Chapter 1

User's Guide of the XMM–Newton Science Analysis System

Ph. Gondoin (pgondoin@astro.estec.esa.nl)
XMM–Newton Science Operation CenterJuly 10, 2000

Contents

1.1 Status

version (from `VERSION`): 0.75.3

1.2 Change history

Acronym	Explanation
AO	Announcement of Opportunity
AOCS	Attitude and Orbit Control and Measurement Subsystem
AMS	Archive Management Subsystem
CAL	Calibration Access Layer
CCD	Charge Coupled Device
CCF	Current Calibration File
CTE	Charge Transfer Efficiency
CTI	Charge Transfer Inefficiency
CVZ	Continuous Viewing Zone
DAL	Data Access Layer
DEC	Declination, δ(J2000)
EPIC	European Photon Imaging Camera
ERMS	EPIC Radiation Monitor Subsystem
ESOC	European Space Operations Centre
ESTEC	European Space Research and Technology Centre
FITS	Flexible Image Transport System
FOV	Field Of View
FWHM	Full Width at Half Maximum
GMT	Greenwich Meridian Time
GO	Guest Observer
GT(O)	Guaranteed Time (Observer)
GTI	Good Time Interval
GUI	Graphical User Interface
HEASARC	(NASA) High Energy Astrophysics Science Archive Research Center
HEW	Half Energy Width
HER	RGS high event rate (selectable mode)
HTR	High Time Resolution (mode of RGS)
LSF	Line-spread Function
MIME	Multipurpose Internet Mail Extensions
MOS	Metal Oxide Semi-conductor
OCB	On-Chip Binning
ODF	Observation Data File
OGIP	(NASA's) Office of the Guest Investigator Program
OM	Optical Monitor
OTAC	Observatory Time Allocation Committee
PHA	Pulse Height Analyser
PI	Principal Investigator
PSF	Point-Spread Function
QE	Quantum Efficiency
RA	Right Ascension, α(J2000)
RFC	RGS Focal Camera
RGA	Reflection Grating Assembly (of the RGS)
RGS	Reflection Grating Spectrometer
RPE	Relative Pointing Error
SAS	Science Analysis System
SOC	XMM Science Operations Centre
SSC	Survey Science Centre
ToO	Target of Opportunity
UHB	XMM Users' Handbook
URL	Unique Resource Location
UT	Universal Time
W90	90% energy width
XMM	X-ray Multi-Mirror Mission

Chapter 2

Introduction

2.1 Scope of this manual

After observation or upon request to the XMM Science Operation Center (SOC), observation data from the ESA XMM observatory are sent to guest investigators on CD-ROMs. The CD ROMs contain raw and calibrated event files collected by the European Photon Imaging Cameras (EPIC), the Reflection Grating Spectrometers (RGS) and the Optical Monitor (OM) on-board XMM. The purpose of this document is to guide XMM guest investigators in analysing these data.

2.2 Required software environment

The analysis of XMM data includes quick look analysis of raw data, calibration of raw event dataset, screening of the calibrated data, extraction of images, extraction of spectra, extraction of time series, source detection and scientific analysis of the calibrated products. A large fraction of these tasks has to be conducted with the XMM Science Analysis System (SAS) which has been developed by a team of scientists located at the ESA XMM SOC and at the XMM Survey Science Centre (SSC). Informations on how to download and install the SAS software package are available on the XMM web server at:

`http://xmm.vilspa.esa.es/user/sas_top.html`

2.3 More information on the web

Before analysing XMM data, XMM guest investigators should get familiar with the basic characteristics and operation modes of the scientific instrumentation on board XMM−Newton. This information is available in the XMM−Newton Users Handbook at:

`http://xmm.vilspa.esa.es/user/uhb_top.html`

An other important document is the XMM Data File Handbook which provides a detailed description of the data distributed to guest observers from the XMM−Newton archive. This handbook includes in particular information about the format, structure and content of individual XMM−Newton files. The document can be downloaded from:

`http://xmm.vilspa.esa.es/sas/doc/dfh.ps.gz`

The Calibration Access and Data Handbook [1] describes the calibration files and the calibration access routines which are used by the SAS to access these files. The document is accessible from the

XMM−Newton calibration web page at:

`http://xmm.vilspa.esa.es/calibration/docs/general/calhb.ps.gz`

The present guide complements detailed descriptions of individual SAS tasks which can be accessed through the SAS web page or by pushing the "Help Task" button of the SAS GUI.

2.4 Structure of the document

The structure of the present document is as follows:

- Chapter 3 introduces the XMM investigator to the analysis of XMM data. It provides a brief description of the XMM observation and calibration files. It outlines the analysis procedures which have to be conducted to produce calibrated event files and to extract scientific products.
- Chapter 4 describes the SAS graphical user interface (GUI). The SAS GUI is a user friendly tool which enables to run SAS interactive analysis tasks without using command lines.
- Chapter 5 describes the successive SAS analysis steps which, in general, have to be conducted prior to the scientific analysis of XMM EPIC final data products.
- Chapter 6 describes the initial SAS analysis steps to be conducted prior to the scientific analysis of XMM RGS final data products.
- Chapter 7 describes the SAS preparation of OM data before analysis using standard astronomical software packages.

2.5 Reporting problems

Problems in using the SAS shall be reported through an observation report at:
`http://xmm.vilspa.esa.es/cgi/or.cgi`

If the observation report points to a software problem, the XMM-Newton SOC will raise a Software Problem Report (SPR), which will be fixed in forthcoming software versions. Any inquiry on the status of the observation reports, as well as any general question about the XMM-Newton data analysis, must be issued only through the XMM-Newton HelpDesk:
`http://xmm.vilspa.esa.es/helpdesk`

Chapter 3

Analysis of XMM data: an overview

3.1 The XMM data files

After completion of an XMM observation or under request by means of the archive management subsystem, XMM data files are distributed to guest investigators on standard CD-ROMs. The CD-ROMs contain:

- top-level disk files which describe the contents of the disks,
- the observation datafiles (ODF) located in an ODF directory. These files includes the raw event science files from the EPIC, RGS and OM instruments, the instrument housekeeping files, the radiation monitor files and the spacecraft files.
- the Current Calibration Files (CCF) located in a CCF directory which includes the calibration data.
- data products generated by the Pipeline Processing Subsystem (PPS) which are grouped in the PPS directory.

The XMM observation data files and current calibration files conform to the Flexible Image Transport (FITS) standard. A detailed description of their format, structure, naming convention and contents is provided in the XMM Data Files Handbook and in the Calibration Access and Data Handbook [1].

3.2 Successive steps in analysing XMM data

In general, the analysis of XMM data proceeds along the following steps:

1. creation of a working directory with access to XMM data and calibration files, SAS analysis system and associated software packages,
2. inspection of the XMM observation datafiles, current calibration files and pipeline products,
3. definition and planning of the analysis activities,
4. creation of calibrated event lists using the pipeline processing batches. The user may skip this step and only use for analysis the calibrated event lists provided as pipeline products.
5. data screening, source detection and extraction of scientific products including images, spectra and time series,
6. generation of auxiliary response files and redistribution matrices,

7. scientific analysis of the final XMM products.

The completion of step 2, 4, 5 and 6 requires the use of tasks specific to the analysis of XMM data which constitute the XMM Science Analysis System (SAS).

3.3 Starting a SAS session

There are two independant ways of operating the SAS, either using the command line as illustrated below, or using the SAS Graphical User Interface, hereafter called the GUI. The SAS GUI is called by typing *sas* on the command line. Configuring the ODF and CCF directories using the SAS GUI is explained in Chapter 4.

When starting a SAS session in a working directory, the user should:

1. link the SAS to the Observation Data File of interest hereafter refered as the ODF
2. establish the list of calibration files (herafter refered as the calibration index file) suited to the analysis of this specific ODF at the current point in time,
3. link the SAS to this list of calibration files (hereafter refered as the current calibration files or CCF) which are relevant to the analysis of the selected ODF.
4. extends the summary file of the Observation Data File (ODF) with data extracted from the instrument housekeeping files and from the calibration database.

The specification of the ODF to be treated can be achieved using the setenv command line.

```
setenv SAS_ODF ‘path to the ODF directory‘
```

Once the Observation Data File (ODF) is being pointed to via the SAS_ODF environment variable, the user should run the task *cifbuild*. This task uses the date of observations, the date of analysis and a library of calibration files to build the list of calibration files suited to the foreseen analysis. This list is called the calibration index file (CIF). As an example, *cifbuild* can be run as follows.

```
cifbuild analysisdate='now' append=no calindexset='ccf.cif' category='XMMCCF'
ccfpath='path to directory of calibration files' fileglob='*.ccf|*.CCF'
fullpath=yes ignorecategory=no masterindex=no masterindexset='ccf.mif'
missingfile='cifmif.asc' observationdate='2000-06-10' recurse=no
withccfpath=yes withmasterindexset=no withmissingfile=no
withobservationdate=yes
```

The SAS command *cifbuild* reads the observation date from the observation start date as reported by the ODF Access Layer (OAL). Alternatively, the user can specify the observation date using the task parameters withobservationdate and observationdate. The analysis date can be specified with the task parameter analysisdate. Its defaults value is the current date. These dates are used to retrieve the CCF constituents from a repository of calibration files which are applicable to the analysis of the selected ODF at the specified point in time.

Once the calibration index set (ccf.cif in the above example) has been generated, the user shall point the SAS to this file via the SAS_CCF environment variable.

```
setenv SAS_CCF ccf.cif
```

Finally, the user shall extend the summary file of the Observation Data File with data extracted from the instrument housekeeping files and from the calibration database. This is performed using the task *odfingest*. *odfingest* examines all the files in the ODF directory and reconstructs the structure of the observation in term of exposures. It then creates a new ODF Summary File (SOSF) which is needed by the SAS to process the ODF. Assuming the presence of an ODF in a directory called /odf/0001/0100240301 (revolution 1, observation identifier 0100240301), *odfingest* can be run as follows from the command line.

```
odfingest -odfdir=/odf/0001/0100240301
```

The above coammand creates the file 0001_0100240301_SCX00000SUM.SAS in the current directory. Alternatively, the file can be created in a directory specified by -outdir.

```
odfingest -odfdir=/odf/0001/0100240301 -outdir=/odf/summary_store
```

3.4 Selecting a SAS task

A list of SAS tasks appears when typing the command *sas* which call the SAS Graphical User Interface (GUI). A SAS task can simply be called by double-clicking on its name. Use of the SAS GUI is explained in chapter 4. SAS tasks include:

- General utility tasks which use both raw events files and calibrated events files,
- Calibration tasks which e.g interact with the current calibration files,
- Pipeline processing tasks,
- EPIC specific tasks,
- RGS specific tasks,
- OM specific tasks,

A list of packages in alphabetic order or classified by groups is accessible at:

```
http://xmm.vilspa.esa.es/user/sas_top.html
```

For detailed descriptions of the tasks, the user is refered to the SAS package descriptions which can be accessed at the above address or via the SAS GUI help button. As a starting point, the user could read the package documentation of the list of frequently used tasks provided in table 3.1

Table 3.1: *Some frequently used SAS tasks*

Task name	Task description
arfgen	Generates an EPIC ancillary response file
calview	An interactive viewer of the XMM calibration database
cifbuild	Creates a Calibration Index File for a given observation date
edetectproc	Automated source detection metatask
emproc	Standard pipeline processing of EPIC MOS observation data files
epproc	Standard pipeline processing of EPIC PN observation data files
evselect	Filters event lists and extracts images, spectra, timeseries and histograms
rgsproc	Standard pipeline processing of the RGS part of an observation
rgsrmfgen	Constructs an RGS response matrix file
rmfgen	Generates an EPIC response file
sas	Launch the SAS graphical user interface
srcdisplay	Display image overlayed with positions of detected sources
xmmselect	Interactive filtering and extraction of images, spectra, timeseries and histograms

3.5 Running a SAS task from the command line

Before operating the SAS using command lines, the XMM Guest Investigator is advised to read the *taskmain* package description. Each sas task can be called by typing its name on the commandline, e.g:

```
evselect -h
```

Tasks may be configured by a set of parameters, which are entered on the UNIX command line. Every task has a Parameter Specification File which defines the name and type of each parameter and possibly a default value or allowable range. Parameters may also be entered using the Graphical User Interface (GUI). This reads the parameter specification file and provides a dialog window into which the user can enter parameters. The GUI then launches the task by generating a command line which can then be read in the sas_log file. Each SAS task supports the following command line options:

- -h provides information about the command line options.
- -p prints all parameters with their default and current value.
- -t turns on tracing information from the libraries.
- -m displays the HTML documentation of the task using Netscape.
- -c no clobber prevents files to be overwritten.
- -v prints the version number of the task.
- -V level set the verbosity level of the task.
- -w (code) suppresses warning messages. If no value is given all warnings are suppressed. Multiple options can be given, e.g.: -w 10 -w NoAttitudeData -w LowDisk will suppress all warnings with code NoAttitudeData and LowDisk and shows only the first 10 of all other warnings.
- -param=value set the value of parameter param to value

If the task uses the ODF Access Layer (OAL), the command line option -odf odf or -o odf is recognized as well. It specifies the Observation Data File (ODF) that is to be accessed by the OAL. This name odf can be the name of an ODF summary file as created by the task *odfingest* or merely the name of a directory containing one. In the latter case the summary file is assumed to have a name ending in either *SUM.ASC or *SUM.SAS. If both are present *SUM.ASC is ignored.

If the task uses the Calibration Access Layer (CAL), the following command line options are recognized:

- -ccf cifname or -i cifname. cifname is the name of a data set comprising a table with the name CALINDEX. This will normally be a Calibration Index File (CIF) created by the task *cifbuild* . cifname can also be the name of a directory. In this case the CALINDEX table is sought in a data set with the default name ccf.cif in the specified directory.
- -ccffiles f1 ... or -f f1 ... White-space, comma, or colon separated list of CCF constituents replace the corresponding ones in the CCF pointed at via the -ccf command line option or the environment variable SAS_CCF (see below). Each specified file must be a valid CCF constituent according to the CCF ICD.

- -ccfpath dir1[:dir2 ... or -a dir1[:dir2 ...] A colon separated list of directories that define a search path along which CCF constituents are to be sought. If a CCF replacement file is passed via -ccffiles with its full absolute path any specified search path is not considered in trying to access this constituent.

The behaviour of a task is modified by the following environment variables.

- SAS_PATH is a list of colon-separated directories that contain SAS packages. It should at minimum contain the toplevel directory where the SAS is installed.
- SAS_CCF points to the directory containing the Current Calibration File (CCF) or contains the name of the Calibration Index File. If a value is provided via the command line option -i (see above) this takes precedence over any environment specification. This variable is only relevant to tasks making use of the Calibration Access Layer.
- SAS_CCFFILES has the same meaning as the command line option -f. If a list of CCF constituents is given both via -f and SAS_CCFFILES the former takes precedence over the latter. This variable is only relevant to tasks making use of the Calibration Access Layer.
- SAS_ODF points to a directory containing an ODF. If a name is specified via -o and SAS_ODF the former takes precedence over the latter. This environment is only relevant to tasks making use of the ODF Access Layer.
- SAS_TRACER determines the debug level of the libraries.
- SAS_VERBOSITY determines the debug level of the task.

SAS tasks produce error messages at levels called messages, warning, error and fatal.

- At message level, a message is reported to the user and processing continues.
- At warning level, a warning message is reported to the user and processing continues.
- At error level, an error message is reported and current operation is aborted. Control may return to the calling program which can take an appropriate action.
- At fatal level, a fatal error is reported to the user and all processing is aborted. A fatal error is generated if the internal state of the program is disrupted, for example if an invalid value is found in a variable.

Error messages are characterised by a layer and a verbosity level. The layer indicates the layer in the system where the message is coming from (e.g. application, user interface, application library or system library). The verbosity level determines whether or not the message will be reported. The user can specify the verbosity level. If the verbosity of a message is large enough compared to the system verbosity level, the message will be reported. The verbosity level differs for the different layers.

3.6 SAS input and ouput files

The SAS commands process input and output data in the form of datasets. These are usually saved on disk as FITS files.

FITS files have a primary header, a primary image array and optional extensions, each with a header. In the case of the XMM FITS files, the primary header contain information about the mission, the instrument, the observation mode, the investigator, the target coordinates, the observation date and initial processing information. In all FITS headers, the information is in the form of keywords with assigned values.

For a detailled description of the XMM files (including format, content, keywords ...etc), the user is refered to the XMM data file handbook which can be accessed on the XMM web server at:

```
http://xmm.vilspa.esa.es/sas/doc/dfh.ps.gz
```

It is important to note that the naming convention of XMM-Newton files is slightly different from that historically used for FITS files. Table 3.2 provides a handy "conversion table" between the two nomenclatures.

FITS name	**XMM-Newton dataset name**
Header	Header
Primary Header Data Units/Array	First Data Block/Primary Array
Extension	Data block
Image	Array
Binary Table	Table
Keyword	Attribute
Column/Rows	Column/Rows

Table 3.2: "Conversion table" between FITS and XMM-Newton dataset constituents naming convention

3.7 Accessing calibration data

The XMM calibration database consists of calibration datasets forming the current calibration file (CCF) and of a number of calibration algorithms and access functions forming the calibration access layer (CAL). Many of the datasets in the CCF contain parameters that are meaningful only in conjunction with the calibration algorithms provided in the calibration access layer. The only manner to access the calibration data, either in raw form (the contents of the datasets themselves) or in the interpreted form (the output of the algorithms) is through the functional interface provided by the calibration access layer. The purpose of the SAS task *calview* allow the user to visualize calibration quantities. The task is simply called by typing:

```
calview
```

calview interacts with the cal through the interfaces provided by the calibration state server. The calibration state server consists of a calibration state editor and a calibration viewer state editor (see Fig. 3.1). The calibration state server allows one to set any of the calibration state parameters. These are: instrument, ccd, node, filter and mode identifiers, ccd and camera temperatures, on-chip binning factor, date, accuracy level, and randomization. The top part of the calview widget pane can be used to edit the calibration state. Through the pull down menu labelled "CCF" of calview, one can direct the cal to use a particular set of calibration datasets. It is possible to point the cal to a CCF directory, to a CCF index file (see task *cifbuild* for information on how to generate one), or add individual CCF components to an existing list.

The calibration view server allows the user to edit a number of state variables that affect the display of the calibration output. The following variables can be set: energy, position in the field of view, spectral order.

These variables function as input data to the calibration algorithms. For instance, for a given instrument and CCD the user can select the energy for which the CCD redistribution should be calculated. Through the calibration view server, *calview* inquires which "viewables" are available given the current calibration state. The list of available viewables is available in the pull down menu labelled "View". calview has no a priori knowledge of what calibration data can be viewed. Viewables can have sub-viewables. These are specialized views of a given calibration quantity. The CCF calibration files and the calibration algorithms are described in the XMM Calibration Handbook [1].

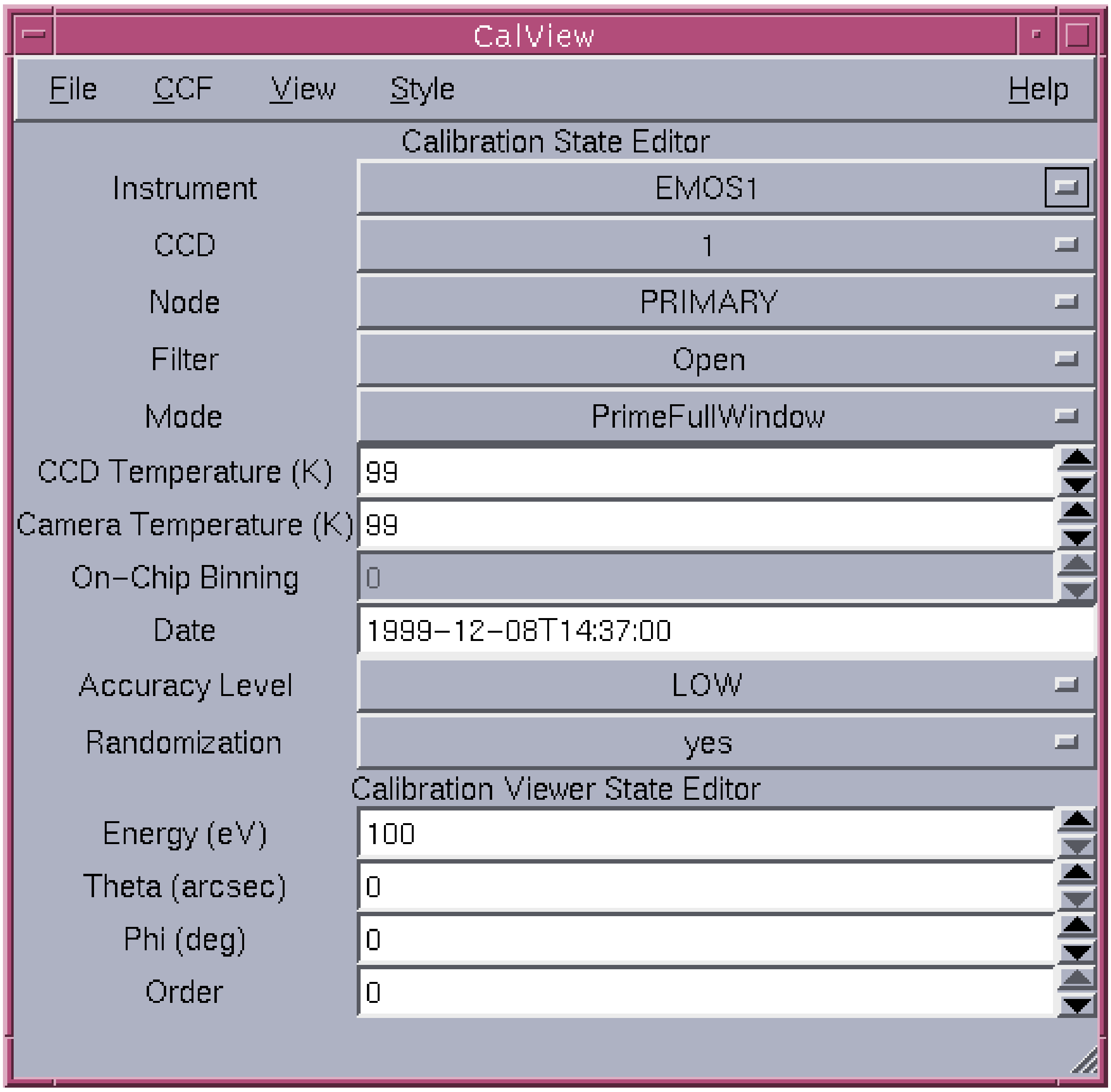


Figure 3.1: Calibration state server (calview window)

The XMM calibration access layer (CAL) requires a calibration index file (CIF), mapping calibration types to current calibration file (CCF) constituents. The SAS task *cifbuild* builds a CIF by scanning a list of directories for CCF components. The list of directories can be specified through the environment variable SAS_CCFPATH, or on the command line with the parameters withccfpath and ccfpath. The list of calibration datasets comprising a CCF is ruled by two dates, i.e (i) when the observation was performed, and (ii) when the analysis is performed. The first date is specified with observationdate, the second with analysisdate. Note that the latter can be any date (in the past or in the future) which is used to retrieve the CCF constituents applicable at the specified point in time. For instance, to generate the CIF applicable on 2010-01-01, say analysisdate=2010-01-01.

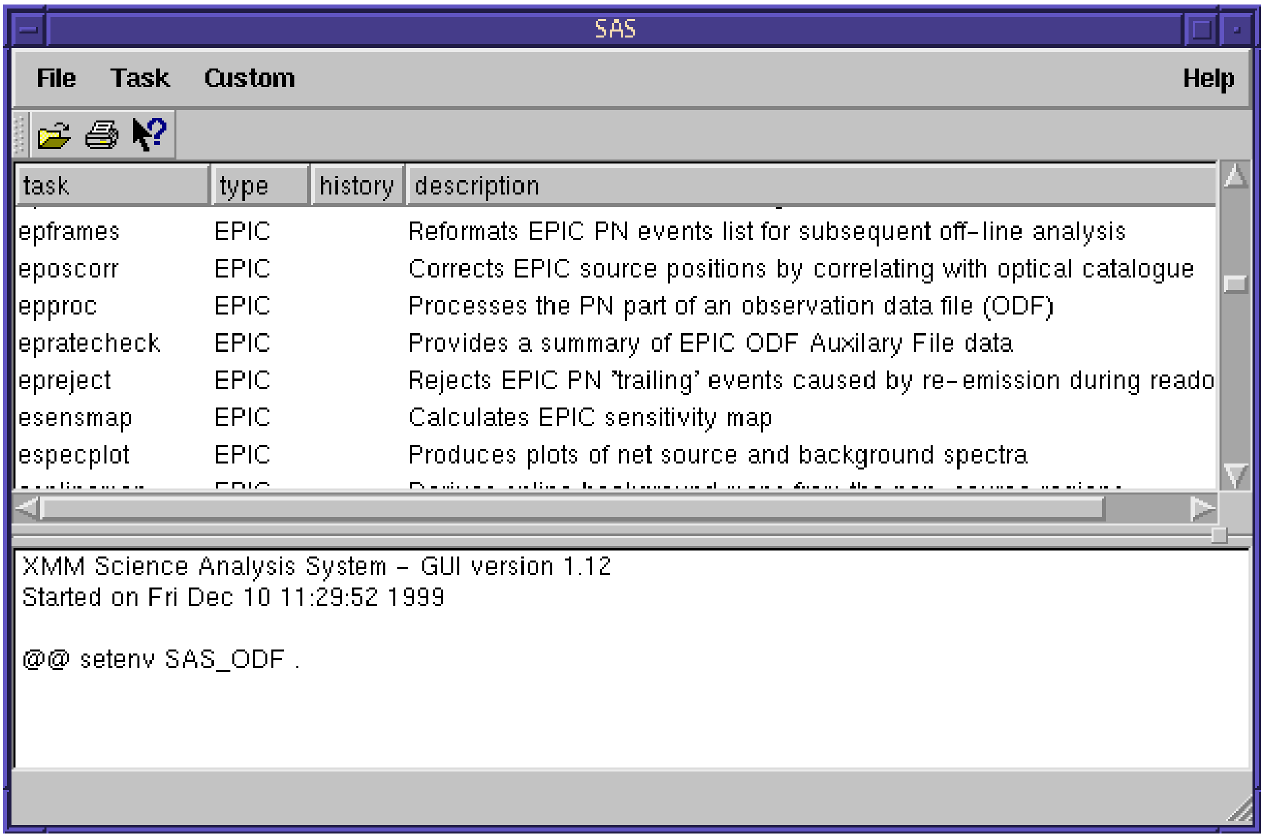
Normally *cifbuild* would be used to construct a CIF based on the CCF constituents available to the user, and stored in the directories indicated either in SAS_CCFPATH or in ccfpath. If the CCF constituents are stored in more than one directory, the CIF must be generated so that it contains the full file path. This is specified with the parameter fullpath. *cifbuild* can also be used to generate a CIF based on the content of the master index file (MIF). The MIF contains the list of all CCF constituents ever released. The MIF is part of the CCF, but its latest version is available from the SOC. The CIF created in this manner from the latest version of the MIF can be used to decide whether any updated CCF constituents are available from the SOC.

Chapter 4

SAS graphical user interface

4.1 Getting started

The SAS can be operated using the UNIX command lines. Alternatively, the SAS Graphical User Interface (GUI) provides an easy mean of running tasks without memorizing commands. The SAS GUI is specially intended for interactive analysis of XMM data. The SAS GUI is started by typing 'sas' on the command line. This should display a main window similar to the following:



The upper half of the window is a task browser, which allows to select and run tasks. The lower half of the window is a log pane, which shows messages from tasks, including warning and error indications. At the very bottom of the window is a *message pane* which shows transient messages.

4.2 A quick tour

1. **Start the SAS**

   It is convenient to create a directory which will hold the processing results and to run the SAS within this current directory:

   ```
   mkdir mydata
   cd mydata
   sas
   ```

2. **Configure CCF and ODF directories**

   Select the Observation Data Files (ODF), which you wish to process, by using the ODF button in the 'File' menu. The user should also set the directory where the SAS can find the Calibration Index Files (CIF) which points to the appropriate set of Current Calibration Files (CCF). This is done using the the preferences dialog in the file Menu. If the same directory is regularly used, a default can be set using the environment variable `SAS_CCF`. As explained in section 3.3, the Calibration Index Files (CIF) can be generated by the task *cifbuild*

3. **Select a task**

   The upper half of the main window is the task browser. You can click on the column headings to sort the tasks in various ways. Double-click on a task to enter the task parameters.

4. **Configure parameters**

   The parameter dialog (see section 4.5) allows you to configure the parameters of a task before you run it.

   You can select the input data by popping up a dataset browser (see section 4.7). This allows you to browse through the file hierarchy and select SAS datasets or individual components, such as tables, arrays and columns.

5. **Press the Run button**

   When the parameters have been configured, press the 'Run' button to execute the task. It is possible to kill a task (see section 4.6) from the Task menu.

6. **Examine the log**

   The lower half of the main GUI window is a log pane (see section 4.8), which displays messages from the task. Check that the task ran successfully. If the task produced any warning or error messages (see section 4.9), those will be displayed in a dialog box.

7. **Run another task**

   The results from a task will be written into the current directory, unless you specified a path for the output files. These files typically form the input to the next task, when you are running a sequence of processing steps.

   You can enter parameters in preparation for the next task, before the current task has finished. It is possible to queue the task for execution when the current task finsishes (see task control).

8. **Save or print the log**

   When you have finished your processing session, the log (see section 4.8) provides a record of the steps you carried out, with all parameters and any messages received from the tasks. You may wish to save this for future reference.

   The log is automatically saved in the file `sas_log` in the current directory. You can also print it from the 'File' menu or *tool bar*.

   All warnings can be suppressed and replaced by a summary by setting the environment variable SAS_SUPPRESS_WARNING

9. **Obtaining help**

   The SAS GUI provides yellow *tool tips* which pop up when the cursor is placed over an item. The main window has a *help tool*, in the tool bar; select the help tool and click it on an item to obtain more detailed help. Finally, the Help menu provides access to the on-line documentation. See section 4.11 for further details.

4.3 Selecting an ODF

At the start of an SAS session, you will normally select the ODF to be processed. Choosing 'Preferences' from the 'File' menu pops up a dialog into which you can enter the path of the ODF directory.

The ODF directory can be specified either by an absolute path (starting with '/') or a path relative to the directory in which you started the SAS GUI. The '...' button pops up a dataset browser (see section 4.7) which allows you search for the directory and select it.

It is possible to select a different ODF directory at any time during the session. Any tasks which you run subsequently will use the new ODF.

If you regularly use the same ODF directory, you can set the default using the environment variable SAS_ODF.

4.4 Task browser

The upper half of the main window is a *Task Browser* which is used to select and run SAS tasks.

Double clicking on a task causes the corresponding parameter dialog (see section 4.5) to pop-up, so that you enter parameters and start the task.

Clicking on one of the column headings sorts the list alphabetically on that field. Clicking again reverses the order. For example, clicking on the 'type' heading groups all tasks by instrument.

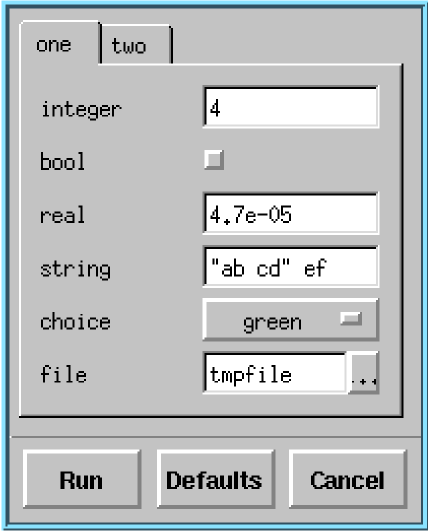
When a task is selected by double clicking, a number appears in the 'history' column. This number is incremented as successive tasks are run. Clicking on the 'history' column heading sorts the list in the order in which the tasks were run; this makes it easier to locate recently run tasks.

It is possible to locate a task quickly by typing the first few characters of its name, after first clicking in the task browser (or using the 'Tab' key) to select the browser. To locate another task, first click on another task to reset the search.

4.5 Parameter dialogs

Each task has an associated *parameter dialog*, which can be used to enter the values of parameters and run the task. The parameter dialog is opened by selecting a task from the 'Task' menu.

The following parameter dialog illustrates some of the basic parameter types. Each parameter type has a corresponding widget type. For example, a boolean parameter is entered using a check-box; a choice parameter is entered using a pop-up menu to select from a set of options; a filename parameter is entered as a string, with the option of popping up a file dialog by pressing the '...' button).



If the task has a large number of parameters, the dialog may have scroll-bars. The scroll bars will disappear if the size of the dialog is increased sufficiently.

Further information on a parameter can be obtained by placing the cursor over the parameter widget. This causes a yellow *tool-tip* to pop-up if the parameter file defines a *prompt* field for the parameter.

The parameter dialog has the following buttons:

Run	Run the task with the selected parameters
Defaults	Reset parameters to their default values
Cancel	Close dialog without running task or changing parameters

When a task has been run, the parameter values are retained until the next time that the task is run (within the same session). The *Defaults* button may be used to reset the parameters of a task to their default values. The 'Task' menu provides a means to reset the parameters of all tasks to their defaults.

4.6 Task control

It is possible to kill a running task from the 'Task' menu. This is useful if a long task is accidentally started with the wrong parameters or if a task appears to 'hang' for some reason. Killing a task may, of course, result in the output files being truncated or corrupted.

For interactive processing, it is normal to run one task at a time. However, once a task has started, you can open a parameter dialog to configure the parameters in preparation for running the next task. If you press the 'Run' button before the previous task has finished, you will be given the option of running the task immediately or queuing it for later execution:

- Immediate execution is appropriate if the tasks make use of separate files, so that they may be run in parallel.
- Queuing is appropriate if the tasks form a processing chain, where each task makes use of output from the previous task.

The next queued task is started automatically, when all currently running tasks have finished. Tasks are logged when they are started.

When a task is started, or queued for later execution, the parameter values are saved along with the task. Consequenty, editing the parameters and queuing a second instance of a task will not affect the parameters of a task that is already running or queued.

The current implementation has the following limitations:

- Changing preferences which affect a task, such as the setting of ODF/CCF directories, will affect queued tasks which have not yet started.
- The 'kill' command on the 'Task' menu kills the last task started. It is only possible to kill an earlier task by killing all later tasks.
- The 'kill' command cannot be used to remove a queued task which has not yet started.
- If a queued task fails, the remaining tasks in the queue are still executed.

4.7 Browsing a dataset

The *dataset browser* is invoked from various widgets in the parameter dialog, to allow the selection of datasets and their component tables, columns and arrays. (see Fig. 4.1). The browser can also be invoked from the 'File' menu or tool-bar, simply to inspect data, without making a selection.

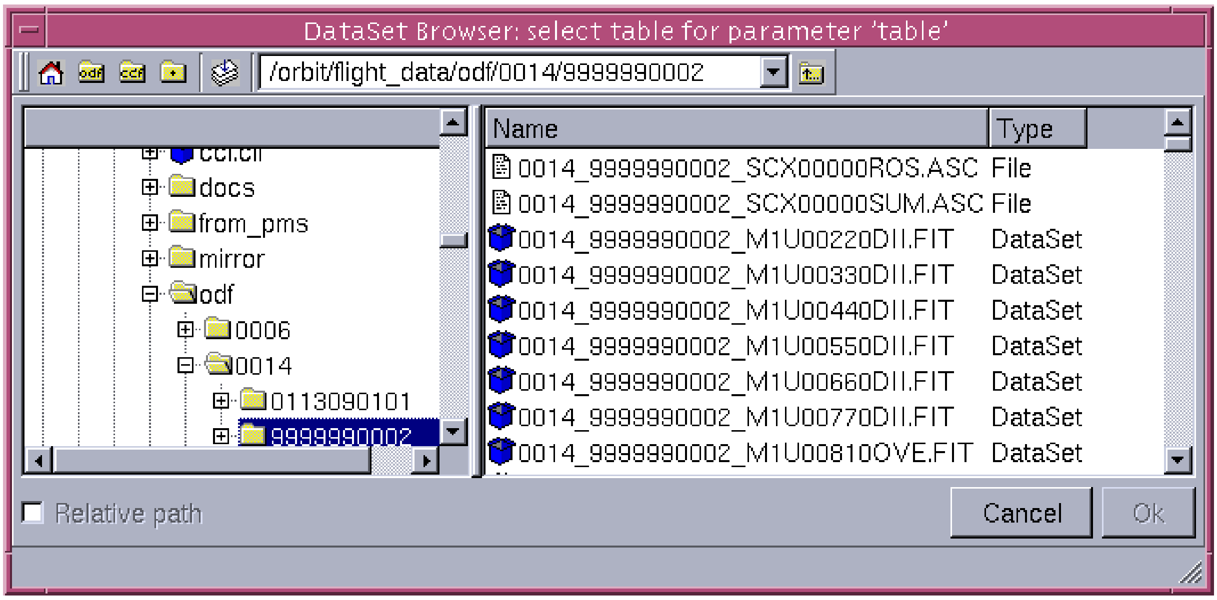


Figure 4.1: SAS dataset browser

The left pane of the browser window shows a hierarchical view of the file system, extending from the root directory, through datasets to their component tables, arrays and columns. Items in the hierarchical view can be expanded by clicking in their 'Name' column. A +/− sign shows whether an item is collapsed or expanded. The browser does not display ordinary (non-FITS) files, to avoid unnecessary clutter.

Clicking on an item in the 'Type' or 'Data' column will cause it to be highlighted, as indicated by rectangle around the name. The right pane shows information on the highlighted item. A tab-bar, along the lower edge, allows the attributes, history or comments to be displayed. If the highlighted item does not have a particular property (e.g. history), the corresponding tab is greyed out.

The right pane and the attribute page of the left pane are arranged in columns. The columns may be sorted in various ways, by clicking on the appropriate column heading. The panes can be resized with the small handle on the the divider line, which separates them.

Double-clicking on a data item (column or array) will pop-up a window showing further information. For a column, this is a scrolling view of the values; for an array, it currently just shows the size of the array.

Warning: Do not try to display the contents of very large columns, as you could have to wait a long while for the data to be displayed. This will be optimised in a later version of the browser.

When the browser is invoked from a parameter dialog, the 'Select' column will show the items which may be selected. For example, a 'Column' parameter allows a column to be selected by clicking in the 'Select' column. A circle indicates that only one item may be selected, as with a 'radio button'; selecting one item deselects the others. A square indicates that multiple selection is possible, as with check-boxes; this is used for list parameter types (e.g. column-list). A '...' in the 'Select' column indicates that a selected item is not visible because its container is collapsed.

When the browser is invoked from a parameter dialog, it returns the selected items as the parameter value, when the 'Ok' button is pressed. The 'Absolute path' check-box determines whether the parameter is specified using an absolute file path or a path relative to the current directory (in which the SAS was started). The 'Cancel' button may be used to close the browser without changing the previous parameter value.

The browser may also be invoked from the 'File' menu or tool-bar, simply to examine data without making a selection. In this case, the 'Cancel' button and 'Absolute path' check-box are not displayed, since they are not relevant.

At the bottom of the browser window are four buttons which can be used to quickly locate some commonly used directories. These are the ODF directory, the CCF directory, the user's home directory and the current working directory from which the SAS GUI was started.

4.8 Using the log

The lower part of the SAS GUI is a *log pane*, which records commands and messages. Each time a task is executed from the GUI, an equivalent command-line is written to the log. These lines have the prefix '@@', so that they may be easily identified. The 'File' menu has an option to extract the commands and save them as an executable script. You may wish to edit the script and replace some file names by command-line arguments, so that the script can be run on different data.

The log also records warnings and error messages (see section 4.9), received from the task. These are prefixed with '**', so that they are clearly visible. Other output from the task (written to STDOUT) is logged without any prefix.

As well as being displayed in the SAS window, the log is also written to the file 'sas_log', in the directory from which the GUI was invoked.

The log may be printed using the 'Print log...' command on the 'File' menu.

4.9 Errors and warnings

Tasks which are run from the GUI can generate various error messages, warnings and information messages. These messages are written to the SAS log (see section 4.8) and may also pop up a dialog to alert the user.

Warning dialogs include a check-box, which allows subsequent occurences of the same warning to be ignored, so that no further user interaction is needed. However, all occurences are still written to the log. The warning is re-enabled next time the task is run.

4.10 User preferences

The 'Preferences...' item on the 'File' menu allows various features of the GUI to be customised to suit the individual user. Environment variables may be used to set the default values of some of the options:

SAS_CCF	Set the directory from which CCFs are read
SAS_VERBOSITY	Set the verbosity of messages from tasks

4.11 Using on-line help

The SAS GUI provides 3 levels of help:

- Yellow *tool tips* pop up when the cursor is placed over an item.
- The main window has a *help tool*, in the tool bar; select the help tool and click it on an item to obtain more detailed help.
- The Help menu provides access to the on-line documentation (currently just the GUI documentation).

The on-line documentation requires that Netscape is already running before selecting 'Help'. The help menu passes the URL of the SAS documentation to Netscape.

4.12 Example task

The GUI package contains a task for testing the GUI, called 'gui_test_task'. This includes examples of all of the available parameter types and layout constructs. There are also options for forcing various messages, warnings and errors.

Chapter 5

Analysis of EPIC camera data

5.1 The EPIC data package

After reception of the CD-ROMs containing the XMM observation data package, the guest investigator willing to analyse EPIC data is advised to verify the EPIC related content of the data package, to view the EPIC pipeline products and subsequently to establish a plan for analysing EPIC data. The EPIC data package includes the ODF files, the CCF files and the pipeline products.

5.1.1 The EPIC MOS Raw Observation Data Files

For each of the two MOS camera, the EPIC related ODF may contain some of the following files depending on the selected observation mode.

- one EPIC MOS auxiliary file per exposure which provides a detailled description of each frame recorded during the exposure,
- "imaging mode" event list files,
- "timing mode" event list files,
- offset/variance mode event files,
- counting cycle report auxiliary files,
- diagnostic mode image files,
- reduced imaging mode event list files,

The relevant files for scientific analysis are the "imaging mode" event list files, the "timing mode" event list files and the auxiliary files. One imaging mode event list file is produced per CCD by the "Full Window" and "Partial Window" instrumental modes. In the "Partial Window" mode, the imaging area of the central chip (CCD0) is reduced to 100x100 or 300x300 pixels. The 6 other surrounding CCDs still produce imaging mode event list files using the 600 x 600 pixels area.

Each of the seven EPIC MOS event list files consists of a primary header and a binary table called EVENTS. The above files can be recognized by the value of the keyword DATATYPE (either IMAGE.EL, TIME.EL or AUX.EL) in their binary table header. The imaging mode event list files have nine columns which are described in table 5.1.1. The "timing mode" event list files have five columns described in table 5.1.1. The auxiliary files have twelve columns described in table 5.1.1. The structure of these files is described in the XMM Observation Data File Handbook.

File name	Description of the file content
0001-0000000000-M1S00100AUX.FIT	EPIC MOS 1 auxiliary file
0001-0000000000-M1S00110IME.FIT	EPIC MOS 1 CCD 1 event list
0001-0000000000-M1S00120IME.FIT	EPIC MOS 1 CCD 2 event list
0001-0000000000-M1S00130IME.FIT	EPIC MOS 1 CCD 3 event list
0001-0000000000-M1S00140IME.FIT	EPIC MOS 1 CCD 4 event list
0001-0000000000-M1S00150IME.FIT	EPIC MOS 1 CCD 5 event list
0001-0000000000-M1S00160IME.FIT	EPIC MOS 1 CCD 6 event list
0001-0000000000-M1S00170IME.FIT	EPIC MOS 1 CCD 7 event list
0001-0000000000-M1X00100PEH.FIT	EPIC MOS 1 periodic HK file
0001-0000000000-M2S00100AUX.FIT	EPIC MOS 2 auxiliary file
0001-0000000000-M2S00110IME.FIT	EPIC MOS 2 CCD 1 event list
0001-0000000000-M2S00120IME.FIT	EPIC MOS 2 CCD 2 event list
0001-0000000000-M2S00130IME.FIT	EPIC MOS 2 CCD 3 event list
0001-0000000000-M2S00140IME.FIT	EPIC MOS 2 CCD 4 event list
0001-0000000000-M2S00150IME.FIT	EPIC MOS 2 CCD 5 event list
0001-0000000000-M2S00160IME.FIT	EPIC MOS 2 CCD 6 event list
0001-0000000000-M2S00170IME.FIT	EPIC MOS 2 CCD 7 event list
0001-0000000000-M2X00100PEH.FIT	EPIC MOS 2 periodic HK file

Table 5.1: Files associated with a MOS 1 and a MOS 2 exposure in full window mode

Colum name	Unit	Column description
FRAME	counter	frame number
RAWX	pixel	event positions in CCD pixel X coordinate
RAWY	pixel	event positions in CCD pixel Y coordinate
ENERGY1	channel	energy in the central pixel
ENERGY2	channel	energy in the 8 peripherical pixels above threshold
ENERGY3	channel	energy in the 8 peripherical pixels below threshold
ENERGY4	channel	energy in the 16 outermost pixels of the 5x5 pixel pattern
PATTERN		pattern number
PERIPIX		number of peripheric pixels above a threshold counter

Table 5.2: Column description of an imaging mode event list file

Colum name	Unit	Column description
FRAME	counter	frame number
RAWX	pixel	event positions in CCD pixel X coordinate
RAWY	pixel	event positions in CCD pixel Y coordinate
ENERGY12	channel	energy in the nine central pixels
PATTERN		pattern number
PERIPIX		number of peripheric pixels above a threshold counter

Table 5.3: Column description of a timing mode event list file

Column name	Unit	Column description
FRAME	counter	frame number
CCDID		CCD identifier
CCDNODE		CCD node identifier
FTCOARSE	seconds	coarse frame time
FTFINE		fine frame time
NPIXEL	pixel	pixel count
NVALID	counts	number of valid event
NBELOW	counts	number of event rejected by lower thresholding
NABOVE	counts	number of event rejected by upper thresholding
GATTIFLG		GATTI flag
FIFOOVF		FIFO overflow flag

Table 5.4: Column description of the auxiliary file

5.1.2 The EPIC pn Raw Observation Data Files

For the EPIC pn camera, the EPIC related ODF contains the following files.

- one EPIC pn auxiliary file per exposure which provides a detailled description of each frame recorded during the exposure,
- "imaging mode", "timing mode" or "burst mode" event list files,
- offset data image files,
- noise data image files,
- discarded lines image files,
- counting cycle report auxiliary files,
- "diagnostic mode" image files.

The relevant files for scientific analysis are the imaging mode event list files, the timing mode event list file, the burst mode event list file and the auxiliary files. The structure of these files is described in the XMM Observation Data File Handbook. One imaging mode event list file is produced per CCD by the "Full Window" and "Partial Window" instrumental modes. Only CCD 4 can be operated in timing, small window or burst mode. Hence the timing, small window and burst instrumental modes produce event list data only for this chip. In these modes, the other chips do not collect data.

File name	Description of the file content
0001-0000000000-PNS00100AUX.FIT	EPIC pn auxiliary file
0001-0000000000-PNS00101IME.FIT	EPIC pn CCD 1 event list
0001-0000000000-PNS00102IME.FIT	EPIC pn CCD 2 event list
0001-0000000000-PNS00103IME.FIT	EPIC pn CCD 3 event list
0001-0000000000-PNS00104IME.FIT	EPIC pn CCD 4 event list
0001-0000000000-PNS00105IME.FIT	EPIC pn CCD 5 event list
0001-0000000000-PNS00106IME.FIT	EPIC pn CCD 6 event list
0001-0000000000-PNS00107IME.FIT	EPIC pn CCD 7 event list
0001-0000000000-PNS00108IME.FIT	EPIC pn CCD 8 event list
0001-0000000000-PNS00109IME.FIT	EPIC pn CCD 9 event list
0001-0000000000-PNS00110IME.FIT	EPIC pn CCD 10 event list
0001-0000000000-PNS00111IME.FIT	EPIC pn CCD 11 event list
0001-0000000000-PNS00112IME.FIT	EPIC pn CCD 12 event list
0001-0000000000-PNX00100PAH.FIT	EPIC pn periodic HK file
0001-0000000000-PNX00100PMH.FIT	EPIC pn main periodic HK file

Table 5.5: ODF files associated with a single EPIC pn exposure

Each of the 12 EPIC PN event list files consists of a primary header and a binary table called EVENTS. The PN EVENTS tables generated in imaging mode, timing mode or burst mode have four columns which are described in table 5.1.2. The auxiliary files contain two binary tables called respectively PNAUX1 and PNAUX2. They are described in table 5.1.2 and 5.1.2 respectively.

Colum name	Unit	Column description
FRAME	counter	frame number
RAWX	pixel	event positions in CCD pixel X coordinate
RAWY	pixel	event positions in CCD pixel Y coordinate
ENERGY	channel	uncalibrated raw energy detected in the pixel

Table 5.6: Column description of the pn event list files. Note that in timing and burst mode RAWY is not a spatial event position but contains information on the detection time.

Colum name	Unit	Column description
FRAME	counter	frame number
CYCLE	counter	cycle identifier
FTCOARSE	second	coarse frame time
FTFINE		fine frame time
CCDID		CCD identifier
QUADRANT		quadrant identifier

Table 5.7: Column description of the first table extension of pn auxiliary files

Colum name	Unit	Column description
CYCLE	counter	cycle identifier
QUADRANT		quadrant identifier
NABOVE	counts	ATHR counter
NDEFA		DEFA counter
NEPDH		EPDH counter
NDISCLIN		discarder line counter
MCOMMODE		mean common mode

Table 5.8: Column description of the second table extension of pn auxiliary files

5.1.3 The EPIC Current Calibration Files

XMM−Newton investigators receive both calibrated events files and data in a raw unprocessed form. In order to run the calibration processing themselves and to correct raw data from the instrument response, they also receive the current calibration files which describe the physical properties of the EPIC instrument at the time of the observation. The information contained in the current calibration files is accessed in particular by the *arfgen* and *rmfgen* tasks through the calibration access layer. The *arfgen* and *rmfgen* tasks generate response matrice files which describe the instrument spectral response. These files are used by spectral fitting techniques which fold the instrument response with simulated spectra for comparison with observed data.

The following table provides a list of the EPIC calibration files. For a detailed description of these files and of their usage, the user is refered to the CCF interface document and to the Calibration Access and Data Handbook. EPIC calibration can be accessed using the SAS command *calview*.

File name	File Content
ADUConv	Conversion factors of the digitized signal to pulse invariant and energy units (keV)
Background	Event files for correlation to ERM spectra.
BadPix	Table of bad columns/pixels including true flickering pixels.
CalSrcData	Spectrum and intensity maps of the on-board calibration sources (TBD).
Contam	Set of coefficients used to model the effect of contaminants (TBD).
CTI	Average charge tranfer inefficiency values for the relevant clock shifts.
DarkCurrent	Parametrized dark current to estimate contribution to false signal.
DarkFrame	Average dark image and offset vectors for a few standard cases.
Efficiency	Detection efficiency versus energy used in the quick look analysis.
EvtSizeDist	Reference event pattern distribution (normalized histograms) vs energy.
FilTransV	Filter transmission curves in the visible spectral range.
FilTransX	Filter transmission curves in the X-ray spectral range.
HKParmInt	House keeping parameter ranges used to generate good time intervals.
LinCoord	matrix which converts CCD pixel coordinates into angular offsets from the boresight.
MiscData	Miscellaneous data.
ModeParam	Main observation modes parameters as a function of user selectable parameters.
PSFPileUp	Coefficients describing the effect of pileup vs countrate on the PSF (TBD).
QuantumEf	Spatially varying CCD quantum efficiency.
Redist	CCD energy redistribution data.
RedistPileUp	coefficients describing the effect of pileup vs countrate on the RMF (TBD).
TimeCorr	time offset corrections to convert S/C time stamp to mean photon absorption time.

Table 5.9: Calibration files of the EPIC camera (one set per camera)

5.2 Inspecting the EPIC raw data

The structure and content of the observation data files can easily be inspected with the ftools package. This can be performed either with the ftools gui viewer (by typing e.g fv "file name") or with the ftools command lines (such as fstruct, flcol, fstatistic, fplot, fhisto ..etc). The user is refered to the ftools manual or fhelp command for further information. The spatial distribution of the events in each of the EVENTS tables can also be inspected using the SAS command *imageanalyse*. Alternatively, the command *evselect* can be used to filter the EVENTS tables and to generate FITS images. Subsequent visualization and preliminary analysis can then be performed with the command *imgdisplay*. However, all the above funtionnality are integrated in the task *xmmselect*. *xmmselect* can be used as follows for the inspection of EPIC MOS raw data.

Run XMMselect

```
double click on xmmselect in the SAS GUI
double click on the double dot to view your directories
select the event list table of a raw EPIC .IME file
press OK
press run
```

A window shall appear with all the column name of the selected event table. For a raw EPIC MOS image file, these are FRAME, RAWX, RAWY, ENERGY1, ENERGY2, ENERGY3, ENERGY4, PATTERN and PERIPIX.

View the raw image

```
select the square button in front of RAWX and RAWY
press the button IMAGE (the evselect window shall appear)
press the button IMAGE in the evselect window
select the appropriate image size (e.g 600 x 600 for MOS full window)
press run
```

An ds9 image appears which is accumulated from individual X-ray events, and false colour coded according to the number of events which were collected in each CCD pixel. The image can be further analysed by the interactive display tool. The image may show a certain number of artefact which can be filtered as follows. Note that these artefacts are present in raw data but are filtered out by the pipeline processing and do not appear in calibrated event lists.

Filter the event list table

```
e.g select behind the button ENERGYE1 a range included between 50 and 3000
press the button ENERGYE1
```

The expression "ENERGYE1 in [50:3000]" shall appear in the upper expression selection window of xmmselect. This ENERGYE1 range is a possible filter against spurious low and high energy events.

```
e.g select behind the button PATTERN a range lower than 12
press the button PATTERN
```

The expression "(PATTERN in [0:12])" shall appear in the upper expression selection window of *xmmselect*, thus indicating that only X-ray related events will now be considered. The XMM user may be interested by e.g a specific point source and can select its associated events as follows.

Select a region

```
display the raw image as described above
select a region (e.g a circle around the point source of interest)
press 2Dregion in the XMMselect window
```

An expression of the type "((RAWX,RAWY) IN circle(317.75,293.25,20)) shall appear in the upper expression selection window of xmmselect, thus indicating that only X-ray related events related to the source of interest are selected. A few quick analysis steps can then be performed on this source as follows.

View a PHA histogram

```
press the circle in front of ENERGYE1
press the button histogram (an evselect window shall appear)
press the button histogram in the evselect window
press run
```

An histogram of the number of occurrences of each PHA channel will be displayed. The PHA channel (number of ADUs generated by an X-ray event) is a proxy for the energy of the incoming X-rays, so that the PHA histogram is analogous to a raw X-ray spectrum.

View a frame count histogram

```
press the circle in front of FRAME
press the button histogram (an evselect window shall appear)
press the button histogram in the evselect window
press run
```

An histogram of the number of events occurring in each frame is displayed. This is a proxy for a light curve, as the frame duration is usually constant.

View a pattern histogram

If unexpected results are obtained, some further tests can be performed, including the display of the pattern histogram.

```
press the circle in front of the button PATTERN
press the button histogram (an evselect window shall appear)
press the button histogram in the evselect window
press run
```

For the MOS cameras, the event are transmitted as one of 32 identified pattern types. Most of the events in the MOS cameras are normally within pattern 0−4. The distribution among patterns help to distinguish whether the CCD is working correctly.

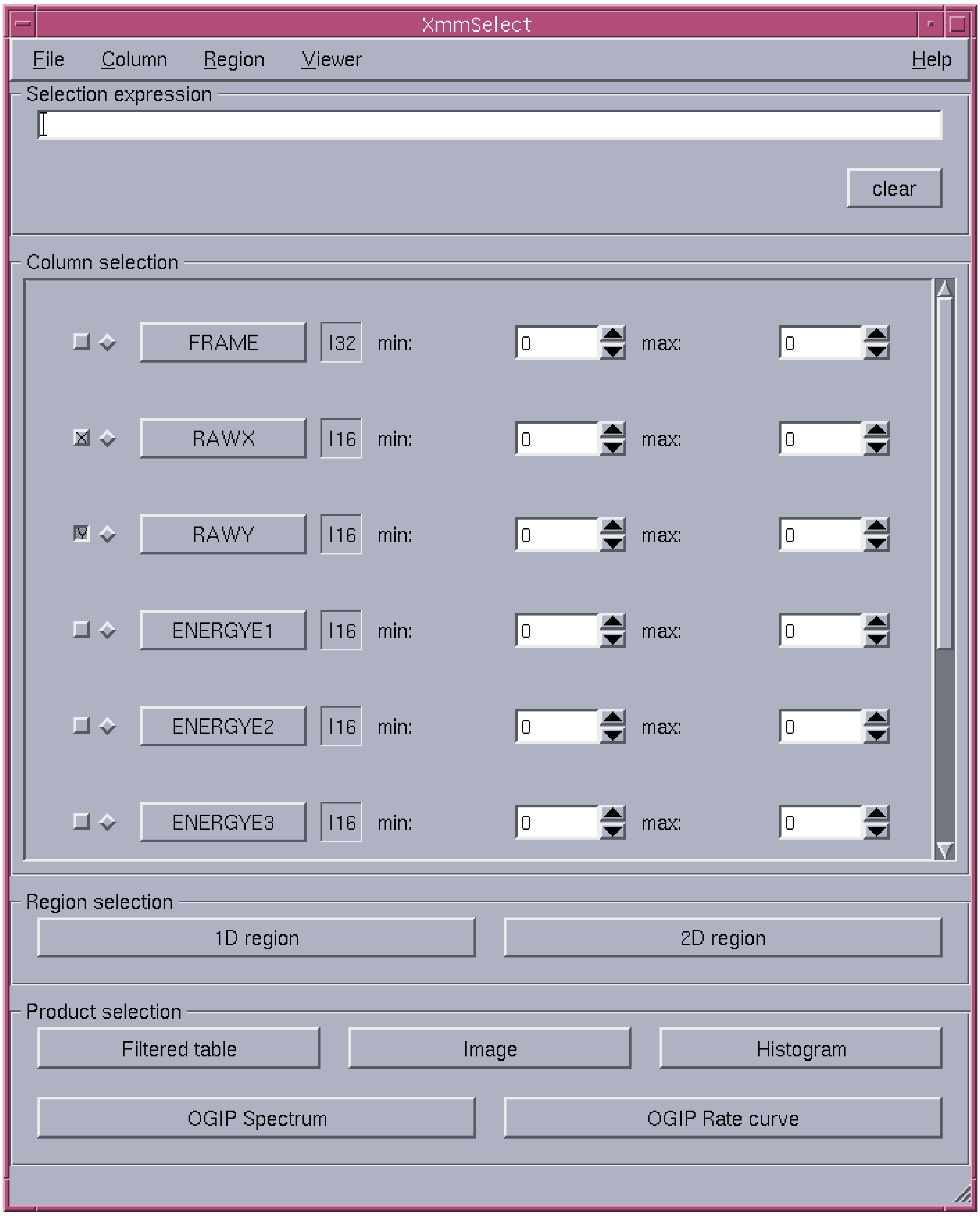


Figure 5.1: XMMselect window

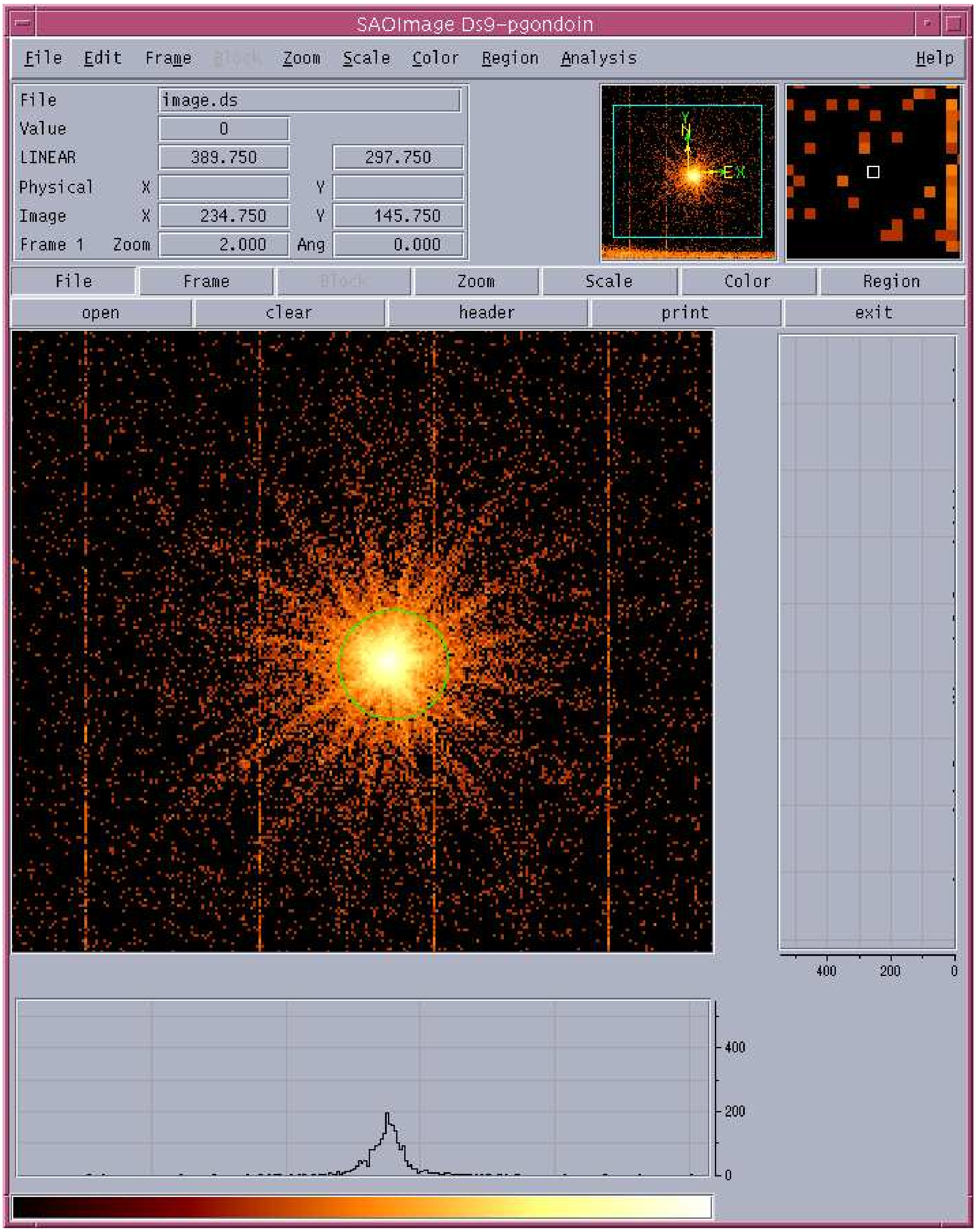


Figure 5.2: A region can be selected using the mouse in a Ds9 image and can then be exported to the SAS by pressing the button 2Dregion in the xmmselect window.

5.3 Running the EPIC pipeline processing

The datapackage received by the investigator contains EPIC pn and EPIC MOS calibrated event lists which normally do not neccessitate further processing for calibration. However, the *emproc* and *epproc* scripts of the pipeline processing batch are made available to the investigator to run the pipeline calibration in an interactive way. A third possibility for the investigator is to run default pipeline processing chains hereafter called *emchain* and *epchain*. The *emchain* and *epchain* tasks run the calibration part of the EPIC MOS and EPIC PN pipelines in their simplest form. They process all ODF components without any interaction from the user. Their calibrated event list ouput is accurate enough to perform quick look analysis. The batch processing tasks are run by specifying the path to the odf directory and applying the *emproc* or *epproc* commands.

```
> setenv SAS_CCF ‘path‘/ccf.cif
> setenv SAS_ODF ‘path‘/odf
> emproc &
or/and
> epproc &
```

5.3.1 Running the EPIC MOS pipeline processing

A description of the pipeline processing functional organigram for the EPIC MOS chain is given in Fig. 4.3. The *emchain* script chains and loops over all first-level EPIC MOS tasks to produce calibrated event lists for all exposures. All files are recognized by their name. Input files are looked for in the directory entered via the odf parameter or the SAS_ODF environment variable, which must also contain the general ODF files (attitude, time, summary file). Output files are created in the current directory.

The main subroutine (PROCESS_FILES) creates one (or two, if a CCD is operated in TIMING mode) event list for a single exposure, from all relevant ODF material and (if they exist) the good time intervals generated by tabgtigen and the list of bad pixels produced by badpixfind. In a first step it loops over all CCD/nodes, calling in sequence:

1. *emframes* on the auxiliary file, the events file and the HK GTI file (if any), creating a frame file as expected by emevents and a CCD/node specific GTI file which will be reinjected in the final call to evselect .
2. *badpix* on the events list, adding the BADPIX extension. If a bad pixels file exists, it is used instead of the CAL calls not in addition.
3. *emevents* on the events list, the offset/variance file and the frame file, creating a new events list which will be propagated through *attcalc* and *emenergy* to *evlistcomb*.
4. *gtialign* on the HK GTI file and the events file, then the FTOOL mgtime to merge the resulting aligned GTI and the CCD/node specific GTI.
5. *attcalc* on the new events list, filling the X/ Y columns.
6. *emenergy* on the new events list, filling the FLAG, PHA and PI columns.

Then all the event list files created (one per CCD/node) are merged by *evlistcomb*, creating one events list per mode (IMAGING, TIMING). Finally *evselect* is called on the resulting events list(s) for all CCD/nodes. The main program loops over all exposures and instruments (MOS1/MOS2) present in the input directory (looking for event list files).

The exposure, instrument and ccd parameters allow to run*emchain* on a subset of the data files (select a single exposure, a single instrument or a single CCD). The pipeline processing creates in the current directory a list of files which description is given in the following table. All the pipeline processing task can be run individually by the user.

File name	File description
0001-0000000000-M1S00100IME.FIT	MOS1 concatenated calibrated event data set
0001-0000000000-M1S00110IME.FIT-filtered.ds	calibrated event data set for MOS 1 CCD 1
0001-0000000000-M1S00120IME.FIT-filtered.ds	calibrated event data set for MOS 1 CCD 2
0001-0000000000-M1S00130IME.FIT-filtered.ds	calibrated event data set for MOS 1 CCD 3
0001-0000000000-M1S00140IME.FIT-filtered.ds	calibrated event data set for MOS 1 CCD 4
0001-0000000000-M1S00150IME.FIT-filtered.ds	calibrated event data set for MOS 1 CCD 5
0001-0000000000-M1S00160IME.FIT-filtered.ds	calibrated event data set for MOS 1 CCD 6
0001-0000000000-M1S00170IME.FIT-filtered.ds	calibrated event data set for MOS 1 CCD 7
0001-0000000000-M2S00100IME.FIT	MOS2 concatenated calibrated event data set
0001-0000000000-M2S00110IME.FIT-filtered.ds	calibrated event data set for MOS 2 CCD 1
0001-0000000000-M2S00120IME.FIT-filtered.ds	calibrated event data set for MOS 2 CCD 2
0001-0000000000-M2S00130IME.FIT-filtered.ds	calibrated event data set for MOS 2 CCD 3
0001-0000000000-M2S00140IME.FIT-filtered.ds	calibrated event data set for MOS 2 CCD 4
0001-0000000000-M2S00150IME.FIT-filtered.ds	calibrated event data set for MOS 2 CCD 5
0001-0000000000-M2S00160IME.FIT-filtered.ds	calibrated event data set for MOS 2 CCD 6
0001-0000000000-M2S00170IME.FIT-filtered.ds	calibrated event data set for MOS 2 CCD 7

Table 5.10: output files of the EPIC MOS pipeline processing

In the EPIC MOS imaging mode, the EVENTS binary table of the calibrated event list files contain 12 columns i.e TIME, RAWX, RAWY, DETX, DETY, X, Y, PHA, PI, FLAG PATTERN and CCD number. DETX and DETY are the event position in the focal plane array. X and Y are the event position in sky coordinate. PHA is the pulse analyser channel and PI the pulse independant channel.

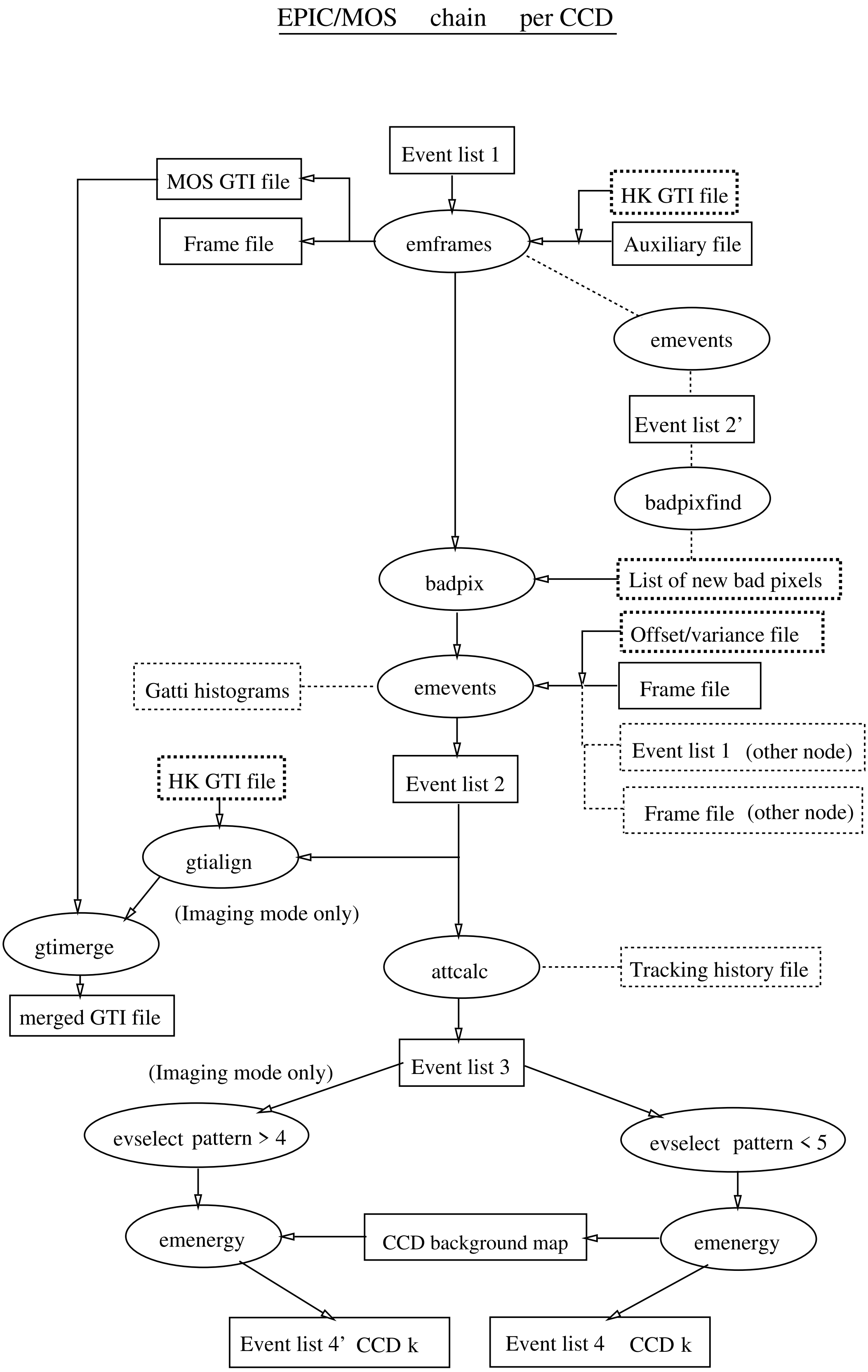


Figure 5.3: Organisation of the EPIC-MOS chain at CCD/node level with file inputs. The files in the bold dashed boxes are either used if they exits or otherwise produced. The files in dashed boxes are options of the individual tasks not used in the current chain.

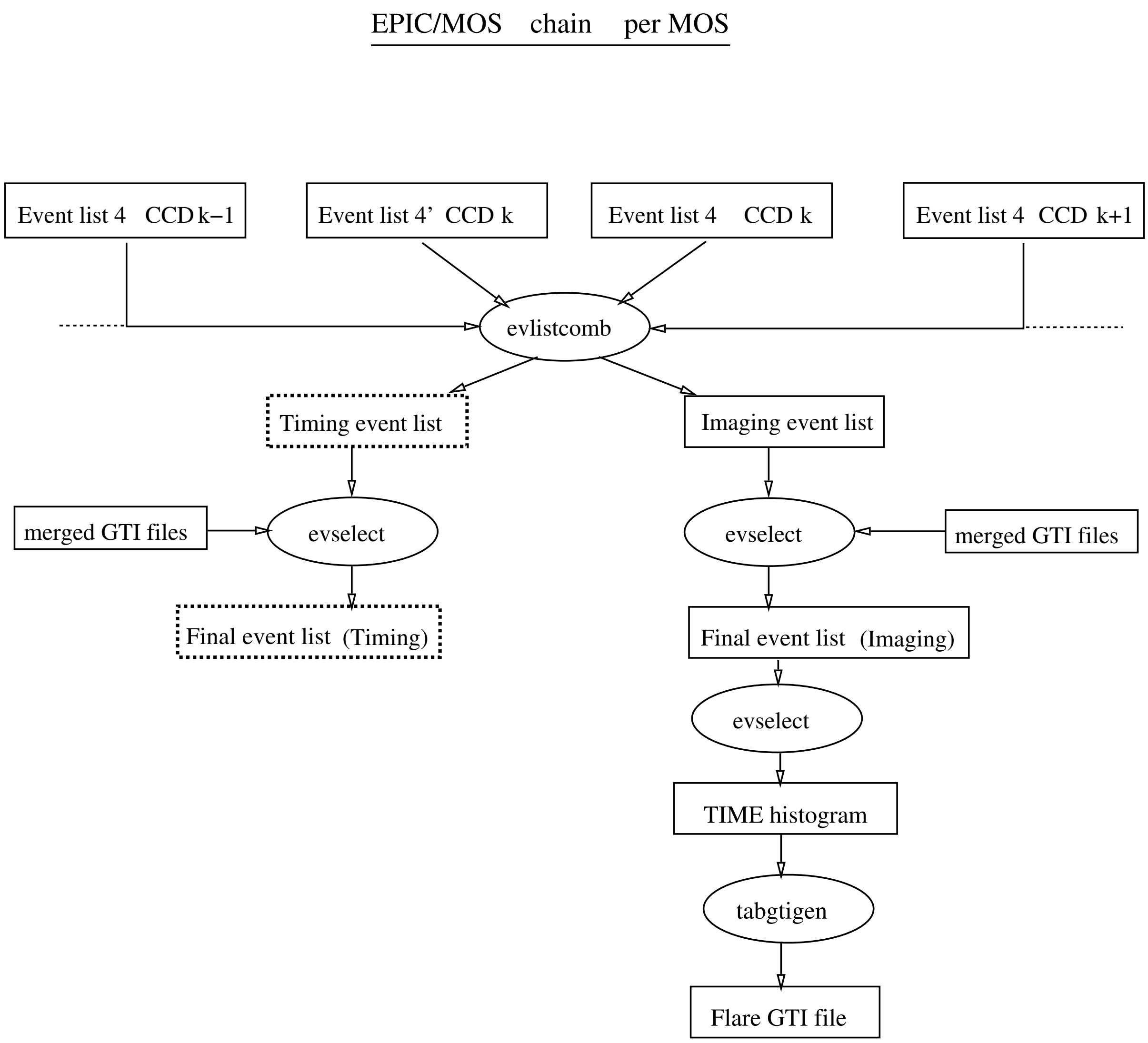


Figure 5.4: Organisation of the EPIC-MOS chain: merging the event lists of different CCDs.

5.3.2 Running the EPIC pn pipeline processing

The EPIC pn pipeline task chains and loops over all first-level EPIC PN tasks to produce an event list. The *epchain* script is executed with a number of command line input parameters. Input files corresponding to the specified readout mode, CCD and exposure number are searched for in the given ODF directory. Badpixfind files (created by *badpixfind* and to be used by *badpix*) will be placed in badpixset. Output files are created in the current working directory.

The PN mini pipeline is sketched out in Fig.5.5. At the beginning of the *epchain* script, the task *atthkgen* creates an attitude history file, called atthk.dat, which will be used by *attcalc* . The main subchain (*epframes*,*badpixfind*, *badpix*, *epevents* and *attcalc*) creates one event list for a single exposure and for a given list of CCDs from all the relevant ODF material and bad pixel lists. The subchain loops over all specified CCDs, calling in sequence:

1. *epframes* to process a CCD, exposure and datamode specific ODF file, creating the output events list rawevents, the GTI data set and creating badpixfind to find new bad pixels,
2. *badpix* to process the rawevents list, adding the BADPIX extension,
3. *epevents* to process the event list file, flag trailing events to create the calibrated events list events,
4. *attcalc* to calculate the X and Y sky coordinates.

Finally, making use of the common PN and MOS task *evlistcomb*, the CCD specific data sets are merged into a single events list. *evselect* selects all those events arriving in good time intervals and writes the output file. In the EPIC pn imaging mode, the EVENTS binary table of the calibrated event list files contain 16 columns i.e TIME, FRAME, RAWX, RAWY, DETX, DETY, PHA, PHA GAIN, PHA CTI, PI, CCDNR, FLAG, PAT TYP, PAT IND, PAT ORI and PAT ID. A detailled description of the calibrated event files is provided in the XMM data files handbook. All the pipeline processing task mentionned above can be run individually by the user.

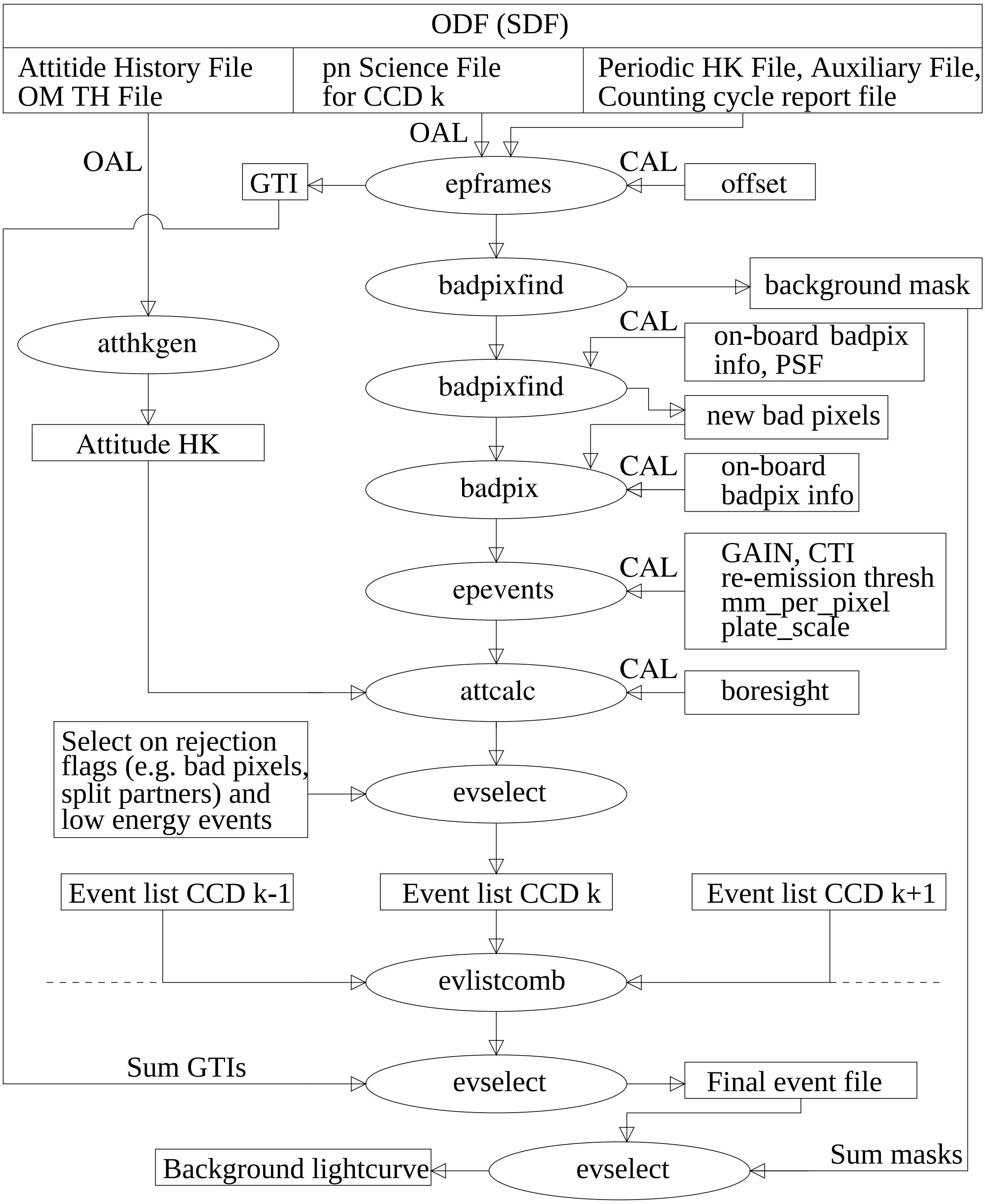


Figure 5.5: Pipeline processing of the EPIC pn observation data

5.4 Filtering calibrated EPIC event lists

EPIC calibrated event lists must be processed to generate data products including images. A first operation is generally to filter the event list within the PI sensitivity range of the selected detector filter combination. Also the user is advised to produce a histogram of TIME values (a light curve) in order to identify the useful period of low background level when the focal plane is not illuminated by low-energy protons. This can be accomplished using the *xmmselect* task. The event lists can then be filtered according to user-specified selection criteria and images can be extracted from this filtered event list.

Before filtering, the user is advised to inspect the backgound lightcurve produced by *epchain* where all sources and bright features/pixels have been masked out.

5.4.1 Filtering EPIC MOS concatenated event lists

Each of the two EPIC MOS cameras consists of seven individual 600 x 600 pixels CCDs. During an exposure, each CCD is read approximately every 3 seconds. Because of telemetry constraints, a real time on-board recognition scheme filters out cosmic-ray tracks and exclusively transmit to the ground the information supposedly related to X-ray events. The on-board recognition scheme looks for a local enhancement of signal in flat fielded images. The signal enhancement is searched in 5 x 5 pixels matrix which is scanned over the full image. The signal is defined with respect to a threshold value set by telecommand for each observation. An event is identified, if in the 5 x 5 pixels matrix, pixels above thresholds formed a predefined pattern.

32 patterns have been predefined. They each correspond to an isolated event i.e to a zone above threshold completely encircled by pixels below threshold. There is however two exceptions. Pattern 30 can be connected to a pixel above threshold on the diagonal of the center pixel. Pattern 31 can have any of the border pixels above threshold. Pattern 30 and pattern 31 are designed to quantify the amount of cosmic-rays extended tracks.

On-ground calibration has shown that soft X-rays mainly generate patterns 0 to 12 corresponding to compact regions of X-ray energy deposition. Pattern 0 events are single pixel events. These comprise most the valid X-ray events with the most accurate energy resolution. Pattern 0 to 12 are the canonical set of valid X-ray events which are well calibrated. Selection of these pattern constitute the best trade-off between detection efficiency and spectral resolution. However, because they deposit energy below the CCD depletion zone, higher energy X-rays also generate pattern 31 events with a probability of 20% and 50% respectively at 6 keV and 9keV. Pattern 31 comprise largely cosmic ray events but can also include pile-up X-ray events. A large density of pattern 30, 31 events (and 26−29 diagonal events) in the core of the telescope point spread function (PSF) is the signature of a piled response which needs careful analysis.

The user is encouraged to test different filtering schemes which could be better suited to their own observation. However, general recommendations are as follows.

- The selection expression (#XMMEA_EM) shall be used to filter out artefacts from the calibrated and concatenated dataset.
- Only pattern 0 to 12 should be kept as X-ray events since pattern with higher numbers are not created by X-ray or are highly contaminated by cosmics.
- Pattern 26 to 29 are mostly created as the result of a two mono-pixel event pile-up. The fraction of such pattern with respect to pattern 0 in a given image area is a good indicator of the pile-up level in that zone (e.g in the core of the Point Spread Function (PSF) generated by a bright target).

- Soft energy (< 1 MeV) protons are stopped in the first microns of silicon of the EPIC CCDs. They give a signal which is absolutely identical to X-rays i.e pattern 0 with normal energy distribution. One possible strategy to recognize a high soft proton flux is to look at the time or frame variation of the background outside the region of interest. High background level and large background fluctuation are indicative of a high soft proton flux. A cleaner image can then be obtained by filtering out those frames which correspond to a high background level.

The *xmmselect* task is well suited to the above filtering exercise. Alternatively, the task *evselect* can be used for event selection and product extraction. Images can be displayed with the task *imgdisplay*.

5.4.2 Filtering EPIC pn concatenated event lists

Filtering of EPIC pn dataset is entirely performed by the EPIC pn pipeline processing. The user is only advise to verify the background level as a function of time. The selection expression (#XMMEA_EP) shall be used to remove artefacts from the calibrated and concatenated pn event list. In some case, the quality of the EPIC pn (and MOS) products can be improved by filtering out those of the CCD frames which have a high background level.

5.4.3 Filtering high background periods

In order to remove high background periods from the event list, the user could apply the following recipe.

- Build an histogram of the time column in the calibrated events list using only events with energy greater than 10keV. This can be performed using *xmmselect* by entering 10000 in the PI lower limit box, pressing the PI button, or alternatively entering "(PI > 10000)" in the expression box. The round radio button in the TIME row shall then be pressed. A suitable time bin shall be selected for the histogram accumulation e.g 25, 50 or 100 seconds depending on the exposure duration.
- Examine the histogram produced and identigy periods with a constant low count rate and periods with a high background level. Select a suitable count rate threshold which lies a little above the low background rate.
- Assuming that the histogram is named histo.ds, a new GTI file can be created from the selected count rate threshold using a command line as follows (changing the count rate value accordingly):

  ```
  tabgtigen table=histo.ds gtiset=BKG_GTI.fits expression='COUNTS < 100'
  ```

- This GTI selection can then be passed to the existing events list using:

  ```
  evselect expression=GTI(BKG_GTI.fits,TIME)' .......
  ```

  When using *xmmselect* from the GUI, the above expression shall just be typed into the selection window.

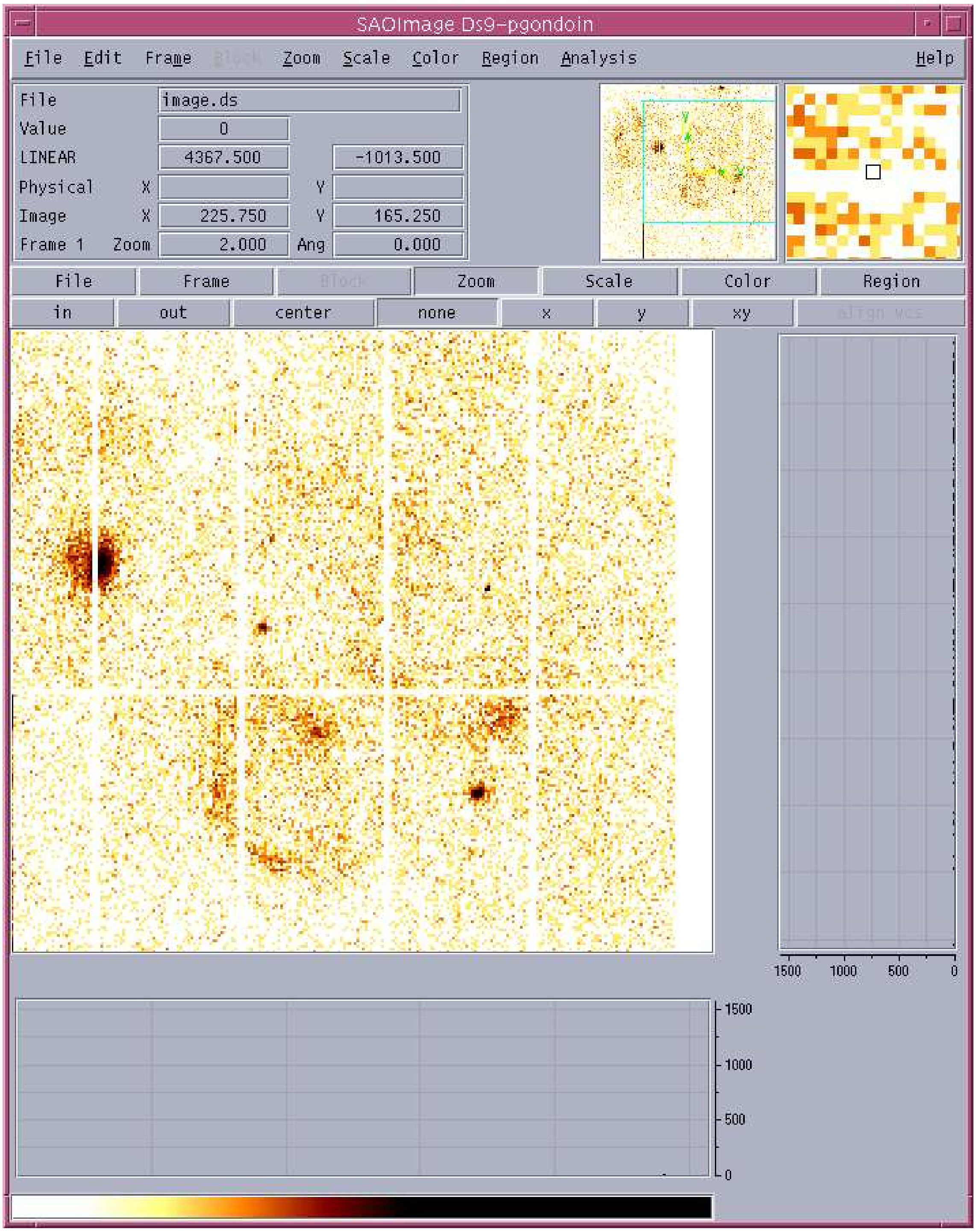


Figure 5.6: Combined image of pn CCD chips

5.5 Merging event lists

Two EPIC files from two exposures (even from different observations or cameras) can be merged using the task *merge*. These files may be the PPS product EPIC event lists or two other additional files (attitude files, orbit files etc...). The files are checked via their extension names (EVENTS, ATTHK, ORBIT) for their type, and depending on the type, certain attributes are checked to see whether the files are compatible with each other. In the attitude and orbit cases, any files can be merged together, whereas in the events case, some care should be taken. For instance event files from different modes should not be merged, though perhaps event files from different instruments can. Warnings are given when the event files are deemed not entirely compatible (i.e. their INSTRUME or FILTER keywords are different). Here it is up to the user to interpret the output files sensibly.

Also, in the event files case, the two pointings need to be close to each other. The accuracy of the reprojection degrades with the offset between the two pointings, and warnings will be given when this offset becomes significantly large. Only if the input files pass all the checks does the merging proceed and an output PPS product (event list, attitude file, orbit file) is produced. The individual blocks (e.g. events) are merged in time order, as are the other relevant extensions (e.g. exposure, stdgti, etc...). In the event files case, the sky coordinates are deprojected and reprojected again onto a common sky position (given either by the user, via the parameters ra and dec, or as the mean of the two input pointings), and new attributes are calculated. The size of the image is given by the parameter imagesize.

The output files can then be used by *evselect* and the higher level detection and time analysis tasks. For example, two event files can be merged reprojecting X/Y coordinates to a mean reference point using the following command.

```
merge set1=ev1.fits set2=ev2.fits outset=evout.fits
```

The two event files can also be merged reprojecting X/Y coordinates to a reference point defined by the user.

```
merge set1=ev1.fits set2=ev2.fits outset=evout.fits withuserradec=Y ra=1.1 dec=-0.7
```

5.6 Detecting X-ray sources

Source detection in the EPIC pipeline is a three stage detection process. First the source detection tasks (*eboxdetect* and *emldetect*) are run on each exposure of each EPIC camera separately. In a second stage the detection tasks are run on the combined exposures of each EPIC camera. In the third stage a combined source detection run is performed on all the exposures of all three EPIC cameras. Each detection run results in a final emldetect source list. Assuming that an observation consists of two exposures in each of the three EPIC cameras, there would be six source lists from the first stage, three source lists from the second stage and one final source list from the third stage.

The third detection stage corrects for possible boresight differences between the three EPIC cameras. For this purpose, a cross-correlation of the three second stage emldetect source lists is performed. It determines the boresight offsets between the three cameras. In the third stage these offsets are accounted for by the detection tasks. After the third stage the final emldetect source list is read in by task *srcmatch* and a summary source list is generated. The summary source list contains the main source parameters from the third detection stage as well as pointers to the emldetect source lists from previous detection stages. In addition cross-identifications with the OM source list are provided.

The user can perform her/himself source detection and generate summary source lists using the meta-task *edetectproc*. In order to perform source detection, the user needs at least one FITS file containing an image, and one attitude history file generated by the pipeline. The event dataset used to generate the input image shall also be provided as input under the parameter eventset. The task *edetectproc* can then be run as follows on the command line.

```
edetectproc eventset=filtered.fits attitudetab=atthk.dat imagesets='image.ds'
```

The meta-task *edetectproc* combines all the EPIC source detection tasks in the following sequence.

- *eexpmap*
- *emask*
- *eboxdetect* (in local mode)
- *esplinemap*
- *eboxdetect* (in map mode)
- *emldetect*
- *esensmap*

Up to 18 input EPIC images can be specified. From these, the following products are created.

- exposure maps, one for each input image
- background maps, one for each input image
- sensitivity maps, one for each input image
- 3 source lists (two from *eboxdetect* run in local and map mode, one from *emldetect*).

Several images can be specified as follows.

```
edetectproc eventset=filtered.fits attitudetab=atthk.dat \\
        imagesets='image1.ds image2.ds image3.ds' \\
        pimins='300 1000 2000' pimaxs='1000 2000 10000'
```

The exposure maps are generated over the channel ranges (i.e enrgy bands) defined by the integer lists pimins and pimaxs. These lists should contain a number of element equal to the number of input images. For *edetectproc* to function correctly, the event dataset used to generate the input images shall be provided as input under the parameter eventset. In addition, an attitude history table shall be specified through the parameter attitudetab . A parameter likemin can also be provided to specify the minimum detection likelihood of the task. The user shall be aware that un-removed bad pixels can generate spurious sources.

The tasks *eboxdetect* and *emldetect* convert the count rates for detected sources to fluxes in units of 10^{-11} erg cm^{-2} s^{-1} using the inverse of an energy-to-counts conversion factor for each energy band. These factors are specified using the parameter ecfs. The results of the *eboxdetect* runs can be observed automatically by setting the parameter showdetect to true. This will call the task *srcdisplay* , which will display an image overlayed with the positions of the sources detected by *eboxdetect*.

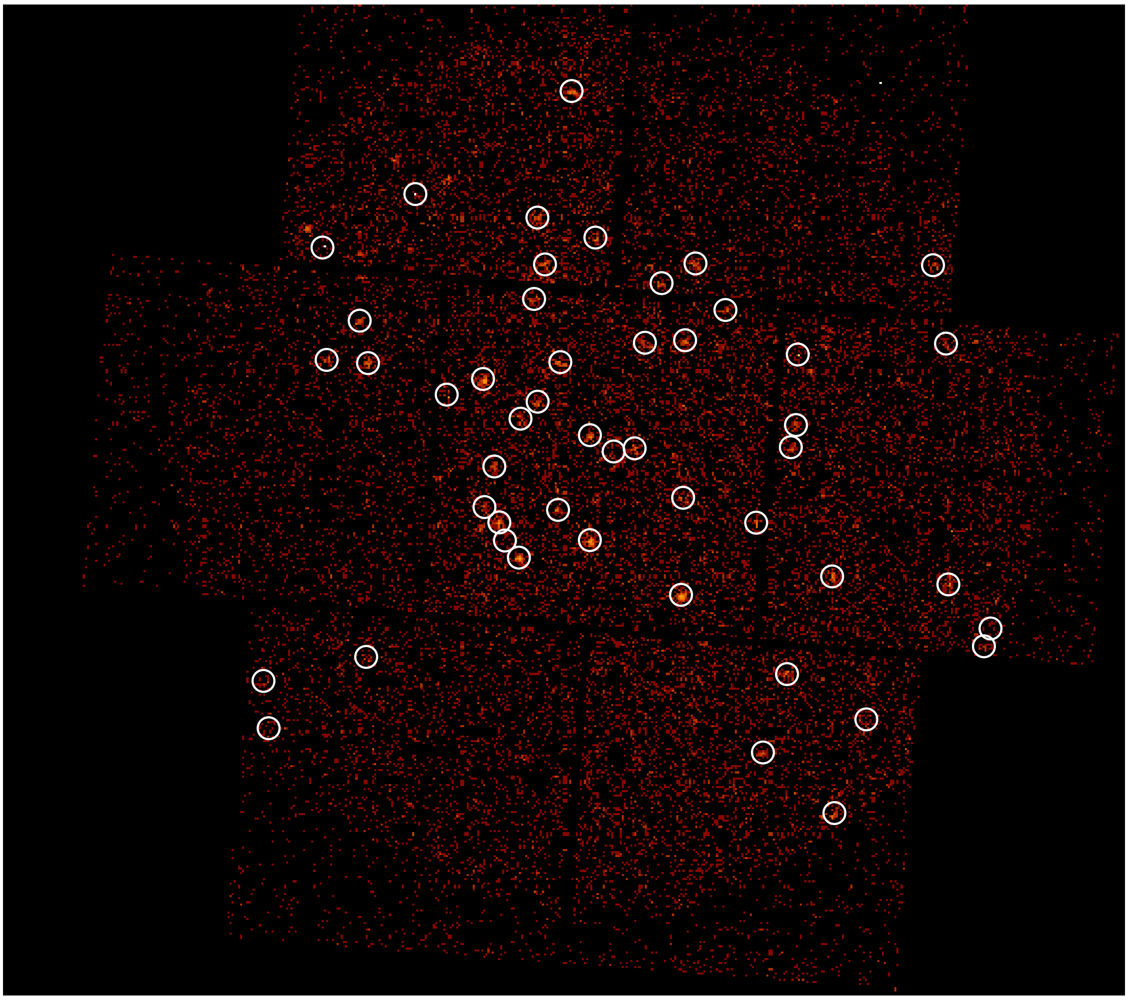


Figure 5.7: Output image of the *srcdisplay* SAS task showing detected source positions.

```
edetectproc eventset=events.ds attitudetab=attitude.ds:
imagesets='image1.dat image2.dat image3.dat image4.dat' \\
showdetect=true uselocal=false \\
pimins='10 10 10 10' pimaxs='10000 10000 10000 10000' \\
```

The image displayed is governed by the parameter withplotimageset. If this parameter is set to true, then the image specified in plotimageset is used, otherwise the first image in the imagesets list is used. The displayed source positions are taken from the output of *eboxdetect* in local mode if uselocal is true, otherwise the output of *eboxdetect* in map mode is used. A DS9 region file corresponding to the displayed source positions can be generated with a name corresponding to the parameter regionfile, if the parameter withregionfile is set to true.

Alternatively, the user can run separately the task *srcdisplay*. This task will display the specified image via an imgdisplay session, and overlays it with circles depicting the positions of sources in an *eboxdetect* output source list. The input image is specified via the parameter imageset and the *eboxdetect* output is specified through the boxlistset parameter. The radius of the displayed circles can be set using the sourceradius parameter. The displayed circles are in fact DS9-type regions, and these regions can be written out to a file for future use (for example, when running a later DS9 session) by setting withregionfile to true, and specifying the desired file name via the regionfile parameter.

```
srcdisplay boxlistset='eboxlist_m.ds' imageset='image.ds' \\
includesources=yes regionfile='regionfile.txt' \\
sourceradius=5 withregionfile=no
```

5.7 Quick analysis of EPIC images

A quick analysis of EPIC images can be performed using the *imageanalyse* task. Regions of interest can be selected using the marker facility provided by the imgdisplay session and hitting the 'SELECT FROM IMAGE' button. This propagates the region definition to the expression widget at the top of the main task window.

Radial profiles can the be generated by hitting the 'RADIAL' button and hitting 'Run' on the two subsequent dialogs. A dsplot session should be displayed, showing the desired profile. To generate a radial profile, the centre must be specified either manually, by editing the 'X POS' and 'Y POS' widgets, either by selecting a position on the image using the POINT marker, and hitting the 'USE IMAGE' button, either by using the centroid determined from statistics or by choosing the maximum of the image.

The 'STATISTICS' button allows statistics to be displayed for the region of interest. An 'IMAGE' button allows to filter the event list around the regions of interest. The original image can be restored by hitting the 'CLEAR' button near the selection expression widget, and hit the 'IMAGE' button again.

The *imageanalyse* task is useful in particular to determine the optimum size of an extraction window for image spatial filtering and further spectral analysis. Alternatively, the azimuthally averaged profile and the encircled energy of a point source image can be computed using the *radial.* The point may be specified exactly with parameters xc and yc, or those parameters may be taken as initial estimates which are then refined by a centering algorithm. No background subtraction is performed by this task, as this can be accomplished with the *arithmetic* task.

The task *regionstat* evaluates a number of statistics for a given region of interest. These are:

- centroid position
- position and value of maximum and minimum
- mean and standard deviation
- total number of counts
- number of pixels and number of bad pixels in the region

The corresponding statistical uncertainties are given as well. The region is given in input as an saoimage compatible region file. If this file is not specified, the whole image is used. The user can optionally specify a quality data set to be used to mask the region of interest. Output positions are expressed in pixels and in WCS coordinates, if the appropriate information is present.

5.8 Extracting X-ray sources and regions

Within the EPIC pipeline, the task *region* automatically determines the optimal source and background extraction regions for a detected source in an XMM observation. The source positions are drawn from the EPIC FITS observation source list produced by the pipeline. The regions are defined in sky coordinates.

The task *region* attempts to maximise the area of the extraction region for each source, whilst ensuring that the extraction regions of neighbouring sources do not overlap, and that the background extraction region for a given source excludes neighbouring sources.

Within the pipeline, the source and background extraction regions are determined as follows:

1. The source celestial coordinates (RA/Dec) are read from the pipeline source list.
2. For each source, the celestial coordinates are transformed to a working coordinate system (arcseconds, tangent plane projection relative to the reference RA/Dec. of the event file).
3. For each source, the source extraction radius is calculated for each source, using the position in the instrument field-of-view. The extraction radius is defined as that at which a user-defined fraction of the PSF is enclosed.
4. The separation to the nearest-neighbour is calculated for each source.
5. In order of increasing nearest-neighbour separation the sources are checked for overlap with their nearest neighbour. If no overlap occurs no action is taken, otherwise the radius is adjusted (see detailed algorithm) and the source flagged as having been adjusted.
6. In a second pass all pairs of sources are checked for overlap. If overlap is detected the size of the larger of the two extraction regions is reduced to eliminate the overlap.
7. The background extraction region is defined as an annulus around the source. The inner radius is the same as the adjusted source extraction radius, the outer radius is three times the inner radius. The background extraction region is checked for overlap with nearby sources. If overlap is found the original source extraction region of the intersecting source is added as an exclusion clause to the background extraction region definition.
8. The source and background regions will be clipped to the edge of the field-of-view.
9. The source and background regions are transformed (scaling + translation) into the same units as the sky positions of the events in the reference event list.

The task *region* can also be used to define the region used for extracting the global background lightcurves and spectra which are also pipeline products.

Alternatively, the task *xmmselect* can be used interactively to extract event lists associated to individual regions. This is normaly performed in two steps. First, an image is displayed and the region of interest is defined. Then the lists of events included in this region are filtered. The task *xmmselect* can be conveniently run in an interactive way through the sas gui.

The filtering process is carried out on an event-by-event basis controlled by user-specified selection criteria which take the form of a single boolean expression. These take the form of single character strings which are composed of a variety of elements including numerical and logical constants, operators, functions, and attributes. Arithmetic and logical operations, file-based filtering, new column or arrays construction and symbolic reference to other arrays can be performed. The user shall consult the *selectlib* package description which details the syntax and semantics of the expressions driving the above operations and give sample expressions to demonstrate typical usage scenarios.

The expression can contain symbolic names of "intrinsic" event attributes, and numeric constants combined with logical and arithmetic operators and functions. Events for which the expression does not evaluate to true will be marked as invalid or discarded from the data set and shall not be considered in the product extraction. *xmmselect* supports selections based on intrinsic event attributes. These attributes are present in the input event list table as corresponding columns, the names of which are to be used in the selection expression. This group further falls into the following sub-categories.

1. Spatial selections in raw pixel (RAWX/RAWY), physical detector (DETX/DETY) or sky pixel (X/Y) space, or in any spatial coordinate system (e.g. TELX/TELY). Special predefined region shapes exists such as CIRCLE, ELLIPSE, ANNULUS, BOX, POLYGON. Spatial filtering with mask images is possible as well.
2. Energy selections in the form of one or more interval specifications in either PHA or PI space.
3. Time selections in the form of interval specifications or Good Time Interval (GTI) files created with e.g tabgtigen.
4. Any other event attributes which are present (e.g. PATTERNID, or CCDID, or WAVELENGTH in case of RGS)

The selection stage yields a modified event list in which the events which do not satisfy the selection expression are removed or appropriately flagged. From this list the extraction stage creates images (and also spectra and ratesfiles). The image creation process can be controlled via various command line parameters such as binning factors, cut-off boundaries, etc. The user is refered to section 5.2 for a detailed example of the *xmmselect* task usage.

5.9 Generating EPIC spectra and time series

EPIC calibrated event lists must be filtered to generate data products such as spectra and timeseries. As for images, this filtering is normaly performed in two steps. First, the event lists are screened to reject spurious data and to select only event lists which contain information of sufficient quality for further scientific analysis. Secondly, the screened data are filtered to construct data subsets adapted to specific spectral or timing analysis. These screening and filtering activities can be performed in an interactive manner by the SAS task *xmmselect*. The *xmmselect* task constructs event lists, images, spectra and time series in a progressive and iterative way along the following steps.

1. Filter the event list under the control of a boolean selection expression
2. Accumulate a product from the events which passed the filters
3. View the product
4. Correlate features in the products with made selections and refine the selection
5. Go back to step 1.

The task *xmmselect* can be conveniently used in an interactive way through the sas gui. An extensive description of the task functionnalities is provided in section 5.2. Alternatively, the task *evselect* can be used for event selection and product extraction. Time series and spectra can be displayed with the tasks *dsplot*. Images can be displayed with the task *imgdisplay*.

5.10 Creating EPIC response matrices

Analysis of EPIC data products is generally performed by specialised software packages including e.g XSPEC or XRONOS. In addition to the calibrated products, some of this packages require the generation of specific files. In particular, the spectral fitting technique used by XSPEC requires a characterization of the EPIC detector response to simulate an output spectrum observed by EPIC. This response function give the probability that an incoming photon of energy E will be detected in a channel I. This discrete function can be calculated as a product of a redistribution matrix file (RMF) by an auxiliary response file (ARF). These response files shield the user from the complexity of the EPIC instrument response which varies accross the field of view.

The user shall first create the response matrix using the *rmfgen* task. The input spectrum shall be specified. The *rmfgen* task can then reformat the detector response and energy bounds according to the information provided by the calibration access layer. It corrects for instrumental effects specific to the event and writes the result to a specified dataset. It groups response data above a threshold value and account for spectral distortion due to pile-up and charge transfer inefficiency (CTI). The RMF dataset also complies with the OGIP standard.

The ARF response file of the EPIC MOS camera shall then be generated by the sas task *arfgen*. This task calculates an effective area curve as a function of energy, to be used in conjunction with the RMF file generated by the sas task *rmfgen*. For each row of the RMF there is a corresponding element in the 1-D ARF. This is normally adjusted by specifying the previously generated response matrix as an input file to the *arfgen* task.

The *arfgen* task generates an ARF files as a function of the following EPIC parameters.

1. Telescope effective area including vignetting by the RGA structure for the MOS cameras,
2. EPIC filter transmission,
3. EPIC CCD quantum efficiency,
4. PSF Wing scattering corrections,
5. Pile-up effect,
6. Spatial response variation over an extended source,
7. Dead time effect in the different imaging modes,
8. Background effects e.g on pile-up,

The above parameters generally depend on the source position in the EPIC field of view. Spatial response variation over an extended source is also taken into account. The ARF file is compliant with the OGIP standard.

Alternatively, ready−made response matrices might be made available from the XMM−Newton web server.

5.11 Scientific analysis of EPIC products

Once the calibrated data products (i.e either images, spectra or timeseries) and supporting files (e.g background files, response matrices ..) have been produced, the investigator can proceed with the scientific analysis of his data. This is generally performed with the support of specialized analysis software such as the XIMAGE, XRONOS and XSPEC packages included in the HEASOFT software suite. Hence, for scientific analysis, the user is refered to the documentation associated with these packages.

The analysis of EPIC data shall however take into account specific characteristics of the XMM telescopes and instruments. As an example, special attention shall be given to the extraction of a background spectrum from an annular region centered on the source of investigation. Due to the specific shape of the XMM telescope point spread function, such a region could indeed contain a non-negligeable fraction of source counts. This contribution could be higher at high energy due to the energy dependance of the scattering wing of the telescope PSF. Its substraction could alter the intrinsic source spectrum.

5.12 Processing examples of EPIC data

5.12.1 Quick start example

After a proper installation of the SAS software package, a quick check of the SAS installation can be performed along the following recipe.

- create and move to a working directory
- point to an observation data file (ODF)

```
setenv SAS_ODF /'path to the selected odf'/
```

- generate a calibration index file (CIF) from a repository of current calibration files (CCFs)

```
cifbuild analysisdate='now' append=no calindexset='ccf.cif' \\
category='XMMCCF' ccfpath=/'path to a ccf repository'/ \\
fileglob='*.ccf|*.CCF' fullpath=yes ignorecategory=no \\
masterindex=no masterindexset='ccf.mif' observationdate='2000-08-10' \\
recurse=no withccfpath=yes withmasterindexset=no \\
withobservationdate=no
```

- point to the newly created ccf.cif calibration index file

```
setenv SAS_CCF ccf.cif
```

- run e.g the EPIC MOS or/and PN pipeline processing

```
emproc &
epproc &
```

- after some time, a set of calibrated event files will have been created into your working directory including the merged event lists of the 7 MOS CCDs and 12 PN CCDs. These files are easily recognizable since they are usually the largest files of the directory. In order to inspect the content of these files, launch the SAS Graphical User Interface (GUI) as follows.

```
sas &
```

- launch the task *xmmselect* by double-clicking on 'xmmselect' in the central GUI window. An *xmmselect* window appear.
- click the double dots button located behind Table and select one of these newly created MOS or PN files on the left panel of the dataset browser. Then click on the table called EVENT on the right panel of the dataset browser. Click on the right mouse button, press select and OK at the bottom of the dataset browser panel.
- press 'Run' in the *xmmselect* window. A large xmmselect window appears.

- click on the square boxes in front of X and Y and click Image. Select the image tab and change ximagesize and yimagesize into 600. Click Run. An image shall be displayed in true sky coordinates in SAOimage−ds9.
- in xmmselect click on the diamond shaped button in front of PI and click OGIP-Spectrum. Select the spectrum tab and click run. The PI spectrum shall be displayed in an xmgr window. In xmgr, zoom in with the magnifying glass button and drag the rectangle which appear around a spectral range of interest. The AS (autoscale) button allows you to zoom out.

5.12.2 Source detection example

The following example illustrates the use of the edetectproc task. The user can work with command lines along the following steps.

- create and move to a working directory
- point to an observation data file (ODF)

```
setenv SAS_ODF /'path to the selected odf'/
```

- generate a calibration index file (CIF) from a repository of current calibration files (CCFs)

```
cifbuild analysisdate='now' append=no calindexset='ccf.cif' \\
category='XMMCCF' ccfpath=/'path to a ccf repository'/ \\
fileglob='*.ccf|*.CCF' fullpath=yes ignorecategory=no \\
masterindex=no masterindexset='ccf.mif' observationdate='2000-08-10' \\
withanalysisdate=yes withobservationdate=yes \\
recurse=no withccfpath=yes withmasterindexset=no
withobservationdate=yes
```

- point to the newly created ccf.cif calibration index file

```
setenv SAS_CCF ccf.cif
```

- run e.g the EPIC MOS or/and PN pipeline processing

```
emproc &
epproc &
```

- define the input event file e.g as the MOS 1 merged event list file.

```
set evfile = *M1S00100IME*.FIT
```

- generate an image

```
evselect --table=$evfile --withfilteredset=true \
 --filteredset=filtered.ds --withimageset=true \
 --xcolumn='X' --ycolumn='Y' --ximagebinsize=100 \
 --yimagebinsize=100 --imageset=image.ds \
--withimagedatatype=true --imagedatatype='Int32'
```

- view the image by launching the *imgdisplay* sas task and opening the previously created image.ds file.

```
imgdisplay &
```

- source detection in this image can then be performed by running *edetectproc*.

```
edetectproc eventset=$evfile attitudetab=atthk.ds imagesets=image.ds
```

- the position of the detected source can be overlayed on the image using the task *srcdisplay*

```
srcdisplay boxlistset='eboxlist_m.ds' imageset='image.ds' \
includesources=yes regionfile='regionfile.txt' \
sourceradius=2 withregionfile=no
```

5.12.3 Spatial filtering example of an event list

The following example illustrates a way to filter out an event list using a boolean expression within the command line.

- specify the observation data file (ODF) directory of interest and its associated calibration index file (CIF).

```
setenv SAS_ODF /'path to the selected odf'/
setenv SAS_CCF ccf.cif
```

- run e.g the EPIC MOS pipeline processing

```
emproc &
```

- define the event file of interest e.g the MOS 1 merged event list

```
set evfile=0001_0000010030_M1S00100IME.FIT
```

- filter the event list by extracting all events with FLAGs smaller than 1, PATTERN included between 1 and 12 and PI energy in the range 300 to 12000 eV.

```
evselect --table=$evfile  \
--withfilteredset=true \
--filteredset=ev_filtered.ds \
--keepfilteroutput=true \
--destruct=true \
--expression='(FLAG<=1)&&(PATTERN<=12)&&(PI in [300:12000])'
```

- filter the created event list by extracting all events located within a circle specified by a boolean expression.

```
evselect  --table=ev_filtered.ds \
--withfilteredset=true \
--filteredset=ev_source.ds \
--keepfilteroutput=true \
--destruct=true \
--expression='((X,Y) IN circle(24361,24665.5,1044))'
```

5.12.4 Timing analysis using the command lines

The following example illustrates how to produce a light curve using evselect. The user can work with command lines as follows.

- In order to avoid to manipulate long names and path, it is first convenient to define some variables including e.g the directory which contain the pipeline processed data, the data file of interest, or a recurrent prefix which can be used in the name of the output datafile.

```
set OBS='target_name'
```

- Filtering of the input datafile and creation of an image, spectrum and a time serie can be performed using the evselect command:

```
evselect --table=ev_source.ds \
--withimageset=true --imageset=${OBS}IMG.FIT \
--withspectrumset=true --spectrumset=${OBS}SPE.FIT \
--withrateset=true --rateset=${OBS}RAT.FIT \
--expression=true
```

- The command creates 3 files named respectively:

```
sourcenameIMG.FIT
sourcenameRAT.FIT
sourcenameSPE.FIT
```

- These files contain the filtered image, spectrum and time serie products. The spectrum and timeserie table can be plotted with the xmgr package using the dsplot command.

```
dsplot table=${OBS}RAT.FIT
dsplot table=${OBS}SPE.FIT$
```

- Postscript files can be created.

```
dsplot table=${OBS}SPE.FIT plotter='grbatch -eps -noask -printfile spectrum.ps'
dsplot table=${OBS}RAT.FIT plotter='grbatch -eps -noask -printfile rate.ps'$
```

- Spectrum and timeserie files can also be generated in ASCII format using the command dstoplot.

```
dstoplot table=${OBS}RAT.FIT > temp.dat
dstoplot table=${OBS}SPE.FIT > temp.dat
```

- Analysis of the timeserie can then be performed using the XRONOS data analysis package to compute power spectra, autocorrelation functions ..etc. Still the dsplot command can be conveniently used to plot the fits table.

```
lcurve 1 ${OBS}RAT.FIT window=- dtnb=100 nbint=10 \
outfile=lcurve.fits plot=no

dsplot table=lcurve.fits \
plotter='grbatch -eps -noask -printfile lcurve.ps'
```

```
powspec ${OBS}RAT.FIT window=- dtnb=INDEF nbint=10 nintfm=1 rebin=0 \
outfile=powspec.fits plot=no

dsplot table=powspec.fits y=POWER \
plotter='grbatch -eps -noask -printfile powspec.ps'

autocor ${OBS}RAT.FIT window=- dtnb=3 nbint=10 nintfm=20 rebin=0 \
outfile=autocor.fits plot=no outfiletype=1

dsplot table=autocor.fits \
plotter='grbatch -eps -noask -printfile autocor.ps'

lcstats ${OBS}RAT.FIT window=- dtnb=3 nbint=30 > lcstats.txt

emacs lcstats.txt

timeskew ${OBS}RAT.FIT window=- dtnb=3 nbint=30 nintfm=1 rebin=0 \
outfile=timeskew.fits plot=no outfiletype=1

dsplot table=timeskew.fits \
plotter='grbatch -eps -noask -printfile timeskew.ps'

gv timeskew.ps

efsearch ${OBS}RAT.FIT window=- sepoch=INDEF dper=12 nphase=INDEF \
nbint=INDEF dres=0.1 nper=100 outfile=efsearch.fits plot=no || exit 1

dsplot table=efsearch.fits \
plotter='grbatch -eps -noask -printfile efsearch.ps'
```

5.12.5 Spectrum extraction of a spatially filtered event list

The following example illustrates a simple analysis session which aims to extract the spectrum of a point source within the field of view of a single EPIC MOS camera. The user can work with command lines along the following steps.

1. specify the observation data file (ODF) directory of interest and its associated calibration index file (CIF).

   ```
   setenv SAS_ODF /'path to the selected odf'/
   setenv SAS_CCF ccf.cif
   ```

2. run e.g the EPIC MOS pipeline processing

   ```
   emproc &
   ```

3. define the event file of interest e.g the MOS 1 merged event list

   ```
   set evfile=0001_0000010030_M1S00100IME.FIT
   ```

4. filter the event list by extracting all events with FLAGs smaller than 1, PATTERN included between 1 and 12 and PI energy in the range 300 to 12000 eV.

```
evselect --table=$evfile  \
--withfilteredset=true \
--filteredset=ev_filtered.ds \
--keepfilteroutput=true \
--destruct=true \
--expression='(FLAG<=1)&&(PATTERN<=12)&&(PI in [300:12000])'
```

5. define some variables

```
set BINSIZE=100
set IMAGEFILE=image.fits
```

6. create an image

```
evselect \
 --table=ev_filtered.ds \
 --withfilteredset=true\
 --filteredset=filtered.ds\
 --withimageset=true \
 --xcolumn='X' \
 --ycolumn='Y' \
 --ximagebinsize=$BINSIZE \
 --yimagebinsize=$BINSIZE \
 --imageset=$IMAGEFILE \
 --withimagedatatype=true \
 --imagedatatype='Int32'
```

7. display the image and select a region around the source from which a spectrum shall be retrieved. The region specification can be saved to an ascii file.

```
imgdisplay --withimagefile=true --imagefile=image.fits &
```

8. extract the event within the region of interest

```
evselect \
 --table=ev_filtered.ds \
 --withfilteredset=true\
 --filteredset=ev_source.ds\
 --keepfilteroutput=true\
 --destruct=true\
 --expression='((RAWX,RAWY) IN circle(280.5,284,10.5))'
```

9. create a spectrum of the region of interest

```
set SPECFILE = "spectrum.pha"

evselect \
 --table=ev_source.ds \
 --withfilteredset=true\
 --filteredset=filtered.ds\
 --withspectrumset=true \
 --spectrumset=$SPECFILE
```

10. view the spectrum

```
dsplot table=$SPECFILE &
```

11. create response matrix files using the *rmfgen* and *arfgen* SAS tasks for further spectral analysis.

```
rmfgen spectrumset=$SPECFILE \
withenergybins=true energymin=0.1 energymax=12.0 nenergybins=1200 \
rmfset=response.rmf

arfgen spectrumset=$SPECFILE detmaptype=psf \
withsourcepos=true sourcecoords=det sourcex=0 sourcey=0 \
withrmfset=true rmfset=response.rmf arfset=ancillary.arf
```

5.12.6 Extraction of a spectrum using xmmselect

Instead of typing commands, the user can work the with the graphical user interface (GUI). For that purpose, the *sas* command shall be run first. The SAS GUI appears and the user can select a task. The *evselect* and *xmmselect* tasks are designed to extract images, spectra or time series. For example, after calling *evselect*, the user could proceed as follows.

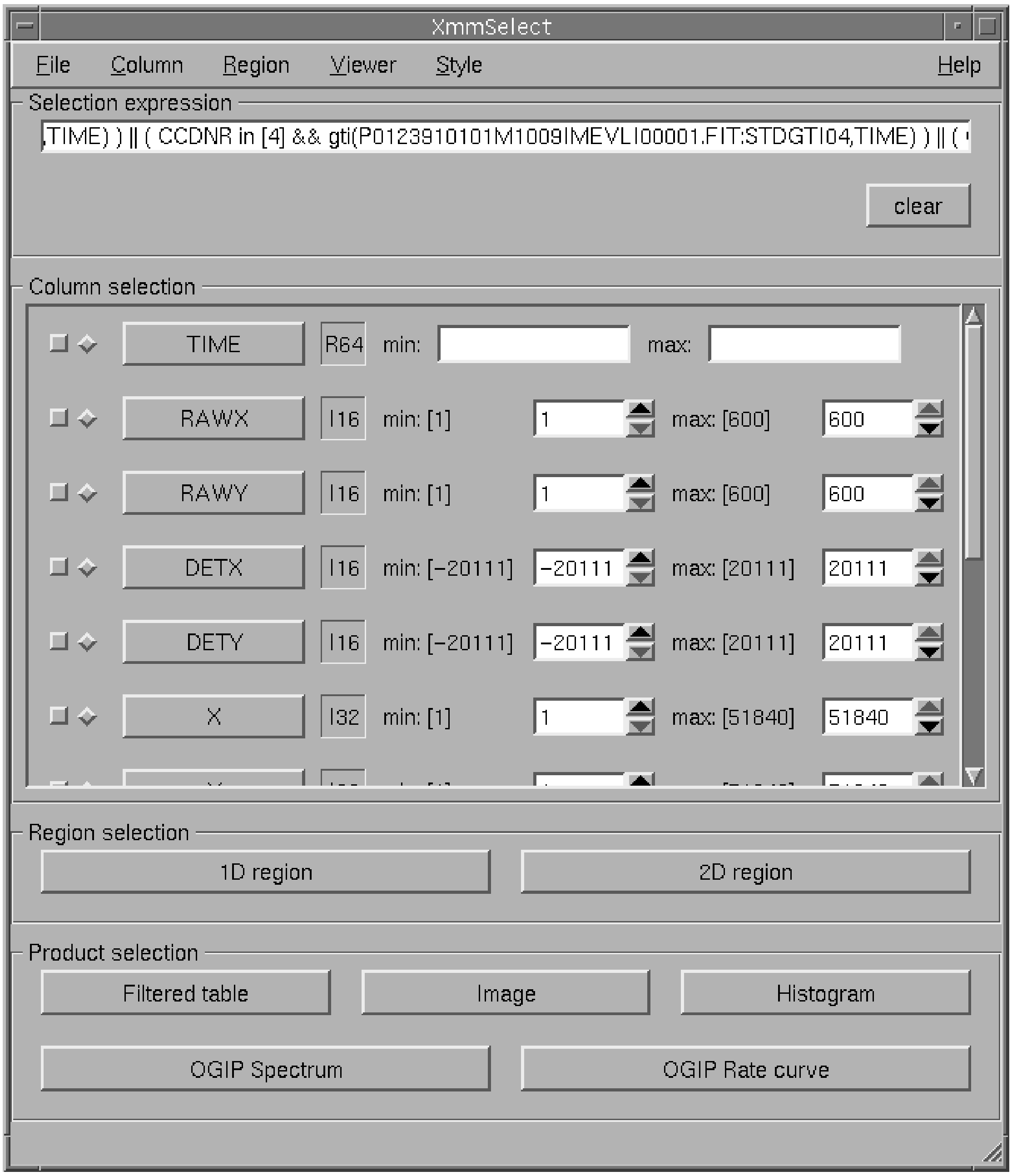
- In the window called "general", the user shall select the event list of the instrument file of interest and type run. This generates a filtered event list which by default is called filtered.fits.
- To generate an image, the user shall recall the *evselect* task, fill-in the image window and press run. This generates an image array which by default is called image.fits.
- To generate a spectrum, the user shall recall the *evselect* task, fill-in the spectrum window and press run. This generates a spectrum table which by default is called spectrum.fits.
- To generate a time serie, the user shall recall the *evselect* task, fill-in the lightcurve window and press run. This generates a timeserie table which by default is called rate.fits.

The spectrum and time serie can then be visualised by pressing the task button of the sas gui and selecting the *dsplot* task. Then the events table of the light curve or spectrum fits file shall be selected together with the xmgr plotter before pressing run. Analysis of the light curve or spectrum fits files can be performed using the xronos datapackage.

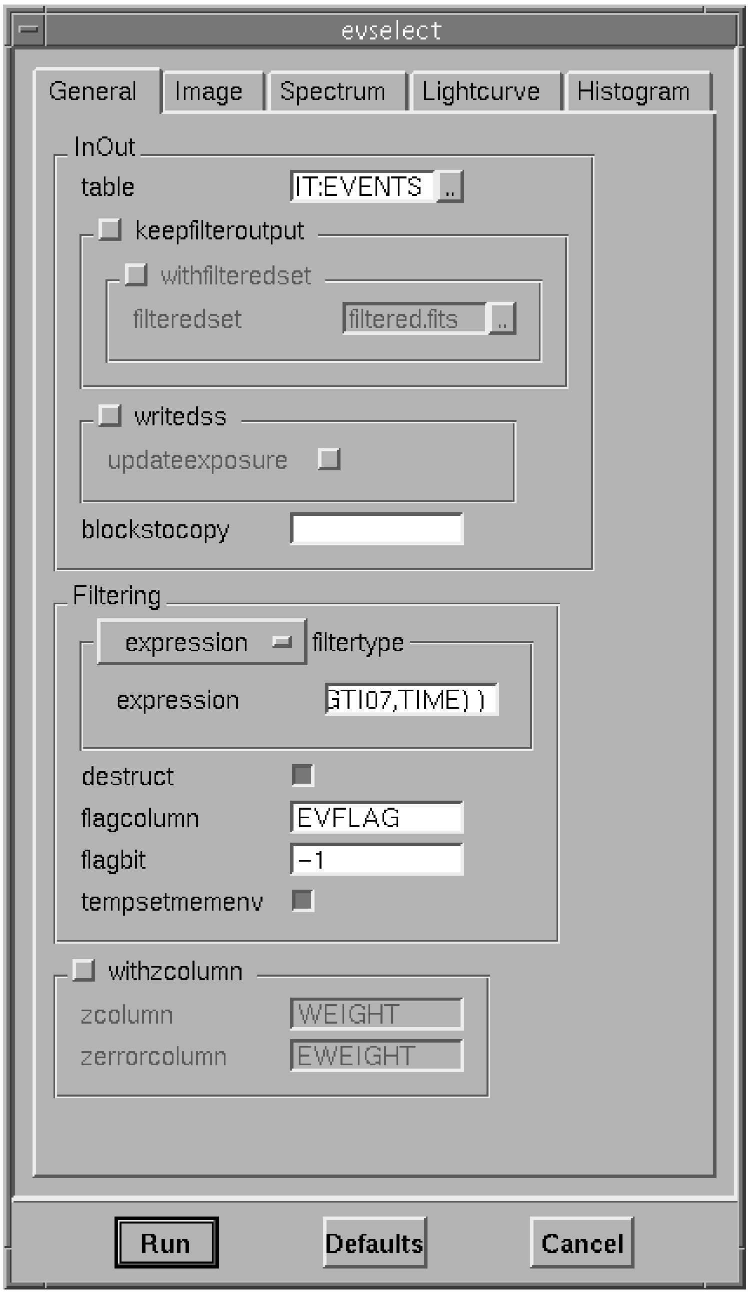
To extract images, spectra or time series, the user can alternatively run the task *xmmselect* from the GUI. In the following example, a spectrum of a region of a supernova remnant is extracted using the *xmmselect* SAS task.

- `xmmselect --table=ev_filtered.ds &`
- select X/Y as image axis and push the button "Image"
- select the "Image Tab" in the parameter dialog and set the imgae size to 600
- push "Run" and wait for the image to pop up in ds9
- center the image with: mode -> "Pan" and drag the viewport to the right until the image is in the center (or move the small box in the panner window into the center)

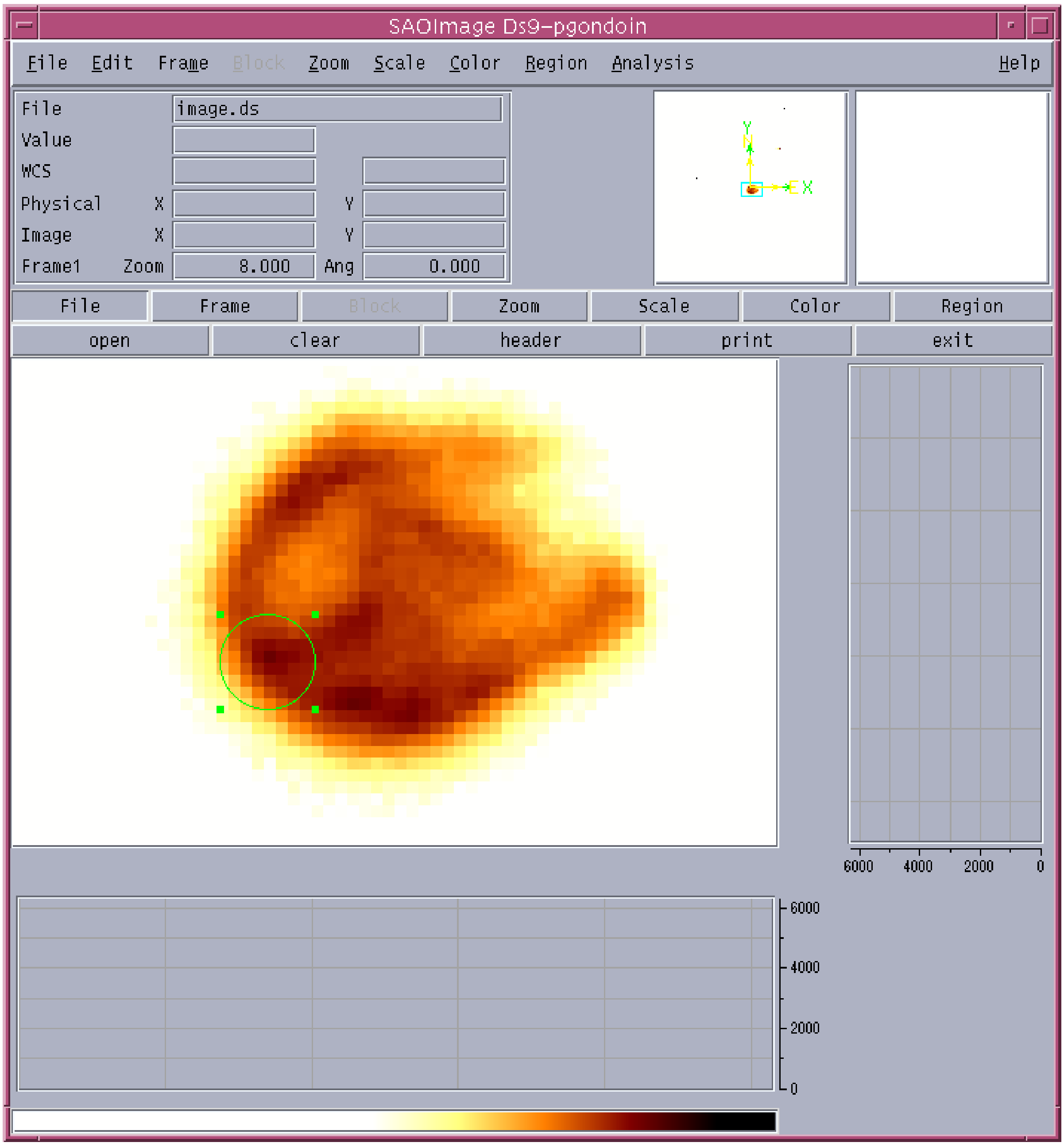
- select color table "bb" and mode "colorbar" and adjust brightness/color
- zoom in and mark the star remnants with a small circle (region -> circle; click on center and drag to the desired radius) or box
- click on "2D region" button in xmmselect
- select "pi" as energy axis for the spectrum extraction and push "OGIP Spectrum". Select the spectrum tab and press and "run"
- the spectrum is shown in xmgr



In the xmmselect main window, an image can be extracted by selecting X/Y as image axis and by pushing "Image".



After selecting the "Image Tab" in the parameter dialog of the main xmmselect window, a small window appears which allows to set the image bin size to the desired value.



When the image appears in ds9, you can center the image, select a color table, adjust the brightness and the contrast, zoom in and mark a region with e.g an ellipse. After clicking on the "2D region" button in the xmmselect main window, selecting an energy axis for the spectrum extraction and pushing "Spectrum", a spectrum of the selected region will appear in an xmgr window.

Chapter 6

Analysis of RGS spectrometer data

6.1 Inspecting the RGS data package

After reception of the CD-ROMs containing the XMM observation data package, the guest investigator willing to analyse RGS data is advised to verify the RGS related content of the data package, to view the RGS pipeline products and subsequently to establish its plan for analysing RGS data. The RGS data package includes the ODF files, the CCF files and the RGS pipeline products.

6.1.1 RGS Observation Data Files

For each of the two RGS spectrometers, The RGS related ODF contain the following files.

- one RGS auxiliary file per exposure which provide description details of each frame recording during the exposure,
- one diagnostic image file per RGS CCD,
- one event list file per RGS CCD and per exposure,
- one periodic housekeeping file,
- one non-periodic housekeeping file.

Each of the 9 RGS event list files consist of a primary header and a binary table called EVENTS. The RGS EVENTS tables generated in spectroscopy mode have six columns called FRAME, RAWX, RAWY, ENERGY, CCDNODE, SER. These provide the frame identifier, the event position in detector pixel coordinate, the energy channel and the CCD node number.

The structure and content of the observation data files can easily be inspected with the ftools package. This can be performed either with the ftools gui viewer (by typing e.g fv “file name”) or with the ftools command lines (such as fstruct, flcol, fstatistic, fplot, fhisto ..etc). The user is refered to the ftools manual or fhelp command for further information.

File name	Description of the file content
0001_0000000000_R1S00100AUX.FIT	RGS1 auxiliary file
0001_0000000000_R1S00101SPE.FIT	RGS1 CCD1 event list
0001_0000000000_R1S00102SPE.FIT	RGS1 CCD2 event list
0001_0000000000_R1S00103SPE.FIT	RGS1 CCD3 event list
0001_0000000000_R1S00104SPE.FIT	RGS1 CCD4 event list
0001_0000000000_R1S00105SPE.FIT	RGS1 CCD5 event list
0001_0000000000_R1S00106SPE.FIT	RGS1 CCD6 event list
0001_0000000000_R1S00107SPE.FIT	RGS1 CCD7 event list
0001_0000000000_R1S00108SPE.FIT	RGS1 CCD8 event list
0001_0000000000_R1S00109SPE.FIT	RGS1 CCD9 event list
0001_0000000000_R1X00100PEH.FIT	RGS1 periodic HK file
0001_0000000000_R2S00100AUX.FIT	RGS2 auxiliary file
0001_0000000000_R2S00101SPE.FIT	RGS2 CCD1 event list
0001_0000000000_R2S00102SPE.FIT	RGS2 CCD2 event list
0001_0000000000_R2S00103SPE.FIT	RGS2 CCD3 event list
0001_0000000000_R2S00104SPE.FIT	RGS2 CCD4 event list
0001_0000000000_R2S00105SPE.FIT	RGS2 CCD5 event list
0001_0000000000_R2S00106SPE.FIT	RGS2 CCD6 event list
0001_0000000000_R2S00107SPE.FIT	RGS2 CCD7 event list
0001_0000000000_R2S00108SPE.FIT	RGS2 CCD8 event list
0001_0000000000_R2S00109SPE.FIT	RGS2 CCD9 event list
0001_0000000000_R2X00100PEH.FIT	RGS2 periodic HK file

Table 6.1: ODF files associated with a single RGS exposure in spectroscopy mode

6.1.2 RGS Current Calibration Files

The XMM investigator receives both calibrated events files and data in a raw unprocessed form. In order to run the calibration processing her/himself and to correct raw data from the instrument response, s/he also receives the current calibration files which describe the physical properties of the RGS instrument at the time of the observation. The information contained in the current calibration files is accessed by most of the task through the calibration access layer. The *rgsrmfgen* task generates response matrices which describe the instrument responses. Spectral fitting techniques fold the instrument response matrices with simulated spectra for comparison with observed data.

The following table provides a list of RGS calibration files. For a detailed description of these files and of their usage, the user is refered to the CCF interface document and to the Calibration Access and Data Handbook.

File name	File Content
ADUConv	Coefficients tables and model to convert PHA to energy.
Background	Background model and spectra.
BadPix	Table of bad (hot/dead/flickering/pin holes) CCD pixels.
CalSrcData	energies, reference pulse heights, CCD images of calibration sources .
CCDrmf	Large version of the CCD energy redistribution.
Contam	Description of the loss of quantum efficiency due to contaminants.
CrossPSF	Parameterized point spread function in the cross dispersion direction.
CTI	Coefficients to correct for the CCD charge transfer inefficiency.
DarkCurrent	Dark current map.
DarkFrame	Dark frame for the baseline frame time, with associated statistical information.
Efficiency	Detection efficiency versus energy, to be used by the quick look analysis.
EvtSizeDist	Distribution of CCD event types (single, 2-pixel, 3-pixel ...).
FilTransV	Optical properties of the CCD filters.
HKParmInt	House keeping parameter ranges used to generate good time intervals.
LinCoord	Data required to transform from pixel to physical coordinates on the CCD.
LineSpreadFunc	Parameterised line spread function.
ModeParam	Observation mode paramaters.
QuantumEf	CCD quantum efficiency.
Redist	CCD redistribution matrix.
XAreaEf	Effective area (mirror and gratings box).

Table 6.2: RGS calibration files (one set per spectrometer)

Access to RGS calibration data is provided by the task *calview*. RGS1 or RGS2 shall be selected as instrument in the main *calview* task window. The button 'view' of the task provide a list of calibration information which can be viewed.

6.1.3 RGS Pipeline Products

The RGS pipeline products include OGIP-compliant FITS files with binary table extensions unless stated. All products are specific to one single observation with one individual RGS instrument. The XMM investigator is advised to look carefully at this files in order to prepare his/her plan for further analysis of the RGS data. The RGS pipeline products include:

1. Timeseries plot of housekeeping parameters. Graphics file (PDF) contain timeseries plots of multiple housekeeping parameter values. The plots include spacecraft and RGS HK data, attitude data, and good time intervals. One product is provided per RGS observation.

2. Auxiliary data plot. Graphics file (PDF) contain timeseries plots of specified RGS auxiliary data. One product is provided per RGS spectroscopy or HTR exposure.

3. An event list which provides for each event the raw and aspect drift corrected dispersion angle, the raw and aspect drift corrected croos-dispersion angle, the PI, the arrival time, the grade and SER, the PHA, event attributes (e.g flagging event near a bad pixel region). Dispersion and cross-dispersion channels assuming on-axis source are also provided for further XSPEC analysis.

4. A Spatial FITS image of the cross-dispersion channel vs the beta dispersion channel giving raw counts per pixel (ie without exposure correction) over the total RGS CCD energy bandpass. One product is provided per spectroscopy mode exposure but none for the HTR mode exposures.

5. A FITS energy image of the PI vs beta channel (the so called "banana plot") giving raw counts per pixel over the entire cross-dispersion channel range. One product is produced per spectroscopy or HTR mode exposure.

6. A spatial image plot which consists of a graphics file (PNG) of the spatial FITS image showing the cross-dispersion channel vs the beta channel. It includes lines showing the extraction regions of extracted spectra. The lines are colour-coded to distinguish different sources. One product is produced per spectroscopy mode exposure, none for the HTR mode exposures.

7. An energy image plot which consists of a graphics file (PNG) of the energy FITS image of the PI vs beta channel giving raw counts per pixel over the entire cross-dispersion channel range. It includes lines showing extraction regions for extracted spectra. The lines are colour-coded to distinguish different sources. One product is provided per spectroscopy or HTR mode exposure.

8. An exposure map which consists of a FITS image of the cross-dispersion channel vs the beta channel giving the fractional live time per pixel. Image values may be less than unity when coordinates do not lie on the active detector area due to attitude or pixel affected by cosmic rays. One product is provided per spectroscopy mode exposure. An equivalent product (TBD) is produced for the HTR mode exposures.

9. The source spectrum. The extraction regions are derived from the EPIC source position with the following priorities. MOS in image mode is the first priority, then EPIC and finally the target position from the observation summary file. The source spectrum is created by evselect with the spectral channel being the beta channel. No detailed exposure information is included. Both orders -1 and -2 are in a single file (TBC). One product is created per EPIC source brighter than 5% of the target brightness in the RGS band providing that it lies within 10 arcmin in dispersion direction and within 1.5 arcmin in cross-dispersion direction from the target direction. One set of products is provided per spectroscopy or HTR mode exposure with all numerical parameters being specified at run-time.

10. A spectrum plot which consists of a graphics file (PDF) showing plot of the source spectrum (i.e counts vs beta-channel) not exposure corrected. Also overlaid are source-specific axes giving wavelength and energy. Theoretical zero-velocity locations of some bright coronal emission lines are also marked. Also plotted are the exposure spectrum cut from the exposure map and the spectrum of the on-board calibration sources as it would appear in the source extraction region. Product includes the -1 and -2 order. One product is provided per source spectrum.

11. A trailed spectrum which consists of a graphics file (PNG) of an image plot of the -1 order spectrum vs time. Pixels give raw counts without exposure correction. One product is provided per bright source and per spectroscopy or HTR mode exposure. Hence the number of trailed spectra is less than or equal to the number of source spectra.

12. A summary HTML file listing the major observation and RGS-specific parameters and providing links to other RGS products. One product is provided per observation.

6.2 Running the RGS pipeline processing

6.2.1 Introduction

The datapackage received by the investigator contains RGS calibrated event lists which normally do not neccessitate further processing for the purpose of calibration. However, a script of the pipeline processing batch will be made available to the investigator who can rerun the pipeline calibration in an interactive way. A third possibility for the investigator is to run the default RGS pipeline processing chain called *rgschain*.

The RGS pipeline processing produces calibrated event files (see Table 6.2.1 from the raw RGS observation datafiles. It is run by specifying the path to the odf directory, building an associalted calibration index file and running the *rgsproc* command.

- point to the observation data file (ODF) containing the raw RGS files of interest.

  ```
  setenv SAS_ODF /'path to the selected odf'/
  ```

- generate a calibration index file (CIF) from a repository of current calibration files (CCFs)

  ```
  cifbuild analysisdate='now' append=no calindexset='ccf.cif' \\
  category='XMMCCF' ccfpath=/'path to a ccf repository'/ \\
  fileglob='*.ccf|*.CCF' fullpath=yes ignorecategory=no \\
  masterindex=no masterindexset='ccf.mif' observationdate='2000-08-10' \\
  recurse=no withccfpath=yes withmasterindexset=no \\
          withobservationdate=no
  ```

- point to the newly created ccf.cif calibration index file

  ```
  setenv SAS_CCF ccf.cif
  ```

- run e.g RGS pipeline processing

  ```
  rgsproc &
  ```

- After some time, a set of calibrated event files will have been created into your working directory including the merged event lists of the 9 CCDs of each of the RGS1 and RGS2 camera. These files are easily recognizable by their suffixe _merged.ds. Also, they are usually the largest files of the directory.

An overview of the RGS pipeline processing tasks is given in the following sections. It summarizes the data flow through the various steps of the processing. For more detailled informations, the user shall consult the individual task descriptions provided in the SAS manual accessible on the web (see introduction) and through the SAS help command. The block diagram of data flow is shown in Figure 6.1. The function of each task is summarized in the following sections, in the order of its execution in rgsproc. The description is grouped according to global functionality.

table PIXELS — *rgsframes*

FRAME	int32	frame number
RAWxy	int16	pixel coordinates as CAL uses them (node oriented)
ENERGY	int16	energy value as in telemetry
CCDNODE	int8	node (0/1) as in telemetry
SHAPE	int8	event shape (SHAPE $<$ 8 is related to telemetry SER)
GRADE	int8	number of pixels combined during reconstruction
EVENT	int32	identifier (added by *rgsevents*

table FRAMES — *rgsframes*

FRAME	int32	frame number
SEQINDEX	int8	sequence number (1-12)
TIME	real64	timestamp of event
FRAMETIME	real32	frame integration time
REJPIX0	int32	dead time fraction node 0
REJPIX1	int32	dead time fraction node 1
FRAMEFLAG	int32	various flags about the frame
EVENTMIN	int32	first event of frame in EVENTS table (added by *rgsevents*
EVENTMAX	int32	idem last row nr
ASPCDSP	real32	Source independent aspect correction parameter in dispersion direction. (produced by rgsangles used by rgsexposure).
ASPCXDSP	real32	same as ASPCDSP but in cross-dispersion direction.

table EVENTS — *rgsevents*

FRAME	int32	frame number
CHIPX	int16	chip oriented CCD column number of event
CHIPY	int16$^{(1)}$	CCD row number of event
FLAG	int32	various flags related to the event.
SHAPE	int8	event shape (SHAPE < 8 is related to telemetry SER)
GRADE	int8	number of pixels combined during reconstruction
TIME	real64	timestamp of frame (copy from frame table)
PHA	int32	sum (over all pixels) of telemetered ENERGY (in units of PI
BETA	float32	beta from hardware geometry only, not aspect drift corrected;
XDSP	float32$^{(1)}$	cross-dispersion angle from hardware geometry only, not aspect drift corrected
BETA_CORR	float32	aspect drift corrected beta based on on-axis target; by *rgsangles*
XDSP_CORR	float32$^{(1)}$	aspect drift corrected cross-dispersion angle; by *rgsangles*
BETA_CHANNEL	int16	channel number of binned BETA_CORR; by *rgsangles* (TBD)
XDSP_CHANNEL	int16$^{(1)}$	channel number of binned CROSSDSP_CORR; by *rgsangles* (TBD)
PI	int32	gain corrected reconstructed PHA
SC_POINTING_RA	float32	(optional) RA of spacecraft pointing. written by *rgsangles*
SC_POINTING_DEC	float32	(optional) DEC of spacecraft pointing. written by *rgsangles*
SC_POINTING_POS	float32	(optional) position angle of spacecraft pointing. written by *rgsangles*
CCDID	int8	ccdid (1-9). Produced by *rgsangles* to allow easy combination of eventlists.
M_LAMBDA	float32	(optional) product of spectral order and wavelength. Units are nm. Produced by *rgsangles* from geometric solution and source position.

table BADPIX — *badpixfind*

CHIPxy	int16	node-independent pixel coordinates of column bottom
YEXTENT	int16	length of bad column
TYPE	int16	bad pixel type code
BADFLAG	int16	bad pixel source id

6.2.2 Pipeline Pre-Cursor Tasks

These tasks are unrelated to the event list, and prepare the required necessary input.

6.2.2.1 rgsoffsetcalc

It is expected that the offset of the CCD's PHA calibration will depend on the actual light background, which is a function of the aspect of the telescope. This tasks utilizes the simultaneously transmitted diagnostic modes data and determines the offset per data set. These are written together with time stamps into a table of offsets. The CAL has to be instructed by the respective task (rgsevents) to use these offset tables. The time tags allow for potential interpolations of drifts of the offsets during an exposure.

6.2.2.2 rgssources

This task prepares the SOURCES table, which holds the list of sources for which the final pipeline products should be extracted. Since the location of the dispersed image on the RFC and the wavelength scale are dependent on the relative position of the source in the FOV, the extraction filters have to be calculated per source. The SOURCES table is used as input to the definition of the extraction regions.

rgssources filters the list of sources detected by EPIC, and adds those sources that are potentially in the FOV of RGS and also fulfill a certain flux criterion. Alternatively the user can also provide the source positions via input arguments, which will be added to the SOURCES table.

The sources data set also contains the blocks with the region definitions per source. Both the SOURCES table and the region filters are indexed and therefore the SOURCES table can be used as an index to the region blocks.

6.2.3 Basic Event Processing

The processing steps described in this section are related to event recognition on the RFC (CCD related processing), and thus are independent of the source position. They will be combined in the meta-task rgsevproc.

These tasks create all tables that are called *intermediate event list*. The layout of these tables is listed in the following. Next to the name of each block, the names of the task that crates the respective table is given. Columns that are added later by other tasks show the name of that task next to the respective column.

6.2.3.1 rgsframes

This is the only interface to the ODF event lists. The event list table are copied and re-formatted into the standard format by the SAS. As a result the PIXELS table is produced.

Basic telemetry consistency checks are being executed and stored in GTI intervals.

The FRAMES table is constructed from the global frame information of the AUX file. Frame times (photon integration times per CCD) are calculated by using the CCD readout sequence. This will be used for the exposure calculations.

6.2.3.2 rgsbadpix

This is a combination of similar EPIC tasks *ebadpixfind* and *badpix*. The BADPIX table is constructed in two steps: First the pixel data are scanned for bad pixels, and the secondly these newly found bad pixels are combined with those stored in the CCF. The BADPIX table is attached to the intermediate events set.

6.2.3.3 rgsevents

This combines calibrations of the energy scale, CTI corrections and event reconstruction. The recognized events are stored in the newly generated EVENTS table. Cross reference indices between the EVENTS table and the FRAMES table are setup such that the corresponding readout frames can easily be found. The PI column contains the calibrated energies of the events.

The dispersion (column BETA) and cross-dispersion (column XDSP) angles as derived from the relative locations of the RGA and of the RFC are calculated and added to the EVENTS table. These angles are subject to corrections, they purely can be calculated from geometry. Note however that the underlying assumption is that the relative position and orientation of the RGA to RFC are stable (no bending of the telescope tube). Should flight operations prove that this needs to be included, then this has to be reviewed.

6.2.4 Source Dependent Event Processing

The following tasks contain the source dependent corrections of the aspect drift, calculation of dispersion angles and finally construction of the response. Note that rgsrmfgen is not part of the pipeline processing. It is listed here purely for completeness only to give the interactive user an overview of the required processing.

6.2.4.1 rgsangles

The main function of this task is to apply the corrections of the aspect drift to the data. From the dispersion equation

$$\frac{m\lambda}{d} = \cos\beta - \cos\alpha$$

with the following definitions:

m	being the number of the inside order ($m < 0$),
λ	the wavelength
d	the line density of the grating
α	the angle of incidence on the grating measured from the surface
β	the diffracted angle measured from the surface

it can be seen that the diffracted angle β is a function of variations of the angle of incidence α for a given wavelength and order. The drift of the telescope's attitude modifies α, and thus indirectly β, from which finally the wavelength scale is calculated. The dispersion equation can be used to relate the aspect drift corrections to corrections of the diffraction angle β. Aspect drifts are corrected as corrections to β, resulting in the column BETA_CORR (see rgsangles for more detail).

The correction $\Delta\beta$ due to an attitude drift $\Delta\alpha$ is non-linear in α. Therefore the amount of $\Delta\beta$ depends on the position in the FOV along the dispersion direction. In the pipeline the attitude is corrected for the first SOURCE of the SOURCES table (mostly the prime target, and therefore expected to be close to on-axis), but could be repeated by the interactive user for any position that is listed in the SOURCES table. Inaccuracies of the wavelength scale for source positions other than the one chosen due to the choice of attitude correction, can be estimated with keys that are added to attributes of the EVENTS table.

Cross-dispersion aspect drift correction is more straightforward and is stored in the columns XDSP_CORR.

6.2.4.2 rgsexposure

Exposure maps are created per CCD node. The dimension of these exposure maps diverts from the commonly used definitions of exposure maps, in that they are 1/time and do not include the effective area. This is due to the fact that the effective area depends on the source off-axis angle and on the wavelength, both are not identical on the RFC surface for all sources. Therefore the effective area is included in the response generation (rgsrmfgen).

The exposure maps are constructed in BETA_CORR versus XDSP_CORR space, and thus include aspect drift correction.

6.2.4.3 evlistcomb

This task is common with the EPIC processing and combines the EVENTS tables per RFC of all RGS CCD's into one table. Additional extensions are copied into the same data file.

6.2.4.4 rgsexpcombine

This task combines the exposure maps generated by rgsexposure into one map. The main purpose of this combined map is to give the user an overview over her/his exposure. The individual maps are however used in further analysis (rgsrmfgen). This combined map is also part of the pipeline products.

6.2.4.5 rgsregion

Regions to filter events are calculated for a given source position (passed by index number of the SOURCES table). The source specific filtering is performed per source in two domains: the imaging domain (BETA_CORR versus XDSP_CORR) and the order separation domain (BETA_CORR versus PI).

This task does not actively perform the filtering, it only stores the regions as part of the same file that contains the SOURCES table. The names of the data set contain the index of source in the SOURCES table for which the regions were calculated.

6.2.4.6 rgsrmfgen

The task generates the response matrix specifically to a source listed in the SOURCES table. It uses the region specifications to filter the effective area and the exposure maps to obtain the accepted response of the instrument. This response matrix should be used together with the region filtered event list (which is source specific too!).

This task is not part of the pipeline processing.

6.2.5 Pipeline Product Generation

rgsspectrum is used to filter the event list with the regions created by *rgsregion*. Net and background spectra are provided as output file of *rgsproc*. The filtered event list is stored, and also provided as a pipeline product. It has the following columns:

table EVENTS — *evlistcomb*

CCDNR	byte	this is CCDID
BETA_CORR	float32	aspect drift corrected beta based on on-axis target
XDSP_CORR	float32$^{(1)}$	aspect drift corrected cross-dispersion angle
TIME	float64	time stamp of event
FLAG	int32	copy of EVENT_ATTRIBUTES; extra flag for reconstruction across nodes is required
BETA	float32	beta from hardware geometry only, not aspect drift corrected
XDSP	float32$^{(1)}$	cross-dispersion angle from hardware geometry only, not aspect drift corrected
CHIPX	int16	chip oriented CCD column number
CHIPY	int16$^{(1)}$	CCD row number
PHA	int16	sum (over all pixels) of telemetered ENERGY (in units of PI
SHAPE	byte$^{(1)}$	shape of event; early version of the on-board DPP will provide the number of combined pixels in SER. This will be mapped in SHAPE in the following way: $0 \rightarrow 0$, $3 \rightarrow 1$, 1 and $2 \rightarrow 0$; later versions of the DPP code will contain the event SHAPE encoded in 4 bits. SHAPES > 7 will be used by rgsevents;
GRADE	byte$^{(1)}$	number of pixels combined into event
BETA_CHANNEL	int16	channel number of binned BETA_CORR
XDSP_CHANNEL	int16$^{(1)}$	channel number of binned CROSSDSP_CORR
PI	int16	gain corrected PHA

Notes: $^{(1)}$ this is not applicable to HTR data.

File name	Description of the file content
0001_0000000000_R1S00400_expmap.ds	RGS 1 exposure map
0001_0000000000_R1S00400_filtered.ds	RGS 1 filtered event list
0001_0000000000_R1S00400_merged.ds	concatenated RGS 1 event list
0001_0000000000_R1S00400_rmf1.ds	RGS 1 response matrix file
0001_0000000000_R1S00400_sources.ds	RGS 1 source list
0001_0000000000_R1S00401_events.ds	CCD 1 event list file
0001_0000000000_R1S00402_events.ds	CCD 2 event list file
0001_0000000000_R1S00403_events.ds	CCD 3 event list file
0001_0000000000_R1S00404_events.ds	CCD 4 event list file
0001_0000000000_R1S00405_events.ds	CCD 5 event list file
0001_0000000000_R1S00406_events.ds	CCD 6 event list file
0001_0000000000_R1S00407_events.ds	CCD 7 event list file
0001_0000000000_R1S00408_events.ds	CCD 8 event list file
0001_0000000000_R1S00409_events.ds	CCD 9 event list file
0001_0000000000_R1S00400_net3o1.pha	RGS 1 first order spectrum
0001_0000000000_R1S00400_bkg3o1.ps	
0001_0000000000_R1S00400_histo.ds	
0001_0000000000_R1S00400_histo.ps	
0001_0000000000_R1S00400_images.ps	
0001_0000000000_R1S00400_net3o1.ps	
0001_0000000000_R1S00400_rates.ds	
0001_0000000000_R1S00400_rates.ps	
0001_0000000000_R2S00500_expmap.ds	RGS 2 exposure map
0001_0000000000_R2S00500_filtered.ds	RGS 2 filtered event list
0001_0000000000_R2S00500_merged.ds	concatenated RGS 2 event list
0001_0000000000_R2S00500_rmf1.ds	RGS 2 response matrix file
0001_0000000000_R2S00500_sources.ds	RGS 2 source list
0001_0000000000_R2S00501_events.ds	CCD 1 event list file
0001_0000000000_R2S00502_events.ds	CCD 2 event list file
0001_0000000000_R2S00503_events.ds	CCD 3 event list file
0001_0000000000_R2S00505_events.ds	CCD 5 event list file
0001_0000000000_R2S00506_events.ds	CCD 5 event list file
0001_0000000000_R2S00507_events.ds	CCD 7 event list file
0001_0000000000_R2S00508_events.ds	CCD 8 event list file
0001_0000000000_R2S00509_events.ds	CCD 9 event list file
0001_0000000000_R2S00500_net3o1.pha	RGS 2 first order spectrum
0001_0000000000_R2S00500_bkg3o1.ps	
0001_0000000000_R2S00500_histo.ds	
0001_0000000000_R2S00500_histo.ps	
0001_0000000000_R2S00500_images.ps	
0001_0000000000_R2S00500_net3o1.ps	
0001_0000000000_R2S00500_rates.ds	
0001_0000000000_R2S00500_rates.ps	

Table 6.3: Output rgsproc files for RGS 1 and RGS 2

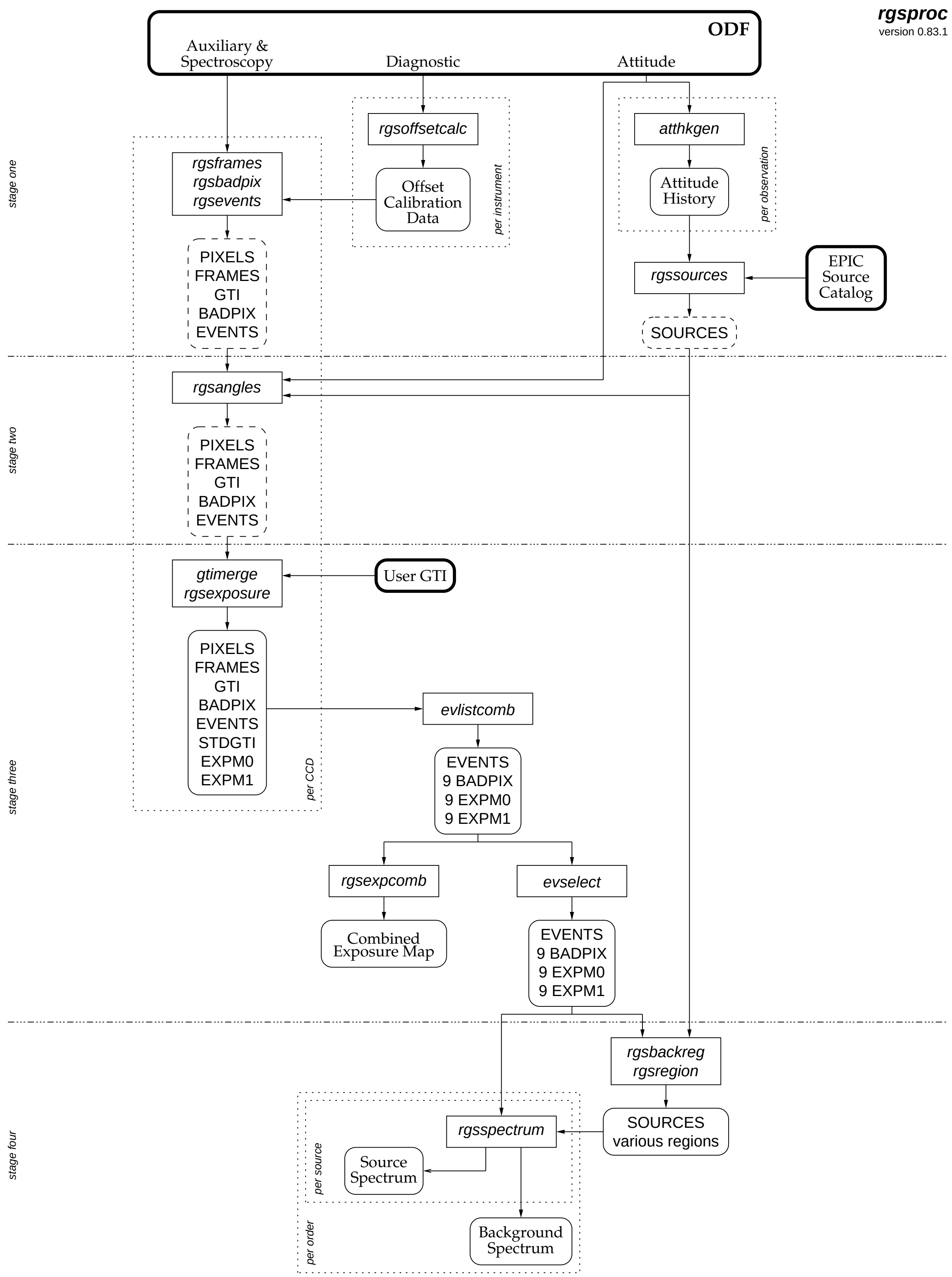


Figure 6.1: Diagram of *rgsproc* sub-tasks and data-flow. This sequence is repeated for each exposure. Sub-tasks appear within solid rectangles, process loops appear within dotted rectangles, and data files appear within rounded boxes with thick borders for input files and with dashed borders for incomplete files.

6.3 Inspecting extracted RGS spectra

The pipeline processing of RGS generate data products including spectra and background files. These can directly be used for further specral fitting after creation of the associated response matrice files. However, the user is advised to inspect carefully the pipeline products.

The task *rgsimgplot* is well suited to inspect the ouput of the pipeline processing and the adequacy of the spectrum extraction region. Before running the *rgsimgplot*, the user shall create (i) a spatial image of the spectrum (i.e an event distribution vs dispersion (BETA) and cross-dispersion (XDISP) direction) and (ii) an energy image of the spectrum (the so-called banana plot which is the distribution of events vs dispersion (BETA) and energy channel (PI)). These images can be created by e.g running the *xmmselect* on the RGS merged event list.

- launch the *xmmselect* task:

  ```
  xmmselect --table=001_0000000001_R1S00400_filtered.ds &
  ```

- In the xmmselect main window (see Fig. 6.2), an image of the spectrum dispersed by the Reflection Grating Assembly can be created by selecting BETA and XDISP as image axis and by pushing the button "Image".The evselect window appears (see Fig. 6.3). The user shall push the button "image", edit the name of the imageset to e.g RGS1image.ds and then press the button "run". A new window appears which displays the dispersed spectrum registered by the RFC camera (see Fig. 6.4) .
- In the xmmselect main window, an image of the two different spectral orders dispersed by the Reflection Grating Assembly (the so-called banana plot) can be created by selecting BETA and PI as image axis and by pushing the button "Image". The evselect window appears (see Fig. 6.3). The user shall push the button "image", edit the name of the imageset to e.g RGS1banana.ds and then press the button "run". The "banana plot" appears in a separate window (see Fig. 6.5).
- The user can now overlay the spectrum extraction region on both the spectrum and energy image by using the command *rgsimgplot*.

  ```
  rgsimplot device=/VCPS srclistset='001_0000000001_R1S00400_sources.ds' \
   endispset='RGS1banana.ds' spatialset='RGS1image.ds' --plotfile=overlay.ps \
  --withspatialregionsets=yes --withendispregionsets=yes

  gv overlay.ps
  ```

By overlaying the extraction region on the displayed spectrum of the source using the *rgsimgplot* task, the user can verify the adequacy of the spectrum extraction region.

The task *rgsimgplot* displays an RGS Spectroscopy spatial-dispersion plot (displaying photon position in beta space) and an energy-dispersion plot (displaying photon energy against beta angle in the dispersion direction) created from an RGS event list by *evselect* . These plots provide extremely useful diagnostics of RGS spectroscopy data, allowing the user to gauge the potential cross-contamination of the RGS orders by pileup, scattering and partial events. Details of the observation are also displayed in each plot. These include the observation and exposure IDs and the observation start and stop times. The plots may be displayed in monotone or using a number of colour tables (e.g grey-scale, rainbow, heat), and any valid PGPLOT graphics device may be specified for the output. A linear, sqrt or logarithmic colour mapping can be specified by the user. RGSREGION extraction region files are overlaid on the energy-dispersion and spatial-dispersion plots. Examples of these are shown in the following figures in which a spatial extraction region and order extraction regions are overlaid on the full images. The rgsimplot task produces an energy-dispersion plot in a format defined by the device parameter. The user can also visualize the extracted spectrum using the task *dsplot*.

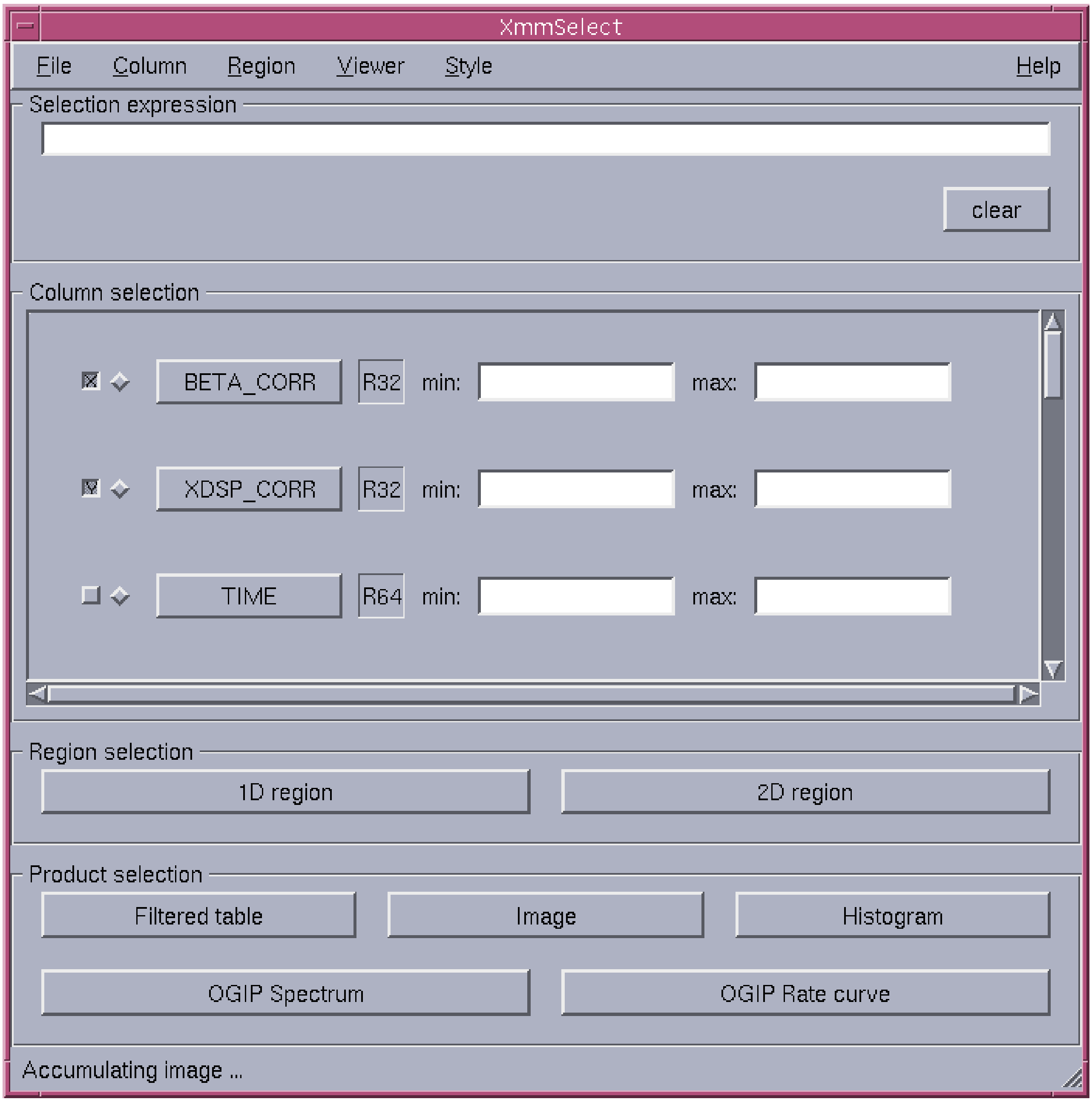


Figure 6.2: XMMselect window

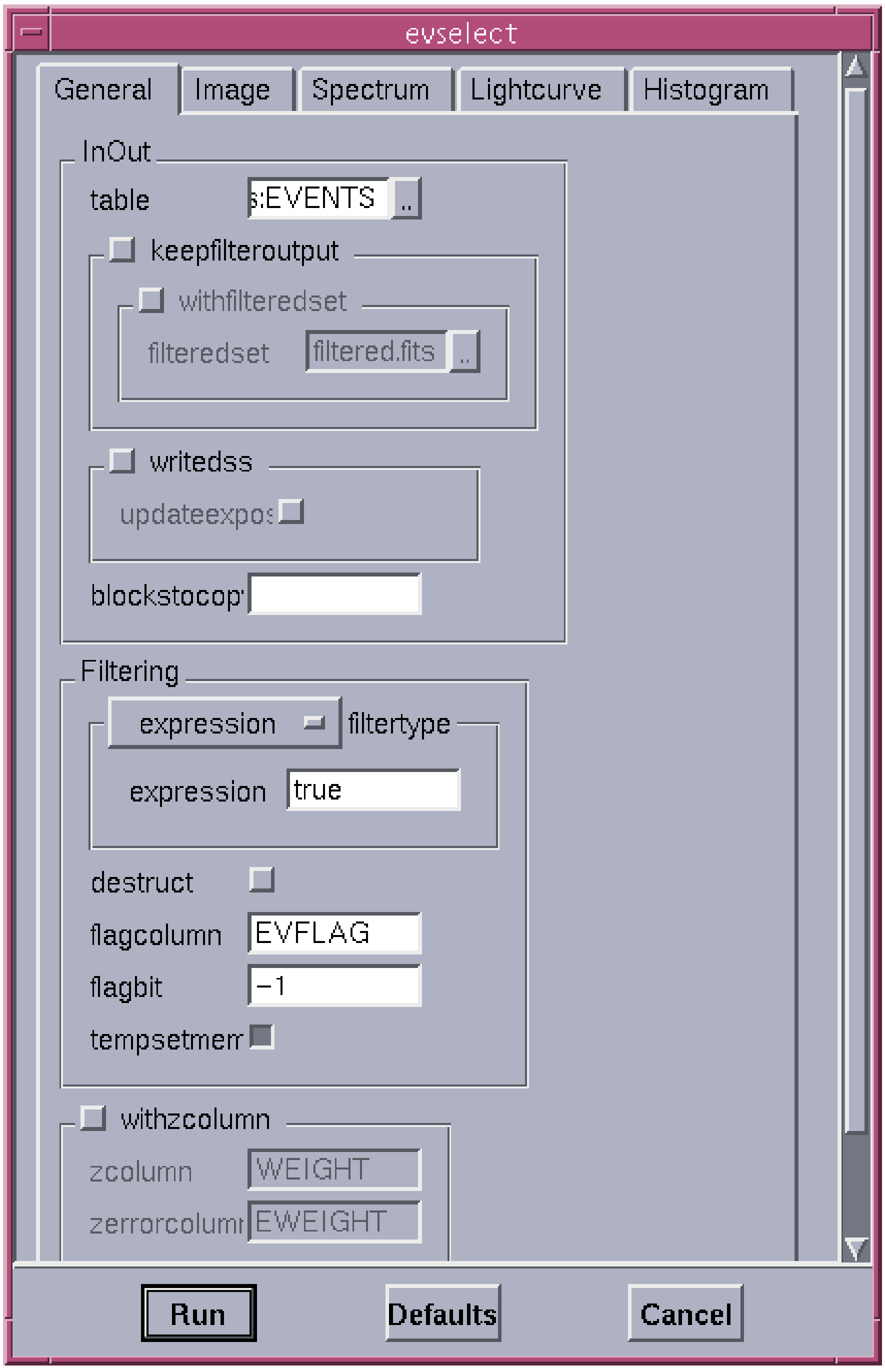


Figure 6.3: Evselect window

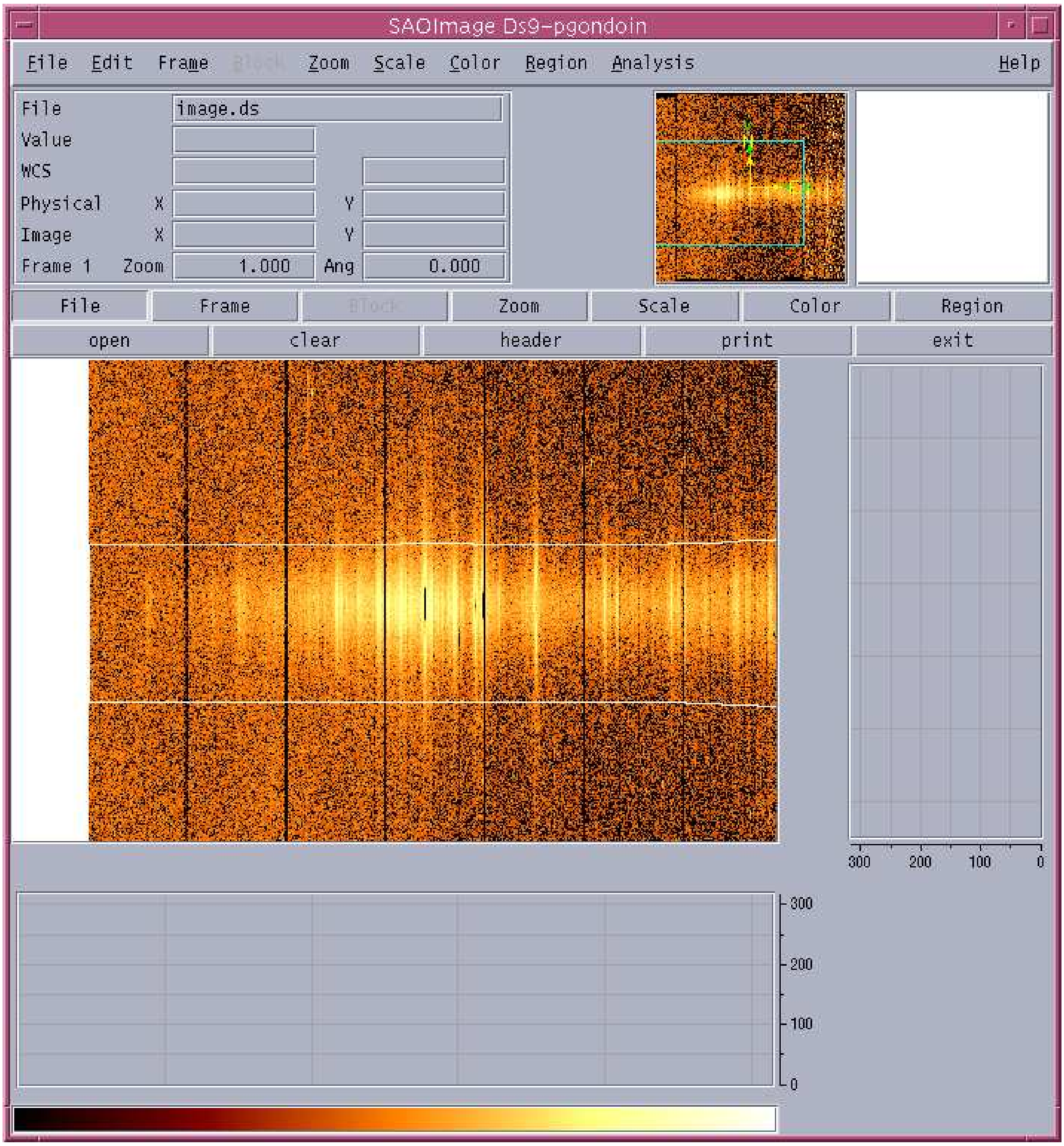


Figure 6.4: Image of the RGS spectrum

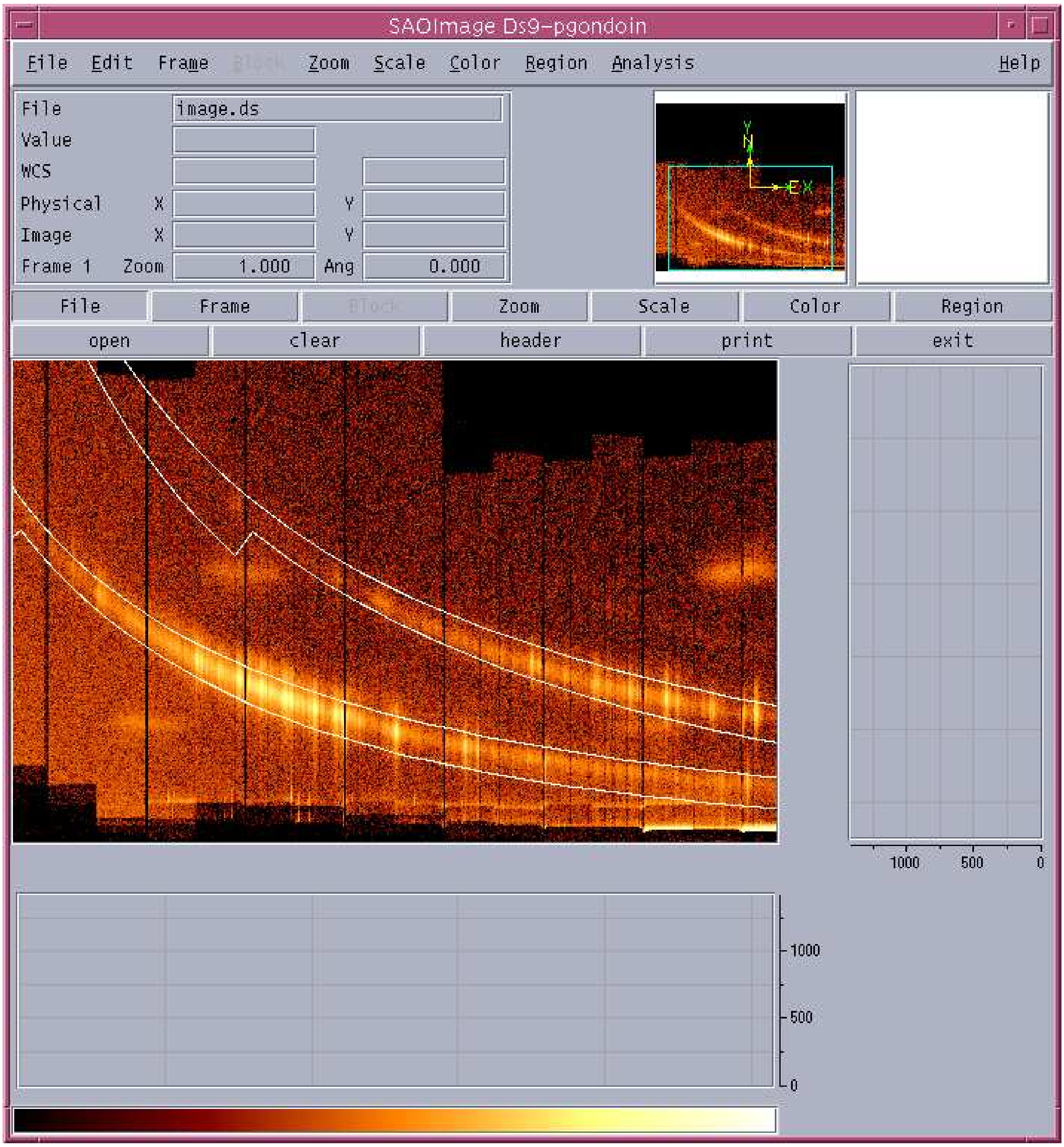
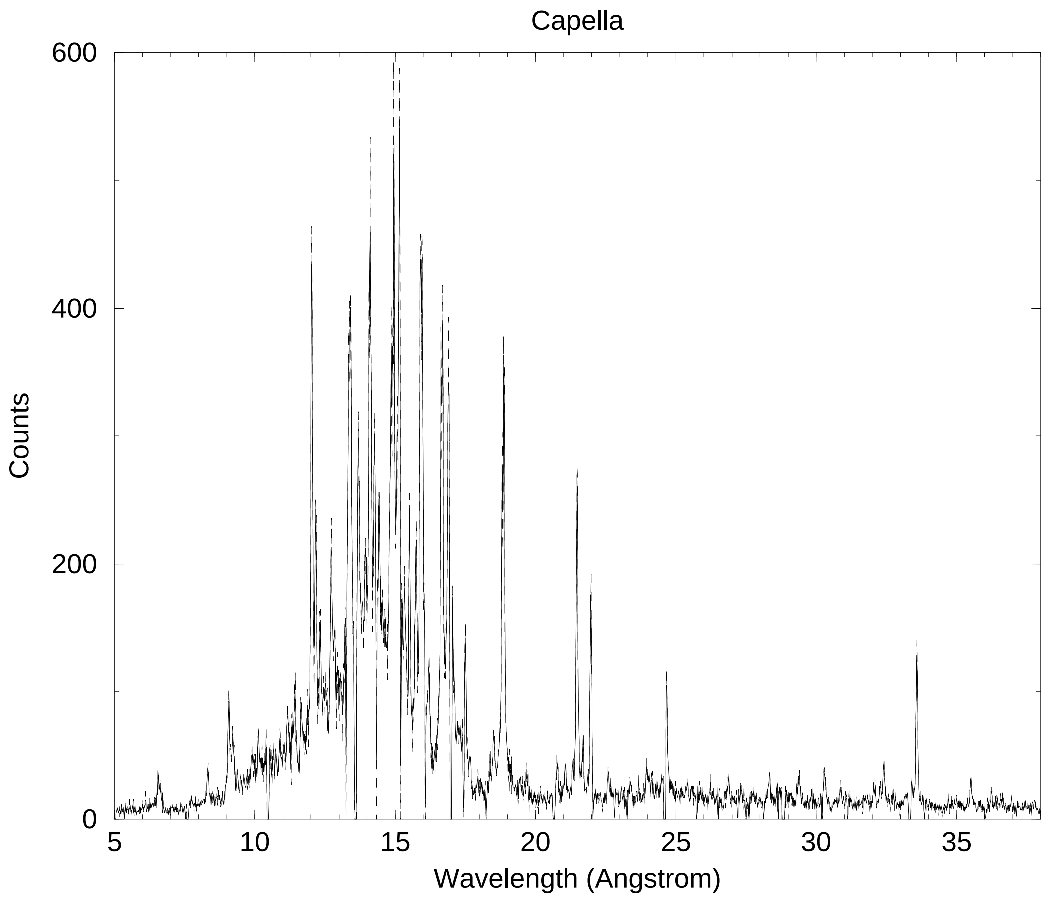


Figure 6.5: Energy image of the RGS spectrum. The two dispersion orders and the calibration source signal are clearly visible.



When inspecting the RGS spectra extraction regions, the user may encounter the following difficulties.

- 1 - Multiplicity of sources. Weak sources are aligned with the prime target in the dispersion direction, and are thus contributing to the background. The mechanism of the order separation is such that this background can be easily separated at high energy. However, in the low energy range, source confusion often occurs essentially because the energy resolution of the CCD is insufficient.
- 2 - Pileup due to spurious events that occurs in the second dipersion order at the spatial locations of strong lines in the first order.
- 3 - Shift of CCD gain due to straylight which effects the order selection and therefore the effective area of the system. The user can identify this problem by displaying a graphical cut in the PI versus dispersion plane, together with the events.
- 4 - Extraction region too wide in cross-dispersion or energy direction leading to the inclusion of a high level of noise in the spectra of targets which are not very bright.

One possibility to solve some of the difficulties mentioned above is to modify the region definition criteria (e.g pifraction or xdispfraction) in the rgsregion window of the *rgsproc* GUI interface. The pipeline processing task *rgsproc* has then to be re-run. The user shall then measure the impact of this modification on the analysis output using again the task *rgsimgplot* and visualizing the extracted spectrum using the task *dsplot*.

6.4 Creating RGS response matrices

Analysis of RGS spectra can be performed using specialised software packages including e.g XSPEC or SPEX. In addition to the calibrated products, these packages require the generation of specific files. The spectral fitting technique used by XSPEC requires a characterization of the RGS instrument responses to simulate output spectra. These response functions give the probability that an incoming photon of energy E will be detected in a channel I. The discrete response datasets are calculated by the sas task *rgsrmfgen*. The response is specific to a single diffraction order only. The order selection procedure uses the PHA distribution, which is provided as an input to this task. The response is a probability distribution calculated as a function of the grating angle. The *rgsrmfgen* core task corrects for:

- grating efficiencies
- vignetting by the telescope and the reflection grating structure
- exposure time
- telescope effective area
- CCD quantum efficiency
- order selection efficiencies.

rgsrmfgen creates an OGIP-compliant response matrix file as required for spectrum analysis within XSPEC. Because a certain level of compatibility between the spectrum and response matrix is required, it is typical and convenient (but not required) to create the spectrum first and provide it as in input to the response generator. The instrument response within each incident (photon) energy bin is calculated for every dispersion channel. The size of the matrix is determined, therefore, by the size of the spectrum and the number of incident energy bins (ebins) specified on the command line. A matrix of one hundred energy bins can be computed fairly quickly (about ten minutes) and is good for visual inspection, but is not good for analysis. 1000 bins is a minimum for use with *rgsfluxer* , and 3000 is probably acceptable for use with XSPEC. Be prepared to wait a while, and for periodic progress updates set the verbosity level to 4.

An example of an RGS data analysis session is as follows.

- specification of the observation data file (ODF) of interest

  ```
  setenv SAS_ODF 'path to the odf directory'
  ```

- creation of the associated calibration index file (CIF)

  ```
  cifbuild analysisdate='now' append=no calindexset='ccf.cif' \
  category='XMMCCF' ccfpath='path to the repository of calibration files' \
  fileglob='*.ccf|*.CCF' fullpath=yes ignorecategory=no masterindex=no \
  masterindexset='ccf.mif' missingfile='cifmif.asc' \
  observationdate='2000-06-10' recurse=no withccfpath=yes \
  withmasterindexset=no withmissingfile=no \
  withobservationdate=no
  ```

- specification of the newly created calibration index file (CIF)

  ```
  setenv SAS_CCF ccf.cif
  ```

- running the default RGS pipeline processing

```
rgsproc &
```

- vizualization of the extraction region after creation of a spatial image plot and energy image plot using *xmmselect*

```
rgsimplot device=/VCPS srclistset='0001_0000000001_R1S00400_sources.ds' \
endispset='RGS1banana.ds' spatialset='image.ds' --plotfile=a.ps \
--withspatialregionsets=yes --withendispregionsets=yes

gv a.ps
```

- if needed, reprocessing of RGS data using e.g. a user specified source position. (note that it is not necessary to specify rgsimgcolnames or withlambdacolumn=yes but radius=0 shall be given as input for a point source).

```
rgsproc sourceposition=user rasource=111.111 decsource=-11.11 \
label='mySource' radius=0
```

- the user can also specify good time intervalls as follows:

```
gtibuild file=gti.txt gtiset=gti.ds
rgsproc withgtiset=T gtiset=gti.ds &
```

- generation of a response matrix file

```
rgsrmfgen ebins=3400 emax=2.3 emin=0.3 file='RGS1_response.ds' order=1 \
 set='0001_0000000001_R1S00400_merged.ds' source=3 \
 sourceset='0001_0000000001_R1S00400_sources.ds'\
 spectrumset='' withbackground=no withspectrum=no
```

- launching e.g XSPEC and importing the newly created data background and response files using the XSPEC command data, backgrnd and resp.

Spectral fitting package, such as XSPEC, must be used for quantitative spectral analysis importing the response matrix file generated by *rgsrmfgen* and the .pha spectrum files produced by the pipeline.

The sas task *rgsfluxer* is an effective tool for visualizing the data free from the peculiarities of the instrument. *rgsfluxer* attempts to make a spectrum equivalent to the XSPEC 'unfolded' plot. However it only uses a default power law, and is therefore not correct. The intention is to provide an overview of the contents of the data, without spectral fitting to a model. *rgsfluxer* combines any collection of spectrum files generated by *rgsspectrum* to produce a single fluxed spectrum. In particular, it combines the data from both cameras into a single spectrum, thus eliminating many of the gaps between functioning detectors. Each raw spectrum is fluxed using the effective area calculated from its corresponding response matrix. It is only required that each spectrum be paired with a response matrix of the same reflection order. Using a response matrix computed with fewer than 500 incident energy bins is not recommended, and it is left entirely to the user to assess the advisability of combining dissimilar spectra.

6.5 Scientific analysis of RGS spectra

Once the calibrated data products (i.e either spectra or timeseries) and supporting files (e.g background files, response matrices ..) have been produced, the investigator can proceed with the scientific analysis of his data. This is generally performed with the support of specialized analysis software such as SPEX or XSPEC. Hence, for scientific analysis, the user is refered to the documentation associated with these packages.

Here are some information the user should be aware of when analysing RGS data.

- It is not possible to produce time series yet.
- the "net" spectra created by *rgsproc* are background substracted. The backgrounfd file are provided for information but shall not be used for spectral fitting.
- The user should estimate the background level. For that purpose, he can look at the spatial region selection plot (BETA_CORR versus XDSP_CORR) and red off the off-axis angle where he does not suspect the presence of any source. The result can be advantageously cross-checked on the EPIC images, where the following coordinate axis have to be considered. The RGS cross-dispersion direction relates to the X axis of the EPIC MOS 1 coordinate system, to the the Y axis of the EPIC MOS 2 coordinate system to the Y axis of the OM coordinate system. A detailled description of the experiments and spacecraft coordinate systems is found in the XMM Calibration Handbook. The RGS cross-dispersion is along the Y axis of the spacecraft. Note that the SAS task *rgsregion* can be called with an argument that gives the offset from a source ID in RGS FOV coordinates. This can be used to introduce an additional source position, which is used to determine the background.

Chapter 7

Analysis of OM optical monitor data

7.1 Inspecting the OM data package

The ODF files produced by a single OM exposure are listed in the following table.

File name	Description of the file content
0001-0000000000-OMS00100IMI.FIT	image
0001-0000000000-OMS00100THX.FIT	tracking history
0001-0000000000-OMS00100WDX.FIT	priority window data
0001-0000000000-OMX00100PEH.FIT	

Table 7.1: ODF files associated with a single OM exposure

OM images files have a file name ending with a "IMI.FIT". There is one imaged file per image mode window per exposure. Raw image files can be displayed with e.g the ds9 image viewer. It is also possible to combine all image windows of an OM default configuration before viewing them using the SAS task *rudi5mosaic*.

OM fast mode event lists have a file name ending with "FAE.FIT". There is one dataset for each BFAST identifier. This means that during normal operation there will be one fast mode event list per fast mode window. Inspection of the data files using e.g. the ftool viewer fv is possible.

OM reference frame data have a file name ending with a "RFX.FIT". Generally one file is present for each exposure, except if tracking was switched off. This is the case for certain OM engineering modes and certain filter elements (e.g. grism).

OM tracking history file data have a file name ending with a "THX.FIT". Generally one file is present for each exposure, except when tracking was explicitly switched off, or when no suitable stars for tracking were found in the reference frame at the beginning of the exposure. Inspection of this files with e.g. the ftool viewer fv is possible. Plot of column DX and DY show the attitude stability during the exposure as follows. (Units are 1/1000 subpixel).

```
dsplot table='0051_0099750501_OMS00700THX.FIT:OMTHX1' &
 --withx=true x='DX' --withy=true y='DY' --withscatter=true
```

Note that a similar plot is part of the PPS products. The PPS product file named TSHPLT.PDF file provides a visualiation of the pointing stability during the exposure.

OM field acquisition data have a file name ending with a "PAX.FIT". There is only one field acquisition dataset present per observation. In order get a feeling on the absolute pointing accuracy, the user is advise to look at this field acquisition datatset (PAX.FIT). The paramaters DX, DY provide the absolute pointing error in units of 0.001 subpixel.

OM priority window data have a file name ending with a "WDX.FIT". One priority window dataset is present for each exposure.

OM priority fast mode data have a file name ending with a "PFX.FIT". There is one dataset for each BFAST identifier. This means that during normal operation there will be one fast mode event list per fast mode window.

The following table provides a list of the OM calibration files. For a detailed description of these files and of their usage, the user is refered to the CCF interface document and to the Calibration Access and Data Handbook. Some of these files (e.g DarkFrame, DiffuseGAla, quickmag, Zodialcal) are not used by the SAS.

File name	File Content
AstroMet	deviations of pixel positions from ideal (i.e. flat grid) locations
BadPix	Table of bad pixels position, type of defect and the severity level.
ColTrans	coefficients for color transformation into a standard system.
DarkFrame	dark current map.
DiffuseGala	intensity map of diffuse galactic emissions
HKParmInt	House keeping parameter ranges used to generate good time intervals.
LightLeak	basic shape of light leak features
LrgScaleSens	correction maps for filter trnasmission variations.
MiscData	parameters for calculations of instrument configuration.
PhotToNat	correction coefficients of count rates for detector non-linearity and aging.
PhotVsLam	Quantum efficiency (filter+detector) as a function of wavelength.
PixToLam	conversion from off-set pixel to wavelength for the two grisms.
PixToPixSens	map describing the non-uniformity of the detector response.
PSF1DRB	supersampled 1-dimensional point spread function for each filter.
QuickMag	a rough count to magnitude conversion table for different spectral types.
StrayLight	parameterisation of the OM straylight rejection performance.
TimeCorr	parameterisation of timing related corrections.
Zodiacal	intensity map of the zodical light and average spectrum.

Table 7.2: Calibration files of the Optical Monitor

7.2 Description of the OM data processing chain

In order to process the OM data successfully a set of input datasets is required. Input datasets are not changed during the om processing with *omichain*. They are copied to intermediate datasets and keyword. Extensions are added as needed. Intermediate task products, which are passed from one task to another, are generally not described.

As can be seen in the flow chart of the OM processing chain, the OM tracking information is treated before and indepedantly from the image data processing. For each task description, a reference is given to the analysis step in the processing example.

7.2.1 Handling of tracking data

Two output dataset related to tracking data are distributed as pipeline products, namely the TSH-PLT.PDF and the TSTRT.FIT file. They are created as follows.

- *omprep* (step1). The task *omprep* doesn't generate final data products but only intermediate files. OM images, tracking history files and fast mode files if any are copied into temporary files. Further information is added via keywords. Intermediate products are generated, which carry information extracted form the tracking history file. If no tracking history file was produced, these are generated on the fly assuming perfect stable pointing. Tracking history data can be missing if no suitable tracking stars were available during observation. This can also occur when certain instrument modes (engineering modes) or when some filter elements (e.g grisms) are used.
- *omdrifthist* (step2). This task creates the TSHPLT.PDF file, which provides an overview of the pointing stability during an exposure. The file contains diagrams illustrating the absolute and incremental OM drift.
- *omthconv* (step3). This task creates the TSTRT.FIT file, which contains the tracking star time series, i.e the average count rate (cts/second) per tracking frame for each tracking star.

7.2.2 Handling of image mode data

There are three final data products per observation window (OSW), namely IMAGE_.FIT, SIMAGE.FIT and SWSRLI.FIT. Two additional products FLAFLD.FIT and OWSRLS.FIT. are generated on a per observation basis. The task *omichain* loops over all image mode data of an observation. The following tasks are applied to each individual observation window.

- *omprep* (step4). The task *omprep* copies the raw images into an intermediated output file. It adds several keywords to this intermedidate output file. Although this file is not a final data product, it is carried forward within the processing of OM imagage data and is eventually contained within the final IMAGE_ product.
- *omcosflag* (step5). The task *omcosflag* generates a quality map. Bad pixels are removed according to the tracking history information. Although the quality array is currently not used in the analysis, the quality array (bad pixels are flagged as non-zero there) itself is written into the quality extension of the intermediate image files. The latter is propagated eventually into the IMAGE_.FIT PPS output file.
- *omflatgen* (step6.1). The task *omflatgen* is an external task running only at the SSC, which generates the flatfield file for the observation. The flatfield file is distributed as a PPS product (FLAFLD). In absence of a FLAFLD.FIT file in the PPS products, a dummy file can be generated by the task *omflatgen*. In this case the flatfield response is set to unity.
- *omflatfield* (step6). The task *omflatfield* applies the tracking history file (as contained in the intermediate processing product of the tracking history task) to the OM flatfield file (FLAFLD). The resulting drift corrected flatfield file is applied to the intermediate OM image file. The flatfield corrected OM image is written as an intermediate data product, which is used as an input to *ommodmap*.
- *ommodmap* (step7). The task *ommodmap* applies the modulo 8 correction to an OM image. The modulo 8 background fluctuations is introduced to an OM image by imperfections of the photon centroiding algorithm. An intermediate image file is generated, which acts as input to the task *omdetect*.

- *omdetect* (step8). The task *omdetect* applies a source detection algorithm to the flatfield and background corrected OM image. The output is a source list (SWSRLI) which contains the source position in absolute detector pixel and the calculated source count rate (cts/sec). An optional output file called the region file is generated if the parameter outputregionfile is set to true. The region file describes the area associated with each identified source in the image. It is used by the image display tool. The source counts are extracted in an ellipse within the 3 sigma source radius.
- *ommag* (step9). The task *ommag* converts count rates into the instrumental OM photometric system. Count rates are corrected for coincidence losses and detector deadtime. The so corrected count rates are converted into instrumental magnitudes. The task requires input from the source list file SWSRLI and the priority window files (WDX). The output is added as extra column to the source list file (SWSRLI). The sensitivity limit of the specific exposure is written as a keyword in the source list.
- *omatt* (step10). The task *omatt* reads the intermediate image generated by ommod8 and reads the source list file generated by *omdetect* and *ommag*. It converts detector position from pixel coordinates into ra/dec positions based on information from the attitude history file, the instrument boresight file and the OM filter distortion map. The celestial position are added as two new columns to the SWSRLI.FIT file. A skyimage (SIMAGE.FIT) is generated from the input image, by resampling the image, applying the distortion correction and north align the image. The skyimage provides the most accurate positional information of an image in the OM image data processing chain. *omatt* also adds WCS to the input flatfielded and mod8 corrected OM image). However no resampling is taken place and no distortion correction is applied, which makes the positional information in this image less accurate than the positional information in the skyimage (SIMAGE). The input image with added WCS coordinates is rename into IMAGE_.FIT by *omichain* and is delivered as a PPS product.
- *omsrclistcomb* (step12). In case of multiple OM observations of the same field, *omsrclistcomb* combines the different source lists into one single observation source list file (OWSRLS.FIT). *omsrclistcomb* is invoked when the analysis of the invidual science windows is finished. The input parameter "nsigma" defines the criteria for the spatial matching of two sources in the merging of the source lists. *omsrlistcomb* generates a master source list with ra/dec coordinates, ra/dec errors and galactic coordinates. The significance of detection, the instrumental magnitude and its error, the semi-axis of the source detection ellipse and a set of flags are written in the source list for each filter used in the observation. If a colour transformation can be applied, the source brightness will also be listed in a standard photometric system.

7.2.3 Further notes on processing of OM data

The default settings of *omichain* overwrites temporary files at several stages, e.g when looping over multiple exposures or even within the analysis of a single window. This can be avoided by specifying unique output file names. It is recommended to clear out (removing or renaming) products of previous *omichain* runs before re-invoking *omichain*. The OM fast mode chain is not yet available.

7.3 List of OM pipeline products

This section provides a general description of the final pipeline products which are distributed to the user. Similar files are generated by running the chain scripts, but the user shall be aware thatthey may have subtle differences compared to the pipeline products.

OM pipeline products are named as "PooooooooooOMSeeettttttt1000.fff", where:

- oooooooooo stands for the observation ID,
- eee is the exposure ID,
- tttttt is the data type,
- fff is the file type (fit, png etc.)

In the following we refer to a data type by the ttttt identifier, e.g. such as tttttt= SIMAGE for the OM sky image product.

There are two pipeline products associated with the tracking history files, namely the TSHPLT.PDF and the TSTRT.FIT file. There are three pipeline data products per OSW, namely IMAGE_.FIT, SIMAGE.FIT and SWSRLI.FIT. There are two pipeline products generated on a per observation basis, which are FLAFLD.FIT and OWSRLS.FIT.

TSHPLT.PDF provides a visualisation of the tracking history file. It gives an overview of the pointing stability in the course of an exposure. The drift history of an exposure is displayed as a vector diagram. Each data point represents the average attitude solution of one tracking frame, lasting typically 20 sec. The pointing drift is calculated with respect to the OM attitude at the time where the reference frame was defined. The reference attitude of an exposure is determined by a short exposure of 10 sec duration at the begin of an science exposure. Histograms at the side of the vector diagram show the projection of the OM pointing drift onto the x- and y-direction. The second output page shows the incremental OM drift between 2 subsequent tracking frames. A clustering of points at one location would indicate a systemtic drift into one direction. Typically the pointing stability is better than 1 arcsec during one exposure, which corrsponds two 2 pixel on the diagrams. There is one TSHPLT PDF file produced for each THX file.

TSTRTS file:. There is one TSTRTS FITS file produced for each THX file. The binary extension comprises n columns, i.e one for each tracking star used. Each column list the tracking star count rate (cts/sec) of a tracking star averaged over the tracking frame duration. Each row indicates a new tracking frame and one column corresponds to a timeseries of a specific star. The number of columns corresponds to the number of tracking stars used in an exposure. It can be any number up to 15.

IMAGE_.FIT file: The Primary array contains the flatfield and mod8 corrected image. The header contains the World coordinate system parameters applied to the raw image. The RAW extension contains the original raw OSW image converted into real number represenation. The MODES extension lists the details of the window configuration. The QUALITY extension contains in an image the quality array. Any non-zero entry in the quality array reflects a bad pixel notified in the calibration bad pixel map or any location of the image where problems were encountered during the image processing.

SIMAGE.FIT file: The primary array of this dataset contains the north aligned sky image. The image is flatfielded, mod8 corrected, resampled and distortion corrected. The image is north aligned. WCS coordinates are contained as header keywords.

SWSRLI.FIT file: The exposure source list file contains beside the source identifier within the exposure and the observation, the sensitivity limit of the plate expressed in count rates and in instrumental magnitudes. The binary extensions holds the list of detected sources. There are the following entries for each detected source: position within the OSW in pixel, the source position in ra/dec and galactic coordinates, the positional uncertainty, the extracted source count rates and its error estimate, the significance of the source detection, the brightness in instrumental magnitudes, the shape of the source expressed as the two semi-axes of an ellipse (for point sources the two axes should agree), the orientation of the ellipse and three quality flags, describing whether the source is thought to be extended, confused or affected by bad pixels.

FLAFLD.FIT file: Image either part of the pipeline products or generated by *omflatgen*. There is one file per observation containing an image extension. The image describes the flatfield response covering the whole detector at coarse resolution.

OWSRLS.FIT file: This file contains the combined source list for all OSWs of an observation. Information of sources whose spatial position coincides within the specified "nsigma" are merged together. An observation source identifier is assigned to each source, which can be used to look up the sources in OSW source lists (SWSRLI files). The list contains, observation source identifier, position in ra/dec and gal. coordinates, positional uncertainty, the detection significance in the different filters, the source brightness expressed in instrumental magnitudes and its uncertainty, the quality, source extension and confusion flag for each filter, the characterisation and orientation of the spatial extent (described as in the SWSRLI files by the two semiaxes and the orientation angle in respect to the image x-axis) for each filter and finally if a transformation was possible (only applicable to U, B, V) the source brightness in a standard photometric system.

7.4 Example of an OM image processing run

This section describes task by task a run of the OM imaging processing. The working directory needs to contain spacecraft and housekeeping files of the following form (the first two sets of numbers in the name depends on the observation/exposure number):

```
0070_0123700101_SCX00000SUM.SAS  - ASCII observation summary file
0070_0123700101_SCX00000TCS.FIT  - Spacecraft Time correlation file
0070_0123700101_SCX00000ATS.FIT  - Spacecraft Attitude file
0070_0123700101_OMX00000NPH.FIT  - Observation non-periodic houskeeping file
0070_0123700101_OMX00000PEH.FIT  - Observation periodic housekeeping file
```

The SAS_ODF environment variable shall be set to a directory containing the data. To process a single exposure mamed image 0070_0123700101_OMS00400IMI.FIT, this directory shall contain the following files:

```
0070_0123700101_OMS00400WDX.FIT  - Exposure priority window file
0070_0123700101_OMS00400THX.FIT  - Exposure tracking history file
0070_0123700101_OMS00400IMI.FIT  - Exposure image file (small window)
```

Note that output postscript file can be easily converted to .PDF format using ps2pdf.

```
step1> omprep set=0070_0123700101_OMS00400THX.FIT
nphset=0070_0123700101_OMX00000NPH.FIT
pehset=0070_0123700101_OMX00000PEH.FIT
wdxset=0070_0123700101_OMS00400WDX.FIT
outset=tmp_tracking


step2> omdrifthist set=tmp_tracking plotfile=P0123700101OMS004TSHPLT0000.ps


step3> omthconv thxset=tmp_tracking
nphset=0070_0123700101_OMX00000NPH.FIT
outset=P0123700101OMS004TSTRTS0000.FIT
```

```
step4> omprep set=0070_0123700101_OMS00400IMI.FIT
nphset=0070_0123700101_OMX00000NPH.FIT
pehset=0070_0123700101_OMX00000PEH.FIT
wdxset=0070_0123700101_OMS00400WDX.FIT
outset=tmp_image_1

step5> omcosflag set=tmp_image_1 thxset=tmp_tracking
```

The flat field file (FLAFLD) used in the following routine is assumed to be a product in the ODF. However, if no flatfield file is present in the ODF, one can be generated using the task *omflatgen* as follows.

```
step6.1> omflatgen outset='P0123700101OMX000FLAFLD0000.FIT'
```

Currently, the output flatfield (primary extension) will be set to unity since in the absence of other arguments, the flatfield is obtained from the calibration file which currently contains a unit image. The uncertainty (secondary extension) is set to zero for the same reason.

```
step6> omflatfield set=tmp_image_1
thxset=tmp_tracking inorbitflatset=P0123700101OMX000FLAFLD0000.FIT
ppsflatset=tmp_flat_field outset=tmp_image_2


step7 >ommodmap set=tmp_image_2  ppsproduct=yes ppsmod8set=tmp_mod8
outset=tmp_image_3 nsig=5


step8 >omdetect set=tmp_image_3 outputregionfile=false
regionfile=region_file outset=osw_list


step9 >ommag set=osw_list wdxset=0070_0123700101_OMS00400WDX.FIT


step10>omatt set=tmp_image_3 sourcelistset=osw_list
ppsoswset=P0123700101OMS004SIMAGE1000.FIT usecat=F
```

Note that currently the usage of a catalogue for source crosscorrelation is disabled. The detected sources can be overlayed on the OSW as follows.

```
step11>implot set=P0123700101OMS004SIMAGE1000.FIT
srclistset=osw_list withsrclistset=y device=/XW colour=4 thick=4 radius=8
```

In the standard automatic SSC pipeline processing, the tmp_image files are re−used and thus overwritten. In the processing chain, they are distinguished so that intermediate stage output can be looked at if desired. In the processing by *omichain*, the file tmp_image_3 is renamed to // P0123700101OMS004IMAGE_1000.FIT. osw_list is assigned the name // P0123700101OMS004SWSRLI1000.FIT throughout. Elsewhere in the above example, product names are used where they are similarly used in the SSC pipeline. If multiple images are processed, the source lists can be combined as follows.

```
step12>omsrclistcomb
sourcelistsets=P0123700101OMS004SWSRLI1000.FIT .........
nsigma=3 outset=P0123700101OMS000OBSRLS0000.FIT
```

where .... is a continuing list of SWSRLI files.

7.5 Detailled description of OM pipeline products

TSTRTS.FIT file: (written by *omprep*)

```
DETNAM  = 'REDUNDANT'          / PRIME or REDUNDANT (detector chain)
DATAMODE= 'TRACKING'           / Instrument mode    (mode of science window)
FILTER  = 'V       '           / Filter ID          (filter wheel position)
CONTENT = 'OM TRACKING STAR TIMESERIES' / Content of file
EXPOSURE= 1.00000000000000E+03 / [Exposure (secs)]  (from housekeeping alerts)
END
```

The binary extension comprises of n columns (one for each tracking star used). Each column list the tracking star count rate [cts/sec] of a tracking star averaged over the tracking frame duration. Each row indicates a new tracking frame (looking down one column corresponds to a timeseries of a specific star). (the number of columns corresponds to the number of tracking stars used in an exposure. It can be any number between TBD and 15).

```
XTENSION= 'BINTABLE'           / binary table extension
TFIELDS =                   10 / number of fields in each row
TTYPE1  = 'RATE1   '           / The name of this column
TFORM1  = 'E       '           / data format of field: 4-byte REAL
TUNIT1  = 'count/s '           / physical unit of field
.
.
.
TTYPEnn = 'RATEnn  '           / The name of this column
TFORMnn = 'E       '           / data format of field: 4-byte REAL
TUNITnn = 'count/s '           / physical unit of field
EXTNAME = 'RATES   '           / The name of this table
HDUCLASS= 'OGIP    '           / Format conforms to OGIP/GSFC conventions
HDUCLAS1= 'LIGHTCURVE'         / File contains a time-series
HDUCLAS2= 'TOTAL   '           / Not background subtracted
HDUCLAS3= 'RATE    '           / Not exposure corrected
TIMEDEL = 2.00000000000000E+01 / duration of tracking frame
TSTART  = 7.31910055306569E+07 / start of exposure
TSTOP   = 7.31919855359959E+07 / end of exposure
TIMEUNIT= 'S       '           / unit for TSTART, TSTOP, TIMEZERO, TELPASE
END
```

IMAGE_.FIT:

Primary array containing the flatfield and mod8 corrected image. The header contains the World coordinate system parameters applied to the raw image.

```
CTYPE1  = 'RA---TAN'           / Coordinate system for X
CRPIX1  = 2.48000000000000E+02 / [WCS reference pixel]
CRVAL1  = 1.63195572951499E+02 / [Coordinate of pointing direction for X]
CDELT1  = -2.64729444444444E-04 / [WCS pixel size]
CTYPE2  = 'DEC--TAN'           / Coordinate system for Y
CRPIX2  = 2.48000000000000E+02 / [WCS reference pixel]
CRVAL2  = 5.74785924331710E+01 / [Coordinate of pointing direction for Y]
CDELT2  = 2.64729444444444E-04 / [WCS pixel size]
CROTA2  = 3.13038482666016E+02 / [image rotation - DEPRECATED]
PC001001= 6.82489465957658E-01 / [Coord. Descrp. Matrix: cos(CROTA2)]
PC001002= 7.30895429495103E-01 / [Coord. Descrp. Matrix: -sin(CROTA2)]
PC002001= -7.30895429495103E-01 / [Coord. Descrp. Matrix: sin(CROTA2)]
PC002002= 6.82489465957658E-01 / [Coord. Descrp. Matrix: cos(CROTA2)]

This extension contains the  original OSW converted into real representation:
XTENSION= 'IMAGE   '           / IMAGE extension
EXTNAME = 'RAW     '           / The name of this array
END

The MODES extension carries the window detail information
TTYPE1  = 'MODE    '           / The name of this column
TFORM1  = 'J       '           / data format of field: 4-byte INTEGER
TUNIT1  = 'none    '           / physical unit of field
TTYPE2  = 'WINDOWX0'           / The name of this column
TFORM2  = 'J       '           / data format of field: 4-byte INTEGER
TUNIT2  = 'pixels  '           / physical unit of field
TTYPE3  = 'WINDOWY0'           / The name of this column
TFORM3  = 'J       '           / data format of field: 4-byte INTEGER
TUNIT3  = 'pixels  '           / physical unit of field
TTYPE4  = 'WINDOWDX'           / The name of this column
TFORM4  = 'J       '           / data format of field: 4-byte INTEGER
TUNIT4  = 'pixels  '           / physical unit of field
TTYPE5  = 'WINDOWDY'           / The name of this column
TFORM5  = 'J       '           / data format of field: 4-byte INTEGER
TUNIT5  = 'pixels  '           / physical unit of field
EXTNAME = 'MODES   '           / The name of this table
END
```

The QUALITY extension contains an image representing the quality array. Any non-zero element of hte quality array reflects a bad pixel notified in the calibration bad pixel map or a image area where problems were encountered during the image processing.

```
XTENSION= 'IMAGE   '           / IMAGE extension
EXTNAME = 'QUALITY '           / The name of this array
END
```

SIMAGE.FIT file:

The primary array of this dataset contains the north aligned sky image. The image is flatfielded, mod8 corrected, resampled and distortion corrected. The image is north aligned. WCS coordinates are contained as header keywords.

```
CTYPE1  = 'RA---TAN'           / Coordinate system for X
```

```
CRPIX1  = 3.39463653564453E+02 / [WCS reference pixel]
CRVAL1  = 1.63195572951499E+02 / [Coordinate of pointing direction for X]
CDELT1  = -2.64729444444444E-04 / [WCS pixel size]
CTYPE2  = 'DEC--TAN'           / Coordinate system for Y
CRPIX2  = 3.50077117919922E+02 / [WCS reference pixel]
CRVAL2  = 5.74785924331710E+01 / [Coordinate of pointing direction for Y]
CDELT2  = 2.64729444444444E-04 / [WCS pixel size]
```

SWSRLI.FIT file:

The exposure source list file contains beside the source identifier within the exposure and the observation, the sensitivity limit of the plate expressed in count rates and in instrumental magnitudes. The binary extensions holds the list of detected sources. There are the following entries for each detected source: position within the OSW in pixel, the source position in ra/dec and galactic coordinates, the positional uncertainty, the extracted source count rates and its error estimate, the significance of the source detection, the brightness in instrumental magnitudes, the shape of the source expressed as the two semi-axes of an ellipse (for point sources the two axes should agree), the orientation of the ellipse and three quality flags, describing whether the source is thought to be extended, confused or affected by bad pixels.

```
RA_PNT  = 1.63170876904662E+02 / [degrees] RA of instrument pointing
DEC_PNT = 5.74519173372917E+01 / [degrees] DEC of instrument pointing
PA_PNT  = 3.13038486320053E+02 / [degrees] ROLL angle of instrument pointing
RADECSYS= 'FK5     '           (standard system)
EQUINOX =                 2000 / Equinox of celestial coordinate system
```

Start and end of data acquisition as listed in observation summary file.

```
DATE-OBS= '2000-04-27T02:50:07' / Date observations were made
DATE-END= '2000-04-27T03:06:47' / Date observations ended
```

Detsize written by *omprep* describes the valid range of image pixels addresses. This includes the binning. 2x2 binned data have a valide range 1 to 1024. Position of the image on the detector is specified by DETSIZE.

```
DETSIZE = '[1:2048,1:2048]'    / Mosaic Keywords
DETSEC  = '[ 912:1136, 912:1136]' / Mosaic Keywords
OSW_ID  = '0       '           / Identifier for current OSW
BLANK   =          -2147483648 / Value indicating missing pixel
BINBPE  =                    F / BPE binning enabled (electronic binning)
BINAX1  =                    0 / CCD x-axis binning 2**N (DPU binning - in x)
BINAX2  =                    0 / CCD y-axis binning 2**M (DPU binning - in y)
FRMTIME =                20480 / [1/1024 secs] Tracking frame duration
TMEXP_ID=                 1101 / Exposure identifier
CRLIMIT = 5.51816374063492E-02 / [counts/sec] Detection limit at 5 sigma
MAGLIMIT= 2.12180881500244E+01 / detection limiting magnitude from CRLIMIT)
NAXIS2  =                   11 / number of rows i.e number of objects detected
TTYPE1  = 'SRCNUM  '           / The name of this column (sequential number)
TFORM1  = 'J       '           / data format of field: 4-byte INTEGER
TTYPE2  = 'XPOS    '           / xposition of source in image pixels,
(relative to lower left corner of OSW (=0,0))
TFORM2  = 'E       '           / data format of field: 4-byte REAL
TUNIT2  = 'pixels  '           / physical unit of field
```

```
TTYPE3  = 'YPOS    '           / y position in image pixels
TFORM3  = 'E       '           / data format of field: 4-byte REAL
TUNIT3  = 'pixels  '           / physical unit of field
TTYPE4  = 'RA      '           / Source RA in degrees
TFORM4  = 'D       '           / data format of field: 8-byte DOUBLE
TUNIT4  = 'degrees '           / physical unit of field
TTYPE5  = 'DEC     '           / source Dec in degrees)
TFORM5  = 'D       '           / data format of field: 8-byte DOUBLE
TUNIT5  = 'degrees '           / physical unit of field
TTYPE6  = 'POSERR  '           / positional uncertainty in arcseconds
TFORM6  = 'E       '           / data format of field: 4-byte REAL
TUNIT6  = 'arcsec  '           / physical unit of field
TTYPE7  = 'RATE    '           / count rate integrated over area
TFORM7  = 'E       '           / data format of field: 4-byte REAL
TUNIT7  = 'counts/sec'         / physical unit of field
TTYPE8  = 'RATE_ERR'           / error estimate on count rate
TFORM8  = 'E       '           / data format of field: 4-byte REAL
TUNIT8  = 'counts/sec'         / physical unit of field
TTYPE9  = 'SIGNIFICANCE'       / ratio of integrated count rate to
 standard deviation on background
TFORM9  = 'E       '           / data format of field: 4-byte REAL
TUNIT9  = 'sigma   '           / physical unit of field
TTYPE10 = 'MAG     '           / instrumental magnitude derived from
 count rate and area used
TFORM10 = 'E       '           / data format of field: 4-byte REAL
TUNIT10 = 'magnitudes'         / physical unit of field
TTYPE11 = 'MAGERR  '           / error estimate on the instrumental magnitude
TFORM11 = 'E       '           / data format of field: 4-byte REAL
TUNIT11 = 'magnitudes'         / physical unit of field
TTYPE12 = 'FWHM_MAJ'           / FWHM of the major axis of the ellipse
characterising the source in image pixels
TFORM12 = 'E       '           / data format of field: 4-byte REAL
TUNIT12 = 'pixels  '           / physical unit of field
TTYPE13 = 'FWHM_MIN'           / FWHM of the minor axis of the ellipse
TFORM13 = 'E       '           / data format of field: 4-byte REAL
TUNIT13 = 'pixels  '           / physical unit of field
TTYPE14 = 'PA      '           / Position angle of the major ellipse angle,
        (measured counterclockwise relative to the positive detector X axis)
TFORM14 = 'E       '           / data format of field: 4-byte REAL
TUNIT14 = 'degrees '           / physical unit of field
TTYPE15 = 'QFLAG   '           / quality flag indicating that the source
  region contains pixels of suspect quality
TFORM15 = 'B       '           / data format of field: BYTE
TUNIT15 = 'none    '           / physical unit of field
TTYPE16 = 'CFLAG   '           / confusion flag: flag indicating source may
be confused with another source
TFORM16 = 'B       '           / data format of field: BYTE
TUNIT16 = 'none    '           / physical unit of field
TTYPE17 = 'EFLAG   '           / Extension flag: indicates that source extent
 may be greater than the instrument PSF
TFORM18 = 'D       '           / data format of field: 8-byte DOUBLE
TUNIT18 = 'Deg     '           / physical unit of field
TTYPE19 = 'BII     '           / source position in galactic latitude (deg.)
TFORM19 = 'D       '           / data format of field: 8-byte DOUBLE
TUNIT19 = 'Deg     '           / physical unit of field
```

```
TTYPE20 = 'SRC_ID  '           / sequence number of this source as found in
the combined source list
TFORM20 = 'J       '           / data format of field: 4-byte INTEGER
EXTNAME = 'SRCLIST '           / The name of this table
END
```

FLAFLD.FIT file:

One such image file is generated by *omflatgen* per observation covering the whole detector at coarse resolution.

```
DATAMODE= 'IMAGING '           / Instrument mode
CONTENT = 'OM OSW FLAT FIELD'  / Contents
BINBPE  = 'T       '           / BPE binning enabled
BINAX1  =                    0 / CCD x-axis binning 2**N
BINAX2  =                    0 / CCD y-axis binning 2**M
CREATOR = 'omflatgen  []'      / name of code that generated this file
```

OWSRLI.FIT file:

This file contains the combined source list for all OSWs of an observation. Information of sources whose spatial position coincides within the specified "nsigma" are merged together. An observation source identifier is assigned to each source, which can be used to look up the sources in OSW source lists (SWSRLI files). The list contains, observation source identifier, position in ra/dec and gal. coordinates, positional uncertainty, the detection significance in the different filters, the source brightness expressed in instrumental magnitudes and its uncertainty, the quality, source extension and confusion flag for each filter, the characterisation and orientation of the spatial extent (described as in the SWSRLI files by the two semiaxes and the orientation angle in respect to the image x-axis) for each filter and finally if a transformation was possible (only applicable to U, B, V) the source brightness in a standard photometric system.

```
XTENSION= 'BINTABLE'           / binary table extension
NAXIS2  =                  747 / number of rows in table i.e number of unique
 sources detected in all exposures
TTYPE1  = 'SRCNUM  '           / sequence number for sources within combined
 list from corresponding OSW source list file
TFORM1  = 'J       '           / data format of field: 4-byte INTEGER
TTYPE2  = 'RA      '           / weighted average of RA's in different
exposures
TFORM2  = 'D       '           / data format of field: 8-byte DOUBLE
TUNIT2  = 'degrees '           / physical unit of field
TTYPE3  = 'DEC     '           / weighted average of DEC's in all exposures
TFORM3  = 'D       '           / data format of field: 8-byte DOUBLE
TUNIT3  = 'degrees '           / physical unit of field
TTYPE4  = 'POSERR  '           / positional uncertainty derived from averaging
TFORM4  = 'E       '           / data format of field: 4-byte REAL
TUNIT4  = 'arcsec  '           / physical unit of field
TTYPE5  = 'LII     '           / average RA/DEC converted to galactic coords
TFORM5  = 'D       '           / data format of field: 8-byte DOUBLE
TUNIT5  = 'degrees '           / physical unit of field
TTYPE6  = 'BII     '           / as LII but galactic latitude
TFORM6  = 'D       '           / data format of field: 8-byte DOUBLE
TUNIT6  = 'degrees '           / physical unit of field
```

```
TTYPE7  = 'SIGNIF_WHITE'       / detection significance for white filter
 only if source detected in this filter. This
is repeated below for all other filters used
 during observation in which at least one
 source was detected.
TFORM7  = 'E       '           / data format of field: 4-byte REAL
TUNIT7  = 'sigma   '           / physical unit of field
TTYPE8  = 'SIGNIF_UVW2'        / The name of this column
TFORM8  = 'E       '           / data format of field: 4-byte REAL
TUNIT8  = 'sigma   '           / physical unit of field
TTYPE9  = 'SIGNIF_UVW1'        / The name of this column
TFORM9  = 'E       '           / data format of field: 4-byte REAL
TUNIT9  = 'sigma   '           / physical unit of field
TTYPE10 = 'SIGNIF_U'           / The name of this column
TFORM10 = 'E       '           / data format of field: 4-byte REAL
TUNIT10 = 'sigma   '           / physical unit of field
TTYPE11 = 'SIGNIF_V'           / The name of this column
TFORM11 = 'E       '           / data format of field: 4-byte REAL
TUNIT11 = 'sigma   '           / physical unit of field
TTYPE12 = 'IST_MAG_WHITE'      / Mean instrumental magnitude of source
determined from white filter data. This is
repeated below for other filters.
TFORM12 = 'E       '           / data format of field: 4-byte REAL
TUNIT12 = 'magnitudes'         / physical unit of field
TTYPE13 = 'IST_MAG_UVW2'       / The name of this column
TFORM13 = 'E       '           / data format of field: 4-byte REAL
TUNIT13 = 'magnitudes'         / physical unit of field
TTYPE14 = 'IST_MAG_UVW1'       / The name of this column
TFORM14 = 'E       '           / data format of field: 4-byte REAL
TUNIT14 = 'magnitudes'         / physical unit of field
TTYPE15 = 'IST_MAG_U'          / The name of this column
TFORM15 = 'E       '           / data format of field: 4-byte REAL
TUNIT15 = 'magnitudes'         / physical unit of field
TTYPE16 = 'IST_MAG_V'          / The name of this column
TFORM16 = 'E       '           / data format of field: 4-byte REAL
TUNIT16 = 'magnitudes'         / physical unit of field
TTYPE17 = 'MAGERR_WHITE'       / 1 sigma error on instrumental magnitude
repeated below for other filters below
TFORM17 = 'E       '           / data format of field: 4-byte REAL
TUNIT17 = 'magnitudes'         / physical unit of field
TTYPE18 = 'MAGERR_UVW2'        / The name of this column
TFORM18 = 'E       '           / data format of field: 4-byte REAL
TUNIT18 = 'magnitudes'         / physical unit of field
TTYPE19 = 'MAGERR_UVW1'        / The name of this column
TFORM19 = 'E       '           / data format of field: 4-byte REAL
TUNIT19 = 'magnitudes'         / physical unit of field
TTYPE20 = 'MAGERR_U'           / The name of this column
TFORM20 = 'E       '           / data format of field: 4-byte REAL
TUNIT20 = 'magnitudes'         / physical unit of field
TTYPE21 = 'MAGERR_V'           / The name of this column
TFORM21 = 'E       '           / data format of field: 4-byte REAL
TUNIT21 = 'magnitudes'         / physical unit of field
TTYPE22 = 'QFLAG_WHITE'        / quality flag set if `bad` pixels affected
 source in any white filter exposure. This is
repeated below for other filters
```

```
TFORM22 = 'B       '           / data format of field: BYTE
TUNIT22 = 'none    '           / physical unit of field
TTYPE23 = 'QFLAG_UVW2'         / The name of this column
TFORM23 = 'B       '           / data format of field: BYTE
TUNIT23 = 'none    '           / physical unit of field
TTYPE24 = 'QFLAG_UVW1'         / The name of this column
TFORM24 = 'B       '           / data format of field: BYTE
TUNIT24 = 'none    '           / physical unit of field
TTYPE25 = 'QFLAG_U '           / The name of this column
TFORM25 = 'B       '           / data format of field: BYTE
TUNIT25 = 'none    '           / physical unit of field
TTYPE26 = 'QFLAG_V '           / The name of this column
TFORM26 = 'B       '           / data format of field: BYTE
TUNIT26 = 'none    '           / physical unit of field
TTYPE27 = 'CFLAG_WHITE'        / confusion flag set if source confused with
another source in any white filter exposure.
 (repeated below for other filters)
TFORM27 = 'B       '           / data format of field: BYTE
TUNIT27 = 'none    '           / physical unit of field
TTYPE28 = 'CFLAG_UVW2'         / The name of this column
TFORM28 = 'B       '           / data format of field: BYTE
TUNIT28 = 'none    '           / physical unit of field
TTYPE29 = 'CFLAG_UVW1'         / The name of this column
TFORM29 = 'B       '           / data format of field: BYTE
TUNIT29 = 'none    '           / physical unit of field
TTYPE30 = 'CFLAG_U '           / The name of this column
TFORM30 = 'B       '           / data format of field: BYTE
TUNIT30 = 'none    '           / physical unit of field
TTYPE31 = 'CFLAG_V '           / The name of this column
TFORM31 = 'B       '           / data format of field: BYTE
TUNIT31 = 'none    '           / physical unit of field
TTYPE32 = 'EFLAG_WHITE'        / extension flag set if source sigificantly
extended relative to PSF in any white filter
(repeated below for other filters)
TFORM32 = 'B       '           / data format of field: BYTE
TUNIT32 = 'none    '           / physical unit of field
TTYPE33 = 'EFLAG_UVW2'         / The name of this column
TFORM33 = 'B       '           / data format of field: BYTE
TUNIT33 = 'none    '           / physical unit of field
TTYPE34 = 'EFLAG_UVW1'         / The name of this column
TFORM34 = 'B       '           / data format of field: BYTE
TUNIT34 = 'none    '           / physical unit of field
TTYPE35 = 'EFLAG_U '           / The name of this column
TFORM35 = 'B       '           / data format of field: BYTE
TUNIT35 = 'none    '           / physical unit of field
TTYPE36 = 'EFLAG_V '           / The name of this column
TFORM36 = 'B       '           / data format of field: BYTE
TUNIT36 = 'none    '           / physical unit of field
TTYPE37 = 'MAJ_WHITE'          / mean FWHM of major axis derived from white
 band detections. (repeated below for other
 filters)
TFORM37 = 'E       '           / data format of field: 4-byte REAL
TUNIT37 = 'pixels  '           / physical unit of field
TTYPE38 = 'MAJ_UVW2'           / The name of this column
TFORM38 = 'E       '           / data format of field: 4-byte REAL
```

```
TUNIT38 = 'pixels  '           / physical unit of field
TTYPE39 = 'MAJ_UVW1'           / The name of this column
TFORM39 = 'E       '           / data format of field: 4-byte REAL
TUNIT39 = 'pixels  '           / physical unit of field
TTYPE40 = 'MAJ_U   '           / The name of this column
TFORM40 = 'E       '           / data format of field: 4-byte REAL
TUNIT40 = 'pixels  '           / physical unit of field
TTYPE41 = 'MAJ_V   '           / The name of this column
TFORM41 = 'E       '           / data format of field: 4-byte REAL
TUNIT41 = 'pixels  '           / physical unit of field
TTYPE42 = 'MIN_WHITE'          / mean FWHM of minor axis derived from white
band detections. (repeated below for other
filters)
TFORM42 = 'E       '           / data format of field: 4-byte REAL
TUNIT42 = 'pixels  '           / physical unit of field
TTYPE43 = 'MIN_UVW2'           / The name of this column
TFORM43 = 'E       '           / data format of field: 4-byte REAL
TUNIT43 = 'pixels  '           / physical unit of field
TTYPE44 = 'MIN_UVW1'           / The name of this column
TFORM44 = 'E       '           / data format of field: 4-byte REAL
TUNIT44 = 'pixels  '           / physical unit of field
TTYPE45 = 'MIN_U   '           / The name of this column
TFORM45 = 'E       '           / data format of field: 4-byte REAL
TUNIT45 = 'pixels  '           / physical unit of field
TTYPE46 = 'MIN_V   '           / The name of this column
TFORM46 = 'E       '           / data format of field: 4-byte REAL
TUNIT46 = 'pixels  '           / physical unit of field
TTYPE47 = 'STD_MAG_U'          / brightness of object converted to standard
 Johnson system. Repeated below for B and V
 filters where applicable.
TFORM47 = 'E       '           / data format of field: 4-byte REAL
TTYPE48 = 'STD_MAG_V'          / The name of this column
TFORM48 = 'E       '           / data format of field: 4-byte REAL
EXTNAME = 'SRCLIST '           / The name of this table
END
```

7.6 Running the OM data processing chain

The datapackage received by the investigator contains OM data which normally do not neccessitate further processing for the purpose of calibration. However, a script of the pipeline processing batch will be made available to the investigator who can rerun the pipeline calibration in an interactive way. A third possibility for the investigator is to run default pipeline processing chains called *omichain* for the image mode and *omfchain* for the fast mode.

The *omichain* and *omfchain* tasks run the calibration part of the OM pipelines in their simplest form. They process all ODF components without any interaction from the user. The calibrated ouput is accurate enough to perform quick look analysis. The tasks *omichain* and *omfchain* can be run after specifying the path to the odf directory of interest, generating an associated calibration index file (CIF) using the task *cifbuild* and specifyng the path to this newly created calibration index file. This can be performed on the command line as follows.

```
setenv SAS_ODF 'path to the odf of interest'
```

```
cifbuild analysisdate='now' append=no calindexset='ccf.cif' \
category='XMMCCF' ccfpath='path to a repository of calibration files' \
fileglob='*.ccf|*.CCF' fullpath=yes ignorecategory=no masterindex=no \
masterindexset='ccf.mif' observationdate='2000-08-10' recurse=no \
withccfpath=yes withmasterindexset=no

setenv SAS_CCF ccf.cif

omproc &
```

7.7 Analysing OM data

Before analysing OM data, the user is advised to inspect the raw OM image files. These images are stored in the first extension of the file and can be displayed with ds9. It is often more convenient to visualize the entire field of view. The SAS task *rudi5mosaic* can be used to combine the five exposures of an OM default configuration into one single image. This allows a quick inspection of the entire OM field of view.

OM images show several artefacts. These artefacts are generallly only visible when viewing the OM images in a high contrast and in logarithmic display. Artefacts can however affect the accuracy of a measurement, e.g. by increasing the background level. The following artefacts may be present in the raw OM images.

- Out of time events: very bright sources with count rates of several tens of counts per second show a strip of events along the readout direction. These events correspond to out of time photons arriving while the detector is read out.
- Smoke rings: bright sources generate smoke ring like structures, which are located radially away from the center of the field of view. These rings are caused by photons which are reflected from the detector entrance window back onto the detector photocathode.
- Fixed pattern noise: raw OM images show a modulo 8 regular pattern originating from imperfections of the event centroiding algotithm in the the instrument electronics. This modulo 8 pattern is removed during image analysis by the task *ommodmap*.
- Straylight features: straylight is caused by a chamfer in the OM detector housing which reflects light from outside the OM FOV onto the active detector area. This reflected celestial background light sums up onto a circular area with an increased background rate in the center of the OM field of view. The light coming from bright sources and reflected by the chamfer can also produce loop like structures in an OM images. These loops can degenerate into long streaks depending on the source positions.

Once the OM calibrated data products (e.g images or timeseries) have been produced and inspected, the investigator can proceed with the scientific analysis of his data. This is generally performed with the support of specialized analysis software such as IRAF, MIDAS, IDL ... Hence, for scientific analysis, the user is refered to the documentation associated with these packages. However, when analysing the optical monitor data, the user shall be aware of the following points.

1. The OM operates in photon counting mode but images are accumulated on board. This has some implication for the selection of good time intervalls (GTI) from the housekeeping data.

In particular, part of an exposure cannot be excluded. Either the full exposure is selected or not.

2. Contrary to the X-ray instrument, an OM exposure does not provide direct energy information except when grisms are selected.

3. A flatfield response of unity is assumed.

4. Images show a fixed pattern noise which results from the centroiding algorithm to one eighth of the OM CCD pixel size.

5. Coincidence losses can occur which depend on the source brightness and on the CCD framerate. The framerate itself depends on the selected configuration of the detector science and tracking windows. If two or more events are located close to each other within the same CCD readout frame, they are detected and counted as one event. Also when photon splashes overlap, a mislocation is given by the centroiding algorithm. Another consequence of coincidence losses is that event depletion occurs at high count rates around the central source position.

6. The OM U and V filters do not form a proper photometric system. During calibration, a zero point correction and a color transformation have to be applied.

7. The OM point spread function (PSF) has wide wings. Careful selection of photometric extraction aperture is required. These aperture have to be wide enough to get full the full PSF.

8. Straylight features complicate the background subtraction in some cases, specially when the target star lays in or close to a straylight feature. We currently recommend to use extraction radii of 12 subpixels.

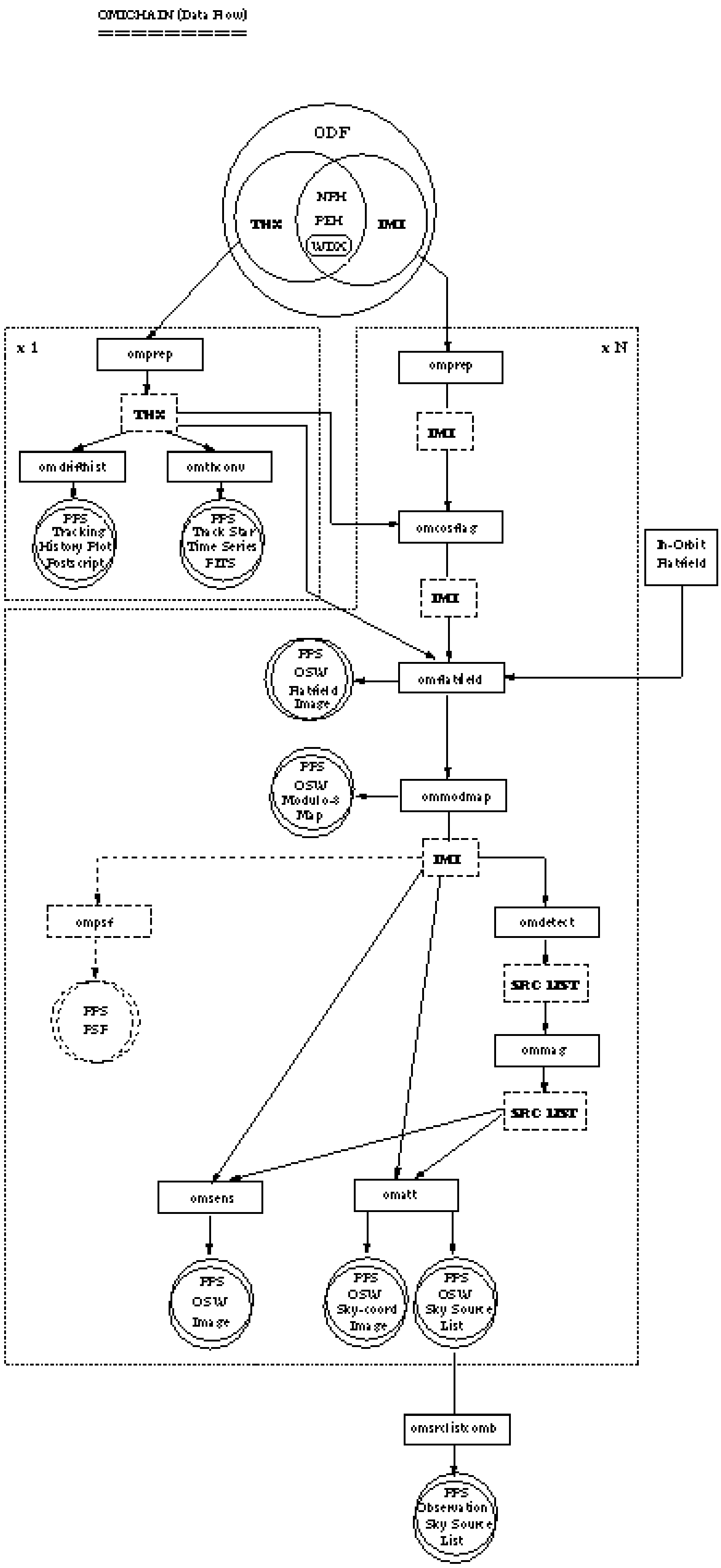


Figure 7.1: Pipeline processing of data acquired in the OM imaging mode

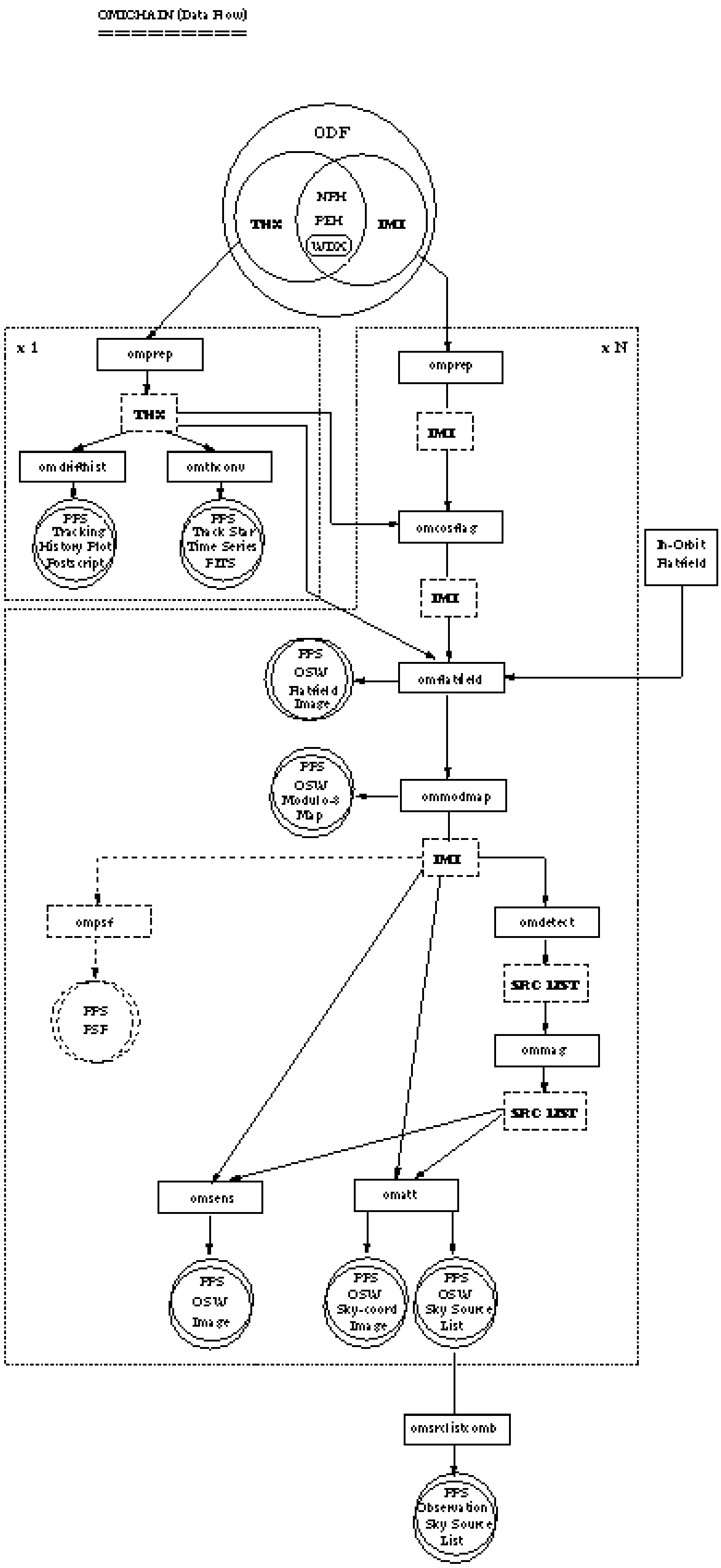


Figure 7.2: Pipeline processing of data acquired in the OM fast mode

Bibliography

[1] Christian Erd, Phillipe Gondoin, David Lumb, Rudi Much, Uwe Lammers, and Giuseppe Vacanti. Calibration Access and Data Handbook. Technical Report XMM-PS-GM-20, ESA/SSD, Jan 14 2000. Found at the URL: `http://xmm.vilspa.esa.es/calibration/docs/general/calhb.ps.gz`.