into the header of out put IMS-format data files. Output SEED and BKNAS files will continue to contain the metadata.

2 Overview of BDS

The principle of the Blacknest Data System (BDS; BEAM, 2010a) is that digital data stored in the BDS data store are combined with station and channel metadata from a database and served to users on request, where the request specifies stations, channels and timespan. The data in the store have been imported from the original archive and are in a common format, BDS version 1.1.0 (BEAM, 2010d). The importing process noted gaps in the data (missing blocks) and re-ordered any data blocks that were out of chronological order as a consequence of backfilling after station-to-base transmission breaks. It did not try to repair other data problems, e.g. header block corruption – these were repaired before import, where possible, and the repairs noted in the imported files. Each stored data file has a substantial header including notes of missing blocks and any corrective processing applied. These and further notes are stored in the database and can be modified.

A primary aim of the BDS project was to make available the data stored on analogue FM tape in the Blacknest archive. These data were digitised in a separate project led by BEAM Ltd. (Bowers, D., Wallis, N. W., Budd, T., and Bartholomew, P., 2008) and now exist in “TapeDigitiser” format, sampled at 100 Hz.

Where the original data were per-sample multiplexed (BDRS, BERS, WRA40, WRA64,

AD_22_YKA and TapeDigitiser data formats) the data in the store are also per-sample multiplexed (format “BDS-SM”) and may be exported in multiplexed BKNAS-1 format, provided there are no missing blocks in the requested timespan. Other multichannel data that have been synchronously sampled but stored in separate files for each channel, i.e. three-component data and broadband EKA array data in GCF format, CD-1.x and CSS-format data and YKA data in SEED format, are stored with all channels combined into a single (hour, 12-hour or day-long) file in channel-multiplexed format (“BDS-CM”), and may be served in BKNAS-1 format provided they were imported with the “-synchronous” option to indicate that they were synchronously sampled. The non-multiplexed SEED and IMS formats may be used to serve all types of data. BKNI infrasound data from the experimental site at Blacknest, for which the channels are not synchronously sampled, are also stored with all four channels in a single file per day or hour, in channel-multiplexed form. These cannot be exported in BKNAS-1 format.

2.1 Database

The database is the source of all information in the headers of exported data from the BDS, except times. It was populated from the database used by the 2006 AutoDRM and instrument response PHP program, which is derived from information in file “array_changes.dat” used by the “robby” AutoDRM and is up to date to September 1997, and manufacturers' calibration sheets and operators' reports for instruments introduced since then. Station locations are taken from the IDC database where possible, otherwise from our own records.

Some non-intuitive points about the database:

(a) The concept of a “channel”, and hence the channel's start and end time, is separate from the concept of the sensors (seismometers, infrasound sensors, etc.) and digitisers attached to the channel, and these again are separate from both the calibration factor and frequency response of the channel. A channel might have start time at time zero and end time at infinity, but it is a valid “seismic” channel only for the timespans for which a calibration exists, and then only providing that instrument, sensor and response also exist for that time. Hence it is possible for a station to have, say, a BH2 and a BHN channel with overlapping timespans, representing output from the same seismometer at different orientations, provided there is a distinct calibration entry for each, with non-overlapping timespans.

(b) Channel order of data in multiplexed original format files stored in the 2006 AutoDRM database has not been imported into the BDS database, so the channel order has to be specified at import for all data except SEED and CD-1.x format data, for which the channel order is extracted from the input file header;

(c) For those stations with location codes, the BDS stores these along with the channel code in format CCC_LL, e.g. BHZ_10. In most circumstances it is essential to specify this “full” channel name to access your data. The AutoDRM does not recognise the “_LL” or any other means of specifying a location code, and will expand a channel code to include all associated location codes;

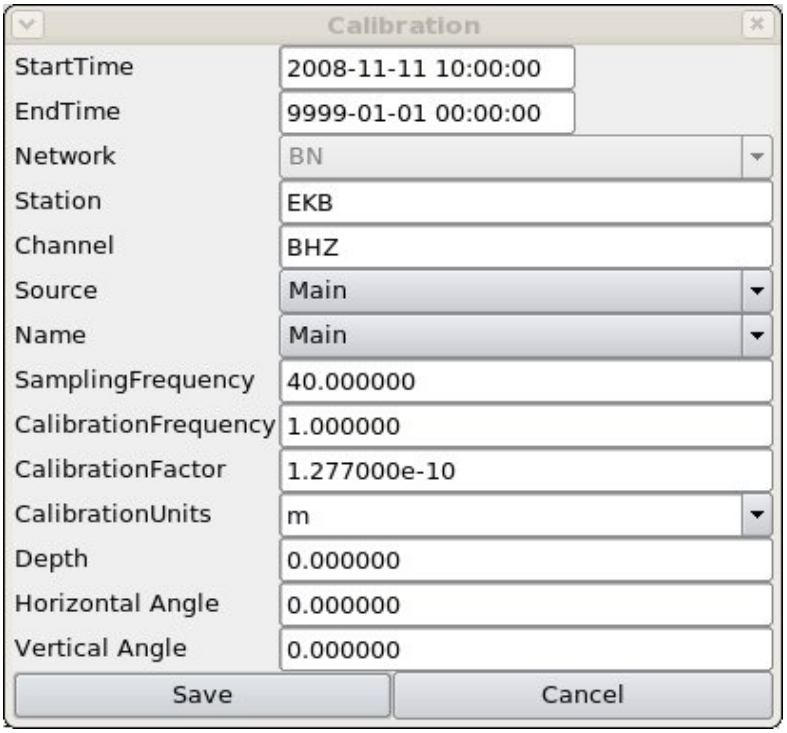
(d) Times are all assumed to be UTC and are stored as character strings representing the time in ISO 8601 format, not “DateTime” entities, because of the lack of support

for fractions of a second in MySQL “DateTime” entities. The string format is “YYYY-MM-DDTHH:MM:SS.FFFFFF”, e.g. “2009-07-09T00:07:59.484595” and all times input on command lines of BDS programs have to be in this format. The time part (the uppercase “T” and everything after it) can be omitted, and it is not necessary to specify fractions of a second (i.e. the “.FFFFFF” part) if these are zero;

(e) Searches for station and channel names are case-sensitive, so “eka” is not the same as “EKA”.

2.2 Calibrations

Figure 6: Calibration edit window for station EKB, channel BHZ (i.e. the 3-component broadband set in the vault at EKA, now taken to be part of AS104 EKA by the IDC), showing “Name” field (“Main” here because data are taken from manufacturer's data sheet) and Calibration Factor in metres per digital count.



The database contains instrument information for “sensors” (mostly seismometers, but also infrasound, hydroacoustic and other sensors) and digitisers. A digitiser is associated with a serial number, sampling rate and gain, all optional. A seismometer is associated with a serial number, number of channels and a gain, also all optional. Stored separately are a “response” (a set of poles and zeros or a FAP table) and one or more timestamped “Calibrations” (Figure 6) consisting of a calibration factor, corresponding calibration frequency, and sampling frequency. These are associated with the channel *not* the sensor. All units for these in the underlying database are basic SI, e.g. metres per count for calibration factors of seismometers, but on output they are converted to the units required by the output format, e.g. nanometres per count for IMS format.

The calibrations are labelled with “Name” as “Main” if the data are from the manufacturer's data sheet or theoretical calculation, or “Measured” if they are from field measurements. Units of calibrations are stored, and in general are in the displacement domain (pressure for infrasound and hydroacoustic sensors) because this was assumed for calibrations in the 2006 AutoDRM database.

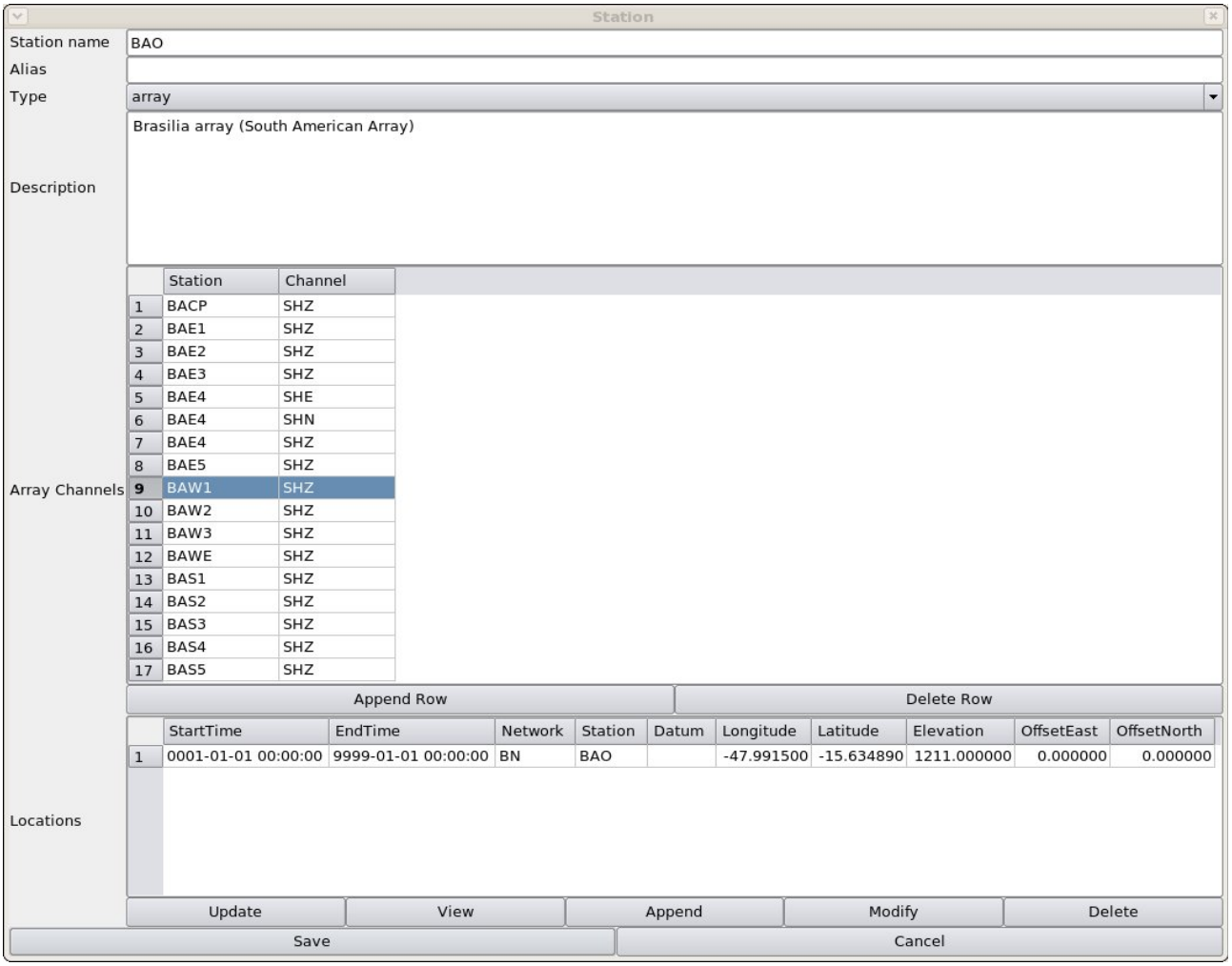


Figure 7: Station Edit window showing array station BAO (Brasilia). Note that the array is specified as a list of channels, not stations, and only the centre-point location is given in the lower pane. East and north offsets are in km, elevations in m

2.3 Stations and Arrays

Figure 8: Station Edit window showing array station BAE (Brasilia). Note that the array is specified as a list of channels, not stations, and only the centre-point location is given in the lower pane. East and north offsets are in km, elevations in m.

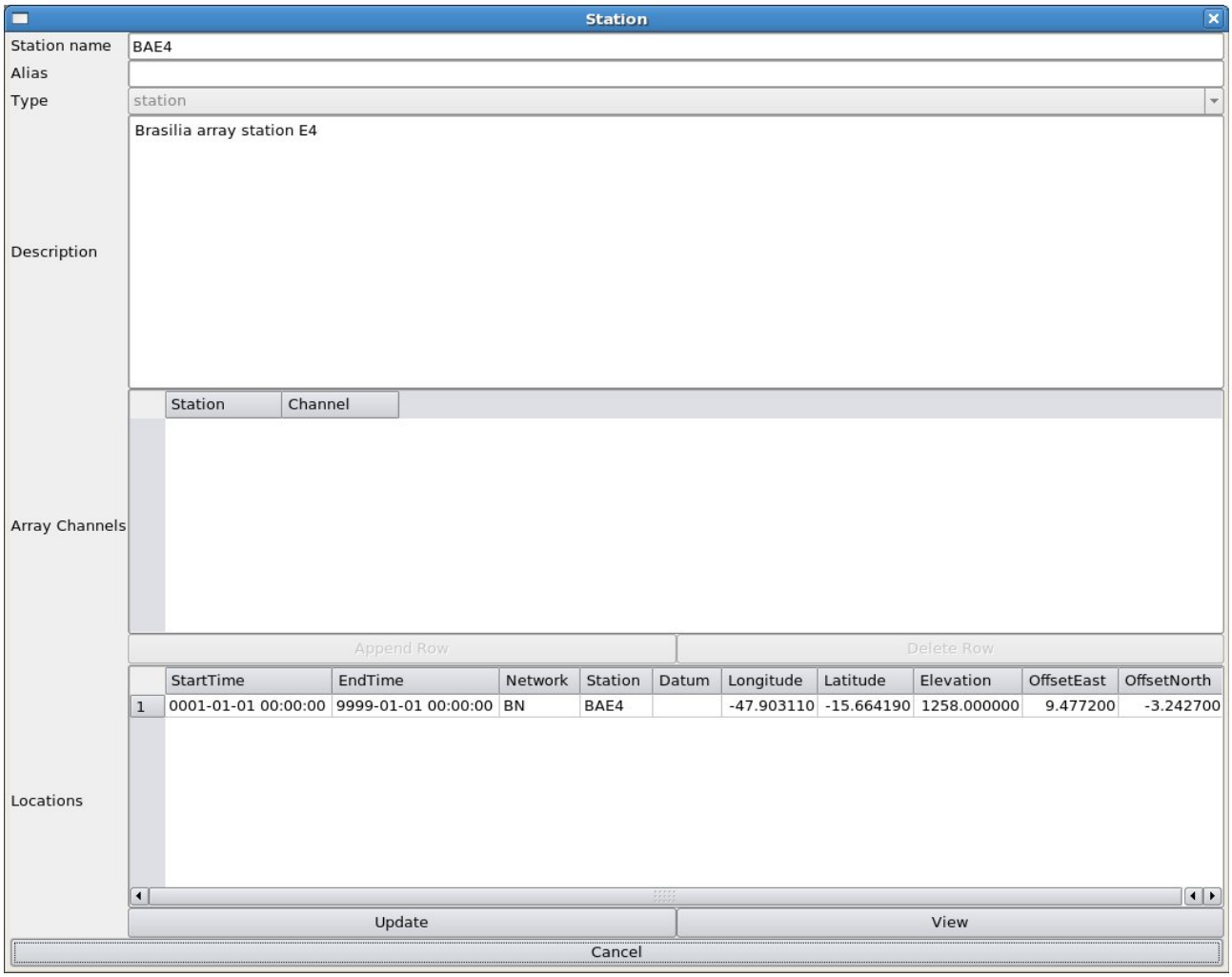


Figure 9: Station Edit window showing station E4 of Brasilia array BAO. The elevation is in metres, the east and north offsets from the array centre are in km. Note that the BDS does not check the consistency of lats/longs with array offsets.

The list of stations in the database includes the latitude, longitude and elevation of each, and for array stations, the x- and y-offsets from the array centre point. The database does not correct inconsistencies between latitudes/longitudes and x/y offsets.

An array is specified as a list of *channels* (Figure 7). The array centre point is stored as if it were a single station, although none of the output data formats has a provision for showing the array centre position if that is not occupied by one of the array stations. The individual stations have to be viewed separately from the array to see their latitudes, longitudes and offsets (Figure 9). An array can be named in a data request, in which case data from all the channels listed in the array will be supplied if available. While any station can be listed as part of an array, the policy has been to list all the channels that were recorded on the 24-track analogue tapes from each of the four “UKAEA” arrays, including the time-code and “error” and any supplemental seismic channels such as the “velocity

broadband" seismometer (actually at the EKB site) routinely recorded alongside the EKA array but not included in array processing. The BDS does not do any array processing (e.g. beamforming).

2.4 Duplicate Data

Data in the Blacknest archive come from a number of sources and there are consequently some duplications. These are faithfully stored in the BDS data store. Some, but not all, the duplicate data streams are distinguished by a value of "source" in the BDS database. Where there is no distinction of "source", or the user request does not specify the "source", e.g. an AutoDRM request, data from all available "sources" are served. This can cause problems, because the IMS and BKNAS output formats lack any means of distinguishing the "source" of data. Also, if array data from more than one "source" are available, none of the data can be served in BKNAS format, because the number of channels is (at least) doubled, exceeding the 31-channel maximum for that format.

Duplications are

(a) Those for which no separate "source" is specified

i. TapeDigitiser data where digitising was stopped and re-started after rewinding (partially or fully) the tape. The overlapping segments are not identical because of slight differences in tape speed and tape-head cleanliness between passes;

ii. Digital data in BDRS format that were transcribed from 1/2-inch reel tape to Exabyte tape in the mid-1990s. Overlapping data occur where the 1/2-inch tape was replayed. They should be identical but differences in tape-head cleanliness sometimes caused different bit-level errors to occur in the two overlapping files.

(b) Those for which a separate "source" is specified

i. EKA SP data from FMDC (20 Hz) and DMOD (40 Hz, but poorer quality) (Bowers 2004) digitisers – the latter have "source" value "EkaDig2";

ii. EKA broadband and EKB data arriving by satellite link and by DVD posted from EKA – these are identical except that the satellite stream is prone to dropouts (missing blocks), whereas the DVD data occasionally have whole files missing because of problems with the computer at EKA on which they are stored before being written on to the DVD. Data from the DVDs have "source" name "EkaCD" and "EkbCD" respectively, and are identical to the satellite data except at the gaps. THIS MIGHT CHANGE WHEN A BULK IMPORTER FOR THE CD DATA IS WRITTEN.

During import and export, "source" is used to relate data to metadata. Data form a particular "source" are associated with metadata for the "source" name given by the value of "sourceMeta" associated with that "source" (Figure 10). For the EKA SP data, the metadata for the DMOD are different from those for the FMDC because the DMOD has a different sampling rate, so "sourceMeta" for data of "source" = "EkaDig2" is also "EkaDig2". For EKA broadband data from satellite and DVD the metadata are the same, so the satellite data have "source" "Main" and the DVD data have "source" "EkaCD", but both have "SourceMeta" value "Main".

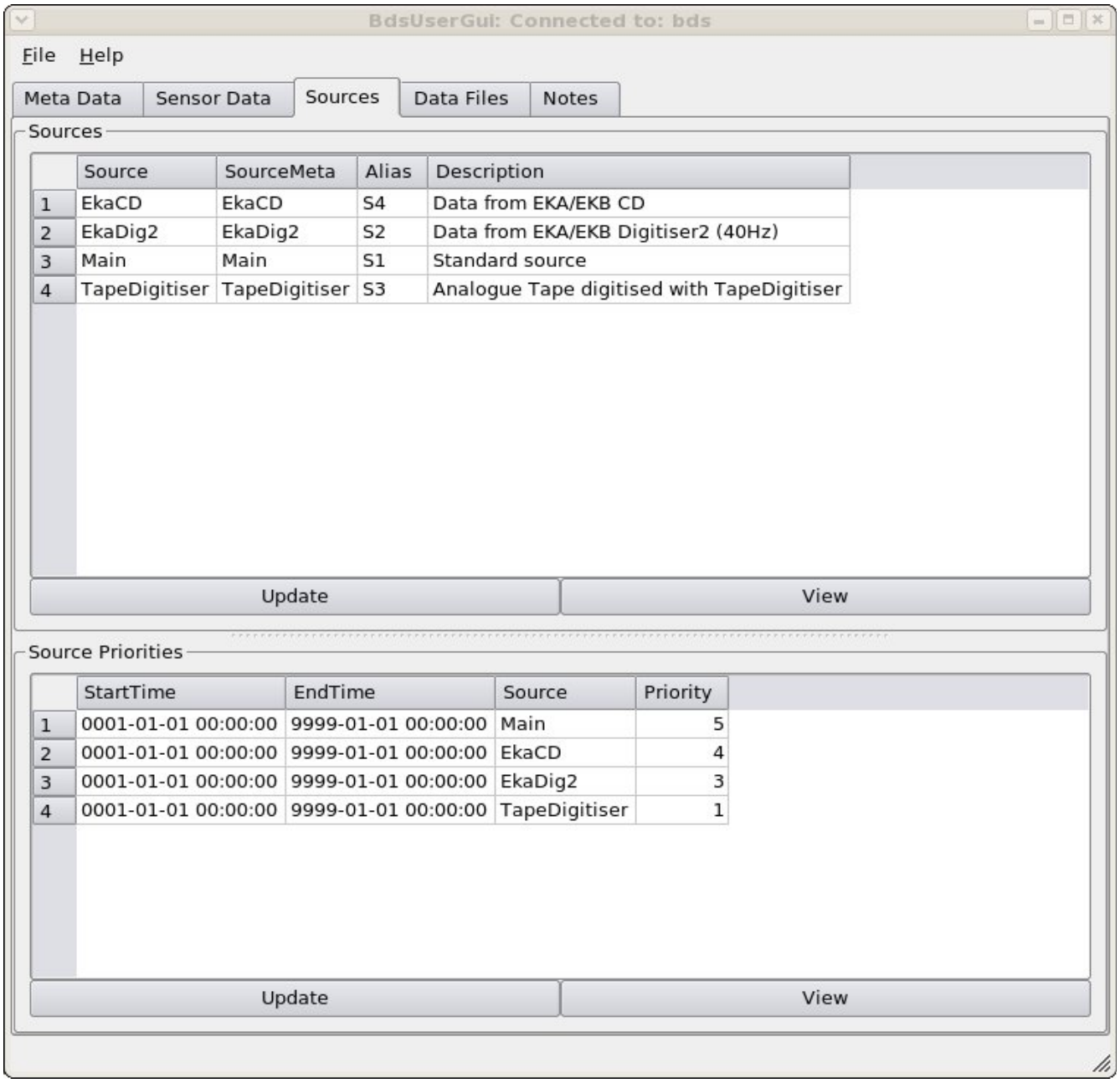
Where there are data from more than one “source”, it is possible to use the “Source Priorities” table (Figure 5) to assign one source higher priority than another. This means that data from the preferred source will be served, wherever data are available, when the user does not express a desired “source”. In the Data Select window (Figure 1), choosing the “source” value “Prioritised” causes only data for the highest priority source to be exported.

2.5 Time codes on digitised data

All the data digitised from analogue tape in the TapeDigitiser project have “source” value “TapeDigitiser”. Time stamps for the TapeDigitiser data are derived from the time code (mostly in VELA format, though some tapes have time codes in Hutchins format, Key, F., 1968; Holdsworth, B, 1969), which was read by a program that required two complete minutes of flawless time code to read a valid time. For tapes where the time code was corrupt, times for data block timestamps are interpolated between times successfully read by the decoder. Users are advised not to rely on the interpolated times but read the time code by eye.

The only other data digitised from analogue tape in the BDS data store are five months of YKA data from January to May 1989, digitised on Blacknest equipment nominally at 20 Hz. Block timestamps for these are estimated from tape play times, so the user should always read the VELA code.

Figure 10: "Sources" tab showing association of "SourceMeta" with "source" (upper frame) and list of priorities for sources (lower frame).



3 Data Export

Data may be exported by (a) the graphical user interface (AdminGui or UserGui), (b) the Web interface, (c) the AutoDRM, or (d) the command line program bdsDataAccess.

3.1 Graphical User Interface

Basic instructions were given in Chapter 1. Data export is carried out from the “Sensor Data” tab (Figure 1).

1. *Date-time entry* - The start time and end time or time duration must be specified in the date/time box at the top of the page. The date/time interface is based on a text editor and is easiest to use if you place the cursor at the beginning of the box and type the number of a valid date, e.g. type “19870331154530”, without separators, to get the date “1987-03-31 15:45:30”. If you make a typing error just press the backspace button and retype (it will overtype the error). To alter an individual digit, place the cursor to the left of the digit (use cursor keys or mouse) and type the new value. The backspace acts as a backwards cursor, as in the “Vi” editor. The mouse drag “select” is ineffective – it merely positions the cursor.

   Entering an invalid date causes the box to turn pink. This might happen while you are overtyping a date, since it tests the validity after every keystroke. Keep typing until you have entered the full date, at which the box should revert to white (unless you made a typing error). Occasionally the date appears in an invalid format, e.g. a year with five digits. In this case, place the cursor at the beginning and type the date in the correct format, WITH the dashes.

   The tab key moves the cursor to the next box. You may type a duration, or type an end time into the third box. If the end time is more than 24 hours after the start time, you should type the end time into the third box, and the duration box will show zero. The maximum data length that can be exported is given by the parameter “MaxTimePeriod” in the file /etc/bdsServer.conf on the server computer. During testing it has been at 86460 s (one day and one minute).

2. *Specifying network, station, channel and “source”* - The boxes for specifying network, station, channel and “source” are below the time entry box. There are two boxes for each, the one-line upper box, into which you can type the name you want or a partial (e.g. “BH*”) or full (i.e. “*”) wildcard; and the lower box, in which the list of stations or channels matching your selection is displayed. The upper box will take regular expressions (see info pages for “grep”, “awk” or “sed” Un*x commands). In particular, the only way to specify two stations (because a comma-separated list doesn't work) is to use the “OR” notation (|), e.g. to get WOL and BKN enter WOL|BKN but beware that this will get you BKNI1-4 and WOL1-7 and WOL#. When the upper box is blank, the lower box displays all possible values. You can select one value or a range of values with the left mouse (followed by ctrl+left mouse to highlight more than one entry, or shift+left mouse to highlight the range of entries between the nearest highlighted entry and the entry at the mouse cursor).

   The “network” box offers “BN”, which is the Blacknest network, and “IN”, which will be the International Monitoring System data in a future release.

   In the “station” box, choosing an array such as EKA will get you the list of channels associated with that array. For digitised analogue data these include the “error” and time-code channels and any additional seismometer channels recorded along with the array. If you don't want all the channels listed, select the ones you do want with

the mouse and shift and Ctrl keys. For EKA digital data after November 2008, where both broadband and short-period array data exist, it is sensible to separate them, so choose channel “SHZ” or “BHZ” as desired. The list of channels associated with an array can be viewed in the “Stations” tab (Figure 8), but it is better to keep this list as inclusive as possible and select channels to be exported case-by-case.

In the “source” box the default is no source specified. If data from more than one “source” exist for your station and timespan, it is sensible to save them in a different file for each “source” because there is no means to distinguish data segments from different sources in the output file formats. For BKNAS, furthermore, there is a 31-channel limit, so it is not possible to export data from an 18 or 20-channel array from more than one “source” in that format.

3. *Press the “update” button* - If you do not press the “update” button after choosing your timespan, stations and channels, the previously chosen timespan, stations and channels are retained in the lower pane, and when you press the “download” button an error message is likely to be displayed because of the confusion between the newly selected timespan and the previous one (the error message might be “Error: data does not overlap across all channels for data time clipping”, or “Internal error: no actual data segments are available from file although database says there should be some”).

4. *Choosing the channels* - After pressing “update”, check the list of channels displayed in the lower pane (Figure 1). By default they are all selected to be downloaded (blue). If you do not want them all, select the first one you want by clicking on it with the mouse (which de-selects all the others) then press the shift or Ctrl key while clicking with the mouse to select the others that you want. It is not at present possible to de-select just one or two. THIS MIGHT CHANGE IN QT4 If you want to export log files, select only “LOG” channels from the list, or choose the channel to be “LOG” before pressing “update”. Don't try to export log and seismic channels together.

5. *Select the format and press “download”*

   Available formats:

   1. ASCII – data in single-column ASCII, channel by channel, with simple header. Digital sample values have been multiplied by “calib”. Default filename tag “.txt”
   2. ASCII-SM – data in ASCII with no. of columns = no. of channels (e.g. 20 for EKA) and header. Digital sample values have been multiplied by “calib”. Default filename tag “.txt”
   3. ASCII-CM – same as “ASCII”
   4. BKNAS, BKNAS-1.0 – data in BKNAS-1.0 format. Default filename tag “.bknas”
   5. BDS – disabled – for test only
   6. BDS-SM - ditto

7. BDS-CM - ditto
8. IMS, IMS-2.0, IMS-2.0-CM6 – data in IMS 2.0 format with CM6 compression. Default tag “.ims”
9. IMS-2.0-INT – data in IMS 2.0 format as integers without compression. Default tag “.ims”
10. SEED – data in IRIS SEED format. Default filename tag “.seed”
11. SEED-MINI – data in IRIS mini-SEED format. Default filename tag “.seeddata”
12. SEED_METADATA – metadata only in SEED format – to be used along with miniSEED file. Default filename tag “.seedmetadata”.
13. LOG and LOG_SCREAM – these are the same and are for exporting the log files written by Güralp digitisers. Default filename tag “.txt”

The default filename for the downloaded data has the prefix “data-STN1-STN2-YYYYMMDD-HHMMSS” and the tag given above denoting the format. “STN1” and “STN2” are the first and last, alphabetically, stations from which data are downloaded, e.g. “EKB1-EKR9”. For a single channel, only “STN1” appears. When the station name includes a “#” (in dummy station names for timecode, error and disconnected channels in array data), the “#” is replaced by a “~” to avoid Unix shell filename problems. “ YYYYMMDD-HHMMSS” is the start time, e.g. 19810315-145200 for 1981-03-15 14:52:00. You can modify the filename and select the directory into which the file will be saved. When you press “save”, if the file already exists, a query whether to overwrite it will appear.

The actual saving might take some time. Once it is complete, a popup box shows any warnings that are attached to the data (including those attached at import by the user invoking the “-addWarning” option of bdsImportData). For poor data there might be a lot of warnings, usually of missing or re-ordered blocks. If the channels that you have chosen include any of type “seismicUnknown” you will receive warnings that the metadata for these are unreliable. This does not prevent their download, but might affect subsequent processing programs, e.g. rdseed when it tries to extract responses or poles-and-zeros from these channels.

Log channel exports are time-limited by the times set in the boxes at the top of the tab, so only entries with timestamps within those limits are exported. Log entries before and after your time limits might be relevant to your exported seismic data, and you will need to widen the time limits to view or export these. If your timespan covers more than one log file and you “select” them all before exporting the logs, all entries in all the files selected that have timestamps within the requested timespan will be exported in a single output file.

3.1.1 Using Data View window

After Item 3 above, if you press the “Traces” button, the “Data View” window appears, showing parts of some of the traces you have selected from the “Channels” list. All of the traces you have selected can be seen by working the scroll sliders. Further selection of the data to be exported can be done:

1. individual traces can be selected or de-selected by ticking or un-ticking the “Select” box at the right of each trace (Figure 11);
2. A timespan can be selected by moving the cursor into position and pressing the left

mouse button, then moving it again and pressing the right mouse button – the cursor positions will appear in the “Start Cursor” and “End Cursor” slots below the trace display;

3. While the “Data View” window is open and traces and cursors are set, if you press the “download” button, then only the selected timespan from the selected traces will be exported (and the default export filename will be set accordingly with the cursor start time).

Other features on the Data View window are:

1. “pixel = sample” button, which increases magnification until each screen pixel represents one digital sample. This allows the trace to be displayed without aliasing;
2. X in, X out buttons – change time scale of display;
3. Y in, Y out buttons – change amplitude of display – only on selected traces;
4. Y up, Y down – change Y shift (DC shift) of traces – only on selected traces.

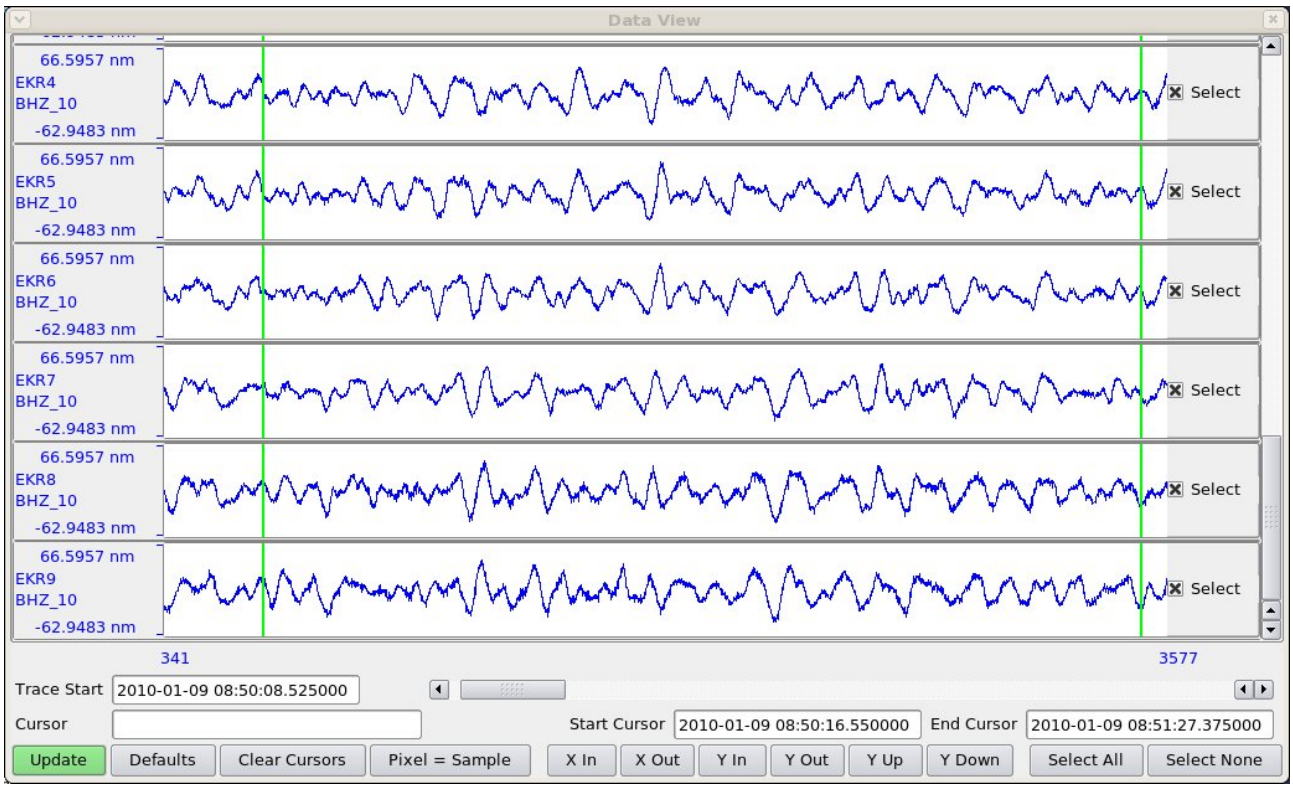


Figure 11: Data View window showing a few traces from EKA with start and end cursors set (vertical green lines) and all traces selected (boxes to right of traces ticked).

3.2 Web interface

Yet to be written.

3.3 AutoDRM

Instructions for the AutoDRM were given in the first section. Besides data in SEED format, data that cause the email to exceed about 1 Mb total length (the exact value is stored in the parameter file /etc/bdsAutodrm.conf on the BDS server as “EmailMaxLength”) are not emailed but stored in a temporary FTP directory to be obtained by anonymous FTP. The details of the FTP site and file are emailed in IMS format, for instance:

```
BEGIN GSE2.0
MSG_TYPE DATA
MSG_ID BDS_2010-11-08_11:48:57
REF_ID EKA tapedig
DATA_TYPE FTP_LOG
 FTP_FILE bds bdsdata /pub/bds bdsAutodrm-kUHwUz.gz

 Bds AutoDRM Version 1.2.10 Nov  4 2010

 This autodrm imposes a limit of 1048576 bytes on the
 quantity of data that can be returned via e-mail
 Also it can only return ACCII formated data within an email

 Your request has exceeded the size limit or is binary data.

 If it is ASCII data, to get the data by e-mail, try requesting
 smaller time segments.
 The data will be stored on our server for 3 days
 before being removed automatically. During this time
 you should copy the data from our anonymous FTP server
 using the information in the above FTP_FILE line.
 Refer to the User Guide for more information on this.

 Warnings
 doc,testAdd,0,EKAlogsheets1977-1990,1977-03-23T00:00:00.000000,1990-09-08T0
 0:00:00.000000,BN:EKA:SHZ:TapeDigitiser,tape logs
STOP
```

The FTP instructions follow the keyword “FTP_FILE” in the format “*net-address login-mode directory file*” (IDC 2004 p. 185). Warnings accompanying the data are also emailed with the instructions. The data file is compressed with “gzip” (GNU version of “zip”) and should be uncompressed with “gunzip” before use. On a Windows system a program such as ZipGenius can be used.

3.4 bdsDataAccess command-line program

The BEAM manual (BEAM, 2010c) describes the options for this program. To export data the option “-command dataGetFormatted” is used. The “-select” command must be repeated for each channel to be exported; but note that the channels are exported in alphabetical order, not in the order in which you specify them. “-format IMS” causes the output to be in IMS CM6 format. “-fullBlocks” has the same effect as ticking the “full blocks” box in the AdminGui “Sensor Data” tab (see above). The bdsDataAccess program defaults to merging segments, i.e. as if you had ticked the “merge segments” box in the AdminGui (see above).

A sample command for exporting is:

```
bdsDataAccess -host bds -user sheila:<password> \
-startTime 2010-01-27T03:58:00 -endTime 2010-01-27T03:59:00 \
-select BN:EKR1:BHZ:Main \
-select BN:EKR2:BHZ:Main \
-select BN:EKR3:BHZ:Main \
-select BN:EKR4:BHZ:Main \
-select BN:EKR5:BHZ:Main \
-select BN:EKR6:BHZ:Main \
-select BN:EKR7:BHZ:Main \
-select BN:EKR8:BHZ:Main \
-select BN:EKR9:BHZ:Main \
-select BN:EKR10:BHZ:Main \
-select BN:EKB1:BHZ:Main \
-select BN:EKB2:BHZ:Main \
-select BN:EKB3:BHZ:Main \
-select BN:EKB4:BHZ:Main \
-select BN:EKB5:BHZ:Main \
-select BN:EKB6:BHZ:Main \
-select BN:EKB7:BHZ:Main \
-select BN:EKB8:BHZ:Main \
-select BN:EKB9:BHZ:Main \
-select BN:EKB10:BHZ:Main \
-command dataGetFormatted -format IMS -o 201001270359-0000EKABBv.1.2.6.ims
```

The “-select” option is capable of taking wildcards in a format similar to a Unix “regular expression” (regexp). It has four parts, Network:Station:Channel:Source, and if any is omitted it will respond providing all the possible values, using the three colons as placeholders (so “:::” would be all networks, stations and channels – not a good idea). “*” has its usual non-regexp wildcard meaning of “all values”. A selection of values can be enclosed in square brackets, e.g. “EK[RB]” meaning “EKR or EKB”.

The “host” and “user” may be omitted if “BDS_HOST” is specified and the user's name on the BDS system is the same as their login name.

When TapeDigitiser data are exported, the warning “TapeDigitiser Digitisation quality warnings/errors list in infoExtra” is always displayed. This information can be accessed by the option “-command dataInfoFull” on the command bdsDataAccess – see manual (BEAM, 2010c) for full description. A sample command is

```
bdsDataAccess -host bds -user sheila:<password> -startTime 1978-08-17T23:50:00
-endTime 1978-08-17T23:52:00 -select BN:EKR1:SHZ:TapeDigitiser -command
dataInfoFull
```

The option “-command dataPlot” produces a png-format plot of the first data block of waveforms in a file specified by the “-o filename” option (Figure 12).

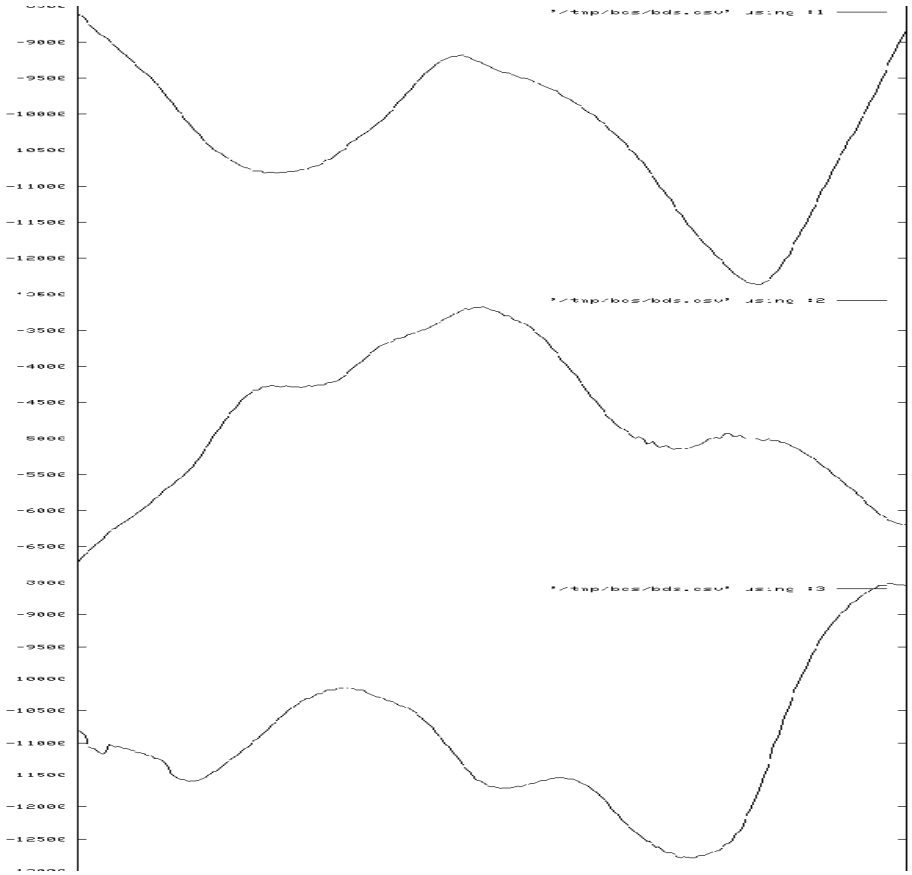


Figure 12: Plot, from “png”-format raster image, of first block of three-component data from FLK starting at 03:00 on 2008-07-01, obtained from "-dataPlot" option of bdsDataAccess program. The plot has been stretched widthways by image-processing software before inclusion in this manual. The traces are, from top to bottom, BHE, BHN and BHZ channels.

The options “-command listNetworks”, “-command listFormats”, “-command listStations” and “-command statistics” need no other parameters than the host and user names, e.g. “bdsDataAccess -host bds -user sheila:<password> -command listStations ”. If your parameter list does include stations specified with “-select” then “listStations” lists only those stations.

The options “-command dataSearch”, “-command dataInfo”, “-command dataInfoFull”, “-command channelInfo”, “-command channelInfoFull” and “-command responses” require a timespan to be defined with “-startTime”/”-endTime”, and one or more channels with “-select”. See below for more about obtaining instrument responses with bdsDataAccess. “channelInfo” provides information on channels with the following headings:
#Chan (the “#” indicates a header line, not an abbreviation of “number”),
Segment,
StartTime,
EndTime,
Network,
Station,
Channel,
Longitude,

Latitude,
Elevation,
Depth,
Source,
Sensor,
SampleRate,
CalibFactor (calibration factor)

“channelInfoFull” gives the same information (but not “Source”) and additionally:
SensorId (database ID number of sensor),
SensorType,
SensorSerial,
DigitiserId (database ID number of digitiser),
Digitiser,
DigitiserType,
DigitiserSerial,
SensorHAng (horizontal angle),
SensorVAng (vertical angle),
CalFreq (calibration frequency),
ResponseId (database ID number of response information for sensor)

“dataInfo” and “dataInfoFull” provide output listing:

- warnings associated with the data (the examples here are warnings attached by the importing user with the option “-addWarnings” in the command “bdsImportData”, but warnings generated by “bdsImportData” itself, e.g. about missing blocks, are also shown);
- the original filenames from which the data were imported into the BDS;
- sample rate, start and end time of each channel;
- for TapeDigitiser data, information from the jobInfo file written during digitisation, including an overall quality measure of the data;
- for TapeDigitiser data, “dataInfoFull” also lists errors in time-code decoding and deviations in the FM carrier frequency on the analogue tape, detected during digitisation.

An example of output from “dataInfo” is:

```
Warning,                    DummyWarning,/
remote/london/asm/blacknest/EKA/BHZ/2010/2010027_EKB1BHZ.gcf,,599,right-file-
untimed-dummy-error-message
Warning,                    DummyWarning,/
remote/london/asm/blacknest/EKA/BHZ/2010/2010028_EKB1BHZ.gcf,,499,wrong-file-
untimed-dummy-error-message
startTime,                  2010-01-27T03:57:36.000000
endTime,                    2010-01-27T03:59:21.000000
array,                      EKA
description,
channels.number,            20
channels.synchronous,       1
importWarning0,             DummyWarning,/
remote/london/asm/blacknest/EKA/BHZ/2010/2010027_EKB1BHZ.gcf,,599,right-file-
untimed-dummy-error-message
importWarning1,             DummyWarning,/
remote/london/asm/blacknest/EKA/BHZ/2010/2010028_EKB1BHZ.gcf,,499,wrong-file-
untimed-dummy-error-message
importWarning2,             DummyWarning,/
```

```
remote/london/asm/blacknest/EKA/BHZ/2010/2010028_EKB1BHZ.gcf,2010-01-
27T23:59:30,699, wrong-file-right-time-dummy-error-message
importFile0.fileName,
/remote/london/asm/blacknest/EKA/BHZ/2010/EKR4/2010027_EKR4BHZ.gcf
importFile1.fileName,
/remote/london/asm/blacknest/EKA/BHZ/2010/EKR5/2010027_EKR5BHZ.gcf
```
… (list of all 20 imported file names for EKA BB followed here)
```
bds.version,                1.1.0
bds.format,                 BDS-CM
channel1.startTime,         2010-01-27T03:57:40.000000
channel1.endTime,           2010-01-27T03:59:20.000000
channel1.network,           BN
channel1.station,           EKB1
channel1.channel,           BHZ_10
channel1.source,            Main
channel1.numBlocks,         4
channel1.numSamples,        4000
channel1.sampleRate,        40.000000
channel1.sampleFormat,      2
channel1.importFormat,      GCF
channel1.importFilename,
/remote/london/asm/blacknest/EKA/BHZ/2010/EKB1/2010027_EKB1BHZ.gcf
channel1.importStartTime,  2010-01-27T00:00:00.000000
channel2.startTime,         2010-01-27T03:57:51.000000
channel2.endTime,           2010-01-27T03:59:06.000000
channel2.network,           BN
channel2.station,           EKB10
channel2.channel,           BHZ_10
channel2.source,            Main
channel2.numBlocks,         4
channel2.numSamples,        3000
channel2.sampleRate,        40.000000
channel2.sampleFormat,      2
channel2.importFormat,      GCF
channel2.importFilename,
/remote/london/asm/blacknest/EKA/BHZ/2010/EKB10/2010027_EKB10BHZ.gcf
channel2.importStartTime,  2010-01-27T00:00:00.000000
```
… (for the remaining channels)

TapeDigitiser-specific messages containing information from the jobInfo file written by the digitisation program are of the form:

```
tapedigitiser.jobNumber,                          142
tapedigitiser.tapeDigitiserName,                  Tape1
tapedigitiser.tapeDriveName,                      Metrum1
tapedigitiser.softwareVersion,                    1.1.11
tapedigitiser.operatorName,                       PF
tapedigitiser.array,                              YKA
tapedigitiser.tapeNumber,                         3247
tapedigitiser.processingDate,                     2008-04-16T00:00:00.000
tapedigitiser.tapeStartTime,                      1988-11-14T16:00:00.000
tapedigitiser.tapeEndTime,                        1988-11-18T15:59:00.000
…
```

TapeDigitiser-specific messages about errors reading the time code track during import of the digitised data into the BDS system with program “bdsImportTapeDigitiserData” are of the form:

```
tapedigitiser.jobSession0.errorList[0].num,        12
tapedigitiser.jobSession0.errorList[0].str,        2008-04-16T13:40:12 1988-11-
15T03:43:44.237 VELA Bit Errors: 1.000000secs
…
```

TapeDigitiser-specific messages about deviations in the analogue-tape carrier frequency (the data are recorded as FM on the analogue tape) detected during digitisation are of the form:

```
tapedigitiser.jobSession0.errorList[72].num,       12
tapedigitiser.jobSession0.errorList[72].str,       2008-04-16T20:32:07 1988-11-
18T16:35:19.824 FM Carrier Frequency deviation: (241.606430) 2.471020secs
…
```

TapeDigitiser-specific messages about quality of signal during digitisation - the final number (here 0.797737, between “TapeDigitiser” and “sheila”) is the fractional quality, a number between zero and one, where one is best:

```
Warning,                                           27,QualitySignalLevel,
/remote/archive/1987/EKA/Array/ALL/EKA3011_Tape2_00000923//data-000000.bs,1987-
09-24T15:30:28.955000,1987-09-
24T22:33:19.763000,BN:EKA::TapeDigitiser,0.797737,sheila
```

3.5 Warning messages at export

Warnings pertaining to the data that you have downloaded or the original file from which they came appear in a popup window (with scroll bars – do not overlook warnings that are initially out of sight), or in the return email for AutoDRM requests. These include:

1. Missing blocks – data will not be exported in BKNAS format if they contain missing blocks on any channel. In IMS and SEED formats the data are served in segments separated by the missing blocks, each segment with its own header;
2. Re-ordered blocks – where data blocks were not stored in chronological order in the original files, usually because of backfilling after data transmission interruptions, the BDS system re-orders them. This occurs only for formats in which backfilling was possible, i.e. gcf and CD-1.x. Usually blocks are correctly reordered, but it is remotely possible that re-ordering takes place when it shouldn't, because of bit-level corruption of block timestamps;
3. Unknown metadata – where seismometer responses are unknown, default values are inserted;
4. “TapeDigitiser Digitisation quality warnings/errors list in infoExtra” - this warning appears for all data originally digitised from analogue in the TapeDigitiser project. “infoExtra” can be viewed by issuing the bdsDataAccess command with option “-command dataInfoFull” (see below). Note that some, but not all, of this information is visible in the “DataViewInfo” tab (after pressing “Update” while the data timespan and channels are selected in the DataSelect tab);
5. Notification of accompanying log files, e.g. scanned tape-logsheets from the original analogue tapes for TapeDigitiser data. At present there is no way to access these log files.

3.6 Failure to export

If the data are not exported after you have chosen the filename and pressed “save” then an error message appears. The most common errors include:

(a) Failure to export in BKNAS format – this is for one of four reasons:

i. the data are (apparently) not synchronously sampled across all the channels to be exported. The message for non-synchronously sampled data is “Error: Time stamps not aligned across all channels”.

ii. there is a gap (missing blocks) in one or more of the channels being exported. The error message is “Error: Can only export consecutive data segments in BKNAS format. The data set has n segments”;

iii. more than 31 channels have been selected;

iv. the selected channels do not all have the same sampling rate. The error message is “Error: data channel sample rates are different”

(b) Pressing “Download” before pressing the “Update” button. The error messages for this do not point clearly to the cause, and might be “Error: data does not overlap across all channels for data time clipping”, or “Internal error: no actual data segments are available from file although database says there should be some”;

(c) File reading errors – these are caused by bugs in the BDS program and should be reported to the administrator. The message might be of form “Error: BdsDataFile: end of data reached on seek...” Changing the requested timespan to avoid missing blocks (which can be identified in the “Data Notes” and “Data Info” button windows on the “Sensor Data” tab of the User GUI and Admin GUI) might overcome the problem;

(d) File writing errors – unlikely unless you lack permission to write in the nominated directory. The program will prompt for permission to overwrite if a file with the same name as your chosen output file exists in the current directory;

(e) Metadata error – the message is of form “Control::channelGetList: no information on channel XX for time period TT”. This might mean that the data file has been labelled with a channel that does not exist in the metadata, e.g. “BH1” when the channel in the metadata is called “BHZ”. It might be caused by changes having been made to the metadata after import of the data. Metadata errors should be corrected by the database administrator.

3.7 Other points about BDS BKNAS format

It is NOT RECOMMENDED to export in BKNAS format any data that might have digital sample numbers more than 6 digits long. This is because the column width is 6 digits long (including sign). The BDS ignores the 6-digit limit and writes the numbers to full length, pushing the number in the next column rightwards; the Fortran programs that read BKNAS (Blacknest, 2008) are hard-wired to 6-digit-wide columns and will attribute digits in excess of 6 to the next channel, or ignore them in one-channel output. If you need data with sample numbers more than 6 digits long in BKNAS format, download them in IMS format and convert them with “conv”, which will write numbers more than 6 digits long as six stars,

preserving the column widths. Data originally in BDRS, BERS, WRA40 and WRA64 formats were digitised on a digitiser incapable of producing numbers greater than 6 digits long. TapeDigitiser and GCF-format data samples are more than 6 digits long. SEED data from YKA from after ????1999-12-16 have samples greater than 6 digits long. If you really need data from high-dynamic-range sources in BKNAS format then export them in IMS integer format and write a program or use “awk” to scale the numbers down (but don't forget to adjust the calibration factor to allow for the scaling – downscaling *increases* the calibration factor).

Data that were imported with “-synchronous” as an option to the “bdsImportData” command can be exported in BKNAS format. This includes TapeDigitiser data, EKA broadband array data (in which the channels are synchronously sampled but stored separately) and data from 3-component seismometers connected to a single digitiser. Beware that it is possible to export data from two or more 3-component stations in a single BKNAS file, giving the misleading appearance that the different stations are synchronous with each other.

Infrasound data from BKNI are not synchronously sampled, but infrasound data from EKAI and IMS infrasound array stations I49, I50, I51 and I52 (channels BDF and BDA) are assumed to be so. Hydrophone data from the separate subarrays at IMS hydroacoustic stations H08 and H10 are synchronous within the subarray but not between subarrays, so it is possible to export the three H08N channels together as BKNAS, separately from the three H08S channels.

Where possible the instrument number in the BKNAS header corresponds to the correct instrument number in Blacknest subroutine “inst” (ref). The pit names are given in full, e.g. “EKR1” rather than “R1” as the “Robby” AutoDRM gives them, but the instrument name is the truncated “type” from the BDS table “Sensors”, e.g. “CMG-3T”, rather than the truncated “name” output by “Robby”, e.g. “EKA DSP”.

BKNAS has no provision for the two-digit location code, so this is written in the otherwise unused columns 73-74 of the first row of the two-row station header. When the channel is of reversed polarity (upwards motion giving a negative-going signal), IMS uses a negative calibration factor. The BKNAS-1 format description states that the calibration factor should be positive and a “-” inserted in column 71 of the first row of the two-row station header. Some BKNAS data have a negative calibration factor as well as or instead of this “-” sign, and “Apple” takes note of the sign of the calibration factor, so the BDS BKNAS-1 data have both the “-” in column 71 and negative calibration factors where appropriate.

4 Instrument Response Export

Instrument responses are output in seven formats: a BDS native format seen in the GUI response window; IMS pole-zero format; IMS frequency-amplitude-phase table (FAP) format; EvalResp FAP format (IRIS, 2013); SAC pole-zero format, and in the headers of IMS-format data, BKNAS-format data (Blacknest, 2008) and SEED-format data and

dataless headers (FDSN/IRIS/USGS, 2006) . The IMS formats are described in IDC (IDC 2004) and SAC pole-zero format in the SAC manual (IRIS, 2010a) for the SAC command “transfer”. Instrument responses in SAC and “RESP” format can be extracted from the SEED header with program “rdseed” (IRIS, 2010b).

The amplitude column of the IMS FAP table is normalised to one at the calibration frequency of the sensor. The pole-zero normalisation factor (“CONSTANT” in SAC pole-zero files) is equal to 1/[(calibration factor) x (real part of ratio of polynomial formed from zeros over polynomial formed from poles at the calibration frequency)]. The calibration factor is in SI units (metres or pascals, usually) for this purpose.

4.1 From the GUI

A quick guide to instrument response output from the UserGui and AdminGui is given in the first part of this manual.

4.2 From the AutoDRM

A quick guide to instrument response output from the AutoDRM is given in the first part of this manual. The AutoDRM does not serve responses in IMS FAP format.

4.3 Instrument responses from bdsDataAccess command-line program

See above for description of the bdsDataAccess program. The option “-command responses”, if no format is specified with “-format”, puts out the responses on the terminal in approximately the format in which they appear in the “responses” window in the GUI. The responses appear on the terminal even if an output filename is specified with the “-o” option – the output filename is ignored. The terminal output can be redirected to a file with “>” or a pipe “|”. This format does not include the pole-zero normalisation factor (“constant”) or the calibration factor or calibration frequency.

When the format is specified (“-format IMS-POLEZERO”, “-format IMS-FAP” or “-format SAC-POLEZERO”) but no output file name is given with “-o”, default output files are written:

- for “-format IMS-POLEZERO” or “-format IMS-FAP”, “response.ims”, containing all the responses;
- and for “-format SAC-POLEZERO”, files with names of form “response-SSS-CCC-SOU-YYYYMMDD-HHMMSS.sac”, one per channel - see the manual for bdsDataAccess (BEAM, 2010c) (SSS = station code, CCC = channel code, SOU = source name, and YYYMMDD-HHMMSS is the specified “startTime”).

You can change the output filename for IMS output with “-o filename”, but not for SAC output if there is more than one channel. This is because the filename that you specify with “-o” is re-opened for each channel, and the program will fail if (as usually) there is an operating-system injunction against overwriting pre-existing files.

An example command to obtain instrument responses with bdsDataAccess can be constructed from the example command for data export in the previous section by replacing “-command dataGetFormatted” with “-command responses” and “-format IMS” with “-format IMS-POLEZERO” or “-format IMS-FAP”. If you want SAC poles and zeros, use “-format SAC-POLEZERO”, delete the “-o” filename and let the program permute the default filename.

5 Station and Channel Metadata Maintenance

5.1 Changing a channel name after data have been imported with the old name

Do this only if you want to change the channel name for all data at all times at present on the BDS system. For changing channel names retrospectively because of an orientation change when there are data on the system from before and after the change, see below.

In the “Meta Data” tab of the AdminGui choose the channel you want to change, and change the channel name i.e. the 3-letter channel name or the location code, or both, at the top of the window. Press the “Channel info Add/Update” button and answer “Yes” to the “are you sure?” prompt. Then press “Save” and again answer “Yes” to the “are you sure?” prompt. The channel name has now been changed in both the metadata and the individual data files, in ALL the places where the old channel name appears. Note that changing the start or end date at the same time as changing the channel name does NOT limit the change of channel name to only those files within the new date span.

5.1.1 Rearranging channels after data have been imported with channel names in wrong order

Most data types require you to specifiy the list of channels being imported, in order corresponding to the order of specification of the data files for a multi-file import (e.g. EKA broadband data where “format” in the 2006 AutoDRM database is “GCF24EKABB”) or the order of the channels in the file for a single file of multiplexed data (e.g. a TapeDigitiser or BDRS file). It is not good to get the channel order wrong at this stage, because the channel names are written into the BDS data file and can't be altered. The database table DataChannels can, though, be changed so that the correct channel can be associated with the mis-named channel in the BDS data file.

Go to the “Data Files” tab and input the timespan for the imported file. Identify the file in the list in the middle subwindow and select it (Figure 13). The lowest subwindow should now list just the channels from this file. Select the misnamed channel and press the “Modify” button. This opens an edit window that should allow you to change the channel name.

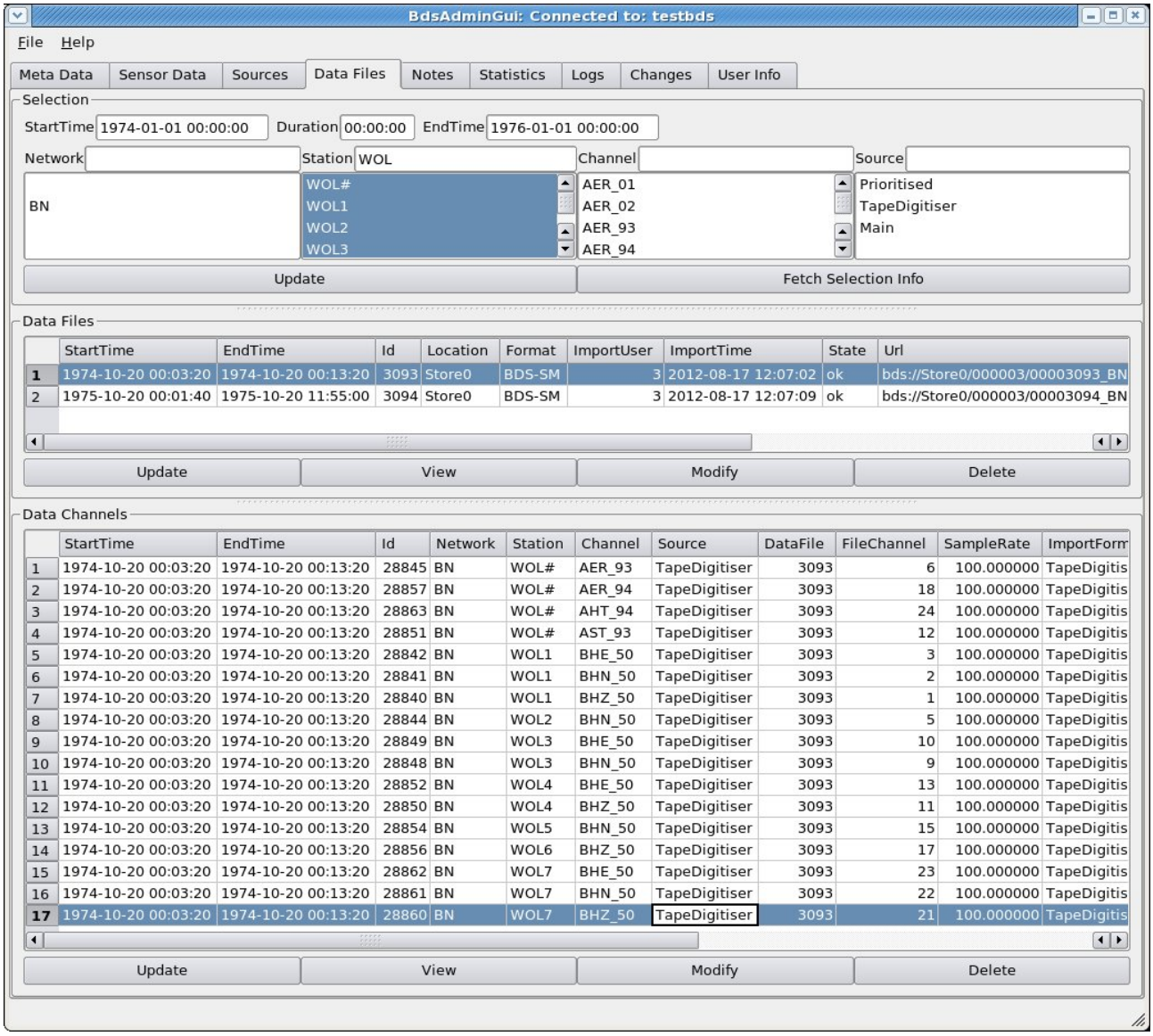


Figure 13: DataFiles tab window. Middle subwindow shows the files available for the timespan and station selection criteria in the upper subwindow. Lowest subwindow shows the channels in the selected file. Note the "ImportTime" is in UT.

If you need to swap two channels, it is a little more complicated because you cannot save a change that causes a channel to be an exact duplicate of another channel in the file. For example, to swap two channels from “BHE”, “BHN”, to “BHN”, “BHE”, you cannot just select the first and change the “BHE” to “BHN”, because it is now identical to the unchanged second. You have to change one of them to have a dummy station (just as you have to do when exchanging two items in an array in a computer program):

(a) Select the first channel to be changed and change its station name to something like “DUMMY”;

(b) Press “Save” - unfortunately the entry you have just changed will disappear or move because the program re-sorts the channels into alphabetical order; also, if “DUMMY” does not exist in the station metadata, you will have to put a wildcard (“*”) into the “Station” slot at the top and press the “Fetch Selection Info” button to see it at all. Beware – this de-selects your file in the middle pane, causing all the channels in all the files to be displayed in the lowest pane, and it is very easy to get mixed up about which file you are changing if there are several from the same network;

(c) Re-select your file from the middle pane, then select the second channel to be changed. Change its channel name to the name of the first one;

(d) Press “Save” and go through instructions (b) again;

(e) Re-select your file from the middle pane, then select the entry with the dummy station name (your first channel). Change its station name back to the proper value and change its channel name to the name that the second channel originally had. Press “Save”.

5.2 Deleting a station

In the AdminGUI it is possible to delete a station without deleting any of the channels associated with it. This is done simply in the “Metadata” tab, “Stations” sub-tab, by selecting the station and pressing the “Delete” button at the bottom of the subwindow. You will be asked for confirmation. If you then go to the “Channels” sub-tab, you will find that all the channels associated with the station are still listed, even after you press “Fetch Selection Info” and “Update”. You have to delete these channels individually as well as deleting the station. (What actually happens is that deleting a station deletes entries from tables Stations and StationLocations, while deleting the channels deletes entries from tables Channels, ChannelInstruments, Sensors, Digitisers, Calibrations and Responses.)

5.3 How to do a horizontal-component orientation change

When you find out that the horizontal components at a station were mis-oriented at installation time, the channels that had been called “BHN” and “BHE” have retrospectively to be renamed “BH2” and “BH3”, and (optionally but usually) the vertical component previously called “BHZ” renamed “BH1”. When the seismometer is re-oriented by the field crew, you have to update the database so that data before the re-orientation are served with the correct orientation (horizontal angle) and channel names “BH1/2/3” and data after the re-orientation are served with channel names “BHZ/N/E” and horizontal angles 0 (N) and 90 (E).

Firstly, are there no data on the BDS system from AFTER the orientation change? And does the mis-orientation extend right back to the instigation of the channel (beware for UKNET stations such as SBD, for which the vertical channel BHZ extends back for many years before the horizontal ones were instigated)? If the answer to both questions is “no”, then:

1. start by following the instructions in the previous section, “Changing a channel name after data have been imported with the old name”. Change the three channel names from BHZ to BH1, BHN to BH2 and BHE to BH3;
2. Change the end dates of the newly renamed BH1, 2, 3 channels to the time at which the seismometer was re-oriented;
3. In the “Calibrations” section of the Channel Edit” window for the BH2 and BH3 components, change the “horizontal angle” to the orientation of the seismometer horizontal components before re-orientation (the BH3 orientation should be 90 degrees clockwise from the BH2 orientation, when looking down on the seismometer), and change the end time of the Calibration entry to the time of re-orientation.

4. Now you have to re-create channels BHZ, N, E for the post-reorientation seismometer. Go to “Channels” tab and select the BH3 channel for the station. Under “Options” choose “Clone the channels”;
5. In the “Clone a Channel” popup window, change the channel name at the top centre from “BH3” to “BHE”. Make sure that only the following checkboxes are crossed: “Clone channel”, “Clone calibrations”, “Clone Instruments” and “Clone Responses”, i.e. NOT “Clone Sensors”, “Clone Digitisers” or “Use Lastest Data” (sic). This will prevent the duplication of the instrument and digitiser, allowing the existing instrument and digitiser to be used for the new channel. If the field crew changed the instrument and/or digitiser as well as re-orienting it, then it is still better not to duplicate the instrument or digitiser, because the “new” seismometer and/or digitiser might already be in the database;
6. Select the newly created BHE channel and press “Modify” button. You need to change the following dates: startTime at top of “Channel Edit” window to date of re-orientation, startTime of most recent Calibration and startTime of most recent Response to date of re-orientation (the latter because the channel name is a primary key to the Responses database table, although you are not changing the responses i.e. the poles-and-zeros). You also need to change the “Horizontal Angle” in all the Calibrations entries to the orientation of the BHE component (90 degrees). You should not need to alter anything in the “Instrument” entries unless the field crew changed the seismometer or digitiser, in which case look at the following section, “how to do an instrument change”;
7. Delete all entries from “Calibrations” that have dates totally outside the timespan between the re-orientation and the present;
8. Save the changes to this channel, then select the “BH3” channel from the list of channels and press the “Modify” button. You need to change the following times: endTime at the top of the “Channel Edit” window to the date of re-orientation, and the endTime for the most recent Calibration and Response to the date of re-orientation. You also need to delete any Calibrations that are totally after the date of re-orientation. Then save the changes.
9. If the field crew changed the digitiser or seismometer as well as re-orienting it, you need to make new entries in the “BHE” channel for “Instrument”, and one or both of its sub-parts, “Digitiser” and “Sensor”, for the new equipment. You will also have to modify the current Calibration (the one for which you changed the start time in part 5) to show the new calibration value. See “How to do an instrument change”, below.
10. Repeat 3-8, and 9 if necessary, for the “BH2” and “BH1” channels (creating cloned channels “BHN” and “BHZ” respectively), remembering to change the “horizontal angle” in the Calibrations of the “BHN” channel to the correct value (0 degrees);

If there are data on the system from before and after the re-orientation, or before the mis-orientation, then do NOT edit the channel names from BHZ/N/E to BH1/2/3 but create new channels, starting on the date of installation (or mis-orientation) and ending on the re-orientation date, called “BH1”, “BH2” and “BH3”. Only if the mis-orientation dates from the first installation of the station, change the start date of the existing “BHZ”, “BHN” and “BHE” channels to be the re-orientation date. The procedure is:

1. Go to “Channels” tab and select the BHE channel for the station. Under “Options” choose “Clone the channels”;
2. In the “Clone a Channel” popup window, change the channel name at the top centre from “BHE” to “BH3”. Make sure that only the following checkboxes are crossed: “Clone channel”, “Clone calibrations”, “Clone Instruments” and “Clone Responses”, i.e. NOT “Clone Sensors”, “Clone Digitisers” or “Use Lastest Data” (sic). This will prevent the duplication of the instrument and digitiser, allowing the existing instrument and digitiser to be used for the new channel (if the field crew changed the instrument and/or digitiser as well as re-orienting it, you still want to use the existing instrument/digitiser, because the new channel you are about to create is for the time before the change);
3. Select the newly created BH3 channel and press “Modify” button. You need to change the following dates: endTime at top of “Channel Edit” window to date of re-orientation, endTime of most recent Calibration and endTime of most recent Response to date of re-orientation (the latter because the channel name is a primary key to the Responses database table, although you are not changing the responses i.e. the poles-and-zeros). If the mis-orientation does not date from the first installation of the station, you need to change the startTime to the start of the mis-orientation at all the places where you have just changed the endTime. You also need to change the “Horizontal Angle” in all the Calibrations entries to the orientation of the pre-correction BH3 component. You should not need to alter anything in the “Instrument” entries;
4. Delete all entries from “Calibrations” that have dates totally outside the timespan of the mis-orientation;
5. Save the changes to this channel, then select the “BHE” channel from the list of channels and press the “Modify” button. If the mis-orientation dates from the installation of the station, you need to change the following times: startTime at the top of the “Channel Edit” window to the date of re-orientation, and the startTime for the most recent Calibration and Response to the date of re-orientation. Then save the changes. If there was a period of time before the mis-orientation when the seismometer was correctly oriented, then do not change the start time at the top. Set the startTime of the most recent Calibration and Response entries to the date of re-orientation, but also ensure that there are valid Calibration and Response entries for times before the mis-orientation, with endTimes at the time of mis-orientation. This usually requires that the endTime of a calibration and response that is during the timespan of the mis-orientation has to be changed backwards to the date of mis-orientation.
6. If the field crew changed the digitiser or seismometer as well as re-orienting it, you need to make new entries in the “BHE” channel for “Instrument”, and one or both of its sub-parts, “Digitiser” and “Sensor”, for the new equipment. You will also have to modify the current Calibration (the one for which you changed the start time in part 5) to show the new calibration value.
7. Repeat 1-5, and 6 if necessary, for the “BHN” and “BHZ” channels (creating cloned channels “BH2” and “BH1” respectively), remembering to change the “horizontal angle” in the Calibrations of the “BH2” channel to the correct value;
8. In the “Data Files” tab, enter the start and end date of the mis-orientation and the

original names of the channels (BHZ/N/E) and press “update” to get a list of all channels in all files affected by the mis-orientation. In the lower, channel edit, pane, change ALL the channel names from “BHZ” to “BH1”, “BHN” to “BH2” and “BHE” to “BH3”.

9. If the re-orientation occurred in the middle of a data file, then you must delete the file and re-import it twice, using the time of re-orientation as “endTime” for the first re-import and “startTime” for the second.

Test by exporting data from before and after the change, in IMS format, and inspecting the “hang” and “vang” values and the channel names in the WID2 line (“grep WID2 datafilename”).

5.4 How to do an instrument change

When a seismometer or digitiser at a station is changed, generally the calibration changes. It is also necessary to show the change of “sensor” or “digitiser”, because the serial number changes even if the other characteristics are identical. The channel names will not change unless the seismometer is re-oriented at the same time (see above). The procedure is:

1. In the Channels tab, choose the station and press “Update”. Then choose the first channel to be affected by the instrument change (e.g. BHE) and press “Modify”;
2. In the Channel Edit window, select the most recent calibration in the Calibrations subwindow and choose “Options – Split at time”. Enter the date and time of the instrument change at the prompt, then, when the “Calibration Edit” window appears, enter the new Calibration Factor and anything else that has changed;
3. In the Instruments subwindow, select the most recent instrument and press “Options – Split at time”, then enter the date and time of the instrument change;
4. In the “Instrument Edit” window, you have choices:
   i. If the new seismometer or digitiser has never been used before but is similar to one that has, then you can “Clone this Sensor” (or digitiser) or “Clone an Existing Sensor” (or digitiser), then choose a similar one from the list, and then edit it to change the serial number. Possibly useful at this stage is the column “old ID” in the list, which refers to the “inst_id” value of the seismometer in the original 2006 AutoDRM database, which, if less than 99, is equal to the value of “inst” used by the “robby” AutoDRM and old Blacknest programs that used subroutine “inst” from the Blacknest library, a list of which is below;

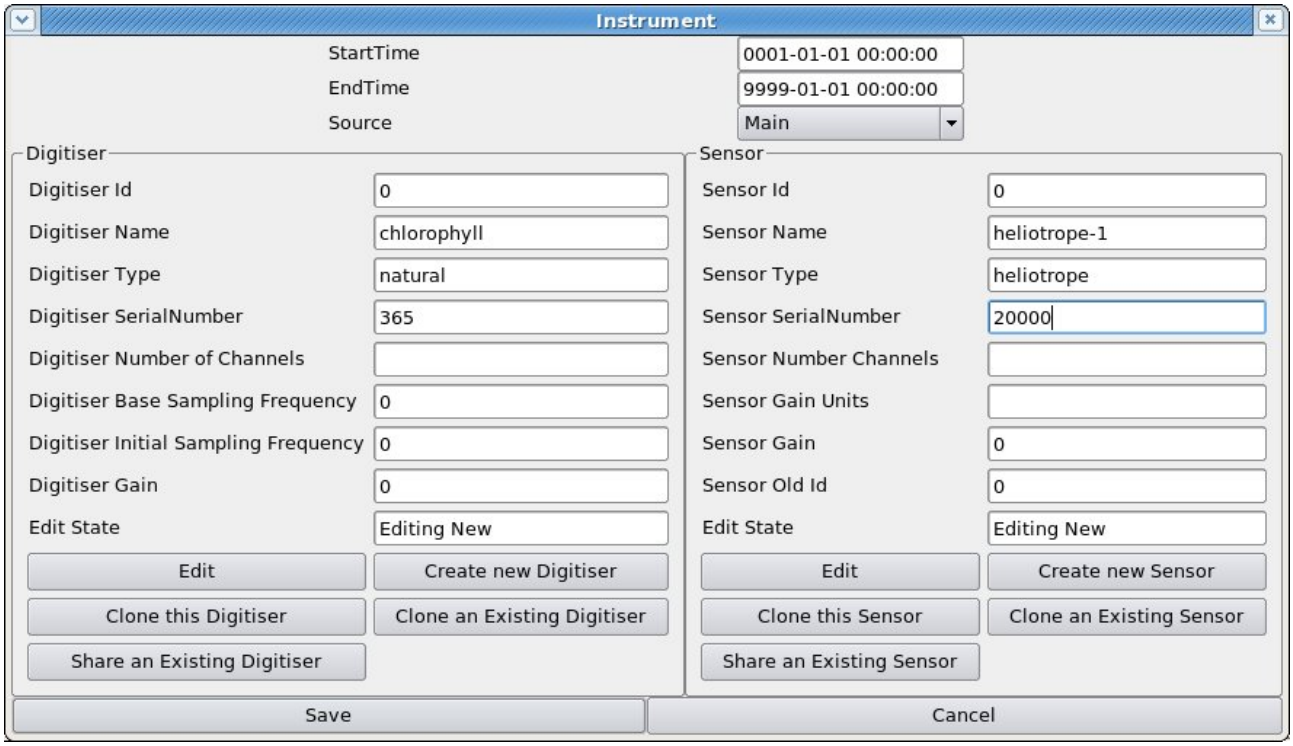


Figure 14: Instrument edit window with new digitiser and new sensor entered. Note "Edit State" slot, showing "Editing New", which shows that we have created brand-new digitiser and sensor. The system will automatically provide the "Digitiser Id" and "Sensor Id". It is not necessary to fill in any of the four entries from "Number of Channels" down.

ii. For a totally new type of sensor or digitiser, you can press "Create new Sensor" (or digitiser) and enter all the details from scratch (Figure 14) (if the "new" sensor is one from the "inst" collection – see below – please enter its "inst" number as "Sensor Old Id");

iii. If the seismometer or digitiser is one that has already been used elsewhere and is in the database, then press "Share an Existing Sensor" and choose it from the list Figure 16) (you can always press "Cancel" on the list if you can't find it). For a brand-new three-component seismometer, once you have entered the details in "Create new sensor" for one of the components, you should use "Share an Existing Sensor" to associate the same seismometer with the other two components. Likewise for the digitiser, use "Share an Existing Digitiser".

5. When one or other, but not both, of the seismometer and digitiser has been changed, check that the unchanged one says "View Shared" in the "Edit Status" slot (Figure 15) and don't alter anything about it.

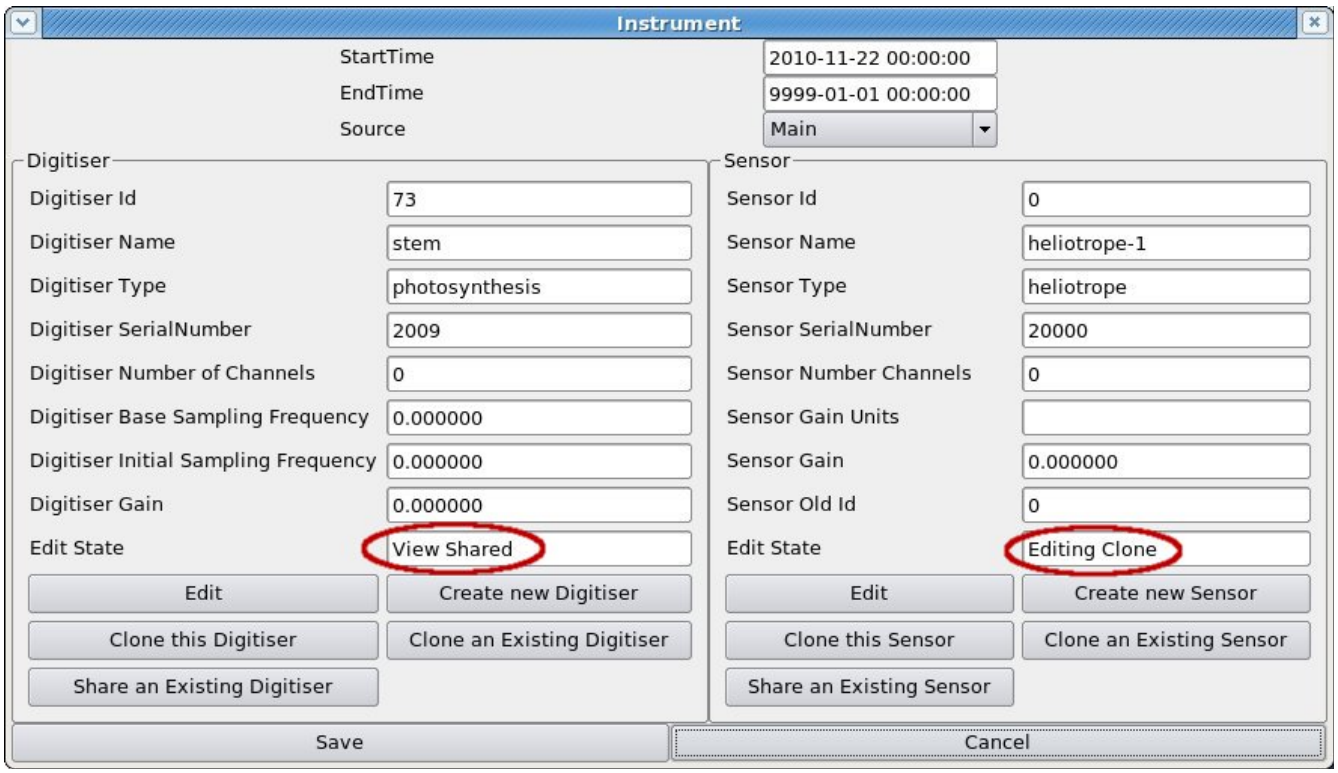


Figure 15: Instrument edit window showing entry of a new sensor based on cloning an existing sensor ("Edit State" is "Editing Clone"), and a digitiser shared between this sensor and others (e.g. in an array) ("Edit State" is "View Shared").

6. Press "Save" when you have finished.

List of old instrument IDs used by "robby" and other old Blacknest programs

CNo	Code	No	Code	No	Code	No	Code	No	Code
C	1	YKA LP	21	LRSM SP	41	LPNB Z	61	CMG-5T VEL	81	KS54M VBB
C	2	LRSM LP	22	NORSAR SP	42	LPNB HOR	62	CMG-40T DISP	82	KS54I VBB
C	3	WWSSN LP	23	YKA SP	43	VBB Z	63	CMG-40T VEL	83	KS36I VBB
C	4	GBA LP	24	WOOD-AND/SON	44	VBB HOR	64	CMG-3ESP VBB	84	STS-2 VBB
C	5	EKA LP6	25	WWSSN SP	45	FLK LPNB Z	65	SP KIRNOS	85	STS-1V VBB
C	6	UKN LPNB(LP2)	26	GBA SP(WRA SP)	46	FLK LPNB N-S	66	LP KIRNOS	86	STS-1H VBB
C	7	SPARE LP	27	EKA SP2 (SP)	47	FLK LPNB E-W	67	YKA VEL	87	BB-13 VBB
C	8	SRO LP	28	SPARE SP	48	FLK VBB Z	68	NORESS SP	88	GS-13 SP
C	9	UKN LPNB (LP)	29	SRO SP	49	FLK VBB N-S	69	SPARE	89	S-13 SP
C10		BB DISP (DBB)	30	DUM DUM DUM	50	FLK VBB E-W	70	BEN BOG	90	S-13 H
C11		BB VEL (VBB)	31	LP (IDEAL LP)	51	FLK VBB Z AA	71	DWWSSN SP	91	EKA SP
C12		OLD BB (DBB)	32	DBB (IDEAL BB)	52	FLK VBB N AA	72	DWWSSN SP2	92	SPITS SP
C13		OLD KIRNOS	33	SP (IDEAL SP)	53	FLK VBB E AA	73	DWWSSN -1	93	ANTIFILT
C14		RED KIRNOS	34	GBA DSP	54	USSR VBB	74	RSTN SP Z	94	SPARE
C15		SPARE BB	35	EKA DSP	55	SWL LPNB Z	75	IDC BUTT	95	BKN BUTT
C16		UKN VBB2	36	WRA DSP	56	SWL LPNB N-S	76	IDC BUTT	96	3B93
C17		YKA VBBX	37	YKA DSP	57	SWL LPNB E-W	77	HFS LPWB	97	
C18		YKA VBB	38	YKA DSP II (2)	58	SWL VBB Z AA	78	SRO SP ANTI	98	
C19		UKN VBBW	39	WRA DSP II (2)	59	SPARE	79	ASRO SP	99	
C20		SASP SP	40	DUM DUM DSP	60	CMG-5T DISP	80	CAN SP	100	

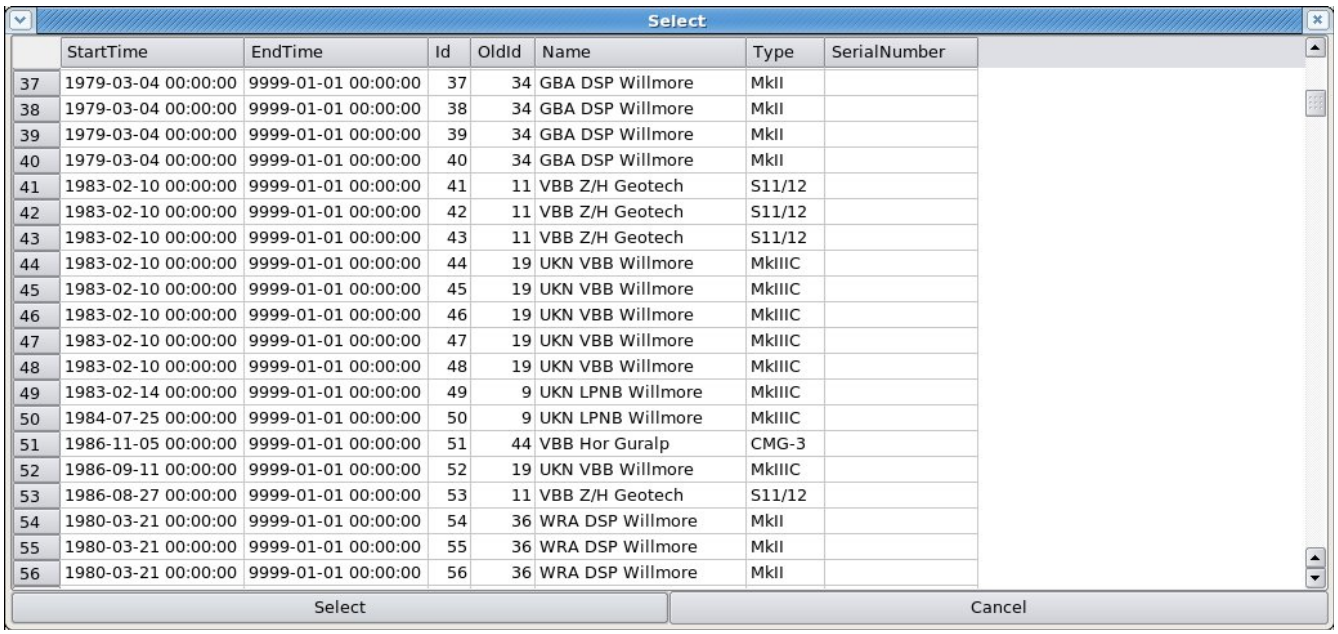


Figure 16: (part of) List from which to choose an existing seismometer to be "shared" or "cloned".

7. In the Responses subwindow, select the most recent response, then press “Options – Split at time”, and enter the date and time of the instrument change. In the “Responses Edit” window, enter the new poles and zeros (unfortunately you have to do this even if they are the same as the old ones).
8. Press “Save” at the bottom of the “Channel Edit” window, then select the next channel to be affected by the instrument change. Go through stages 2-7 above.

5.5 How to change a station or array name

The station name has to be changed in the list of stations, then each channel that was attached to the old station has to be found and the station name on it changed to the new name (rather than creating new channels for the new station name, duplicating the old ones). Otherwise the channels are left “orphaned” in the database. Changing the station name unfortunately causes the station location to be lost, so this needs to be noted and re-typed.

1. In the Meta Data tab, enter the old station name in the Station box and press “Update”;
2. Go to the Stations tab and make a note of the station location (latitude, longitude, elevation) ;
3. For a single station only – this is not necessary for an array - go to the Channels tab and make a note the names of all the channels associated with that station (don't forget the location codes – the part after the underscore);
4. Select the old station name from the list of stations in the Stations tab below and press “Modify”;
5. In the “Station” window, change the station name to the new value. If there is an alias, change that to reflect the new station name. Press “Save”;
6. At the top of the Meta Data tab, first press “Fetch Selection Info” (to update the

display to account for the change to the database that you have just made), then select the new station name from the list of stations and press “Update”;

7. In the “Stations” tab, select the new station and press “Modify”;
8. In the “Station” window, press “Append” beneath the “Location” subwindow and enter the latitude, longitude and elevation of the station (or array centre), then “Save” the “Station” window;
9. For an array you have now finished, but for a single station you now have to re-attach the associated channels. Put a single asterisk into the “Station” box at the top of the Meta Data tab and enter the name of the first channel that was associated with the old station name. Press “Update”;
10. In the Channels tab, choose the channel that has the old station name and press “Modify”;
11. Change the station name in the “Channel” window to the new name and press “Channel Info Add/Update”, and then “Save”, answering “Yes” to the resulting “Are you sure?”-type prompts;
12. Repeat 9-11 for all the other channels associated with the station, then select the new station name from the list of stations at the top of the “Meta Data” tab and enter “*” for the channels and press “Update”. Check that all the channels previously associated with the old station name are present for the new station name.

5.6 How to create a new (3-component) station

1. Create the station in the “Stations” tab by pressing the “Append” button at the bottom. In the “Station Edit” window (Figure 17), enter the name of the station and any description you want, but don't try to enter the station location yet. Press “Save” (if you enter the station location before pressing “Save” then it will return with the error that the station does not exist).
2. Select your newly created station from the stations list and press “Modify”. Back in the Station Edit window, you can now enter its location by pressing the “Append” button beneath the “Locations” subwindow. “OffsetEast” and “OffsetNorth” are for stations that form part of an array, and can be left at zero for non-array stations.

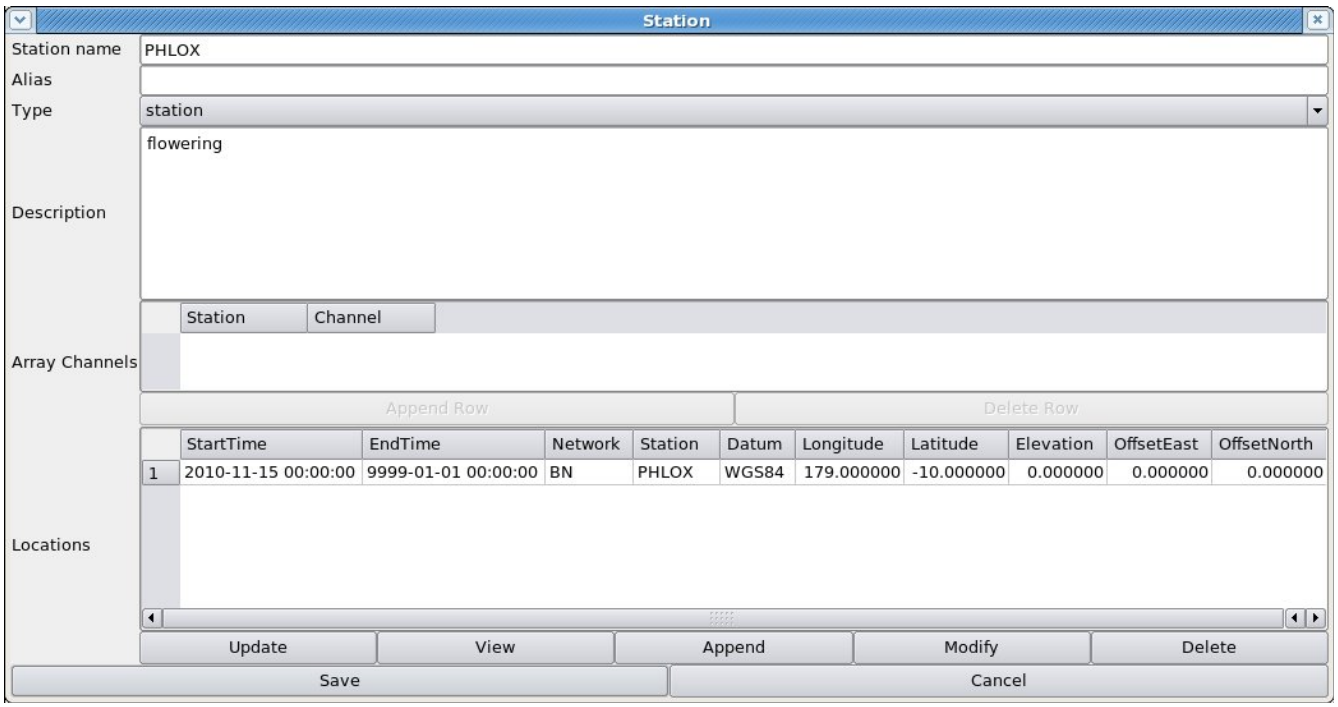


Figure 17: Station Edit window for single station, with location added.

3. Create the channels in the “Channels” tab. Press “Fetch Selection Info” (to bring the displayed station list up to date with the newly added station) then choose your new station and press “Update”. Then press “Append”.

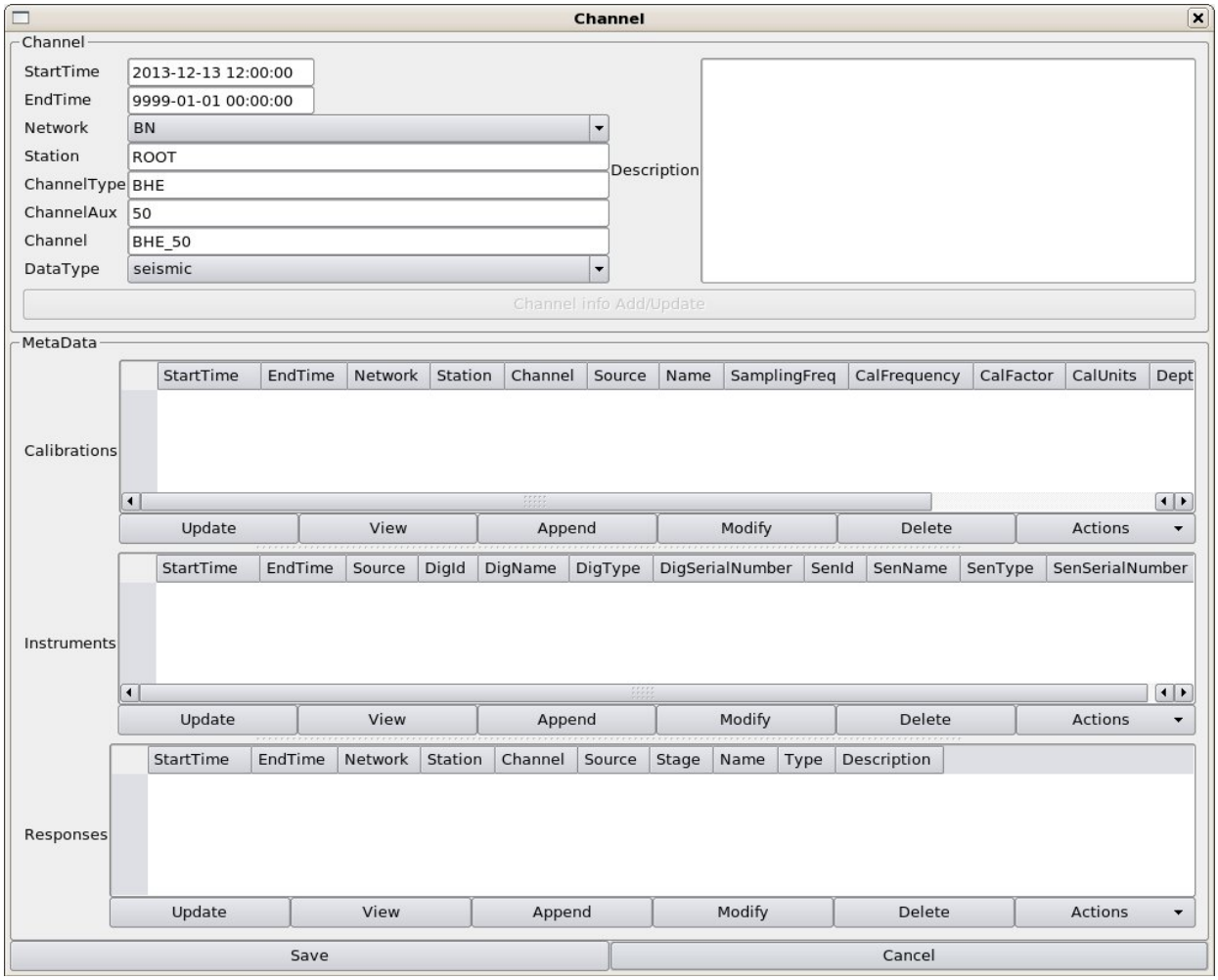


Figure 18: Channel edit window for new channel.

4. At the top of the “Channel Edit” window (Figure 18) enter “BHZ” (or “SHZ” or similar channel designation – SEED manual, ISDN/IRIS/USGS, 2006, Appendix A, contains the rubric) as the “ChannelType”, and the location code, if any, in the “ChannelAux” slot, and any description you want in the “Description” box, then press “Channel Info Add/Update”.
5. Press the “Append” button beneath the “Calibrations” subwindow and enter the appropriate values in the “Edit Calibrations” popup window. The minimum entries in this window are the sampling frequency, calibration frequency and calibration factor. Since for a new station the latter two come from the manufacturer's data sheet, the “Name” should be set to “Main” (the default).
6. Press the “Append” button beneath the “Instruments” subwindow (Figure 19). If your digitiser and/or sensor are similar to those already installed at another station, then you can save effort by pressing the “Clone an Existing Digitiser” and/or “Clone an Existing Sensor” buttons, and choosing the digitiser or sensor of the other station from the resulting lists. The details of the existing digitiser or sensor are displayed, and you should then change the serial number, and anything else that is different, to the values for your new digitiser and/or sensor. Then press “Save”.
7. Press the “Append” button beneath the “Responses” subwindow. This brings up the “Responses Edit” window (Figure 20). Unfortunately the cloning of the sensor does

not copy the “Response” (poles-and-zeros, FAP file, digitiser anti-alias filters, or combination of these), so you have to enter these from scratch. See section below on “Responses”. Before pressing “Save”, check that the “name” of the response is “Overall”. You can type any word into the “name” box, but if it is not “Overall” whatever poles-and-zeros or FAP table you have read or typed in will not be served in the header of output data from this sensor.

8. Once Calibrations, Instruments and Responses sub-windows are displaying the correct values, press “Save” at the bottom of the “Channel Edit” window. Your newly created channel will not appear in the “Channels” tab (disconcertingly), until you press “Fetch Selection Info”, re-select your station from the station list at the top of the page, then press “Update”.

Figure 19: Instrument Edit window with Digitiser and Sensor sub-windows. Note "Edit State" slot, now showing that we are viewing the parameters of a digitiser and a sensor that are shared between this and other "instruments".

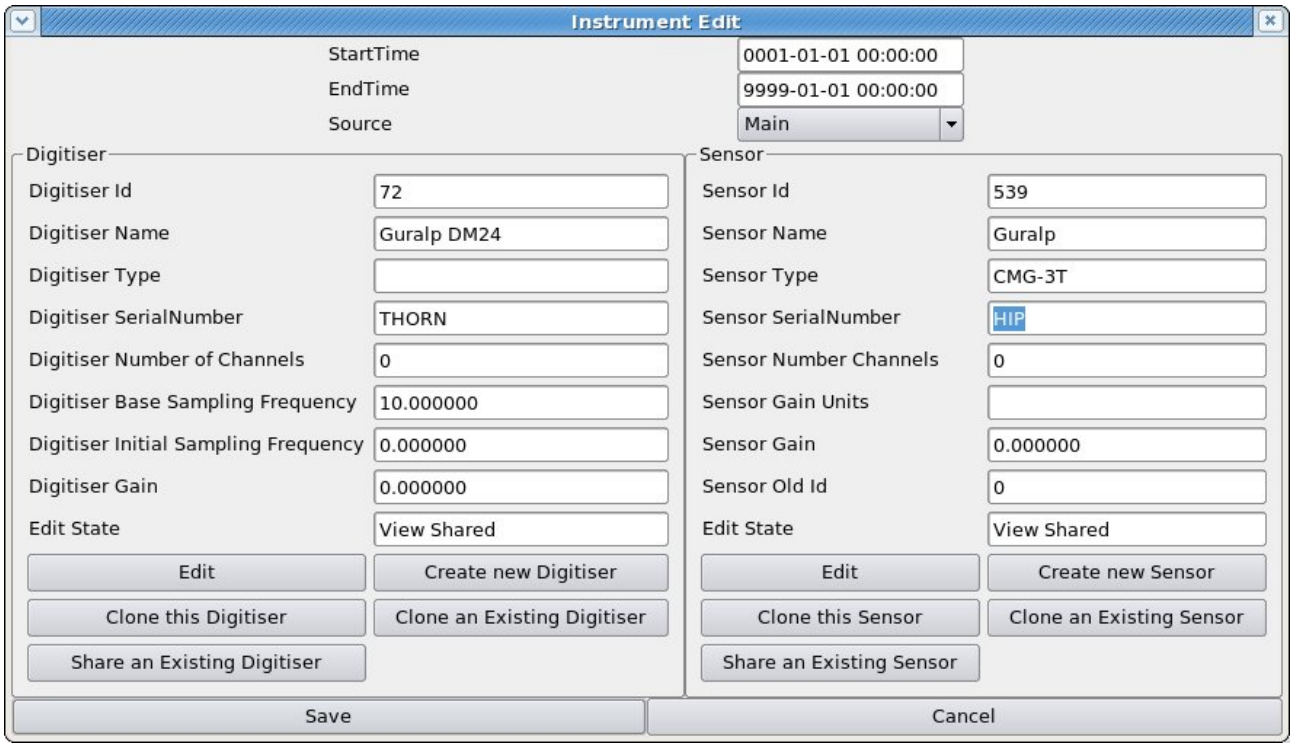
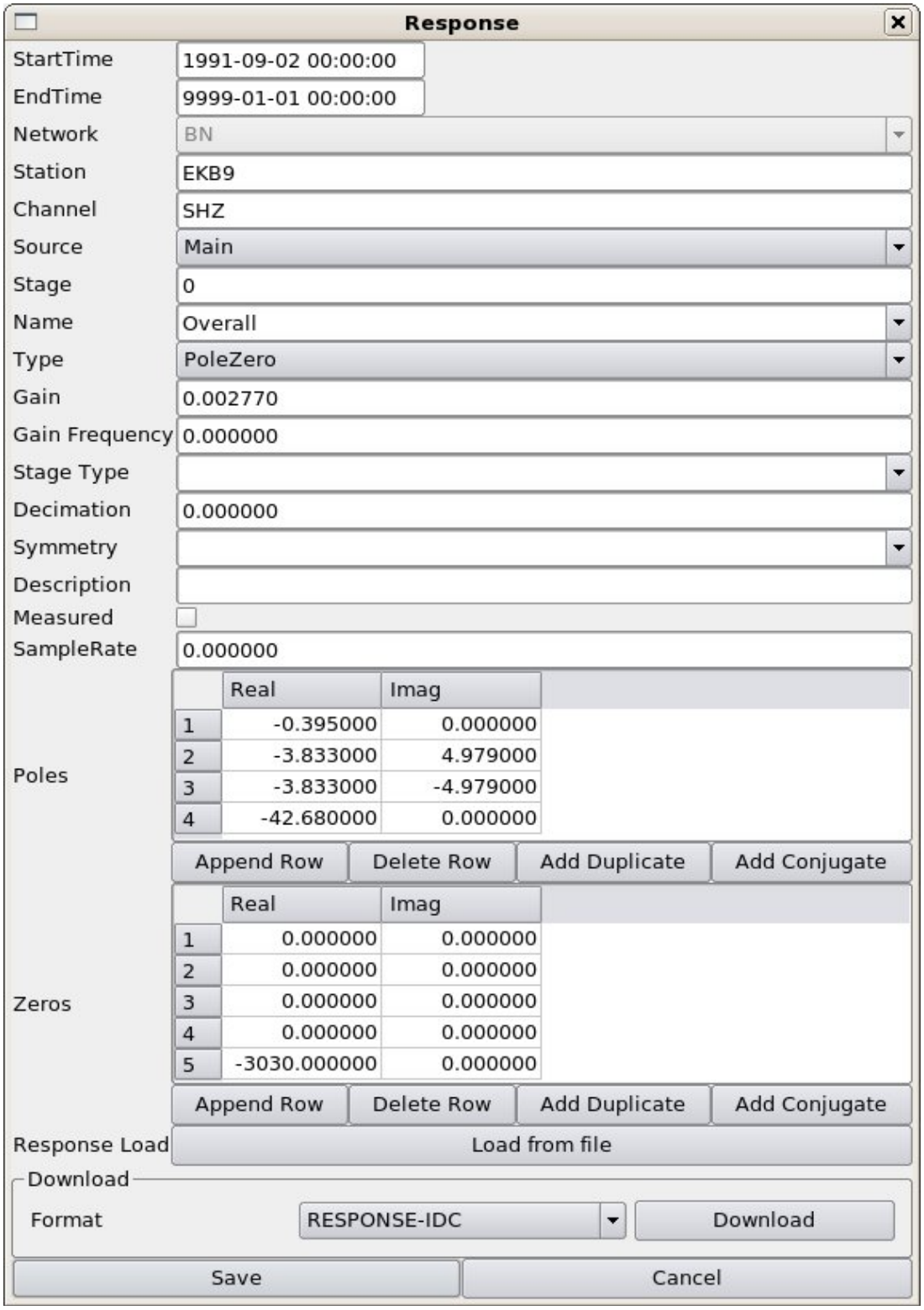


Figure 20: Response Edit window. The "Load from file" button allows you to load responses from a file in various formats (see text). The "Add Duplicate" and "Add Conjugate" buttons add a pole or zero that is the duplicate or conjugate of the currently highlighted pole or zero in the list. If you press "Save" on this window it will warn about the poles with positive real parts (acausal) and refuse to save poles/zeros not in complex-conjugate pairs.



9. Once you have created one channel, the others are created most quickly by selecting the new channel in the “Channels” tab, then press the “Actions” button and choose “Clone the channels”. In the “Clone a Channel” box, enter the next channel name (e.g. “BHN”) at the top, then UNtick the boxes “Clone Sensors” and “Clone Digitisers” (Figure 21). This allows the digitiser and sensor to be “shared” rather than “cloned” between the two channels (Figure 19 – see “Edit State” slot), which is what is required for a 3-component seismometer using a single digitiser.

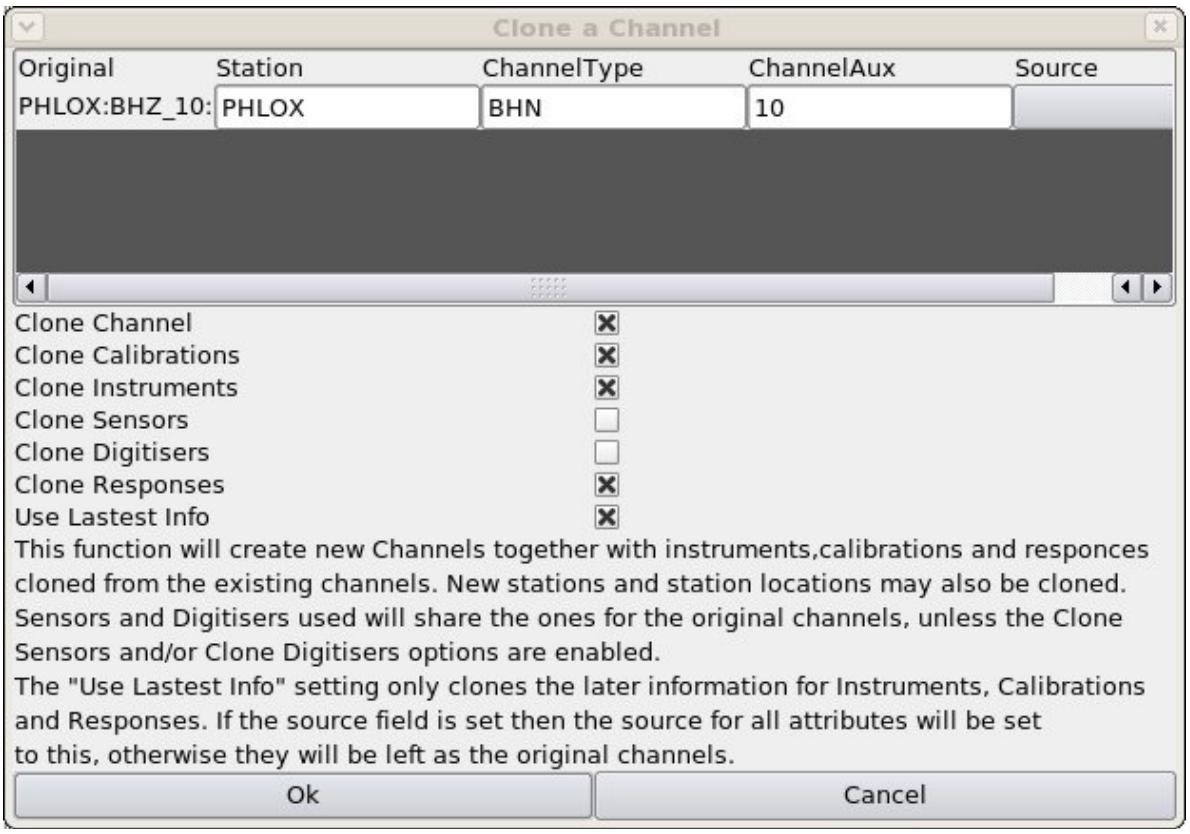


Figure 21: "Clone a Channel" window showing the creation of new channel "BHN_10" from the previously entered "BHZ_10". Note that "Clone Sensors" and "Clone Digitisers" are NOT ticked, so that the sensor and digitiser are "shared" rather than "cloned" between the channels.

10. Select the newly created channel in the “Channels” tab and press the “Modify” button. You need to modify the cloned “Calibration” entry to have the calibration values and horizontal and vertical orientation angles for the new channel (e.g. Vertical Angle 90 degrees, horizontal angle 0 degrees, for a “N” component). You might need to alter the cloned “Responses” entry if the new component does not have the same poles and zeros as the one from which it was “cloned”.

5.7 Responses

Figure 20 shows the “Responses” edit window. “Responses” are described in the BDS manual “BdsResponses” (BEAM, 2013a) and may consist of:

1. A set of poles-and-zeros;
2. A FAP table;
3. A composite response, usually read in from an IDC internal format response file, which usually consists of a set of poles-and-zeros for the instrument, then possibly

another set for the analogue anti-alias filter of the digitiser, then coefficients of one or more finite-impulse-response (FIR) filters describing the digital anti-alias filters of a “cascade” digitiser. Some IDC files, e.g. for infrasound arrays, consist of poles-and-zeros for the sensor and a FAP table representing the digitiser.

Values for all of these may be entered by hand, but in practice it is sensible to read all but the simplest poles-and-zeros from a file. The files recognised by the BDS are:

- Poles-and-zeros in SAC format or in IDC internal response-file format;
- FAP files in IDC internal format or EVALRESP (essentially raw 3 columns of floating-point numbers) format;
- A composite response read from an IDC internal-format file.

The composite response has a number of stages, which are numbered erratically in the IDC internal-format files (often stage 2 or 3 is missing). The BDS reads the stage number and stores it as it is in the input file. It adds a stage “0”, though, which is usually identical to the first stage, representing the instrument only. For most purposes the “Stage 0” response is all that is required in the output. Where the response is represented by a single set of poles-and-zeros or a single FAP, the stage should always be 0 and the “name” should be “Overall”.

In a composite response read from an IDC internal file, the values of “Gain”, “GainFrequency”, “Stage Type” and whether or not the response is “measured” i.e. the outcome of a field or lab calibration, will be read from the file. You may add, or alter, these. “Stage Type” is a letter, “A” - analogue (radians) or “B” - analogue (Hz) (these refer to poles-and-zeros); “D” - digital; “C” - composite (e.g. a FAP representing both instrument and digitiser). “SampleRate” is the sampling rate of the response stage, which for an overall response should be the sampling rate of recording of the data, but for intermediate stages in a cascading digitiser it will be the sampling rate of that stage. The BDS does not use this sampling rate but the one stored in “Calibrations” when writing output data headers.

Simple poles-and-zeros may be typed in, and it is easy to create duplicates, e.g. repeated zeros at (0.0,0.0) or complex-conjugate pairs of poles, with the “add duplicate” and “add conjugate” buttons. When you save, the BDS will complain if the poles have positive (acausal) real parts or are not in complex-conjugate pairs.

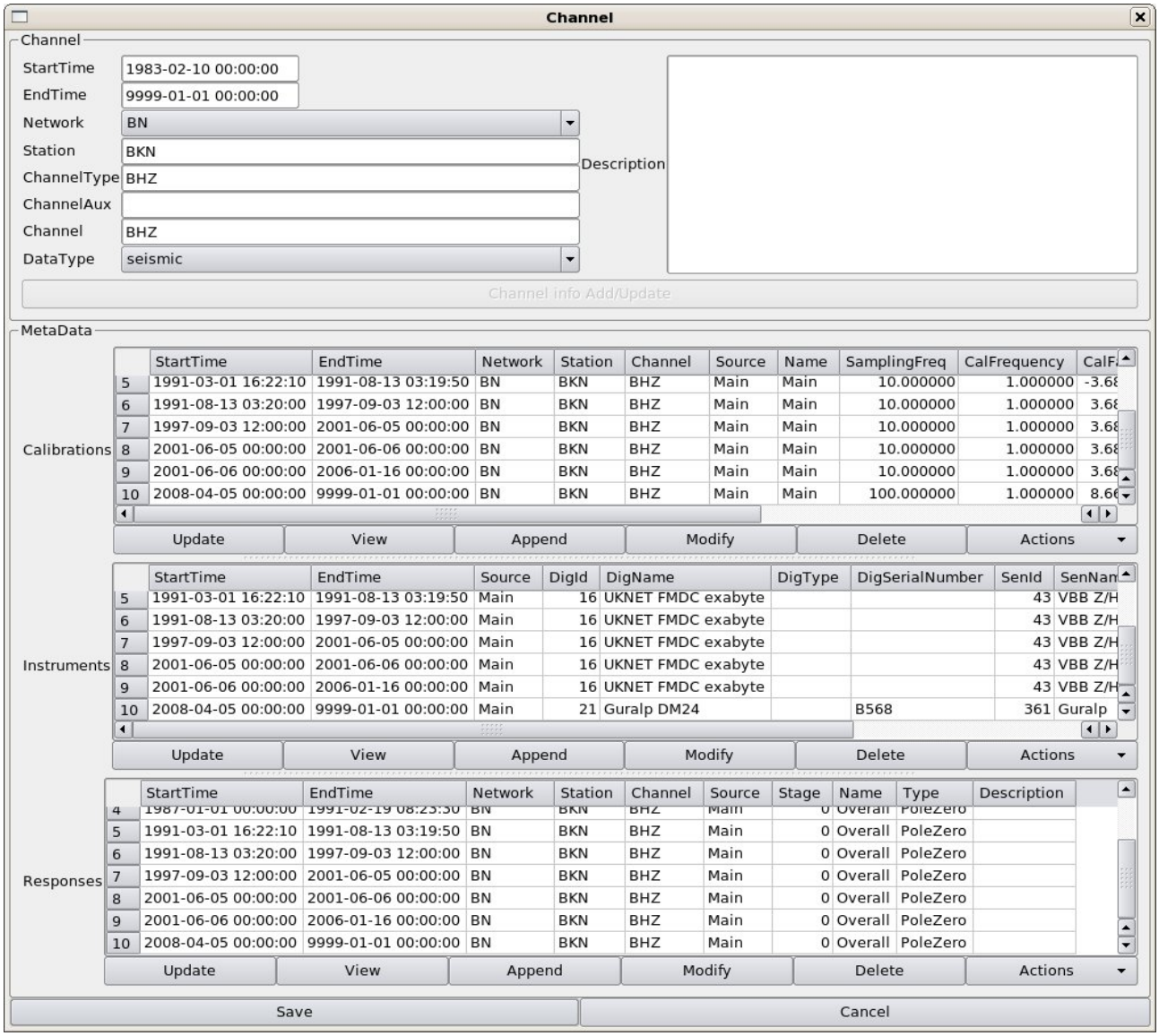


Figure 22: Channel edit window for channel with a long history of instrument changes (BKN)

5.7.1 Aside – creating a dataless SEED file for use alongside miniSEED data in “rdseed”

IRIS program “rdseed” requires the metadata for a station to be provided in a “dataless SEED” file before it will read miniSEED data. The environment variable “ALT_RESPONSE_FILE” has to be set up as the name of the dataless SEED file before invoking “rdseed”. The BDS will write out any response in its store in dataless SEED format, so if you have the poles-and-zeros, FAP or composite format response in a form readable by the BDS (or you type the poles-and-zeros in by hand) you can create a dataless SEED file. The downside is that you have to create a “station” and “channels” within the BDS with names accurate and a timespan corresponding to your data, so that station and channel names (including location codes) match those in the miniSEED data headers, otherwise “rdseed” will reject the data.

5.8 How to create a new Array

An array consists of individual stations but you can create the array and stations independently. To create an array, do:

1. In the Stations window, press the “Append” button. In the resulting “Station Edit” window (Figure 8), enter the name of the array at “Station Name”, choose “Type” to be “Array”, and add any description you want.
2. Add the Array Channels, i.e. the station name and channel name of each channel in the array. After typing the channel name you must hit Return, Tab or a cursor key, otherwise it does not save the channel name. The program makes no check that the station and channel exists, so type carefully if your array contains already-entered stations. Do not forget to include the location code, if any: it should be typed after the channel name with an underscore in between, e.g. “BHZ_10”. When all are added, press “Save” (don't try to insert the location for a brand-new array or you will get the message that it does not exist).
3. Press the “Fetch Selection Info” button then select your new array from the Stations list and press “Update”.
4. Then select the array in the Stations tab and press “Modify”;
5. When the Station Edit window re-opens, press “Append” below the “Locations” subwindow and enter the array centre latitude, longitude and elevation in the “Location Edit” box (Figure 23). Press “Save” in the Location Edit box, and “Save” again in the Station window.
6. If the individual stations of your array do not already exist in the database, create them – see “How to create a new Station” above.

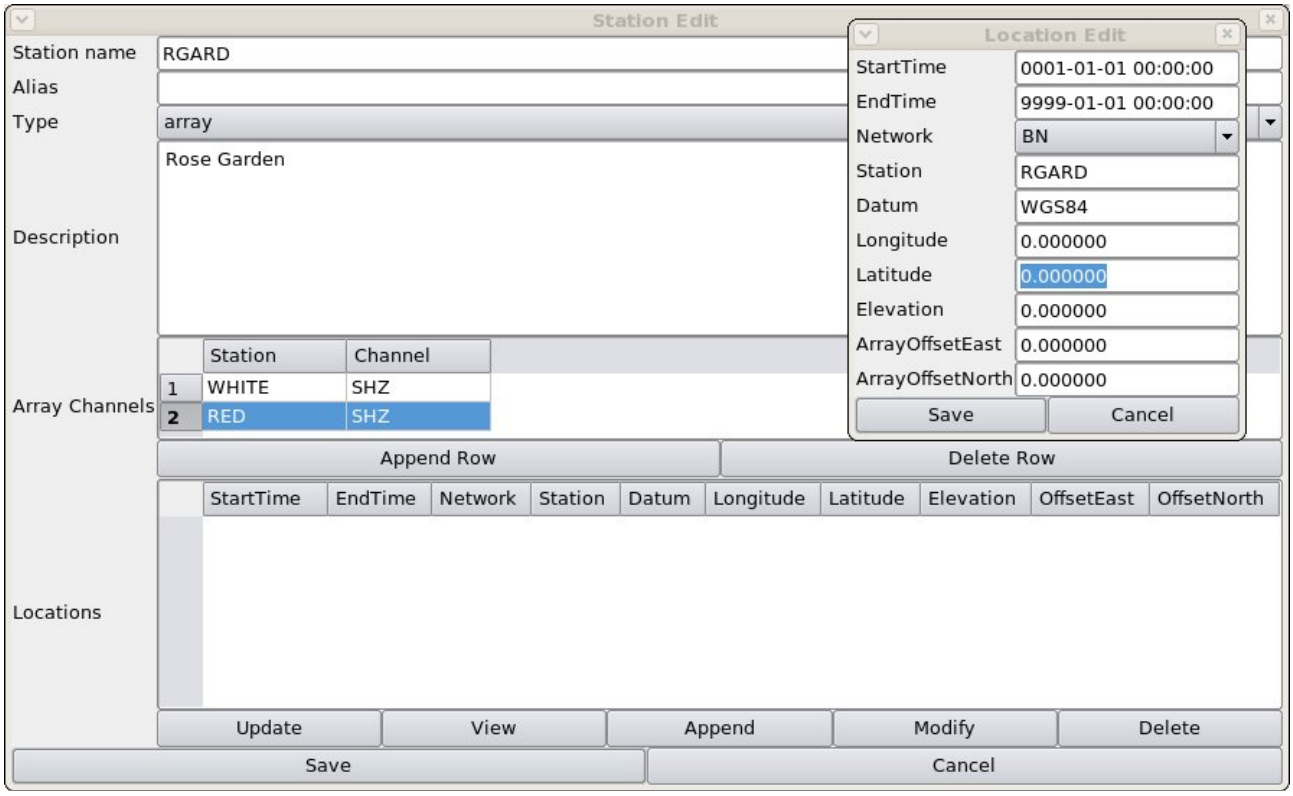


Figure 23: Station Edit window for an array, with Location Edit popup window into which array centre co-ordinates should be typed.

5.9 How to extend a channel back in time

If data older than any that were previously available are to be imported into the BDS, they will require metadata with a timespan covering them. If the channel start date is later than the date of the new data, you will need to backdate the channel start date AND either backdate the current Calibration, Instrument and Response values or create new ones with the older date. The BDS “split at time” function is intended for creating new Channel, Calibration, Instrument and Response entries at later dates, and will not accept a “split time” less than the start time of the entry: to make it do entries for earlier dates requires some editing.

1. Select the desired station, then the desired channel, in the Metadata tab, and press the “Modify” button;

2. Modify the channel start date (at the top of the “Channel” window) to the new, earlier start date;

3. If the oldest “Calibration” entry does not have a start date as early as the new date, select it and choose “Split at time” from the list under the “Actions” button. When the “Split at time” popup appears, set the splitting time to be a short time (one second is good) AFTER the current start time of the “Calibration” entry;

4. When the new “Calibration” entry appears in an “edit” window, SAVE IT WITHOUT EDITING IT – this will remain as the existing Calibration for the original timespan for which it was valid, and so you don't want to change it;

5. There are now two “Calibration” entries with start times one second apart. Choose the

earlier one and press “Modify”, then change its start date to the new, earlier start date and its end date to the original start date, i.e. remove the one second, then save. You may also make other modifications, e.g. to the “calibration factor” or sampling frequency, and in particular, if you are extending the channel back in time to accommodate TapeDigitiser data, then all the new, backdated entries must have “Source” changed from “Main” (or whatever they are at present) to “TapeDigitiser”;

6. Choose the second “Calibration” entry and modify its start date back to the original start time, i.e. remove the one second. You have to do this after modifying the earlier entry, otherwise the two entries will overlap in time and you will get an error message;

7. Follow the same procedure for the “Instrument” and “Response” entries;

8. Press “save” at the bottom of the “Channel” window to save all your changes.

6 Importing Data

Data are imported into the BDS by the command-line programs bdsImportData and bdsImportTapeDigitiserData, manuals for which are available on the BEAM website (BEAM, 2010b). Those manuals list the options available for each command. This manual focuses on the options required to import the particular formats of data available in the Blacknest archive.

Data can be imported in the following digital formats:

1. BDRS (SDAT) (Trodd, H., 1986), including LAC_BDRS produced by the “rescue” reading of old 1/2-in tapes by specialist firms in 2014-15, and the extended BDRS containing multi-multiplexed long-period data in Channels 19 and 20;
2. WRA40 and WRA64 (A. Arcidiaco, ANU, pers. comm., 30/06/2009);
3. WRA_AGSO (K. Muirhead, ANU, pers. comm. 03/12/1996);
4. TapeDigitiser (BEAM, 2008a, BEAM, 2008b);
5. Güralp Compressed Format (GCF) (Güralp, 2010);
6. SEED (FDSN/IRIS/USGS, 2006);

6.1 bdsImportData options

6.1.1 “-ignoreMissingBlocks”

All seismic data in the Blacknest archive have gaps, caused by instrument, telecommunication or storage failures. All the data formats have the data stored in blocks of a few hundred or thousand digital samples, each with a timestamp and known sampling rate. Gaps within data files take the form of one or more missing blocks, which are identified by the break in succession of timestamps. The flag “-ignoreMissingBlocks” should be set for all imported formats, since otherwise the import will fail wherever there are missing data. The timespans and channels for which blocks are missing are noted during import and this information is stored in two places: the BDS file header and the Notes table in the database. Gaps between the end of one data file and the next are not flagged as “missing blocks” and can be detected on import only by scrutiny of the actual start- and end-times of files. This is most likely to be necessary for short data files, e.g.

10-minute SEED files.

6.1.2 “-source”

The “source” value of the imported data should always be set. For most data this is “Main”; the common exceptions are: “TapeDigitiser” for all TapeDigitiser data, “EkaDig2” for all EKA short-period data digitised with the 40-Hz DMOD digitiser, “EkaCD” and “EkbCD” for EKA broadband and EKB data from CDs posted from EKA, and “Scream” for GCF-format data continuously imported straight from the SCREAM data stream.

6.1.3 “-array”

Whenever data from one of the arrays, EKA, YKA, GBA, WRA, BKNI, BAE/SAAS, UKNET or any newly created array are imported, “-array” should be set to the correct array name. This allows the BDS to name the data files accordingly and associate them with the metadata.

6.1.4 Other general flags

“-ignoreFilenameTime” prevents the import program from checking the start time of a file extracted from block headers with the time deduced from the filename (e.g. filenames such as 2008213_0000.EKA for a file beginning at midnight on day 213 of year 2008). For BDRS format files in which the year of the file is not encoded in the header, import will fail if “-ignoreFilenameTime” is set. The BDS is not aware of all the possible formats of file names, and this flag should be set for unusual data , e.g. on a CD-ROM, from which it cannot deduce a time from the file name.

“-warnings” and “-verbose” are useful for testing, but also for bulk importing when it is necessary to determine why a particular import failed, or whether there were data missing from beginning or end of files. “-dryRun” can be set so that import will not take place but all actions short of import will be carried out and the resulting warnings and errors shown (if “-verbose” and “-warnings” are set).

“-startTime” and/or “-endTime” can be used to import part of the data from a file, if, for instance, it is known to be corrupt beyond or before a certain time. They may also be used to import, in two passes, data for which there was a change of station name, or, more usually, channel name in the middle of a data file. Beware that import ceases at the end of the last block before “endTime”, or starts at the beginning of the first block after “startTime”, so if data from the same file are imported in two passes, with “startTime” for one pass and “endTime” for the other both set to the same time, there will be a one-block gap between the two parts of the imported data.

“-ignoreMetaData” should *not* be set, because failure to import data through lack of metadata is a fault that should be investigated and repaired. For a new station or channel, this flag could be set but you will not be able to export the data until you introduce the required metadata to the database. The advantage of entering the metadata before importing the data is that the check against already installed metadata during import is useful for detecting errors. See the instructions above on creating a new station and new

channels. The only place where “-ignoreMetaData” must be set is when you use “-endTime” and “-startTime” to import data from a single file in two passes because of a station or channel name change in mid-file. If the metadata are correctly set up with the time of the station/channel change in mid file, importing will fail with a no-metadata error for the remainder of the file even though it is not being imported because it is after “endTime” or before “startTime”.

“-description” can be set to a user-supplied string (between quotes). This is stored in two places: the BDS data file, and the database table “DataFiles”. The latter, but not the former, can be modified after import, or added if none was specified at import, by selecting the file in the “Data Files” subwindow of the Data Files tab and pressing the “modify” button, then adding/modifying text in the “Comment” box. To view the “description” stored in the data file, go to the “Sensor Data” tab, enter a timespan encompassed by the file, press “Update”, then press the “Data Info” button and look for the entry at “description”, near the top of the window. (If you really want to see it in the raw, then it is also revealed by running “strings” on the BDS-format data file on the BDS server.)

“-addWarning” can be set to add a line to the Notes database with entries in its columns – see Section 7, below.

6.2 Multiplexed formats (BDRS, LAC_BDRS, BERS, WRA40, WRA64)

The flag “-synchronous” should always be set, because these sample-multiplexed data are all, by definition, synchronously sampled. “-allowOverlaps” should be set because there are a number of places where (nominally) the same data are repeated in two overlapping files. The data might not be exactly the same because of differences in tape head cleanliness during transcription from 1/2-in tape to Exabyte, hence we keep both copies.

Do not set “-ignoreCorruptions”, “-ignoreAll” or “-ignoreTimeBackwards”, because block timestamp corruptions should be fixed by pre-processing, otherwise the importing program ignores these “corrupt” blocks. Also, do not set “-deleteDuplicates” or “-reorder”, because BDRS/WRA40/WRA64 data blocks are always in chronological order and never have backfilling – it is only corruption of timestamps that makes them appear otherwise. “-ignoreFilenameTime” should never be set for BDRS data, because the program needs to use the year encoded in the filename to supplement the deficient information in the block headers.

The channel names should be specified in the order in which they are multiplexed, which can be determined from the variable “offset”, and for UKNET multi-multiplexed 1-Hz LP data, “suboffset”, in the 2006 AutoDRM database. The BDS database does not contain “offset” or “suboffset”, so the program must be told the correct channel order. For UKNET data the 10-Hz sampled broadband channels should be specified in the order given by “offset”, then the LP channels with offset “18” (which are multi-multiplexed on to channel 19) should be specified in the order given by “suboffset”, followed by the LP channels with offset “19”, again in order of “suboffset” - 38 channels in all.

6.3 GCF (Güralp compressed)

GCF data are stored in one trace per file and have block boundaries occurring at different times in different channels from the same station. GCF data are never per-sample multiplexed. Nevertheless, data that come from the same digitiser, e.g. all 20 channels of EKA broadband array, and the three channels from each three-component set, are synchronously sampled and hence should be imported with “-synchronous” set. These data are best exported with the “Full Blocks” switch not set, otherwise the exported traces all start and end at different times. Note that BKNI (infrasound array) data are NOT synchronously sampled, although the four stations are designated as an array, because they have separate digitisers and GPS units, so these should not have “-synchronous” set during import.

Multichannel GCF data, e.g. the entire EKA broadband array, or a whole three-component set, should be imported with a single command. Take care to specify the channels in the “-channels” list in the same order as the filenames at the end of the command.

After breaks in transmission, the SCREAM system tries to backfill missing data. This causes data blocks not to be in chronological order, and occasionally data blocks are transmitted twice. The flags “-reorder” and “-deleteDuplicates” should always be set for GCF data. “-deleteDuplicates” operates only when it encounters two blocks that are bitwise identical. Otherwise, where two blocks overlap in time, import will fail with a “time backwards” error. The program carries out the reordering and deleting of duplicate blocks before checking for blocks that are still not contiguous in chronological order. If “-ignoreTimeBackwards” is also set, the overlapping data can be imported, and it is up to the seismological user to heed the warnings (in the “Data Notes” window of the UserGUI “Sensor Data” tab, Figure 24) and deal with the overlap (Figure 25).

6.3.1 “duplicate timed block”

Sometimes two blocks with identical start timestamps but different data are transmitted. The import will fail with the error message "Error,18,Duplicate timed block with differing data", because the BDS is incapable of storing two data blocks with the same network/station/channel/source and start time, even if they are different lengths. There are two possible causes of blocks with same timestamp but different data: 1. backwards clock reset where one of the blocks after the reset happens to have the same timestamp as one before the reset; and 2. backfilling combined with recompression. In the first case the two blocks will genuinely contain different data, in the second, they contain the same data. They are still “differing” because they are different lengths; hence “-deleteDuplicates” does not eliminate the duplicate block.

According to Murray McGowan at Güralp, recompression occurred only for EKB between November 2008 (when the digitiser was changed as part of the upgrade) and a firmware upgrade in early 2013. Data sent from the EKB vault to the SCREAM PC at EKA were in 1-s uncompressed blocks, but they were compressed by the PC for satellite transmission. Any interruption of the line between the vault and the PC led to backfilling, and the backfilled data inevitably had some overlap with the initially sent data. The first block of

backfilled data was often recompressed into a different length from the original data before the interruption, so the two blocks would start at the same place and contain the same data for the timespan for which they overlapped.

Only two hours' worth of data, in four files in total, give this error. They are:

2008-12-02 10:00 EKB BHN and BHE

2009-04-17 17:00 EKB BHZ and BHE

The procedure for importing them is to use the option “-ignoreBlocks” to discard one of each pair of “duplicate timed blocks”. This option can be used only after you have obtained an ordered list of blocks by using the option “-printBlocks” with “-dryRun” and with all the other block-reordering options you intend to use at import: “-reorder”, “-deleteDuplicates”, “-ignoreTimeBackwards” and “-ignoreCorruptions”, usually. The list of blocks produced by “-printBlocks” is then in chronological order, and it is possible to select the blocks to be ignored by number. At minimum one of the two with same start time that are causing the “duplicate timed block” error can be deleted. It is possible to detect two “parallel” strands of contiguous blocks where the data had been sent twice, each with different block boundaries, and select one to be deleted.

The blocks are different for each of the three channels, so it is unfortunately necessary to inspect and import each channel separately, sacrificing the “-synchronous” option usually used when importing a three-component set. The import script containing the list of discarded blocks should be preserved, or the “-addWarning” option used to put the list of discarded blocks into the header of the BDS data file. A suitable”-addWarning” warning can be derived from the list of blocks to be discarded with “bash” shell script commands:

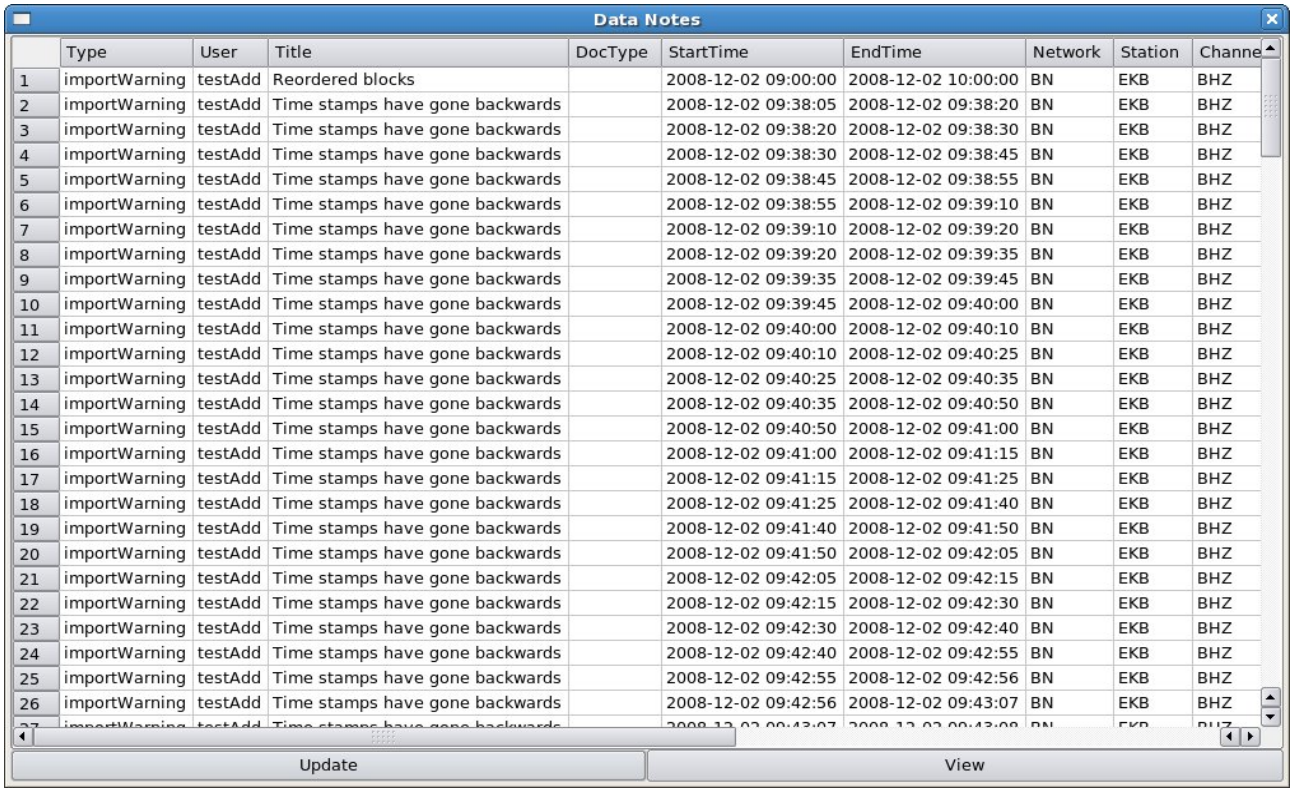
```
ignoreblocks="-ignoreBlocks 1,2,4,6,8,10"

warningmessage=`echo $ignoreblocks | sed -e 's/ /(/' | sed -e 's/$/)/' | sed -e 's/,/;/g'`

warning="-addWarning 99,ignoredBlocks,${file},,,BN:${channels}:${source},$
{warningmessage}"
```

(the “sed” is to remove the space, enclose the list of blocks in brackets, and replace the separating commas with semicolons, so that the warning message can be used as a variable within the script. The commas are replaced because otherwise bdsImportData thinks you are trying to add more parameters to the “-addWarning” option than allowed).

6.3.2 Corrupt blocks

If the import fails with “Unsupported block type: -1” then it should succeed with the flag “-ignoreCorruptions” set. Block type -1 corresponds to the variable G2BLOCKTYPEUNKNOWN, so the contents of the block might or might not be valid. “-ignoreCorruptions” causes the block to be discarded.

Figure 24: "Data Notes" window showing warnings of time-backwards, indicating overlapping data.



Data Notes

	Type	User	Title	DocType	StartTime	EndTime	Network	Station	Channe
1	importWarning	testAdd	Reordered blocks		2008-12-02 09:00:00	2008-12-02 10:00:00	BN	EKB	BHZ
2	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:38:05	2008-12-02 09:38:20	BN	EKB	BHZ
3	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:38:20	2008-12-02 09:38:30	BN	EKB	BHZ
4	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:38:30	2008-12-02 09:38:45	BN	EKB	BHZ
5	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:38:45	2008-12-02 09:38:55	BN	EKB	BHZ
6	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:38:55	2008-12-02 09:39:10	BN	EKB	BHZ
7	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:39:10	2008-12-02 09:39:20	BN	EKB	BHZ
8	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:39:20	2008-12-02 09:39:35	BN	EKB	BHZ
9	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:39:35	2008-12-02 09:39:45	BN	EKB	BHZ
10	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:39:45	2008-12-02 09:40:00	BN	EKB	BHZ
11	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:40:00	2008-12-02 09:40:10	BN	EKB	BHZ
12	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:40:10	2008-12-02 09:40:25	BN	EKB	BHZ
13	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:40:25	2008-12-02 09:40:35	BN	EKB	BHZ
14	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:40:35	2008-12-02 09:40:50	BN	EKB	BHZ
15	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:40:50	2008-12-02 09:41:00	BN	EKB	BHZ
16	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:41:00	2008-12-02 09:41:15	BN	EKB	BHZ
17	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:41:15	2008-12-02 09:41:25	BN	EKB	BHZ
18	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:41:25	2008-12-02 09:41:40	BN	EKB	BHZ
19	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:41:40	2008-12-02 09:41:50	BN	EKB	BHZ
20	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:41:50	2008-12-02 09:42:05	BN	EKB	BHZ
21	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:42:05	2008-12-02 09:42:15	BN	EKB	BHZ
22	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:42:15	2008-12-02 09:42:30	BN	EKB	BHZ
23	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:42:30	2008-12-02 09:42:40	BN	EKB	BHZ
24	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:42:40	2008-12-02 09:42:55	BN	EKB	BHZ
25	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:42:55	2008-12-02 09:42:56	BN	EKB	BHZ
26	importWarning	testAdd	Time stamps have gone backwards		2008-12-02 09:42:56	2008-12-02 09:43:07	BN	EKB	BHZ

Update View

Some corrupt GCF data files contain “zero content” blocks with headers containing no information. The importer issues the message: "Error,There are more channels of data in the files than there is station/channel name info: (n – m)" (where n is the actual number of channels it thinks it has found and m is the number specified in the import command). These can be ignored on import by versions 2.0.10 or greater with the flag “-ignoreCorruptions” set.

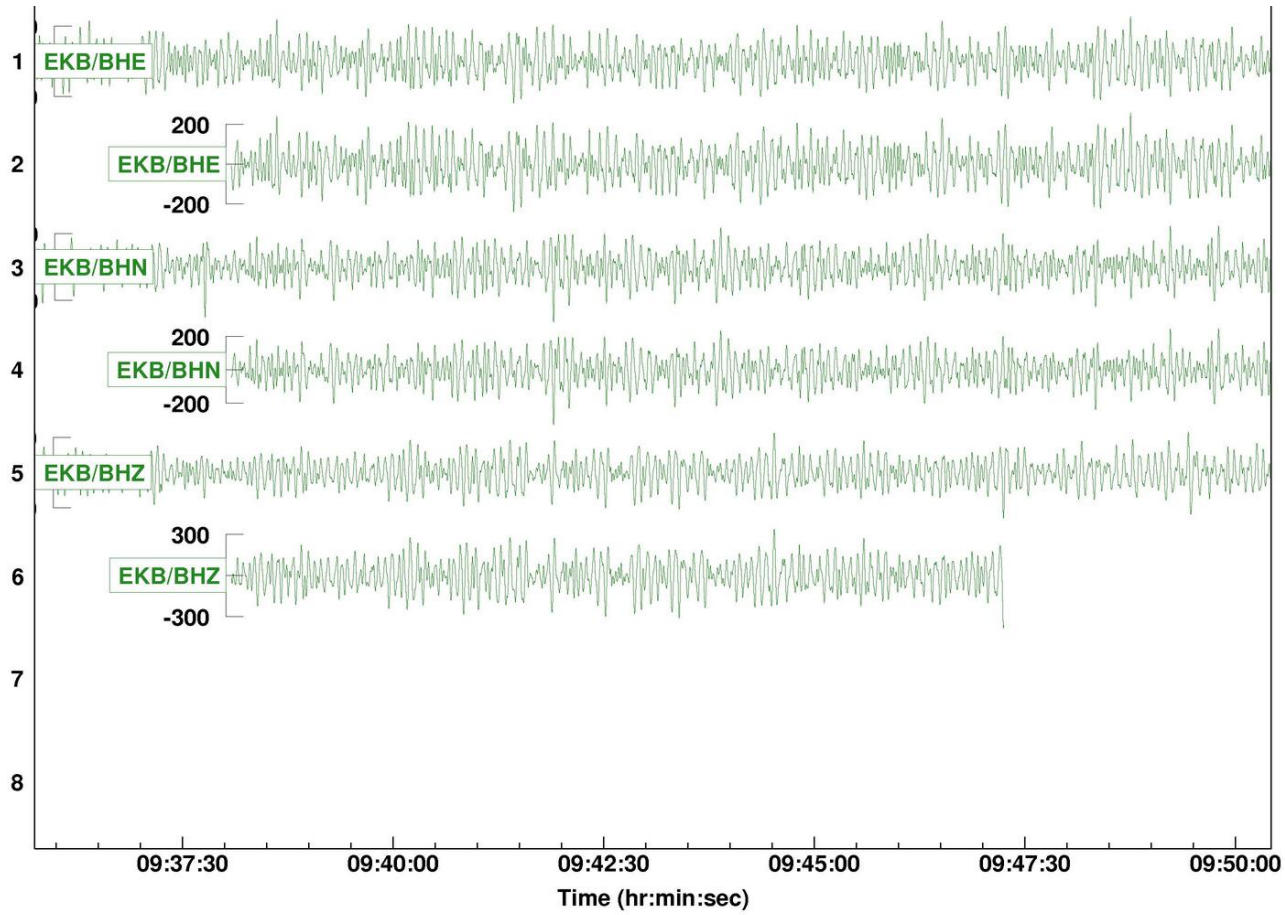


Figure 25: Seismograms from EKB, 2008-12-02, showing overlapping data.

6.3.3 Per-channel-multiplexed GCF

Data from 2000 to 2005 from EKA and EKB were put together into large files containing blocks from all the different channels intercalated. The files were sent to Blacknest on Exabyte tapes each containing about a fortnight's worth of data. The contents of these tapes were dumped into directories named in form “EKBYYYYMMDDHHMM-YYYYMMDDHHMM”, representing the start and end date and time of the data in the directory; each directory contained about 15 files with names YYYYMMDDHHMM-YYYYMMDDHHMM-file*n*.gcf (where *n* = 1 to 15 or so), each containing about a day's worth of data.

V. 2.0.6 onwards is capable of importing these data. It is possible to import some channels and disregard others from the same file, which is desirable for these files because the EKA data are inferior duplicates from the DMOD digitiser of data already imported in BDRS format from the FMDC digitiser. The “bdsImportData” command requires its “-channels” option to include the “system ID” and “stream ID” of each input channel coupled with the desired channel name and location code for that channel, in the format: *Station:Channel[_locn]=Sysid:StreamId*. If the channel is not to be imported, then omit the *Station:Channel[_locn]* keeping only the *=Sysid:StreamId* (the “=” is important). An

example is EKB:BH3=WO1242:3442E2 (note that EKB was mis-oriented prior to 2003-06-23, as described in Bartholomew, P., 2004, so all channel names for the 3-component set should be “BH1[2,3]” before, and “BHZ[N,E]” after, that date). There is sometimes a few seconds of overlap between adjacent files, so the switch “-allowOverlap” must be set.

Some inside knowledge of the digitisers (which generate the “SysID” part of the identifier) is useful. There are four most of the time:

EKA digitiser D850

EKA digitiser D851

EKA digitiser D852

EKB digitiser WO1242 (Guralp D0297)

The EKB digitiser (Güralp serial number D0297) gives rise to either seven or ten channels, with system ID and stream ID numbers:

WO1242:3442[ZNE]2 - seismic data sampled at 40 Hz;

WO1242:3442[ZNE]6 - the same seismic data sampled at 4 Hz;

WO1242:3442M[89A] - mass position data sampled at 1 Hz;

WO1242:344200 - ASCII log data from GPS unit and digitiser,

where the second character is a capital “O” not zero (0). The EKA data are from Willmore Mk II seismometers, which don't generate a mass-position data stream; each seismometer generates two streams, which are labelled according to the digitiser that processes them:

seismic data sampled at 40 Hz ("D85[012]:EKAx4"); and

seismic data sampled at 20 Hz ("D85[012]:EKAx2").

The "x" is 1-9 for EKR1 to EKR9, and A to K for EKR10 and EKB1-10. Each digitiser has 16 channels plus log (I think). The array channels are partitioned among the digitisers:

D850 - R1 to R8 at 40 and 20 Hz, labelled "D850:EKA[1-8][4,2]";

D851 - R9 to R10, B1 to B6 at 40 and 20 Hz, "D851:EKA[9,A-G][4,2]";

D852 - B7 to B10 and four unknown channels at 40 and 20 Hz, "D852:EKA[H-O][4,2]".

The four further streams at each sampling rate, D852:EKA[L-O]4 and D852:EKA[L-O]2, did seem to have data in them, but I don't know what these are.

There are also three log streams, one for each digitiser, D85[012]:EKA00.

There are eleven further temporary data streams and a log stream from a fifth digitiser active in November-December 2002, recording an additional seismometer with sysId "GURALP" and streamId beginning "3B93". I think this is a CMG-3ESP with serial number T3B93 that was active in the EKB vault. It produced the following channels:

GURALP:3B93[ZNE]2 - seismic channels sampled at 40 Hz;

GURALP:3B93[ZNE]6 - seismic channels sampled at 4 Hz;

GURALP:3B93M[8,9,A,B,E] – 8, 9 and A are probably 3-component mass position channels (this labelling is used for mass-position channels with more recent SCREAM GCF data); I don't know about B and E; but all are sampled at 4 Hz.

GURALP:3B9300 - text log stream.

The program does not mind an over-specification of channels; nor does it mind the order, when the sysId and streamId are specified, so the following command works although it specifies more channels than are actually available.

bdsImportData -host <host> -user <user:pass> -ignoreMissingBlocks -reorder -deleteDuplicates -allowOverlap -ignoreMetaData -ignoreFilenameTime -synchronous -format GCF -network BN -array EKA -verbose -warnings -channels EKB:BH3_6=WO1242:3442E6,EKB:BH2_6=WO1242:3442N6,EKB:BH1_6=WO1242:3442Z6,EKB:BH3=WO1242:3442E2,EKB:BH2=WO1242:3442N2,EKB:BH1=WO1242:3442Z2,EKB:MN1=WO1242:3442M8,EKB:MN2=WO1242:3442M9,EKB:MN3=WO1242:3442MA,EKB:BH1_3B=GURALP:3B93Z2,EKB:BH2_3B=GURALP:3B93N2,EKB:BH3_3B=GURALP:3B93E2,EKB:BH1_6B=GURALP:3B93Z6,EKB:BH2_6B=GURALP:3B93N6,EKB:BH3_6B=GURALP:3B93E6,EKB:MN1_3B=GURALP:3B93M8,EKB:MN2_3B=GURALP:3B93M9,EKB:MN3_3B=GURALP:3B93MA,EKB:MN1_6B=GURALP:3B93MB,EKB:MN2_6B=GURALP:3B93MC,EKB:MN3_6B=GURALP:3B93MC,EKB:MN1_7B=GURALP:3B93ME,EKB:MN2_7B=GURALP:3B93MF,EKB:MN3_7B=GURALP:3B93MG,=D852:EKAJ2,=D850:EKA22,=D850:EKA74,=D850:EKA84,=D850:EKA62,=D852:EKAI2,=D852:EKAI4,=D850:EKA52,=D851:EKAA4,=D850:EKA72,=D851:EKAB4,=D851:EKAE2,=D852:EKAL2,=D852:EKAM2,=D851:EKAA2,=D851:EKAD2,=D852:EKAL4,=D851:EKAF2,=D852:EKAM4,=D851:EKAG2,=D850:EKA12,=D850:EKA54,=D852:EKAO4,=D851:EKAF4,=D851:EKA92,=D852:EKAN4,=D851:EKAB2,=D850:EKA42,=D851:EKAD4,=D850:EKA24,=D850:EKA32,=D850:EKA14,=D852:EKAH2,=D851:EKAC2,=D850:EKA64,=D852:EKAO2,=D851:EKA94,=D852:EKAN2,=D850:EKA82,=D852:EKAK2,=D851:EKAG4,=D851:EKAE4,=D850:EKA34,=D852:EKAJ4,=D851:EKAC4,=D850:EKA44,=D852:EKAH4,=D852:EKAK4,=D850:EKA00_log,=D851:EKA00_log,=D852:EKA00_log,EKB:LOG=WO1242:344200_log,EKB:LOG_3B=GURALP:3B9300_log /remote/bath/seismicdata/gcftapes/EKB200009270716-200010110726/200009270716-200010110726-file1.gcf

6.3.4 “log” files

Güralp digitisers write logs in ASCII text format, usually into separate twelve-hourly files distinguished by suffix “.txt” (in the multiplexed format described in the previous section, the log files are intercalated with the seismic files). Log files are imported with bdsImportData with channel name conventionally “LOG”, with a location code if necessary, e.g. “LOG_3B”, hence the “-channels” parameter is “XXX:LOG” where XXX is the station name. Location codes should be used to distinguish between log files from different

digitisers at the same site. The “-format” is “LOG-SCREAM”. Otherwise the command is identical to the command for importing seismic data.

Metadata for log channels have to be set up beforehand unless “-ignoreMetaData” is used. There is no need for calibration, instrument or response metadata to be set, but it makes sense to set up instrument metadata showing the digitiser generating the log file and the sensor attached to it. Changes in digitiser, in particular, should be shown. The sensors and digitisers for log channels should be “shared” with the corresponding seismic channels (using the “share existing sensor/digitiser” buttons in the AdminGUI). An example of a log-file import comand is:

bdsImportData -host <host> -user <user>:<pass> -warnings -verbose -source Main -network BN -format LOG-SCREAM -channels BKN:LOG 20120507_12000.txt

6.4 SEED, miniSEED and tarred SEED

SEED is a data format for which it is not necessary to specify the channel order, i.e. no “-channels” list is required. The order given in the SEED header is considered to be trustworthy. “-synchronous” should be set for YKA SEED data, although the data are not multiplexed and strictly only the array channels, not the broadband and high-frequency channels stored in the same SEED files, are synchronously sampled. MiniSEED data are similar to SEED data and contain channel information in headers but do not contain instrument response information. YKA data after upgrade on 16th January 2014 are in miniSEED format, with the instrument information in a “dataless SEED header”. At present the dataless SEED header is updated occasionally by the YKA staff and we transfer it to a file in /sharedprograms/data named “YKA.dataless.yyyymmdd” wehre the suffix shows the creation date. The BDS does not read the header: to extract instrument response information it is necessary to run “rdseed” to read the header, then insert the information into the BDS database. A program using the BDS API to extract metadata from the dataless SEED header is under development.

“-ignoreCorruptions” and “-ignoreTimeBackwards” should generally not be set, because corruptions should be solved by pre-processing. “-reorder” and “-deleteDuplicates” should not be set, because backfilling is not expected. “-ignoreMissingBlocks” should be set, but note that missing blocks may occur for two “spurious” reasons: microsecond rounding errors in calculating block end times, for which there are no data missing in reality, and genuinely missing sets of one, two or three samples from post-upgrade YKA miniSEED files, which are restricted to channels B4, R8 and R9. These seem to be caused by a software error at YKA, and do not affect data from YKA sent to the IDC in CD-1.1 format and stored in CSS, only data in miniSEED stored locally at YKA and received by us via ftp. We believe that both errors stem from code used to convert GCF format from Güralp digitisers into miniSEED. Data with microsecond rounding errors may be imported with “-ignoreCorruptions” set, which causes the importer to round timestamps to the nearest 10 µs.

“Tarred SEED” files consist of (up to) 144 ten-minute SEED files combined into a Un*x Tape Archive (“tar”) file, representing 24 hours' worth of data. The data were originally received from Canada as separate 10-minute files and were combined into the tar files by a Blacknest-written program. For tarred SEED data the options “-sequential” should be set, as well as the above options (set or not set) for SEED data. This causes all of the 10-minute SEED files in a single daily tarfile to be combined into a single daily BDS file. Importing will fail if one of the 144 10-minute files is corrupt, at which point you have to decide whether to discard it and import all the remaining 10-minute files.

“-ignoreTarErrors” should be set for the first few files of tar SEED data, from 1991-07-10 (day 191) to 1991-09-20 (day 263), for which the tar files do not have a proper End-of-File. This does not affect the data nor the ability of the tar program to read the data, but the BDS importer fails on detecting the error message, unless “-ignoreTarErrors” is set. After that time a new version of the tar-file writer program was used, which did not have this problem, hence tar errors in subsequent tar SEED files are more serious and should be considered case-by-case.

6.5 TapeDigitiser

TapeDigitiser data are stored in subdirectories containing the data files (with names ending in “.bs”) and a file “jobInfo.tdi” (BEAM, 2008b), written during the digitising process, and optionally, a file “jobInfo.tdim”. The program bdsImportTapeDigitiserData reads the start- and end-time and data file name information from the jobInfo file, preferring “jobInfo.tdim” (“modified jobInfo.tdi”) to allow the user to supply a modified version of “jobInfo.tdi” for one that was incorrectly written during the digitising process. The channel order of the data has to be supplied by the user in the “-channels” statement and must include the two “error” channels and the VELA or other time code channel, and dummies for any dead channels, up to a total of 24 channels.

An example “-channels” parameter list is:

```
-channels
 BAW3:SHZ,BAW2:SHZ,BAW1:SHZ,BAE#:ODD_91,BAE1:SHZ,BAE#:AER_01,BAE2:SHZ,BAE3:SHZ,B
AE4:SHZ,BAE5:SHZ,BAE#:ODD_92,BAE#:AOT,BACP:SHZ,BAS1:SHZ,BAS2:SHZ,BAS3:SHZ,BAS4:S
HZ,BAE#:AER_02,BAS5:SHZ,BAWE:SHZ,BAE4:SHN,BAE4:SHE,BAE#:ODD_93,BAE#:AVT
```

The name of the subdirectory (one per run), not the files within it, is given as the parameter for bdsImportTapeDigitiserData. The options “-ignoreSessions” and “-includeSessions” allow individual data files within that directory to be ignored, or included, in the import. Each analogue tape was digitised in a single session whenever possible, with all the data going into a single output file data-000000.bs. There are more than one files for those tapes where digitisation had to be stopped in mid-tape, for instance for cleaning of tape heads. The tape was usually rewound a short distance before digitising was resumed into a new output file. Sometimes where there were problems, a number of short test sessions were recorded. The session numbers (starting at zero) correspond to the data file names e.g. data-000000.bs is from the zeroth session and data-000001.bs from the first session. Very short sessions can sometimes be ignored, especially if the jobInfo.tdi file gives zero start- and end-times for them. Substantial overlapping sessions should both be imported, unless one is known to be hopelessly corrupt, so that the end user of the data has the final choice about which to use.

6.5.1 TapeDigitiser native format

Users of TapeDigitiser data should know the following about the format. The original analogue data were stored as frequency modulations of a carrier frequency recorded on analogue magnetic tape (New, S. V., 1974, James, E. W., and Burch, R. F., 1974). Voltage values output by the seismometer/pre-amplifier are converted into deviations from a central value of the carrier frequency. Older digitisers demodulated the carrier before

converting the resulting voltage levels into digital samples, but the TapeDigitiser system digitised the carrier itself, at a sufficiently high sampling rate to capture all the frequency variations, then demodulated it in the software (BEAM, 2009) (in practice the tape play speed is 8x the recording speed, so the carrier is sampled at a correspondingly higher rate, 50 kHz in practice, BEAM, 2005). The digitiser used for this was 16-bit, but the subsequent demodulation produced a stream of 32-bit floating-point numbers with values between 1.0 and -1.0, corresponding to the maximum and minimum frequencies of the modulated carrier. On import into the BDS, the TapeDigitiser data are multiplied by 2^{24} and stored as 32-bit floating point numbers (with 8-bit exponent and 24-bit fraction) (strictly this should have been 2^{23} to create signed 24-bit values).

The analogue gain of the seismometer/pre-amplifier at each of the four arrays is different: for instance in AG note 209 (Key, F., and Marshall, P. D., 1977),

YKA 22 nm/V;

EKA 80 nm/V;

GBA, WRA 40 nm/V

(all values are displacement at 1 Hz for a single seismometer in the array). TapeDigitiser data represent the voltage output of the seismometer/pre-amplifier *scaled to fit within the bounds -2^{24} and 2^{24}*, so to get true ground velocity or displacement out of them, you have to scale them by (maximum deflection)/2^{24}. “maximum deflection” is given by “maximum voltage” x gain, where the gain is as listed above (or whatever other values it might occasionally have taken) and “maximum voltage” is the voltage corresponding to the largest allowable deviation of the frequency of the FM carrier. AG note 144 (New, S. V., 1974) gives this as 5 V, corresponding to a one-third (33.3%) deviation of the 270-Hz carrier.

Hence the calibration factors for the TapeDigitiser data for the four arrays are:

YKA 22 nm/V x 5 V / 2^{24} counts = 6.5565e-6 nm/ct;

EKA 80 nm/V x 5 V / 2^{24} counts = 2.3842e-5 nm/ct;

GBA, WRA 40 nm/V x 5 V / 2^{24} counts = 4.7684e-5 nm/ct.

The TapeDigitiser system under test (Bowers, D., Wallis, N. W., Budd, T., and Bartholomew, P., 2008) showed a 10% loss, so these calibration factors should be multiplied by 1.1, giving:

YKA 7.2122e-6 nm/ct;

EKA 2.6226e-5 nm/ct;

GBA, WRA 5.2452e-5 nm/ct

(note that the BDS stores calibration factors in m/ct so these should be multiplied by 10^-9 to match the BDS database).

The corresponding calibration factors for the various old Blacknest digitisers, in nm/ct displacement at 1 Hz, can be taken from John Young's compilation of explosion data from masterfiles. In particular for a 1977-06-30 explosion at Degelen Mountain, Soviet Kazakh test site are compared with TapeDigitiser calibration factors derived from the 1977 stated gains in AG note 209. The values for EKA and YKA are quoted in Bowers et al. 2008 but without a source; the source for these and GBA is Marshall, P. D., 1989 but for WRA the source is Burch, R., 1984. They are given below with the ratio to the TapeDigitiser values, and for comparison the ratio to TapeDigitiser values given in the final paragraph of Bowers

et al. 2008:

YKA 0.0275 (x3813) (Bowers et al. x3750)

EKA 0.0976 (x3722) (Bowers et al. x3689)

GBA 0.0488 (x9304) (Bowers et al. x3689 is in error: should be twice that i.e. x7378)

WRA -0.0261 (x4976) (Bowers et al. x3689 is in error: should be twice that i.e. x7378)

John says that WRA had two possible calibration factors, 0.0261 for “SDAT standard digital recordings” and 0.0488 (same as GBA) for “old BDRS”.

Seismometers not part of the array but recorded on the same analogue tape, e.g. broadband seismometers, had different gains, and hence need different calibration factors from these. Values for EKA and YKA velocity broadband seismometers are given in Marshall, P. D., 1989.

6.5.2 Options for bdsImportTapeDigitiser

If the timecode is VELA on track 24 then you need set no timecode options; otherwise you might need:

-timecodeChannel <channelnumber>

-timecode Hutchins (or Manual)

If the timecode is Hutchins you need also to specify:

-timecodeInvert 1 (1 inverts, 0 does not)

-timecodeStartTime yyyy-mm-ddThh:mm:ss.sss (specify to no more than THREE decimal places) because Hutchins code does not specify the year or the day. Use the official start time in the jobInfo.tdi file unless you know it's wrong.

-timecodeIgnoreTimeJumps 1 (1 enables, 0 disables) – ignoring jumps means do not try to update timecode to the new time after a jump. The usual reason for a jump is a break in recording, for instance a power cut, so you usually DO want to update the timecode after the jump.

-doNotReprocess – do not reprocess the data with program bdsTapeDigitiserFile. It is unlikely that you will want this setting unless the data are known to be corrupt and you want to import them anyway.

“-synchronous” need not be set because it is assumed as the default for TapeDigitiser data.

“-ignoreMissingBlocks” needs to be set for some TapeDigitiser data, but the reason for the missing blocks should be investigated, so a pass without “-ignoreMissingBlocks” but with “-verbose” and “-warnings” should be made first, and then the data around the reported missing blocks viewed in TapeView (BEAM, 2010e) to check for errors in reading of the timecode track. Gaps in recording of the original analogue tapes are caused by power failures and glitches, sometimes attributed to lightning storms. Apparent gaps also occur when the clock was reset forwards. The time track decoder used by bdsImportTapeDigitiserData does not directly detect the break in the time code. It

requires two full minutes of perfect timecode following a break before it can read a valid time and recognise that a break has occurred. It then flags “missing blocks”, but the nominal time of these is two minutes (or more, if the time code track is poor) after the actual break. The data can be imported with “-ignoreMissingBlocks”, and the end user is expected to read the time code and find the exact time of the break (usually evident from a spike on all channels, Figure 26).

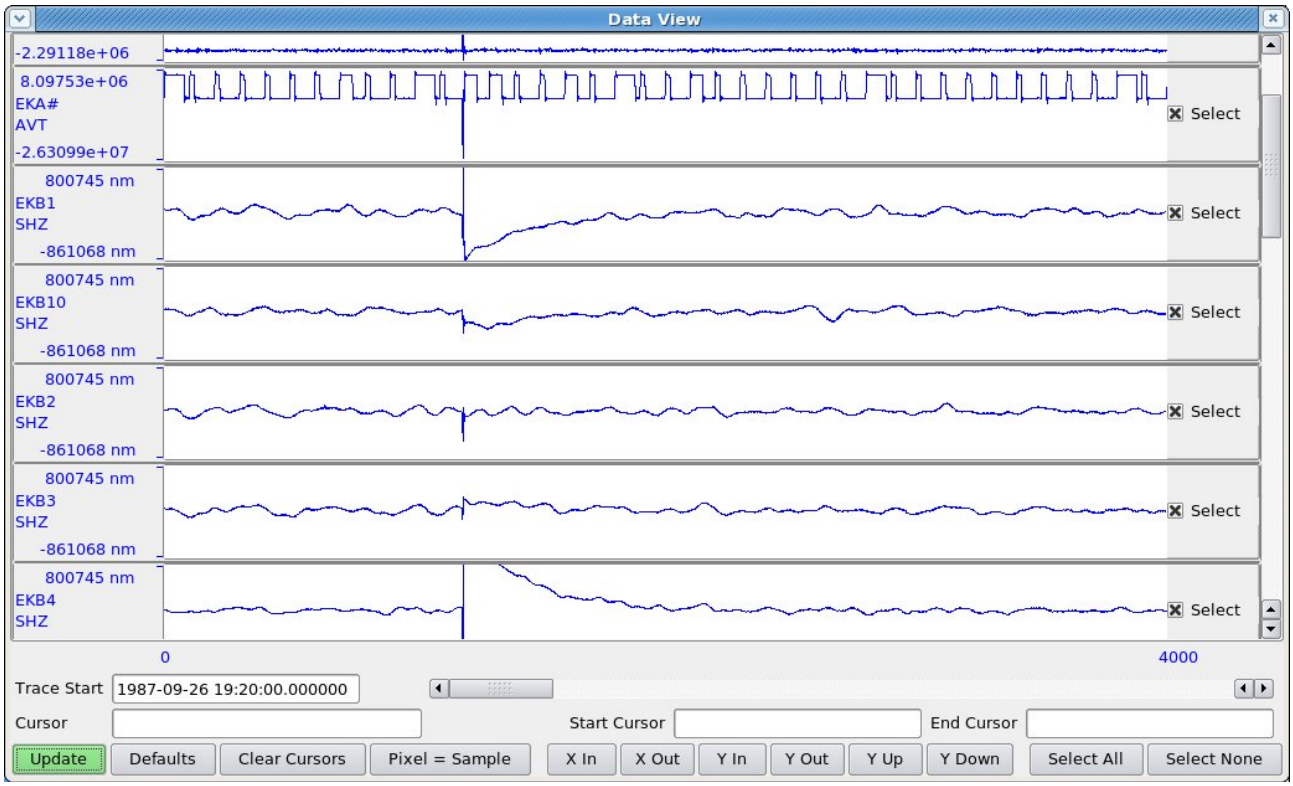


Figure 26: Trace display of break in data from TapeDigitiser tape EKA3011. 12 s of data are missing at the spike. "Missing blocks" are flagged at 19:22:32, over 2 min later, since the program has to read 2 min of time code before it recognises that a break has occurred.

A clock reset can occasionally cause a “time backwards” error. The reset is required because of (forwards) clock drift or problems with the time-code generator, particularly just after the turn of a year, where the Julian day value sometimes carries on beyond 365/366 for a day or more until it is re-set manually.

Data causing this error can be viewed with “TapeView” to decide whether the “time backwards” error is genuine, not due to the program misreading the time code. If there is a gap or disturbance in the data about two minutes before the reported “time backwards” error then the error might safely be assumed to be connected with the disturbance. The data can be imported with “-ignoreTimeBackwards”. As a check, a timespan including the “time backwards” error might be exported in IMS format (with “full blocks” ticked) and compared with the same timespan exported from “TapeView” (although the channels are in different order in the two outputs) to check that no data have been lost. The start times on the IMS WID2 lines of the two data files will not be identical even when the actual data samples are (you might need to uncompress the CM6 with “uncmpgse” to carry out a sample-by-sample comparison). This is because the TapeView output has block timestamps determined by a different time-code reading program (BEAM, 2010f).

The “-ignoreTimeBackwards” flag does not cause data from blocks with backwards timestamps to be discarded, so no data are lost on import. When exported in IMS or SEED/miniSEED format the data are split into segments with the second segment starting at the first block with a backward timestamp. Beware that if you read the IMS data into Geotool, it will “merge” the two data segments (because they have identical station and channel names), discarding the overlap. SAC, either with “readgse” or after running “gsetosac” on the IMS file, or after “rdseed” on the SEED file, will show the two overlapping segments as separate traces. The two segments are likely to have slightly different sampling rates, so if you wish to merge them into a single trace, by using “chnhdr” to change the start time of the second to make it contiguous with the end of the first, you will have to use “chnhdr” to make the sampling rates equal.

Time-backwards errors due to problems with the time-code generator … (to be written!)

An occasional error is “SampleRate Incorrect”, which might be due to tape speed errors or excessive tape stretch during analogue tape writing or digitisation. It can be bypassed with “-ignoreSampleRate”.

The jobInfo.tdi file may report “signoffComment Tracks x, y and z disabled” (where x, y and z are numbers), and you will see this in the Data Info button window under the “Sensor Data” tab in the AdminGui. The tracks on the analogue tape listed here have not been disconnected and were digitised along with the other tracks; “disabled” means merely that error-checking during digitisation was switched off for these channels. It was done usually because the channels were “flat” or otherwise obviously bad, or in mid-session because too many errors were being flagged for that channel. After digitisation, “flat” channels appear as a reversed-polarity copy of the “error compensation” channel, because the error compensation process consists of subtracting the error compensation channel from the other channels.

6.5.3 Selecting timespans of TapeDigitiser data to import

In addition to ignoring “bad” TapeDigitiser sessions with “-ignoreSessions” it is possible to import sections of a TapeDigitiser (data-00000x.bs) file that are demarcated by sample numbers. The sample numbers can be indicated by setting cursors in TapeView, a separate program, or by manually altering the jobInfo.tdi file to include the line “jobSessionx.startSample: n” and/or “jobSessionx.endSample: m” where x is the number of the file (data-00000x.bs) and n and m are the start and stop samples (n=0 means beginning of file and m=0 means end of file; otherwise m > n, of course;). In TapeView you set a start cursor (shown in green) by clicking with the left mouse button at the desired point on the traces, and a stop cursor (shown in red) with the right mouse button. The sample number at which the cursor is set is shown at the bottom of the display. Save the data (File – Save or Ctrl+E) to have the jobInfo file rewritten. TapeView saves a copy of the previous jobInfo file as “jobInfo_x.tdi” where x is the previous version number (starting from 1).

For this to work you need write permission for the file jobInfo.tdi and for its directory. The copy on the archive server is read-only, but it is possible to avoid copying the large data files by creating a directory in writeable space, into which you put a COPY of the jobInfo.tdi file from the archive server, and LINKS to the data files data-000000.bs, data-000001.bs, etc., and directing TapeView to this directory rather than the one on the archive server.

You do not have to set both cursors. Setting only the start cursor causes import to start there and end at the end of the file; setting only the end cursor causes import to start at the beginning and end at the cursor.

Once the cursors are set, bdsImportTapeDigitiserData can be invoked with the directory containing the modified jobInfo.tdi file as the data source. One option you might then need to invoke is “-timecodeStartTime yyyy-mm-ddThh:mm:ss”. This is required if the start year in the jobInfo.tdi file is not the correct year for the data demarcated by the cursors. The time decoder always uses the year from the jobInfo file and does not attempt to decode year numbers (the last digit of the year) that are encoded in VELA timecode, because for many tapes these are blank, and for others they are wrong.

Particular examples where cursors are needed for import are where: (a) data on a tape have overwritten older data, and some of the older data remain at the end; (b) the timecode switches from one type to another in mid-tape (Tape WOL123, Hutchins to VELA); (c) data have “New Year” timecode problems; (d) a long stretch of particularly noisy traces not containing any “real data” occurs with good data on either side.

Beware that the Hutchins time code is not always reliable in producing the correct day (Hutchins has day-of-month encoded but this is often blank), and so the BDS ignores the day values in Hutchins code. It may then be necessary to set a “cursor start time” in the jobInfo.tdi file (new feature post v. 2.0.37). This is the time corresponding to the start cursor position in samples. If you do not do this then the day encoded into the BDS header of the imported data is the day retrieved from block time stamps, which may or may not be correct.

Once the data have been successfully imported, ensure that the jobInfo.tdi file used for the importing is preserved.

6.5.4 2x12-track tapes

A small number of tapes, e.g. those with numbers beginning “EL”, which recorded data from instruments in and around the vault at Eskdalemuir, were recorded in two sessions, with twelve channels of data per session. The tape was rewound between sessions. Timecode for the first session is on channel 12, for the second, on channel 24. They have been digitised all 24 channels in a single pass. BdsImportTapeDigitiserData can at present (v. 2.0.23) import only all 24 channels at once, so it is necessary to import each TapeDigitiser file twice, once with “-timecodeChannel” = 12, once with “-timecodeChannel” = 24, and the end user has just to ignore the twelve channels from the “other” pass, for which of course the data are wrongly timestamped. A version that imports only twelve channels at once is awaited.

6.6 AD_22_YKA data

Five months of data from YKA between 1988-12-31 and 1989-05-19 were digitised on the Blacknest equipment in 1998 by Norma Wallis and added to the “Robby” AutoDRM, where they are referred to as “ANALOG_22”. Entries were made for these data in the 2006 AutoDRM database, with the value of “format” set to “AD_22_YKA”, but software to read them was never incorporated into the 2006 AutoDRM.

22 channels, omitting the two “error” channels, were digitised, nominally at 20 Hz, with the timecode on the last channel. These channels were: 1-8 – array red arm 1-9 omitting R8 and R0; 9 – VBB seismometer; 10,11 – horizontal SP seismometers; 12-21 – array blue arm 1-9 then 0; 22 – timecode.

The “Robby” AutoDRM crudely read the timecode to find the data samples corresponding to the user's requested timespan. The BDS does similarly. The original analogue tapes have been digitised in the TapeDigitiser project, so some of these data (all but four analogue tapes' worth) are available in both formats. Furthermore, those short stretches of data that had particular interest at the time, e.g. announced nuclear tests, were digitised separately and put into the “masterfiles” collection. We found only one file, in the end, of data obtained by “Robby” from the AD_22_YKA files:

890309_0255.YKAASfromRobby (5 minutes long)

890310_2208.YKAASfromRobby (5 minutes long)

and the originating AD_22_YKA files are:

1989067_141100.YKA

1989069_173800.YKA

The AD_22_YKA format consists of a 44-byte ASCII header then 22 channels of per-sample-multiplexed 16-bit two's complement big-endian integer data values. They are not a multiple of 44 bytes long, though, because they end in mid-sample, possibly because the digitiser's buffer size was 16000 samples, which is not divisible by 44. The ASCII header includes a rough start time, based on timing the tape winding to the beginning of digitisation, and accurate only to about the nearest minute.

The creation process of the AD_22_YKA data had three stages:

1. Digitisation on a dedicated computer;

2. Transfer to Vax computer, byte reversal to big-endian and multiplication of all samples by two with program “TIMES2”, which says it also “replaces the first 44 bytes of the first data block, i.e. the first time sample of each channel, with an ASCII header string”. The factor of two was deemed necessary because the digitising system used had a response equal to half the response of another digitising system in common use at the time. The files were renamed to the conventional name containing the start time;

3. Transfer to Exabyte tape.

An example of the import command, showing the channel assignment, is:

/usr/bds/bin/bdsImportData -host mortlake -user testAdd:beam00 -network BN -arra y YKA -source Main -synchronous -allowOverlap -ignoreCorruptions -ignoreMissi ngBlocks -warnings -verbose -format AD22 -channels 'YKR1:SHZ,YKR2:SHZ,YKR3:SHZ,YKR4:SHZ,YKR5:SHZ,YKR6:SHZ,YKR7:SHZ,

YKR9:SHZ,YKC:BHZ_70,YKC:SHN_70,YKC:SHE_70,YKB1:SHZ,YKB2:SHZ,

YKB3:SHZ,YKB4:SHZ,YKB5:SHZ,YKB6:SHZ,YKB7:SHZ,YKB8:SHZ,YKB9:SHZ,

YK B0:SHZ,YKA#:AVT' /remote/archive/1989/YKA/Array/SHZ/1989078_054500.YKA

(this particular file gave the timecode reader problems that were not fixed until BdsServer v. 2.0.24).

6.7 CD-1.0 and CD-1.1 data

Continuously streamed data from International Monitoring System stations are available from the IDC in Vienna in CD-1.0 format, for older stations, and CD-1.1 format, for newer stations. These are described in IDC 2002a and IDC 2002b. The data are received by the IDC program “cdrcv” which is part of “cdtools”, provided in the “NDC-in-a-box” package. It writes the data from the input streams into 24-hour-long (ish) files, usually starting at about 23:59 on the previous day, with names in form SSSS.YYYYMMDD.HHMM.10bin (for CD-1.0 data; .11bin for CD-1.1 data), where DD, HH and MM usually represent the midnight immediately after the actual start of the data at 23:59ish (so HH and MM are usually zero). SSSS is the station name, e.g. “I52GB”. The file contains all the data channels, including auxiliary data sampled at different rates from the seismic data. For infrasound stations these include temperature, wind direction and speed, and absolute pressure, and for hydroacoustic stations, electrical current and voltage sensors.

During normal operation the data streams are transferred from Vienna as they come, and the data blocks or “frames” within them might not be in chronological order. After any interruption in communication between the station and Vienna, backfilling takes place, and backfilled data are added to the CD-1.0 and CD-1.1 files as often as needed during the days following their original creation. As an aside, “cdtools” also includes the program “cd2wng”, which reads the CD-1.0 or CD-1.1 files and converts them into CSS-format files. It can be set up to run continuously alongside “cdrcv”, in which case it will also backfill the CSS-format files that it has created. At present it is not necessary to run “cd2wng” except for test comparisons, because the BDS works directly on the CD-1.0 or CD-1.1 files.

The BDS continuous importer of CD-1.0 or CD-1.1 files, bdsImportCd, does NOT backfill, so it is set to run 24 hours after real time to avoid importing incomplete files that might be backfilled later. One-off runs of bdsImportData to import CD-1.0 or CD-1.1 data should also not be run too soon after “real time”; or the imported data should be deleted and re-imported after any backfilling is complete.

CD-1.0 and CD-1.1 data contain channel information, so it is not necessary to set “-channels” when using bdsImportData. A typical command is:

bdsImportData -host <host> -user <user:pass> -warnings -verbose -network BN -array I52 -source Main -ignoreMissingBlocks -deleteDuplicates -reorder -ignoreTimeBackwards

-allowOverlap -ignoreCorruptions -synchronous -format CD1.1 <filename>

As with GCF, -reorder and -deleteDuplicates allow for backfilled blocks. -allowOverlap is required otherwise you get “data already present” errors if the file following the one you are importing has already been imported. There are often missing blocks even after backfill is complete. “Time backwards” errors may occur if the station clock is corrected during the data timespan. “-ignoreCorruptions” is needed for some data in which blocks overlap (these otherwise give the error message “Excessive authSize in packet”). Where the data are from an array, the “-synchronous” and “-array” switches should be set, and the array set up as such in the database, e.g. with bdsMetadata commands such as:

add Station $I52G {"I52GB",,"array","Diego Garcia"}
{"I52H1","BDF","I52H2","BDF","I52H3","BDF","I52H4","BDF","I52H5","BDF","I52H6","BDF"
,"I52H7","BDF"};

add Location $I52Gl {$I52startdate,
$endoftime,"BN","I52GB","WGS84",72.48416,-7.37781,1.000,,};

where the “location” is the centre of the array).

A hazard of importing CD-1.0 or CD-1.1 data is that the channel name set up in the data blocks might include a location code, which is never reported in the IDC database and changes without notice when equipment is upgraded. bdsImportData will then give metadata errors unless “-ignoreMetaData” is set. bdsImportCd runs as if “-ignoreMetaData” had been set, but then you cannot view or export the data until new channels incorporating the changed location codes have been created in the BDS database. bdsImportData error/warning messages will not tell you any channels other than the first encountered with a changed location code. So far I have found two ways of finding out which channels have changed location codes: importing a sample file (with “-ignoreMetaData” set and not as a “dry run”) then viewing the channels in the “Data Files” tab of the AdminGUI; and running “bdsDataFileDump” with the “-info” option and inspecting the “stderr” output for lines beginning “channeln.channel” (where “n” is the channel number). The latter is easier: “bdsDataFileDump -h” gives you a list of options for this command, which is otherwise undocumented.

Known problems with location codes are:

I52 – all channels gained location code “01” at upgrade on 2013-03-27;

H09W – all channels gained location code “ “ (yes, a single space! And yes, the BDS needs to have it, and not two spaces!) at upgrade on 2011-10-14 and the IDC was unable to remove it (I asked them and they tried);

I51 elemet H4 had a location code “NO” until 2011-09-20, when the IDC removed it at my request;

H08, H10, H11 – all channels have location code “US” (H11 Wake Island does belong to the USA, but I don't know whether that's the reason for the location code).

6.7.1 Sampling rates of hydroacoustic data

The hydrophones and digitisers are under water, so they are not synchronised with GPS or similar radio-transmitted time signals. The internal clocks used to drive the digitisers may therefore be out of synchronisation, fast or slow. A 2005 email to Neil Selby from the IDC states:

“Our understanding is that the sample rate contained in the hydroacoustic data files is a derived quantity, calculated by dividing the number of samples, 2500, by the length of each data segment, ~10 seconds. The length of each data segment is calculated from the difference of first data sample's time stamp, and the 2500th data sample's time stamp. The time stamps are generated by a TCXO crystal clock, which has a known linear drift rate, and is a floating time base. Although the clock drifts, the software compensates for that. ”

The clock time is resynchronised, apparently four times a day (so Terry discovered by dumping headers of a CD-1.1 file). The NDC-in-a-box software “cd2wng” (or the equivalent running at the IDC) smooths out the resulting jumps in time. The BDS, importing CD-1.x files, reads the block start times and number-of-samples, and works out a sampling rate for each block - ??

6.8 CSS data

We receive CSS data from the IDC in response to special requests. Other sources of CSS data are “cd2wng” (part of the “CDtools” package supplied with the NDC-in-a-box), which produces CSS from the continuously arriving CD-1.x data on the VPN from the IDC, and “conv”, which is capable of converting IMS or BKNAS data into CSS and has been used routinely to write CSS files into “John's CDs” for external distribution.

“bdsImportData” imports CSS when the keyword “-format CSS” is given and the “wfdisc” file name is given as the input filename. “bdsImportData” looks in the directory name given in the “wfdisc” file for the files named, so it is necessary to edit the “wfdisc” file if the directory name is not correct (data from the IDC, for instance, usually have the full absolute pathname on the IDC system as the directory name). The directory name “./” can be used rather than a blank, if the data files are in the same directory as the “wfdisc” file, but if the directory name is blank, then “./” i.e. current directory, is assumed.

The data format may be s3, s4 and t4 i.e. big-endian (s) or little-endian (t) 3- or 4-byte (24- or 32-bit) numbers. Beware that the IDC database schema v. 5.1.1rev5, October 2002, gives an out-of-date list of recognised formats that does not include “s3” or “t3” (24-byte formats), which are used by the IDC.

The most important problem with CSS is that the format does not support separate location codes. There is enough space in the “channel” slot (8 characters) to store a location code in addition to the channel code, but this is not used because the IDC does not use location codes. Even when data in CD-1.1, which does support location codes, contain them (e.g. H08 and H11 which have location codes “US”) these are not written into the CSS file by “cd2wng”. Hence import of a CSS file will fail with metadata errors if the resident metadata in the BDS system contain the location codes (which they will need to do for successful import of the corresponding CD-1.1 data). If the import is forced with “-ignoreMetaData” then the imported data cannot be exported. To overcome this it is necessary to use the switch “-channelRename <channel-in-CSS>=<channel-in-database>” e.g. “-channelRename EDH=EDH_US,LEA=LEA_US,LEV=LEV_US” for a hydroacoustic station.

The BDS is not capable of writing CSS data. If you want CSS data output then export the

data as IMS then use “geotool”, “conv” or “SAC”, all of which are nominally capable of reading IMS and writing CSS data.

7 Adding notes and warnings to data

Notes and warnings can be attached to data in the BDS system by three methods: the “-addWarning” option of “bdsImportData” and “bdsImportTapeDigitiserData” programs, adding a note via the Notes tab of the AdminGui, and with the program “bdsNoteAppend”. Only admin-accredited users can add notes and warnings, but a non-admin user may view them in the “Notes” tab of the User GUI, or the “Data Notes button at the bottom of the “Sensor Data” tab.

7.1 “-addWarning” option

“bdsImportData” and “bdsImportTapeDigitiserData” automatically add warnings to the data when they encounter issues during import, including missing blocks, re-ordered blocks and incomplete metadata. The “-addWarning” option allows user-specified warnings to be attached to the data in the same way as the automatic warnings.

The format of the “-addWarning” option is “errorno,errorstring,fileName,startTime,endTime,network:station:channel:source,description,

e.g. `-addWarning 99,earthquake,/remote/archive/2010/EKA/BHZ/EKR1/20100313_1500z.gcf, 2010-03-13T15:13:00.0,2010-03-13T15:14:00.0,BN:EKR1:BHZ:Main,”earthquake M6.9 offshore Camelot”`

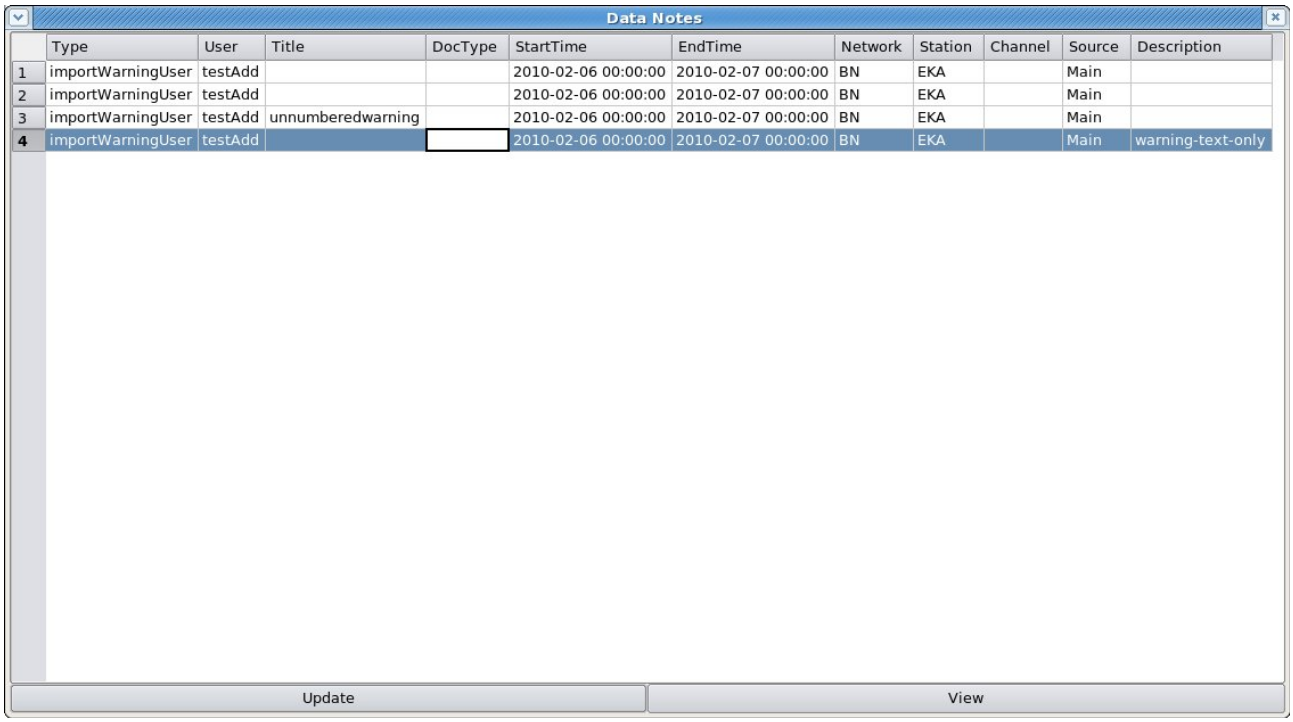


Figure 27: "Data Notes" window showing entries created by "-addWarning" option of bdsImportData. The parameters given to "-addWarning" were, for 1., none (six commas); for 2., an error number only; for 3., an "errorstring" only (reproduced here as "Title"), and for 4., a description only. The error number is not shown in this window.

There is at present very little error checking on any of this input, so it is up to the user to ensure that the filename agrees with one of the filenames being imported, the timespan delimited by “startTime” and “endTime” lies within the span of the data file, and the channel specified is in the data file. The resulting warning is stored in two places: the header of the BDS-format data file in the BDS store, and an entry in the “Notes” database. Only the latter can be altered after import.

All of the above parameters can be omitted. If the error number is omitted it is set to zero. If the “network:station:channel:source” is omitted then the warning is taken to apply to all stations being imported and will be stored in the “Notes” table with the array name as the “station name” and a null “channel name”. If the times are omitted then the warning is taken to apply to the entire timespan of the imported file.

The resulting warnings can be viewed in the AdminGui or UserGui by pressing the “Data Notes” or “Data Info” button on the “Sensor Data” tab, or in the “Notes” tab, once the station and a timespan encompassed by the imported file have been entered at the top of the tab. The “Data Notes” button and “Notes” tab show some entries from the “Notes” database table (Figure 27), including all the input parameters except the error number. The window from the “Data Info” button shows all the details including the error number.

A note for shell-scripters. The BDS expects a single string following the “-addWarning” flag, so quotes as delimiters for the entire string are necessary, and further quotes to delimit the warning itself if this contains spaces. Imports carried out by shellscripts in which the import command is packed into a single variable, e.g. (in bash) “command=bdsImportData -host ...” etc. with the entire option string including the “-addWarning” option, and list of filenames, followed by “$command” or “time $command” to execute the import, will fail if the warning message contains spaces. This is a shell problem, well known, but none of the “fixes” I have found online works (e.g. using backslashes, single quotes, doubled-up quotes). In the end I just replaced the spaces with hyphens or underscores.

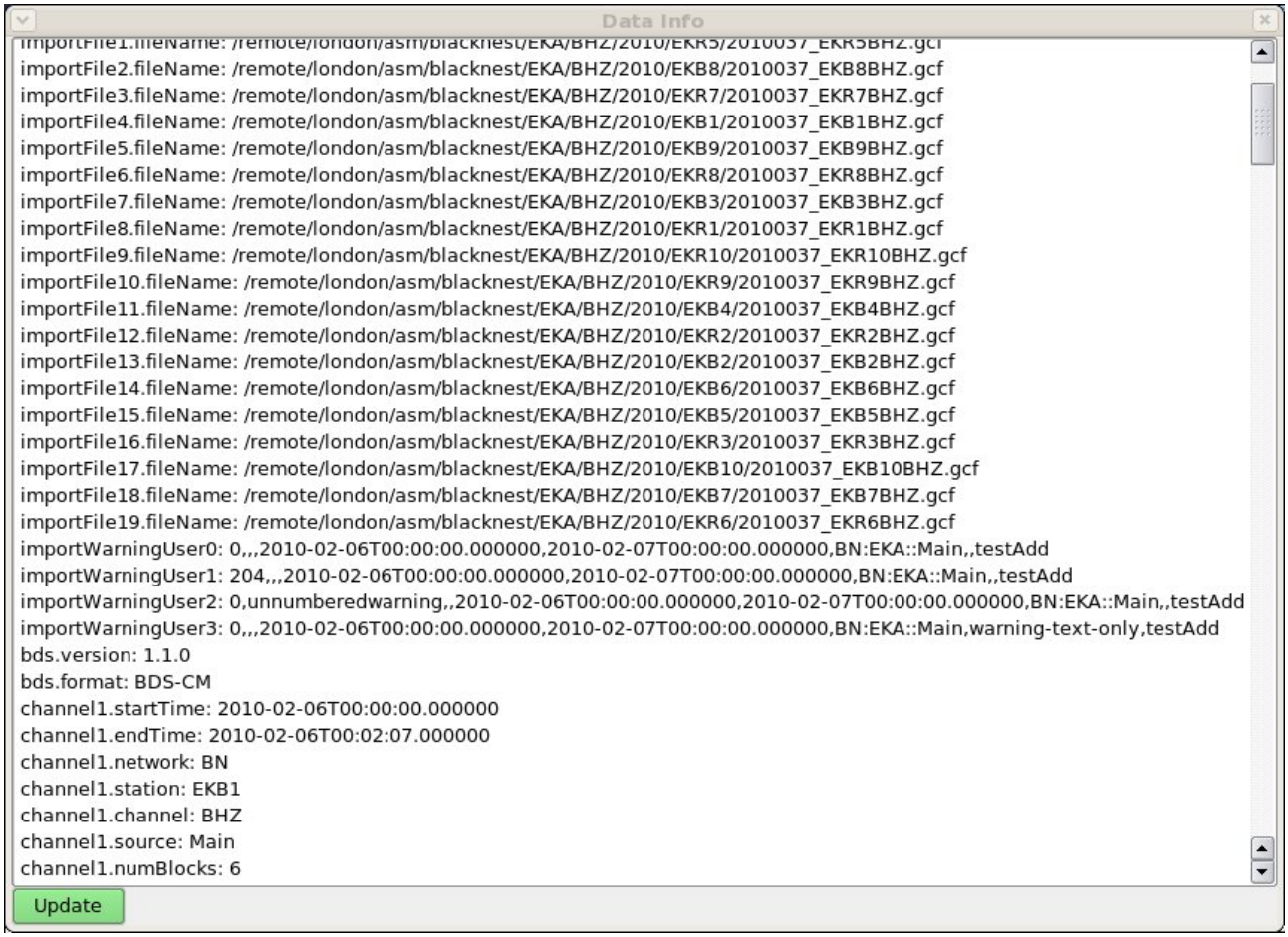


Figure 27: Data Info window for the same four warnings added with "-addWarning" as for the previous figure. The parameters are in the order in which they were specified in the input command (see text). The error number is shown here (default zero because it was unspecified for all except the second warning). "testAdd" at the end is the name of the user that did the importing.

While at present there is no checking of the error number, there is a list in the BDS User Manual (BEAM, 2010a) from which the appropriate error number should be drawn.

7.2 Adding note via Notes tab

In the “Notes” tab a note can be added by pressing the “Append” button at the bottom of the window. The default start time and end time of the new note (Figure 29) are set to zero and infinity and are not affected by the start time and end time entered at the top of the “Notes” tab. The “type” of the note can be chosen from a drop-down list: “note”, “warning”, “doc”, “importWarning”, “importWarningUser” and “exportWarning”. Type “doc” is for a note referring to a separate document filed with the data: this document can be in any format, for instance a “pdf” file containing a scanned logbook.

Because there is no checking of the consistency of items in a “Note”, it is possible to create a “Note” for non-existent or yet-to-be-loaded data. When data are present for the timespan and station/channel specified, and the timespan and station are specified in the “Sensor Data” tab, the “Note” will be found by pressing the “Data Notes” button. If you leave the station and channel empty and the dates at the default “zero” and “infinity”, then a note is created with a blank station and/or channel, and cannot be associated with any

data.

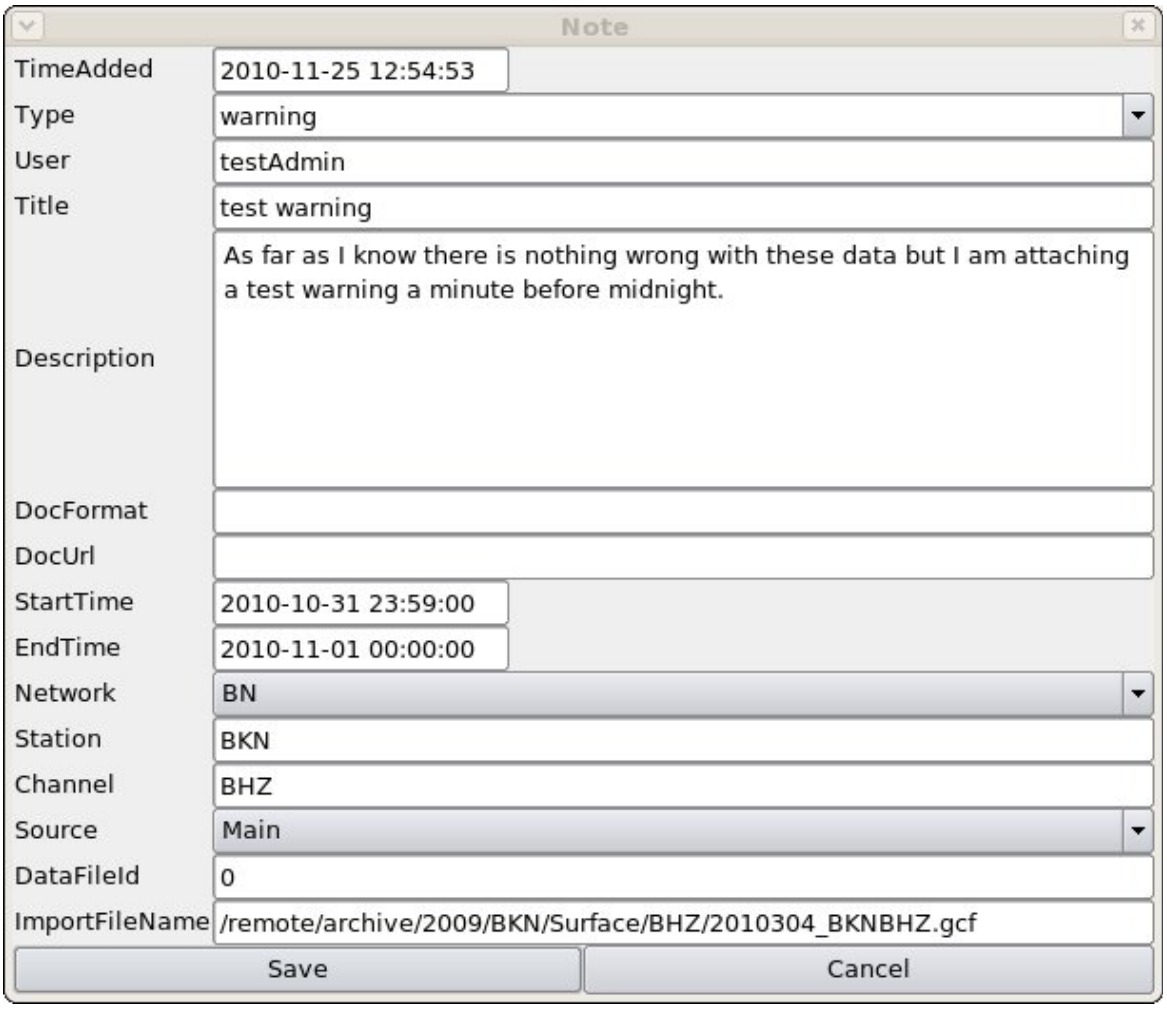


Figure 29: Note edit window. The "StartTime" and "EndTime" default to infinity and should be set by the user; so should the ImportFileName, but there is no consistency check between them nor with the station/channel/source specified.

7.3 bdsNoteAppend

bdsNoteAppend allows you to create a note with (optionally) a reference to a separate document, of which a copy is loaded into the data store. The format of the command is:

```
bdsNoteAppend [options] note-type title
```

where “note-type” is “note”, “warning” or “doc” - use “doc” for importing documents. More-or-less it takes as options the same items as the Note edit window in Figure 23: start time, end time, network, station, channel, source, description, the format of the document to be imported and the path to the document. For example:

```
bdsNoteAppend -host bds -user <user>:<password> -description tape\ logs
-startTime 1977-03-23T00:00:00 -endTime 1990-09-08T00:00:00 -network BN -station
EKA -channel SHZ -source TapeDigitiser -file
/shareddata/documents/Digitisation/Tape_Logs/Batch\ 24/Batch_24_EKA.pdf -format
pdf doc EKAlogsheets1977-1990
```

Again, some or all of the options can be left blank. Any file specified with “-file” must exist. The “note-type” can be any word, not just “note”, “warning” or “doc”, but applications that search for particular types of “note-type” might then not find it. The “format” can also be anything, but it is appended on to the filename of the file stored in the BDS data store (the filename might be in the form “note-0000000259.pdf”, for instance), so if the file is of a well-used format identified by a suffix, e.g. “doc”, “pdf”, “jpg”, “xls” or “txt”, then it would be as well to use the correct “-format” so that the document will be opened by the correct program by default.

8 Deleting data from BDS filestore

Deleting a data file from the BDS store may be necessary if the data were imported with the wrong station order, sampling rate or other parameters set, or if the import was interrupted by system failure. In that case the partly imported file appears in the list in the GUI “Files” tab as having status “importing”, and an attempt to re-import it will fail with “data already present”.

The bdsDataManage program can be used only by users with dataDelete privileges, and has the format, for deleting, of:

bdsDataManage -host xxx -user uuu:ppp -dryRun (or -noDryRun) -startTime yyyy-mm-ddThh:mm:ss -endTime yyyy-mm-ddThh:mm:ss -select Net:Sta:Chan:Source delete”

where the timespan encompassed by -startTime and -endTime covers the file or files to be deleted but need not be exactly at the boundaries of these files, provided it does not include time from any neighbouring file that you don't want to delete. The station specified by the “-select” command must be one included in the file to be deleted, but it is not necessary to specify all the stations in the file for the whole file to be deleted.

Hence there is some scope to delete files accidentally, and it is wise to use “-dryRun” to check that only the desired file is deleted.

At present the file is not deleted but moved to a “Deleted” subdirectory within the data store, and at v. 2.0.36 there is no way within the BDS to “empty the trash” in this subdirectory, so that has to be done by an administrator using Unix commands. The database entries associating the file with metadata are, though, deleted, so it would be necessary to restore these from a database backup (if one exists) before the “deleted” file could be restored.

9 User administration

The user's group membership determines the level of the user's access. Only users in the “admin” group can use the AdminGui, but further group memberships are required to carry out a number of tasks offered by the AdminGui. Users in the “userAdmin” group can modify the privileges of themselves and other users. “dataDelete” membership is required to delete data files and metadata. Loading of data can be done without “admin” membership for those with “dataAdd” membership. Modification of metadata (other than that done automatically when data are loaded) requires “admin” membership. “data.Blacknest” and “dataModify” are not at present used. A user to whom only AutoDRM access is to be granted should be excluded from all the access groups but must have a name registered in the list of users, with the exact email address from which AutoDRM requests will originate.

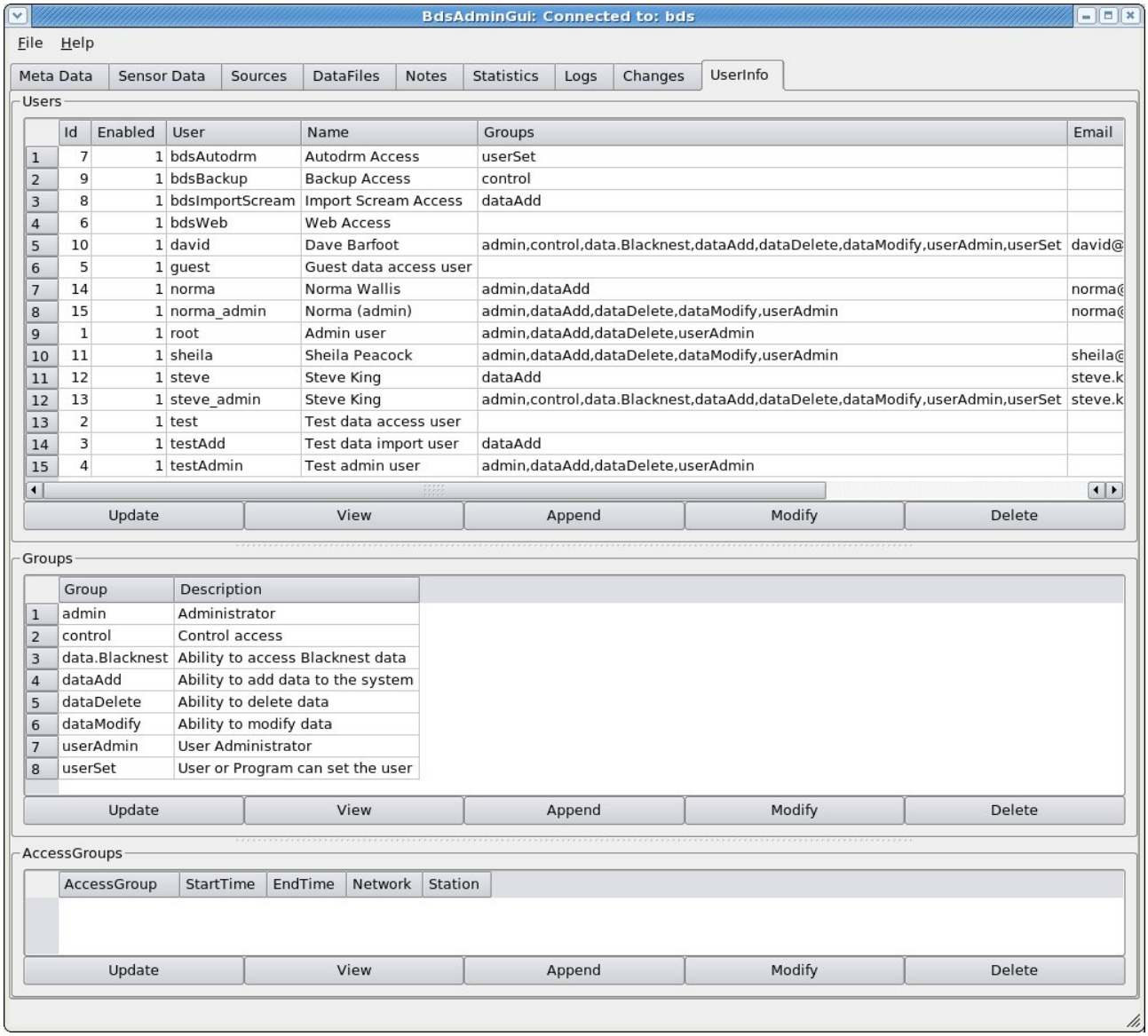


Figure 30: UserInfo tab showing list of users with the groups to which they belong and the list of groups.

9.1 8.1 Access Groups – restricting access to data

"AccessGroups" are used to control access to particular data sets, specified by network, station, start and end times. Only a user belonging to the required Access Group (or groups) can gain access to the restricted data or metadata. To restrict access to a particular dataset using an Access Group,

1. Create a new group by pressing "Append" below the "Groups" subwindow of the UserInfo tab (Figure 30);
2. For each user to whom access to these data is to be granted, select their line from the Users list, press "Modify" and, in the User Edit window, tick the box next to the new group name;
3. In the Access Groups window, press "Append". In the Access Group Edit window (Figure 31), select your newly created group from the dropdown list of groups, then

set the start time, end time, network and station (or array) of the restricted data. Leave the dates as 0001-01-01 and 9999-01-01 to restrict access to all data from the station;

4. Test the AccessGroup by attempting to download data or response metadata from the restricted station in the “Sensor Data” tab. You will not be able to see the station name at all if the start and end dates are 0001-01-01 and 9999-01-01. If the start or end date is not either of these, then you will be able to see the station name but if you set a date/time span within the restricted range then you will get an empty channel list when you press “Update”. If you then press the “Responses” buttons you will get the error message “error – end time before or equal to start time” (even though this is not so). Data files from within the restricted time range cannot be listed in the “Data Files” tab, and in the AdminGui, a user with “dataDelete” privileges will have to be given membership of the group featured in the AccessGroup before they can see and delete data files from within the restricted time range.
5. If you later wish to modify the AccessGroup, then you will need to assert that your (admin-privileged) user name by which you accessed the AdminGui is a user allowed to access the data covered by the AccessGroup (do this by choosing your user name from the “Users” list, pressing “Modify” and ticking the box to the left of the group name privileged to access the data). Otherwise your unprivileged admin user will not be allowed to see the station name in the dropdown list proffered by the “Modify Access Group” popup window.

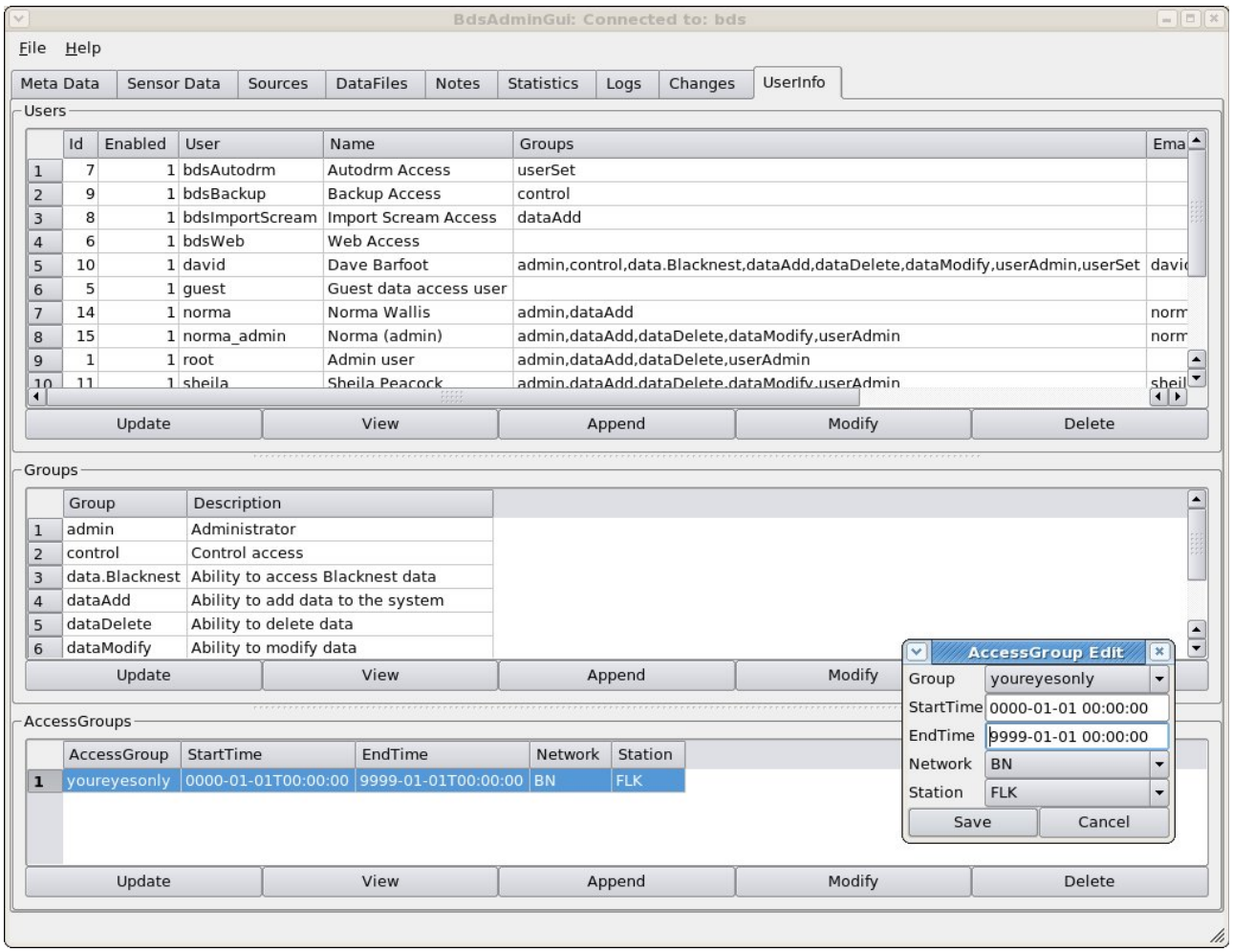


Figure 31: UserInfo tab showing AccessGroup Edit window. To prevent access to data from a particular station for anyone not in the "AccessGroup", fill in this window with the AccessGroup name and the start time, end time, network and station to which access is to be restricted. In this example station FLK for all time is available to group “youreyesonly”.

It is possible to include a station in more than one Access Group, in which case users must be members of both Access Groups to have access to it. This can be used to restrict access to different timespans of data from the same station to different users. For instance an Access Group can be set up to have access to the entire data holding for one station, and a separate Access Group to have access to a short timespan of data from that station. Users must be in both access groups to access this short timespan. Users in the first access group but not also in the second will not be able to access the data from the short timespan, although they will be able to access all the other data from the station. Users in the second access group but not the first will not be able to access any data at all from the station.

The Access Group also restricts access to data via the command-line program “bdsDataAccess”: the “-command” options “dataInfo”, “dataInfoFull”, “channelInfo”, “channelInfoFull”, “responses”, “dataGet”, “dataGetFormatted” and “dataPlot” are all disabled if the user is not in the appropriate Access Group for the data specified in the “-select”, “-startTime” and “-endTime” parameters. The other options, including “dataSearch”, still work.

Users not in the Access Group for a station will not be able to obtain either data or

responses for that station from the AutoDRM.

10 bdsMetadata command-line interface to database

10.1 Introduction

bdsMetadata is a program for querying and modifying the station metadata in the BDS database. It allows you to change several database items at once, and is thus more convenient than the AdminGUI for multiple changes, deletions or additions. Also, the commands for bdsMetadata can be preserved in scripts, for repeat running and reference. The definitive reference for bdsMetadata is the BEAM manual (BEAM, 2011a).

The basis of bdsMetadata is the BDS API set for metadata handling, NOT the underlying database and SQL. Hence the object classes, not the database tables directly, are addressed and modified. For a user used to SQL, this is disconcerting. It is wise to run the bdsMetadata command “info <*object*>” on each of the object classes used by bdsMetadata and keep these handy while writing bdsMetadata programs. See Appendix C – class list of classes used by bdsMetadata (listed in include/BdsD.h) for the full list.

Beware, particularly, that bdsMetadata treats a “start time” or “end time” in a search pattern as an instant (or, when you specify both start and end times, as a timespan) at which a set of metadata is valid, NOT as a pattern to be matched by start- or end-times in the database. So if you specify 'startTime=”2009-01-18T00:00:00.000000”' it will return all metadata with endTime greater than 2009-01-18, not just the metadata that happen to have startTime exactly 2009-01-18 00:00.000000. In fact it interprets “startTime=XXdate” as “endTime > XXdate OR endTime < 0000-01-01T00:00:00”.

10.2 Invocation

The command line for bdsMetadata for normal use is:

bdsMetadata -host <hostname> -name <bdsinstancename> -user <user:password>

and optionally, -maxChanges <n>, -v (for verbose), -d <variable>=<value>

then finally either -c '<command>' or -f <filename>

“host” and “name” should both be specified, in case there is more than one instance of bdsServer running on the host. The “name” can be seen at the top of the AdminGUI frame.

maxChanges is the maximum number of times a command will be executed if it changes the database. It defaults to one, and bdsMetadata will not execute the command at all if more than one execution would occur. This is a safety feature, very helpful to novice users given the ease of misinterpretation of date-matching described above. maxChanges doesn't quite equate to the number of table rows that would be changed (because of one-to-many relations between tables), but approximately to the number of channels that would be modified by the command. Set maxChanges to the number of channels you expect to change (or delete). You can also set it within a bdsMetadata script with “set $maxChanges=<n>;”

The command introduced by “-c” should be between single quotes because most of the command items include variable names, which in bdsMetadata are preceded by $ signs that have to be hidden from the shell by the single quotes. Commands in a file need not

have quotes. All commands are terminated by a semicolon, as in “C”-family languages, so multiple commands can be placed on one line.

10.3 Commands

10.3.1 Backup and restore

backup $<reference> and restore $<reference> back up and restore, respectively, the database. The “reference” should be a variable name preceded by $, e.g. $fridaybackup.

10.3.2 Variables

Variables are always preceded by “$”, as in Perl (and different from csh and bash). “set $<variable>” and “print $<variable>” let you set and view the value of a variable. “clear” clears the values of all variables. Think of variables as being C++ variables of various types; so you can do:

set $station = “H09N1”;

which is a simple character variable, or

add PoleZero $pz0 {values of poles}{values of zeros}; (see below for details)

which creates a variable $pz0 of type PoleZero (or an object called $pz0 of class PoleZero, if you prefer).

When you “print” a simple variable, it will print the value, but if you try to “print” a derived variable like $pz0, you will get a blank or zero. The “find” command initiates and fills an object of the named type, by searching the database according to the given “pattern” and assigns it a variable name: if you “print” that, it will print the “id” number of the relevant row in the database, e.g.

-c 'find Channel $ctest {station="EKB1", channel="BHZ", startTime="2011-07-31T00:00:00.000000"}; print $ctest;'

returns the number 241, which is the “id” of the row in BDS.Channels (the query below, which hinges on “endTime”, is equivalent to bdsMetadata's interpretation of the specified startTime, mentioned above).

mysql> select * from BDS.Channels where (station="EKB1" and channel = "BHZ" and endTime > "2011-07-31T00:00:00.000000");

```
+-----+-------------------------+-------------------------+---------+---------+---------+-------------+------------+----------+-------------+---------------------+
| id  | startTime               | endTime                 | network | station | channel | channelType | channelAux | dataType | description | lastUpdate          |
+-----+-------------------------+-------------------------+---------+---------+---------+-------------+------------+----------+-------------+---------------------+
| 241 | 2008-12-02T00:00:00.000000 | 9999-01-01T00:00:00.000000 | BN      | EKB1    | BHZ     | BHZ         |            | seismic  |             | 2011-02-23 12:18:18 |
+-----+-------------------------+-------------------------+---------+---------+---------+-------------+------------+----------+-------------+---------------------+
1 row in set (0.00 sec)
```

When commands are introduced by “-c” on the command line, variable-setting commands have to be chained on to action commands that use the variables, because variable names are not preserved between invocations of bdsMetadata. For example, “-c 'set $station=”EKR1”; set $channel=”BHZ”; add Channel $c {};'” creates a Channel object with station and channel names as set and all the other items as default.

Variable names are of two types: names of elements within one of the objects, such as “$station” and “$channel”, and user-specified names, such as $myvariable and $x. See “Object types” and “Defaults” below.

You cannot set one variable equal to another, e.g. set $startTime1=$endTime2 – it simply sets the variable to zero.

10.3.3 Object types recognised by bdsMetadata

The object types recognised by bdsMetadata are:

Network, Station, Location, Channel, Sensor, Digitiser, ChannelInstrument, PoleZero, FAP, FIR, Calibration, Response, Source.

These correspond with database tables except that “Location” corresponds to the table StationLocations, and PoleZero, FAP and FIR are is required to allow the user to input values into the “blob” within table “Responses” in which the actual values of instrument responses are stored.

The command “info <object>” returns a list of the elements of each object, e.g.

-c 'info Channel;'

#Channel Members

startTime

endTime

network

station

channel

channelType

channelAux

dataType

description

You need these names to access the elements, either in a pattern-match (e.g. startTime = “1998-05-01T12:30:00.000000”) or to specify values, in which case the values can be listed in the order in which the “info” command gives them, comma-separated, without needing to name the elements, e.g.

-c 'add Channel {“2011-08-17T00:00:00.000000”,,”BN”,”UGG”,”BHZ_20”,,,”seismic”, ”UnderGroundGrotto Z”};'

Omitted items are set to defaults.

When specifying a channel with a location code, specify it as <channel>_<location-code> as shown above (“BHZ_20”) and leave “channelType” and “channelAux” as default.

The order of items in the object is not always the same as their order in the corresponding database table (as given by the SQL command “describe”), and some of the database table items are missing from the object, in particular, “lastUpdate” and “id”.

See Appendix C – class list of classes used by bdsMetadata (listed in include/BdsD.h) for

the full list.

10.3.4 Regular Expressions

In those bdsMetadata commands that seek for an item or items of existing metadata, a “pattern” must be specified for matching key elements in these items (e.g. the station name). This corresponds more-or-less to the “where” clause of a “select” command in SQL. The “pattern” can include regular expressions: these are interpreted by the BDS API, but where they are converted into SQL they use the “regexp” command of MySQL, which uses “extended regular expressions”, so they should follow strictly the syntax required by that command. There is some guidance on acceptable regular expressions in the SQL v. 5.1 manual, section 11.5.2, “Regular Expressions”; and the pages accessed by the Linux commands "info gawk" and "info sed" include descriptions of regular expressions. Note that bdsMetadata does not support 'if”/” comparison'; and it pre-pends and post-pends the regexp characters .* to the regexp before passing them to the MySQL interpreter.

e.g. 'list Calibration {station = "WOL[1-7]", channel=".*_50"};' finds all the Calibrations for stations WOL1 to WOL7 with location code “50” (i.e. in the vault but unknown plinth).

10.3.5 Defaults

When items are not specified, bdsMetadata assumes the following defaults:

startTime 0000-01-01T00:00:00.000000

endTime 9999-01-01T00:00:00.000000

network “BN”

source “Main”

channelType and channelAux are filled in from the value supplied for “channel” (which should be of form XXX_YY where XXX is the channel name in FDSN standard form and YY is the location code; omit the “_YY” if there is no location code).

name “Main”

sourceMeta defaults to same as source(?)

Other numerical values that accept a default go to zero and character values default to NULL.

In a script file, a default can be changed or added by setting a variable, e.g.

set $station = “H08N1”;

add Station $newstation {,”station”,Diego Garcia North 1”};

Dates as variables are possible, e.g.

set $startDate = “2012-05-30T16:00:00”;

add Calibration $newCal {$startDate,,....(etc.)}

will add a new Calibration.

10.3.6 Storage and specification of times

Times are specified in ISO standard yyyy-mm-ddThh:mm:ss.ssssss but can be shortened to as little as yyyy-mm-dd. It is certainly not necessary to give the fractions of a second. Times are stored in the underlying SQL database as strings, not Date-Time variables, because standard SQL Date-Time variables do not include fractions of a second.

The default start time is “0000-01-01T00:00:00.000000” and the default end time is “9999-01-01T00:00:00.000000”. This means that if you specify only a start time (e.g. “2006-06-12T14:00:00.000000”) then you will find not just entries that have exactly that start time but all entries with later start times as well. To find a specific entry with that start time you have to specify the start time to be equal to that value and the end time to be equal to that value plus some very small amount of seconds, e.g. 'list Calibration {startTime=”2006-06-12T14:00:00” endTime=”2006-06-12T14:00:00.000001”};'.

To a SQL user the defaults cause the disconcerting behaviour that if you specify “startTime=exact value of a known start time”, but do not specify an end time, you get all the entries with start time greater than or equal to that value, and if you specify an exact end time, you get all the entries with end time less than or equal to that value. The approximate equivalent to “startTime=value” in bdsMetadata is in SQL, “endTime > value”, and the equivalent of bdsMetadata “endTime=value” in SQL is “startTime < value” (note < and > not <= and >=).

A further caveat is that if you want to “find” a Digitiser or Sensor from the database, you have to specify the Station, Channel, startTime and endTime of the ChannelInstrument entry that uses the Digitiser or Sensor entry that you are seeking. Digitisers and Sensors do have their own start- and end-times, but the “find” command ignores these, and it is common for them to be “zero” and “infinity” (0000-01-01T00:00:00 and 9999-01-01T00:00:00 respectively). The “find” command should specify station, channel (including location code), start time and end time, and possibly “Source”, to specify the ChannelInstrument entry exactly, e.g.

find Digitiser $SWGD {station=”SWG”, channel=”BHE”, startTime=$startdate, endTime=$enddate, source=”Main”}; print $SWGD;

returns as $SWGD the index number of the digitiser used for SWG channel BHE over the timespan described by $startdate and $enddate (which are presumed previously set with the “set” command). The usual reason for this “find” command is so you can use $SWGD to specify the digitiser in an “add ChannelInstrument” command.

10.3.7 Creating a new Station

A new station requires “Station” and “Location” objects. Although “list” says that the “Station” object has five elements:

-c 'info Station;'

#Station Members

name

alias

type

description

channels

only four may be specified: “name”, “alias”, “type” and “description”. “type” will not accept a default and must be either “station” or “array”, the latter only for the centre point of an array. “alias” is not needed and can be null unless the station name contains a non-standard character such as “#”; it is used to replace the true station name in the headers of output data.

add Station $H08N1 {"H08N1",,"station","Diego Garcia North 1"};

add Location $H08N1l {"2010-01-24T00:00:00.000000",

"9999-01-01T00:00:00.000000","BN","H08N1","WGS84",71.01430,-6.34210,0,0.0,0.0};

The final two elements of “Location” are the array offsets to east and north in km, and need not be included for a non-array station. H08N1 is an array station that happens to be at the centre of its array, so the offsets must both be set to zero as shown; for H08N2 they would be (1.8909,-1.5697). ***Beware that the longitude precedes the latitude, and the elevation is in metres (it is in km in the IDC database).***

The centre point of the array H08N is set up separately with commands:

add Station $H08N {"H08N",,"array","Diego Garcia North"}
{"H08N1”,”EDH_US”,”H08N2”,”EDH_US”,”H08N3”,”EDH_US"};

add Location $H08Nl {"2010-01-24T00:00:00.000000","9999-01-01T00:00:00.000000","BN","H08N","WGS84",71.01430,-6.34210,0,,};

The array stations/channels must all be listed in full between a second pair of braces as shown, i.e. <station name><comma><channel name><comma> (note commas not colons), and the channel name must include any location code in the form “YYY_ZZ” where YYY is the channel name and ZZ the location code.

10.3.8 Creating channels: Channel, Sensor, Digitiser and ChannelInstrument

(*Beware that the company standard sans-serif font does not distinguish the lowercase “l” (“ell”) at the end of “Channel” from the uppercase “I” (“eye”) at the beginning of “Instrument” in the compound name “ChannelInstrument”!*)

Creating a new channel of “dataType” = “seismic” for a station requires that objects of type “Channel”, “Sensor”, “Digitiser” and “ChannelInstrument” be created, and the instrument response requires objects of types “Response” and “Calibration”. “dataType” should be set as “seismic” if it is possible to deconvolve an instrument response from the data with the information in the database only, i.e. you have calibration factors and poles-and-zeros or a FAP or FIR table for the channel. This includes seismic, hydroacoustic and infrasound data, mass-position data from feedback seismometers, and the current and voltage data from the hydroacoustic installations (which have FAP tables). If the channel would be “seismic” except that you don't have the calibration or instrument information, e.g. pre-1999 three-component broadband data from WRA, “dataType” should be “seismicUnknown”. If it contains meaningful data that you don't want to deconvolve, e.g. temperature, wind-speed, and the time-code and “error” channels of TapeDigitiser digitised analogue data, “dataType” should be “data”; if it's known to contain no data , i.e. flat traces or open-circuit noise, e.g., the last twelve channels of WRA64 data, “dataType” should be

“empty”.

The ChannelInstrument object is a three-way key connecting a “Sensor” and a “Digitiser” to a “Channel”. Its members are:

-c 'info ChannelInstrument;'

#ChannelInstrument Members

startTime

endTime

channelId

source

digitiserId

sensorId

When creating a new Channel it is necessary to create a “Channel”, “Digitiser” and a “Sensor”, then use the variable names of these in the command to create a new “ChannelInstrument”, e.g.

add Channel $newchannel
{<startTime>,<endTime>,<network>,<station>,<channel>,,,<dataType>,<description>};

add Digitiser $newdigitiser
{<startTime>,<endTime>,<name>,<type>,<serialNumber>,<numberChannels>,
<baseSamplingFrequency>,<initialSamplingFrequency>,<gain>,<shared>};

add Sensor
$newsensor{<startTime>,<endTime>,<name>,<type>,<serialNumber>,<numberChannels>
, <gainUnits>,<gain>,<oldId>,<shared>};

add ChannelInstrument $newCI {<startTime>,<endTime>,$newchannel, <source>,
$newdigitiser, $newsensor};

Notes: 1. In the “Channel” specification above, “ChannelType” and “ChannelAux” have been left to defaults, as described above;

2. “oldId” should be set to the value used by Blacknest library subroutine “inst” (Peacock, S, 2014) to identify the seismometer, if possible, or to the value of “inst_id” in the 2006 AutoDRM database for instruments that appear there but not in “inst”, or to “inid” in the IDC database for instruments from there, or zero if no prior instrument ID number is associated with the instrument.

10.3.9 Creating a new Calibration

A new Channel requires one or more Calibrations unless it is of type “SeismicUnknown”, and even then, if you know the sampling rate, you need a Calibration to contain that (e.g. the unused channels of BDRS_UKNET, and the pre-1999 WRA 3-C broadband seismometer “WRLPV”). A Calibration is created from scratch with:

add Calibration $newcalibration
{<startTime>,<endTime>,<network>,<station>,<channel>,<source>,<name>,<samplingFre
quency>,<calibrationFrequency>,<calibrationFactor>,<calibrationUnits>,<depth>,<horizont
alAngle>,<verticalAngle>};

Nearly all the items in this command should not be left to default. It is possible to create a

Calibration completely on defaults (“add Calibration $newcalibration {};”) but this produces a table entry with empty station and channel, which cannot be associated with other items. It is important to choose startTime and endTime not to overlap with other “Calibration” entries for the same network/station/channel/source/name combination; bdsMetadata will reject this with a message like “Error: Line: 1: Time overlap in: Calibrations (id: 663 startTime: 2008-12-01T00:00:00.000000 endTime: 9999-01-01T00:00:00.000000) ”.

Beware that calibrationFactor is in metres per count, so “calib” in nanometres per count from the IDC and 2006 AutoDRM tables should be multiplied by 10^{-9}, and that “calper” in seconds from the IDC and 2006 AutoDRM tables is 1/calibrationFrequency in Hz.

The “name” of the Calibration should usually be “Main”, which generally denotes a manufacturer-supplied calibration valid for the duration of the sensor/digitiser pair. The alternative is “Measured”, which denotes field calibrations. For these, startTime and endTime should be set equal to the time at which the field calibration took place. Exported data will generally be labelled with the “Main” calibration valid at the start of the timespan.

10.3.10 Creating a new Response: PoleZero

To create a new “Response” entry including new poles-and-zeros requires two stages, the first to set up the poles and zeros, the second to insert them into the “blob” in table “Responses”:

1. either: add PoleZero $newpolezero {p1r,p1i,p2r,p2i,...}{z1r,z1i,z2r,z2i...};

(where (p1r,p1i) is the first pole expressed as real and imaginary parts, (z1r, z1i) is the first zero, and so on);

or: read PoleZero $newpolezero {“filename”};

(where filename is the name of a file containing poles-and-zeros in SAC format, and you need the quotes; or you can use a variable, preceded by a “$”, and no quotes.);

2. add Response $newresponse {<startTime>,<endTime>,<network>,<station>,<channel>,<source>,<name>,0 (zero), “Overall”, ”PoleZero”,<gain>,<gain frequency>,”A”,<decimation>,<symmetry>,<description>,0 (zero),<sample rate>, $newpolezero};

Note that the name of the new PoleZero object has to be inserted as the final item, “response”, in the Response specification. The elements in “Response” are shown by the “info” command:

-c 'info Response;'

#Response Members

startTime

endTime

network

station

channel

source

stage

name

type

gain

gainFrequency

stageType

decimation

symmetry

description

measured

sampleRate

response

The “name” has to be “Overall” and “type” has to be “PoleZero”. “gain”, “gainFrequency”, “decimation” and “symmetry” can often be left to be defaults. “stageType” should be set to either “A” (analogue, poles-and-zeros in radians), “B” (analogue, poles-and-zeros in Hz), (the other possibilities are “C”, composite, or “D”, digital). The “description” can be anything between inverted commas. “measured” is a boolean to show whether or not the response was measured in the field, so usually for poles-and-zeros it should be 0 (zero). “sampleRate” can be left out, or set to the recording sampling rate (it is the value in the Calibration, not here, that is used in data output).

It is important to choose startTime and endTime not to overlap with other “Response” entries for the same network/station/channel/source/name combination; bdsMetadata will reject this with a message like “Error: Line: 1: Time overlap in: Responses (id: 599 startTime: 2008-12-01T00:00:00.000000 endTime: 9999-01-01T00:00:00.000000)”.

10.3.11 Creating a new Response: FAP

FAP responses have to be read from a file, with the “read” command:

read FAP $fapname {“fullpathtoFAPfile”};

(you need the inverted commas around the file pathname, or you can use a variable, preceded by “$”.).

$fapname now refers to the FAP as a “blob” and can be inserted into the Response in the same way as for a pole-zero response:

add Response $newresponse
{<startTime>,<endTime>,<network>,<station>,<channel>,<source>,0 (zero),<name>,”FAP”,<gain>, <gain frequency>,”C”,<decimation>,<symmetry>,<description>,<measured>,<sampleRate>, $fapname};

Again the “name” has to be “Overall”; and the “type”, FAP. The “stageType” is “C” since a stand-alone FAP file is usually a composite response of the instrument and digitiser. If the FAP file is the output from a field calibration, then “measured” should be boolean 1,

otherwise 0 (zero).

10.3.12 Creating a new composite response

To read in a composite response from an IDC internal-format response file, the command is:

read Responses $responsename {“fullpathtoResponsefile”};

(where you need the inverted commas, or you can use a variable, preceded by “$”, to represent the filename).

The contents of the composite response file are then inserted into the “Response” entry in the database by the command:

add Responses $newresponses
{<startTime>,<endTime>,<network>,<station>,<channel>,<source>,0 (zero),<name>,”PAZFIR”,<gain>, <gainFrequency>,”C”,<decimation>,<symmetry>, <description>,<measured>,<sampleRate>,$responsename};

When a composite response is read in, many of these variables are ignored, because each stage of the composite response will have its own value. These include the stage number (the zero), the “name”, “type” (you can put in “PAZFIR” or “PAZFAP” to remind yourself what the format of the response is, but the program ignores it), “gain”, “gainFrequency”, “stageType” (the “C” reminds you it's a composite response), “description” (though again it is useful to put something in to remind yourself where the response came from when you next inspect your script), “measured” and “sampleRate”.

The format of the composite responses in IDC response files is not documented, but can be deduced from source code in the module “gbase-libs” provided with “Geotool” in the NDC-in-a-Box software package. There is a comment in the header of libsrc/libresp/response.c containing some of the variables expected from these files:

```
* typedef struct
 * {
 *      GObjectPart core;
 *
 *      char        source[13];     /\* "theoretical" or "measured"  *\/
 *      int         seq_num;        /\* sequence number              *\/
 *      char        des[13];        /\* description                  *\/
 *      char        type[7];        /\* fap, paz, or fir             *\/
 *      char        units[2];       /\* d, v, or a                   *\/
 *      char        author[45];     /\* author or information source *\/
 *      double      a0;             /\* normalization for paz / gain for fir *\/
 *      int         npoles;         /\* number of poles for paz type *\/
 *      int         nzeros;         /\* number of zeros for paz type *\/
```

```
*     ComPlex     *pole;          /\* poles for paz type          *\/
*     ComPlex     *pole_err;      /\* poles for paz type          *\/
*     ComPlex     *zero;          /\* zeros for paz type          *\/
*     ComPlex     *zero_err;      /\* zeros for paz type          *\/
*     int         nfap;           /\* number of fap triplets      *\/
*     float       *f;             /\* frequency for fap type      *\/
*     float       *a;             /\* amplitude for fap type      *\/
*     float       *p;             /\* phase for fap type          *\/
*     float       *a_error;       /\* amplitude error for fap type *\/
*     float       *p_error;       /\* phase error for fap type    *\/
*     float       samprate;       /\* samples/sec for fir type    *\/
*     int         num_n;          /\* # numerator coeffs. for fir  *\/
*     int         num_d;          /\* # denominator coeffs. for fir*\/
*     float       *n;             /\* numerator for fir type      *\/
*     float       *n_error;       /\* numerator error for fir type *\/
*     float       *d;             /\* denominator for fir type    *\/
*     float       *d_error;       /\* denominator error for fir   *\/
*
*     GSE_CAL     *cal;           /\* associated CAL for GSE format *\/
* } *Response;
```

10.3.13 Cloning channels

The command for cloning a channel is:

clone Channel <variable-name> {pattern to identify channel to be cloned}{items in new channel that will be different from clone pattern channel} {list of objects to be cloned}

The <variable-name> is the name of the NEW channel (and must of course begin with a “$”). The names in the list of objects to be cloned begin with lowercase letters, so “calibration” not “Calibration”.

The following example clones a channel at one station to create a similar channel at another station. Hence it is desirable to clone, rather than share, all the items.

clone Channel $H10N3 {startTime="2010-01-24T00:00:00.000000", endTime="9999-01-01T00:00:00.000000", station="H10N1", channel="EDH"}{startTime="2010-01-24T00:00:00.000000", endTime="9999-01-01T00:00:00.000000", station="H10N3", channel="EDH"}{calibration, response, instrument, digitiser, sensor};

All the “Calibration”, “Response” and “Instrument” entries from the pattern channel that have timespans within or overlapped by the specified timespan of the new channel are duplicated for the new channel. If no timespan is specified for the new channel, it is

assumed to be the default 0000 to 9999

The next example is to create a North horizontal channel by cloning an East horizontal channel from the same station. Here the Digitiser and Sensor must not be cloned because they should be shared, but the Instrument must be specified as being cloned, otherwise no instrument at all is created for the station. The Calibration and Response have to be cloned because they might in principle be different between the two channels.

clone Channel $H9N1N {startTime="2004-03-17T00:00:00.000000", endTime="9999-01-01T00:00:00.000000", station="H09N1", channel="EHE"}{channel="EHN"} {Calibration,Response,Instrument};

It is then necessary to change the Calibration of the new channel to make the horizontalAngle zero (north) not 90 degrees (east) and insert the calibrationFactor if that is different:

change Calibration $c9N1N {startTime="2004-03-17T00:00:00.000000", endTime="9999-01-01T00:00:00.000000", station="H09N1", channel="EHN"}{calibrationFactor=0.00835e-9,horizontalAngle=0.0};

[Within the database there are three tables involved in storing instrument information: Sensors and Digitisers, which contain the details of the sensor (seismometer) and digitiser, and ChannelInstruments, which relates the sensor and digitiser to the channel (actually to the entry for the channel in table Channels). When you clone a channel it is necessary to clone the entry in ChannelInstruments even if you intend to “share” pre-existing instrument and digitiser, otherwise there is nothing to relate these to the new channel (the new entry in Channels).]

10.3.14 9.3.13 Importance of setting “oldId”

When you clone a Sensor, the new Sensor has the value “oldId” set to zero, not to the value of the pattern Sensor. It is UP TO YOU, but very important, to insert the correct “oldId” for the new Sensor – important because this allows the instrument response to be traced back to the 2006 AutoDRM and possibly to the “inst” responses used in pre-Linux Blacknest processing (Peacock, S, 2014) and hence in publications pre-dating about 2006. See the section on creating a new channel, above, for instructions on choosing the correct “oldId”. The bdsMetadata command to be issued is:

change Sensor $<newsensor> {Station=<station>,Channel=<channel>,startTime=<startTime>,endTime=<endTime>} {oldId=<value>};

e.g., if there is only one Sensor entry for this station and channel, you can get away with a command like:

change Sensor $s8N2 {station="H08N2", channel="EDH_US"}{oldId=1115513};

If there are several and you want to change only one, then you need to specify startTime and endTime as well.

The other thing you need to change after cloning is the “description” of the new channel. This is easily changed with a command like:

change Channel $c8N2 {station="H08N2", channel="EDH_US"}{description="D.G.North element 2"};

10.3.15 Seismometer/Digitiser change at an existing station

If the seismometer and/or the digitiser are completely new to the system, it is necessary to create a “Sensor” and a “Digitiser” for them. You also need to create the “PoleZero” or “FAP” object containing the seismometer response, by reading the data in from a file, usually. The commands are:

read PoleZero $<newpolezero> {“<filename>”};

add Sensor $<newsensor> {“<startTime>”,”<endTime>”, “<name>”, “<type>”, “<serial number>”, <number-of-channels>, “<gain units>”, <gain>, <oldId>, };

The final comma precedes the element “shared”, but leave the system to decide what to put for that. number-of-channels, gain units and gain are optional and you can simply put four commas between the serial number and “oldId”. It is VERY important to include a sensible value of “oldId” - see above, on Cloning Sensors.

add Digitiser $<newdigitiser> {“<startTime>”,”<endTime>”, “<name>”, “<type>”, “<serial number>”, <number-of-channels>, <base sampling frequency>, <initial sampling frequency>, <gain>,};

Again the final comma precedes the element “shared”, which the system should be allowed to find for itself. “number of channels”, “base sampling frequency” and “initial sampling frequency” are optional; indeed, for a digitiser with several outputs at different sampling rates, it is necessary to create a separate “Digitiser” for each output if the “base sampling frequency” item is used. The “initial sampling frequency” is the frequency at which the digitiser initially samples the data, after which the digital data are decimated down to the “base sampling frequency”, at which the data are output.

With these items created, each channel of the new seismometer/digitiser has to have a new Calibration, Response and ChannelInstrument set up, and the corresponding Calibration, Response and ChannelInstrument for the old seismometer/digitiser have to have their endTimes changed from open-ended (“9999-01-01T00:00:00”) to the time at which the new seismometer was substituted. This is easy for Calibration and Response, because you can hunt for them using the station and channel name. You can use either “change” on the existing Calibration and “add” for the new one, or “split”:

change Calibration $<somename> {endTime=”9999-01-01T00:00:00”, station=<station>, channel=”<channel>”}{endTime=”<switch date>”};

add Calibration $<newname> {“<switchdate>”, “9999-01-01T00:00:00”, “BN”, “<station>”, “<channel_locncode>”, “Main”, “Main”, “<sampling frequency>”, “<calibration frequency>”, “<calibration factor>”, “<calibration units>”, “<sensor depth of emplacement>”, <horizontal angle>”, “<vertical angle>”};

OR

split Calibration $<newname> {startTime=<oldstarttime>, endTime=<switchdate>} <switch date> {calibrationFactor=<new calibration factor>, calibrationFrequency=<new calibration frequency>, samplingFrequency=<new sampling frequency>};

“split” is easier because it automatically changes the date of the old entry and creates the new entry with a single command.

Beware that the calibration factor should be in metres per digital count and calibration units

should be “m” for seismometers; and pascals per count and pascals (“Pa”), respectively, for incrasound and hydroacoustic sensors. The '“Main”, “Main”' are the “source” and “name”. The “name” tells you whether the calibration was from the manufacturer's data sheet (“Main”) or a field measurement (“Measured”). The “source” should be the same as the “source” of data that require this calibration. For most current calibrations this will be “Main”. For calibrations that are valid for TapeDigitiser data only, the “source” should be “TapeDigitiser”.

Response entries can be “split”, using the date at which the seismometer was changed. This changes the end time of the current Response to the switch date, and create a new Response starting on the switch date and ending at the previous end date of the original Response (which will be infinity for a currently valid response). Here is the example for a currently valid response

split Response $<newname> {endTime=”9999-01-01T00:00:00”, station=”<station>”, channel=”<channel_locationcode>”} <switch date> {“<switch date>”,,,,,,,,,,, $<newpolezero>};

where $<newpolezero> is the variable name you gave to the new set of Poles and Zeros when you read them in (see beginning of this section) – hence the “split” statement has to follow in the same script as the statement creating the new poles-and-zeros.

The “switch date” may be a variable, e.g. $switchdate. When defining the variable (or any time variable) with “set”, do not use inverted commas, e.g. this is correct:

set $switchdate = 2012-07-12T09:30:00

10.3.16 Creating a log channel

The “.txt” files created by the Güralp SCREAM system are imported into the BDS and assigned to channels with “DataType” = “log”. Conventionally these channels are called “LOG”, with or without a location code. Usually there is one “LOG” channel per station, but if there are two or more, it is necessary to distinguish them by location codes, e.g. the two “LOG” channels from EKA pit EKB3, one from the seismometer, one from the infrasound sensor, are called “LOG” and “LOG_IS” in the BDS.

The log channel is created with a command such as

add Channel $HEALOG {$startdate, $enddate, "BN", “HEA”, “LOG”,,, "log", "HEA T34515/B565 log"};

(the above example assumes that $startdate and $enddate have been assigned already with “set” commands).

There should be a ChannelInstrument entry for the “LOG” channel, containing the details of the sensor and digitiser producing the log file. A “LOG” channel does not need Calibration or Response entries.

If the sensor and digitiser are already in the system it is necessary to “find” them so that their indices can be inserted into the “add ChannelInstrument” command, along with the index of the Channel just created ($HEALOG in the example below):

find Sensor $HEAS {station=$sta, channel="BHE", startTime=$startdate, endTime=$enddate};

find Digitiser $HEAD {station=$sta, channel="BHE", startTime=$startdate, endTime=$enddate};

add ChannelInstrument $HEALOGCI {$startdate, $enddate, $HEALOG, "Main", $HEAD, $HEAS};

If either sensor or digitiser is changed, a new ChannelInstrument entry should be created. This can be done with “split ChannelInstrument” in the same way as for creating a new entry for a seismic channel such as BHZ.

11 References

Bartholomew, P., 2004, Eskdalemuir 3-component seismometer orientation correction, AG note 412, AWE.

BEAM, 2005, Proposed Blacknest Digitising System, https://portal.beam.ltd.uk/support/blacknest/files/tapeDigitiser/info/ProposedSystem-1.pdf, BEAM Ltd., Bristol.

BEAM, 2008a, Blacknest Tape Digitising System, File Format – Version2.0, https://portal.beam.ltd.uk/support/blacknest/files/tapeDigitiser/info/TapeDigitiserFileFormat-2.0.pdf, BEAM Ltd., Bristol.

BEAM, 2008b, Blacknest Tape Digitising System, Information File Format Version 1.1, https://portal.beam.ltd.uk/support/blacknest/files/tapeDigitiser/info/TapeDigitiserInformationFile-1.1.pdf, BEAM Ltd., Bristol.

BEAM, 2009, Blacknest Tape Digitising System, User Manual Version 1.1.33, https://portal.beam.ltd.uk/support/blacknest/files/tapeDigitiser/doc/TapeDigitiserUserManual.pdf, BEAM Ltd., Bristol.

BEAM, 2010a, Blacknest Data System (BDS) User Manual, https://portal.beam.ltd.uk/support/blacknest/files/bds/doc/BdsUserManual.pdf, BEAM Ltd., Bristol.

BEAM, 2010b, BDS Data Import Programs, https://portal.beam.ltd.uk/support/blacknest/files/bds/doc/BdsImport.pdf, BEAM Ltd., Bristol.

BEAM, 2010c, BDS Command Line Data Access Client, https://portal.beam.ltd.uk/support/blacknest/files/bds/doc/BdsDataAccess.pdf, BEAM Ltd., Bristol.

BEAM, 2010d, Blacknest BDS data file Overview, https://portal.beam.ltd.uk/support/blacknest/files/bds/doc/BdsDataFile.html, BEAM Ltd., Bristol.

BEAM, 2010e, Blacknest Tape View user manual 1.1.23, https://portal.beam.ltd.uk/support/blacknest/files/tapeDigitiser/doc/TapeView.pdf

BEAM, 2010f, Blacknest TapeDigitiser Processing, https://portal.beam.ltd.uk/support/blacknest/files/tapeDigitiser/doc/TapeDigitiserProc

essing.pdf

BEAM, 2011a, Blacknest Data System (BDS), BdsMetadata Program 1.2.14, https://portal.beam.ltd.uk/support/blacknest/files/bds/doc/bdsMetadata.pdf

BEAM, 2013a, Blacknest Data System (BDS), Instrument Response handling, 2.0.11, https://portal.beam.ltd.uk/support/blacknest/files/bds/doc/bdsResponses.pdf

BEAM, 2013b, Blacknest BDS Development, Programming Manual – 2.0.5, https://portal.beam.ltd.uk/support/blacknest/files/bds/doc/BdsDevelopment.pdf

Blacknest, 2008, Blacknest (BKNAS) digital data formats, AG note 429, AWE.

Bowers, D., 2004, Practical digital anti-alias filters for the Eskdalemuir upgrade, EKA, AWE report 2/04.

Bowers, D., Wallis, N. W., Budd, T., and Bartholomew, P., 2008, Assessment of the prototype BEAM analogue-tape digitiser system, AWE report 393-08.

Burch, R., 1984, A recalculation of the SP responses for GBA, EKA and WRA, AG note 290, AWRE.

Güralp, 2010, Güralp Compressed Format (GCF) Quick Reference, http://www.guralp.net/articles/20060404-howto-gcfformat/support, Güralp Ltd., Aldermaston.

Holdsworth, B, 1969, VELA and Hutchins time code converter, AG note 95, AWRE.

Key, F., 1968, YKA time code, AG note 91, AWRE.

Key, F., 1983, BDDS facilities for handling LP data recorded on the Blacknest DRS, AG note278, AWRE.

Key, F., and Marshall, P. D., 1977, Array amplitudes on ANDAC, AG note 209, AWRE.

IDC 2004, Formats and Protocols for Messages, IDC-3.4.1 Revision 6, 2004, International Data Centre, CTBTO, Vienna.

IDC 2002a, Formats and Protocols for Continuous Data, CD-1.0, revision 0.1, IDC3.4.2Rev01, International Data Centre, CTBTO, Vienna.

IDC 2002b, Formats and Protocols for Continuous Data, CD-1.1, version 0.3, IDC3.4.3, International Data Centre, CTBTO, Vienna.

FDSN/IRIS/USGS, 2006, SEED Reference Manual, Standard for the Exchange of Earthquake Data, SEED format Version2.4, Incorporated Research Institutions for Seismology (IRIS), Seattle, WA, USA

IRIS, 2010a, SAC Users Guide, webpages at http://www.iris.edu/software/sac/manual.html

IRIS, 2010b, Rdseed v5.0 Manual, webpages at http://www.iris.washington.edu/manuals/rdseed.htm

IRIS, 2013, EvalResp v. 3.3.3 manual, webpages at www.iris.edu/dms/nodes/dmc/manuals/evalresp/

James, E. W., and Burch, R. F., 1974, The short-period FM sender (system 497) – this replaces AG no. 138, AG note 159, AWRE.

Marshall, P. D., 1980, Amplitudes from ANDAC playouts, AG note 255A, AWE.

Marshall, P. D., 1989, Calibration data for UK-type array stations and Blacknest recording systems, AG note 306, AWE.

New, S. V., 1974, System 478A 14 Dual Channel Card Unit Design notes on the FM demodulator KA1200, AG note 144, AWRE.

Peacock, S, 2014, Blacknest subroutine “inst” for applying instrument response corrections to seismic traces”, AG note 429, AWE.

Trodd, H., 1986, Summary of processed data types and their formats, AG note 304, AWRE.

12 Appendix 1 – Database operations via API

12.1 Correcting mistakes

“Mistakes” means wrong values of metadata items such as calibration factor, poles-and-zeros, latitude/longitude, but not channel name, so generally not mis-orientation (see below for that). The changes are retrospective, so that data served before and after making the change will have different metadata attached. No new rows in database tables are created.

In bdsMetadata these can be corrected with the “change” command, e.g., to correct a mistyped calibration factor,

change Calibration $R4newcalib {startTime=$startdate, endTime=$endoftime, station="EKR4", channel="BDF", source="Main"}{calibrationFactor=6.87e-6};

If the erroneous entry is a time at which something, e.g. a calibration, changed, then you need to change the end date of one entry and the start date of the other. You have to do these in an order so that at no time do the end and start dates overlap e.g. make the end date earlier before making the start date earlier, otherwise you will get a database error.

Correcting an error in a “sensor” or “digitiser” is difficult because they lack unique keys. Also many of them were unnecessarily duplicated, e.g. a three-component set has three copies of the sensor and digitiser instead of one shared between the three components. This is because duplication was the way it was done in the 2006 AutoDRM database, which was imported into the BDS database. At present no “Sensor” or “Digitiser” attributes are essential to import or export except “oldId” for BKNAS export, although it is sensible for the serial numbers to be correct so that calibration sheets can be obtained from the Güralp “caldoc” service. A “sensor” or “digitiser” cannot be specified by its serial number or “oldId”: you have to specify it by the station and channel that uses it, e.g. bdsMetadata command “list Sensor {station=”EKB”, channel=”BHZ.*”};” (the “.*” is a wildcard in “regexp” format for the location code, so that you get sensors for all the different broadband vertical channels that have existed at EKB). If the “list” command gives only one line, or several lines with the same serial number, then you can correct the serial number with a “change” command e.g. 'change Sensor $EKBBHZ {station="EKB", channel="BHZ"} {serialNumber="T3442"};'

BEWARE – in bdsMetadata if you try to specify a channel including its location code with the syntax “CCC_LL” and get the “LL” (location code) wrong so that it specifies a non-existent channel, then it will access all sensors, not none. Fortunately the “change” command will fail with “maximum number of changes exceeded”.

Affects objects: any, depending on the error.

12.2 mis-orientation

Mis-orientations are also retrospective, but new rows will be necessary if the channel

names change as a consequence of the mis-orientation. Usually a 3-component set was thought to be oriented N and E, and its channels were named with names ending “Z”, “N” and “E”, but then you find it was mis-oriented and you need to change the channel name retrospectively so that the three channels end in “1”, “2” and “3”, and on re-orienting it you need new entries with channel names “Z”, “N” and “E”.

If the re-orientation is not to N and E and the channels were already called “1”, “2” and “3” then you do not need to create new channels. Proceed as for “correcting mistakes” above by changing the “horizontalAngle” (and “verticalAngle” if necessary) of the Calibration entry covering the timespan of the mis-orientation.

Affects objects: Channel, ChannelInstrument, Calibration, Response

1. Create new Channel entries for the channels ending in “1”, “2” and “3” by cloning the existing channels ending “Z”, “N” and “E”, but when cloning do not clone the sensor, instrument or digitiser;

2. “share” the sensor and digitiser of the existing “Z”, “N” and “E” channels with the new channels;

3. Change the Calibration entries for the new “1”, “2” and “3” channels to contain the correct start and end times of the mis-orientation and the mis-oriented seismometer's orientation;

4. If the start date of the mis-orientation is after the start date of the original Calibration entry (that is, the one with “Z”, “N” and “E” channels), then change the end date of the original Calibration entry to be the start date of the mis-orientation. Then you have to create (by cloning it) a new Calibration entry to apply after the date of the re-orientation.

5. Usually, though, a mis-orientation occurs when a seismometer is replaced or first installed, in which case a Calibration entry for “Z”, “N” and “E” channels will start at the time of the mis-orientation. Change the start time of this entry to be the date of re-orientation (assuming that the re-orientation actually put the seismometer to N and E orientation).

6. Because you cloned the Response (you cannot “share” a response the way you can a Digitiser and Instrument), you will have to modify the dates on the Response entries for the “1”, “2” and “3” and “Z”, “N” and “E” channels, too.

12.3 A whole new station

1. Add a “Station” entry for the station, or individual stations if it is an array, with “type” = “station”, and corresponding entries for the location (arrayOffsetEast and arrayOffsetNorth can be left at zero if not known);

2. If it is an array, add a separate “Station” entry for the array, with “type” “array” and a list of the stations and channels (in station-channel pairs) contributing to the array, e.g., in bdsMetadata,

add Station $H08N {"H08N",,"array","Diego Garcia North"}
{"H08N1","EDH_US","H08N2","EDH_US","H08N3","EDH_US"};

3. Create one or more “Sensor” and “Digitiser” entries. For a modern 3-C station only one of each should be required. The most essential piece of information for both is the serial number, since this can be used to get the manufacturer's response, e.g. from Güralp's “caldoc” service. A value of “oldId” is also important because this can be correlated with responses hard-wired into the Blacknest “inst” subroutine and in headers of archive data in BKNAS format, principally on “John's CDs” (Peacock, S, 2014). Use the “inst_id” value of the seismometer from the 2006 AutoDRM database.

4. For one channel at each station, create a “Channel” to represent one of the seismic-type channels e.g. BHZ, with “dataType” = “seismic”;

5. For that channel, create a “Calibration”, which includes the calibration factor and frequency, sampling frequency and orientation of sensor (horizontal and vertical angles), and if the sensor is down a borehole, its depth below the “elevation” that you gave for the station location (if you omit this it defaults to zero). The “source” value within the “Calibration” should be the same as the “source” of data to which the calibration applies. The default is “Main”.

6. If the channel has frequency response information (most seismic channels do), create a “Response”: this requires two operations, reading in the response as poles-and-zeros or a FAP table or composite response from a file, and creating a “Response” entry including the information from the file (which is stored within the database as a binary string). Again the “source” value should be the same as the “source” used to import data to which the response applies. The variable “measured” should be set to 1 (boolean) if it was measured during an in-situ calibration exercise.

7. Once one Channel (with associated Response and Calibration) and one Sensor and Digitiser are created, then other (seismic-type) channels derived from the same seismometer/digitiser may be created by cloning. Each Channel should have its own ChannelInstrument entry containing the channel ID, Sensor ID and Digitiser ID, and the last two should be the same for all ChannelInstrument entries. This is done in bdsMetadata by cloning the “Channel”, which also clones the ChannelInstrument. Prevent the cloning of the Sensor and Digitiser by including only “response”, “calibration” and “instrument”, not “digitiser” or “sensor”, in the list of items to be cloned (at the end of the command). This causes the single Sensor and Digitiser to be “shared” among all the channels.

8. After cloning it is usually necessary to adjust the calibration information of the cloned Channel, in particular calibrationFactor (which for Güralp equipment is usually slightly different among channels from the same seismometer/digitiser), and horizontalAngle and verticalAngle. In bdsMetadata this is done with the “change” command.

8. For a digitiser that also produces non-seismic data channels, e.g. the log info saved as “.txt” files by SCREAM, the channel should be called “LOG” and given “dataType” of “ascii”. There are no corresponding “Response” or “Calibration” entries.

Affects objects: all.

12.4 Change of seismometer and/or digitiser without change to channel name

A “like-for-like” replacement of seismometer or/and digitiser should not require the channel name to be changed, e.g. no need to change “BHZ” to “LHZ” if a broadband seismometer is replaced by an even broader-band seismometer. The necessary change is to “Calibration”, and maybe “Response”, if a seismometer with a different frequency response is inserted. You need to know the date and time of the change and the new calibration factor and poles/zeros (or FAP or composite response file) and, if a new seismometer or digitiser has been installed, its serial number and preferably make/model.

1. Either: if necessary, create new sensor and/or digitiser entries with the serial numbers of the new seismometer and digitiser, or, if a seismometer and/or digitiser already existing in the database but not currently in use for this channel has been used, find their “id” numbers. Finding ID numbers is best done with the GUI, by inspecting previous entries in channels in which the seismometer and/or digitiser were used. The “find” command in bdsMetadata does not work for “sensors” or “digitisers”.

2. You need to change the end date of the current “ChannelInstrument” entry for the channel from “infinity” to the date/time of the change, and insert a new “ChannelInstrument” entry for the new seismometer and/or digitiser. In bdsMetadata this is done with the “split” command, with the change date as the “split” date, and specifying the “id” numbers of the new sensor and digitiser (if necessary) after the split time, thus:

split ChannelInstrument $newchannelinstrument {station=<station>, channel=<channel_locncode>, startTime=<start date of current ChannelInstrument>} <split time> {sensorId=<ID of new sensor>, digitiserId=<ID of new digitiser>};

If you created a new sensor and/or digitiser earlier in the same bdsMetadata script, then the “ID of new sensor” is the variable name you specified (preceded by a dollar sign) immediately after the “add Sensor” command (and likewise for the digitiser). Otherwise it is the ID number you found by looking with the GUI.

3. A new “Calibration” entry has to be created, and the end date on the current one has to be changed from “infinity” to the date of change of the calibration. In bdsMetadata the change to the Calibration can be done with “split”, e.g.

split Calibration $newcalibration {station=<station>, channel=<channel_locncode>, startTime=<start date of current Calbration>} <split time> {calibrationFactor=<new calibration factor>, calibrationFrequency=<new calibration frequency>, samplingFrequency=<new sampling frequency>};

(omit any of the last three parameters if they do not need to be changed).

4.If the Response entry needs changing then you should not use “split” but amend the current Response entry to end at the change date and create a new Response entry. To create a new Response you will need to create first a new “PoleZero” entity or a new “Fapfile” or composite-response-file entity. You can read in the poles-and-zeros or the FAP file with the bdsMetadata “read Response” command (this is the only way to load a FAP file) or use the “add” command to present poles-and-zeros through the command line, e.g.

add PoleZero $pz0 {<re1>,<re2>,<re3>,...}{<im1>,<im2>,<im3>...};

For a composite response use the “read Responses” command.

The advantage of using the GUI for adding poles-and-zeros is that it allows you to add

duplicate or conjugate Ps and Zs without retyping, and it checks for sanity i.e. that the poles have negative real parts. The poles-and-zeros in a file can be in SAC format. FAP files can be in a three-column format: frequency, amplitude, phase, or in IDC format as found in the configuration files directory “station_specs/rsp”. Composite responses are read from IDC internal-format files, usually found in directory station_specs/rsp underneath the “configuration files” directory (currently /shareddata/documents/idc/config_files).

12.5 Correcting a calibration factor in an existing channel(s)

It is necessary to find the “Calibration” entries that need correcting (usually for calibration factor, but maybe for “hang” and “vang” or sampling frequency), then change the calibration factor in all these entries. In bdsMetadata the command is something like “change Calibration $newcalib {startTime=yyyy-mm-ddThh:mm:ss, endTime=yyyy-mm-ddThh:mm:ss, station=”STA”, channel=”CHAN_LOCN”}{calibrationFactor=newfactor, calibrationFrequency=newfreq};” (beware that the BDS uses calibration frequency while the IDC and 2006 AutoDRM databases uses period here). If the change in calibration factor is associated with a particular seismometer or digitiser, check that the seismometer/digitiser with the changed calibration factor does not appear earlier or later in the station history: seismometers/digitisers are swapped in and out, particularly at EKA where there is a “pool” of two spares of each, so the same seismometer or digitiser might appear at other stations at different times and in combination with other digitisers/seismometers.

12.6 Correcting a response in an existing channel(s)

This is not quite the same as correcting a Calibration because you usually need to load up the new response from a file. For small changes in one or two poles/zeros in a single-stage response, or the addition of a pole or zero, with nothing else changed, it is probably easiest to use the AdminGUI. For more major changes it is necessary to load up the new “response” with, in bdsMetadata, a “add” command, e.g. “add PoleZero $newPAZ{...}” or “read PoleZero $newPAZ{“filename”}” (or read FAP $newFAP{“filename”}”) (note you need the quotes either side of the filename). After that, the Response entry needs to be changed to incorporate the new response, with “change Response{startTime=...(give enough specifics to identify the station and channel and timespan uniquely)} {response=$newPAZ};” If you have changed the type of the response (e.g. from poles-and-zeros to FAP) you need to add “type=FAP” to this command.

For a composite response (from an IDC response file, with PAZ or FAP and FIR stages) it is necessary to delete the existing response and read in the entire new response.

12.7 Correcting the co-ordinates of a station

Co-ordinates (lat and long) are in the Locations object, and the command in bdsMetadata is “change Location $newloc {startTime=yyyy-mm-ddThh:mm:ss, endTime=yyyy-mm-ddThh:mm:ss, station=”STA”}{longitude=value, latitude=value, elevation=value}”. For a station within an array you can change the offset within the array by changing “arrayOffsetEast” and “arrayOffsetNorth”, which are in km from the array centre. Note the array centre should be contained in the Location object distinguished by “station” = array name (e.g. EKA), which is coupled to the Station object with the same “station” name and “type” = “array”. If the array centre coincides with an actual station (e.g. YKB5) then there

should be two Station entries, each with its own Location entry, one for YKA, with type “array”, one for YKB5, with type “station”. Each has its separate Location object, and in both of these, “arrayOffsetEast” and “arrayOffsetNorth” are zero.

If you want to correct the depth of the sensor (for a borehole sensor), the entry for depth has to be changed in the Calibrations object. This is the depth in metres (beware the IDC database uses km) measured downwards from zero at the elevation given in the Locations object. For tethered hydrophones, “elevation”, negative, is the seafloor depth at the position of the hydrophone, and “depth” is the depth below sea surface, positive, at which the hydrophone is tethered – this was my lazily reading “elev” from table static.site and “edepth” from static.sitechan in the IDC database (and converting both from km to m).

12.8 Upgraded YKA, January 2014

YKA was upgraded and re-opened in January 2014 with new digitisers from Güralp. Data are now supplied in miniSEED. All the pit names in the headers are different, with “YKA” rather than “YK” prepended, so “YKR1” becomes “YKAR1”, for instance. It was necessary, therefore, to close all the entries referring to the old pit names and open essentially a whole new array station. bdsMetadata commands were run as follows:

add Station $YKAB0 {"YKAB0","","station","YKA pit Blue0"};

(etc.)

Then it was necessary to delete the “YKA Array” entry in table “Stations” and re-enter it, rather than adding the new stations to the existing entry:

delete Station {station="YKA"};

add Station $YKA {"YKA","","array","Yellowknife Array Beam Reference Point"} {YKR1,SHZ,YKR2,SHZ,YKR3,SHZ,YKR4,SHZ,YKR5,SHZ,YKR6,SHZ,YKR7,SHZ,YKR8,SHZ,YKR9,SHZ,YKB1,SHZ,YKB2,SHZ,YKB3,SHZ,YKB4,SHZ,YKB5,SHZ,YKB6,SHZ,YKB7,SHZ,YKB8,SHZ,YKB9,SHZ,YKB0,SHZ,YKAR1,SHZ,YKAR2,SHZ,YKAR3,SHZ,YKAR4,SHZ,YKAR5,SHZ,YKAR6,SHZ,YKAR7,SHZ,YKAR8,SHZ,YKAR9,SHZ,YKAB1,SHZ,YKAB2,SHZ,YKAB3,SHZ,YKAB4,SHZ,YKAB5,SHZ,YKAB6,SHZ,YKAB7,SHZ,YKAB8,SHZ,YKAB9,SHZ,YKAB0,SHZ};

Beware of syntax in creating an array "Station" entry: the list of stations and channels is in a separate set of curly braces at the end and has commas throughout as separators rather than the usual bdsMetadata notation of "STA:CHAN" with colon separators.

add Location $YKAB0L {$newstarttime,$endoftime,$network,"YKAB0",$datum,-114.6060, 62.6059,196,0.4599, 12.5648};

(etc.)

Change the end date on the old locations, channels, instruments, calibrations and responses (using a regexp wildcard):

change Location $YKRBWold {station="YK[RBW].*", startTime=$oldstarttime, endTime=$endoftime}{endTime=$oldendtime};

change Calibration $YKRBWC {station="YK[RBW].*", startTime=$oldstarttime, endTime=$endoftime}{endTime=$oldendtime};

change Response $YKRBWR {station="YK[RBW].*", startTime=$oldstarttime, endTime=$endoftime}{endTime=$oldendtime};

change Channel $YKRBWCh {station="YK[RBW].*", startTime=$oldstarttime, endTime=$endoftime}{endTime=$oldendtime};

change ChannelInstrument $YKRBWCI {station="YK[RBW].*", startTime=$oldstarttime, endTime=$endoftime}{endTime=$oldendtime};

Create a new digitiser (the digitiser was the only thing that changed at all stations):

add Digitiser $YKAB0D {$newstarttime, $endoftime, "Guralp","DM24","",1,40.0,40.0,1.0,};

Clone one old channel and modify the clones:

clone Channel $YKAB0Ch {station="YKB0", channel="SHZ", startTime=$oldstarttime, endTime=$oldendtime}{startTime=$newstarttime, endTime=$endoftime,station="YKAB0"} {response,instrument,calibration};

Change the calibration - don't forget units are metres not nm.

change Calibration $YKAB0Cal {station="YKAB0", channel="SHZ", startTime=$newstarttime, endTime=$endoftime}{calibrationFactor=.0037168e-9, calibrationFrequency=4.0, samplingFrequency=40.0};

Note we don't need to change the Response because the seismometer hasn't changed.

Change the ChannelInstrument to insert the new digitiser.

change ChannelInstrument $YKAB0CI {station="YKAB0", channel="SHZ", startTime=$newstarttime, endTime=$endoftime}{digitiserId=$YKAB0D};

Clone YKAB0 to make all the other array channels, fixing the calibrations as we go (they are all different - Guralp digitisers).

clone Channel $YKAB1Ch {station="YKAB0", channel="SHZ", startTime=$newstarttime, endTime=$endoftime}{station="YKAB1"}{response, instrument, calibration};

change Calibration $YKAB1Cal {station="YKAB0", channel="SHZ", startTime=$newstarttime, endTime=$endoftime}{calibrationFactor=.003827e-9, calibrationFrequency=4.0, samplingFrequency=40.0};

There are new instruments at the YKW1 and YKW3 3-component stations.

Create a new polezero.

read PoleZero $YKAW1PZ {"/home/sheila/db/yka/YKABB2014.pz"};

Create a new digitiser and sensor for YKW1 BHZ, then new sensors:

add Digitiser $YKAW1D {$newstarttime, $endoftime, "Guralp","DM24","",1,40.0,40.0,1.0,};

add Sensor $YKAW1ES {$newstarttime, $endoftime, "Guralp","CMG-3T","",3,"m",1,1118403,};

(etc.)

13 Appendix B – Notes on Python API

13.1 Error handling

The Python API (BEAM, 2013b) has two forms, one that uses exceptions, the other that passes an error flag out of functions. These are mutually exclusive and require libraries bdslibe or bdslib, respectively, to be loaded. When bdslibe is loaded, code has to be written thus:

try:

<result = function(params)>;

(etc.)

except exceptionBError as <variable>:

<do the exception actions, e.g. print error message, then exit, or “raise <variable>”;

The function calls in this form have one fewer return values than when bdslib is loaded: then, the functions return, as first return value, an object containing error information (which I have called “err”). This object has to be created beforehand with a statement like “err = BError()”. Programming then follows the form:

(err, result) = function(params)

if (err):

<do the actions in case of error, e.g. print error message, then exit>;

Both the BError object “err” and the exceptionBError object contain information about the error, which is accessed by the method “err.getString()” for the BError object “err” or class variables “e.number” and “e.string” of the exceptionBError object.

13.2 Creating or updating metadata items

The functions for updating or creating “station” and “location” elements, stationUpdate and locationUpdate, generally take three parameters, the flag “append”, the object containing the station or location information, and the ID number. If “append” is true (value 1 in Python), then the ID number should be omitted (i.e. invoke the function with two parameters, “true” (1) and the information object), as the program will use the next available one (and return it from the function). If “append” is false then the function updates an existing element, and the ID number then needs to be specified (which means you need to have found it out earlier).

13.3 Reading metadata from standard files

The BDS can read metadata from the following standard-format files:

- SAC pole-zero;

- Raw FAP file (3 columns ASCII floating-point);
- IDC standard station response files (which contain pole-zero, FAP and FIR elements);
- SEED files (with data or dataless headers).

The object class DataFileResponse, method “getMetaData”, is used to read the first three, while class DataFileSeed, method “getMetaData”, is used for SEED files (the file has to be opened first with the “open” method of either class, and closed aftewards with the “close” method). “getMetaData” returns an array of objects of type “ChannelInfos”, which is an object that contains an array of objects of type “ChannelInfo”. Each individual “ChannelInfo” object contains the metadata for one channel for a single timespan. This allows a file (usually a SEED file) that contains several entries with different timespans for the same channel to be parsed. The output of “getMetaData” can be thought of as a two-dimensional array, with columns corresponding to channels and rows corresponding to timespans.

The other three formats tend to contain only a single entry, effectively metadata for a single channel for an open-ended timespan. The [0][0] element of the return from “getMetaData” is then the only one filled. In C++ the pole-zero response can be extracted as an object of class “Response” by a statement:

```
Response&       r = ci.channels[0][0].responses[ i];
```

where “ci” is the array of ChannelInfos objects , ci.channels[0][0] refers to the “channels” member of the first element of the array, which in turn contains an array of i “responses” corresponding to the stages of the response of that channel. For a SAC pole-zero file or a raw FAP file there is only one stage, but an IDC response file may contain a multi-stage response, e.g. a pole-zero instrument response followed by a cascade digitiser with a FIR filter at each level of the cascade.

13.4 Database update management by transactions

When you write a script to amend the database, the calls “xx.transactionStart()” and “xx.transactionEnd()” (where xx is the name of the instance of AdminAccess that you created to carry out the changes) can be put at the beginning and end of the script. All changes to the database made between the two calls are saved up and made all at once when the “transactionEnd” call is reached. This prevents a faulty script from leaving a half-updated database, but interferes with debugging because you can't tell which alterations did work.

14 Appendix C – class list of classes used by bdsMetadata (listed in include/BdsD.h)

#Channel Members

1 startTime

2 endTime

3 network

4 station
5 channel
6 channelType
7 channelAux
8 dataType
9 description
#ChannelInstrument Members
1 startTime
2 endTime
3 channelId
4 source
5 digitiserId
6 sensorId
#Calibration Members
1 startTime
2 endTime
3 network
4 station
5 channel
6 source
7 name
8 samplingFrequency
9 calibrationFrequency
10 calibrationFactor
11 calibrationUnits
12 depth
13 horizontalAngle
14 verticalAngle
#Response Members
1 startTime
2 endTime
3 network
4 station
5 channel
6 source

7 stage
8 name
9 type
10 gain
11 gainFrequency
12 stageType
13 decimation
14 symmetry
15 description
16 measured
17 response
#Digitiser Members
1 startTime
2 endTime
3 name
4 type
5 serialNumber
6 numberChannels
7 baseSamplingFrequency
8 initialSamplingFrequency
9 gain
10 shared
#Sensor Members
1 startTime
2 endTime
3 name
4 type
5 serialNumber
6 numberChannels
7 gainUnits
8 gain
9 oldId
10 shared
#Station Members
1 name

2 alias
3 type
4 description
5 channels
#Location Members
1 startTime
2 endTime
3 network
4 station
5 datum
6 longitude
7 latitude
8 elevation
9 arrayOffsetEast
10 arrayOffsetNorth
#Source Members
1 source
2 sourceMeta
3 alias
4 description
#Network Members
1 network
2 description
3 stations