User's Manual for the QMW Validation Browser (QVB)

Markus Fränz and Anthony R. Hare
Astronomy Unit
Queen Mary & Westfield College
Mile End Road, London E1 4NS, U.K.
E-mail: csc_support@qmw.ac.uk or M.Fraenz@qmw.ac.uk
WWW: http://www.space-plasma.qmw.ac.uk/

Document Status Sheet			
1. Document Title: **QMW Validation Browser, User's manual**			
2. Document Reference Number: **CU-QMW-MA-0005**			
3. Issue	4. Revision	5. Date	6. Reason for Change
1	0	28 Jun 95	QVB 1.0
1	1	2 Aug 95	QVB 1.1
2	0	12 Feb 96	QVB 2.0
3	0	18 Jun 96	QVB 3.0
3	1	8 Aug 96	QVB 3.1
4	0	30 Jun 97	QVB 4.0
4	1	2 Feb 98	QVB 4.1
4	2	10 Jul 98	QVB 4.2
4	3	7 Jul 99	QVB 4.3
4	5	17 Apr 00	QVB 4.5

Contents

1 Introduction

1.1 Background

This document gives instructions for installing and using the QMW Validation Browser (QVB) application developed at Queen Mary and Westfield College (QMW).
The QVB provides a graphical user-interface for inspecting the metadata and data of a Common Data Format (CDF) file, and intelligently handles many of the International Solar Terrestrial Physics (ISTP) and Cluster Science Data System (CSDS) standard attributes if they are present in the file. It supports retrieval and plotting of all scalar or array time series from such a CDF file. The QVB was developed by Markus Fränz with support from the Cluster Science Group at QMW. It makes use of routines from the QCDF Browser Version 1.0 (22 Dec 1994) developed by Anthony R. Hare at QMW.
The release 4.5 software is written in 'Interactive Data Language (IDL)' (Trademark) by Research Systems, Inc. (RSI) and will only run on platforms which have the IDL compiler installed. The README_QVB file included with the distribution will specify version numbers against which the software was built. The IDL source code for QVB is also available for transferring QVB to other platforms. See also the section '5.1 Requirements and Restrictions'.

1.2 QVB Release 4.5 Capabilities

The QVB Release 4.5 should fulfill following tasks:

- Search and open a CDF file for reading only.
- Give general information on file structure and content:
  - List all global and variable attributes and their content.
  - List of all variables and their content.
  - Show the contents of each record.
- Plotting:
  - Take the time dependent variables of the selected file or search all CDF files in a directory and its subdirectories for time dependent variables and allow the selection of a subset of these variables for plotting.
  - Plotting selected time dependent variables for a user defined time interval on screen. The plot can be saved in PostScript format or in qft-flatfile format.
  - Provide an GUI for setting plot parameters for all selected variables. The plot options include simple linear and logarithmic plots, overplot of multicomponent variables, colour spectrograms and bit pattern plots.
  - Provide an advanced plot interval editor.

- Validation:
  Development of QVB for Cluster Validation has been transferred to JSOC at RAL under the acronym CVB as part of the http://cdhf.bnsc.rl.ac.uk:8000/. CVB is essentially build on QVB version 4.2. The Validation GUI is not further maintained in QVB

1.3 What's new in Release 4.5

- QVB4.5 comes with a separate Data Mapping Tool which stores CDF metadata for all time dependent variables found in a directory in an IDL binary saveset. The tool can be invoked from within QVB or as a separate command line application.
- QVB4.5 has been tested on a MSWindows system by Patrick Canu (Patrick.Canu@cetp.ipsl.fr).
- Support of multiple timelines in same CDF file (Wind mission data).
- Support of compressed savesets.
- QVB now takes the command line parameters 'path' and 'file'. If these are provided they will override the CDF_PATH defined in QVB.cfg and open up the specified CDF file on startup.

The last version released for use by the Cluster community was Release 4.3 (7 Jul 1999).
A version 4.0 QVB.cfg file can still be used in QVB4.5. On most sites a replacement of the QVB.sav saveset will be sufficient.

1.4 Disclaimer: QVB, QSAS and CVB

The QVB is *not* part of the QSAS Science Analysis System for Cluster, which is also being developed at QMW. It is a stand-alone application for viewing the contents of ISTP standard CDF files. It is also *no longer* intended to support CDF validation for the Cluster mission. The Cluster Validation Browser CVB at http://cdhf.bnsc.rl.ac.uk:8000/is the tool designated for that purpose.

1.5 Acknowledgement

The authors thank the Cluster Science group at QMW and the European Cluster science community for critical testing and many suggestions improving the software.

2 Installation

This section details the procedure for installing the QVB on your system, starting from the distributed kit of binaries and startup files, some of which must be customized before use. If this has already been done for you, then you can skip this section. Ask the person who installed the Validation Browser what the local commands are to run it, and where to find the essential startup files to copy into your home directory. Then go to the next section, 'Operation', to learn how to use the Validation Browser.

2.1 Obtaining the distribution kit

The QVB kit for IDL-Unix and IDL-OpenVMS environments is available via World Wide Web URL http://www.space-plasma.qmw.ac.uk/ which also contains an electronic up-to-date version of this manual. The Browser files are in the compressed tar-format archive QVB_4_51.tar.gz and in the zipped archive QVB_4_51.zip; if you need example CDF-files for testing purposes you should also copy the archive CDF_test_files.tar.gz, which contains sample CSDS-like CDF files that you can examine with the QVB. In case of difficulty obtaining the files, please contact csc_support@qmw.ac.uk.
To unpack the files, place the archive file in a suitable directory , and issue the unix commands gunzip QVB_4_51.tar.gz and % tar xvf QVB_4_51.tar, or unzip QVB_4_51.zip.
The files will be unpacked into the current directory (from here on called QVB_DIRECTORY). Note that QVB is no longer unpacked into a subdirectory to allow installation on different platforms.
Your unpacked QVB_4_51.tar should yield the following files (change those files marked CHANGE):

```
README_QVB      ! contains copyright, late release notes, etc
                ! please READ IT before proceeding further
QVB_manual.ps   ! postscript file of manual text
qvb             ! example c-shell script for automating startup (CHANGE)
qvbmap          ! example c-shell script for mapping CDF files (CHANGE)
qvbmap.start    ! IDL start up file for mapping CDF files (CHANGE)
QVB.cfg         ! configuration file containing the default
                ! directory paths for the CDF files
                ! and the Postscript output and
                ! several other optional settings.
                ! The user should copy this file to
                ! his home directory (CHANGE)
QVB.hlp         ! Help file for adding options to your
                ! QVB.cfg file.
QVB_IDL_startup ! IDL commands to restore and run browser
QVB.sav         ! compiled IDL procedures save-file
```

2.2 Obtaining QVB for Other Environments

The standard QVB4.5 saveset distribution was developed under IDL version 5.3 on a Sun Solaris platform. The saveset is expected to operate on Unix, OpenVMS and MSWindows platforms running IDL5.3, but there will be minor differences in operation due to the differences in the IDL implementations.

If the saveset does not operate on your platform you can download the source code called 'QIDL Source code' from the same website. The source code is available as compressed tar-set or as zip-set. You have the choice to either install the complete source code in one directory or obtaining the plot and cdf handling packages separately (see also the *QSAS and QVB IDL Routines. Reference Guide.*). If you wish to obtain the code in another form please contact the authors.

Once you have unzipped or untarred the code you may edit the file qvb_make.pro contained in the set. This file is an IDL startup file producing a QVB saveset. The lines to be changed for your system are marked and commented in the file. Once you have done this the file is executed by the command:

```
idl qvb_make
```

If execution succeeds it will produce a saveset file QVB.sav in your current directory. Replace the file with the same name in your saveset distribution.

2.3 Updating from older versions of QVB

If you are updating from an older version than QVB3.0 you have to setup a new QVB.cfg file which comes with the distribution.
If you are updating from QVB3.0 or QVB3.1 note that the configuration file QVB.cfg does no longer contain a plot option section since this has been replaced by the GUI setting. If you want to reuse your old QVB.cfg file please delete all lines with the prefixes A, V and N from your QVB.cfg file.

If you are updating from QVB 3.1 you may also reuse your QVB_IDL_startup and start_QVB shell script though the new shell script qvb has extended options. On non-Sun systems regard the note on widget-fonts in section 2.7.

If you are updating from QVB4.0 you can keep your current QVB.cfg file. If you would like to use QVB for validation regard the extended parameter list in section 2.5.

2.4 Installing the files on your system

First, make a backup copy of all the original files that you received in a safe place before modifying anything. Read the README_QVB file which should contain a more detailed packing-

list than that given above, and may also give revised or additional installation instructions. Check that your operating system and IDL versions satisfy the minimum requirements specified in the README_QVB.
Then, make a copy of the configuration file QVB.cfg to your home directory. You should customize this file for your system and preferences as described below. You will find a similar description in the QVB.hlp, a copy of which should stay in the QVB_DIRECTORY since it can be viewed interactively. You should also make a private copy of the qvb shell script in your bin or other preferred path, and customize it to automate the process of launching the Validation Browser on your chosen machine. The compiled QVB.sav file and the IDL startup file QVB_IDL_startup should remain in QVB_DIRECTORY, as individual users do not need their own copies. The test CDF files should also be stored centrally.

2.5 QVB.cfg file

The Validation Browser uses a startup file, which should be present in the current directory or in the user's home directory. Otherwise QVB searches for the file in the QVB_DIRECTORY. It is a text file, which you may examine and customize.

```
QMW Validation Browser DEFAULT SETTINGS FILE: QVB.cfg
comment-lines start with "!", all lines not starting with
the letters P, M, F are ignored.
Path section
————————

This section contains the initial paths for the CDF-datafiles ("CDF_path") the PostScript output ("PS_dir")
the users local QVB path for auxiliary files ("QVB_local").
Lines must start with the qualifier "P". The syntax is "qualifier parameter content".
  P    CDF_path      ***PUT CDF PATH HERE***: e.g. /CDF_test_files/
  P    PS_dir        ***PUT PATH HERE***: e.g. your home dir
  P    QVB_LOCAL     ***PUT PATH FOR QVB LOGGING AND TEMPORARY CON-
                     TROL FILES HERE***
File section [optional]
————————

This section contains the default filenames for Plot parameter saving and Annotation saving.
Lines must start with the qualifier "F". The syntax is "qualifier parameter content". If no pathname is
supplied QVB assumes QVB_local as path.
  F    Plot_Save_File    qvbplot.sav
  F    Annotation_file   annotation.txt

Validation Section
——————————

This section contains optional parameters essential when QVB is used for validation and not for brows-
ing only.
There are five Mode parameters:
```

```
M   MISSION              CLUSTER | EQUATOR-S ! predefined settings exist
                         ! only for these otherwise define
                         ! INST_LIST
M   VALIDATION_MODE      OFF | ON ! determines whether QVB runs in
                         ! Validation Mode [default = OFF].
M   LOGGING_MODE         OFF | ON ! determines whether QVB runs in
                         ! Logging Mode [default = OFF].
M   VALIDATOR            name ! determines default validator attribute.
M   INST                 ASP|CIS|DWP|EDI|EFW|FGM|PEA|RAP|STA|WHI
                         ! default instrument acro for CLUSTER
M   INST                 MAM|EDI|3DA|EPI|ESIC|SFD|PCD
                         ! default instrument acros for EQUATOR-S
M   INST_LIST            a,b,c,d... ! provide comma separated list
                         ! for other missions
F   LOCAL_CAVEAT_FILE    path/filename ! Filepath and name for storing your own
                         ! set of caveat strings [mycaveats.txt].
M   DATA_MAPPING         OFF | ON ! use QVB data maps
M   TEXTEDITOR           command ! command will be launched by SPAWN
                         ! using QVB.cfg as argument [textedit].
```

The Optional Section for customizing the plot layout contained in QVB 3.1 has been replaced by a plot GUI as described in section 4.9.
Use the interactive **Help** button on the QVB main window to display the help file for changing QVB.cfg .

2.6 Example shell scripts: qvb,qvbmap

2.6.1 Unix C-shell

For starting QVB you may use the c-shell script qvb which comes with the distribution. You have to edit the following lines to point to your QVB installation directory.:

```
# EDIT FOLLOWING to point to IDL LICENCE on YOUR SYSTEM,
# or comment-out if this is already set up at user-login.

setenv LM_LICENSE_FILE /usr/local/idl/licenses/license.dat

# Invoke IDL: EDIT COMMAND LINES below as necessary for YOUR SYSTEM.

setenv QVB_DIRECTORY ***PUT QVB PATH HERE***

/usr/local/bin/idl ***PUT QVB PATH HERE***/QVB_IDL_startup
```

After customizing this file you may define an alias in your .cshrc file of the following form:

```
alias QVB 'rsh -n ***IDLmachine*** ***QVB PATH***/qvb'
```

where IDLmachine is the remote machine running IDL and QVB PATH is the directory where you installed QVB. Afterwards you can run QVB from every location using the command QVB host, where host is your displaying machine name.

Command line parameters The c-shell script qvb has following usage:

qvb [-d DISPLAY] [-p pathname] [filename]

where DISPLAY is the name of your terminal, pathname overrides the CDF_path setting in QVB.cfg and filename is an optional CDF file to open on startup.

The c-shell script qvbmap can be used to invoke the QVB Data Mapping Tool as a batch job (see section 3.4). Dependent on your configuration you may also have to customize this file. It starts IDL with the startup file qvbmap.start which should not need customization. The usage is:

qvbmap [pathname]

where pathname is the directory to map. Note that you need write access to that directory.

2.6.2 OpenVMS

If IDL is setup on your system and the display is set to your terminal, all you should do to run QVB is adding following lines to your LOGIN.COM file where PATH is the QVB_DIRECTORY holding your QVB installation.:

```
$ define qvb_directory PATH
$ qvb :== idl qvb_directory:qvb_idl_startup
```

The command qvb will from then on start QVB.

2.7 QVB_IDL_startup

This file is called by qvb as an IDL startup file, restoring the saveset QVB.sav and running the QVB procedure. If the QVB_DIRECTORY has been defined and contains the saveset this file must not be changed except your terminal does not support the default widget font "screen-bold" used by QVB. If this is the case you may wish to put in a font name of your choice at the line

```
QVB,FONT=" .
```

A list of fonts available on your terminal can be obtained by using the IDL command

```
DEVICE,FONT='*',GET_FONTNAMES=fonts
```

as described in the IDL Reference Guide.

2.8 IDL Colours

You can define your own set of **Widget Colours** valid for all QVB interfaces. This is done by changing the .Xdefaults(Unix) or DEC$SM_GENERAL.DAT (VMS) file in your home direc-

tory to specify the IDL widget background colours. Add the following lines to that file:

```
Idl*background: lightgrey
Idl*qvb*background: #b0b0d0
Idl*qvb*button*background: #b0b0ff
Idl*qvb*info*background: #b0d0b0
Idl*qvb*text*background: #d0b0b0
Idl*qvb*label*background: grey
```

This takes effect when you start/restart the window manager, or you may do
% xrdb .Xdefaults (Unix) at the command line, and then restart the IDL application to see it take effect immediately. There is a standard set of X colours, and many English colour names are supported, like red, purple, green, or you can make your own colour hues.

The colours appearing in the **Plot Window** can be changed on runtime using the QVB Plot user interface (see section 4.8), but see also the section 5.2.8 on terminal dependent problems with IDL Colours.

2.9 Test CDF files

The archive CDF_test_files.tar should unpack to a README file (please read it!) and a selection of Cluster-CDF-like test files. Some contain CSDS standard attributes and minimal fake data to allow testing of syntax only; others have been populated with real data from other sources. The accompanying README may contain more information than incorporated below.

README file from CDF_test_files.tar:
==========

CDF test files to accompany QVB Release 4.0 30 Jun 1997
==========

This archive contains sample CDF files that can be examined using the QMW Validation Browser.

There are 15 sample "Prime Parameter" and "Summary Parameter" files (one PP and one SP for each UK-led Cluster instrument), dated 23 January 1995. These CDF files conform to CDF skeletons V1.6, and are for testing CSDS standard attributes and syntax. Each contains ten records of fake data:

C1_PP_DWP_19950123_V00.cdf ! "Prime Parameter" file
CL_SP_DWP_19950123_V00.cdf ! "Summary Parameter" file

C1_PP_FGM_19950123_V00.cdf
CL_SP_FGM_19950123_V00.cdf

C1_PP_PEA_19950123_V00.cdf
CL_SP_PEA_19950123_V00.cdf

There are additionally five CDF files for the FGM instrument, populated with real Ampte IRM magnetometer data. The four PP files, one per Cluster spacecraft, contain real independent vector data taken

by Ampte IRM at different times, and will not correlate as one might expect of real Cluster data. The SP file has been derived by averaging data for one spacecraft. Note that the files are dated 18 October 1968, and that all records have status 255, to emphasize that these are not real Cluster data. The spin timing is on second boundaries, without the drift expected for real data timed to millisecond accuracy

C1_PP_FGM_19681018_V00.cdf
C2_PP_FGM_19681018_V00.cdf
C3_PP_FGM_19681018_V00.cdf
C4_PP_FGM_19681018_V00.cdf
CL_SP_FGM_19681018_V00.cdf

3 Operation

Unix The QVB is either started using an IDL Start Up file or by starting an IDL session. If you have defined a shell script and command alias as explained in section 2.6 start it now by typing QVB host where host is the host name of your terminal. You can specify a default host by editing the qvb script.
If you are not using a shell script and if the environment variable QVB_DIRECTORY is not set in your .login or .cshrc file define it now by typing:
setenv QVB_DIRECTORY ***path***

OpenVMS The QVB is either started using an IDL Start Up file or by starting an IDL session. If you have defined a command as explained in section 2.6 start it now by typing QVB

If you are not using an IDL Start Up and if the environment variable QVB_DIRECTORY is not set in your LOGIN.COM file define it now by typing:
$ define QVB_DIRECTORY ***path***

Start the IDL application using your local recipe for this. Load and execute the saved Browser routines using the IDL commands
IDL> RESTORE,GETENV('QVB_DIRECTORY')+"QVB.sav"
IDL> QVB

The main Validation Browser control panel should appear on your display shortly afterwards. If it does not, check the IDL-window for explanatory error messages, *e.g.* failure to open your terminal display.

The qvb shell script also allows the command line parameters '-p pathname' and 'file'. If these are provided they will override the CDF_PATH defined in QVB.cfg and open up the specified CDF file on startup (see section 2.6).

3.1 QVB main control panel

The lowest part of the panel contains a scrolling text window where informational and error messages are written. The panel is built using X-widgets, which are manipulated using only the *left* button of a three-button mouse. Button-widgets are shaded to appear raised from the panel surface, whereas areas for click-and-type user-input appear recessed. Pop-up menus look like buttons with a line border around the label: click on the label to pop-up the menu, which then stays up until you click on a menu item. Inactive buttons and menus are greyed-out to indicate that they are unresponsive.

If you have defined IDL widget colours as explained in section 2.8 all buttons will appear in light blue. All editable or clickable text areas in purple, fixed labels in grey and information

windows in green.

Before you can do anything else using QVB, you must

- **either** instruct it to select a CDF file using the **File** | **Select** menu item. This option will launch the **File Selection** interface. See the separate section '3.2 The File Selector' for its operation. The **File** | **Open** menu item will open a previously selected file. If you do not want to look at the metadata contained in the file you can just **Accept** the file for plotting.

- **or** Select a CDF variable using the **Variables** button. The **Variables** button allows to select time dependent variables for plotting from a larger set of files. See the separate section '3.3 The Variable Selector' for its operation.

After selecting a CDF file or Variables the **Plot** button will launch the Plot User Interface **QVB_pui** which allows plotting of time dependent data. See the section '4 Plotting' for its operation.

3.1.1 QVB.cfg

This button opens a pulldown menu for editing and loading the QVB.cfg file. This allows the change of setup options during run-time. The menu item **Edit** will launch the text editor specified in your QVB.cfg file with the file as argument. There can always be only one QVB-launched edit process active.

The menu item **Help** views the QVB.hlp file. If you have saved the changes to your QVB.cfg file use the menu item **Load** to reload the file. From now on the changes will take effect. The QVB-launched edit process will be killed if you quit QVB. See also section '5.2.10 Problems editing QVB.cfg'.

3.1.2 Help

This button opens a pulldown menu giving a short one line information on the operation of each item in the QVB main window.

3.1.3 Quitting QVB

To quit QVB, click the **File** | **Quit QVB** menu item. If QVB runs in logging mode (see section 3.7) the logging file will be closed. If QVB runs in validation mode (see section 3.6) and there is a temporary validation file opened QVB will not close down but issue a warning message allowing the user to switch back to the validation operation. If QVB is closed down using the Xwindow 'Quit' menu item on the QVB main window the temporary validation file will not be processed.

3.2 The File Selector

The **File Selector** is a user interface for the selection of a single file which might be examined by QVB or a list of files for validation. The bottom line of the interface contains information and error messages. Always check this line when there is an unexpected behaviour. The top line contains path and name of the currently selected file. A short description of all entries in this interface will be shown if you use the **Help** button.

3.2.1 File: Accept or Open

This window shows the currently selected file including its path. The **File:** | **Open** operation will call the CDF File Browser with the currently selected file as argument. The user may also edit the filename if this is more convenient than searching through the **Matching** list. If the initial opening succeeded respective messages will appear in the **Info** windows of both interfaces. The **File:** | **Accept** operation will put the dependent variables found in the file on the Plot Variables list of the plot interface. This option is useful when you are not interested in looking at the metadata but would like to proceed to the plot operation immediately. Simply use the **File:** | **Open** entry on the QVB main panel to look at the metadata at a later stage. The **File** | **Dismiss** menu item makes the **File Selection** interface invisible to the user. It can be relaunched with the same window contents by using the **File** | **Select** menu item in the QVB Main interface. The window contents will only be lost when QVB is quitted.

3.2.2 Path:

The default path appearing on the top is defined in your QVB.cfg-file. You may change this path using the mouse and keyboard buttons. The path will be accepted when you hit the RETURN key. If the selected path is not a valid path on your system an error message will appear.

3.2.3 Filter:

The **Filter** text determines which subset of files in the current subdirectory is shown in the **Matching Files** window. You may enter any textstring allowed as file filter on your OS-system as text. Note that only filters referring to the current directory are allowed, i.e. '*' will not show files in subdirectories in contrast to unix 'ls' behaviour.

Operation examples (assuming that all files are in same subdirectory):

Filter Text	Matching Files
'*.cdf'	all CDF files in current directory
'C1*.cdf'	Cluster 1 files only
'C1*.cdf C2*.cdf'	Cluster 1&2 files only
'*PP*.cdf'	Prime Paramter files only
'*V02.cdf'	Version 2 files only
'*960507*.cdf'	Cluster Start Date files only

3.2.4 Sorting

This pull down menu allows sorting of **Matching** and **Selected Files** either alphabetically or numerically.

- **No sort** means that **Matching Files** are sorted as given by Unix 'ls' command.
- **Alphabetic** means all files are sorted by non-numeric characters, then by numeric characters. This will sort files by Prime and Summary Parameter, then by spacecraft, then by date, then by version.
- **Numeric** means all files are sorted by numeric characters, then by non-numeric characters, but with an offset of 2 characters, i.e. sorting starts with the third character of each item. This will sort files with CDHF names by date, then by version, then by spacecraft, then by Prime and Summary Parameter.

3.2.5 Subdirectories

Doubleclicking on these entries allows switching between current directories. It changes the **Path** entry. See also section '5.2.12 Problems using the File Selector'.

3.2.6 Map All and Map This

These buttons invoke the QVB data mapping tool. (See section '3.4'.) The **Map All** operation will try to create QVB datamaps in all subdirectories listed. WARNING: This may take a long time. The **Map This** operation will only map the currently selected subdirectory. These buttons are only visible if you have selected M DATA_MAPPING ON in your QVB.cfg file.

3.2.7 Matching Files

Shows the list of files matching with the selected **Filter** in the directory defined by **Path**. Doubleclicking on an entry moves the entry to the **Selected Files** and to the **File:** window.

3.2.8 Select All and Clear

Select All moves all files from the **Matching Files** list and moves it back to the **Selected Files** list. Clear will empty **Matching Files** list. You may repopulate the list by hitting the RETURN key in the **Filter** or **Path** window.

3.2.9 Selected Files

Shows the list of files selected from the **Matching Files** window. If the current **Path** is changed the **Selected Files** entries will remain. This allows the selection of files from different directories in one set. Their path information is stored internally. The list of selected files is picked up when you start the validation process. Clicking on an entry copies the entry to the **File:** window.

3.2.10 Remove and Clear

Remove removes the highlighted entry from the **Selected Files** list and moves it back to the **Matching Files** window if the file's path agrees with the current **Path**, otherwise the entry is completely removed from the selection. Clear does the same for all entries in the **Selected Files** list.

3.2.11 Validation (Sel. Files)

This button launches the Validation user interface with the logical file IDs of all **Selected Files** as argument. See the section '3.6 Validation' for its operation. This button will only be visible if QVB is used in Validation mode.

3.3 Selecting Variables

The **Variable Selector** is a user interface for the selection of time dependent variables from a set of CDF files. This interface works by scanning all CDF files in a given subdirectory for dependent variables. You may then select specific variables from this list for plotting. This approach is an alternative to the **File Selector** in the case that either do not look at the files metadata or want to see a long time plot or a relation between variables in different files.
The bottom line of the interface contains information and error messages. Always check this line when there is an unexpected behaviour. A short description of all entries in this interface will be shown if you use the **Help** button.

Capacity Limits **If your data directory contains more than 50 CDF files there will be a confirmation dialog before scanning the files since scanning a large number of files can take several minutes. There is a limit of 10000 files you can scan at one time, but you may reach the physical limit of your machine's memory, if you are scanning several thousand files in one go.** If you have write permission for the directories containing the CDF files you may map the CDF files using the QVB Mapping Tool (see section 3.4).

3.3.1 Variables

The **Variables** menu allows either to accept the variables from the **Selected Variables** list or to dismiss the **Variable Selector**. After dismissing it can be relaunched with the same window contents by using the **Variables** button in the QVB Main interface. The window contents will only be lost when QVB is quitted.

3.3.2 Sorting

This pull down menu allows sorting of **Dependent** and **Selected Variables** either alphabetically or numerically.

- **Alphabetic** means all variables are sorted by non-numeric characters, then by numeric characters.
- **Numeric** means all variables are sorted by numeric characters, then by non-numeric characters.

3.3.3 Time Interval Setting

The **Start** and **Finish** fields show the start and finish time spanning all dependent variables found in all CDF files scanned. If the **Show/Set** button to the right of that field is pushed you will be able to set the time interval before scanning a subdirectory. **This is strongly recommended when scanning a large number of files.**

3.3.4 Path:

The default path appearing on the top is defined in your QVB.cfg-file. You may change this path using the mouse and keyboard buttons. The path will be accepted when you hit the RETURN key. If the selected path is not a valid path on your system an error message will appear. If the directory defined by **path** contains CDF files, the number of these files will appear in the message windows and all the files will be scanned for time dependent variables.

3.3.5 Subdirectories

Doubleclicking on these entries allows switching between current directories. It changes the **Path** entry. Double Clicking on the ***** entry will cause a scan of all subdirectory trees of the current level. **Be careful when using this button since scanning a deep directory structure may take a long time.** Also set the time interval before using this button. See also section '5.2.12 Problems using the File Selector'.

3.3.6 Map All

This button invokes the QVB data mapping tool (see section '3.4'). The **Map All** operation will try to create QVB datamaps in all subdirectories listed. WARNING: This may take a long time. If you would like to map a single directory use the **Map This** button in the File Selector. These buttons are only visible if you have selected M DATA_MAPPING ON in your QVB.cfg file.

3.3.7 Dependent Variables

Shows the list of time dependent variables matching with the selected **Filter** in the directory defined by **Path** and found during previous scans. 'Time dependent variables' are all record varying variables which have the DEPEND_0 attribute set to an Epoch variable. In contrast to the **File Selector**'s behaviour files which do not use this attribute will be rejected .
Single Clicking on an entry shows the FIELDNAM attribute of the respective variable. Doubleclicking on an entry moves the entry to the **Selected Variables** window. Use the **Sorting** menu to sort Dependent Variables and the **Filter** to subset or clear the list.

3.3.8 Filter:

The **Filter** text determines which subset of variables is shown in the **Variables** window. An empty or '*' string in this field means that all time dependent variables found while scanning CDF files are shown. The filter is NOT case sensitive and '*' is the only wildcard allowed. To refill the **Dependent Variable** window after filtering you have to rescan the directories using the **Path** or **Subdirectories** fields with a broader filter.

3.3.9 Selected Variables

Shows the list of variables selected from the **Dependent Variables** window. This list is passed to the plot interface when choosing the **Variables** | **Accept** menu item. **Remove** removes

the highlighted entry from the **Selected Variables** list and moves it back to the **Dependent Variables** window. **Clear** does the same for all entries in the **Selected Variables** list.

3.4 The QVB Data Mapping Tool

To shorten the time for data ingestion you may produce QVB Datamaps using the QVB Data Mapping Tool. The tool will extract metadata and time ranges for all time dependent variables in all CDF files contained in one directory. The tool can be invoked:

- from the IDL command line by typing:

  ```
  files=FINDFILE('*.cdf',PATH=path)
  success=QCDF_DATASAV(files,1)
  ```

  path is the directory to map.

- by using the c-shell script qvbmap path. path is the directory to map (see section 2.6).

- The **Map All** and **Map This** buttons in the File Selector and Variable Selector will invoke the QVB Data Mapping Tool. The **Map All** operation will try to create mapping files in all subdirectories listed. WARNING: This may take a long time. The **Map This** operation will only map the currently selected subdirectory. These buttons are only visible if you have selected M DATA_MAPPING ON in your QVB.cfg file.

The QVB Data Mapping Tool will create a compressed IDL saveset file under the name QVB-datamap.sav in the directory path. You must have write permission on path to use the tool. Whenever you add CDF files to path you should repeat the mapping operation.

3.5 The CDF File Browser

After selecting a CDF file and pushing the open button on either the QVB main control panel or on the CDF File Selector the CDF File Browser will pop up. This GUI allows viewing all the meta data of a CDF file and its content record by record. The name of the CDF file will appear in the top center. Pressing the **File | CDF Info** menu item gives a general information about structure, size and creation software of the currently opened CDF file. The **File | Close** menu item will hide the The CDF File Browser. It can be reopened using the **File | Open** menu item on the QVB main interface.

3.5.1 The QVB Data Window

After opening a CDF file all information you request will appear in a separate text window which is resizable. Note that all other QVB windows except the plot windows are not resizable.

3.5.2 CDF: and Validation

This line shows the CDF file currently opened by QVB.
The **Validation** button will only be visible when QVB runs in validation mode (as defined in QVB.cfg). It will launch the Validation user interface with the currently opened CDF file ID as argument. See the section '3.6 Validation' for its operation.

3.5.3 CDF global and variable attributes

When a CDF file is opened, the QVB queries the CDF file about the metadata and data present in the file. It uses this information to populate the lists of **Global Attributes** and **Variables** on the left of the main panel. The CSDS-standard attributes are fully-specified by: *CSDS Standard CDF File Format* (DS-QMW-TN-0003). You will get general information about size and structure of the opened file using the **CDF Info** button.
Global Attributes are metadata that apply to a whole CDF file: you may retrieve their values and write them to the data window by clicking on individual entries in the list, or click the button below it to display all of them in sequence.
Variable Names are either record varying or non varying. Clicking on a variable will give information on this variable and will populate the **Variable Attribute** list with the variable attributes defined for this variable.
Variable Attributes apply to individual variables in the CDF file, and in CSDS these are used to hold information on valid ranges of variables, their units, and scaling and labeling hints for plotting routines, *etc.* Clicking on an individual entry in the list will retrieve the value of that attribute for the currently selected variable, or click on the button beneath the list to retrieve them all for the currently selected variable. The currently selected variable is set by clicking on an entry in the **Variable Names** list.

3.5.4 Browsing individual records

Individual records in the CDF file may be retrieved and written to the text window using the **Browse Record** slider. Click on the sliders button to show the contents of the respective record. For record 0 this will contain the contents of the non-varying variables also. Move the button to scan the file contents very fast. This is done in steps of 10-30 records depending on the file size. To access individual records click to the right or left of the slider button. Text data are displayed as single characters since they are stored as character arrays and not as strings, and numeric data are written as formatted text, byte values are converted to integer, epoch time values to CSDS time.

3.6 Validation

QVB does no longer support CDF Validation for Cluster II CDF files. It is strongly recommended to use the Cluster Validation Browser CVB
(http://cdhf.bnsc.rl.ac.uk:8000/) for that purpose. QVB can be adapted to produce Validation Control Files in Cluster I format for other missions.
The Validation user interface allows the generation of a Validation Control File in CSDS standard format (see the *CSDS-UI Principal Investigator User Manual* (DS-ESR-SM-003, Issue 3, 14 May 1996). It will be accessible from the **File Selector** or **File Browser** only in QVB Validation mode. It is recommended to define all entries of the Validation Section in your QVB.cfg file before using this interface.
The bottom line of the interface contains information and error messages. Always check this line when there is an unexpected behaviour. A short description of all entries in this interface will be shown if you use the **Help** button.

3.6.1 Environment

Always define the current mission name via the M MISSION entry and M INST entry in QVB.cfg. For CLUSTER and EQUATOR-S instrument acronym lists are predefined.

CLUSTER See note above.
The validation tool has been designed to work on accounts where CSDS UI has been set up and where the cdfvalbatch operation is accessible. That means that the QVB validation for Cluster has to run locally on a cui_ account installed at a CSDS data center. Specifically following directories as defined in DS-ESR-SM-003 must be accessible:$CUI_USR_ROOT/valctl, $CUI_USR_ROOT/log, $CUI_<INST>_CDF_NEW, $CUI_<INST>_CDF_NEW/bck.
The val_<INST>.cfg file is currently not used by QVB since QVB uses the validator name and caveats file location defined in QVB.cfg.

Other missions For other missions than Cluster the validation tool can be used to produce Validation Control Files in CSDS standard format only. In this case it is recommended to define the environment variable $CUI_USR_ROOT pointing to a directory which should contain the subdirectories valctl and log. For other missions than CLUSTER or EQUATOR-S define an instrument acronym list in QVB.cfg (see section 3.1.1).

3.6.2 Operation

The interface starts either with a single logical file ID (as defined in DS_QM_TN_0003) or with a list of logical file IDs as **Selected Files** entry. The different **Caveat** windows allow the selection of caveats to be added to each file in the list of **Selected Files**.

The **Process Sel. Files** button creates a Temporary Validation Control file under the name val_<INST>.tmp in the directory defined by QVB_LOCAL in QVB.cfg and moves the file entries to the **Validated Files** window. Up to now the entries have been validated only in your

local Validation Control file. The operation can be reversed by clicking on a single entry and on the **Remove** button in the **Validated Files** window. This will move the entry back to the **Selected Files** window and will remove the entry from the Temporary Validation Control file. It can now be validated again with different caveats. To put all Validated Files back to the Selected Files list use the Clear button in the Validated Files window. The Temporary Validation Control file will be removed then.

The **Validate!** button will close the Temporary Validation Control file and move it as val_<INST>_yyyymmdd_hhmmss.ctl (with the current time) to the directory defined by $CUI_USR_ROOT/valctl. Then QVB will launch the cdfvalbatch operation with this filename as argument. If the directory is not accessible or the operation terminates without creating all _VAL files in $CUI_<INST>_CDF_NEW/bck the temporary file will not be deleted and QVB will issue a warning message.
On successful completion the content of the validation control file will be appended to QVB_logging.txt. The file contains also the opening time of the respective Temporary Validation Control file. The entries from the **Validated Files** window will be removed. **Thus operation of the Validate! button starts the batch validation irreversibly.** For CLUSTER this means that the cdfvalbatch operation is initialized, for other mission only a 'final' validation control file is produced.

During a QVB session you may dismiss the **Validation** interface and relaunch it to add further files to the Temporary Validation Control file. The current content of the Temporary Validation Control file will be displayed when you press the **View** button. The viewer will not automatically update if the file is changed, so close and relaunch the viewer in this case. If you quit QVB and have not closed the Temporary Validation Control file you will be prompted to either **Validate** or **Edit** the file. If you choose the validate option the operation of the **Validate!** button will be executed. A message containing the name of the file will appear in your command window. The **Edit** option will launch the **Validation** window again. This will allow you to make further changes and **Validate!** the file or to remove it using the **Clear** button.

If there are technical problems (like file access errors) in closing the file you can force the close down of QVB by quitting the Xwindow containing the QVB main widget. In this case and when the QVB session is interrupted by an internal or external error, the Temporary Validation Control file will still contain the information you have stored so far. When you start QVB and the Validation interface again you are asked to reload the existing Temporary Validation Control file. It is NOT recommended to edit the Temporary Validation Control file externally, since editing errors may inhibit the reloading of the file.

3.6.3 Validity: MAJOR

There are three types of **Validity**: MAJOR (caveats), MINOR (caveats) and NONE (no caveats). If there are no caveats selected the files will always be validated with validity NONE, other with the value defined by the **MAJOR/MINOR** button.

3.6.4 INST

This parameter is the CSDS instrument abbreviation as defined in DS_QMW_TN_0003. A single Validation Control File can only contain file IDs for one instrument. Changing INST during run will clear the **Selected File** window. You can restart the Validation interface with file IDs belonging to the newly selected INST. The default INST parameter should be defined in QVB.cfg.

3.6.5 Validator

This parameter is the CSDS Validator global attribute as defined in DS-QMW-TN-0003. The default **Validator** attribute should be defined in QVB.cfg. The maximal length of this entry is 80 characters.

3.6.6 All Caveats

This list shows all caveat entries currently available in your LOCAL_CAVEAT_FILE as defined in QVB.cfg. All non-empty lines not starting with '!' will be regarded as a caveat entry. You can reload, save or extend this file using the **Caveat** button. Clicking on an entry will move the entry to the **Selected Caveats** window. Doubleclicking on an entry will remove the entry from the list (but not from your caveats file).

3.6.7 Selected Caveats

This list shows the caveat entries which will be written to the Validation Control File together with the **Selected Files** entries. Doubleclicking on an entry will move the entry back to the **Add:** window for editing.

3.6.8 Remove and Clear Caveats Buttons

These buttons can be used to empty the **Caveats** windows. They have no effect on the caveats file.

3.6.9 Caveats Button

This menu button can be used to reload, save or extend your LOCAL_CAVEAT_FILE as defined in QVB.cfg.

- **Load** means that the entries in the **All Caveats** window are overwritten by the file content.
- **Save** means that the file is overwritten by the **All Caveats** window content (leaving comment lines unchanged).
- **Add** means that the entries in the **All Caveats** window will be added to the entries in your file.

3.6.10 Add Caveat:

This window may be used to add caveat entries to the **Selected** and **All Caveats** windows (max. 80 characters).

3.6.11 Selected Files

Shows the list of logical file IDs currently selected for validation. These IDs will be written to the Temporary Validation Control File together with the Selected Caveats entries when you use the **Process Sel. Files** button. Doubleclicking on an entry will process this single entry only.

3.6.12 Validated Files

Shows the list of logical file IDs contained in the Temporary Validation File. These IDs will be written to the Final Validation Control File when you use the **Validate!** button. Note that if the file contains several sets of validated IDs with different caveats the **Selected Caveats** window will not show the caveat entries valid for all sets. You may look at previously validated sets by viewing the Temporary Validation Control File with an external viewer.

3.6.13 Remove and Clear Files Buttons

The **Selected Files Remove** button removes the entry from the **Selected Files** list. The **Validated Files Remove** button moves the entry back to the **Selected Files** window and removes it from the Temporary Validation File. The **Clear** buttons can be used to empty the **Files** windows. Note that the **Validated Files Clear** button will remove the Temporary Validation Control File!

3.6.14 Process Sel. Files

The **Process Sel. Files** button creates or extends a Temporary Validation Control file under the name val_INST.tmp in the directory defined by QVB_LOCAL in QVB.cfg and moves the file entries to the **Validated Files** window. The operation creates a set of file IDs together with the current **Validity**, **Validator** and **Selected Caveats** entries. At this stage the entries have been validated only in your local Validation Control file. The operation can be reversed by doubleclicking on single entries in the **Validated Files** window. This will move the entry back to the **Selected Files** window and will remove the entry from the Temporary Validation Control file. It can now be validated again with different caveats.

3.6.15 Validate!

The **Validate!** button will close the Temporary Validation Control file and move it as val_INST_yyyymmdd_hhmmss.ctl (with the current time) to the directory defined by $CUI_USR_ROOT/valctl.

For the CLUSTER CSDS-UI installation QVB will then launch the cdfvalbatch operation with this filename as argument. If the directory is not accessible or the operation terminates without creating all _VAL files in $CUI_<INST>_CDF_NEW/bck the temporary file will not be deleted and QVB will issue a warning message.

On successful completion the content of the validation control file will be appended to QVB_logging.txt. The entries from the **Validated Files** window will be removed. Thus operation of the **Validate!** button starts the batch validation irreversibly.

3.6.16 View

The current content of the Temporary Validation Control file will be displayed when you press the **View** button. The viewer will not automatically update if the file is changed, so close and relaunch the viewer in this case.

3.6.17 Dismiss

This button makes the **Validation** interface invisible to the user. It can be relaunched by using the **Validation** button on the QVB Main or File Selection interface. The window contents will only be lost when QVB is quitted. The **Dismiss** button will NOT close or remove the Temporary Validation Control File.

3.7 Logging

If the entry LOGGINGMODE in your QVB.cfg file is not set to OFF QVB will log the following operations into a file called QVB_logging.txt in the directory QVB_local:

1. Opening of a CDF file.
2. Creation of a postscript or flatfile.
3. Commission of a Validation Control File, including its name and content.

QVB does not make any further use of this file. If the file is removed from its location, QVB will create a new file under the same name. The file QVB_logging.txt will thus increase with time and the user may delete parts of the file which are no longer needed for inspection with an external texteditor.

4 Plotting

In addition to inspecting attributes and individual records, the QVB can retrieve and plot series of time-tagged data versus time. The 'QVB_pui' plot user interface will pop up when pushing the **Plot** button on the QVB main interface. The plot user interface allows the interactive setting of plot parameters like scaling, plot type and distribution of panels over output pages. The plot output can be on screen, on a postscript or flat file. The time interval for plotting can be chosen using the **Time editor** (see section 4.6). The plot annotation can be changed using the annotation tool **QPAN** (see section 4.5).

4.1 Plot Variables

If you have accepted a file or mapped dependent variables before starting the plot user interface all dependent variables of that file or the selected dependent variables will appear in the **Plot Variables** window. If you did not select and accept a file or variables the system will search a plot save file in your QVB_local directory under the name qvbplot.sav or under the name specified in the Plot_Save_File entry in your QVB.cfg file. If there is no such file and you did not accept any selection the list will stay empty.
DoubleClicking on a variable in the **Plot Variables** list will fill the **Selected Variable** window with the plot parameters for that variable. The list determines the order in which variables are plotted. It can be rearranged, shortened or extended using the **Page** and **Varnum** entries (see section 4.9). The **Duplicate** button will create a copy of the highlighted variable for which you might choose other plot parameters. The **Remove** button will remove the highlighted variable from the list. The Clear button will remove all entries from the list.

4.2 Plotting

The **Plot/Save** menu allows to create multiple plot windows on the screen containing plots for all variables in the **Plot Variables** list.

The **On Screen** menu item will produce a plot in multiple (up to 32) IDL graphics windows on your terminal. The windows are labeled QVB_plot[n].
The menu items **PostScript, PostColor and Flat File** create either a multiple page postscript file containing the displayed plots (in black and white or colour) or a flatfile of the variable values in the time interval in QSAS qft-format. The path for all these files is determined by the 'PS_dir' entry in your QVB.cfg file. The filename consists of the string qvb_ and the chosen time interval appended by .ps, .psc or .qft.
The menu item **Options...** will open the **QVB PlotOptions** GUI. See section 4.8 for its operation.

4.3 Dismissing plot

When you have finished plotting, you may use the **Plot/Save** | **Close** menu item or the QVB_pui window Quit menu item to close down QVB_pui. This closes also the QVB_plot windows. The

QVB main interface remains active awaiting your input.

4.4 Scaling

The **Scaling:** menu provides four general choices to scale all panels contained in the plot:

- **predefined Scale** uses the SCALEMIN and SCALEMAX attributes which should be present for each variable contained in a CSDS CDF file.
- **predefined Valid** uses the VALIDMIN and VALIDMAX attributes which should be present for each variable contained in a CSDS CDF file.
- **QVB set** determines the scaling from the minimum and maximum values defined for the variables by your selection in the **Selected Variable** window.
- **Autoscale** determines the scaling from the minimum and maximum values reached by the variable in the current time interval for the data read from the first file which contains that variable.

For byte variables (Status) the default scaling is from -1 to the maximum reached by the data in each panel. The fill value 255 is converted to -1.

4.5 Annotate

This button will launch a plot Annotation interface allowing to write comments on a plot. The interface contains a **Help** button describing the basic functionality. You should start by writing one line of text into the **Annotation Texts** window, then push the respective **Positions** button. Now you have to click with the left mouse button into the plot window. You can drag along the text using the left mouse button, resize and rotate it using the middle mouse button. The right mouse button will close the operation. Notice, that the execution of all other QVB operations will be delayed until the position operation is closed. Now you can repeat the procedure with the next line of text. If you reposition a textline the old text remains visible until you replot the whole page. The postscript file will always contain the annotation text at their latest position.

4.5.1 Hidden Annotation

All main titles and axis titles annotation contained in QVB plots are accessible via the annotation tool. They get invisible if you press the **Hide** button, visible if you press the **Show** button. You can change or resize or reposition them. These operations will automatically convert the entry into a 'shown' entry. As long as there are any 'shown' entries in the annotation, QVB will NOT automatically update the annotation. If you select new variables or change the order of plot variables you should clear the Annotation Tool window. If you would like to keep some entries, save them to a file and reload that file after clearing and replotting.

All annotation can be saved to a flat file. The default name of that file is annotation.dat in your QVB_local directory. It can be changed using the Annotation_File entry in your QVB.cfg file

or by editing the respective textstring in the annotation interface. For further information on the Annotation Tool see the chapter 'The Annotation Tool QPAN' in *QSAS and QVB IDL Routines. Reference Guide.* (CU-QMW-TN-0009)'.

4.5.2 Fonts

The default fonts used by QVB are Simplex Roman for screen plots and Helvetica for postscript plots. This setting can be changed using the **PlotOptions** GUI. You may change the font in additional annotation using '!n' in front of your texts where n is an IDL font index (table 9.1 of IDL4.0 User's Guide). But make sure to change back to the default font by closing your texts with '!3' in this case. Notice also that the font index will be interpreted differently for postscript output (table 3.15 of IDL4.0 Reference Guide). You can also use the **Type** entry in the annotation tool to select a font number.

4.6 Setting the time interval

The **TimeInterval** button pops up the **Time Editor**. The **Time Editor** lets you change the year, month, day, hour, minute, second and millisecond fields for the start and finish times either by unit increments/decrements with the buttons, or by clicking-and-typing a value into one of the fields, *followed by a* Return. The fields are coupled, not independent, so if you increment the hour past 23 it will restart from 0, after incrementing the day (which may also increment the month, which might also increment the year). The +- Day, Hour, Minute, Second and Millisecond buttons are timer buttons, this means clicking once on the button will continuously increment or decrement the respective value. A second click stops the process.
The center column of the editor contains buttons to **COPY** the start time to the finish, or *vice versa*, which can be useful when setting up small intervals, and to **RESET** either time to its limits given in the CDF file. The **STEP+** button copies the finish time to the start time and increments the finish time by the previous interval between start and finish time. The **STEP-** button works in negative direction, respectively. The default setup times accessible via the **RESET** buttons are always defined by the highest and lowest time value in the current CDF file. On opening a new file the selected times are NOT automatically updated. This allows the quick comparison of the same time intervals for different files. Use the **RESET** buttons to set the interval to the full timerange of the current CDF file.
The **PICKUP** button allows the interactive pickup of a time interval from the plot window with your mouse. It will only be active after creation of a plot window and using any other of the time editor buttons.

4.7 Help

This button explains the basic functionality of all entries in the QVB Plot user interface.

4.8 Plot Options

The **Options...** menu entry on the QVB Plot User Interface launches the **PlotOptions** interface defining general parameters mainly for hardcopy output. When launched for the first time QVB uses following default parameters for plotting:

- PostScript- or flatfile name: '<default>.ps' results in a file name consisting of qvb_ and the chosen timeinterval with the extension .ps, .psc or .qft for PostScript/Color PostScript or flatfile. If you choose a different name it will stay valid for all files produced later on. Type default into the text field to reset to automatic setting.
- PostScript Colour default: black/white.
- Page Orientation: Portrait.
- Colour Depths: 4 bit per pixel. For spectrograms you may set this value to 8BpP but this results in bigger files.
- Encapsulation: No.
- Sizes and Offsets: Xsize [18.0 cm] defines for any orientation the shorter extension of the paper, Ysize [26.0] the longer. The offsets [1.0 cm]/[1.0 cm] are determined from the lower left corner, but you may choose other values depending on your printer.
- Scale: [1.0] can be used to scale down or up PostScript output.
- LogoFile: If you would like to add a logo to the plot type in the respective GIF file location and name. The logo will then appear in the upper right corner of the plot scaled to LogoSize.
- LogoSize: [1.0 cm]
- Annotation Settings
  Individual entries can be changed using the **Annotation Tool**.
  - CSize: The overall character size. A value of [1.0] corresponds approximately to 14pt.
  - CThick: The character thickness [1.0] valid only for Vector Drawn Fonts.
  - Font: [3] results in Simplex Roman fonts for screen plots and Helvetica for PostScript plots. Other preset PostScript fonts are: HelveticaBold[4], HelveticaNarrow[5], HelveticaNarrowBold[6], Times[8], Courier[11], CourierOblique[12], PalatinoItalic[13], PalatinoBold[15], PalatinoBoldItalic[16], AvantGard[17], Schoolbook[18].

    If you choose a negative font number (< -2) IDL Vector Drawn Fonts will be used also for PostScript output (see the chapter in IDL User's Guide).
- Colours
  - BackGround: [255] results in a white background.

  - Pen : [0] black default pen colour.
  - Table : [39] IDL colour table number used by QVB. This is a rainbow colour table ranging from black [0] to red [254] with white [255] added.

- Miscellaneous
  - LineThick: [1.0] Overall thickness of axis lines and ticks.
  - Date: [Y] Include small print date string in upper left corner.
  - Ticks: [Out] Draw tickmarks inward or outward.
  - Iconify: [No] Iconify plot windows after plotting.

The **Accept** button will dismiss the Options GUI with it's current state accepted as valid, the **Cancel** button will reset the options to their last valid value and dismiss the GUI, the **Reset** button will reset to the default option values. When saving your plot parameters the PlotOptions settings will also be saved. If you would like to use the same options with a different plot layout, reload the plot parameters before selecting the data.

4.9 Page layout

The plot pages are subdivided into main panels for each scalar variable and each vector component. Byte variables (e.g. Status) are displayed byte by byte in subpanels. Each plot window contains by default up to five main panels. The plot contains the CDF file creation date contained in the Global Attribute 'Creation_date' in the upper right corner, the plot creation date is contained in the upper left corner. The Y axis labels are taken from the Variable Attributes 'UNITS' and 'LABLAXIS' accompanying CSDC variables. Change these entries using **Annotation**. The Y axis type will use the 'SCALETYPE' attribute to create logarithmic axes.

4.9.1 Fill values and missing data

Fillvalues appear as data gaps in the plot (except for byte data). If the chosen time interval contains no data at all (not even fill values) the panel will contain the message 'No data in this interval', if it contains fill values only the message will be 'Fill values only'.

4.9.2 Time axis

The time axis will usually extend over the chosen time interval to allow major time tickmarks at both ends of the axis. The axis title is always 'Universal Time', but the time represented is always the Epoch time contained in the CDF file and converted by the CDF library routines to time strings. Change the title using **Annotation**.

4.10 Saving Plot Parameters

The **File** menu allows to save the complete Plot Setup to an IDL saveset file. This includes the Plot Options and the customized settings and may include the Plot Variables list. The default

name of the save file is qvbplot.sav. The default location is the QVB_local directory. This default setting is overwritten by the Plot_Save_File entry in your QVB.cfg file. . You can also edit the textstring to the right of the **File** button to change name and location of that file.

The **File** menu items have following meaning:

- **Save Data and Plot to**
  Not only the plot parameters for the Plot Variables list are saved, but also the physical location of the underlying CDF data files. When you use that operation you must not go through the data selection process when restoring your QVB session. But plotting of the data will not work if the physical location of the data has changed between QVB session or if you wish to plot a time interval not covered by the current data selection.

- **Save Plot to**
  Only the Plot Variables list and the plot parameter setting is saved, but NOT the physical location of the underlying data. When restoring a save set produced that way you must have gone through the data selection process first. This option is useful when you want to use the same parameter setting on different data base locations or on different time intervals, but it will not work if the variables selected under **Plot Variables** are not contained in the **Selected Variables** list on variable selection.

- **Restore Data and Plot from**
  This restores a save set produced by the **Save Data and Plot to** operation.

- **Restore Plot from**
  This restores a save set produced by the **Save Plot to** operation.

4.11 Customized Plot Design

SingleClicking on a variable in the **Plot Variables** list will make that variable active for the control buttons underneath the **Plot Variables** window:

The **Duplicate** button puts a copy of the variable entry on the **Plot Variables** list. You can use that option to plot the same variable with different plot options on the same plot.

The **Remove** button removes the current variable from the list. It will not be plotted then.

The **Clear** button clears the **Plot Variables** list. This should be used before selecting new variables with the **Map Variables** selector, if you want to choose a completely new set of variables.

DoubleClicking on a variable in the **Plot Variables** list will fill the **Selected Variable** window with the plot parameters for that variable. The list determines the order in which variables are plotted. It can be rearranged, shortened or extended using the **Page, Varnum** entries in the **Selected Variable** window together with the **Duplicate, Remove** buttons. The plot layout is determined variable by variable.

Note that all parameter settings get active only after pressing the Accept button in the Selected Variable window.

The entry **Name** shows the CDF name of the selected variable.

The **Component** slider shows the current component of a non-scalar variable. All parameters shown are only valid for the respective component. The '0' component defines the magnitude of a non-scalar variable. You can change the current component using the **Component Slider**. If you change the current component the selection for the current component will automatically be stored but all settings will be valid only after pressing the **Accept** button.

4.11.1 Plot Panels

The panel position of a Selected Variable is determined by setting the Page and variable number position. Each component of a non-scalar variable will be plotted in a separate panel. Byte variables are distributed over subpanels of a main panel. Color spectrograms and component overplots need only one main panel per variable. The vertical height of each panel is determined by the number of variables plotted on a page and can not be changed.

The **Page** entry determines the output page number on which the variable will appear. Page numbers count from 1 to 32.

The **Varnum** entry determines the position of the selected variable on the page. Variable numbers count from 0 on top of each page. The Page and Varnum entries and the Plot Variables list are automatically resorted after pressing the **Accept** button. QVB4.5 accepts up to 128 components for each variable and up to 32 panels per page. Multidimensional array variables are always split into scalar timeseries ('components') by QVB. Variables with more than 8 components are plotted as spectrograms by default.

4.11.2 Plot Types

The **Type** menu determines the plot type for the current variable or its component. Following types are implemented:

- **NoPlot** Suppress plotting of respective component or Variable. Suppressing a variable has the same effect as removing it from the plot Variables list except that you can reactivate it more easily when needed.
- **Lin** Linear y-axis.
- **Log** Logarithmic y-axis.

- **Spe** Linear color spectrogram for non-scalar floating point variables. Setting this plot type affects all components of a variable. 'Linear' means that the colour coding of the spectrogram is linear. The Y-axis counts always the components only.
- **LSp** Logarithmic color spectrogram for non-scalar floating point variables. Setting this plot type affects all components of a variable.
- **bit** Bit pattern plot for byte and integer variables only. Setting this plot type means that the respective component is regarded as a byte value consisting of 8 bits. Each bit is plotted as a horizontal line with the value Black for 1 and White for 0 and Red for fill value (byte value 255).
- **Com** Linear vector component overplot. Setting this plot type affects all components of a variable. The selected components of a non-scalar variable are overplotted in different colours in one plot panel.

4.11.3 Plot Parameters

The **Ymin** and **Ymax** values determine the Y-scaling when the **Scaling | QVB set** menu item is selected. For non-scalar variables the setting affects only the current component except for spectrogram and overplot plot types.

The **Avsec** entry defines the length of a box window in seconds for averaging the data before plotting. If a timeseries for a given time interval has more than 10000 points QVB will automatically apply a box average such that the resulting series has less than 10000 points. The chosen average time will appear in the bf Info window.

The **Symb** menu selects a specific plot symbol for each component of a variable, corresponding to the IDL !p.sym value. Note that one of the symbols defines a histogram plot.

The **Color** entry selects a specific color type for each component of a non-scalar variable. The range of colours is from 0(black) to 255(white) in rainbow color table.

The **Thick** entry selects a specific line thickness for each component of a non-scalar variable.
The **Symb** menu selects a specific plot symbol for each component of a non-scalar variable.

The **Line** menu selects a specific line type for each component of a non-scalar variable.

4.11.4 Variable Control Buttons

The **Accept** button makes the actual parameter selection of the Selected Variable valid. **Note that if you switch to another variable without accepting the changes all changes will be lost.**

The **Cancel** button cancels all changes made to the Selected Variable but keeps the variable on the Plot variables list.

The **Reset** button resets all parameters of the Selected Variable to the settings valid when the variable was selected.

The **Help** menu gives a short description of all entries in the Selected Variable window.

5 Restrictions and Troubleshooting

This section lists some problems you might encounter when installing or using the QVB application, and suggests how to diagnose and cure them. See also the actual bug list on the QVB homepage.

5.1 Requirements and Restrictions

5.1.1 QVB 4.5 Restrictions

- CDF requirements
  QVB uses the CDF_* data retrieval routines supplied with IDL. These are derived from the CDF library version valid for your local IDL version (e.g. CDF2.6 for IDL5.x). Files produced by newer versions of the CDF library may not be browsable by QVB. QVB has been optimized to run with CSDS standard CDF files which contain zVariables only. It has been tested on non CSDS files and rVariable CDF files also. It is expected to work on those files, but the plot output will be of inferior quality because the labeling attributes defined for CSDS will be missing. For plotting only time dependent variables are accepted .i.e. those for which the DEPEND_0 attribute points to a valid epoch variable or if there is at least a valid epoch variable of a matching dimension contained in the file. Time dependent array data will be split up into scalar timeseries. Image data can not be plotted by QVB.

- Software Requirements
  QVB is expected to run on Unix and OpenVMS systems with IDL version 5.3 or higher installed. It has also been shown to operate in an MSWindows environment. In this environment the Variable Selector does not allow to browse through a subdirectory structure. If you would like to run QVB4.5 with an older version of IDL you have to recompile the source code which is available for IDL4.0 and IDL5.0. QVB4.5 supports the /COMPRESS feature for saving code and Data Maps. Thus savesets are not backwards compatible and should be reproduced when upgrading IDL. QVB has not been tested in environments other than those specified in the file README_QVB distributed with the release. Since QVB does not make use of OS commands (except for the optional feature of launching a texteditor), it can easily be transported to other systems running IDL. If this is desired please contact the address at the end of this document.

- Hardware Requirements
  The QVB widgets are developed for screen sizes of at least 1024×768 pixels. If the screen size is smaller the QVB main widget and the File Selection widget will come up with fewer lines of text. The plot windows will scale with the screen size. These and the QVB Data widget are resizable by the user. On smaller screens not all parts of the validation widget will be visible at the same time. On terminals which do not provide a 'screen-bold' font the QVB FONT keyword should be set on installation to a suitable font.

The maximal number CDF data files per variable which can be handled by QVB 4.5 is 10000. Since QVB does NOT store variable values memory space is only needed for metadata structures, .i.e. scanning several thousand CDF files will usually need less than 10MB of memory.

- Plot Restrictions
  The maximal number of components of non-scalar variables is 128. The maximal number of output pages is 32. It is also not recommended to use more than 32 panels per page since this confuses the interactive value pickup.

5.1.2 QVB Maintenance

The source code for QVB 4.5 is released together with the distribution of the executable. This should allow the adaption of QVB to other platforms and customizing of QVB features. The source code distribution is documented in the *QSAS and QVB IDL Routines. Reference Guide.* (CU-QMW-TN-0009). Support will be restricted to failures in software code unchanged by users.

5.1.3 Related Material

CU-QMW-TN-0006	Introduction to QSAS.
CU-QMW-TN-0009	QSAS and QVB IDL Routines. Reference Guide.
DS-QMW-TN-0003	CSDS Standard CDF File Format.
DS-QMW-TN-0006	Possible CSDS-UI Validation Operations.
DS-RAL-SM-0001	CSDS UI Validation User Manual.
DS-ESR-SM-0003	CSDS-UI Principal Investigator User Manual.

5.2 Troubleshooting

This section lists some problems you might encounter when installing or using the QVB application, and suggests how to diagnose and cure them. See also the actual bug list on the QVB homepage.

5.2.1 Problems installing QVB

On systems running IDL5.3 on Solaris 2.7 the procedure described in the Installation section should work without problems. In other environments you may try the installation described in section 2.2 'Obtaining QVB for Other Environments'. When installing QVB on a Unix system using a saveset produced on OpenVMS you may encounter messages like:
% Not a legal system variable: !COM_ROOT.
% Not a legal system variable: !THIS_OS.
% Temporary variables are still checked out - cleaning up..

These are produced differences in the IDL implementation and can usually be ignored.
If there are further problems installing QVB please contact the authors.

5.2.2 QVB main panel not displayed

Look at the messages in the window from which you loaded and ran QVB in IDL (or from which you ran the shell script that did this for you). If QVB cannot load the QVB.cfg file, it writes an error message here and exits without displaying the main panel. Check that the required file is in the right place as specified in the 'Installation' section, and that the paths in them are set correctly for your system. Check whether the environment variable DISPLAY is correctly set in your qvb shell script, or as given in the command line argument.

5.2.3 QVB main panel dies or widgets don't respond

This has been observed in an inhomogeneous environment, when QVB was being run on a remote Solaris machine with the display directed to a workstation running SunOS. This is not a supported configuration. Such problems have not been observed in a homogeneous Solaris environment, with QVB running locally or remotely, and this is the recommended configuration.

5.2.4 Attempted cut-and-paste crashes IDL

In some installations, attempting to cut-and-paste text between the QVB display (or any other IDL widget) and other windows has been found to crash the IDL application. This is believed due to a 'missing' library /usr/lib/X11/nls in some releases of OpenWindows. If you encounter this problem, consult your System Administrator. A similar problem has been observed when double clicking on list widgets.

5.2.5 Cannot open CDF file

The default path to a CDF file specified in QVB.cfg will almost certainly be wrong for your system. You should edit it at installation. If you use the **File Selector** dialogue on the QVB main panel you should be able to open a CDF file by entering the full path, starting from the root directory /.

5.2.6 Problems with Data Mapping

Whenever you add CDF files to a directory you should run the QVB Mapping Tool (see section 3.4) again. Note that you need write access to the directory to do so. From IDL version 5.3 the mapping tool will use the /COMPRESS feature when saving QVBdatamap.sav. If you get the message
% RESTORE: Obsolete SAVE/RESTORE file:
you are running an IDL version different from the one which produced the QVBdatamap.sav. You should remap the directory in this case. Note that you may switch of the mapping feature in your QVB.cfg file.

5.2.7 Typed text apparently ignored

Text typed from the keyboard into a recessed text field (e.g. the year, hour, min,... fields in the Time Editor) is ignored until a carriage return is entered while that field is active.

5.2.8 IDL screen colour problems

Parallel colour applications In principle it is possible to run multiple sessions of QVB in parallel. But since the first IDL session will grab all colours available in the current X-window session it will not be possible to view two plot windows at the same time. This conflict will also arise with other graphics application running parallel on the same host.
If there are multiple colour applications running on your terminal you may have following effects:

1. IDL widgets do not appear in full colour. This can only be solved by closing down other colour grabbing applications. Some applications (like NetScape) you can start in a reduced colour mode.

2. The IDL Plot window has reduced colors (especially a red background). QVB needs at least 129 colours to run without colour problems. QVB4.5 sets this minimum automatically. If the automatic setting does not work you may try to add the entry
   idl.colors: 129
   to your .Xdefaults(Unix) or DEC$SM_GENERAL.DAT (VMS) file in your home directory.

True Colour Devices On X-terminals with more than 8 bits of colour depths it is necessary to reduce the number of colours available to IDL by adding following entries to your .Xdefaults(Unix) or DEC$SM_GENERAL.DAT (VMS) file:

idl.gr_visual: PseudoColor
idl.gr_depth: 8

Otherwise the representation of colours in the IDL plot window will not be well defined.

5.2.9 IDL keysym translation errors

Depending on your IDL/Solaris version there may be startup warnings produced by every IDL-widget application you are running of the type:
Warning: ... found while parsing '<Key>osfSelect:KeySelect()'
Warning: translation table syntax error: Unknown keysym name: o
These errors mean that the IDL widget application can not identify all your keyboard buttons. Generally this will have no effect on the QVB operation except for using the Del button for text input.

5.2.10 Problems editing QVB.cfg

If the **Edit** button in the **Settings** menu refuses to launch an editor, check whether you have specified a TEXTEDITOR entry in your QVB.cfg file. If so check whether the respective command will actually launch an Xwindows texteditor on the machine where QVB is running. If this is the case please send a bug report to the address below. Notice that only Xwindows editors but not editors running in a Unix Command window are allowed. Meanwhile you can always edit QVB.cfg in a process independent of QVB. When closing QVB the edit process will be automatically killed. This will probably not work in all unix shell environments. In this case please quit the edit process yourself. Also the killing of the process will result in OS warnings like:
XView warning: invalid object (not a pointer), xv_get
or
/bin/sh: kill: no such process.
Please ignore these messages.

5.2.11 Problems using the Annotation Tool

Please read the operation description for the tool in section '4.5 Annotate' carefully. Notice that not all characters on your keyboard may be available in the plotting font.
For further information on the Annotation Tool see the chapter 'The Annotation Tool QPAN' in *QSAS and QVB IDL Routines. Reference Guide.* (CU-QMW-TN-0009).
If you encounter further problems send a bug report to the address below.

5.2.12 Problems using the File Selection

The file selection tool uses the IDL 'CD' and 'FILEFIND' commands to obtain the contents of the directory structure on your system. Generally only directories open for reading should show up in the **Subdirectories** list. Some problems have been observed when the structure contains automounted directories: The **Subdirectories** list may not always contain all directories available or the directories listed are not all open for reading. Also the system might hang as long as the automounting process is active. Changing the **Path** entry is in this case more reliable since the system is not urged to automount all directories on the path.

5.2.13 Problems quitting QVB

This will be the case when there are still active QVB processes running. If you used the Annotation Tool within your QVB session the quitting of that tool will take some seconds. Be patient in that case. If you launched an editor to edit QVB.cfg QVB will try to kill this process before exiting. This may not work for all system configurations. In this case quitting the editing session will solve the problem. Otherwise try to quit QVB using the menu provided by Xwindows in the upper lefthand corner of the QVB main window. If this does not work try 'Control C' or 'Control Z' and kill the QVB process. In this case the following note applies.

5.2.14 QVB session dies

Notice that in this case an existing Temporary Validation File will not be validated. The file will still exist in your QVB_local directory. You will be prompted to reload or overwrite the file when you start a new validation process.

5.2.15 QVB Internal Errors

In rare cases there will be an error message appearing on your screen stating a QVB Internal Error. These errors are not caused by an erroneous installation or use of QVB but by errors in the QVB program code missed by the authors of QVB. When the error occurs in a QVB subroutine you cannot proceed with the specific task requested but may still continue using QVB. When the error occurs on the main level QVB will close down after you have acknowledged the message. In any of these cases a message to the address given below will be appreciated.

5.2.16 Bug Reports and Suggestions

Bug Reports containing a screencopy of the error message, a specification of the selected CDF file and the selected time interval and all suggestions for improvements should be sent to:

csc_support@qmw.ac.uk

Index