XMM-Newton

User's Guide
to the
XMM-Newton Science Analysis System

Issue 4.1
Edited by: N. Loiseau

20.08.2007

Based on contributions from M. Ehle, A.M.T. Pollock, A. Talavera, C. Gabriel, B. Chen, J. Ballet, K. Dennerl, M. Freyberg, M. Guainazzi, A. Ibarra, M. Kirsch, N. Loiseau, L. Metcalfe, E. Ojero, J. Osborne, W. Pietsch, R. Saxton, M. Smith, E. Verdugo

Revision history

Revision number	Date	Revision author	Comments
Issue 4.1	20.08.2007	N. Loiseau	Official release - SAS v7.1 update
Issue 4.0	02.08.2006	N. Loiseau	Official release - SAS v7.0 update
Issue 3.2	16.12.2005	N. Loiseau	Official release - SAS v6.5 update
Issue 3.1	16.12.2004	N. Loiseau	Official release - SAS v6.1 update
Issue 3.0	29.03.2004	N. Loiseau	Official release - SAS v6.0 update
Issue 2.1	17.03.2003	N. Loiseau	Official release - SAS v5.4 update
Issue 2.0	22.11.2002	E. Verdugo	Official release - SAS v5.3 update
Issue 1.0	10.07.2000	Ph. Gondoin	Official release

Contents

List of Figures

List of Tables

Glossary

Acronym	Explanation
AO	Announcement of Opportunity
AMS	Archive Management Subsystem
CAL	Calibration Access Layer
CAMEX	Multichannel readout chip
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 Mean 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)
JD	Julian Date
LSF	Line-spread Function
MIME	Multipurpose Internet Mail Extensions
MOS	Metal Oxide Semi-conductor CCD camera
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
OSW	Observed Science Window
QLA	Quick Look Analysis
PHA	Pulse Height Analyser (uncalibrated spectral channel)
PI	Pulse-Invariant (calibrated spectral channel)
PPS	Pipeline Processing Subsystem
PSF	Point-Spread Function
RA	Right Ascension, α(J2000)
RGA	Reflection Grating Assembly (of the RGS)
RGS	Reflection Grating Spectrometer
RPE	Relative Pointing Error
SAS	Science Analysis System
SOC	XMM-Newton Science Operations Centre
SSC	Survey Science Centre
ToO	Target of Opportunity
UHB	XMM-Newton Users' Handbook
UT	Universal Time
WCS	World Coordinate System
XDA	XMM-Newton Data Archive
XID	X-ray source identification (by the SSC)
XMM-Newton	X-ray Multi-Mirror Mission
XSA	XMM-Newton Science Archive

1 Introduction

1.1 Scope of this manual

The XMM-Newton Scientific Analysis Subsystem (SAS) is the main analysis tool for XMM-Newton data reduction. The purpose of this document is to guide the investigators in analysing the data corresponding to an observation.

The data corresponding to an observation performed with XMM-Newton comprise among a large number of products the fundamental ones: the raw and calibrated event files collected by the instruments on-board XMM-Newton: the European Photon Imaging Cameras (EPIC), the Reflection Grating Spectrometers (RGS) and the Optical Monitor (OM).

1.2 Required software environment

The full analysis of XMM-Newton 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 SAS, which has been developed by a team of scientists located at the ESA XMM-Newton SOC and at the XMM-Newton Survey Science Centre (SSC). Informations on how to download and install the SAS software package are available on the XMM-Newton web server at:

`http://xmm.esac.esa.int/sas`

Improvements of current SAS version over older versions can be checked in the ”SAS release notes” in this same URL address.

1.3 SAS installation

Once you have downloaded the specific SAS tar.gz archives for your operating system the installation is quite straightforward and platform independent

(see: http://xmm.esac.esa.int/sas/7.1.0/installation).

Choose a directory (e.g. /top_dir) to place the SAS software and unpack the tar.gz archives. The unpacking process will create a subdirectory named

xmmsas_YYYYMMDD_HHMM

which identifies uniquely each SAS release by the date (YYYYMMDD) and time (HHMM) of its production. For example, for the SAS 7.1.0 version, this is xmmsas_20070708_1801. In SAS terms, this is called the release manifest.

Move on to this subdirectory and execute the script

./configure_install

which if completed successfully, will produce two additional shell scripts within the same directory, named respectively setsas.sh for the Bourne/Bash/Korn shells and setsas.csh for the csh/tcsh shells. These are the scripts to be used later on to initialize the SAS environment.

The configure_install script makes some additional sanity checks on the installation environment as for example checks for the proper installation of the perl binary in the /usr/local/bin directory, which is required to run some SAS perl procedures.

1.4 More information on the web

The SAS user is assumed to be familiar with the basic characteristics and operation modes of the scientific instruments on board XMM-Newton. This information is available in the XMM-Newton Users Handbook [1] at:

http://xmm.esac.esa.int/external/xmm_user_support/documentation/

Another important document that can be found in the same URL address is the XMM-Newton Data File Handbook [8] which provides a detailed description of the data distributed to the users from the XMM-Newton Archive (XSA, at http://xmm.esac.esa.int/xsa) [2]. This handbook has information on the format, structure and content of individual XMM-Newton files.

The Calibration Access and Data Handbook [3] describes the calibration files and the calibration access routines which are used by SAS to access these files. The document is accessible from the XMM-Newton calibration web portal at:

http://xmm.esac.esa.int/external/xmm_sw_cal/calib/

Detailed descriptions of individual SAS tasks can be accessed through the SAS on-line documentation web page at:

http://xmm.esac.esa.int/sas/current/doc/

or by clicking the “Help Task” button of the (main) SAS Graphical User Interface (GUI).

Some of the analysis process described here are also synthesized in the SAS threads at:

http://xmm.esac.esa.int/sas/current/documentation/threads/

1.5 Structure of the document

The structure of the present document is as follows:

- Chapter 2 introduces the investigator to the analysis of XMM-Newton data. It provides a brief description of the XMM-Newton observation and calibration files. It outlines the analysis steps to produce calibrated event files and to extract scientific products.

- Chapter 3 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 4 describes the successive SAS analysis steps required prior to the scientific analysis of EPIC final data products.

- Chapter 5 describes the initial SAS analysis steps to obtain RGS final data products prior to their scientific analysis.

- Chapter 6 describes the SAS reduction and calibration of OM data which can be afterwards be analyzed using standard astronomical software packages.

1.6 Reporting problems

Problems in using the SAS shall be reported submitting an Observation Report (OR) at:
http://xmm.esac.esa.int/sas/7.1.0/feedback

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. Workaround will be provided to the user whenever is possible. Any inquiry on the status of the Observation Reports, as well as any general question about the XMM-Newton data analysis, must be issued through the XMM-Newton HelpDesk:
`http://xmm.esac.esa.int/sas`

2 Analysis of XMM data: an overview

2.1 The XMM-Newton data files

The full set of data corresponding to an XMM-Newton observation is basically composed of:

- the observation datafiles (ODF): these files include the raw event science files from the EPIC, RGS and OM instruments, the instrument housekeeping files, the radiation monitor files and the spacecraft files.
- all the data products generated by the Pipeline Processing Subsystem (PPS) at the XMM-Newton Survey Science Center http://xmmssc−www.star.le.ac.uk/newpages/pipe_top_ext.html.

The XMM-Newton data can be retrieved via ftp at the XMM-Newton Science archive:

http://xmm.esac.esa.int/xsa/

To analyse the data the observer needs to have available also the current calibration files (CCF), which are available on the XMM-Newton web server at:

http://xmm.esac.esa.int/external/xmm_sw_cal/calib/

The XMM-Newton observation data files as well as the current calibration files (CCF), conform to the Flexible Image Transport (FITS) standard. A detailed description of their format, structure, naming convention and contents is provided in the XMM-Newton Data Files Handbook and in the Calibration Access and Data Handbook, both to be found under:

http://xmm.esac.esa.int/external/xmm_sw_cal/calib/documentation.shtml

2.2 Successive steps in analysing XMM-Newton data. Threads and Watchout items

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

1. preparation of the analysis environment, including access to the XMM-Newton data and calibration files, SAS analysis system and associated software packages,
2. inspection of the XMM-Newton 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 meta-tasks. The user may skip this step and only use for analysis the calibrated event lists provided as pipeline products (the step is necessary in case of better calibration available or the pipeline products been generated with an older SAS version than the current one),
5. data screening, source detection and extraction of scientific products including images, spectra and time series. Again, much of this work is done also by the pipeline, e.g. generation of images in different spectral ranges as well as source detection in

the case of EPIC data, also spectra extraction in first and second order from the RGS data,

6. generation of auxiliary response files and redistribution matrices,

7. scientific analysis of the final XMM-Newton products.

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

Users are strongly recommended to first have a look at the **Data analysis threads** for the data reduction of each instrument, and to check regularly the **Watchout items and evergreen tips**, both under:

http://xmm.esac.esa.int/sas

2.3 Starting a SAS session

2.3.1 Seting up the basic SAS environment

To start a new SAS session the SAS environment has to be initialized by means of one of the shell scripts created during the SAS software installation process (see section § 1.3). The way this is done depends on the shell used. Assuming the working directory is /my_work, and that SAS software installation was done at:

```
/top_dir/xmmsas_YYYYMMDD_HHMM
```

the script can be executed as follows:

```
. /top_dir/xmmsas_YYYYMMDD_HHMM/setsas.sh  [sh, bash, ksh]

or

source /top_dir/xmmsas_YYYYMMDD_HHMM/setsas.csh [csh, tcsh]
```

We recommend not to work in the directory where SAS is installed, i.e. not to mix the working directory, /my_work, with the SAS installation directory, /top_dir/xmmsas_YYYYMMDD_HHMM.

The setsas.[c]sh script takes care of all necessary environment setup required to work with SAS. However, to start working, SAS has to be told where are the observation data files(ODFs) and the calibration files (CCFs).

2.3.2 Telling SAS where to find the ODF and CCF files

The SAS_ODF environment variable must be defined as the directory where are placed the ODFs of the observation to be analyzed. This can be done as follows:

```
export SAS_ODF=/path/to/<ODF> [sh, bash, ksh]

setenv SAS_ODF /path/to/<ODF> [csh, tcsh]
```

The SAS_CCFPATH environment variable must be defined as the directory where all SAS calibration files are stored,

```
export SAS_CCFPATH=/path/to/<CCF> [sh, bash, ksh]

setenv SAS_CCFPATH /path/to/<CCF> [csh, tcsh]
```

We recommend to store all CCFs in a shared accessible directory so any other SAS user may refer to them as well.

2.3.3 cifbuild

A detailed description of the way to access the calibration data files appropriated for a certain data set, and the parameters that need to be specified, can be found in the section 3.11. The SAS command `cifbuild` reads the observation date from the observation start date of the observation to be analyzed, as reported by the ODF Access Layer (OAL), to select the default calibration files. Alternatively, the user can specify the observation date of the calibration to be used by modifing the `cifbuild` task parameters.

All calibration files are kept in a repository accessible to the SAS user for building the calibration database.

Calling `cifbuild` with no parameters assumes the following default parameters:

```
cifbuild calindexset = 'ccf.cif' withccfpath = no usecanonicalname = no \
         ccfpath = '.' recurse = no fileglob = '*.ccf|*.CCF' fullpath = no \
         withobservationdate = no observationdate = 'now' analysisdate = 'now'\
         category = 'XMMCCF' ignorecategory = no masterindex = no \
         withmasterindexset = no masterindexset = 'ccf.mif' append = no
```

The output of `cifbuild` is the calibration index set (ccf.cif in the example above).

Once this command ends successfully, SAS has to be pointed to the ccf.cif file by means of the $SAS_CCF environment variable, as follows:

```
export SAS_CCF=/path/to/ccf.cif [sh, bash, ksh]

setenv SAS_CCF /path/to/ccf.cif [csh, tcsh]
```

2.3.4 odfingest

Afterwards the information extracted from the instruments housekeeping files and from the calibration database have to be appended to the summary file, within ODF. This task is done through the SAS command odfingest, resulting in an extended SAS summary file which will be needed for the subsequent data processing.

Then run:

odfingest

without any additional option. The resulting summary file is produced in the working directory and named as

```
'REV'_'OBS'_SCX00000SUM.SAS
```

where 'REV' is the revolution number and 'OBS' is the observation number of the data.

The ODF Summary File is an ASCII file, which can be easily edited. It contains a lot of information necessary to run SAS on the data, including the path to the ODF files (changing the location of the ODF files makes therefore necessary to edit the file accordingly). This file provides all the information about the ODFs. SAS needs to be pointed to it by substituting the current definition of the $SAS_ODF environment variable by the full definition of the summary file, as follows

```
export SAS_ODF=<Summary_file> [sh, bash, ksh]

setenv SAS_ODF  <Summary_file> [csh, tcsh]
```

where in <Summary_file> we need to write the full path and file name of the summary file.

Write access is not required for the ODF and CCF directories.

Now you are ready for starting the work on the XMM-Newton data. The configuration setting as described above is the recommended one, with a working directory per observation, on which all products of the SAS tasks will be put (unless explicitly stated differently). At this point the working directory should have two files: the calibration index file and the ODF summary file.

All the steps listed above could be contained in a setup script to be called the first time one works on a dataset. The following is an example for such a script to be run on csh/tcsh shells, giving the path to the directory containing the ODF data as the only parameter,

```
source /top_dir/xmmsas_YYYYMMDD_HHMM/setsas.csh
setenv SAS_ODF $1
setenv SAS_CCFPATH /path/to/CCF
cifbuild
setenv SAS_CCF $PWD/ccf.cif
odfingest
set sumfile=`ls -1 *SUM.SAS`
setenv SAS_ODF $PWD/$sumfile
```

2.4 Running SAS: Command Line or Graphical User Interface

There are two independent ways of operating the SAS, either using the command line as discussed in section § 2.6, or using the SAS Graphical User Interface, hereafter called the GUI (discussed in detail in § 3).

The SAS GUI main interface is called by typing `sas` on the command line, but also almost every SAS task has its own GUI, which can be individually started by typing `task -d`.

There are advantages and disadvantages in using command line or GUI, depending on the type of work the user has to do and his/her level of proficiency in using SAS. Intensive interactive work on a single dataset is generally more efficiently done with the GUI. For beginners the GUI is generally better, because it eases the understanding of the parameters. For processing of large amounts of data the scripting capability of SAS, putting together a large number of line commands is of great advantage and extremely efficient. After some tests every user should be able to make use of the best of both SAS worlds.

2.5 Selecting a SAS task

A list of SAS tasks appears when typing the command `sas` which calls the SAS Graphical User Interface (GUI). A SAS task can simply be called (through its GUI) by double-clicking on its name. The use of the SAS GUI is explained in § 3. 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.esac.esa.int/sas/current/doc/

For detailed descriptions of the tasks, the user is referred 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 1.

Table 1: Some frequently used SAS tasks

Task name	Task description
arfgen	Generates an EPIC ancillary response file
calview	An interactive viewer of the XMM-Newton calibration database
cifbuild	Creates a Calibration Index File for a given observation date
edetect_chain	EPIC source detection metatask
emproc	Fully configurable “pipeline” processing of EPIC MOS observation data files (ODFs)
epproc	Fully configurable “pipeline” processing of EPIC PN ODFs
evselect	Filters event lists and extracts images, spectra, timeseries and histograms
odfbrowser	Interactive graphical viewer of an XMM-Newton observation
omichain	Fully configurable “pipeline” processing of the OM imaging mode ODFs
omfchain	Fully configurable “pipeline” processing of the OM fast mode ODFs
omgchain	Fully configurable “pipeline” processing of the OM grism mode ODFs
rgsproc	Fully configurable “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

2.6 Running a SAS task from the command line

For a better understanding of the SAS using command lines, the SAS user can 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. The task parameter handling system (see: http://xmm.esac.esa.int/sas/current/doc/param/index.html) is capable of understanding complex expressions for allowable values, in addition to setting default values. Furthermore, parameters can have child-parameters, depending on its value, and this makes possible the implicit setting of controlling parameters, if a child parameter is activated on the command line.

Example:

evselect –xcolumn=RAWX implies –withimageset=yes (unless –withimageset=NO explicitly set)

This represents large advantages at the time of writing scripts, which can be much more compact than in the past. However, ”old” scripts have to be revised to avoid setting implicitly undesired parameters.

Each SAS task supports the following command line options:

- -h provides information about the command line options.
- -d launches the GUI for this specific task
- -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 (code) level set the verbosity level of the task. The value ranges from 0 to 10 and the larger the verbosity value, the more messages will be produced. Default value is 1, a value of 4 is recommended for a good verbose control. Larger values are only recommended for debugging purposes.
- -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. The variable `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 *SUM.SAS.

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 ...`

  Blanks, commas, or colons 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 list of directories separated by commas, 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.

2.7 General settings affecting all SAS tasks

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 or to a SAS summary file (*SUM.SAS). If a name is specified via -o and SAS_ODF is defined the former takes precedence over the latter. This environment variable is only relevant to tasks making use of the ODF Access Layer.
- SAS_TRACER determines the debug level of the libraries.
- SAS_MEMORY_MODEL determines how the internal memory is used by the data access (through the Data Access Layer, DAL). There are basically two options for the memory model: High Memory (*high*) and HighLow Memory (*highlow or low*) model. In the *high* model every time a dataset is opened it is kept entirely in memory and all subsequent operations are performed on the memory-loaded dataset. In the *low* model the data is loaded into memory only when their access is needed (when opening a dataset only their attributes are kept in memory). While the *high* model should bring higher performance, its use implies normally a high memory consumption, which can lead to swapping, producing a poorer performance. Therefore for machines with less than about 1GB RAM it is recommended to use the *low* model. The default value for SAS_MEMORY_MODEL is *high*
- SAS_VERBOSITY determines the debug level of the task. The value ranges from 0 to 10 with increasing verbosity level. The default value is 1, a value of 4 is recommended for getting a very communicative SAS, larger values should be only used for debugging purposes.

SAS tasks produce error messages at levels called message, 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.

2.8 SAS input and output 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-Newton 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 detailed description of the XMM-Newton files (including format, content, keywords ...etc), the user is referred to the XMM-Newton Data File Handbook [8] which can be accessed from the XMM-Newton web page at:

http://xmm.esac.esa.int/external/xmm$_{u}ser_{s}upport/documentation/$

It is important to note that the naming convention of XMM-Newton files is slightly different from that historically used for FITS files. Table 2 provides a handy "conversion table" between the two nomenclatures.

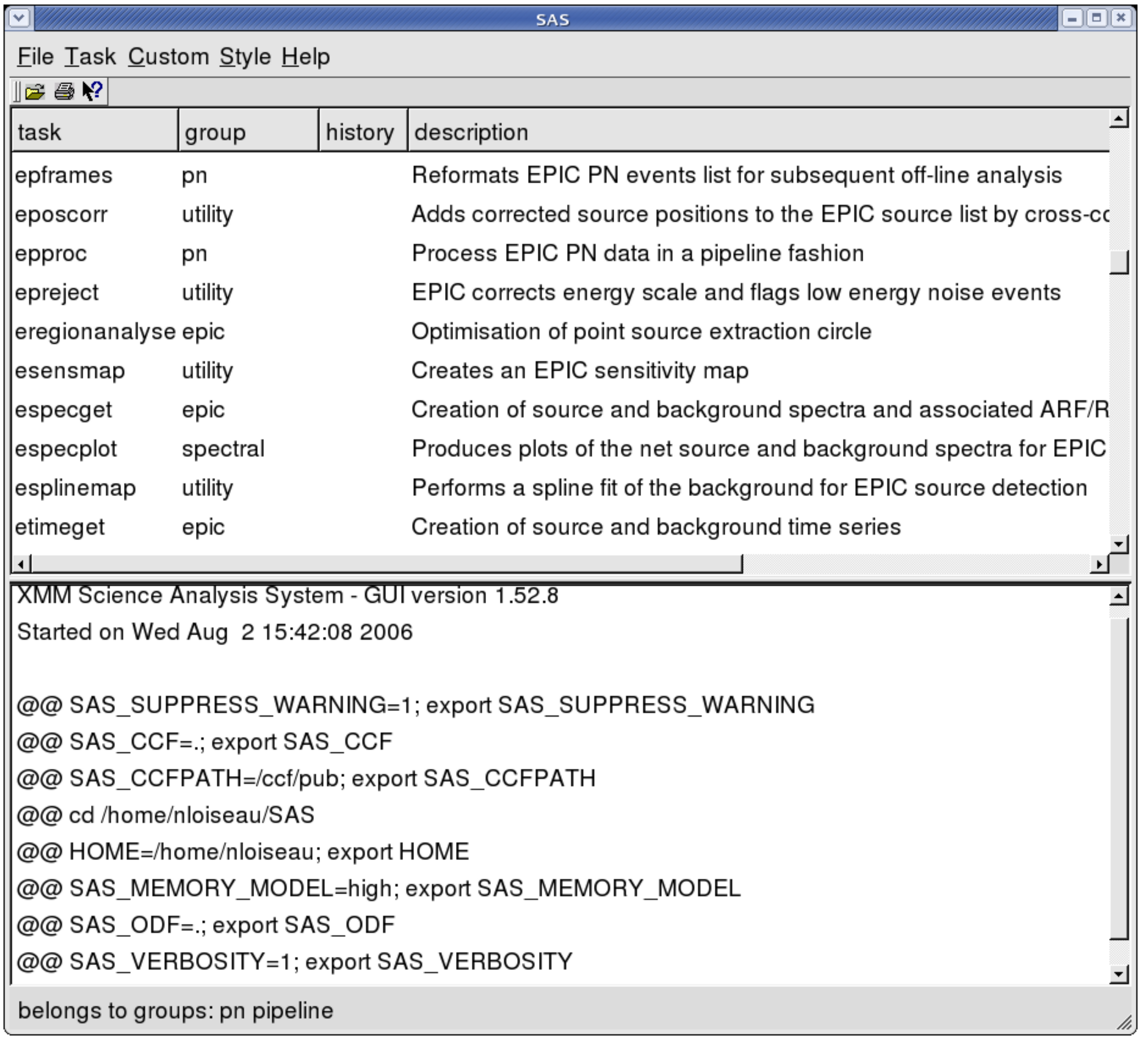
FITS name	**XMM-Newton dataset name**
Header	Header
Primary Header Data Units/Array	First Data Block/Primary Array
Extension	Data block
Image	Array
Binary Table	Table
Keyword	Attribute
Column/Rows	Column/Rows

Table 2: "Conversion table" between FITS and XMM-Newton dataset constituents naming convention

3 SAS graphical user interface

3.1 Getting started

The SAS can be operated using the command line. 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 main GUI is started by typing `sas` on the command line. This should display a main window similar to the following:



task	group	history	description
epframes	pn		Reformats EPIC PN events list for subsequent off-line analysis
eposcorr	utility		Adds corrected source positions to the EPIC source list by cross-cc
epproc	pn		Process EPIC PN data in a pipeline fashion
epreject	utility		EPIC corrects energy scale and flags low energy noise events
eregionanalyse	epic		Optimisation of point source extraction circle
esensmap	utility		Creates an EPIC sensitivity map
especget	epic		Creation of source and background spectra and associated ARF/R
especplot	spectral		Produces plots of the net source and background spectra for EPIC
esplinemap	utility		Performs a spline fit of the background for EPIC source detection
etimeget	epic		Creation of source and background time series

```
XMM Science Analysis System - GUI version 1.52.8
Started on Wed Aug  2 15:42:08 2006


@@ SAS_SUPPRESS_WARNING=1; export SAS_SUPPRESS_WARNING
@@ SAS_CCF=.; export SAS_CCF
@@ SAS_CCFPATH=/ccf/pub; export SAS_CCFPATH
@@ cd /home/nloiseau/SAS
@@ HOME=/home/nloiseau; export HOME
@@ SAS_MEMORY_MODEL=high; export SAS_MEMORY_MODEL
@@ SAS_ODF=.; export SAS_ODF
@@ SAS_VERBOSITY=1; export SAS_VERBOSITY
```

belongs to groups: pn pipeline

Figure 1: SAS GUI window

The upper half of the window is a task browser, which allows the user 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.

3.2 A quick tour

1. **Preconfigure the SAS session**

   Although most of the settings can be created / modified within a SAS session, we recommend to start the SAS with all environment variables already defined, eg via a startup script as described under 2.3.

2. **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 &
   ```

   The command sas launches the main SAS GUI.

3. **Preferences: (Re-)Configure CCF and ODF directories, verbosity level and memory model**

   Using the preferences dialog in the file Menu of the SAS GUI it is possible to set or reset the environment variables pointing to the ODF Summary file or directory, to the CCF index file, to the directory(ies) containing the calibration files and to the working directory. In addition the **verbosity** level (determining the debug level of the tasks, 0=low, 10=maximum) can be set as well as the **memory model** to be used (see section 2.7).

4. **Select a task**

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

5. **Configure parameters**

   The parameter dialog (see section 3.5) allows to configure the parameters of a task before running it.

   If the task requires input data they can be selected by popping up a dataset browser (see section 3.7). This allows to browse through the file hierarchy and select SAS datasets or individual components, such as tables, arrays or columns.

6. **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 3.6) from the Task menu.

7. **Examine the log**

   The lower half of the main GUI window is a log pane (see section 3.8), which displays messages from the task. Check that the task ran successfully. If the task produced any warning or error messages (see section 3.9), those will be displayed in a dialog box.

8. **Run another task**

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

   One 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 finishes (see section 3.6).

9. **Save or print the log**

   When the processing session is finished, the log (see section 3.8) provides a record of the steps carried out, with all parameters and any messages received from the tasks. This log can be saved for future reference. Scripts can also be constructed from those single calls.

   The log is automatically saved in the file `sas_log` in the current directory. It can also be printed out from the 'File' menu or *tool bar*.

   The environment variable SAS_SUPPRESS_WARNING (by default set to a value of 1 in the SAS initialization) set the maximum number of occurrences a specific warning shall appear in the log (and in the corresponding warning windows if run from the GUI). If set to the value 0 only the summary for each warning will be written to the log instead. In all cases a summary line specifying the number of occurrences for each issued warning is written to the logfile.

10. **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 3.10 for further details.

3.3 Selecting an ODF

At the start of a SAS session, one normally selects the ODF to be processed, setting the environment variable SAS_ODF pointing either to the directory containing the ODF files, or to the ODF Summary File (as explained under 2.3). Choosing 'Preferences' from the 'File' menu pops up a dialog into which the path of another ODF directory (or ODF Summary File) can be entered.

The ODF directory can be specified either by an absolute path (starting with '/') or as a path relative to the directory in which the SAS GUI was started. The '..' button pops up a dataset browser (see section 3.7) which allows to search for the directory and select it.

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

3.4 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 3.5) to pop-up, so that parameters can be entered and the task started.

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

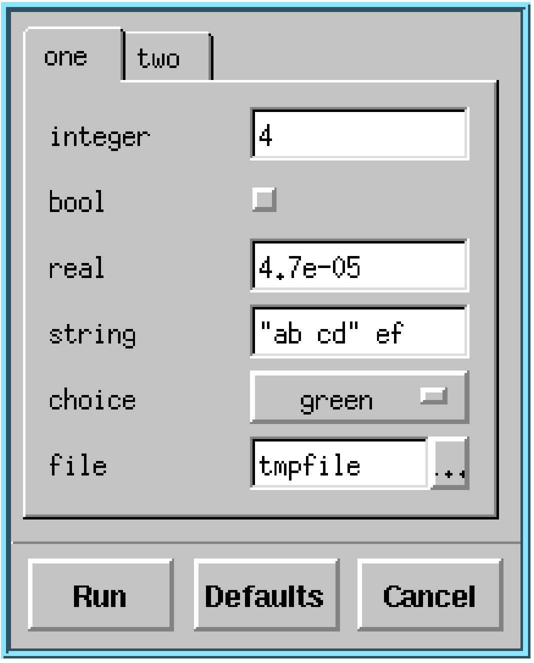
When a task is selected by double clicking, a number appears in the 'history' column. This number is incremented as successive tasks are run. Clicking on the 'history' column heading sorts the list in the order in which the tasks were run (putting them in the bottom or the top of the list, depending if one or two clicks); 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.

3.5 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 browser by pressing the '..' button (see section 3.7).



integer 4
bool □
real 4.7e-05
string "ab cd" ef
choice green
file tmpfile

Figure 2: SAS parameter dialog

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 an option ”Revert to defaults” to reset all the parameters of all tasks to their defaults.

3.6 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, a parameter dialog can be opened to configure the parameters in preparation for running the next task. By pressing the 'Run' button before the previous task has finished, one gets 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.

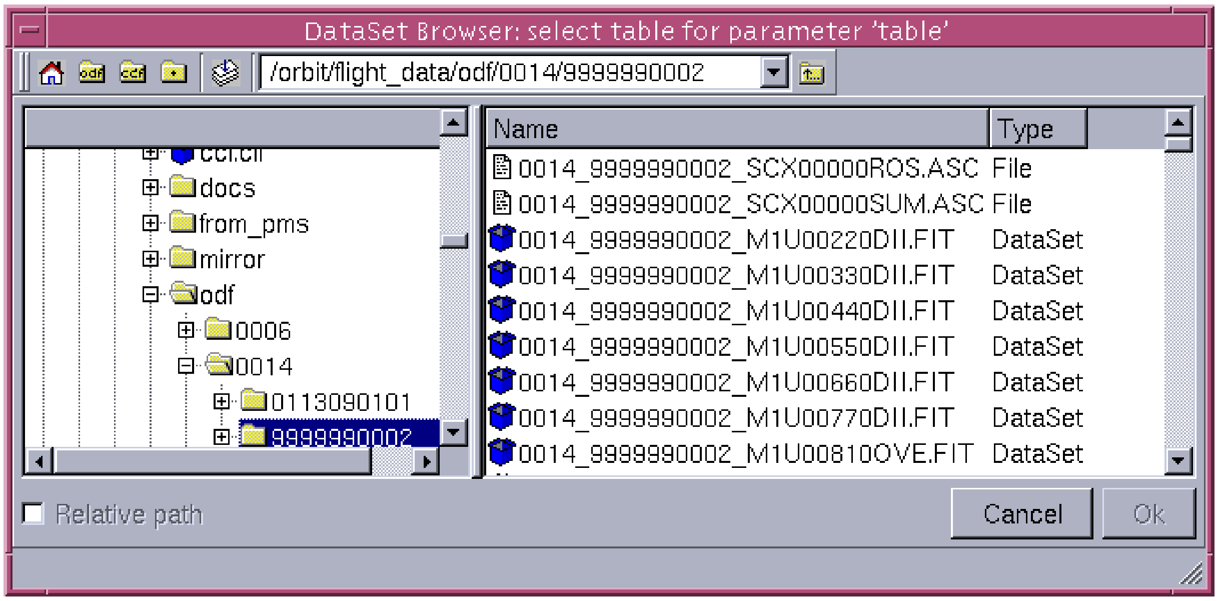
When a task is started, or queued for later execution, the parameter values are saved along with the task. Consequently, editing the parameters and queuing a second instance of a task will not affect the parameters of a task that is already running or queued.

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.

3.7 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. 3). The browser can also be invoked from the 'File' menu or tool-bar, simply to inspect data, without making a selection.



DataSet Browser: select table for parameter 'table'

/orbit/flight_data/odf/0014/9999990002

- cci.cif
- docs
- from_pms
- mirror
- odf
  - 0006
  - 0014
    - 0113090101
    - 9999990002

Name	Type
0014_9999990002_SCX00000ROS.ASC	File
0014_9999990002_SCX00000SUM.ASC	File
0014_9999990002_M1U00220DII.FIT	DataSet
0014_9999990002_M1U00330DII.FIT	DataSet
0014_9999990002_M1U00440DII.FIT	DataSet
0014_9999990002_M1U00550DII.FIT	DataSet
0014_9999990002_M1U00660DII.FIT	DataSet
0014_9999990002_M1U00770DII.FIT	DataSet
0014_9999990002_M1U00810OVE.FIT	DataSet

☐ Relative path

Figure 3: 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 a rectangle around the name. The right pane shows information on the highlighted item.

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 divide the right pane in two 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.

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 top of the browser window are four buttons which can be used to quickly locate some commonly used directories. These are the user's home directory, the ODF directory, the CCF directory, and the current working directory from which the SAS GUI was started. Then next button to the right is useful when selecting parameters for a task. Pressing this button will show or hide a window, that lists all currently selected entries.

The (non-editable) pull-down menu allows to quickly jump to a previous visited item. After the button to jump to a higher level (e.g. parent directory) from SAS 6.0 on a new button is found for refreshing the contents. This allows to see new files created during the session.

3.8 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. It is possible 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 3.9), 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 not only be printed using the 'Print log...' command on the 'File' menu, but also saved as a script for further use by the 'Save as script' option in the same menu.

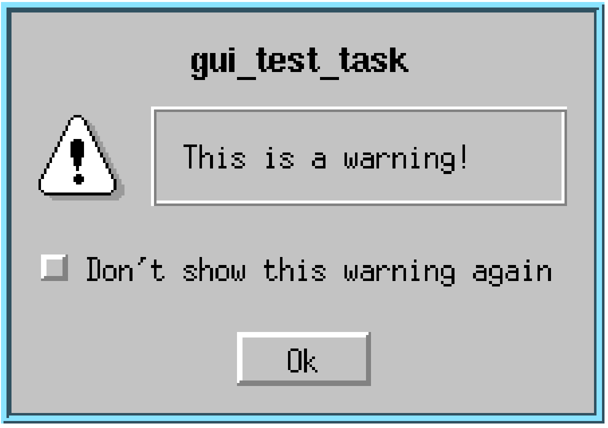
3.9 Errors and warnings

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 3.8) and may also pop up a dialog to alert the user.

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

3.10 Using on-line help

The SAS GUI provides 3 levels of help:


gui_test_task

This is a warning!

□ Don't show this warning again

Figure 4: warning

- 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 (each SAS task is fully documented, its documentation tree including several different sections, like algorithm, input, output, parameters, etc).

The on-line documentation request will start Netscape if it is not already running before selecting 'Help'. The help menu passes the URL of the SAS documentation to Netscape.

3.11 Accessing calibration data

The XMM−Newton 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 appropriate way 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 SAS task `calview` allows the user to visualize calibration quantities (also derived ones). 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. 5). 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.

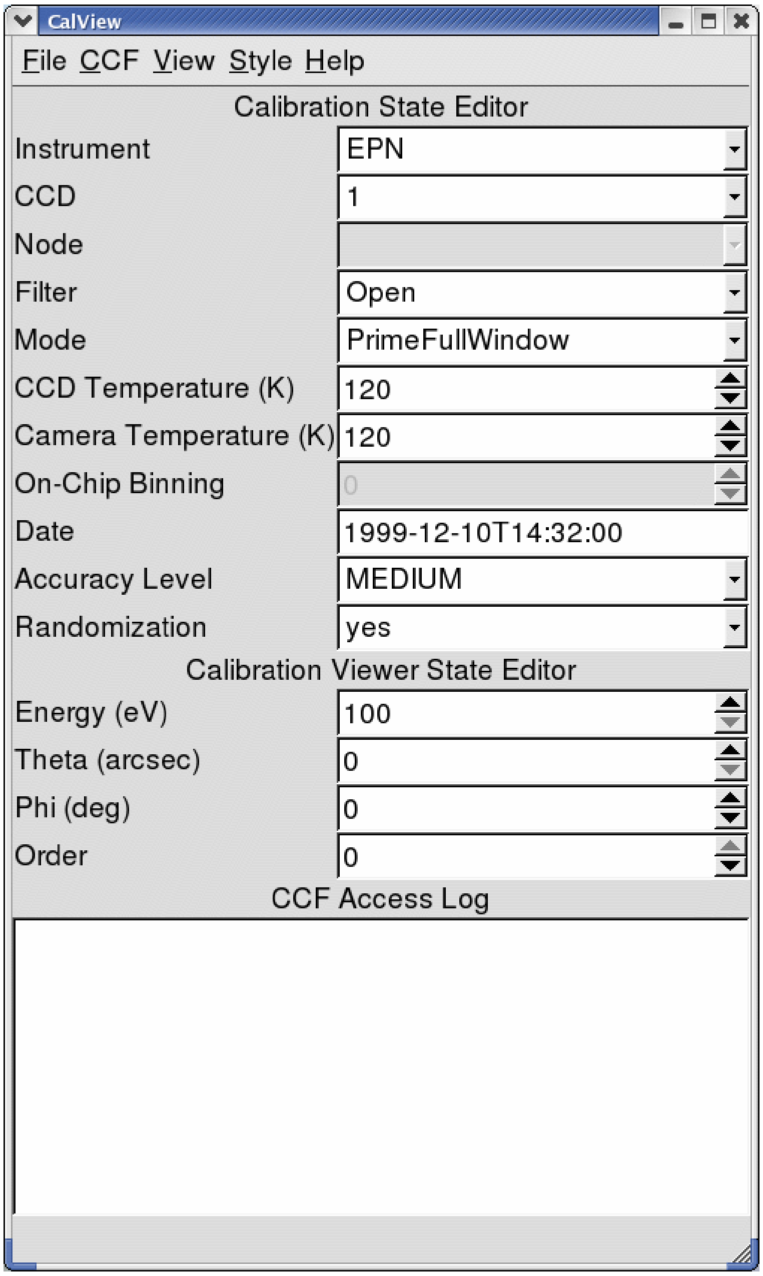
It is possible to point the cal to a CCF directory, to a CCF index file (see task `cifbuild` for information on how to generate one), or add individual CCF components to an existing list.

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−Newton Calibration Handbook [1]

The XMM−Newton 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 is 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`. In this way not only the time dependency of calibration is taken into account (eg. different tables of hot pixels corresponding to different dates), but also an old data calibration can be reproduced.

Normally `cifbuild` would be used to construct a CIF based on the CCF constituents available to the user. These are kept in the directories indicated either by the environment variable SAS_CCFPATH or in the parameter `ccfpath` specified in the call to `cifbuild`. 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.



Calibration State Editor

Instrument	EPN
CCD	1
Node	
Filter	Open
Mode	PrimeFullWindow
CCD Temperature (K)	120
Camera Temperature (K)	120
On-Chip Binning	0
Date	1999-12-10T14:32:00
Accuracy Level	MEDIUM
Randomization	yes

Calibration Viewer State Editor

Energy (eV)	100
Theta (arcsec)	0
Phi (deg)	0
Order	0

CCF Access Log

Figure 5: Calibration state server (calview window)

4 Analysis of EPIC camera data

In this chapter we describe the analysis of XMM-Newton data sets obtained with the European Photon Imaging Camera (EPIC).

In § 4.1 the structure of the EPIC related observation data files (ODF) is discussed.

The EPIC products created by the SSC standard Pipeline Processing are listed in § 4.2.

§ 4.3 explains in a rather technical way how an EPIC ODF can be re-processed up to the level of generating calibrated event lists. Such re-processing should be performed by the investigator only if the calibration has improved significantly between the time of the pipeline processing and the current time. While users may want to specify different time selections or different energy selections to those used in the pipeline, in general the SSC pipeline offers the best analysis possible, as it is the result of years of experience tuning the individual task parameters. Only improvements in the calibration files and in some of the tasks would require re-processing of data processed with old SAS versions. Some of the analysis process described here are also synthesized in the SAS threads at:

http://xmm.esac.esa.int/sas/current/documentation/threads/

All following sections in this chapter assume that calibrated event lists exists (either from the pipeline or from a re-processing) and demonstrate how to create data products like images, spectra, rate-curves and source lists for further scientific analysis.

4.1 The EPIC data package

After reception of the XMM-Newton observation data package from the XMM-Newton Science Archive (XSA), the guest investigator wishing to analyse EPIC data is advised to verify the EPIC related content of the data package and to inspect the EPIC pipeline products.

4.1.1 The EPIC Observation Data Files

The ODF names for the EPIC data will look something like:

- mmmm_iiiiiijjkk_aabeeeccfff.zzz, where
  - mmmm: revolution number
  - iiiiiijjkk: observation number
  - aa: detector ID (M1 - MOS1, M2 - MOS2, PN - pn)
  - b: flag for scheduled (S), unscheduled (U) observations, or (X) for general use files
  - eee: exposure number within the observation
  - cc: CCD identifier
  - fff: data identifier (see Table 3)
  - zzz: Format (FITS - FIT, ASCII - ASC)

Data Identifier	Contents
IME	Event list for individual CCDs, imaging mode
RIE	Event list for individual CCDs, reduced imaging mode
CTE	Event list for individual CCDs, compressed timing mode (MOS)
TIE	Event list for individual CCDs, timing mode
BUE	Event list for individual CCDs, burst mode (pn)
AUX	Auxiliary file
CCX	Counting cycle report (auxiliary file)
HBH	HBR buffer size, non-periodic housekeeping (MOS)
HCH	HBR configuration, non-periodic housekeeping
HTH	HBR threshold values, non-periodic housekeeping (MOS)
PEH	Periodic housekeeping (MOS)
PTH	Bright pixel table, non-periodic housekeeping (MOS)
DLI	Discarded lines data (pn)
PAH	Additional periodic housekeeping (pn)
PMH	Main periodic housekeeping (pn)

Table 3: EPIC ODF data file identifiers (see Table 35 in the XMM-Newton Data Files Handbook)

4.1.2 The EPIC MOS Observation Data Files

The most relevant files for scientific analysis of a MOS observation are (depending on the observing mode) the imaging mode and/or the timing mode event list files together with the auxiliary file providing a detailed description of each frame recorded during the exposure. The structure of these files is described in the XMM-Newton Data Files Handbook. One imaging mode event list file is produced per CCD by the "Full Frame" and "Partial Window" instrumental modes. In the "Partial Window" mode, the imaging area of the central chip (CCD 1) is reduced to 100×100 ("Small Window" mode) or 300×300 ("Large Window" mode) pixels. In timing mode, the central chip collects spatial info only in one dimension (the X axis) whereas the Y axis is a measure of the detection time. The six surrounding CCDs always produce imaging mode event list files using the 600×600 pixel area.

4.1.3 The EPIC pn Observation Data Files

The most relevant files for scientific analysis of a pn observation are (depending on the observing mode) the imaging mode, the timing mode or the burst mode event list files together with the auxiliary file providing a detailed description of each frame recorded during the exposure. One imaging mode event list file is produced per CCD by the "Full Frame" and "Large Window" instrumental modes. Only CCD 4 is active in "Small Window", "Timing" or "Burst" mode. Hence these modes produce event list data only for a single chip. The other chips do not collect any data. In timing and burst mode, CCD 4 collects spatial info only in one dimension (the X axis) whereas the Y axis is a measure of the detection time.

4.2 The EPIC pipeline products

The Pipeline Processing produces quite a number of useful products which allow a first look at the data. With each grouping of the pipeline products (Table 4) there is an HTML (.HTM extension) file which lists the associated files and gives a short description of those files. The HTML file names are of the following format:

- PPiiiiiijjkkAAAAAA000_0.HTM, where
  - iiiii: proposal number
  - jj: target ID - target number in proposal
  - kk: exposure ID - exposure number for target
    NOTE: The ten-digit combination of iiiiiijjkk is the observation number and is used repetitively throughout the file nomenclature
  - AAAAAA: Group ID (Table 4)

The data file names are of the form (see Table 36 in the XMM-Newton Data Files Handbook, [8]):

- PiiiiiijjkkaablllCCCCCCnmmm.zzz, where
  - iiiiiijjkk: observation number
  - aa: detector, M1 - MOS1, M2 - MOS2, PN - pn, CA - for files from the CRSCOR (crosscorrelation) group
  - b: S for scheduled observation, U for unscheduled, X for files from the CRSCOR group (and any product that is not due to a single exposure)
  - lll: exposure number
  - CCCCCC: file identification (Table 4)
  - n: exposure map band number, unimportant otherwise for EPIC data
  - mmm: source number
  - zzz: file type (e.g., PDF, PNG, FTZ, HTM)
    - ASC: ASCII file, use Netscape, other web browser, or the “more” command
    - ASZ: gzipped ASCII file
    - FTZ: gzipped FITS format, use e.g. ds9, xmmselect, fv
    - HTM: HTML file, use web browser
    - PDF: Portable Data Format, use Acrobat Reader
    - PNG: Portable Networks Graphics file, use web browser
    - TAR: TAR file

Group ID	File ID	Contents	File Type
PPSOBS	OBX000SUMMAR	Observation Data File Summary (also copied into INDEX.HTM)	HTML
	EPX000SUMMAR	EPIC Processing Summary	HTML
	PPSSUM	PPS processing summary	HTML
	SCRLOG	PPS script log	gzipped ASCII
	ATTTSR	Attitude Time Series	gzipped FITS
ESKYIM	IMAGE_8	Sky image 0.2 - 12.0 keV	gzipped FITS
	IMAGE_1	Sky image 0.2 - 0.5 keV	gzipped FITS
	IMAGE_2	Sky image 0.5 - 2.0 keV	gzipped FITS
	IMAGE_3	Sky image 2.0 - 4.5 keV	gzipped FITS
	IMAGE_4	Sky image 4.5 - 7.5 keV	gzipped FITS
	IMAGE_5	Sky image 7.5 - 12.0 keV	gzipped FITS
	IMAGE	EPIC observation image	gzipped FITS, HTML
EANCIL	EXPMAP8	Exposure map 0.2 - 12.0 keV	gzipped FITS
	EXPMAP1	Exposure map 0.2 - 0.5 keV	gzipped FITS
	EXPMAP2	Exposure map 0.5 - 2.0 keV	gzipped FITS
	EXPMAP3	Exposure map 2.0 - 4.5 keV	gzipped FITS
	EXPMAP4	Exposure map 4.5 - 7.5 keV	gzipped FITS
	EXPMAP5	Exposure map 7.5 - 12.0 keV	gzipped FITS
	EXSNMP	Exposure sensitivity map	gzipped FITS
EEVLIS[1]	MIEVLI	MOS imaging mode event list	gzipped FITS
	PIEVLI	pn imaging mode event list	gzipped FITS
	TIEVLI	EPIC timing mode and also pn burst mode event list	gzipped FITS
EPICEXP	FBKTSR	EPIC flare background time-series	gzipped FITS
ESRLIS	EBLSLI	Box-local detect source list	gzipped FITS
	EBMSLI	Box-map detect source list	gzipped FITS
	EMSRLI	Max-like detect source list	gzipped FITS
	OBSMLI	Summary source list	gzipped FITS, HTML
CRSCOR	FCHART	Finding chart	PDF
	ROSIMG	Overlay EPIC & ROSAT image	PDF
	SNNNNN[2]	EPIC Source cross-correlation results	gzipped FITS
	DNNNNN[2]	Catalog descriptions	HTML
	FNNNNN[2]	EPIC FoV cross-correlation results	gzipped FITS

[1]:Files for only those modes which were active will be included

[2]:NNNNN - Alphanumeric ID

Table 4: Pipeline Processing data files relevant for EPIC

4.3 Running the EPIC pipeline processing

The data package received by the investigator contains EPIC pn and EPIC MOS calibrated event lists generated by the SSC pipeline processing. As the SSC pipeline is the result of years of experience tuning the individual task parameters, in general no re-processing is needed. The script of the pipeline processing (SCRCLOG) is made available to the investigator to check how the pipeline processing created EPIC products. Only if the calibration has improved significantly between the time of the pipeline processing and the current time, the investigator should reprocess the EPIC data by running the default pipeline processing meta tasks `emproc` and `epproc` (also known as `epicproc`) or the chain tasks `emchain` and `epchain`. These tasks run the calibration part of the EPIC MOS and pn pipelines in their simplest form processing all ODF components without any interaction from the user and creating calibrated EPIC event lists. As these tasks are scripts they can easily be edited and they allow to produce several additional output files (mainly for diagnostic purposes). The processing tasks are run by specifying the path to the ODF directory and applying the following task commands

```
setenv SAS_CCF <path>/ccf.cif
setenv SAS_ODF <path>/odf
emproc / emchain
```

or/and

```
epproc / epchain
```

Input files are looked for in the directory entered via 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.

At the beginning of the `epicproc` meta tasks, the task `atthkgen` creates an attitude history file, which will be used by `attcalc`.

The `withinstexpids` and `selectccds` parameters allow to run `epicproc` on a subset of the data files (select a single exposure, a single instrument or a single CCD). The processing of timing modes for cases in which the pointing is offset with respect to the source, requires that the user specifies the additional parameters `withsrccoords`, `srcra`, and `srcdec`. For pn optionally one can use `timingsrcposition` in RAWY coordinates. In most other cases the default values are adequate.

The pipeline processing creates in the current directory output data sets with the following naming convention:

Every output file name starts with a string composed by one or more of the following parts, each separated by an underscore character (_):

- rrrr: the revolution number
- iiiiiijjkk: the observation identifier (see § 4.2)
- EPN or EMOS1 or EMOS2: the instrument name
- blll: the exposure identifier. For instance: S001 (see § 4.2)

- cc: the CCD number
- nn: the node number

These parts are hierarchically structured, so that, say, any data set name that contains the exposure identifier will also contain the revolution number, the observation identifier, and the instrument name.

Additional strings indicate the contents of the data set:

- AttHk: attitude housekeeping
- Gti: GTI
- AuxGti: GTI based on the auxiliary file
- FrmGti: GTI created by `emframes` or `epframes`
- HkGti: housekeeping GTI
- Evts: the data set contains an event list
- ImagingEvts: imaging mode event list
- TimingEvts: timing mode event list
- BurstEvts: burst mode event list
- Badpixels: Badpixel files created by `badpixfind` and used by `badpix`
- Temp: a temporary data set, usually removed

All the pipeline processing tasks inside the proc meta tasks can also be run individually by the user.

4.3.1 Running the EPIC MOS processing meta-task

A description of the functional organigram for the EPIC MOS processing is given in Figs. 6 and 7. The `emproc` meta task concatenates all first-level EPIC MOS tasks to produce calibrated event lists for all selected exposures.

The main subroutine (Fig. 6) loops over all selected exposures and instruments (MOS1/MOS2) present in the input directory. It 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 (from the CCF or produced internally). In a first step Fig. 7) it loops over all CCD/nodes, calling in sequence:

1. `emframes` on the auxiliary file, the event file and the external GTI file (if any), creating a frame file as expected by `emevents` and a CCD/node specific GTI file which will be re injected in the final call to `evselect`.
2. `badpix` on the event list, adding the BADPIX extension. If a bad pixels file exists, it is used instead of the CAL calls for the non-uplinked bad pixels.

3. `emevents` on the event list, the offset/variance file and the frame file, creating a new event list which will be propagated through `attcalc` and `emenergy` to `evlistcomb`.

4. `gtialign` on the external GTI file and the event file, then the task `gtimerge` to merge the resulting aligned GTI and the CCD/node specific GTI.

5. `attcalc` on the new event list, filling the X/Y columns.

6. `emenergy` on the new event list, filling the FLAG, PHA and PI columns.

Then `evselect` is called on the resulting event list(s) applying (by default) the destructive filter selection "(#XMMEA_EM) && (FLAG & 0x762a0000) == 0". Note that in case of `emchain`, it is not true that "(#XMMEA_EM)" is applied: here events flagged as OUT_OF_FOV and REJECTED_BY_GATTI are kept in the list (as they are useful for background assessments and flare screening, respectively). For a description of the event attribute based selection, refer to the documentation of the SAS package `evatt`.

All the event list files created (one per CCD/node) are merged by `evlistcomb`, creating one event list per mode (IMAGING, TIMING). Finally `evselect` is called on the resulting events list(s), with "(CCDNR == $node$ccd) && GTI(merged GTI file,TIME)" for all CCD/nodes.

In the EPIC MOS imaging mode, the EVENTS binary table of the calibrated event list file contain 12 columns i.e TIME, RAWX, RAWY, DETX, DETY, X, Y, PHA, PI, FLAG, PATTERN and CCDNR. DETX and DETY are the event position in the focal plane array. X and Y are the event position in sky coordinates. PHA is the pulse analyser channel and PI the pulse independent channel. CCDNR is the CCD number. For a description of the FLAG column, see the documentation of the SAS package `evatt`. The PATTERN definition is given in the documentation of the task `emevents` (see also Fig. 12).

In the EPIC MOS timing mode, the EVENTS binary table of the calibrated event list file contains only 7 columns: the spatial coordinates RAWY, DETX, DETY, X, Y are not present as (in this mode) only one axis (RAWX) contains spatial information whereas the other axis is a measure of the arrival time of the event.

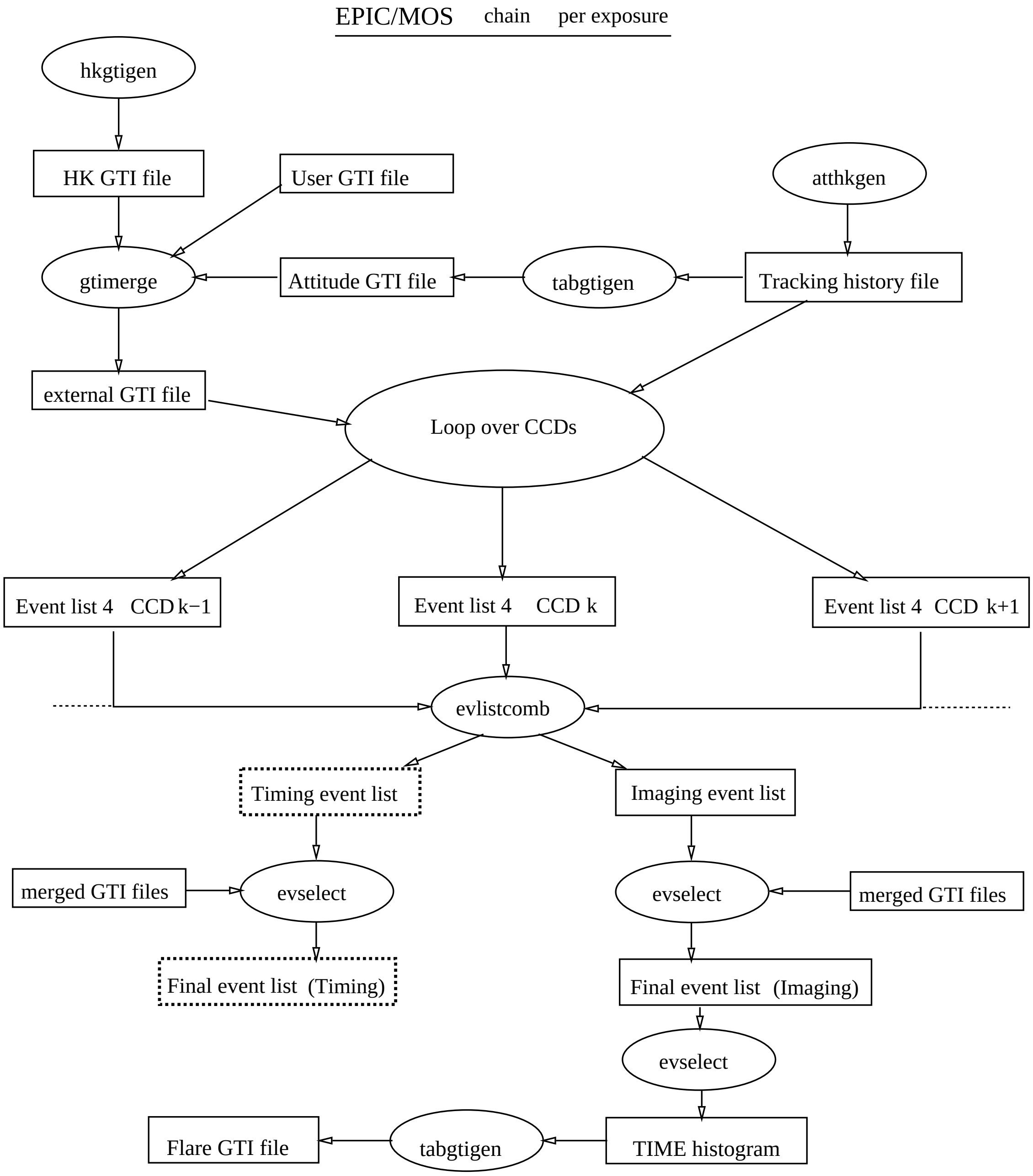


Figure 6: Organisation of the EPIC-MOS chain: merging the event lists. The files in boldly dashed boxes are used (or produced) if they exist. The files in simply dashed boxes are options of the individual tasks not used in the current chain.

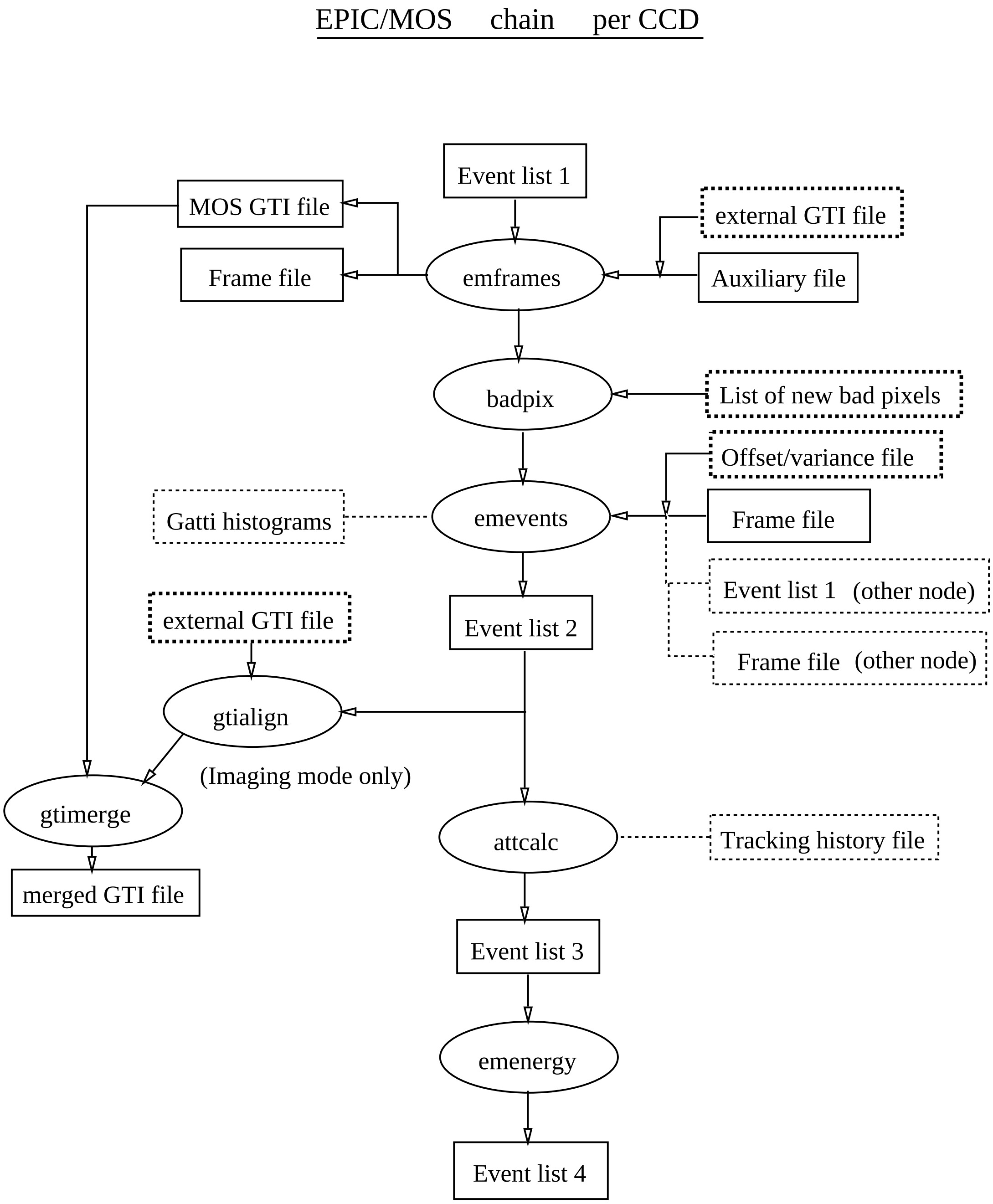


Figure 7: Organisation of the EPIC-MOS chain at CCD/node level with file inputs. Same conventions as in Fig. 6.

4.3.2 Running the EPIC pn processing meta-task

The EPIC pn meta task `epproc` concatenate all first-level pn tasks to produce calibrated event lists. The pn processing is sketched out in Fig. 8. The main subroutine (`epframes`, `badpixfind`, `badpix`, `epevents` and `attcalc`) creates one event list for a single exposure and for all selected CCDs from all the relevant ODF material and bad pixel lists calling in sequence:

1. `epframes` to process a CCD, exposure and datamode specific ODF file, creating the output raw event list and GTI data set,
2. `badpixfind` to find new bad pixels,
3. `badpix` to process the raw event list, adding the BADPIX extension,
4. `epevents` to process the event list file, flagging trailing events, performing split events pattern recognition, CTI and gain correction to create the calibrated event list,
5. `attcalc` to calculate the X and Y sky coordinates.

Finally, making use of the task `evlistcomb`, the CCD specific data sets are merged into a single event list. `evselect` selects all those events arriving in good time intervals, applies (by default) the destructive filter selection “(#XMMEA_EP) && (PI > 150 || PI < -150)” and writes the output file. For a description of the event attribute based selection, refer to the documentation of the SAS package `evatt`.

In the EPIC pn imaging mode, the EVENTS binary table of the calibrated event list file contains 14 columns i.e TIME, RAWX, RAWY, DETX, DETY, X, Y, PHA, PI, FLAG, PATTERN, PAT_ID, PAT_SEQ and CCDNR. For a description of the PAT* columns, refer to the documentation of the task `epevents`. The definition of the PATTERN values is also visualized in Fig. 13.

In the EPIC pn timing mode, the EVENTS binary table of the calibrated event list file contains only 9 columns: the spatial coordinates DETX, DETY, X, Y are not present as (in this mode) only one axis (RAWX) contains spatial information whereas the other axis is a measure of the arrival time of the event.

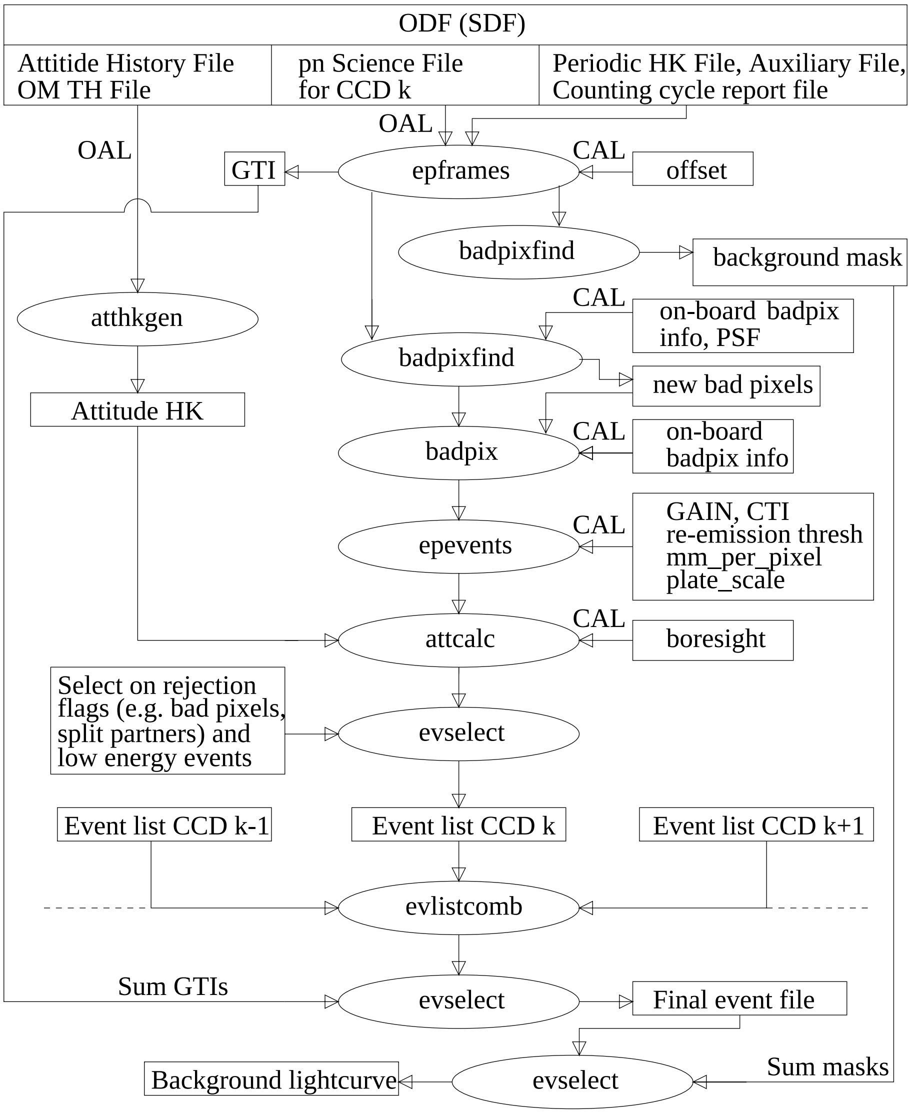


Figure 8: Pipeline processing of the EPIC pn observation data.

4.3.2.1 Improving the quality of EPIC pn data: `epreject`

The SAS task `epreject` allows the investigator to extend possible EPIC pn data analysis down to the softest energies (120 eV for imaging and 200 eV for timing mode data, respectively). Hence its main applications will be in the study of soft X-ray sources. Based on the intended type of science, the user needs to decide if the additional data reduction steps are worthwhile to be applied for a specific EPIC pn exposure.

The task `epreject` performs different actions for imaging and timing mode data which are described below in § 4.3.2.1.1 and § 4.3.2.1.2.

4.3.2.1.1 Correcting EPIC pn imaging mode data

The current implementation supports only data taken in fullframe mode, but might be extended to the other imaging modes (extended fullframe, large window, small window) if this task is found to be useful. It comprises the two following separated subtasks: [1]

1. Correcting the energy scale in specific pixels:
   The purpose of this subtask is to correct accidental shifts in the energy scale for specific pixels and thus to improve the spectral quality throughout the full EPIC pn bandwidth. The reason for this shift are incorrect values in the offset map (caused by high-energy particles hitting these pixels during the offset map calculation), which is usually computed onboard before each observation.

   There are two methods for retrieving the affected pixels and the corresponding energy shifts: (i) directly from the transmitted offset map or (ii) from the spatial and spectral distribution of the transmitted events. The first method is more straightforward and reliable than the second one, but it requires that the offset map is available, which is not always the case.

   (i) The offset map, if available, contains for each pixel the adu value which was subtracted from all events in the particular pixel before transmitting this information to ground. As it contains both correct and incorrect offsets, the main goal is to distinguish between both cases. This distiction is possible under the (realistic) assumption that the generic offsets of all pixels (not affected by electronic effects during readout) are very similar and that the areas where incorrect offsets were computed occur in isolated patches. In this case, all the information necessary for correcting accidental shifts in the energy scale can be extracted from the offset map in a straightforward way (Dennerl et al. 2004, Proc. SPIE 5488, pp 61-72). This is the default mode of `epreject`, if the offset map is available.

   (ii) If the offset map is not available, then the information about the affected pixels and the corresponding energy shifts can be (approximately) reconstructed from the event file in the following way: an image is accumulated from all events which have a raw amplitude of 20 adu (the lowest amplitude transmitted). This image is then analysed for the occurrence of bright patches, and from the brightness of each patch

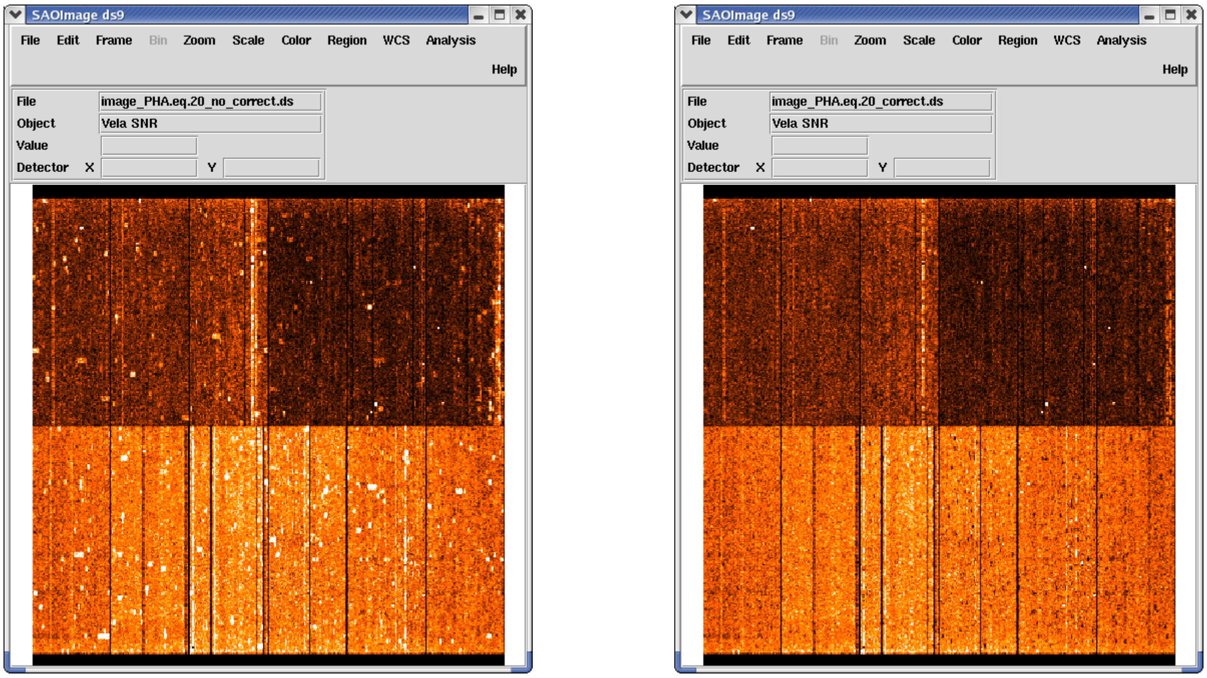
[1] **Important note:** both `epreject` subtasks require the presence of all events down to raw amplitudes PHA=20 adu.

the corresponding energy shift is reconstructed, by the method described in [25]. This energy shift is then applied to all events in the corresponding patch.

This method relies on the quality of the 20 adu image, which depends on the exposure time of the observation. In particular for short exposures, the presence of Poissonian noise in the 20 adu images limits the sensitivity for spotting the bright patches and deriving the appropriate energy correction. A parameter is supplied to the user for adjusting this sensitivity: the parameter `sigma` specifies the minimum significance which a block of four consecutive pixels along readout direction must have in order to trigger the energy correction for events in this block.

Tests indicate that `sigma`=4.0 (the default) is a good choice for short (∼ 5 ks) exposures; for longer exposures this parameter can be increased (to ∼ 5 − 6 for more than 20 ks). It is recommended to control the results by accumulating an image below 20 adu after this task, as this image shows then all the pixels where an offset shift was applied. In most cases, the default setting should provide a reasonable result.

A comparison of images accumulated at the lowest transmitted amplitudes, before and after applying `epreject`, should directly show the improvement in image quality which can be obtained by this subtask (Fig. 9).



File image_PHA.eq.20_no_correct.ds
Object Vela SNR

File image_PHA.eq.20_correct.ds
Object Vela SNR

Figure 9: Images of all events with PHA = 20 adu in detector coordinates, before (left) and after (right) correcting the energy scale in specific pixels, but before suppressing noise events. The color scale extends from 0 to 50 events per pixel. Black patches indicate areas where no 20 adu information is available.

Current restriction:

After a successful run of this task, the next step in data processing would be to remove events which have received amplitudes below 20 adu by this correction, in order to ensure a homogeneous treatment of all events. Otherwise, e.g., doubles with one component below 20 adu might show up in the recombined event list, although they would be considered as singles if they had occurred at a different location at the detector. Unfortunately, SAS v6.5 does not support removal of events before `epevents` is applied. Thus, the only technique currently available to reject events below 20 adu is to filter the recombined event list for PHA ≥ 20. This will at least remove all single events below this threshold. It is planned to include the possibility of removing events immediately after `epreject` in a future SAS release.

Practical tips:

If one is only interested in correcting the energy scale in specific pixels, without using the noise suppression described in the next section, then the easiest application is to run `epchain` with the standard settings, adding just the '`epreject`' part (with the possibility to change the parameter `sigma`, if desired):

```
epchain runepreject=Y [sigma=..] \
        noiseparameters="0 0 0 0 0 0 0 0 0 0 0 0 0"
```

The parameter `noiseparameters="0 0 0 0 0 0 0 0 0 0 0 0 0"` is necessary in order to prevent any screening of events which may have been caused by detector noise.

If an iteration should become necessary (for example in order to find an optimum value for `sigma`), `epchain` can be called repeatedly. This approach, however, requires more computing time than necessary. Tips for faster iterations will be given at the end of the next section.

2. Suppressing the low-energy detector noise:
   The purpose of this subtask is to suppress the detector noise at energies below ∼250 eV. While there is practically no EPIC pn detector noise present at higher energies, X-ray data below ∼200 eV are considerably contaminated by noise events. The noise properties of EPIC pn are fairly stable in time, but vary with position and energy. This fact complicates the background subtraction at low energies, which must take both the presence of cosmic diffuse X-ray background ('sky background') and detector noise into account.

   Compared to the sky background, the detector background is characterized by

   - a non-uniform spatial distribution: the detector background is not vignetted, but rather increases toward the CAMEX chip (to the top and to the bottom in Fig. 9 or Fig. 10) , with a steep rise in the rows closest to the CAMEX
   - a non-uniform spectral distribution: noticeable detector background is present only below a few hundred eV, with a steep rise toward the lowermost energies

   This subtask is intended to simplify the background subtraction at low energies. It makes use of the fact that the noise properties of EPIC pn vary with position and energy, but are fairly stable in time. The information about the spatial and spectral dependence is used in order to flag, on a statistical basis, the amount of events which correspond to the expected detector noise [25]. Subsequent removal of such events reduces the file size considerably, extends the useful energy range down to ∼120 eV, and makes a correct treatment of the spatial and spectral properties of the detector noise more straightforward than the conventional background subtraction technique. It even includes the possibility to take temporal changes of the detector noise into account.

A further interesting consequence of this subtask is an improved treatment of out-of-time events. As described in § 4.6, such events can be suppressed by constructing an out-of-time event file from the original data set and subtracting a binned version of this data set (spectrum or image), appropriately scaled, from the original data. The out-of-time event file is produced by randomly shifting the patterns along the RAWY axis and performing the gain and CTI corrections after-wards. This method works well for energies above ∼250 eV. At lower energies, however, the detector noise is steeply rising toward the rows closest to the CAMEX. Thus, the technique of randomly shifting the patterns along the RAWY axis also spreads this noise over the column. This can be avoided if the events flagged as noise events by `epreject` are removed from the data set before producing the out-of-time event file.

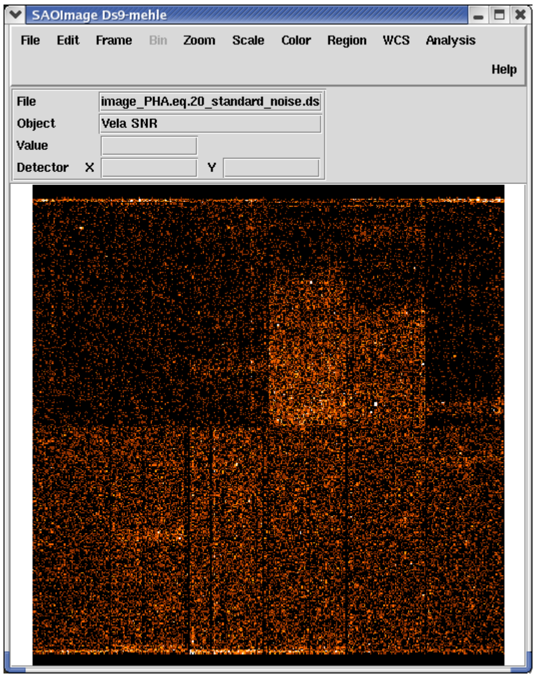
Contrary to the subtask for correcting the energy scale in specific pixels, where the default setting usually leads to reasonable results, the complexity of the noise suppression may require a fine-tuning of some settings, in order to take slight temporal changes of the detector noise into account. It is important to keep in mind that `epreject` will usually flag more than half of all the events as noise events and that the result will respond very sensitively to any change of the flagging criteria.

Parameters which can be set by the user:

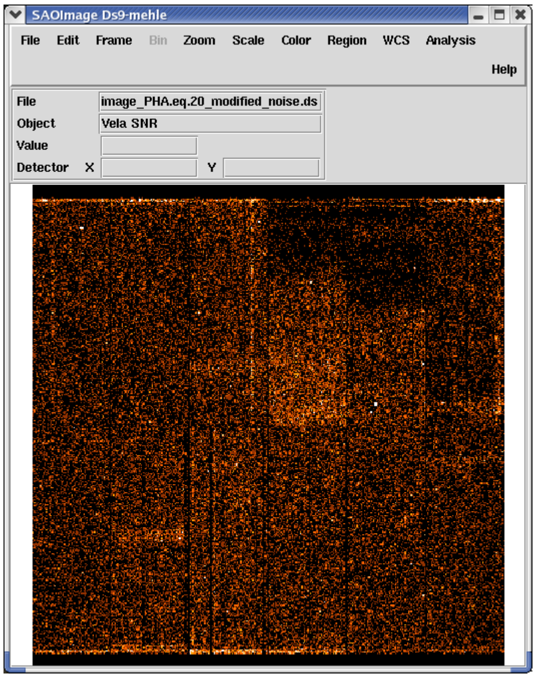
There are 13 parameters which can be set by the user: a cutoff parameter and 12 chip specific correction factors, all contained in the array `noiseparameters`.

The cutoff parameter controls the maximum percentage of events in the CCFs derived from exposures with the filter wheel closed, which may be considered as noise. This parameter should be set to a value which is slightly below 100% (default: 98%), to take the fact into account that even in these exposures not all events are due to noise. There is, e.g., some additional flux present from fluorescence of the filter wheel itself, triggered by energetic particles. This component would change with the position of the filter wheel. The tests which were done so far indicate that this parameter can usually be left at its default value.

The 12 chip specific correction factors control the relative amount of noise in each CCD. These parameters may require some adjustment, in order to take slight temporal variations of the detector noise into account. These variations are often similar for all CCDs of each quadrant. The default value for all CCDs is 1.0. Increasing this number will increase the percentage of events which will be considered as noise events. Changes of these values by a few percent should be sufficient in most cases. An example of how these parameters work is shown in Fig. 10. The recommended way for finding the appropriate setting of these parameters is by iteration, starting with the default setting, and controlling the image which is accumulated from “no-noise” 20 adu events for homogeneity (Fig. 10).



File image_PHA.eq.20_standard_noise.ds
Object Vela SNR



File image_PHA.eq.20_modified_noise.ds
Object Vela SNR

Figure 10: Images of all events with PHA = 20 adu in detector coordinates, after suppressing noise events. Left: processed with the standard setting of `epreject`. Right: processed with `noiseparameters="0.98 0.97 0.97 0.97 1.0 1.01 0.96 1.0 1.0 1.0 1.0 1.0 1.0"`. The color scale extends from 0 to 5 events per pixel.

Practical tips for running `epreject` on data sets in imaging mode:

These tips are demonstrated on a specific example. The data set used for this purpose is a 33.9 ks observation (in rev. 367) of the Vela SNR, an extended soft X-ray source, which was performed in fullframe mode with the medium filter.

`epreject` operates individually on each of the 12 CCD specific event files, which can be produced by either running `epframes` 12 times, e.g.,

```
epframes set=0367_0137550901_PNS00301IME.FIT eventset=rawevents01.dat gtiset=gti01.dat
...
epframes set=0367_0137550901_PNS00312IME.FIT eventset=rawevents12.dat gtiset=gti12.dat
```

or by running only the '`epframes`' part of `epchain`:

```
epchain runatthkgen=N runepframes=Y runbadpixfind=N runbadpix=N runbackground=N \
        runepreject=N runepevents=N runattcalc=N runevlistcomb=N runevselect=N
```

The second approach may be a bit more convenient, but it will take more time, because `epchain` will perform some additional tasks. Both methods will produce the files `rawevents01.dat` ...`rawevents12.dat`, which can then be analysed either directly with `epreject`:

```
epreject eventset=rawevents01.dat sigma=.. noiseparameters=".."
 ..
epreject eventset=rawevents12.dat sigma=.. noiseparameters=".."
```

or, with epchain:

```
epchain runatthkgen=N runepframes=N runbadpixfind=N runbadpix=N runbackground=N \
        runepreject=Y runepevents=N runattcalc=N runevlistcomb=N runevselect=N \
        sigma=.. noiseparameters=".."
```

This will add the columns OFF_COR and NOISE to the files rawevents01.dat .. rawevents12.dat, where the NOISE = 1 flag marks events which were considered as having been caused by detector noise (otherwise NOISE = 0). The column OFF_COR contains the adu values which were subtracted from the PHA values in order to readjust the energy scale in specific pixels. This column is present for the purpose of information and in order to avoid multiple corrections if epreject is run several times. As epreject has no destructive effect on the event file, it can be run repeatedly on the same file; the original information will be automatically restored at the beginning of this task.

Please note that, when epreject is called directly, all 12 CCD specific noise correction factors have to be given in each of the 12 epreject commands, although only one of them will be used in each command.

From the 12 rawevents*.dat files, images can be binned which contain only the "no-noise" events at the lowest transmitted energies (where such checks are most sensitive), e.g.:

```
evselect table=rawevents01.dat withimageset=true imageset=i01.fits \
         xcolumn=RAWX ycolumn=RAWY \
         imagebinning=binSize ximagebinsize=1 yimagebinsize=1 \
         expression="PHA.eq.20 .and. NOISE.eq.0"
```

Such images (for all 12 CCDs) should contain a similar density of events in the regions which are free from bright X-ray sources.

In this example (Fig. 10, right), a reasonable parameter setting was found to be noiseparameters="0.98 0.97 0.97 0.97 1.0 1.01 0.96 1.0 1.0 1.0 1.0 1.0 1.0" This assigns noise flags to the following percentage values of all events:
42.5% 48.8% 59.9% 21.9% 29.1% 42.7% 57.4% 57.1% 61.9% 69.7% 74.6% 79.4%

For comparison, the default setting noiseparameters="0.98 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0" flags the following percentages of events as noise, for CCDs 1-6: 43.8% 50.3% 61.8% 21.9% 28.8% 44.5% (Fig. 10, left).

These percentages vary from observation to observation. If there are no bright X-ray sources in the field of view, they can be considerably higher.

Once an appropriate setting has been found, the full epchain can be run with these settings, e.g.:

```
epchain runepreject=Y sigma=4.0 \
        noiseparameters="0.98 0.97 0.97 0.97 1.0 1.01 0.96 1.0 1.0 1.0 1.0 1.0 1.0" \
        screenlowthresh=0 screenrejected=Y
```

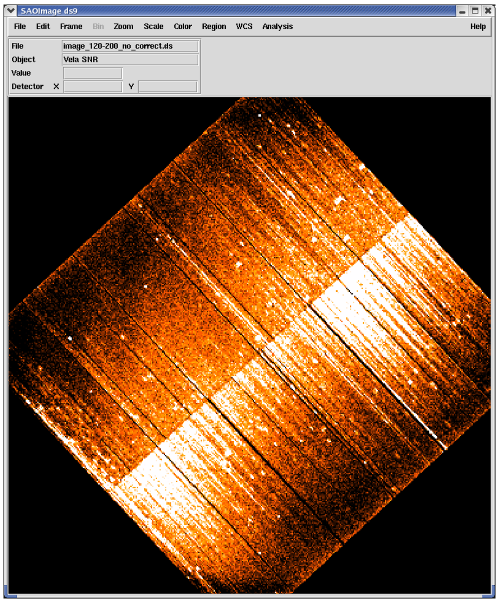
The parameter screenlowthresh=0 ensures that events with PI <150 eV will not be removed (unless they are regarded as noise events). This will allow to extend the useful energy range down to ∼120 eV. On the other hand, this file will also contain events with PHA <20 adu, which are the result of correcting the energy scale in specific pixels. These events should be removed, in order to get a homogeneous data set (it is currently not possible in epchain to apply a PHA filter), e.g., by:

```
evselect table=P0137550901PNS003PIEVLI0000.FIT keepfilteroutput=true \
         withfilteredset=true destruct=yes filteredset=cleanevents.fits \
         expression="PHA.ge.20"
```

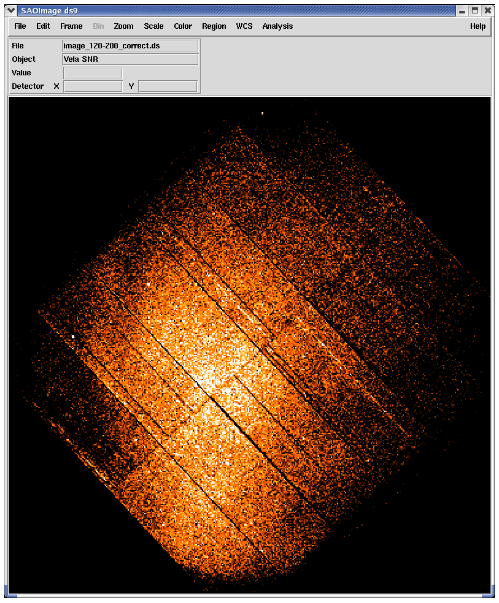
Note that the attempt to just complete the epchain processing by starting with the rawevents*.dat files, which were already processed by epreject before, would not work, because the screening will not be performed if the parameter runepreject=Y is not explicitely given in the call of epchain.

In Fig. 11 we show Vela SNR images in the very soft (120 - 200 eV) energy band processed without and with epreject to demonstrate the type of data quality improvement that one can expect to achieve for imaging mode EPIC pn data. Both images were produced with the following command (see § 4.9 for details on the generation of EPIC images):

```
evselect table=P0137550901PNS003PIEVLI0000.FIT withimageset=true \
         imageset=image.fits xcolumn=X ycolumn=Y \
         imagebinning=binSize ximagebinsize=100 yimagebinsize=100 \
         expression="PHA.ge.20 .and. PI.gt.120 .and. PI.lt.200 .and. \
         TIME.lt.124393155"
```



File: image_120-200_no_correct.ds
Object: Vela SNR
Value:
Detector X: Y:



File: image_120-200_correct.ds
Object: Vela SNR
Value:
Detector X: Y:

Figure 11: Vela SNR images in the energy range 120-200 eV, in sky coordinates. Left: processed without epreject, scale: 0-40 events/pixel. Right: processed with epreject, scale: 0-20 events/pixel.

4.3.2.1.2 Flagging soft flare events in EPIC pn timing mode data

The `epreject` task also offers a method to flag so-called "soft flare events" in EPIC pn timing mode data: The timing mode was designed for observing bright variable sources with a very high time resolution. Up to now it has been possible to use the spectra down to 300 eV in this mode. Below this energy the data is affected by so-called soft flares which are caused by stack overflows generated by high energy particles. These soft flares have a sharp rise (from 0 counts/s to several thousand counts/s in ∼0.10-0.15 s) then decaying back to 0 counts/s slightly slower in about 0.3-0.4 s). In some cases (extremely bright flares) they can lead to a FIFO overflow thus causing a small gap in the data of about ∼0.08 s. These gaps are not yet handled by the SAS.

`epreject` provides a method to mitigate the effect these flares have on the data by flagging such soft flare events so that they can later be screened from the data.

With this method it becomes possible to use the spectral information in the data down to the lowest energies detectable in the pn timing mode i.e. 200 eV.

This is particularly interesting for timing studies of isolated neutron stars, X-ray binaries and other variable objects, such as magnetic CVs, with very soft spectra.

In order to perform the soft flare event selection, the `withsoftflarescreening` parameter in `epreject` must be enabled. The program then generates a light curve in 0.1 s bins in the specified PHA range (parameter `softflareenergyrange`, default: 40-50 ADU) using the output event lists from the `epframes` task. The light curve is smoothed with a boxcar function whose width can be modified via the parameter `softflaresmoothparams`. This smoothed light curve is then used to help performing the soft flare event selection: each event is checked to see whether it has a PHA value in the user given PHA range and the light curve is above the user given threshold (parameter `softflarethreshold1`; this value has to be optimised for each observation as the contribution of the source photons in the given PHA range varies a lot from observation to observation).

Again, `epreject` has no destructive effect on the event file: A flag column labeled `Flare` is added to the rawevents file. A "1" in this column denotes that an event has been flagged as a soft flare event. This column can be used to screen the data.

The user should check the `epreject` task description for information on caveats to keep in mind when interpreting the pn timing mode data down to 200 eV.

4.4 Filtering calibrated EPIC event lists

EPIC calibrated event lists must be processed to generate data products including images, spectra and rate curves.

All product creations can be accomplished using the `evselect` task or (via a user friendly Graphical User Interface; GUI) the `xmmselect` task. The event list is filtered on the fly according to user-specified selection criteria. The filtered event list can be stored on disk and/or products can be extracted from this filtered event list.

The filtering process is carried out on an event-by-event basis controlled by user-specified selection criteria. These criteria take the form of character strings which can be 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.

Events for which the selection 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` (and `evselect`) support selections based on intrinsic event attributes falling into the following sub-categories.

1. Spatial selections in raw pixel (RAWX/RAWY), camera coordinates (DETX/DETY) or sky pixel (X/Y) space, or in any spatial coordinate system. Special pre-defined region shapes exists such as CIRCLE, ELLIPSE, ANNULUS, BOX, POLYGON. Spatial filtering with mask images or region files 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. Event selections based on informational and rejection attributes (see `evatt` package description for details).

5. Event selections based on any other event properties which are present (e.g. PATTERN or CCDNR).

The user should consult the `selectlib` package description which details the syntax and semantics of the expressions driving the above operations and gives sample expressions to demonstrate typical usage scenarios.

Note, while all of the data products generation can be done on the original event list, it can save significant computing time and memory first to create a filtered event list (e.g. removing high flaring background periods, restricting the analysis to certain regions and energies) and operate after-wards on the reduced filtered event file.

4.4.1 Filtering EPIC MOS concatenated event lists

Each of the two EPIC MOS cameras consists of seven individual 600 x 600 pixel CCDs. Because of telemetry constraints, a real time on-board recognition scheme filters out cosmic-ray tracks and exclusively transmits 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 pixel 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 pixel matrix, pixels above thresholds formed a predefined pattern.

In case of imaging mode, 32 patterns have been predefined (see Fig. 12, upper panel). They each correspond to an isolated event i.e. to a zone above threshold completely encircled by pixels below threshold. There are 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 of the imaging mode 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 of the valid X-ray events with the most accurate energy resolution. For imaging mode data patterns 0 to 12 are the canonical set of valid X-ray events which are well calibrated. Selection of these patterns constitutes 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 9 keV. Pattern 31 comprises 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 (see § 4.7).

In case of timing mode data, the pattern analysis is purely 1-dimensional (i.e. insensitive to other rows, see Fig. 12, lower panel), because each timing 'row' is actually the sum of 100 true rows, so the rows are not physically related.

The users are encouraged to test different filtering schemes which could be better suited to their own observation. However, general recommendations are as follows (for further details see XMM-SOC-CAL-TN-0018[7], a document describing the current status of EPIC calibration and data analysis):

- The selection expression (#XMMEA_EM) shall be used to filter out artifacts from the calibrated and concatenated dataset.
- In case of imaging mode, only patterns 0 to 12 should be kept as X-ray events since patterns with higher numbers are not created by X-rays or are highly contaminated by cosmics.
- In case of timing mode, only pattern 0 events should be selected for conservative data analysis.

4.4.2 Filtering EPIC pn concatenated event lists

The EPIC pn camera consists of twelve 64 x 200 pixel CCDs on a single wafer. For full frame and extended full frame modes the first 12 rows at the readout node are not transmitted to ground (are set to "bad", equivalent to "bad pixels"). In contrast to the MOS, all non bad pn events supposedly related to X-rays are transmitted to the ground and the pattern recognition and recombination of split partner is done off-line by the task `epevents`. Filtering of an EPIC pn dataset is entirely performed by the EPIC pn pipeline processing. The user is only advised to check the background level as a function of time, with as main aim the identification of any periods of enhanced low-energy proton flux, see § 4.4.4.

For pn, 13 valid patterns have been defined. As in case of the MOS, pattern 0 events are single pixel events. These comprise most of the valid X-ray events with the most accurate energy resolution. For the imaging mode patterns 0 to 12 (see Fig. 13) are the canonical set of valid X-ray events which are well calibrated. All higher patterns are not created by single X-ray photons and are due to pattern pileup. For the timing mode only singles plus doubles (pattern 0 to 4) should be selected for spectral analysis. Selection of these patterns constitutes the best trade-off between detection efficiency and spectral resolution.

The users are encouraged to test different filtering schemes which could be better suited to their own observation. However, general recommendations are as follows (for further details see XMM-SOC-CAL-TN-0018, [7] a document describing the current status of EPIC calibration and data analysis):

- The selection expression (#XMMEA_EP) should be used to filter out artifacts from the calibrated and concatenated dataset.
- For the imaging mode only patterns 0 to 12 should be kept as valid X-ray events since patterns with higher numbers are not created by X-ray or are highly contaminated by pileup.
- For the timing mode only singles plus doubles (pattern 0 to 4) should be selected for spectral analysis.

Users need to beware of the spatial non-uniformity of low energy multi-pixel events, e.g. when defining source and background accumulation regions (§ 4.10.1).

4.4.3 How to get pixels flagged "ON_BADPIX" back into the eventlist

Using the recommended selection expressions #XMMEA_EP and #XMMEA_EM, or even the more restrictive FLAG==0, pixels flagged as bad are not taken into account in the analysis. For specific circumstances it might be desired to revise a bad pixel flag and to make the pixel available again to the analysis using the task `ebadpixupdate`.

An important point regarding the bad pixels is that there are essentially three separate (though not exclusive) sets of bad pixels that must be dealt with. These are (1) bad pixels up-linked to the satellite and eliminated on-board, (2) bad pixels identified in the CCF but not up-linked, and (3) additional bad pixels associated with the particular observation in question. A corresponding list of bad pixels, individually for each CCD, is provided by the BADPIX**-extension tables inside the eventlist FITS-file.

Create eventlist files still containing bad pixels:

Per default, all EPIC pipeline processing tasks (`e[m/p]proc` or `e[m/p]chain`) perform the search for bad pixels both in CCFs and in the individual observation and flag the corresponding events from these bad pixels in the temporary eventlists. However the treatment of bad pixels at the time of the creation of the final eventlist file is different for the pn and the MOS pipeline tasks. The pn tasks take over the events from bad pixels into the final event list and assign to them the FLAG==ON_BADPIX, whereas the MOS tasks filter out these bad pixel events such that they do not appear any more in the final eventlist file. To get the MOS bad pixels back into the eventlist, it is necessary to re-run the MOS pipeline tasks using a special parameter to switch off the final filtering:

```
emproc flagfilteredevents=yes
```

or

```
emchain rejectionflag=760aa000
```

The use of the `emchain` parameter `rejectionflag=760aa000` (instead of `rejectbadevents=no`) is recommended because `emchain` keeps the events flagged as being ON_BADPIX while it still removes all other types of bad events.

Extract bad pixel table from the eventlist file:

The eventlist FITS files contain individual bad pixel tables of all CCDs. These tables can be extracted with the Ftool `fv`. After opening the eventfile with `fv <eventlist_name>`, two windows appear: a general `fv` menu window and a window listing FITS headers and tables of all extensions inside the eventlist file. The extensions of the bad pixel tables are named BADPIX** with ** being the CCD number. The bad pixel table of a specific CCD is presented in a separate window if the `All` button under the `Table` column is pressed. The table contains for each entry the pixel position in RAWX/RAWY coordinates, an extension size within the column (YEXTENT), the TYPE (1=hot, 2=flickering, 3=dead) and the BADFLAG indicating to which of the sets the bad pixel belongs: (1) for onboard bad pixel table, (2) for CCF bad pixel table and (3) for detections via the `ebadpixfind` task.

To extract one or more bad pixels tables, mark the corresponding extensions in the `Index` column and chose `Export HDUs...` from the `File` menu.
Of course `fextract` (and `fappend`, if several bad pixel tables are required) does the job as well.

Edit the bad pixel table:

After saving the extracted bad pixel tables in a new file, open this new file using `fv` and open the table you want to edit. To delete a bad pixel entry, mark the corresponding row by clicking on the row number and chose `Edit` → `Delete` → `Selected rows` from the `Edit` menu. Because bad pixels with BADFLAG=1 are eliminated onboard, these pixels cannot be recovered and their entries must not be changed. You have to pay attention on the YEXTENT column for the corresponding pixel entry. A single bad pixel has YEXTENT=1. If just one bad pixel within an extended region within a column should be recovered, the YEXTENT value can be edited directly. It might be necessary to add an additional row (`Edit` → `Insert` → `Row`) to define the remaining bad pixels in the column correctly. Save the changed bad pixel table (`File` → `Save`).

Update the bad pixel tables in the eventlist file:

The last step is to update the FLAG column inside the eventlist. This can be done by the task `ebadpixupdate`. The task needs as input the changed bad pixel tables and the eventlist file. The following example shows the update of an EPIC eventlist EPICEVENTLIST.FIT. The original bad pixel tables of all 12 CCDs have been exported to a file BADPIXTABLE.FIT. The bad pixel tables of CCD1 and CCD3 have been changed. The FLAG keywords inside the eventlist file are updated by the command:

```
ebadpixupdate eventset=EPICEVENTLIST.FIT fromccf=N overwrite=yes \
              badpixtables='BADPIXTABLE.FIT:BADPIX01 BADPIXTABLE.FIT:BADPIX03'
```

4.4.4 Filtering high background periods

The user is advised to produce a histogram of TIME values (a rate curve) in order to identify the useful period of low background level when the focal plane is not illuminated by low-energy protons.

Before any filtering, the user is also advised to inspect the background light curves produced by SSC pipeline processing (FBKTSR) where all sources and bright features/pixels have been masked out (see task descriptions of `emchain` and `epchain` for further details). If bright hard point-like sources are in the FoV, such sources should be excluded prior to the interactive rate curve generation as well.

In order to check for and remove any additional high background periods from the event list, the user can apply the following recipe:

- Build a rate curve of the TIME column in the calibrated event list using only valid single events with energy greater than 10 keV. This can be performed using `xmmselect` by entering 10000 in the PI lower limit box (in case of the pn in addition the PI upper limit should be set to 12000 avoiding noisy pixels with energies above 12 keV), 0 in the PATTERN upper limit box, pressing the PI and PATTERN button, and in addition appending "&& #XMMEA_EM" in case of MOS, or "&& #XMMEA_EP" in case of pn in the filter expression box. The round radio button in the TIME row should then be pressed. A suitable time bin needs to be selected for the OGIP rate curve accumulation, e.g. 25, 50 or 100 seconds, depending on the exposure duration.

- Examine the rate curve produced and identify 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. Recommended values are 0.35 counts/s for the MOS and 0.4 count/s for the pn camera, respectively, but depend on the science the user wants to do, and on the source signal-to-noise.

- Assuming that the rate curve is named `rates.fits`, a new GTI file can be created from the selected count rate threshold using a command line as follows (changing the count rate value for pn accordingly):

  ```
  tabgtigen table=rates.fits gtiset=BKG_GTI.fits \
            expression='RATE < 0.35'
  ```

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

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

  When using `xmmselect` from the GUI, the above expression should be typed in the 'Selection expression' window.

Low background intervals might differ significantly between different EPIC cameras and therefore a GTI created for one EPIC camera should not blindly be used for all the other cameras as well.

Users can also make use of GTIs which were created to make the pipeline images (only these - not the pipeline created event lists - have the flaring background removed): [2]

```
evselect table=inevlist.fits \
         filtertype=dataSubspace dssblock=image.fits:PRIMARY \
         keepfilteroutput=yes withfilteredset=yes filteredset=outevlist.fits
```

where `inevlist.fits` is the input event list (e.g. from the pipeline), `image.fits` is the pipeline image from which to take the GTI and `outevlist.fits` is the output flare GTI filtered event list. If the user prefers the task `xmmselect` to apply such pipeline processed GTI to an input event list, the "Filtered table" product selection needs to be started. On the "General" `evselect` GUI parameter page, the "Filtering" method must be changed to "dataSubspace" and other `evselect` parameters need to be set as in the example above.

A third method to clean an EPIC event list from flaring background is offered by the SAS task `espfilt`. `Espfilt` applies a user-selected method for filtering an event list of cosmic soft proton events. The filtering produces a new event list, lightcurves of the object and corners of the detector (where X-rays produced by cosmic soft proton events are registered), raw and filtered images and a plot of the lightcurves and GTIs. In contrast to the cleaning methods described above which are based on a rate lightcurve threshold, `espfilt` creates a histogram of rate values from the light curve, finds the most likely value assuming that to be similar to the mean of the quiescent rate, then fits a Gaussian to a small window around that value in the histogram to determine the true mean and dispersion of the quiescent background rate. The program then excludes time intervals with rate higher than a multiple of the dispersion above the mean quiescent background and excludes good regions shorter than some limit.

[2] Note, that due to a backward incompatibility since the introduction of `param-2.0`, this method only works if the pipeline image also has been generated with SAS version 6.0 or later.

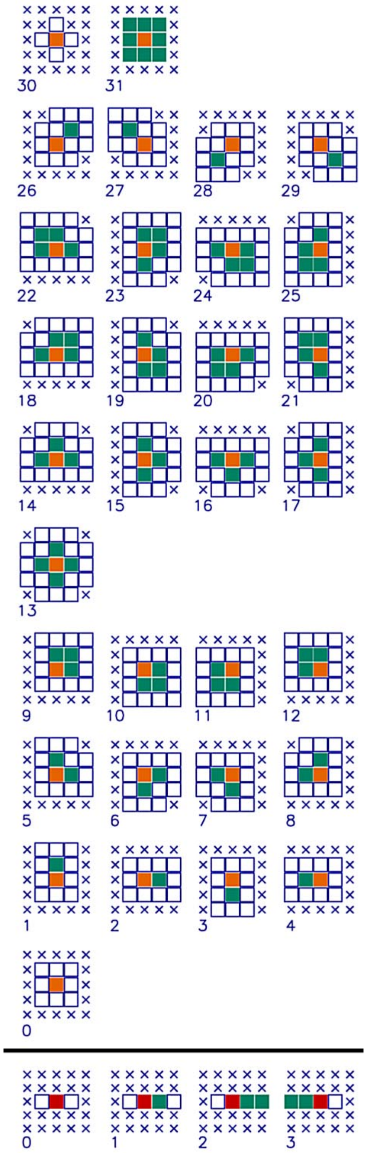


Figure 12: List of EPIC-MOS patterns: the *upper panel* for **imaging mode** should be interpreted as follows: each pattern is included in a 5 x 5 matrix used for proximity analysis, a pattern is centered by definition on the pixel with highest charge, this central pixel is colored in red, the other pixels above threshold in the pattern are colored in green, all pixels colored in white must be below threshold, the crossed pixels are indifferent (they can be above threshold). The philosophy for patterns 0-25 is that a good X-ray pattern must be compact, with the highest charge at the center, and isolated (all pixels around are below threshold). Patterns 26-29 are the so-called diagonal patterns, not expected from a genuine X-ray, but which can arise in case of Si-fluorescence or of pileup of two monopixel events. The *lower panel* for **timing mode** should be interpreted in the same way as in imaging mode, with the difference that the place where maximum charge occurs is ignored. Due to this, all doubles appear as Pattern 1, whether leading or trailing. Patterns 2 and 3 are mostly not due to true X-rays, but to cosmic-ray tracks.

```
                     . . .
single event         . X .
                     . . .

                   . . . . .   . . . . .   . . . . .   . . . . .
                   . . x . .   . . . . .   . . . . .   . . . . .
double pattern     . . X . .   . . X x .   . . X . .   . x X . .
                   . . . . .   . . . . .   . . x . .   . . . . .
                   . . . . .   . . . . .   . . . . .   . . . . .

                   . . . . .   . . . . .   . . . . .   . . . . .
                   . . x . .   . . x . .   . . . . .   . . . . .
triple pattern     . x X . .   . . X x .   . . X x .   . x X . .
                   . . . . .   . . . . .   . . x . .   . . x . .
                   . . . . .   . . . . .   . . . . .   . . . . .

                   . . . . .   . . . . .   . . . . .   . . . . .
                   . m x . .   . . x m .   . . . . .   . . . . .
quadruple pattern  . x X . .   . . X x .   . . X x .   . x X . .
                   . . . . .   . . . . .   . . x m .   . m x . .
                   . . . . .   . . . . .   . . . . .   . . . . .
```

Figure 13: List of valid EPIC-pn patterns (cf. Fig. 12). Here "." marks a pixel without an event above threshold, "X" is the pixel with the maximum charge ('main pixel'), 'x' is the pixel with a non-maximum charge, "m" is the pixel with the minimum charge. These 13 figures refer to the SAS PATTERN codes 0 (singles), 1-4 (doubles), 5-8 (triples) and 9-12 (quadruples), respectively. The RAWX co-ordinate is running rightward and the RAWY co-ordinate running upward.

4.5 Merging event lists

The task merge merges two EPIC event lists or two additional files (attitude files, orbit files etc.) from two exposures (even from different observations and instruments), pointing in the same or an adjacent direction. The output files can be used by evselect for the product generation and higher level detection and time analysis tasks.

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 in some cases 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 increasing 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 file (event list, attitude file, orbit file) is produced. 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 WCS attributes are calculated. The size of the image is given by the parameter imagesize.

For example, two event files can be merged reprojecting X/Y coordinates to a mean reference point using the following command.

```
merge set1=inevlist1.fits set2=inevlist2.fits outset=outevlist.fits
```

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

```
merge set1=inevlist1.fits set2=inevlist2.fits outset=outevlist.fits \
      withradec=Y ra=1.1 dec=-0.7
```

The user has to be careful when using any merged output for a quantitative analysis, as this task is not yet optimal for this purpose. There are many open questions regarding merging files originated in different exposures, and the handling of EXPOSURE, GTI and BADPIX extensions (check the task description of merge for further details).

4.6 EPIC-pn Out-of-time events

For the EPIC imaging observing modes, photons are not only registered during the actual integration interval but also during the readout of the CCD (shift of charges along a column toward the readout node). These so called Out-of-Time (OoT) events get a wrong RAWY value assigned and thus finally a wrong energy correction (the correction for charge transfer inefficiency (CTI) depends on the distance from the readout node).

The effect of OoT events broadens spectral features and can be seen in EPIC images as a strip of wrongly reconstructed event positions in RAWY. The fraction of Oot events scales with the mode-dependent ratio of integration and readout time and is highest for the pn full frame (6.3 %) and extended full frame (2.3 %) mode (see the UHB for further details).

If OoT events are a problem for the analysis (if highest spectral resolution is required or if a clean image of e.g. extended emission surrounding a bright source is needed), the tasks `epproc` (§ 4.3.2) and `epchain` offer the possibility to simulate OoT events based on the original event list. Note that OoT events do not need to be removed for the purpose of source detection as they are dealt with in the background characterisation stage there (§ 4.12.3 and task description of `esplinemap`). The pipeline tasks need to be run twice, first creating an OoT event list and then the "normal" calibrated event list. The OoT event list is produced by calling the subtask `epevents` with the non-default parameter setting `withoutoftime=yes`. The OoT event list is created treating all events as out-of-time events: after the pattern recognition for the same TIME, PHA, and RAWX a new RAWY value is simulated by randomly shifting the pattern along the RAWY axis and performing the gain and CTI correction afterwards.

`epchain` resembles largely the `epproc` task but can speed up the process to remove OoT events as raw intermediate files from the first run (to create the OoT event list) can be kept for the second run (to create the "normal" calibrated event list) allowing the user to skip some of the already performed processing steps.

To produce an OoT event list from a pn imaging mode ODF, the task `epchain` should be called in the following way:

```
epchain runbackground=N keepintermediate=raw withoutoftime=Y
```

This will create an event list that contains the same number of OoT events as there are events in the "normal" event file. The Out-of-Time event file will have a name like PiiiiiijjkkPNblllOOEVLInmmm.FIT (cf. § 4.2).

The "normal" event list is created with:

```
epchain runatthkgen=N runepframes=N runbadpixfind=N runbadpix=N
```

In order to save time one does not have to re-create the attitude file, re-run `epframes`, `badpixfind` and `badpix` either. `epchain` will create an output event file with the name PiiiiiijjkkPNblllPIEVLInmmm.FIT (the standard name of a calibrated pn imaging mode event list).

The two event lists can now be taken to create output products like images and spectra:

4.6.1 Removing Out-of-Time events from pn images

The effect on images is shown in Fig. 14 (btw. note the arc-like structures due to single mirror reflections (stray light) at the top left of the FoV).

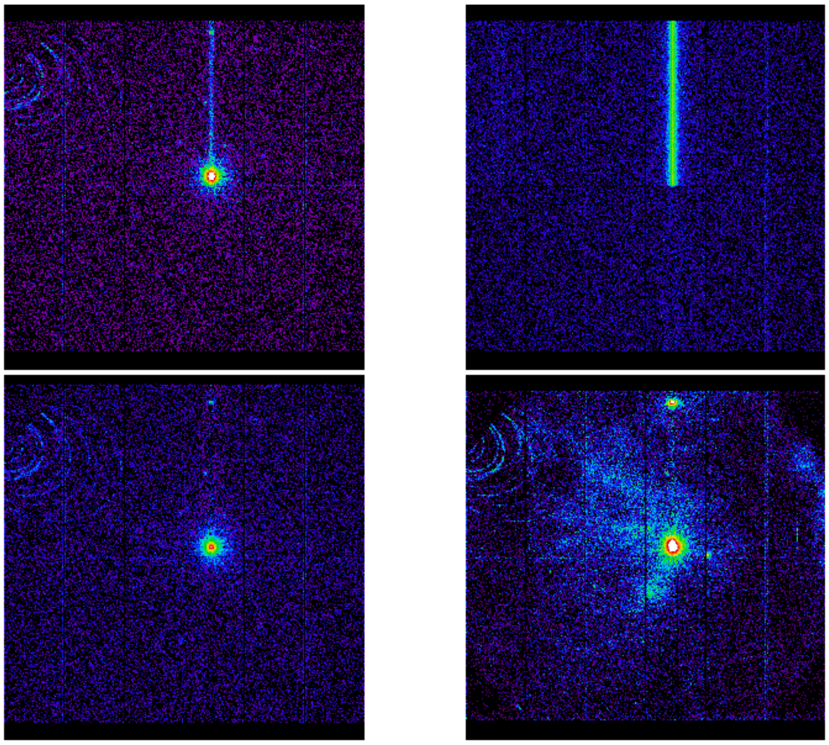


Figure 14: Effect of OoT events on images: The upper left panel contains a 2-10 keV band image of a pn observation of a bright source in full frame mode with the OoT events visible as a strip running along the length of the CCD. The upper right panel depicts the modeled OoT event distribution whereas in the lower left panel these are subtracted from the original image. The lower right panel shows the distribution of cleaned events in the soft (0.2-2 keV) energy band for comparison.

As mentioned above, 6.3 % of all events in a full-frame mode event file are OoT events. Because the OoT event list contains the same number of events as the original event list, the OoT image needs to be multiplied by 0.063 before subtracting it from the original image. Assuming that the user has created two images (see § 4.9) from the OoT and the "normal" event lists (named `image_oot.fits` and `image.fits`, respectively), the necessary image arithmetic can be performed with the `FTOOL` task `farith`:

1. The OoT image is multiplied by 0.063 creating the output file `image_oot_scaled.fits`:

```
farith image_oot.fits 0.063 image_oot_scaled.fits MUL
```

2. The scaled OoT image is subtracted from the "normal" image:

```
farith image.fits image_oot_scaled.fits image_clean.fits SUB
```

The resulting image `image_clean.fits` is more or less cleared from OoT events.

4.6.2 Removing Out-of-Time events from pn spectra

The handling of Out-of-Time events in spectra relies on dealing with FITS tables. As in the case of image cleaning of OoT events (see § 4.6.1 above), the necessary table manipulation and arithmetic can be performed with `FTOOL` tasks, e.g. `fv` and `faddcol`. The example we give here is from an exposure of the internal calibration source. Due to the strong lines in this source it is easier to see the effects that OoT events have on a spectrum. First create two spectra, one from the "normal" event file, e.g. `P0122320101PNS003PIEVLI0000.FIT` and the other one from the OoT event file `P0122320101PNS003OOEVLI0000.FIT`:

```
evselect table=P0122320101PNS003PIEVLI0000.FIT withspectrumset=yes \
         spectrumset=source.fits energycolumn=PI \
         withspecranges=yes specchannelmin=0 specchannelmax=20479 \
         spectralbinsize=5 \
         expression='(FLAG==0)&&(PATTERN==0)'

evselect table=P0122320101PNS003OOEVLI0000.FIT withspectrumset=yes \
         spectrumset=source_oot.fits energycolumn=PI \
         withspecranges=yes specchannelmin=0 specchannelmax=20479 \
         spectralbinsize=5 \
         expression='(FLAG==0)&&(PATTERN==0)'
```

The next step is to copy the column `COUNTS` from the OoT event spectrum `source_oot.fits` into the source spectrum file `source.fits` (which is to be cleaned from OoT events): Then multiply this new column in the file `source.fits` by 0.063 (same explanation as given above for images). Then, as the last step, subtract this column from the original column COUNTS of the source spectrum file `source.fits`. The result is shown in Fig. 15.

The effect of OoT events basically is that they broaden the shoulders of a line and can cause a blending of lines. OoT events do not change the Full-width-at-half-maximum. The red spectrum shows that the lines in a spectrum cleared from OoT events are much easier to separate than in an untreated spectrum (black data points). Please keep in mind that the spectrum is displayed in a double logarithmic scale and restricted to a small energy band so as to emphasize the effect. Furthermore, the example shown here is data from the internal calibration source, which is very bright. **For most targets a correction for OoT events in a spectrum should not be necessary. In any case, a correction is only necessary if OoT events overlap the source being investigated.**

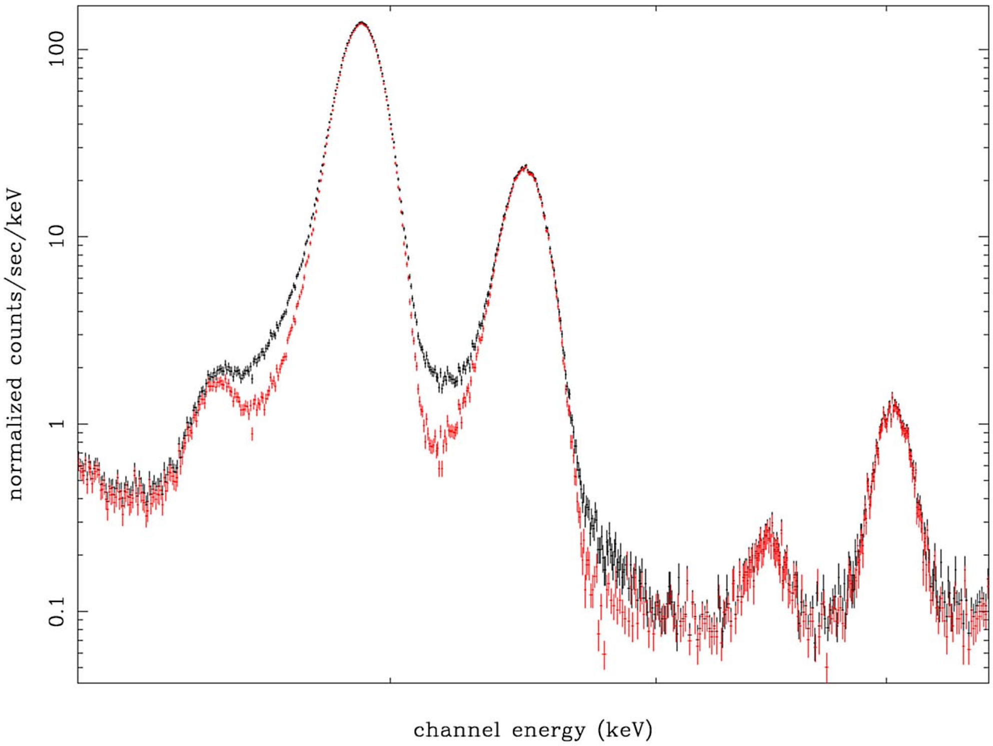


Figure 15: Effect of Out-of-Time events on a source spectrum (in this case the internal calibration source): the black data points display the source spectrum which is still contaminated by OoT events, the red points mark the source spectrum being cleared from OoT events. The energy range displayed is 5-8.5 keV.

4.7 Pile-up

Pile-up occurs when a source is so bright that there is the non-negligible possibility that two or more X-ray photons deposit charge packets in a single pixel ("photon pile-up"), or in neighboring pixels ("pattern pile-up", i.e. singles pileup to doubles etc.), during one read-out cycle (i.e. one frame). In such a case these events are recognized as one single event having the sum of their energies. If this happens sufficiently often, this will result in a hardening of the spectrum as piled-up soft events are shifted in the spectrum to higher energies.

In addition, pile-up leads to a more or less pronounced depression of counts in the central part of a bright source, resulting in flux loss. Pile-up also affects light curves, suppressing high count rates.

The XMM-Newton UHB lists (readout mode dependent) maximum count rates above which a source suffers from pile-up. In general the MOS camera is much more susceptible to pile-up than the pn.

To check whether pile up indeed is a problem, use the SAS task `epatplot`. (see the thread at:

http://xmm.esac.esa.int/sas/current/documentation/threads/epatplot.html)

To run `epatplot` one needs to create an event file for the source as described below in step 3) of § 4.10.1. The input event file name (e.g., `src_evlist.fits`) must be specified via the `epatplot` task `set` parameter. If the resulting plot shows the model distributions for single and double events diverging significantly from the measured distributions, this is a strong indication that pile-up has occurred. Fig. 16 shows an example of a bright source observed in pn full-frame mode which is strongly affected by pile-up. Due to "pattern pile-up" more doubles are produced at the expense of single pixel events.

A common strategy adopted to analyse spectra of piled up sources has been to excise the core of the point-spread function (PSF). Recent investigations have shown that, at least in some cases, this strategy can yield incorrect results. The problem is most likely related to an inaccurate modeling of the energy dependence in the PSF model used by arfgen to evaluate encircled energy fractions.

The user who suspects that pile-up may be affecting his/her observation is advised to first use the epatplot tool to asses the presence and level of pile-up. In case of moderate pile-up (see XMM−Newton Users Handbook, [1]) the user is advised to make use of pattern 0 spectra, which are less sensitive than other patterns, for both MOS and pn cameras and use the new method for pile-up correction provided by S. Molendi and S. Sembay (2003) [4].

The older method, of excising the inner part of the source emission from the event list used for the creation of the spectrum, is achieved via the `xmmselect` task, first displaying the whole source region as an image (see § 4.9) and then defining an annulus (via the `ds9:Region` menu and importing the region into the `xmmselect` selection expression via the "2D region" button (see § 4.10.1, step 5)). With this selection expression, a filtered event list, named e.g. `src_annulus_evlist.fits`, can be created with `xmmselect`. Finally `epatplot` should be called again now with the `src_annulus_evlist.fits` as input data set. Inspecting the created pattern distribution curves, the inner radius of the annulus should be increased as long as the pattern distribution agrees with the model. Note, excluding the inner part of the source from the analysis will of course reduce the number of events for further analysis. So an iterative process for finding the best exclusion inner radius should be performed. Fig. 17 shows the pattern distribution of the same source as above, but after exclusion of the inner part of the source. The pattern

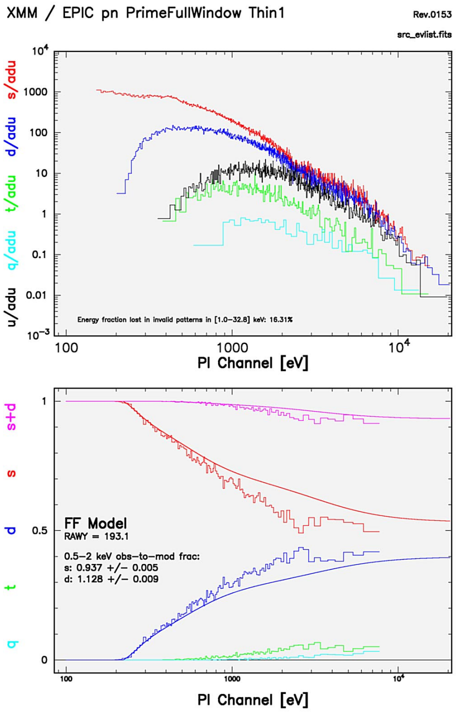


Figure 16: Plot of the pn pattern distribution with energy as produced by `epatplot`. The deviations of the single, double and single+double distributions from the model are clearly visible.

distributions now agree with the model curves and a resulting spectrum would in principle be free of pile-up effects.

4.7.1 How to analyse a piled-up Timing mode observation

If the source count rate is greater than ∼ 800 counts/s for a pn Timing mode observation or ∼ 100 counts/s for MOS Timing mode then the source events are likely to be affected by photon pile-up.

This can be checked by selecting events from regions which exclude the inner column(s) of the data and running `epatplot`. For instance the spatial selection region, which shows no pile-up for the MOS may be:

```
( RAWX in [280:298] || RAWX in [303:321] )
```

The BACKSCAL value can be calculated for the created spectrum in the normal manner using `xmmselect` or the `backscale` task. Similarly the task `rmfgen` may be used to create the redistribution matrix (RMF) in the normal manner (see § 4.10.2).

The calculation of the ARF is a little more complicated because the encircled energy (PSF) correction is not performed correctly by `arfgen` in this case. To overcome this problem the following steps need to be taken.

1. Produce a spectrum (spec_full.ds) from the full area without excluding the central columns, e.g. `( RAWX in [280:321] )`

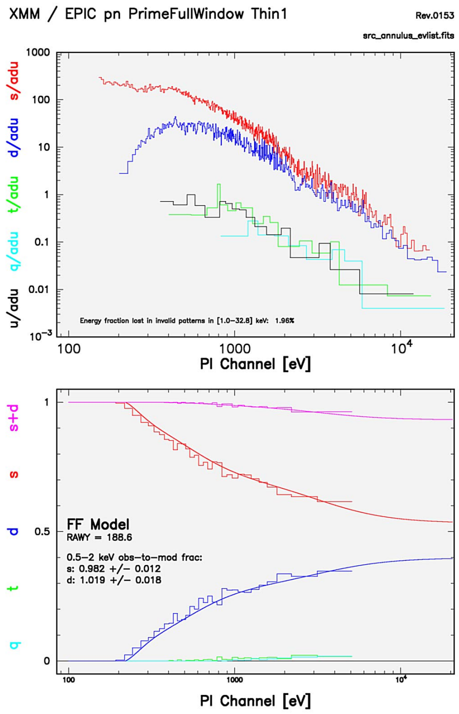


Figure 17: Plot of the pn pattern distribution with energy as produced by `epatplot`. After exclusion of the inner part of the source, the pattern distributions are in agreement with the model curves.

2. Produce a spectrum (spec_inner.ds) from the excluded region, e.g.

   ```
   ( RAWX in [299:302] )
   ```

3. Produce an ARF for both of these files

   ```
   arfgen spectrumset=spec_full.ds arfset=arf_full.ds
   arfgen spectrumset=spec_inner.ds arfset=arf_inner.ds
   ```

4. Subtract the inner ARF from the total ARF by using the ftool:

   ```
   addarf "arf_full.ds arf_inner.ds" "1.0 -1.0" arf_outer.ds
   ```

The arf_outer.ds file can then be used as the ARF for fitting the spectrum created from the region:

```
( RAWX in [280:298] || RAWX in [303:321] )
```

together with the RMF produced earlier.

4.8 Analysis of extended sources

Analysis of extended sources is complex, challenging and time-consuming. There is currently neither an official SAS recipe, nor a simple thread. Several groups were/are independently working in this field. The XMM-Newton EPIC Background Working Group (BGWG) was therefore founded in 2005 as a steering and supervising committee to provide the users with clear information on the EPIC Background and (SAS)-tools to treat the background correctly for various scenarios.

A dedicated Background Analysis web page is available at http://xmm.esac.esa.int/external/xmm_sw_cal/background/ with information and tools collected and documented by the Background Working Group:

- A table summarizing the temporal, spectral and spatial properties of the different EPIC background components,
- Progress and meetings of the XMM-Newton EPIC Background Working Group,
- A list of currently available products, like:
  - The XMM-Newton Extended Source Analysis Software package, XMM-ESAS,
  - XMM-Newton 'blank sky' background files and related software,
  - Filter wheel closed data,
  - Other scripts,
  - Links to related papers.

A main problem in the analysis of extended sources is that often no statistically useful blank background region can be defined in the observational field-of-view. A workaround is to make use of the provided 'blank sky' background files to generate background spectra corresponding to the camera/mode/filter combination rescaled to the actual observation.

An alternative approach is to model the background spectra based on the background conditions of the indivual observation under study: The recommended method is to make use of the Extended Source Analysis Software (XMM-ESAS) package. This tool is available at http://xmm.esac.esa.int/external/xmm_sw_cal/background/epic_esas.shtml and allows to model the quiescent particle background both spectrally and spatially for the MOS detectors (support for pn is planned for the near future). XMM-ESAS produces background spectra for user-defined regions of the detectors and background images.

Further details are available from the Background Analysis web page and from a dedicated SAS Workshop presentation at
http://xmm.esac.esa.int/external/xmm_sw_cal/sas_workshop/sas_ws7_files/presentations.shtml

4.9 Generating EPIC images

EPIC images can be created from an event file with the `evselect` task from the command line or with the `xmmselect` task in an interactive GUI driven way.

4.9.1 Image generation with `evselect`

The task `evselect` creates a simple output image file in the following way:

```
evselect table=inevlist.fits xcolumn=X ycolumn=Y imagebinning=binSize \
         ximagebinsize=100 yimagebinsize=100 \
         withimageset=true imageset=image.fits
```

where `table` specifies the input event list, `x/ycolumn` the coordinates used for the image (here sky coordinates), `imagebinning` is a switch for an automatic or user defined binsize, `x/yimagebinsize` is the user defined bin size of the image in each direction, `withimageset` is the switch to create an image and `imageset` defines the name of the output image file. In this example the bin size is set to 100 which corresponds in the sky pixel system (units are 0.05 arcsec) to 5 arcsec.

The image can be displayed with the command:

```
ds9 image.fits
```

4.9.2 Image generation with `xmmselect`

An alternative (and more interactive) approach is to create images via the `xmmselect` GUI, which is started with the command:

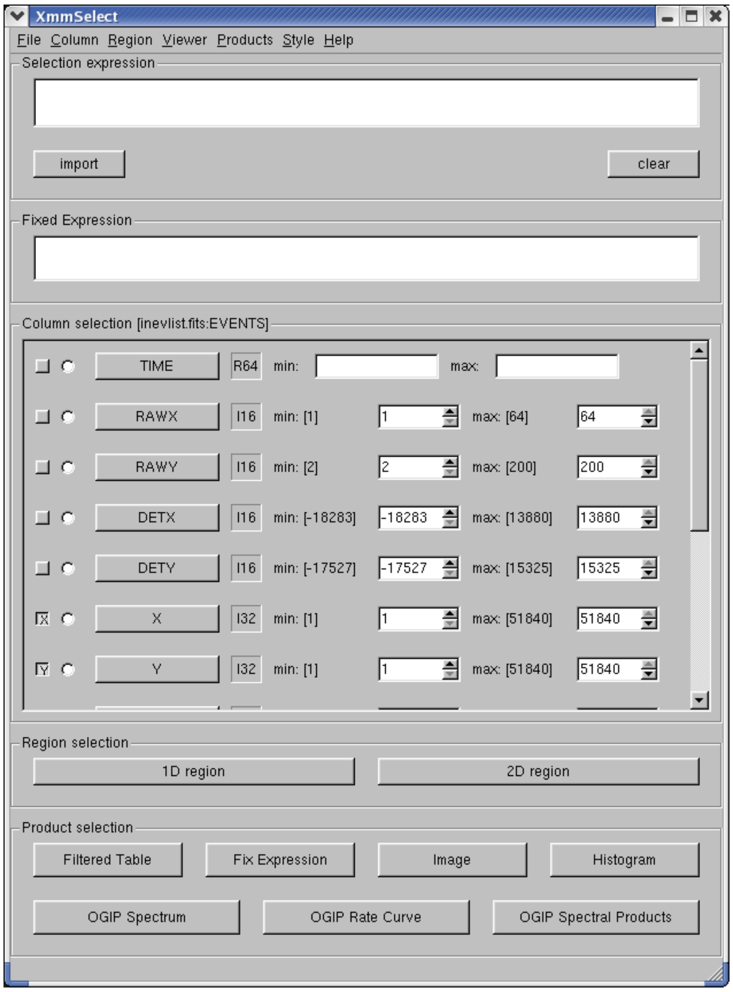
```
xmmselect table=inevlist.fits
```

Afterwards, the following steps need to be performed to generate and display an EPIC image:

1. Choose X and Y in the Column selection (input columns for the image, e.g. X and Y (sky coordinates), RAWX and RAWY (raw CCD-specific coordinates) or DETX and DETY (camera-specific coordinates)) and click in the Product selection part on the "Image" tab (see Fig. 18). This will start the general window of the `evselect` task.

   Caution: if `x/ycolumn` in `evselect` are the CCD raw coordinates RAWX and RAWY, all selected CCDs will be projected on top of each other. If an image in RAWX and RAWY is needed e.g. for diagnostic purposes, an additional filter expression selecting a single CCD only should be applied. In order to get an image of all active CCDs in the camera, the `x/ycolumn` should be set to X and Y or DETX and DETY.

2. In the `evselect` main window (Fig. 19) one now has the option to apply further events filtering (if not yet specified in the `xmmselect` main window). If one is happy with the set-up, the next step is to click on the "Image" button on the top of the GUI.

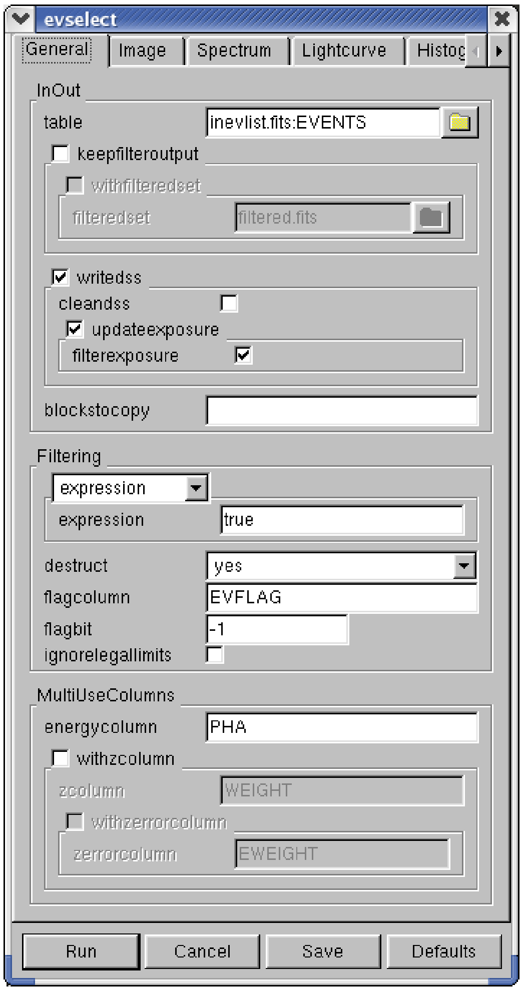


□	○	TIME	R64	min:		max:	
□	○	RAWX	I16	min: [1]	1	max: [64]	64
□	○	RAWY	I16	min: [2]	2	max: [200]	200
□	○	DETX	I16	min: [-18283]	-18283	max: [13880]	13880
□	○	DETY	I16	min: [-17527]	-17527	max: [15325]	15325
☒	○	X	I32	min: [1]	1	max: [51840]	51840
☑	○	Y	I32	min: [1]	1	max: [51840]	51840

Figure 18: In the xmmselect main window, an image can be extracted by selecting X/Y as image axis and by pressing the "Image" button.

3. In the now visible evselect window with the image creation related parameters (Fig. 20) one needs to specify a name for the output image (parameter imageset). By default, events will be binned into an image with 600 x 600 square pixels. The image size can be modified as well as the binning mode: setting Binning to binSize allows e.g. to project events onto a grid with specified pixelsize (a value of x/yimagebinsize=100, e.g. will result in 5 x 5 arcsec pixels).

4. Click on the "Run" button to start the image creation process. After the image is created and stored in the working directory, xmmselect will automatically launch the image viewer. The ds9 viewer allows the user to zoom in on a region of special interest, to change the intensity scale and the color. The position of the mouse pointer is displayed (in RA and Dec in case of a sky image, or in linear coordinates in case of a camera or raw coordinate image). For details on the image viewer (developed by Smithsonian Astrophysical Observatory), follow the links to further information that are available under the ds9 "Help" button.



InOut

table: inevlist.fits:EVENTS

☐ keepfilteroutput

☐ withfilteredset

filteredset: filtered.fits

☑ writedss

cleandss ☐

☑ updateexposure

filterexposure ☑

blockstocopy:

Filtering

expression

expression: true

destruct: yes

flagcolumn: EVFLAG

flagbit: -1

ignorelegallimits ☐

MultiUseColumns

energycolumn: PHA

☐ withzcolumn

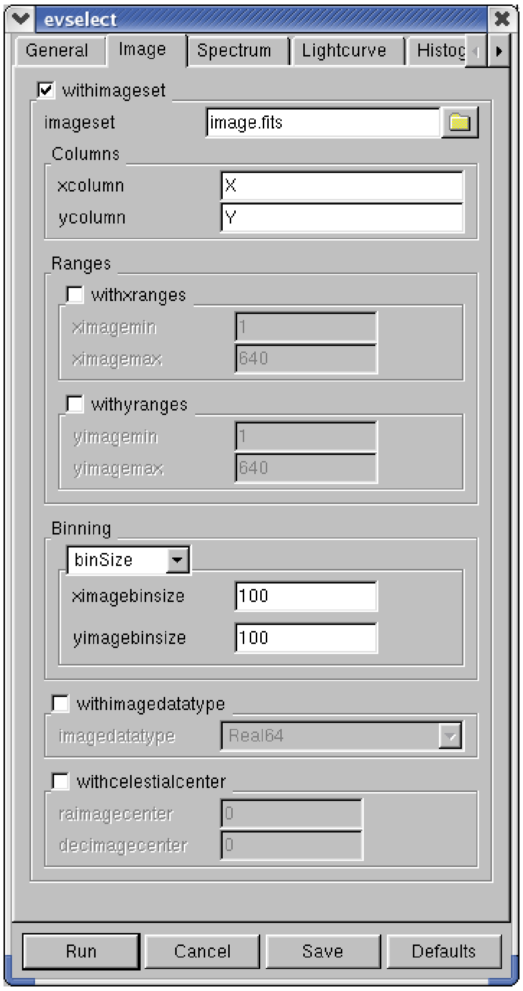
zcolumn: WEIGHT

☐ withzerrorcolumn

zerrorcolumn: EWEIGHT

Figure 19: The `evselect` main window, where e.g. the selection expression still can be modified.

Note, if running the image creation for a second time with the same name for the output image (parameter `imageset`), the previously created image file will be overwritten. This can be avoided by starting `xmmselect` with the "-c" (no clobber) option (see documentation of the `taskmain` SAS package).



☑ withimageset

imageset image.fits

Columns

xcolumn X

ycolumn Y

Ranges

☐ withxranges

ximagemin 1

ximagemax 640

☐ withyranges

yimagemin 1

yimagemax 640

Binning

binSize

ximagebinsize 100

yimagebinsize 100

☐ withimagedatatype

imagedatatype Real64

☐ withcelestialcenter

raimagecenter 0

decimagecenter 0

Figure 20: The `evselect` window with the image related parameters, where e.g. the output image name and the binning of the events can be modified.

4.10 EPIC spectral analysis

4.10.1 Generating spectra

EPIC calibrated event lists must be filtered to generate spectra. As for images, this filtering is normally performed in two steps. First, the event lists are screened to reject spurious data and to select only events which contain information of sufficient quality for further scientific analysis. Secondly, the screened data are filtered to construct data subsets adapted to specific spectral analysis.

For latest information on general recommendations for a conservative spectral analysis (i.e. recommendations about where the data should be taken from the CCD, which energy and pattern range should be used, which event quality flags should be selected), the user is strongly advised to check the related sections in document "EPIC status of calibration and data analysis" (XMM-SOC-CAL-TN-0018, [7]), available on the XMM-Newton SOC "Calibration" Web page.

See also the SAS threads at:

http://xmm.esac.esa.int/sas/current/documentation/threads/

The screening and filtering activities can be performed in an interactive manner with the SAS task `xmmselect`. Basically, there are two possible approaches offered by `xmmselect` to generate EPIC spectral products:

- generate source and background spectra separately via the `xmmselect` product selection "OGIP Spectrum". This approach requires more analysis steps but is described here as users might be interested in details,
- generate source and background spectra (together with related response files) with the meta-task `especget` called via the `xmmselect` product selection "OGIP Spectral Products".

For both cases, an EPIC image must be created and displayed (via `ds9`) with `xmmselect` running on a filtered (or the original) event list (see § 4.9.2).

In the following, the analysis steps for the "OGIP Spectrum" approach are described:

1. In the `ds9` window, create a region for the source of interest. Click once on the `ds9` image and a region circle will appear (other region shapes are available as well). Click on the region circle and the region will be activated, allowing the region to be moved and its size to be changed. Having created, placed, and sized the region appropriate for the source, click the "2D region" button in the `xmmselect` GUI. This transfers the region information into the "Selection expression" text area in `xmmselect`. A newly defined `ds9` region file can optionally be saved to disk via the `ds9` "Region" → "Save Regions..." menu and re-loaded for further analysis steps via the "Region" → "Load Regions..." option.

   Note: For pn data of bright sources and of sources with narrow lines it might be good to extract two spectra and corresponding backgrounds, response and ancillary files: one set for single pixel events (`PATTERN==0`) and another set for doubles (`PATTERN IN [1:4]`). Fitting these two spectra simultaneously will show if there are any problems with pile-up (see § 4.7) and - as the energy calibration for singles is slightly better than the one for doubles - will show the line features at highest energy resolution in the single events spectra.

2. To extract the spectrum, first click the circular button next to the PI column in the `xmmselect` GUI. Next click the "OGIP Spectrum" button. Select the "Spectrum" page of the `evselect` GUI to set the file name and binning parameters for the spectrum. For example, set `spectrumset` to `src_spectrum.fits`. If the canned responses are going to be used (see § 4.10.2), the `spectralbinsize` must be set to 15 for the MOS or 5 in case of the pn. `withspecranges` must be checked, `specchannelmin` set to 0, and `specchannelmax` set to 11999 for the MOS or 20479 for the pn. Fig. 21 shows a plot of an example output spectrum which is automatically displayed in a `grace` window.

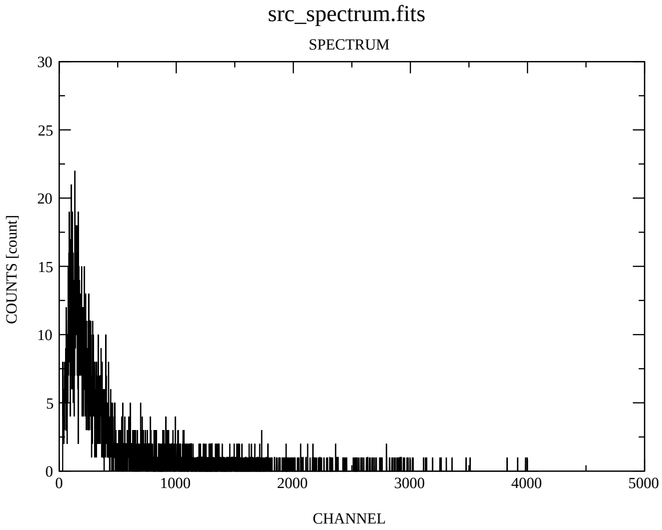


Figure 21: Spectrum of a source.

Note: `xmmselect` performs the calculation of the `BACKSCAL` factor, which takes into account CCD gaps, bad pixels and the size of the extraction region, on the fly during the spectrum generation. If `evselect` is used instead for the extraction of the source and background spectra, the `BACKSCAL` factor must be calculated explicitly by executing the `backscale` task before any subsequent quantitative analysis, e.g. with `Xspec`, can be performed. For a purely qualitative check of the source spectrum, however, the `BACKSCAL` computation step can be skipped to save significant computing time especially in case of large extraction areas.

3. For checking whether pileup (§ 4.7) might be a problem, create a source event file by checking the `keepfilteroutput` and `withfilteredset` boxes on the `evselect` "General" page and provide a `filteredset` name, e.g. `src_evlist.fits`, for the resultant file. For this event file, remove the `&&(PATTERN<=4)` phrase for the pn so that single, double, triple, and quadruple events are all included. This filtered event list should be used as input file for the pattern analysis to be performed by `epatplot` (see § 4.7).

4. To extract a background spectrum from a source-free area, first remove the spatial selection (previously defined for the source) from the "Selection expression" window in `xmmselect`. Next repeat step 1) except using now the "Region" option of `ds9` to define the background area and then click the "2D region" button. This will transfer the background region description to the "Selection expression" in `xmmselect`. Finally, repeat step 2) except setting now the `spectrumset` parameter to a different file name, e.g. `bkg_spectrum.fits`.

   Note: Background extraction from a source-free area might be a problem in case of crowded fields. As a general advice for MOS, the source-free background region should be extracted at roughly the same off-axis angle as the position of the source. For pn, the recommendation is to select a source-free background region at the same RAWY position on the chip as the source (RAWY being the long axis of the CCD). So if the pn source is located e.g. at RAWY line 150 on CCD 4, one should aim for selecting the background from around line 150 on e.g. CCD 1.

In case of bright sources observed in **small window** mode or **extended sources**, where virtually no emission free regions exist on the CCD, the user is advised to make use of EPIC background files and tools available through the SOC pages (further details are given below). An alternative approach for MOS might be to extract source-free background regions from the outer CCDs (which always are collecting data in imaging mode).

In the case of EPIC **timing** and **pn burst** mode observations that generally are performed for bright pointlike sources, the background will usually be an issue. Note that in these modes, the RAWY coordinate is not giving spatial but timing information and the source is visible as a bright strip when plotting RAWX against RAWY. In case of the MOS cameras, the timing strip is only 100 pixels wide and if a background spectrum needs to be extracted it should be taken from the outer CCDs which are collecting data in imaging mode. In case of the pn, background regions can be extracted in emission-free strips parallel to the readout direction (i.e. defining a spatial filter expression in RAWX only - a selection in RAWY would incorrectly exclude certain time intervals) excluding the region with registered events from the source.

For the analysis of **pn burst mode** data special care has to be taken in the way of selecting the source and background regions. RAWY may have to be chopped in order to exclude contamination from the source, see e.g. the analysis of burst mode data of the Crab in Kirsch et al. (2006) [6], section 3, available online as XMM-SOC-CAL-TN-0069

http://xmm.esac.esa.int/docs/documents/CAL-TN-0069-1-0.pdf.

Some information on the analysis of extended objects, particularly where a determination of the background is difficult from the indivdual data set, is provided in § 4.8.

4.10.2 Creating response matrices

Analysis of EPIC data products is generally performed by specialised software packages including `Xspec` [17] , `Ximage` [19] or `Xronos` [18] (http://heasarc.gsfc.nasa.gov/docs/xanadu/xanadu.html). In addition to the calibrated products, some of these 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. The response function gives 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 across the field of view.

There are currently two approaches to obtain RMF redistribution matrix files:

1. The user can make use of ready made (canned) response matrices made available by the EPIC team and accessible through the EPIC Response Files page at `http://xmm.esac.esa.int/external/xmm_sw_cal/calib/epic_files.shtml`. They are virtually identical to the files produced by the SAS task `rmfgen`.

   Special care must be taken in choosing the appropriate canned RMFs as they depend on the readout mode, the pattern selection, the observation date and the distance from the readout node where the spectrum was extracted.

2. The user can also create the RMF using the `rmfgen` task (even though `rmfgen` might take some time to complete, depending on the hardware). The input spectrum file contains the necessary ancillary information to allow the correct response to be made. 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 spectrum and writes the result to a specified dataset. It groups response data above a threshold value.

The RMFs are spatially dependent for both MOS and pn. If source extraction regions are large, e.g. for extended sources, timing mode data or complex regions containing many excluded sources, it is important to specify an appropriate number of detector map bins to allow the SAS to calculate an average response matrix (also see § 4.10.4). It is recommended to use 160 bins in each dimension by:

```
rmfgen spectrumset=<spectrum_file> rmfset=<rmf_file> detxbins=160 detybins=160
```

The ARF response file of the EPIC camera shall then be generated by the task `arfgen`. This task calculates an effective area curve as a function of energy, to be used in conjunction with the RMF file generated before. 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 file taking into account the following effects:

1. Telescope effective area including vignetting by the RGA structure for the MOS cameras,

2. EPIC filter transmission,

3. EPIC CCD quantum efficiency,

4. Complex region and pattern selections,

5. Fraction of the PSF in the accumulation region (including chip gap, bad pixel and out of observing window effects),

6. Out-of-time events smearing (pn).

The above effects generally depend on the source position in the EPIC field of view. Spatial response variation over an extended source is also taken into account.

4.10.3 Generating source and background spectra in one go

In the following we will have a look at the alternative approach to generate source and background spectra (optionally together with related response matrices - see § 4.10.2) in a single step: the `xmmselect` product selection “OGIP Spectral Products” starts the meta-task `especget` which is a one-stop task producing all the files necessary for the spectral fitting of an XMM-Newton source. `Especget` runs the tasks `evselect`, `arfgen` and `rmfgen`. It also calculates the size of the source and background areas by calling `arfgen`. The end result is a set of files which can be used directly in a spectral fitting program like e.g. `Xspec`.

The interactive steps needed during the “OGIP Spectral Products” approach (assuming that an EPIC image was created with `xmmselect`) are listed below:

1. In the `ds9` window, create a region for the source of interest. Click once on the `ds9` image and a region circle (default shape) will appear. Click on the region circle and the region will be activated, allowing the region to be moved and its size to be changed. Having created, placed, and sized the region appropriate for the source, one needs to define a region from which to extract the background: this is done in a similar way as before for the source region, but now this second region must be placed in a source-free area (see recommendations on where to place the background region given earlier in this section). Click on the background region and modify its position and size. Via the `ds9` “Region” → “Properties” menu the region types must be defined as “Source” for the source and “Background” for the background region, respectively.

2. Start the spectral product generation by clicking on the “OGIP Spectral Products” product selection in `xmmselect` (cf. Fig. 18). The task `eregionanalyse` performs a source region optimization and the optimized source region is shown in the `ds9` window, the proposed region parameters are given in a popped-up window and the user is asked to confirm if the optimized or the original region should be used for further analysis (or if the spectral product generation is aborted at this stage).

3. If a source region is accepted, the `especget` GUI appears and shows the corresponding spatial selections for the source (parameter `srcexp`) and background (parameter `backexp`) regions (Fig. 22). The user might want to modify the stem for the output filenames (parameter `filestem` on “filenames” related parameter page). For extended sources, the parameter `extendedsource` (on the “effects” related parameter pages) should be set to true. Pressing the “Run” button of the GUI starts the generation of source and background spectra (and response matrices). The meta-task `especget` finishes the processing displaying the generated source spectrum.

Note: `especget` writes the names of the created files into the source spectrum header keywords BACKFILE, RESPFILE, ANCRFILE. These may be automatically read by spectral fitting

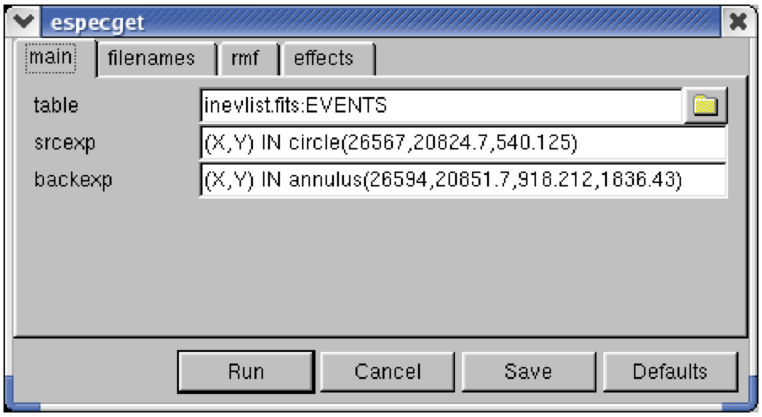


table inevlist.fits:EVENTS
srcexp (X,Y) IN circle(26567,20824.7,540.125)
backexp (X,Y) IN annulus(26594,20851.7,918.212,1836.43)

Figure 22: Graphical User Interface (GUI) of especget showing source and background spatial selection expressions.

programs to link the files and perform area weighted background subtraction. especget applies the following default event selections, for pn '(FLAG==0)&&(PATTERN<=4)' and for MOS '#XMMEA_EM&&(PATTERN<=12)'. especget also takes care of the different spectral ranges and binnings for pn and MOS spectra that need to be applied if canned response matrices are going to be used (see § 4.10.2).

4.10.4 Response files for extended sources

In this section, general recommendations for the generation of the appropriate redistribution matrix and auxiliary response file for extended sources are given.

As the response function of the EPIC cameras is spatially dependent, the rmfgen and arfgen tasks need to know how the source flux is distributed within the extraction region. A detector map mechanism, has been adopted for this purpose. Essentially a grid is placed over the source region and the response parameters are calculated at each point and then averaged taking into account the flux distribution. For point sources the flux distribution is that of the point spread function (default setting detmaptype=psf).

Large, non-pointlike sources should be either approximated by a uniform distribution, setting detmaptype=flat, or modelled accurately by creating an image of the source region. Such a detector map can be created using evselect or xmmselect (see § 4.9). The detector map should entirely cover the region used for extracting the spectrum and should be created with sufficient resolution to sample the variation in flux distribution; a 100 by 100 pixel image may be sufficient. The detector map is supplied to the rmfgen and arfgen tasks by setting

```
rmfgen spectrumset=<spectrum_name> rmfset=<rmf_name> detmaptype=dataset \
       detmaparray=<detector-map_name>
```

and

```
arfgen spectrumset=<spectrum_name> arfset=<arf_name> detmaptype=dataset \
       detmaparray=<detector-map_name> extendedsource=yes
```

The task descriptions give further info and provide some examples for the generation of response matrices for extended sources.

4.11 Detecting EPIC X-ray sources

The SAS metatask `edetect_chain` is a powerful tool to perform source detections on EPIC datasets. It allows searching for sources simultaneously in several energy bands, applying different source detection methods, creation of a final combined source list and the computation of sensitivity maps which contain info on point source detection upper limits. It performs the same kind of analysis as occurs in the SSC pipeline to make PPS products.

The several subtasks called from `edetect_chain`, their main purposes, needed input files and created output data sets are summarized in Table 5.

Task	**Purpose**	**Input files**	**Output files**
`eexpmap`	creation of exposure maps	attitude file, event list, image	exposure maps
`emask`	creation of detection masks	exposure map	detection mask
`eboxdetect` (local mode)	sliding box detection	images, exposure maps, detection mask	box detect source list
`esplinemap`	creation of background maps	image, exposure map, detection mask, local box list	background map
`eboxdetect` (map mode)	box detection using background maps	images, exposure maps, detection mask, bkg. maps	map detect source list
`emldetect`	maximum likelihood fitting	images, exposure maps, bkg. maps, map detect list	final source list
`esensmap`	creation of sensitivity maps	exposure map, detection mask, bkg. map	sensitivity map

Table 5: Schematic overview of the EPIC source detection chain and related input and output data sets

The final output of the EPIC source detection process is a merged source list FITS file which - for every detected source - lists among other parameters the source identification number, the instrument and energy band where it was detected, the source counts, source position and extent, source flux and count rate as well as hardness ratios. The user should read the task description of `emldetect` for a complete list of the output source list columns.

Another output file is a sensitivity map giving (rough) point source detection upper limits (vignetting corrected source count rate corresponding to the likelihood of detection as specified in the parameter file) for each image pixel.

In § 4.12.3 we show an example of how `edetect_chain` and related subtasks actually perform the EPIC source detection.

The task `ewavelet` provides an alternative source detection method based on a Mexican hat wavelet algorithm which is especially well suited for the detection of extended sources. However, the user is cautioned that the determination of source parameters computed by `ewavelet` is not (yet) in general as reliable as that obtained from `edetect_chain`.

4.12 Processing examples of EPIC data

4.12.1 Example of calibrated events files creation

After a proper installation of the SAS software package, EPIC calibrated events lists maybe created and quickly inspected by means of 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
  ```

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

  ```
  setenv SAS_CCF ccf.cif
  ```

- extend the ODF summary file with data extracted from the instrument housekeeping data files and the calibration database creating a new summary file

  ```
  odfingest outdir=$SAS_ODF odfdir=$SAS_ODF
  ```

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

  ```
  emproc &
  epproc &
  ```

  or, alternatively, use the chain tasks

  ```
  emchain &
  epchain &
  ```

  After some time, a set of calibrated event files will have been created in the working directory including the merged event lists of all active MOS and pn CCDs, respectively.

- In order to inspect the content of these files, the Graphical User Interface (GUI) of the `xmmselect` task can be started:

  ```
  xmmselect -d &
  ```

- to select one of these newly created MOS or pn event files as input data set, either enter its name directly in the `table` parameter window or click the button on the right hand side to start the file browser: in the right panel of the dataset browser you can select one of the event lists by double-clicking on it with the left mouse button, then search for the table extension called EVENTS, click on the left mouse button again to select it and finally click on “Ok” at the bottom right of the dataset browser panel.

- press "Run" in the `xmmselect` window. The main `xmmselect` window (Fig 18) appears.
- make use of the 'Product selection' tasks offered via `xmmselect` to generate filtered event lists, images, spectra or rate curves.

4.12.2 Example of EPIC product generation: images, spectra & light curves

The following example illustrates a simple analysis session which aims to extract an image, spectrum and light curve of a point source within the field of view of a single EPIC camera.

We assume that a calibrated EPIC event list (either from the pipeline processing or from running the EPIC chain tasks) is present in the working directory and that the setup steps (`cifbuild`, `odfingest` and the definition of the SAS environment variables, see § 4.12.1) already were performed.

The following analysis steps are described:

1. Create light curves to check for times of background flares.
2. Filter the EPIC event list to exclude bad events and periods of high background.
3. Create images of the EPIC data.
4. Extract source and background spectra for a bright point-like field source.
5. Create RMF and ARF files for this source.
6. Alternatively, make use of the `especget` meta-task to perform spectral analysis steps 4 and 5 in one go.
7. Prepare the spectra for further analysis with `Xspec`.
8. Create a light curve for the source.

Commands to analyse the spectrum using `Xspec` and the light curve using `Xronos` are beyond the scope of this document. Please, check the manuals of these program packages and the SOC provided SAS data analysis threads (available from the SAS web page) for further info.

4.12.2.1 Working from the command line

The user can perform the described analysis with command lines by executing the following steps:

In order to avoid long names and path, it might be convenient to first define the name of the input calibrated event list as a variable:

```
set evfile=/path to PPS directory/P0123700401PNS003PIEVLI0000.FIT
```

Here we are using the pn event list from the pipeline processing of the public Lockman Hole data set.

1. Create a light-curve for the observation to check for flaring high background periods (which are best visible above 10 keV):

```
evselect table=$evfile withrateset=yes rateset=rates.fits \
         timecolumn=TIME timebinsize=100 maketimecolumn=yes \
         expression='#XMMEA_EP && PI in [10000:12000] && (PATTERN==0)'
```

   Note, in case of MOS `'#XMMEA_EM && PI > 10000 && (PATTERN==0)'` shall be used.

   Plot the light-curve (see Fig. 23):

```
dsplot table=rates.fits x=TIME y=COUNTS &
```

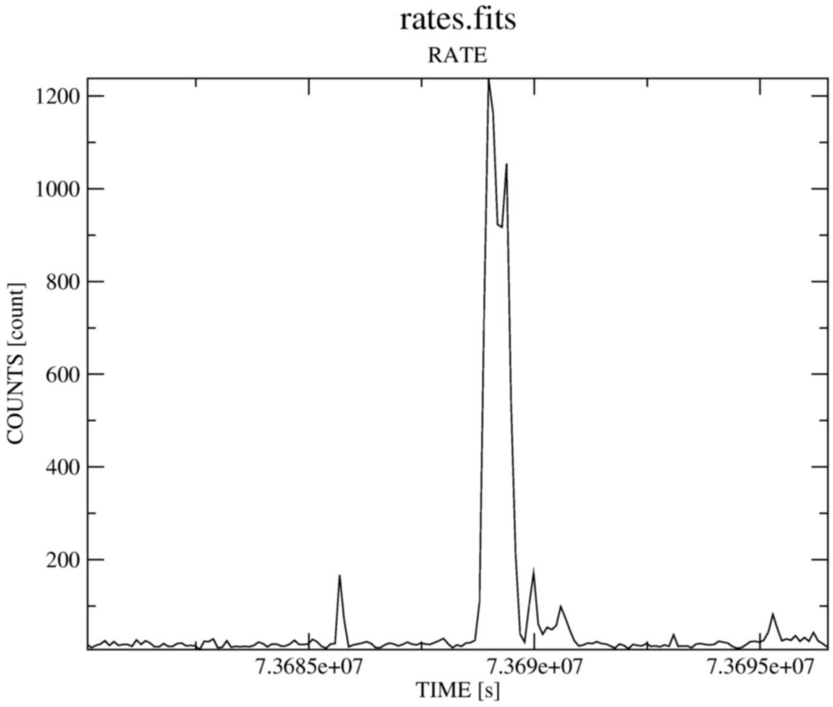


Figure 23: Light-curve of the example data set. Flaring high background periods are clearly visible.

Determine a threshold on the light-curve counts, defining "low background" intervals (in our example: 0.4 counts/s) and create a corresponding good time interval (GTI) file (as `timebinsize=100` was chosen above, one needs to select `COUNTS<=40` for the threshold of 0.4 counts/s):

```
tabgtigen table=rates.fits expression='COUNTS<=40' gtiset=gti.fits
```

Note: the recommended cut value for MOS observations is 0.35 counts/s but the choice of the thresholds strongly depends on the science the user is interested in.

2. Create an event list which is free of high background periods. Also this might be the place to restrict the further analysis to the well calibrated patterns and energy band:

```
evselect table=$evfile withfilteredset=true filteredset=filtered.fits \
         keepfilteroutput=true destruct=true \
         expression='(gti(gti.fits,TIME) && (PI in [100:15000]) \
              && (PATTERN<=12))'
```

Create a new light curve to make sure that the flaring background time intervals were removed:

```
evselect table=filtered.fits withrateset=yes rateset=rates_new.fits \
         timecolumn=TIME timebinsize=100 maketimecolumn=yes \
         expression='#XMMEA_EP && PI in [10000:12000] && (PATTERN==0)'
```

Note, in case of MOS `'#XMMEA_EM && PI > 10000 && (PATTERN==0)'` shall be used.

Plot the new light-curve (see Fig. 24):

```
dsplot table=rates_new.fits x=TIME y=COUNTS &
```

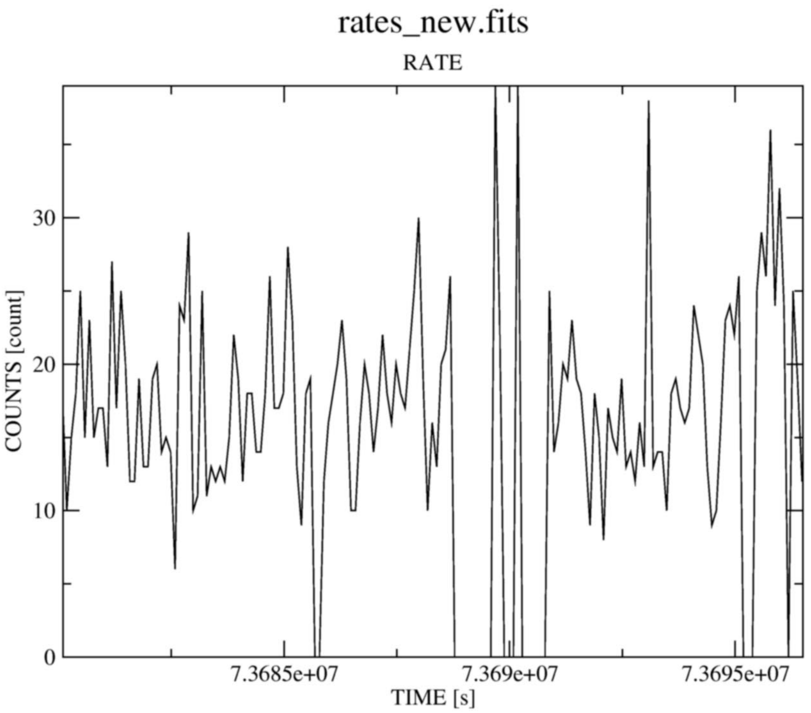


Figure 24: Light-curve of the example data set after removal of flaring high background periods.

From now on, the filtered events file (`filtered.fits`) will be used for further analysis. In case of the example data set, the number of events in the filtered event list is less the half of what the original event list had, a fact which will speed up further processing significantly. The size of the event list (especially in case of bright sources) can be further reduced a lot by also excluding the following columns: RAWX/Y, DETX/Y, PHA.

3. Create a sky image of the filtered data set:

```
evselect table=filtered.fits withimageset=true imageset=image.fits \
         xcolumn=X ycolumn=Y \
         imagebinning=binSize ximagebinsize=80 yimagebinsize=80
```

and display the image with `ds9`:

```
ds9 image.fits &
```

The parameter settings `imagebinning=binSize` and `x/yimagebinsize=80` bin the image into squared pixels of 80 × 0.05 = 4 arcsec (0.05 arcsec being the unit of the X, Y sky coordinate system).

Specifying an additional selection expression of the form `expression='PI in [..:..]'` allows creation of images in different energy bands. E.g. Fig. 25 shows the image in the energy range 0.5-7 keV.

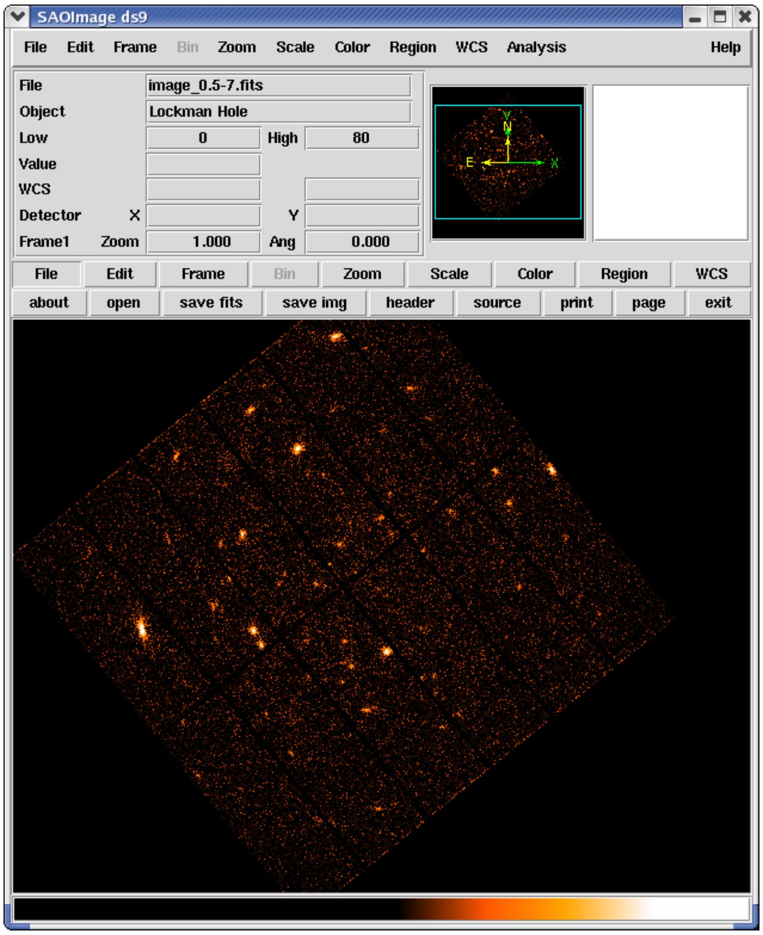
4. Extract source and background spectra for a bright point-like field source: in the displayed image a region around the source can be selected by positioning the cursor on the source center. Keeping the left mouse button pressed, the size of the circular extraction region can be changed. Clicking after-wards again on the region selects it. With the left mouse button the position and size of the region can interactively be changed. It is also possible to change characteristic region parameters via double clicking on the region as well as to save a region via the "Region/Save Regions..." menu in `ds9`.

To apply the specified region as a spatial filter expression in `evselect`, the properties of the region need to be given in physical sky coordinates. To do this, make sure that the coordinates and radius are displayed in physical units in the selection region properties window (see Fig. 26).

In our example, the spatial selection expression for the source spectrum is '(X,Y) IN circle(26285.6,22842.1,600)'. The extraction radius is 30 arcsec. The same values would also be propagated into the selection expression by pressing the "2D Region" button in `xmmselect`.

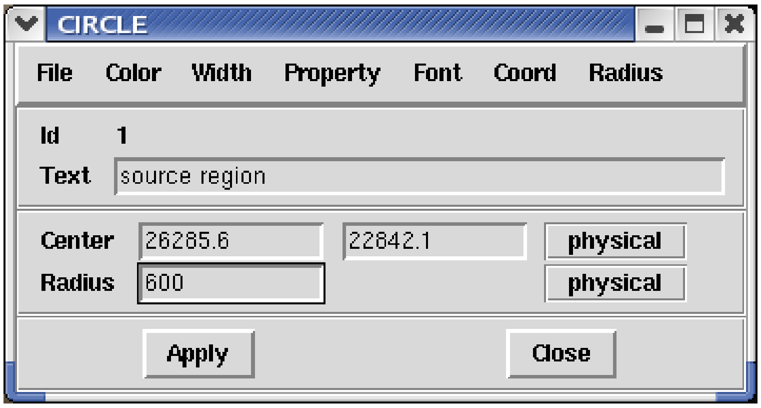
The source spectrum is extracted with the following command line:

```
evselect table=filtered.fits withspectrumset=yes \
         spectrumset=spectrum.fits energycolumn=PI \
         withspecranges=yes specchannelmin=0 specchannelmax=20479 \
         spectralbinsize=5 \
         expression='((X,Y) IN circle(26285.6,22842.1,600)) && \
                     (FLAG==0) && (PATTERN<=4)'
```


SAOImage ds9
File Edit Frame Bin Zoom Scale Color Region WCS Analysis Help
File Edit Frame Bin Zoom Scale Color Region WCS
about open save fits save img header source print page exit
N
E
Y
X

File: image_0.5-7.fits
Object: Lockman Hole
Low: 0 High: 80
Value:
WCS:
Detector X: Y:
Frame1 Zoom: 1.000 Ang: 0.000

Figure 25: Sky image of the example data set in the energy range 0.5-7 keV displayed with `ds9`.



Id: 1
Text: source region
Center: 26285.6 22842.1 physical
Radius: 600 physical

Figure 26: Selection region properties window, popped-up by double-clicking on the region in the main `ds9` window.

Parameters `specchannelmax=20479` and `spectralbinsize=5` are the recommended settings for a pn data set. In the case of MOS data, they should be set to `specchannelmax=11999` and `spectralbinsize=15`. This setup allows the optional usage of the "canned" response matrices.

Including `(FLAG==0)` in the selection expression is recommended for pn. This selection is even more restrictive than the `#XMMEA_EP` event attribute flag as it rejects, in addition, events which are close to CCD gaps or bad pixels. In the case of MOS, the filter expression `#XMMEA_EM` might be sufficient.

For pn we restrict the spectral analysis to the best calibrated single and double events (`(PATTERN<=4)`). In the case of MOS, all valid patterns (`(PATTERN<=12)`) might be included.

The source spectrum can be displayed with the following command:

```
dsplot table=spectrum.fits &
```

In a next step, one needs to extract a background spectrum:

In our pn example the source is located close to the edge of a CCD at a RAWY position of about 138. A surrounding annulus for the background extraction would include the CCD gap and eventually also part of a neighbouring CCD. Hence in this case it might be better to define the background via a circle on the same CCD where the source is located at roughly the same distance from the readout node (same RAWY as the source) placed in a source free region (see Fig. 27 for the selected source and background regions parameters):

```
evselect table=filtered.fits withspectrumset=yes \
         spectrumset=background.fits energycolumn=PI \
         withspecranges=yes specchannelmin=0 specchannelmax=20479 \
         spectralbinsize=5 \
         expression='((X,Y) IN circle(25120.3,21879.9,600)) && \
                     (FLAG==0) && (PATTERN<=4)'
```

The background spectrum can be displayed again with the following command:

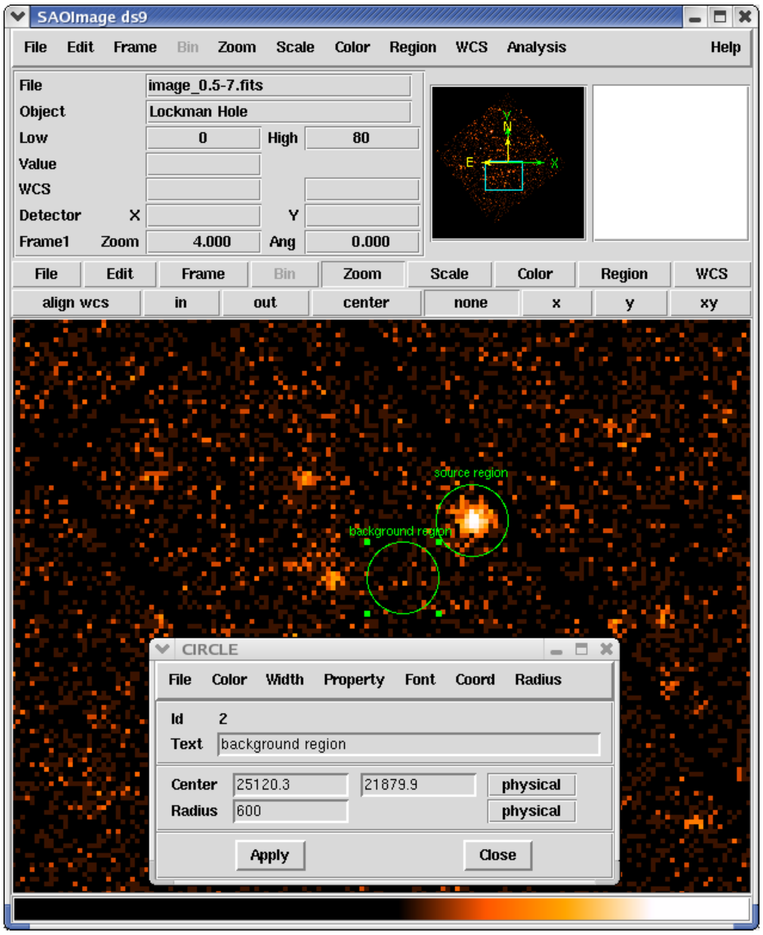
```
dsplot table=background.fits &
```

In the next step, the area of the extraction regions used to make the source and background spectral files must be calculated taking into account CCD boundaries and bad pixels. The area is written into the header of the SPECTRUM table of the input file as the keyword BACKSCAL:

```
backscale spectrumset=spectrum.fits badpixlocation=filtered.fits

backscale spectrumset=background.fits badpixlocation=filtered.fits
```

Note, if spectra are created via the `xmmselect` task, `backscale` will have automatically been applied "on the fly" during the product generation process.

5. Create response matrix files (Redistribution Matrix File (RMF) and Ancillary Response File (ARF)) using the `rmfgen` and `arfgen` tasks:


SAOImage ds9
File Edit Frame Bin Zoom Scale Color Region WCS Analysis Help
File Edit Frame Bin Zoom Scale Color Region WCS
align wcs in out center none x y xy
source region
background region
CIRCLE
File Color Width Property Font Coord Radius
Apply
Close

File image_0.5-7.fits
Object Lockman Hole
Low 0 High 80
Value
WCS
Detector X Y
Frame1 Zoom 4.000 Ang 0.000

Id 2
Text background region
Center 25120.3 21879.9 physical
Radius 600 physical

Figure 27: Selection regions for the extraction of source and background spectra.

```
rmfgen spectrumset=spectrum.fits rmfset=spectrum.rmf
```

An alternative approach to obtain a RMF file is to use the ready-made "canned" response matrices available from the rmf files at the following URL: `http://xmm.esac.esa.int/external/xmm_sw_cal/calib/epic_files.shtml`.

```
arfgen spectrumset=spectrum.fits arfset=spectrum.arf \
       withrmfset=yes rmfset=spectrum.rmf \
       badpixlocation=filtered.fits detmaptype=psf
```

Note, `arfgen` reads the pattern range from the data subspace (DSS) information in the spectrum dataset, and accumulates the quantum efficiency curves over those patterns, which are then combined to the other constituents of the ARF. Be aware that the entire range of allowed patterns (0-12 for the pn and 0-31 for the MOS, respectively) are assumed if no pattern range was found in the DSS.

6. Alternatively, make use of the `especget` meta-task to perform spectral analysis steps 4 and 5 in one go

Assuming that the user wants to use the same extraction regions for the source and background spectra as given above, the generation of spectra and related response files can be performed in one go with the following command:

```
especget table=filtered.fits filestem=mysource \
         srcexp='(X,Y) IN circle(26285.6,22842.1,600)' \
         backexp='(X,Y) IN circle(25120.3,21879.9,600)'
```

By default the following event selection is added to the spatial selection expressions: for pn `'(FLAG==0)&&(PATTERN<=4)'` and for MOS `'#XMMEA_EM&&(PATTERN<=12)'`. The spectra are generated with spectral ranges and binnings automatically set in a way that canned response matrices (§ 4.10.2) could be used. The computation of the area of the extraction regions is performed "on the fly". In addition, `especget` writes the names of the created files into the source spectrum header keywords BACKFILE, RESPFILE, ANCRFILE. These may be automatically read by spectral fitting programs to link the files and perform area weighted background subtraction.

The parameter `filestem` defines the names of the produced output files. In the example given above, `especget` generates the following output files:

- mysource_src.ds: the source spectrum
- mysource_bgd.ds: the background spectrum
- mysource_src.arf: the source related Ancillary Response File (ARF)
- mysource_src.rmf: the source related Redistribution Matrix File (RMF)

Note, if `xmmselect` is used to generate "OGIP Spectral Products" (see § 4.10.1), the user can interactively define source and background regions and (before `especget` is started) a source region optimization is performed via the task `eregionanalyse`.

7. Prepare the spectra for further analysis with `Xspec`

The created spectrum should be grouped (binned) now depending on the available signal to noise ratio in the data and the science the user wants to perform. The `FTOOLS` task `grppha` allows regrouping of the source spectrum based on different criteria. Here we show an example of how to perform a grouping based on a minimum number of counts in each bin.

```
grppha
Please enter PHA filename[] mysource_src.ds
Please enter output filename[] mysource_src.grp
GRPPHA[] group min 25
GRPPHA[] exit
```

If the source spectrum was **not** generated with `especget`, `grppha` also offers a possibility to associate the RMF, ARF and background spectrum with the source spectrum which is convenient to do before using `Xspec` for further analysis:

```
grppha
Please enter PHA filename[] spectrum.fits
Please enter output filename[] spectrum.grp
GRPPHA[] group min 25
GRPPHA[] chkey BACKFILE background.fits
GRPPHA[] chkey RESPFILE spectrum.rmf
GRPPHA[] chkey ANCRFILE spectrum.arf
GRPPHA[] exit
```

Note, that after having read the grouped source spectrum into `Xspec`, the `ignore bad` command shall be issued to ignore spectral bins which did not fulfill the grouping condition used in `grppha`.

8. Create a light-curve for the source:

```
evselect table=filtered.fits withrateset=yes \
         rateset=lightcurve.fits timecolumn=TIME timebinsize=1 \
         expression='((X,Y) IN circle(26285.6,22842.1,600))'
```

The `Xronos` program package can now be used to produce a binned light-curve, to calculate a power spectrum, search for periodicities etc.

The data analysis based on command lines was described here, as combining command lines into a script might offer a good method for further re-performing of (part of) a data analysis session (but see also § 4.12.4 for a discussion of possible advantages of a GUI based analysis).

4.12.2.2 Inspection of spectra or timeseries using the command line

Although `xmmselect` interacts in a convenient way with the external plotting program `grace`, the user might want to plot, inspect or export spectra and timeseries which have been created in an already closed `xmmselect` session or via `evselect` from the command line. As the produced output files are FITS files, the `FTOOLS` task `fv` and `fplot` offer many possibilities here, but also the SAS task `dsplot` can be used:

- The spectra or timeseries table can be plotted with the `grace` package using the `dsplot` command. We assume that the spectrum or timeseries are named `product.fits`:

  ```
  dsplot table=product.fits
  ```

- Postscript files can be created.

  ```
  dsplot table=product.fits plotter='gracebat -printfile product.ps'
  ```

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

  ```
  dstoplot table=product.fits > product.asc
  ```

4.12.3 Source detection example

In the following section (§ 4.12.3.1) we show an example on how to perform EPIC source detection calling in sequence the individual `edetect_chain` subtasks. This gives detailed information about the detection process. If, however, the user is happy with the default parameter settings for a standard source detection session, he/she might skip this section and continue directly with § 4.12.3.2 where the source detection is performed in one go via the `edetect_chain` task.

In the following we assume that the user has created a calibrated EPIC event list (or uses one from the pipeline processing) and has cleaned it for high flaring background periods (see § 4.4.4 and analysis step 1 and 2 in § 4.12.2.1). The assumed name of this filtered event list is `filtered.fits`.

The user can make use of the pipeline produced images (IMAGE_1 to IMAGE_5, see Table 4). Alternatively, images in these or different energy bands need to be generated first with e.g. the following command (see also § 4.12.2.1, analysis step 3):

```
evselect table=filtered.fits withimageset=true imageset=image_b1.fits \
         xcolumn=X ycolumn=Y imagebinning=binSize ...\
```

in case of MOS, a binsize of 22 = 1.1 arcsec fits well the camera pixel size, all valid patterns should be included, and the MOS specific bit mask should be used:

```
         ... ximagebinsize=22 yimagebinsize=22 \
         expression='(PI in [200:500])&&(FLAG==0)&&(PATTERN in [0:12])&&#XMMEA_EM'
```

in case of the pn, a binsize of 82 = 4.1 arcsec fits well the camera pixel size, all single and double events should be included, and the PN specific bit mask should be used:

```
         ... ximagebinsize=82 yimagebinsize=82 \
         expression='(PI in [200:500])&&(FLAG==0)&&(PATTERN in [0:4])&&#XMMEA_EP'
```

To perform source detection in all the standard pipeline processing energy bands, four more images need to be generated, modifying the energy selection expression in the following way: for `image_b2.fits` use (PI in [500:2000]), for `image_b3.fits` use (PI in [2000:4500]), for `image_b4.fits` use (PI in [4500:7500]) and for `image_b5.fits` use (PI in [7500:12000]). The energy bands are defined via PI intervals in units of eV. Note that such a multi-band approach to source detection is not essential (it could also be performed in a single energy band).

With images extracted in the required energy bands, the user now can start the EPIC source detection process, either performing it step by step (§ 4.12.3.1) or in one go via the `edetect_chain` meta task (§ 4.12.3.2).

4.12.3.1 EPIC source detection performed via single task commands

The user can further make use of the pipeline produced exposure maps (EXPMAP1 to EXPMAP5, see Table 4). Alternatively, exposure maps in different energy bands need to be generated with the `eexpmap` task. This task makes use of calibration information on the spatial quantum efficiency, filter transmission, mirror vignetting and field of view to calculate for each attitude bin the exposure time values projected onto the sky. Assuming that the attitude information is available from the file `attitude.fits` (created by the `atthkgen` task; alternatively, the attitude file exists as a pipeline product with file identification ATTTSR), exposure maps `image_biexp.fits` (i = 1 to 5) are generated in the following way:

```
eexpmap attitudeset=attitude.fits eventset=filtered.fits imageset=image.fits \
        expimageset='image_b1exp.fits image_b2exp.fits image_b3exp.fits \
                     image_b4exp.fits image_b5exp.fits' \
        pimin='200 500 2000 4500 7500' pimax='500 2000 4500 7500 12000'
```

The imageset `image.fits` can be any of the already created energy band images. It is only used to extract information from the FITS header about the Instrument ID, Mode/Submode, filter ID, GTI and WCS keywords.

In the next step, a detection mask needs to be generated. The detection mask is a FITS image containing the integer values 0 and 1 where 1 marks the image area on which subsequent source searching will be performed. The following command should be performed:

```
emask expimageset=image_biexp.fits detmaskset=mask.fits
```

The expimageset `image_biexp.fits` can be any of the previously created exposure maps. The user should inspect the created mask with `ds9` to see if the area selection is appropriate for further source detections. If not, the values of the threshold parameters `threshold1` and `threshold2` can be tuned. By default, the detection mask contains 0 where the value of the exposure map is less than 30% of the maximum exposure (`threshold1`) and where the gradient of the exposure map is steeper than 0.5 (`threshold2`).

Now all necessary files have been generated to start the sliding box source detection in the so-called local mode. The purpose of the local detection step is to provide an input list of source positions for task `esplinemap` (see below) which then constructs a background map from the non-source locations. Source counts are accumulated from a 3×3 or 5×5 (controlled by parameter `boxsize`, default value = 5, also used in the SSC pipeline) pixel window and the background is determined from the surrounding 40 (7×7 pixel window) or 56 pixels (9×9 pixel window), respectively. Detection of moderately extended objects (up to several times the PSF size) is achieved by searching the image in up to four consecutive detection runs each doubling the pixel size (parameter `nruns`, default value = 3).

Following the definition which was, e.g., used by the ROSAT mission, detection likelihoods (per energy band and total) are given for each source in the form $L = -\ln p$ where p is the probability of Poissonian random fluctuation of the counts in the detection cell which would have resulted in at least the observed number of source counts. The value of p is calculated as a function of raw source counts and raw background counts in the detection box (see `eboxdetect` task description for further info on the detection algorithm).

The local mode sliding box source detection is performed executing the following command:

```
eboxdetect usemap=no likemin=8 withdetmask=yes detmasksets=mask.fits \
           imagesets='image_b1.fits image_b2.fits image_b3.fits \
                      image_b4.fits image_b5.fits' \
           expimagesets='image_b1exp.fits image_b2exp.fits image_b3exp.fits \
                         image_b4exp.fits image_b5exp.fits' \
           pimin='200 500 2000 4500 7500' pimax='500 2000 4500 7500 12000' \
           boxlistset=eboxlist_l.fits
```

The local mode detection is switched on setting `usemap=no`. It is recommended to use a detection threshold of `likemin=8` to provide a complete source list as input for `esplinemap`. Note, the parameter `ecf` will need to be specified if one already at this stage of the source detection chain is interested in the computation of source fluxes (see section on `emlselect` below).

In the next step, the task `esplinemap` makes use of the `eboxdetect` generated local mode source list to derive spline background maps from the non-source regions. Sources found in the local detection step at significance levels (column SIGMA of the source list) exceeding a user-specifiable threshold (input parameter `mlmin`) are removed from the image using a suitable PSF and source brightness dependent cut-out radius (determined to be the radius at which each source contributes more than a user-specifiable number of counts/arcsec^2 to the background; parameter `scut`):

```
esplinemap bkgimageset=image_b1bkg.fits imageset=image_b1.fits \
           boxlistset=eboxlist_l.fits scut=0.005 nsplinenodes=16 \
           withdetmask=yes detmaskset=mask.fits \
           withexpimage=yes expimageset=image_b1exp.fits
```

This commands needs to be performed for all energy bands (b1 to b5) separately. An optionally diagnostic output photon image where sources have been masked out (the so called cheesed image) can be created setting `withcheese=yes` and defining the name of the cheesed image via `cheeseimageset`. In addition, `esplinemap` is able to determine the background caused by OoT events registered during the readout process of the pn CCDs (§ 4.6). If the flag `withootset` is set, the photon event table specified in `ooteventset` is read and the background caused by OoT events is included in the output background map.

To improve the detection sensitivity reached before with the local mode detection, the sliding box source detection now should be performed in map mode (`usemap=yes`). Here the background will be taken from the background maps that were determined by `esplinemap`:

```
eboxdetect usemap=yes likemin=8 withdetmask=yes detmasksets=mask.fits \
           imagesets='image_b1.fits image_b2.fits image_b3.fits \
                      image_b4.fits image_b5.fits' \
           expimagesets='image_b1exp.fits image_b2exp.fits image_b3exp.fits \
                         image_b4exp.fits image_b5exp.fits' \
           bkgimagesets='image_b1bkg.fits image_b2bkg.fits image_b3bkg.fits \
                         image_b4bkg.fits image_b5bkg.fits' \
           pimin='200 500 2000 4500 7500' pimax='500 2000 4500 7500 12000' \
           boxlistset=eboxlist_m.fits
```

The next step is to generate the final source list with the task `emldetect`: a simultaneous maximum likelihood PSF fit is performed to the source count distribution in all energy bands of each involved EPIC telescope with the following free parameters: source location (RA, Dec), source extent (Gaussian sigma) and source count rate. Source location and extent are constrained to the same best-fit value in all energy bands whereas count rates are the best-fit values in each band. The PSF fitting may either be performed in single source (default) or in multi-source mode (parameter `nmaxfit` > 1). Unless `nmaxfit` > 1, `emldetect` will not detect new sources, but characterizes the detected sources by making a PSF fit. Energy conversion factors (ECFs) can be supplied for a conversion of source count rates into flux values. The ECFs for each energy band depend on the pattern selection and the filter used during the observation (see SSC-LUX-TN-0059). Here we are assuming a pn thin filter observation where patterns 0 - 4 are considered:

```
emldetect imagesets=’image_b1.fits image_b2.fits image_b3.fits \
                     image_b4.fits image_b5.fits’ \
          expimagesets=’image_b1exp.fits image_b2exp.fits image_b3exp.fits \
                        image_b4exp.fits image_b5exp.fits’ \
          bkgimagesets=’image_b1bkg.fits image_b2bkg.fits image_b3bkg.fits \
                        image_b4bkg.fits image_b5bkg.fits’ \
          pimin=’200 500 2000 4500 7500’ pimax=’500 2000 4500 7500 12000’ \
          boxlistset=eboxlist_m.fits \
          ecf=’10.596 6.816 2.054 0.995 0.259’ \
          mlmin=10 mllistset=emllist.fits
```

Especially in cases where an expected X-ray source could not be detected by the source detection steps described above, the user might be interested in using task `esensmap` to generate a sensitivity map giving (rough) point source detection upper limits (vignetting corrected source count rate corresponding to the likelihood of detection as specified in the parameter file) for each image pixel. The task `esensmap` may either be called for individual energy bands or combinations of energy bands and instruments. The upper limits are calculated by assuming Poissonian count statistics in each 3×3 pixel detection cell, using the exposure and background values read from the input images. In the case of multiple input energy bands the upper limits are expressed in units of counts per seconds for the combined band (i.e. they refer to the detection sensitivity which would be achieved by adding up the photons observed in the individual bands). Below we give an example on how to compute a sensitivity map for a single energy band:

```
esensmap expimagesets=image_b1exp.fits bkgimagesets=image_b1bkg.fits \
         detmasksets=mask.fits mlmin=10 sensimageset=image_b1sen.fits
```

This command eventually should be performed for all energy bands (b1 to b5) separately.

Alternatively, if one is interested in a sensitivity map for the combined energy range, the command to be issued could be:

```
esensmap expimagesets=’image_b1exp.fits image_b2exp.fits image_b3exp.fits \
                       image_b4exp.fits image_b5exp.fits’ \
         bkgimagesets=’image_b1bkg.fits image_b2bkg.fits image_b3bkg.fits \
                       image_b4bkg.fits image_b5bkg.fits’ \
         detmasksets=mask.fits mlmin=10 sensimageset=image_sen.fits
```

4.12.3.2 Running the EPIC source detection chain

Assuming that the user either wants to make use of the pipeline generated image products or that he/she has created images in different energy bands, the task edetect_chain can be called to perform all source detection steps described in § 4.12.3.1 via a single command.

The attitude information is assumed to be available from the file attitude.fits (created by the atthkgen task). The example below further assumes that the source detection will be performed on a background cleaned pn thin filter observation (the name of the calibrated and filtered event list being filtered.fits). As noted above, the ECFs for each energy band depend on the EPIC camera, the pattern selection and the filter used during the observation (see SSC-LUX-TN-0059).

The complete EPIC source detection chain is started with the following command:

```
edetect_chain eventsets=filtered.fits attitudeset=attitude.fits \
              imagesets='image_b1.fits image_b2.fits image_b3.fits \
                         image_b4.fits image_b5.fits' \
              pimin='200 500 2000 4500 7500' pimax='500 2000 4500 7500 12000' \
              ecf='10.596 6.816 2.054 0.995 0.259' \
              eboxl_list=eboxlist_l.fits eboxm_list=eboxlist_m.fits \
              esp_nsplinenodes=16 eml_list=emllist.fits esen_mlmin=10
```

After completion of the edetect_chain task, the following output files have been created: exposure maps (from task eexpmap), detector mask images (from task emask), background maps (from task esplinemap), eboxdetect source list (local mode), eboxdetect source list (map mode), emldetect source list, source maps, if eml_withsourcemap=yes (from task emldetect) and sensitivity maps (from task esensmap).

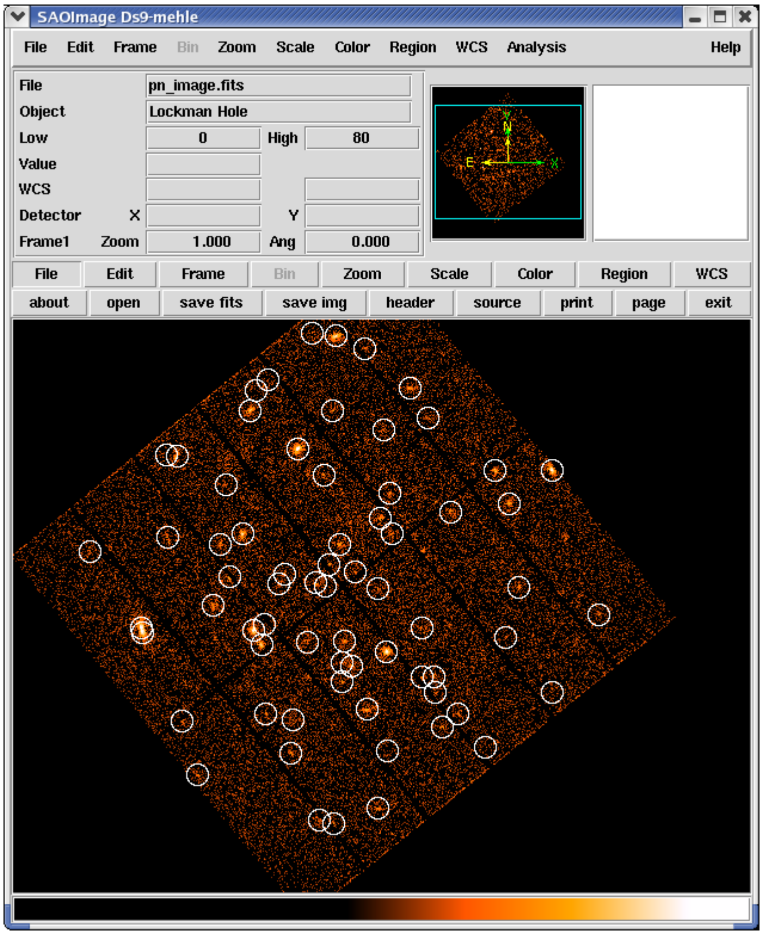
To verify the quality of the performed source detection, the user might want to check generated maps with the image viewer program ds9. In addition, the task srcdisplay can be used to overlay all the different derived source lists onto an image. Circles are used to depict the source positions. The radius of these circles can be set (in degrees) using the sourceradius parameter. An optional ID label can also be displayed, corresponding to the row number of that source in the input source list. This can be enabled through the uselabel parameter. Plotting the ID helps the user to refer back to source properties documented in the source list.

As the plotted circles are in fact ds9-type regions, they 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 (set to regionfile.txt by default).

To show e.g. the generated merged source list overlaid onto an EPIC image (called pn_image.fits in the example) the following command can be issued:

```
srcdisplay boxlistset=emllist.fits imageset=pn_image.fits sourceradius=0.01
```

Fig. 28 shows an example of such a srcdisplay run.


SAOImage Ds9-mehle
File Edit Frame Bin Zoom Scale Color Region WCS Analysis Help
Value
WCS
Detector X Y
N
E
X
File Edit Frame Bin Zoom Scale Color Region WCS
about open save fits save img header source print page exit

File pn_image.fits
Object Lockman Hole
Low 0 High 80
Frame1 Zoom 1.000 Ang 0.000

Figure 28: Resulting ds9 display of a `srcdisplay` command issued to overlay a final source list onto a pn image.

4.12.4 Working with the GUI

The user can perform the described analysis (instead of typing commands) also with the graphical user interface (GUI) of the SAS. In general, the analysis via the GUI might be judged more user friendly for an interactive analysis session. But the GUI, too, allows the saving of performed analysis steps in a log file (default name “sas_log”) as well as the saving of performed commands as a shell script (via the “File: Save as script” menu) which later can be used for reprocessing.

For that purpose, the `sas` command shall be run first. The SAS GUI appears and the user can select a task. The `xmmselect` task interacts in a convenient way with the `evselect` task to extract images, spectra, timeseries or filtered event lists.

5 Analysis of RGS spectrometer data

The SAS provides a number of tools for analysis of data from the RGS1 and RGS2 spectrometers, starting from the raw data in the ODF and finishing with one or more high-resolution X-ray spectra and their associated RMF response matrices. The RGS meta-task `rgsproc` , which runs all the way from ODF to RMF, is broken down into a sequence of component tasks that deal with successive stages of the work required, allowing the user flexibility to make adjustments where necessary. While it is possible simply to type `rgsproc` at the command line and get reasonable results, the investment of some thought and advanced planning usually gives significant improvement in the quality of the final products. Help is always immediately available through the SAS on-line help system:

http://xmm.esac.esa.int/sas/current/doc/packages.rgs.html

of which the `rgsproc` entry is a suitable RGS starting point:

http://xmm.esac.esa.int/sas/current/doc/rgsproc/index.html.

Detailed descriptions of the methods used and parameter specifications are given there and liberal use of those pages is recommended. As a complement to this SAS manual chapter, RGS data analysis threads at http://xmm.esac.esa.int/sas/current/documentation/threads/ [11] and the RGS chapter of the NASA/GSFC ABC guide (last updated in 2004)[13] may also be consulted. Please do not hesitate to contact the helpdesk with any comments, complaints or calls for improvements in any of these documents.

In common with the vast majority of XMM data [8], RGS files are in FITS format so that it is useful to have at hand general-purpose tools such as FTOOLS including `fv` in particular [15] to inspect files. The spectra and response matrices that are the final products of the RGS SAS procedures are intended for further analysis outside the SAS with standard tools such as XSPEC [17], SPEX [20] or ISIS [21].

The results of an automatic version of `rgsproc` that is run on all ODFs are also available via the archive and will have been sent to observers with the original ODF distribution. These automatically generated results, known as the PPS products, are discussed in section 5.15 below. Naturally, these procedures are run without human intervention and thus are not expected to emulate the quality that can often be produced by the individual attention of a well-informed and creative astronomer. Experience shows, however, that they are useful, especially when used in conjunction with the other XMM instruments. It is most likely the case, though, that an RGS observer will benefit from running the analysis again starting from the ODF. This chapter is organised roughly along the lines of a typical session of RGS work.

Immediately, the most important things to be considered are

- the source coordinates;
- the source extent;
- the behaviour of the background;
- calibration data;

and RGS analysts should make explicit decisions about each of these things for every single observation on which they work. SAS v7.0 coincided with the release of significant improvements in the calibration of the RGS effective area, particularly at long wavelengths.

5.1 Source coordinates

RGS data are individual events. The wavelength, λ, assigned to a source photon of spectral order m is determined by the diffraction equation

$$m\lambda = d(\cos\beta - \cos\alpha)$$

where d is the grating spacing. The dispersion angle β is fixed by the photon's position on the CCD detector and the constant geometry of the detector and grating assemblies. The only remaining variable, therefore, is α, the angle of incidence of light on the gratings so that the X-ray photon wavelength reduces to a function of the source celestial coordinates $RA(J2000), DEC(J2000)$ only

$$\lambda = \lambda(\alpha) = \lambda(RA(J2000), DEC(J2000))$$

and it is the analyst's responsibility to make sure that they are as accurate as possible. Differentiation of the diffraction equation shows that a systematic error of up to 2.3mÅ is introduced for every arcsecond error in the source coordinates. The PPS procedure makes an automatic choice between the proposal's pointing coordinates and X-ray coordinates derived from the simultaneous EPIC images, both of which have been known to be unreliable for a number of reasons. It is therefore worthwhile to take as much care as possible over the coordinates used in the analysis including proper motion as necessary. The SIMBAD [23] and NED [24] reference databases are excellent in this regard. The source coordinates required by `rgsproc` are thus best explicitly supplied by the analyst in decimal degrees to six decimal places, wherever possible. In the absence of any better information, the proposal pointing coordinates are used by default with whatever consequences for the wavelength calibration.

5.2 Source extent

The SAS is easy to use for a single dominant point source in the RGS apertures. In this case, the PPS procedure is also successful most of the time. The means also exist to treat more complex source configurations and, with proper care, both multiple point sources and extended sources may be tackled as discussed in below in section 5.9 and section 5.10 respectively. Not surprisingly, the PPS is of limited use in these circumstances.

5.3 Background behaviour

The instrumental background also requires attention, particularly because of its variability. Most of the time the background is low enough for the RGS to be more-or-less photon limited for the detection of point-source spectra over most of its bandwidth once the proper events selections have been made. However, solar flares and other particle events have caused changes of two or more orders of magnitude in the background rate on time scales short in comparison with the typical length of an observation. In such unfortunate but rare cases the signal spectrum can be completely swamped but an entire observation is seldom affected. It is imperative, therefore, to make an assessment of the background's behaviour during an observation and exclude any periods that are too badly contaminated following, for example, the procedure described below in section 5.7.3.

5.4 Calibration data

In order to derive physical parameters from the data in an observation, the SAS uses the RGS instrument model encapsulated in the calibration data stored in the CCF. As mentioned above, SAS v7.0 coincided with the release of significant improvements in the RGS calibration. These data are not supplied with the ODF or PPS products but are available for immediate download [9], along with procedures to ensure your calibration data remain up-to-date and the latest instrument news [10]. The CCF is an accumulation of all the calibration data ever released so that one of the initial SAS tasks is to use the task `cifbuild` to build a calibration index file `ccf.cif` that enumerates which CCF components apply to an observation. A more complete discussion of these matters appears elsewhere in this manual in section 2.3.3 and section 3.11.

5.5 Taking delivery of RGS data

It is a good idea to adopt a systematic naming scheme for a hierarchy of XMM directories in which to store and distinguish raw ODF data, pipeline PPS products, and in this case RGS files of your own making. For example, when dealing with the Mkn421 observation with ID 0136540101 in revolution 259 appropriate directories might be called

```
/data/XMM/0136540101/ODF

/data/XMM/0136540101/PPS

/data/XMM/0136540101/RGS
```

These directories will be used in some examples below. After your XMM-Newton data have been delivered on CD-ROM or downloaded from the archive, the contents of the data package should be verified with reference, should the need arise, to the XMM Data Files Handbook [8] which contains details of the structure of all XMM files.

5.5.1 Raw RGS Data Files in the ODF

Raw data of all seven XMM instruments, including RGS1 and RGS2, are to be found in a single directory `\odf`. If the delivery includes data from more than one observation, there will be a separate directory for each. The ODF names for RGS data are as follows:

- mmmm_iiiiiijjkk_aabeeeccfff.zzz, where
  - mmmm: revolution orbit number
  - iiiiiijjkk: observation number
  - aa: detector ID (R1 - RGS1, R2 - RGS2, SC - XMM spacecraft)
  - b: flag for scheduled (S) or unscheduled (U) observations, or (X) for general use files
  - eee: exposure number within the observation or “000”
  - cc: CCD identifier or “00”
  - fff: data identifier shown in Table 6
  - zzz: Format (FITS - FIT, ASCII - ASC)

Data Identifier	Contents
RGS files	
AUX	on-board processing statistics
SPE	raw event list for one CCD
DII	diagnostic images
D1H	CCD readout settings
D2H	CCD readout settings
PFH	housekeeping data
ODX	pixel offset data
XMM files	
ATS	spacecraft attitude history

Table 6: ODF data file identifiers relevant for RGS analysis

5.6 Running the RGS processor rgsproc

The work done by `rgsproc` (see: http://xmm.esac.esa.int/sas/current/doc/rgsproc/) falls into two main parts: first the construction of a calibrated event list; and second the selection of events to be accumulated into spectra for one or more of the entries in a source list. Event construction starts from the spatial coordinates and pulse heights among the 6 columns of raw readout data stored in one file for each CCD and ends with 15 calculated columns in one merged file for each RGS. A CCD delivers a few hundred thousand events in a typical observation and the merged file contains, therefore, perhaps a few million before event selection begins. The final selected event file can contain less than half of the merged events after exclusion of hot pixels and columns, although after the RGS instruments were cooled in November 2002 the hot stuff became much less numerous. Evidently, the rejection of hot columns and pixels is a fundamental part of the analysis. The further selection of events for source and background spectra then depends on the position of the source within the RGS apertures as determined by its coordinates held in the source list and the extent of the source and background selection regions chosen explicitly or by default.

Here is an example `rgsproc` run on Mkn421 using accurate NED source coordinates in the directory scheme discussed above. As discussed in detail in section 2.3, the environment variables `SAS_CCFPATH`, `SAS_ODF` and `SAS_CCF` supply the SAS with calibration and data information so that `rgsproc` , which always writes its output files to the working directory, may be run in any location.

```
setenv SAS_CCFPATH /data/XMM/CCF
cd data/XMM/0136540101/ODF
setenv SAS_ODF $PWD
cifbuild
setenv SAS_CCF $PWD/ccf.cif
odfingest
setenv SAS_ODF $PWD/0259_0136540101_SCX00000SUM.SAS
cd ../RGS
rgsproc withsrc=yes srclabel=Mkn421 \
                        srcstyle=radec srcra=166.113808 srcdec=+38.208833
```

The task `odfingest` needs only to be run once after receipt of the ODF, while a new `ccf.cif` calibration index file should be regenerated with `cifbuild` whenever a new relevant CCF component is available. `rgsproc` has about a hundred potentially adjustable parameters

http://xmm.esac.esa.int/sas/current/doc/rgsproc/node25.html.

In this case, the source coordinates only have been specified, leaving all other parameters at their default values, some of the more important of which are shown in Table 7.

As it is running, `rgsproc` gives regular messages about its progress, although these are not designed to prompt any action except in the unlikely event of error. The final outcome, as shown in table 8, is a set of 12 FITS files for each RGS consisting of

- 2 event lists
- 1 source list

parameter	default value
`entrystage`	`1:events`
`finalstage`	`5:fluxing`
`orders`	1 2
`betabinning`	`binSize`
`betabinref`	+0.03578524 radians
`betabinwidth`	+0.00001208 radians
`nbetabins`	3400
`xdispbinning`	`binSize`
`xdispbinref`	−0.0009126 radians
`xdispbinwidth`	+0.0000108 radians
`nxdispbins`	170
`bkgcorrect`	`no`
`keepcool`	`yes`

`rejflags`	`BAD_SHAPE`
	`ON_BADPIX`
	`NEXT_TO_BADPIX`
	`BELOW_ACCEPTANCE`
	`NEXT_TO_CCD_BORDER`

`xpsfincl`	95%
`xpsfexcl`	98%
`pdistincl`	95%
`rmfbins`	4000

Table 7: Default values of key `rgsproc` parameters.

- 2 lists of good time intervals
- 1 exposure map
- 3 1st order files
- 3 2nd order files

and 3 other general-purpose FITS files. Since SASv6.0, source spectra are not background corrected by default so that the corresponding background spectra must explicitly be included in subsequent analysis with XSPEC, for example. The RGS files have names of the form

- P0136540101RnSxxxCCCCCCm003.FIT, where

  0136540101 is the observation number;
  Rn is R1 for RGS1 or R2 for RGS2;
  xxx is the exposure number, 001 for RGS1 or 002 for RGS2 in this case;
  CCCCCC is the product type;
  m is the negative order number, 1 and 2 by default;
  003 is the number in the SRCLI list of sources.

If a file does not apply to a particular exposure, “Sxxx” is replaced by “X000”. Similarly, ”0” and “000” show that the order and source number, respectively, are deemed irrelevant.

File	Contents
P0136540101R1S001BGSPEC1003.FIT	RGS1 1st order background spectrum
P0136540101R1S001BGSPEC2003.FIT	RGS1 2nd order background spectrum
P0136540101R1S001EVENLI0000.FIT	RGS1 screened event list
P0136540101R1S001EXPMAP0000.FIT	RGS1 exposure map
P0136540101R1S001RSPMAT1003.FIT	RGS1 1st order response matrix
P0136540101R1S001RSPMAT2003.FIT	RGS1 2nd order response matrix
P0136540101R1S001merged0000.FIT	RGS1 raw merged event list
P0136540101R1S001SRCLI_0000.FIT	RGS1 source list
P0136540101R1S001SRSPEC1003.FIT	RGS1 1st order source+background spectrum
P0136540101R1S001SRSPEC2003.FIT	RGS1 2nd order source+background spectrum
P0136540101R1X000hkgti_0000.FIT	RGS1 housekeeping good time intervals
P0136540101OBX000attgti0000.FIT	RGS1 spacecraft attitude good time intervals
P0136540101R2S002BGSPEC1003.FIT	RGS2 1st order background spectrum
P0136540101R2S002BGSPEC2003.FIT	RGS2 2nd order background spectrum
P0136540101R2S002EVENLI0000.FIT	RGS2 screened event list
P0136540101R2S002EXPMAP0000.FIT	RGS2 exposure map
P0136540101R2S002RSPMAT1003.FIT	RGS2 1st order response matrix
P0136540101R2S002RSPMAT2003.FIT	RGS2 2nd order response matrix
P0136540101R2S002merged0000.FIT	RGS2 raw merged event list
P0136540101R2S002SRCLI_0000.FIT	RGS2 source list
P0136540101R2S002SRSPEC1003.FIT	RGS2 1st order source+background spectrum
P0136540101R2S002SRSPEC2003.FIT	RGS2 2nd order source+background spectrum
P0136540101R2X000hkgti_0000.FIT	RGS2 housekeeping good-time intervals
P0136540101OBX000attgti0000.FIT	RGS2 spacecraft attitude good time intervals
P0136540101OBX000ATTTSR0000.FIT	XMM spacecraft attitude data
P0136540101OBX000fluxed1000.FIT	Combined RGS1 & RGS2 1st order source flux spectrum
P0136540101OBX000fluxed2000.FIT	Combined RGS1 & RGS2 2nd order source flux spectrum

Table 8: Files produced by (rgsproc) for an observation of Mkn421 with default parameters except for the source name and coordinates.

5.6.1 Events, sources and selection criteria

In the default spectroscopy mode, the most important parameters for each detected photon in the event lists are the two spatial coordinates and one energy coordinate. These go through a series of improvements in the initial stages of `rgsproc` in which the initial raw set of

```
[CCDNR, CCDNODE, RAWX, RAWY, ENERGY]
```

is transformed using the detailed geometry of the grating and detector assembles into coordinate angles along the grating dispersion direction, β or `BETA`, and the perpendicular cross-dispersion direction, χ or `XDSP`,

```
[BETA, XDSP, PHA]
```

which, in turn, are combined with knowledge of the history of the spacecraft's pointing and CCD gain and CTI characteristics to yield

```
[BETA_CORR, XDSP_CORR, PI]
```

in units of radians, radians and eV respectively. The detectors are almost invariably read out using a method which combines a 3×3 area of original CCD pixels into a single value, so-called 3×3 on-chip binning or OCB. `BETA_CORR` and `XDSP_CORR` reconstruct the angular distribution of photons emerging from the gratings and include randomisation of the quantised CCD coordinates in order to simulate a continuous distribution. The coordinates of the nominated source make a subtle appearence at this stage as they are used in the correction of `BETA` to `BETA_CORR` caused by variations in the angle of incidence on the gratings caused by spacecraft pointing jitter. The otherwise redundant `PI` energy column is used to distinguish between spatially overlapping orders.

Once the event list has been filtered for bad pixels of the various types signalled by `rejflags` in Table 7, the source list plays a central role. The source list here has 5 binary-table extensions

P0136540101R2S002SRCLI_0000.FIT:SRCLIST	1=PROPOSAL 2=ONAXIS 3=MKN421
P0136540101R2S002SRCLI_0000.FIT:RGS2_BACKGROUND	BETA_CORR-XDSP_CORR region
P0136540101R2S002SRCLI_0000.FIT:RGS2_SRC3_SPATIAL	BETA_CORR-XDSP_CORR region
P0136540101R2S002SRCLI_0000.FIT:RGS2_SRC3_ORDER_1	BETA_CORR-PI region
P0136540101R2S002SRCLI_0000.FIT:RGS2_SRC3_ORDER_2	BETA_CORR-PI region

which specify selection regions for the nominated source #3 and the orders 1 and 2 chosen by default and the background region. The calculation of the position and width of the selection regions depends on the source's celestial coordinates, which the observer has taken so much trouble to get right, and the fractions of the `XDSP_CORR` and `PI` response curves specified by the parameters `xpsfbelow`, `xpsfabove`, `xpsfexcl` and `pdistincl`. The events which contribute to source and background spectra are those that fall in selection regions defined, for example, for 1st order in RGS2 :

((BETA_CORR,PI) IN REGION(P0136540101R2S002SRCLI_0000.FIT:RGS2_SRC3_ORDER_1))
&& ((BETA_CORR,XDSP_CORR) IN
REGION(P0136540101R2S002SRCLI_0000.FIT:RGS2_SRC3_SPATIAL))

Finally, the simulated continuous BETA_CORR distributions of selected events are rebinned into channels according to the betabinning instructions to yield the final binned counts spectra.

5.6.2 Pixel-by-pixel offset subtraction

One of the standard procedures applied during rgsproc is a so-called offset correction which is designed to deal with an additive noise property of CCDs. Usually, an identical value is used for every pixel in a CCD node. Since SAS v6, an alternative has been offered that takes advantage of offset values that have been determined for each of the individual 171 × 128 pixels in a node from routine instrumental monitoring data. These offset data, which form part of the ODF, are accumulated every 3 revolutions thus also providing each pixel with its offset history. While experience of this alternative approach is being built up, it is not offered as the automatic choice but users are encouraged to try it and let us know if they are able to endorse the advantages described in early trials (see Technical report XMM-SOC-CAL-TN-0046 [12]).

The alternative offset method is selected with the withdiagoffset switch as in the following example :

```
rgsproc withsrc=yes srclabel=Mkn421 \
                    srcstyle=radec srcra=166.113808 srcdec=+38.208833 \
                    withdiagoffset=yes
```

5.6.3 The new class of cool pixels

Hot pixels and columns are a routine feature of CCD detectors and rgsproc takes action to exclude them from the spectra it generates. The accumulation of nearly 1 million seconds data on the bright blazar Mkn421 over the course of the mission has allowed a new class of cool pixels to be identified. These are single columns that give signals a few percent below the values expected from their immediately neighbours and are only likely to be relevant when studying weak absorption features in spectra with high statistics. By default, rgsproc does not discard these data but they can be excluded using the keepcool switch as in the following example :

```
rgsproc withsrc=yes srclabel=Mkn421 \
                    srcstyle=radec srcra=166.113808 srcdec=+38.208833 \
                    keepcool=no
```

5.7 The use of xmmselect on RGS data

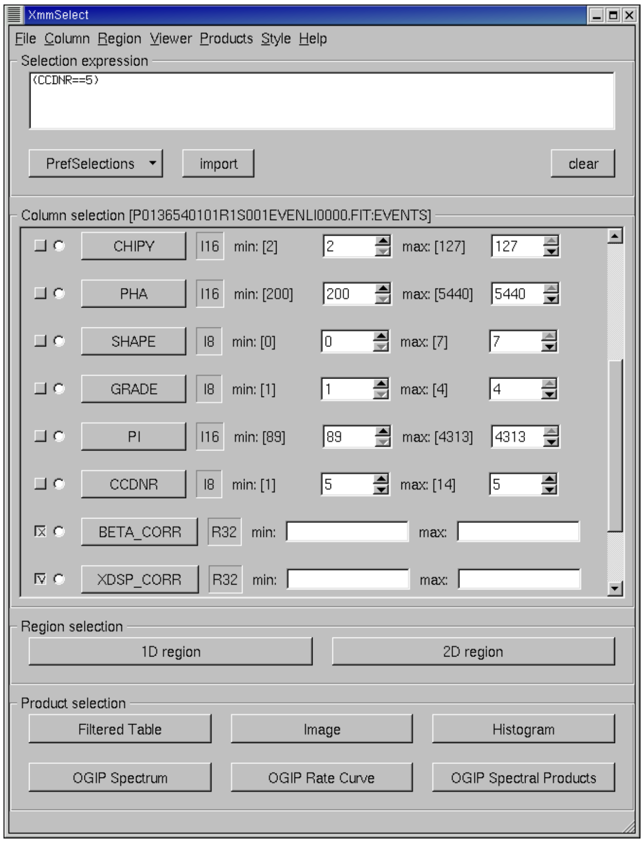
Although the spectra and response matrices produced by rgsproc are usually trustworthy, it is worthwhile getting to know the events files on which they are based as well as making routine checks on the data. As its name suggests, xmmselect

http://xmm.esac.esa.int/sas/current/doc/xmmselect/index.html

is a tool for the manipulation and display of XMM data in general including RGS event files via a GUI and is often invoked as a background process

```
 xmmselect table=P0136540101R1S001EVENLI0000.FIT:EVENTS &
```

to provide a graphical interface to evselect, its command-line equivalent. Fig. 29



Selection expression

(CCDNR==5)

Column selection [P0136540101R1S001EVENLI0000.FIT:EVENTS]

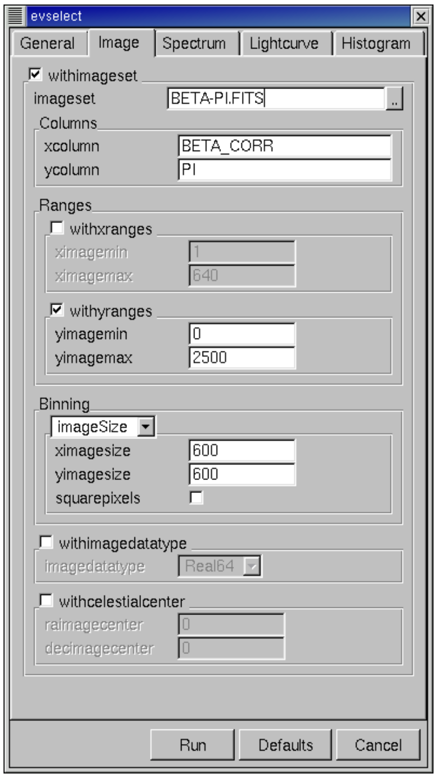
□ ○	CHIPY	I16	min: [2]	2	max: [127]	127
□ ○	PHA	I16	min: [200]	200	max: [5440]	5440
□ ○	SHAPE	I8	min: [0]	0	max: [7]	7
□ ○	GRADE	I8	min: [1]	1	max: [4]	4
□ ○	PI	I16	min: [89]	89	max: [4313]	4313
□ ○	CCDNR	I8	min: [1]	5	max: [14]	5
☒ ○	BETA_CORR	R32	min:		max:	
☒ ○	XDSP_CORR	R32	min:		max:	

Region selection

Product selection

Figure 29: A common use of xmmselect is to inspect details of RGS data. In this example, the user has pressed the **CCDNR** button after adjusting the adjacent minimum and maximum values to select CCD 5 only. The selection criterion is then shown in the top window. Columns BETA_CORR and XDSP_CORR have been selected using the checkboxes on the left in preparation for pressing the **Image** button to produce a pointing-corrected spatial plot of the data of the single CCD selected.

illustrates the main panel of xmmselect in use on an RGS1 event list. It is possible to specify both simple selection criteria, such as the one shown there, or the more complex RGS spectrum regions discussed above in section 5.6.1 before making an image, for example. For every operation, xmmselect displays the results and writes them to an appropriate standard-format FITS file. The displays are done by launching automatically one of several flexible third-party graphical tools, such as ds9 [16], that allow a great deal of flexibility in tailoring plots to individual requirements. It is always possible to choose the name of the file in which the results are recorded to overrule the default name that otherwise will appear, perhaps unexpectedly, in the working directory.



☑ withimageset
imageset BETA-PI.FITS
Columns
xcolumn BETA_CORR
ycolumn PI
Ranges
☐ withxranges
ximagemin 1
ximagemax 640
☑ withyranges
yimagemin 0
yimagemax 2500
Binning
imageSize
ximagesize 600
yimagesize 600
squarepixels ☐
☐ withimagedatatype
imagedatatype Real64
☐ withcelestialcenter
raimagecenter 0
decimagecenter 0

Figure 30: The xmmselect Image panel is shown here for preparing a BETA_CORR/PI banana plot. Values have been specified for imageset, the image output file, and for a y-axis range suitable for the range of photon PI values. Pushing the **Run** button gets the work done.

As a matter of routine, it is advisable to look at BETA_CORR/XDSP_CORR and BETA_CORR/PI plots of the screened event list and source and background lightcurves to be able to make a critical assessment of the

- overall strength of the spectral orders
- quality of the selection regions
- success of hot pixel detection
- source variability
- background strength and variability
- possible presence of other sources
- source spatial extent

5.7.1 Generating RGS images

Once the axes have been selected and any selection criteria specified, pushing the **Image** button initiates dialogue panels that includes an *Image* section like that shown in Fig. 30 which gives the opportunity to review the choices made. The `BETA_CORR/PI` images, commonly known as banana plots, are improved by overruling the default y-range by checking the `withyranges` box and setting

`yimagemin`	0
`yimagemax`	2500

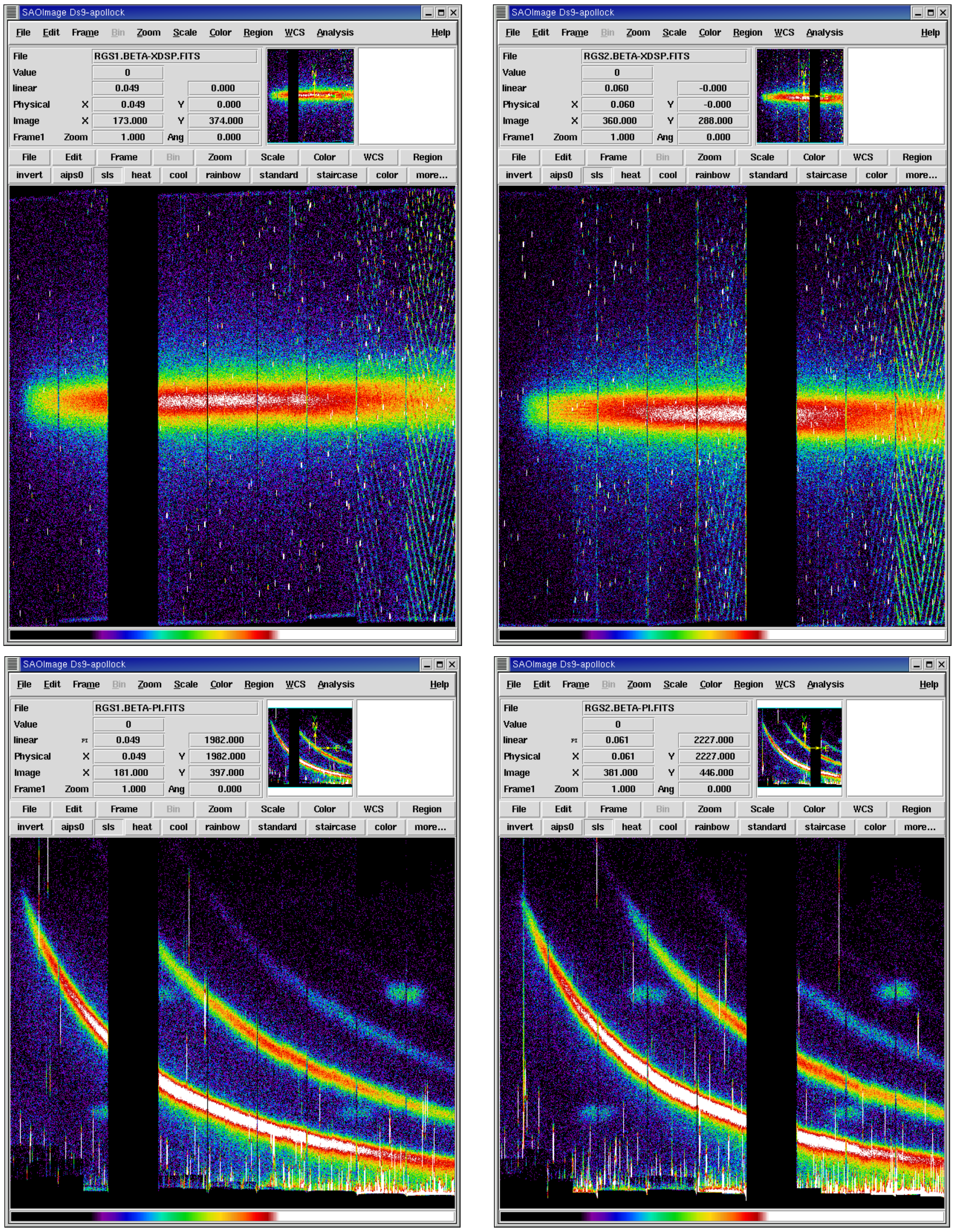
Fig. 31 shows plots of the merged RGS1 events. These plots are orthogonal projections of the all-important `BETA_CORR,XDSP_CORR,PI` 3-D RGS data space of the merged list of all the events detected in the observation, thus showing all the features, both good and not so good, of RGS data. The edges are clear of the 9 CCDs, numbered 1-9 from right to left. In both plots, wavelength and dispersion angle increase from left to right.

From quite early in the mission, each RGS has been missing data from one CCD because of electronics failures, RGS1 CCD7 and RGS2 CCD4, that happily do not cover the same wavelengths. The pointing coordinates were evidently chosen well enough to put this bright source central in the aperture. The source was bright enough to be seen up to 4th or even a weak 5th order in the characteristic hyperbolic-shaped areas occupied by photons that have passed through the gratings. There are plenty of hot pixels and columns and the so-called fixed pattern noise shows as the herring-bone pattern in CCDs 8 and 9. Calibration sources of F Kα at `PI`= 677eV span CCDs 2&3 and 7&8 ; and Al Kα at `PI`= 1487eV span CCDs 3&4 and 8&9.

The locations of the selection regions may be checked using the task `rgsimplot`

http://xmm.esac.esa.int/sas/current/doc/rgsimplot/index.html

which plots them over `BETA-XDSP` and `BETA-PI` images that are, in this case, most usefully generated from the screened events files. Fig. 32 shows such an example.



SAOImage Ds9-apollock
File Edit Frame Bin Zoom Scale Color Region WCS Analysis Help
File Edit Frame Bin Zoom Scale Color WCS Region
invert aips0 sls heat cool rainbow standard staircase color more...
N
E

File		RGS1.BETA-XDSP.FITS		
Value		0		
linear		0.049		0.000
Physical	X	0.049	Y	0.000
Image	X	173.000	Y	374.000
Frame1	Zoom	1.000	Ang	0.000

File		RGS2.BETA-XDSP.FITS		
Value		0		
linear		0.060		-0.000
Physical	X	0.060	Y	-0.000
Image	X	360.000	Y	288.000
Frame1	Zoom	1.000	Ang	0.000

File		RGS1.BETA-PI.FITS		
Value		0		
linear	PI	0.049		1982.000
Physical	X	0.049	Y	1982.000
Image	X	181.000	Y	397.000
Frame1	Zoom	1.000	Ang	0.000

File		RGS2.BETA-PI.FITS		
Value		0		
linear	PI	0.061		2227.000
Physical	X	0.061	Y	2227.000
Image	X	381.000	Y	446.000
Frame1	Zoom	1.000	Ang	0.000

Figure 31: The `BETA_CORR/XDSP_CORR` and `BETA_CORR/PI` plots produced by `xmmselect` using `ds9` for the merged RGS1 and RGS2 event lists of Mkn421 before filtering, thus showing all hot pixels and columns and other blemishes in addition to the bright smooth continuum of this active galaxy. It can take some experience with ds9 **Zoom**, **Color** and **Scale** controls to display the data to best effect. These images used **sls** colours on a logarithmic scale.

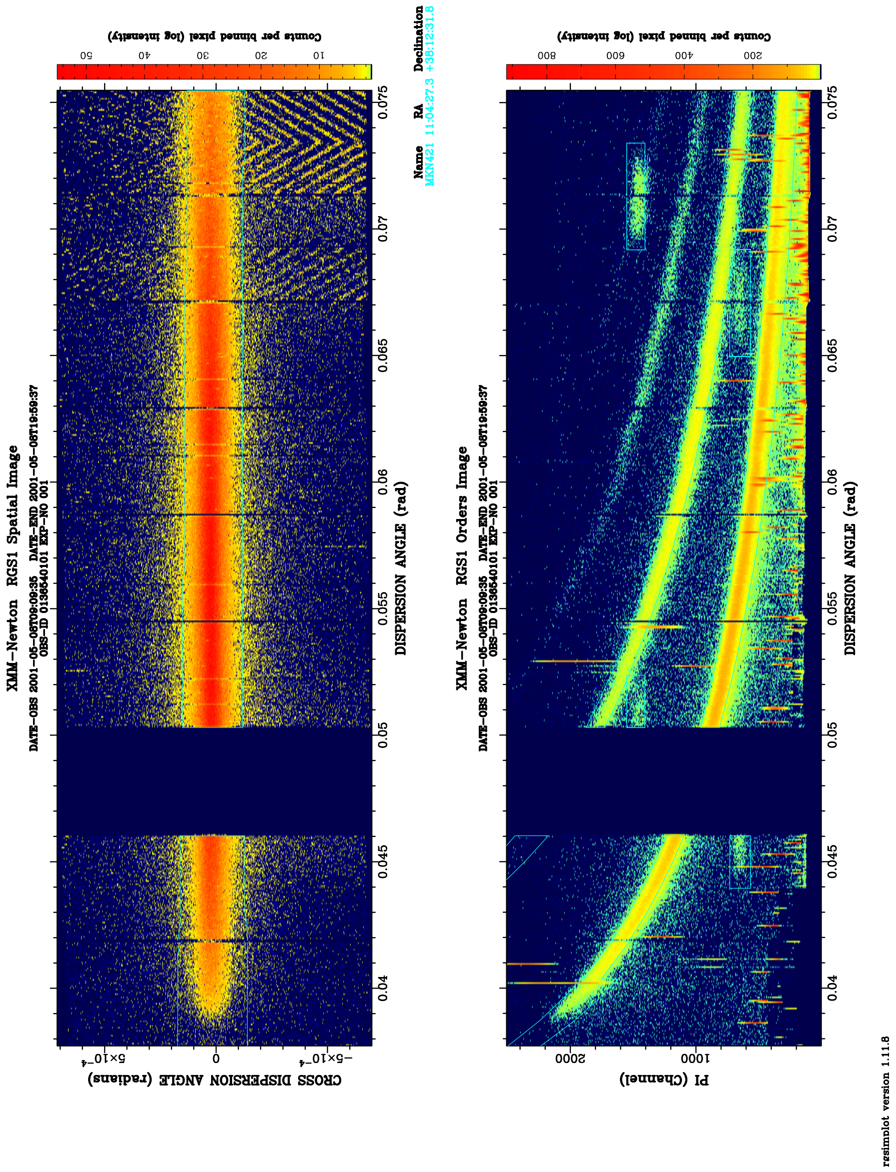


Figure 32: The BETA_CORR-XDSP_CORR and BETA_CORR-PI images produced by xmmselect from the screened event list can be shown with the selection regions using rgsimplot.

5.7.2 Generating RGS lightcurves

By default, `rgsproc` calculates a single spectrum per RGS per order per observation even though variability is a common property of many X-ray sources. A prototype dedicated `rgslccorr` task has been released in SAS v7.1 to allow users to create RGS light curves from the same spectral and background selection regions that are used to generate spectra. It uses event and source-list files to calculate background and dead-time corrections to produce light-curves defined by parameters such as time bin size; start and stop time; and and spectral order. RGS1 and RGS2 data may also be combined into a single light curve. The XMM-SOC would be happy ro receive any comments or suggestions about this task.

Otherwise, the RGS variability of both source and background may be checked by generating lightcurves using `xmmselect`'s [**OGIP Rate Curve**] button with suitable source and background selection criteria. Source counts can be selected with the same criteria stored in the source list used to generate spectra.

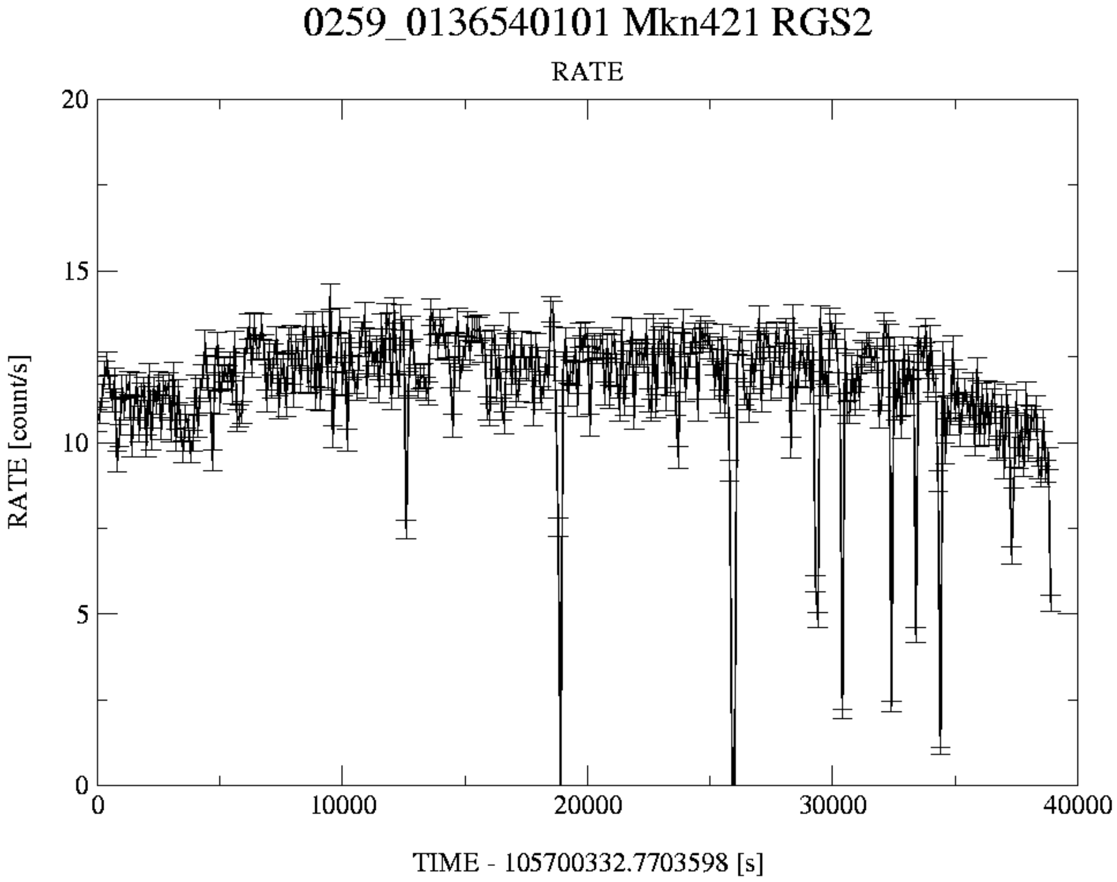


Figure 33: RGS2 light curve of Mkn421 generated with `xmmselect` showing variability of about 20%. The dozen or more spurious low points in this curve are due to poor treatment of gaps in the data.

Fig. 33, for example, shows the variable count rate of Mkn421 of events within the 3-D 1st order source spectrum selection region as follows:

((BETA_CORR,PI) IN REGION(P0136540101R2S002SRCLI_0000.FIT:RGS2_SRC3_ORDER_1)) && ((BETA_CORR,XDSP_CORR) IN REGION(P0136540101R2S002SRCLI_0000.FIT:RGS2_SRC3_SPATIAL))

5.7.3 Assessing the RGS background

Perhaps the principal use of an RGS lightcurve is to make an assessment of the background that often accounts for the majority of detected events over the whole instrument and which must therefore be explicitly quantified. The background is made up of several contributions and is usually weak enough to cause few problems: RGS spectra are photon limited more than 80% of the time. However, solar flares and other particle events can cause significant or in rare cases even overwhelming contamination. Rapid variability is also common so that only part of an observation may be affected so that it is possible to generate supplementary GTIs in order to exclude periods of unacceptably high background. Fig. 34 shows the background lightcurve generated using:

(CCDNR = 9) && ((BETA_CORR,XDSP_CORR) IN REGION(P0136540101R2S002SRCLI_0000.FIT:RGS2_BACKGROUND))

What constitutes unacceptably high background is often a matter of personal judgement depending on the type of analysis undertaken. High-contrast features like strong emission lines can often tolerate higher background levels than smooth continuum spectra and the overall source brightness is clearly a consideration. It is best to experiment. As a rule of thumb, data with CCD9 background rates above 1 count/s might be considered suspect and be excluded by using the GTI mechanism to flag periods of low background.

```
tabgtigen table=RGS2.background.FITS gtiset=RGS2.background.GTI.FITS \
          expression='(RATE<1)'
```

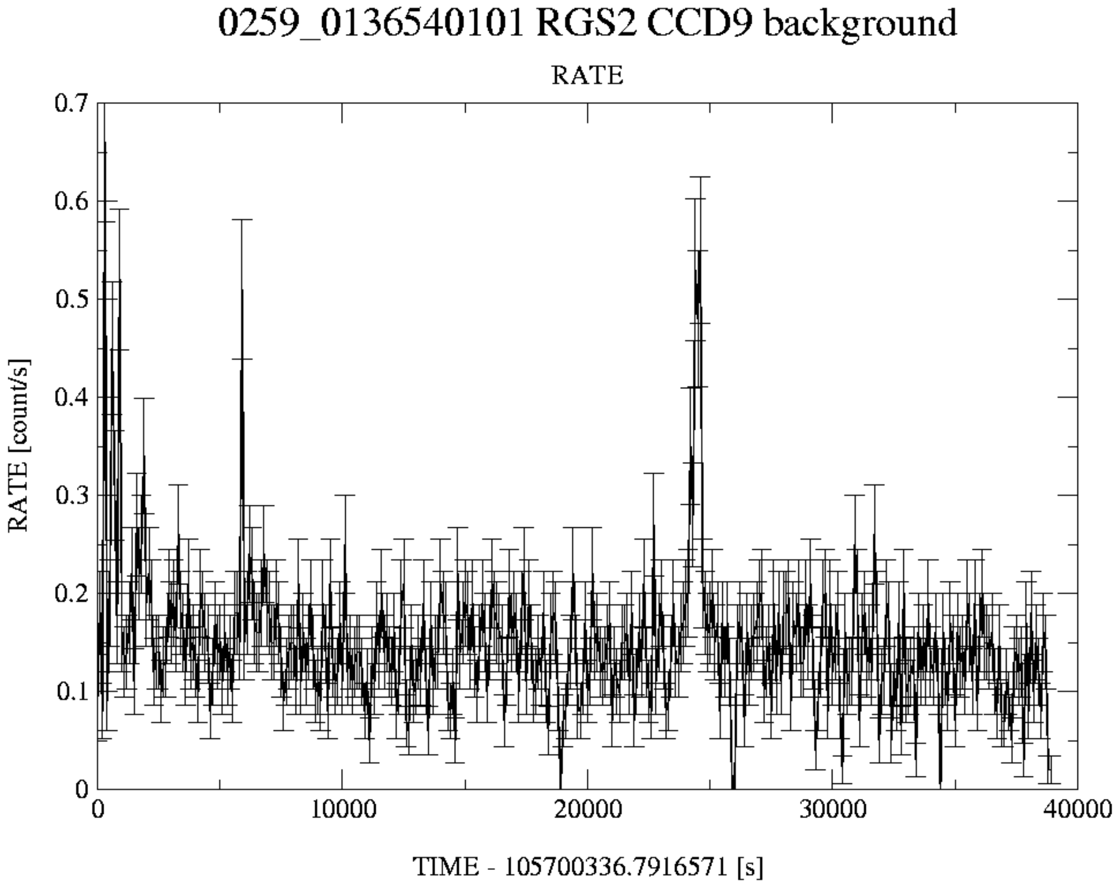


Figure 34: RGS2 light curve of the background near Mkn421 generated with `xmmselect`. Although there are a few small flares, none are bad enough to consider removing from the source data.

5.8 Rerunning the RGS processor rgsproc

As many of the quality checks described above can only be done after an initial run of `rgsproc`, a decision will be often be made to rerun the analysis with some adjustments. It might not be necessary to rerun the whole of `rgsproc`, which is divided into five different stages that are alternative entry and exit points. Fig. 35 lays out how individual tasks are grouped within the five stages. For example, the background `GTI` file made above would properly be incorporated at the filter stage as follows

```
rgsproc auxgtitables=RGS2.background.GTI.FITS \
         entrystage=3:filter finalstage=5:fluxing
```

A similar approach could be used to generate spectra as a function of intensity using the source light curve of Fig. 33. Other examples involving further runs of `rgsproc` for multiple or extended sources are discussed below.

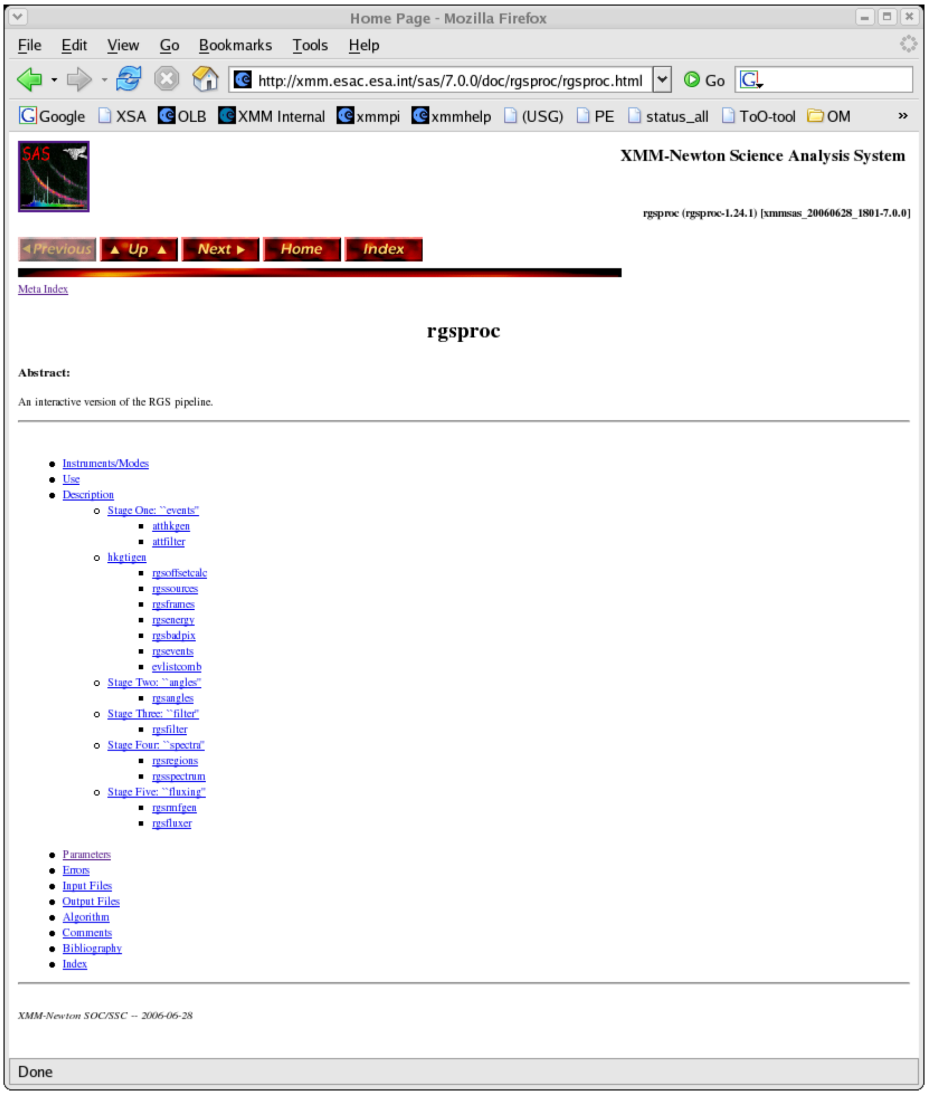
5.9 Treatment of multiple RGS sources

In the rare cases when two or more sources appear in the RGS aperture, `rgsproc` should be instructed to calculate the required number of spectra and modify its treatment of the background. In all such cases encountered so far, such as the observation of the active stars YY Gem and Castor shown in Fig. 36, the spectra have been well enough separated so that their selection regions can be treated independently. The current tools do not allow simultaneous estimates of overlapping spectra. It is not possible initially to specify more than one source, so a step-by-step approach is required adding sources one at a time with `rgssources` and setting source and background parameters accordingly before rerunning `rgsproc` to recalculate the spectra. Armed with accurate SIMBAD coordinates for the brighter YY Gem and weaker Castor (not forgetting Castor's high proper motion), the sequence of events might be as follows to make spectra for both stars and exclude both from the background

```
rgsproc withsrc=yes srclabel=YYGem \
                     srcstyle=radec srcra=113.655858 srcdec=+31.869386

rgssources srclist=P0123710201R1S004SRCLI_0000.FIT addusersource=yes \
                      label=Castor ra=113.649428 dec=+31.888276

rgssources srclist=P0123710201R2S005SRCLI_0000.FIT addusersource=yes \
                      label=Castor ra=113.649428 dec=+31.888276

rgsproc entrystage=4:spectra procsrcexpr=’INDEX==3&&INDEX==4’ \
                            exclsrcexpr=’INDEX==3&&INDEX==4’
```

5.10 Treatment of extended RGS sources

Even though the methods currently used by `rgsproc` were designed for point sources, the 5 arcmin cross-dispersion width of the RGS apertures also makes the instrument suitable for


Home Page - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://xmm.esac.esa.int/sas/7.0.0/doc/rgsproc/rgsproc.html
Go
Google XSA OLB XMM Internal xmmpi xmmhelp (USG) PE status_all ToO-tool OM
Previous
Up
Next
Home
Index
Done

XMM-Newton Science Analysis System

rgsproc (rgsproc-1.24.1) [xmmsas_20060628_1801-7.0.0]

Meta Index

rgsproc

Abstract:

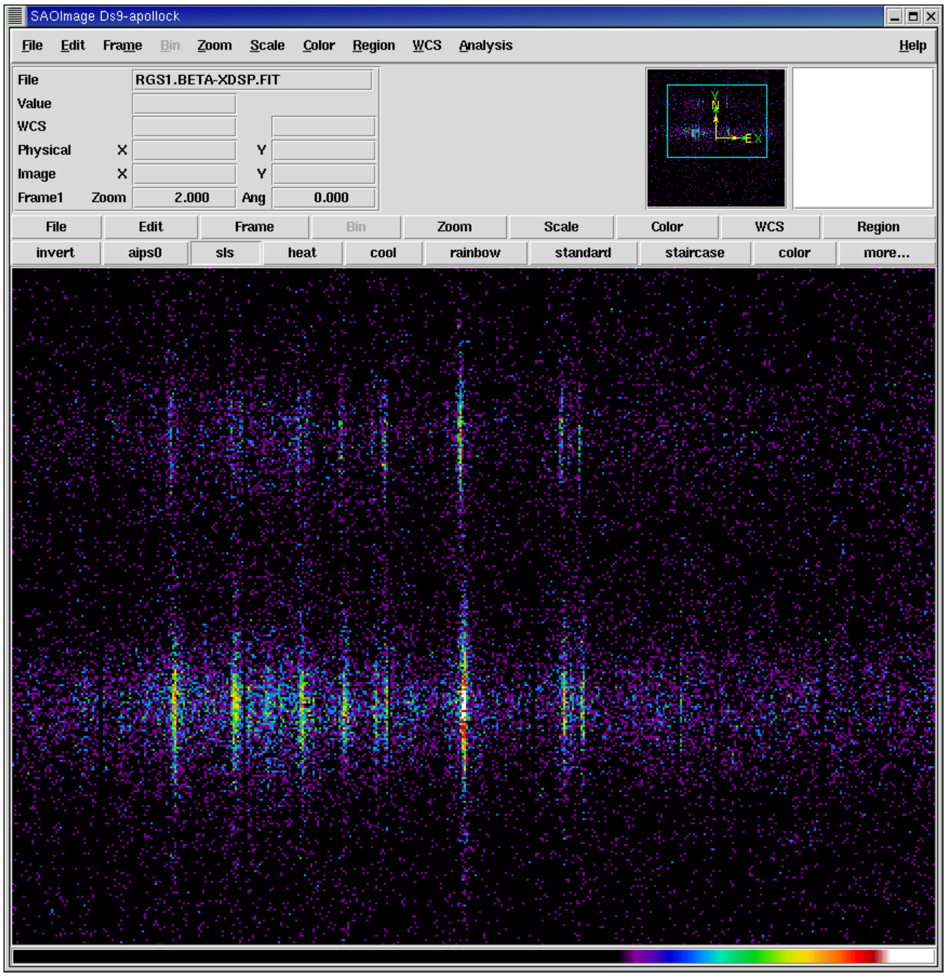
An interactive version of the RGS pipeline.

- Instruments/Modes
- Use
- Description
  - Stage One: ``events''
    - atthkgen
    - attfilter
  - hkgtigen
    - rgsoffsetcalc
    - rgssources
    - rgsframes
    - rgsenergy
    - rgsbadpix
    - rgsevents
    - evlistcomb
  - Stage Two: ``angles''
    - rgsangles
  - Stage Three: ``filter''
    - rgsfilter
  - Stage Four: ``spectra''
    - rgsregions
    - rgsspectrum
  - Stage Five: ``fluxing''
    - rgsrmfgen
    - rgsfluxer
- Parameters
- Errors
- Input Files
- Output Files
- Algorithm
- Comments
- Bibliography
- Index

XMM-Newton SOC/SSC -- 2006-06-28

Figure 35: The `rgsproc` home page in the SAS on-line help system, showing the five stages which form the possible start and end points of RGS data analysis.

observing extended sources such as supernova remnants or clusters of galaxies. Obviously, such objects can fill part or all of the aperture and invalidate the usual means of differentiating source and background selection regions in terms of the parameters `xpsfincl` & `xpsfexcl` that are defined in terms or point-source fractions. For moderately extended sources which do not fill the aperture, such as the LMC SNR N132D shown in Fig. 37, it is possible to adjust the equivalent point-source fractions by a process of trial and error to nominate suitable selection regions. For more extensive sources, when this is not possible, there are RGS background template files and techniques available through the XMM web pages that allow independent estimates of background spectra. Source extension in the dispersion direction causes confusion with the wavelength scale. XSPEC v11.2 [17] contains a new `rgsxsrc` model which specifically deals with spectra of extended sources.



File RGS1.BETA-XDSP.FIT

Frame1 Zoom 2.000 Ang 0.000

Figure 36: An example of two sources in the RGS aperture. In this observation early in the mission of YY Gem and Castor, it was already known from previous work that the two stars are bright, so the aperture was aligned perpendicular to the line joining the stars to allow maximum separation of the two spectra.

5.11 Computing a model spectrum of the RGS background

The task `rgsbkgmodel` can be used to compute a model spectrum of the RGS background applicable to a given observation from a combination of observations of empty fields.

For a detailed description of this task see the technical note CAL-TN-0058-1-1 at:

http://xmm.esac.esa.int/external/xmm_sw_cal/calib/documentation.shtml

5.12 Creating RGS response matrices

As emphasised above, the source coordinates are most crucially under the observer's control and have a profound influence on the accuracy of the wavelength scale as recorded in the RGS response matrix, or RMF, that is produced automatically by `rgsproc` using the task `rgsrmfgen`

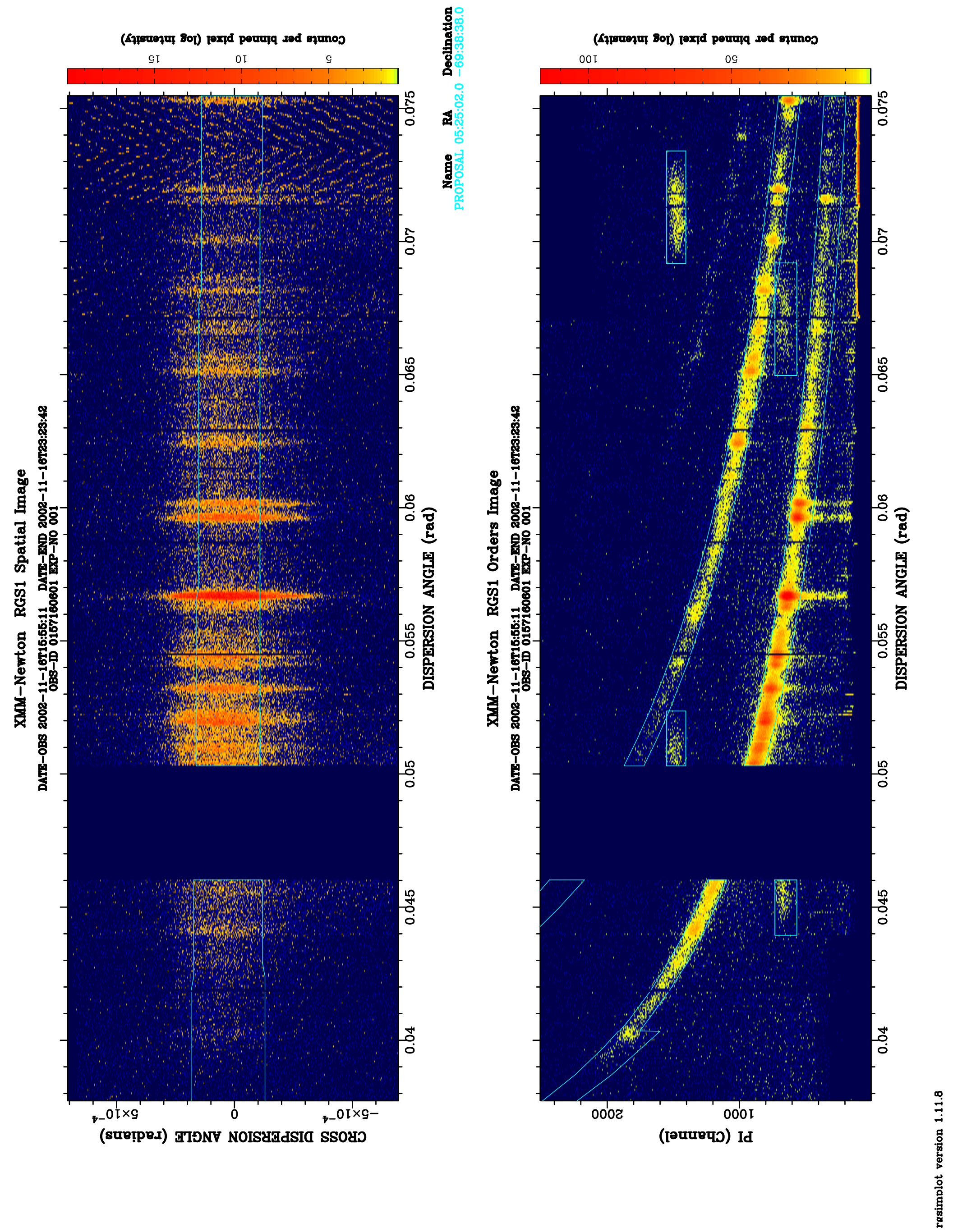


Figure 37: The bright emission lines of the SNR N132D, which is about 2 arcmin in diameter, fit comfortably within the RGS aperture but overflow the default selection regions defined for point sources. By extending the cross-dispersion selection region to 98%, in this case, most of the source flux is recovered.

although not for pipeline PPS products. There are different RMFs for RGS1 and RGS2 and for each order. Many parts of the instrument model held in the CCF are also used to describe the geometry and radiation transfer properties of each component of the instrument from telescope, through grating assembly to detector and these are not normally subject to change. An RMF is used to connect an input wavelength or energy grid representing cosmic physical units to the output instrumental BETA_CORR channels in which the observed selected data have been cast. While the spacing of the output grid is fixed by a requirement to sample sufficiently the instrumental resolution and must agree with the relevant accumulated spectrum, the input grid is another quantity under the user's express control via the rgsrmfgen parameters emin, emax and ebins. Although wavelength units are natural for grating spectrometers, the OGIP RMF standard requires energy units, given here in keV. Here is an example of an independent run of rgsrmfgen from the command line to generate the 2nd order RGS2 matrix for Mkn421 :

```
rgsrmfgen spectrumset=P0136540101R2S002SRSPEC2003.FIT \
          evlist=P0136540101R2S002EVENLI0000.FIT \
          emin=0.4 emax=2.5 rows=5000 rmfset=RGS2.o2.rmf
```

Despite the elongated appearance of spectral lines in plots like Fig. 36, the point-source monochromatic response is in fact reasonably symmetric as shown in Fig. 38. Although a typical line covers several pixels, it is possible to locate the line centroid to a fraction of a pixel. In this case, it is necessary to generate a response matrix covering the line and some surrounding continuum with a much more dense grid than usual in order to take full advantage of all the information available in the data and calculate reliable errors.

5.13 Combined analysis of spectra

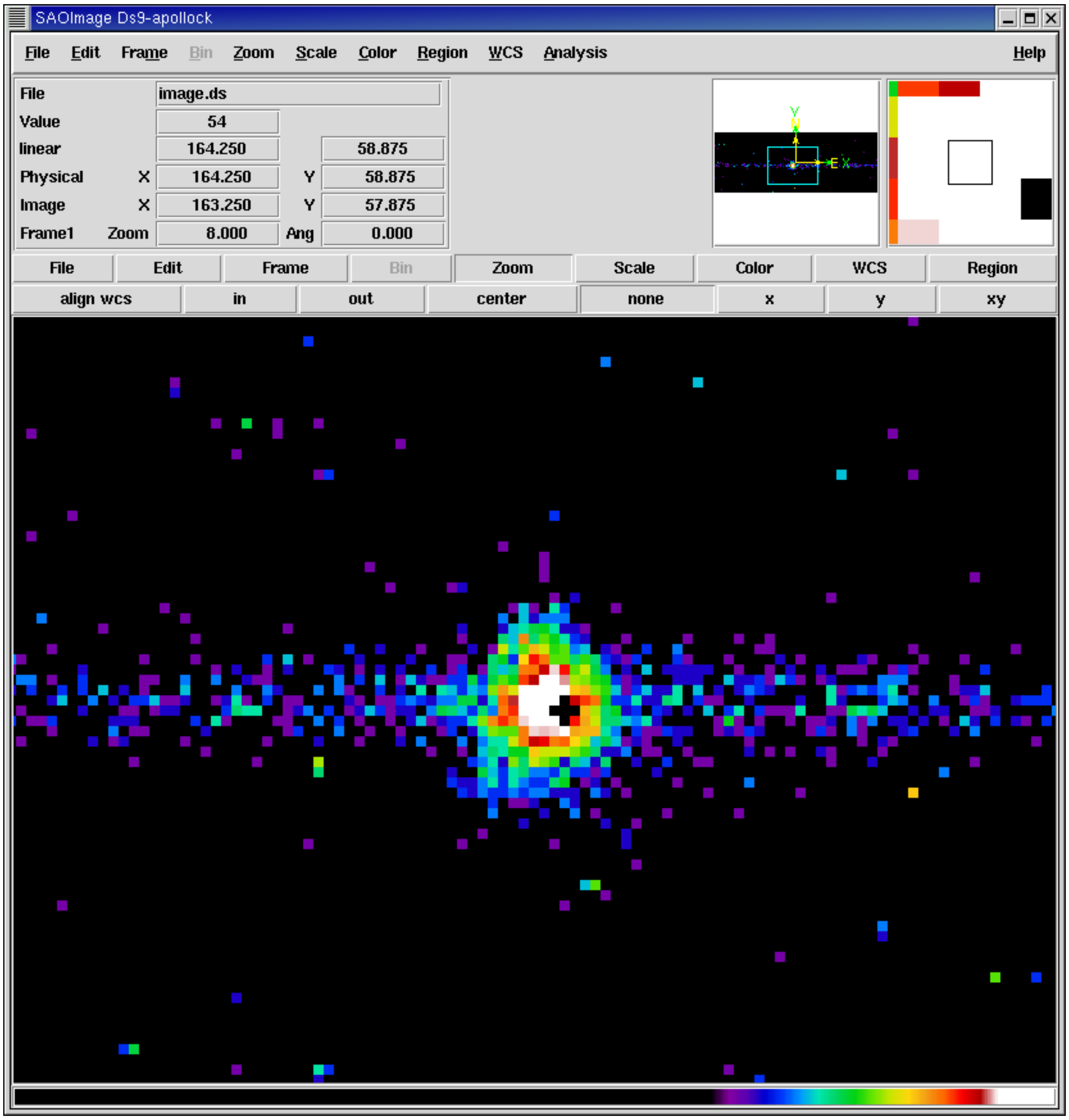
The task rgscombine can be used to combine two or more spectra that have been produced by rgsproc for the same RGS and order into a single spectrum and to calculate a combined response matrix. In order for the task to run, the component spectra must have been aligned, so there are implications for the way in which rgsproc is run initially. For details, see the technical note XMM-SOC-SW-TN-0021 at:

http://xmm.esac.esa.int/sas/current/documentation/technicalnotes/

5.14 Use of rgsfluxer

Although a rigorous analysis of RGS spectra for calculating physical model parameters requires the simultaneous use of individual spectra and their associated response matrices, it is often useful to produce a figure to show the overall properties of an X-ray spectrum observed with the RGS. The task rgsfluxer can be used to produce a single illustrative fluxed spectrum from any number of component spectra and response matrices from any combination of RGS1 and RGS2 and 1st and 2nd order. In contrast with rgscombine described in 5.13, the spectra do not need to be aligned. If the spectra are not background corrected, which is the default for rgsproc, don't forget to tell rgsfluxer about the background spectra. For example, if several total and background spectra of Capella are collected in a directory, the following commands will produce the rough picture of the mean stellar spectrum shown in

```
setenv SPECTRA *SRSPEC*
setenv BKGs    *BGSPEC*
setenv RMFs    *RSPMAT*
rgsfluxer pha=''$SPECTRA'' bkg=''$BKGs'' rmf=''RMFs'' file=Capella.RGS.spectrum
```



File image.ds
Value 54
linear 164.250 58.875
Physical X 164.250 Y 58.875
Image X 163.250 Y 57.875
Frame1 Zoom 8.000 Ang 0.000

Figure 38: A bright OVIII emission line in raw detector coordinates showing the reasonably circular symmetry of the RGS's monochromatic response. A hot pixel has been detected near the core causing it to be masked out with its immediate neighbours when the default rejection flags are applied.

The output of `rgsfluxer` is a tabulation of physical flux against wavelength and it can be useful for this to serve as an model-free input spectrum for comparison with lower-resolution spectra from the EPIC instruments, for example, in particular. The task `rgsfluxmodel` recasts the output of `rgsfluxer` into a file format suitable for use as an XSPEC table model that can be read into XSPEC using the `atable` command.

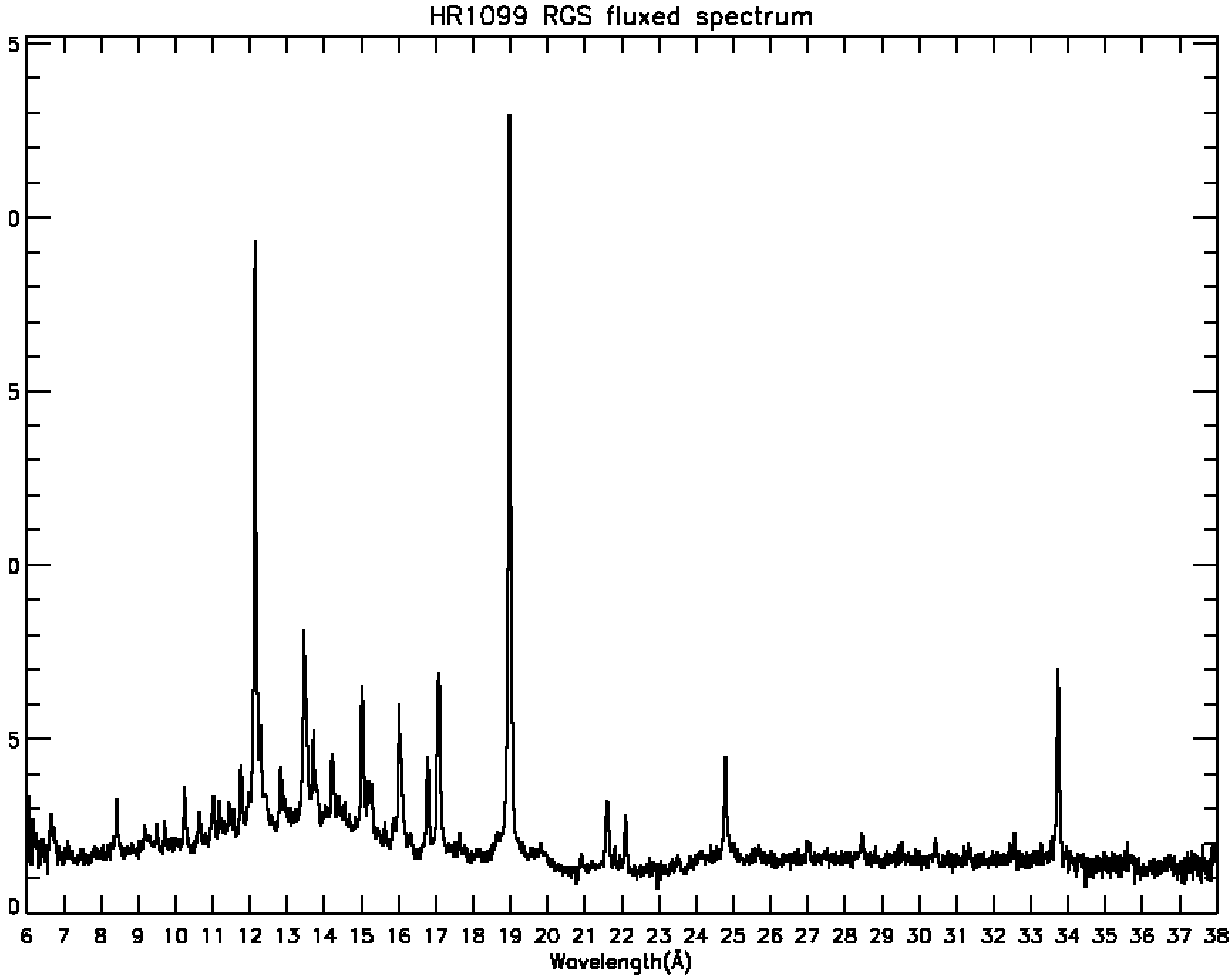


Figure 39: The fluxed spectrum of HR1099 produced with `rgsfluxer` made from a combination of 86 1st and 2nd order RGS1 and RGS2 spectra.

5.15 The RGS pipeline products

The PPS products generated by rgdprods, an automatic version of `rgsproc`

http://xmm.esac.esa.int/sas/current/doc/rgsprods/index.html

have some differences compared to the files produced interactively although, since SAS v6, 1st-order response matrices are also included. The number of RGS pipeline products is quite large but may be handled with ease by pointing a browser to the hypertext index files, starting with **INDEX.HTM**, included in the distribution. RGS pipeline product file names are of the form :

- PiiiiiijjkkaablllCCCCCCmnnn.zzz, where

  iiiiiijjkk observation number

  aa detector, R1 - RGS1, R2 - RGS2, RG - both

  b S for scheduled observation, U for unscheduled, X for general purpose

  lll exposure number or 000 if none

  CCCCCC file identification shown in Table 9

  m order number

  nnn source number

  zzz file type

  FTZ gzipped FITS format for use, for example, with `xmmselect`, `ds9` or `fv`
  HTM HTML file for use web browser
  PDF Portable Document Format file
  PNG Portable Networks Graphics file
  TAR TAR file

File ID	Contents	File Type
PPSSUM	PPS summary	HTML
OBX000SUMMAR	Observation Data File Summary	HTML
EPX000SUMMAR	EPIC Processing Summary	HTML
OMX000SUMMAR	OM Processing Summary	HTML
RGX000SUMMAR	RGS Processing Summary	HTML
RGX000FLUXED	Combined RGS1 & RGS2 1st order fluxed spectrum	FTZ & PDF
BGMODL1	synthetic 1st order background spectrum	FTZ
BGMODL2	synthetic 2nd order background spectrum	FTZ
BGSPEC1	1st order background spectrum	FTZ
BGSPEC2	2nd order background spectrum	FTZ
DSPHIS1	cross-dispersion histogram	FTZ
EVENLI	combined events list	FTZ
EXPMAP	exposure map	FTZ
FBKTSR	background lightcurve	FTZ
IMAGE_	χ v β spatial image	FTZ & PNG
ORDIMG	PI v β banana plot	FTZ & PNG
RSPMAT1	1st order response matrix	FTZ
SBSPEC1	1st order source total spectrum	FTZ
SBSPEC2	2nd order source total spectrum	FTZ
SRCLI_	source parameters	FTZ
SRSPEC0	1st & 2nd order source net spectrum plot	PDF
SRSPEC1	1st order source net spectrum	FTZ
SRSPEC2	2nd order source net spectrum	FTZ
WFSPEC0	1st & 2nd order whole-field total spectrum plot	PDF
WFSPEC1	1st order whole-field total spectrum	FTZ
WFSPEC2	2nd order whole-field total spectrum	FTZ
WREMAT1	1st order whole-field response matrix	FTZ

Table 9: RGS pipeline products

6 Analysis of OM optical monitor data

6.1 The OM data

6.1.1 OM observing modes and data types

The OM can be operated in two basic science modes: image and fast mode (see the UHB for further details). These modes can be used separately or combined, in different instrument configurations:

- Default modes: Image_only and Image+Fast. This is a set of 5 succesive exposures with predefined windows configurations covering nearly 92% of the field with/without a Fast mode window centered at the boresight location.
- Science User Defined Image and/or Fast mode windows: the user can define in pixels or in (RA, Dec) up to 5 windows, from which up to 2 can be defined in Fast mode
- In addition, the OM detector can be operated in Full Frame mode to obtain images of the entire FOV, either at high resolution (0.5 arcsec/pix, 2048 × 2048 format) or at low resolution (1 arcsec/pix, 1024 × 1024 format).

In the windowed modes, the detector windows are re-centered to correct for pointing errors (Field Acquisition, FAQ). Also a shift_and_add mechanism is used to compensate for spacecraft tracking stability. Full frame modes do not have these capabilities.

The observing modes determine the type of data obtained with the OM: 2-D images for image modes and events lists if fast mode is used.

The default modes mentioned above will produce the following data types: since each exposure has associated two image windows, there will be two image files per exposure which correspond to a small high resolution window at the center of the FOV and a large low resolution one placed at different locations in each exposure. If the default mode was Image+Fast, there will be an additional event list file corresponding to the Fast window.

In the Science User Defined windows configuration there will be one image file or event list corresponding to each of the windows.

The Full Frame images consist of a single image file if high resolution was used and four adjacent image files in case of low resolution. (It should be noted that the four image files correspond to a unique exposure).

In addition to these image and/or event list files, for each exposure there will be several auxiliary files containing instrument configuration parameters. There will also be files containing OM house-keeping data for the whole observation. All these files together with the spacecraft attitude files, common for all instruments, constitute the ODF for OM data.

Table 10 contains a summary of the possible OM files contained in an ODF (for one exposure).

OM Images files have a file name ending with a “IMI.FIT”. There is one image 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 processed image windows of an OM default configuration before viewing them using the SAS task `ommosaic`.

File name	Description of the file content
0001-0000000000-OMS00100IMI.FIT	image window 1
0001-0000000000-OMS00101IMI.FIT	image window 2
0001-0000000000-OMS00102FAE.FIT	event list (if fast mode was used)
0001-0000000000-OMS00100THX.FIT	tracking history
0001-0000000000-OMS00100WDX.FIT	priority window data
0001-0000000000-OMS00100PFX.FIT	priority fast mode data
0001-0000000000-OMS00100RFX.FIT	priority reference frame data
0001-0000000000-OMS40000PAX.FIT	field acquisition data
0001-0000000000-OMX00000PEH.FIT	periodic house-keeping
0001-0000000000-OMX00000NPH.FIT	non-periodic house-keeping

Table 10: ODF files associated with a single OM exposure.

OM Fast mode event lists have a file name ending with “FAE.FIT”. There is one dataset for each fast window. It is possible to inspect the data files using e.g. the ftool viewer `fv`.

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. It is possible to inspect these files with e.g. the ftool viewer `fv`. A plot of column DX and DY shows the attitude stability during the exposure. The PPS product file whose name ends as “ TSHPLT.PDF” provides a visualization of the pointing stability during the exposure in PDF format.

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 to get a feeling on the absolute pointing accuracy, the user can look at this field acquisition dataset (PAX.FIT). The parameters 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. This file is used, e.g. to compute the exposure dead-time fraction.

OM priority fast mode data have a file name ending with a “PFX.FIT”. There is one dataset for each exposure containing fast mode window(s).

6.1.2 OM: filters and grisms

Before reaching the detector, photons collected by OM are intercepted by one of the elements of the Filter Wheel. These are six wide band filters, called in increasing wavelength order UVW2, UVM2, UVW1, U, B and V. Another filter, called White covers the full spectral band from UVW2 up to V (1800 Å to 6000 Å approximately). There is a Blocked (opaque) filter used in some engineering modes and also to protect OM when there is a too bright object in the field of view. All these filters can be used in both observing modes, image and/or fast mode, with all the configurations described in previous section (§ 6.1.1).

The filter wheel has in addition two grisms, Grism1 or UV grism and Grism2 or Visible grism, which can be used to obtain low resolution spectra in the range 1800 to 3600 Å and 3000 to 6000 Å respectively. Both grisms are used only in image mode. There is a user defined default configuration intended for single object spectroscopy, in which a rectangular window is defined so that it contains the spectrum of the target sitting at the boresight. Alternatively, the grisms can be used in full frame low resolution to register the spectra of all objects in the field of view. Data obtained with the grisms are therefore image data with the same characteristics as defined above.

It should be noted that in early phases of the project some observations with the OM grisms were done using the image default configuration described above (§ 6.1.1). These few cases cannot be dealt with SAS. Even if grism data can be considered as normal image data, only the observations obtained as single object spectroscopy or in full frame as explained before, can be processed by SAS.

6.1.3 Listing the OM Current Calibration Files

Table 11 provides a list of the OM calibration files. For a detailed description of these files and of their usage, the user is referred to the CCF interface document and to the Calibration Access and Data Handbook (see § 1.4). Some of these files are only used by the real time PHS and QLA systems, although they are contained in the SAS file ccf.cif. The SAS tasks using a given CCF are indicated as well.

6.2 Description of the OM image data processing chain `omichain`

The processing of OM data is run in pipeline mode at the SSC in Leicester. Two processing chains, `omichain` and `omfchain`, are used for image and fast mode data respectively. The process can be repeated by the users by running the same chains at their home institute. The chains are Perl scripts which concatenate the individual SAS tasks so as to go from the input raw data up to the final output results without further interaction. The tasks can also be run one by one having more control on all needed parameters.

The whole processing of image and fast mode data is depicted in Fig. 40 and Fig. 41.

Starting in Version 6, SAS contains also some tasks to process grism data and also a chain, `omgchain` to extract automatically the spectra from grism data. Fig. 42 gives an overview of the process. It should be noted that grism data processing has not been included up to now in the pipeline run at SSC in Leicester. The general reprocessing of all XMM-Newton data includes OM grisms, which from now on will be processed in the pipeline.

In order to process the OM image data successfully an input dataset is required. Input datasets are not changed during the OM processing with `omichain`. They are copied to intermediate datasets and keywords. Extensions are added to the FITS data files as needed. Intermediate task products, which are passed from one task to another, are generally not described.

The processing of OM image data can be divided in three parts (see Fig. 40)

- **Data preparation.** Before real processing, the four files composing a full frame low

File name	File Content	Usage
ASTROMET	coefficients for geometric distortion correction	omatt omdetect omdrifthist omgprep
BADPIX	bad pixels position, type of defect and the severity level	omcosflag
BORESIGHT	alignment of the instruments and star tracker	omatt
COLORTRANS	coefficients for color transformation into a standard system	ommag omdetect
GRISMCAL	wavelength and flux calibrations for the grisms	omgrism
HKPARMINT	house keeping parameter ranges	ommag
LARGESCALESENS	flat field map (set to unity)	omflatgen
LINCOORD	parameters for calculations of instrument configuration	not used
PHOTTONAT	correction coefficients of count rates for detector non-linearity and aging	ommag omprep omdetect
PIXTOPIXSENS	flat field map (set to unity)	omflatgen
PSF1DRB	point spread function for each filter	ommag omdetect
DARKFRAME	dark current map	QLA
DIFFUSEGALA	intensity map of diffuse galactic emissions	PHS
QUICKMAG	a rough count to magnitude conversion table for different spectral types	QLA
ZODIACAL	intensity map of the zodiacal light and average spectrum	QLA

Table 11: Calibration files of the Optical Monitor.

resolution exposure are combined into a single one using `omcomb`. Also a flat-field for later usage is obtained with `omflatgen`.

- **Image processing.** All corrections, source detection, astrometry and photometry are applied to each image file, exposure by exposure. `omichain` will process automatically all image data consecutively. However, the user can process single image files by applying the different tasks one by one in a semi interactive way.

  This core of the image processing can be divided as well

  – **Preparation of tracking corrections**

  – **Corrections: bad pixels, fixed pattern, flat-field**

  – **Source detection, astrometry and photometry**

- **Combined results.** The source lists corresponding to different exposures, using different filters, are combined with `omsrclistcomb`. Color indices are also calculated.

The images corresponding to different windows observed with the same filter are mosaiced in a single image file using `ommosaic`.

As it can be seen in the flow chart of the OM processing chain (Fig. 40), the OM tracking information for each exposure is treated before and independently from the image data processing. In the following, for each task description, a reference is given to the analysis step in the processing example.

6.2.1 Data preparation

- `omcomb` is used before real processing, to combine the four files composing a full frame low resolution exposure into a single one which will be the subsequent input for the process.
- `omflatgen` 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.

It should be noted that the characteristics of the OM detector make that flat field correction has to be treated in a different way than in a normal CCD. The physical pixels are subsampled by the centroiding algorithm, thus the concept of pixel to pixel variations changes since the real pixel cannot be recovered. A fixed pattern appears instead and it is corrected at a later stage in the processing.

It has been shown by studying the accuracy of the OM photometric calibration, which is based in observations of very many stars with OM and from the ground, and the statistical errors of the measurements, that the results obtained with OM are accurate within a few percent without flat field correction. Furthermore, observations of the same stars in different parts of the detector show that large scale sensitivity variations are smaller than two per cent. Therefore the SAS uses a unity flat field file for processing OM data. Studies of flat field correction in OM are being done to further demonstrate the validity of this approach.

6.2.2 Handling of tracking data

Two output datasets related to tracking data are distributed as pipeline products, namely the TSHPLT.PDF and the TSTRT.FIT file. They are created as follows.

- `omprep` (step1). The task `omprep` does not 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 from 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 the observation. This can also occur when certain instrument modes (engineering modes) or when some filter elements (e.g. grisms) are used.

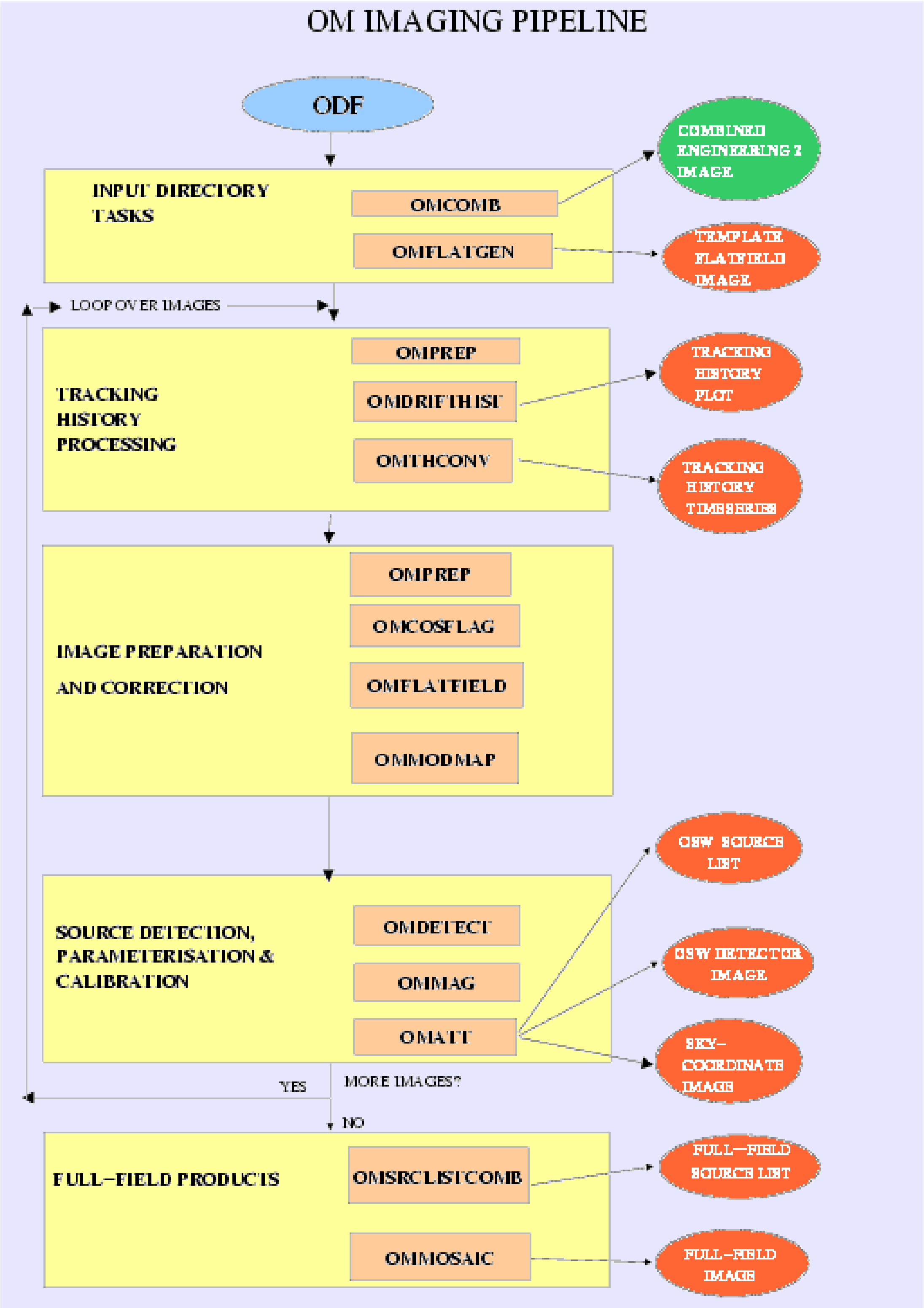


Figure 40: Pipeline processing of data acquired in the OM imaging mode.

- 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 TSTRTS.FIT file, which contains the tracking star time series, i.e the average count rate (cts/s) per tracking frame for each tracking star.

6.2.3 Handling of corrections applied to image mode data

As pointed out before, an observation with OM may be composed of several exposures which contain different observed windows. There are three final data products per observed science window (OSW), namely IMAGE_.FIT, SIMAGE.FIT and SWSRLI.FIT. Two additional products, FLAFLD.FIT and OBSMLI.FIT, are generated on a per observation basis. The task omichain loops over all image mode data files of all exposures in an observation. The following tasks are applied to each individual observation window.

- omprep (step4). The task omprep copies the raw images into an intermediate output file. It adds several keywords to this intermediate output file. Although this file is not a final data product, it is carried forward within the processing of OM image data and ends up being contained in the final IMAGE_ product.

- omcosflag (step5). The task omcosflag generates a quality map. Bad pixels are flagged taking also into account the tracking history information. The quality array (bad pixels are flagged as non-zero there) itself is written into the quality extension of the intermediate image files. It is used later by other tasks and it is propagated into the IMAGE_.FIT PPS output file.

- 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. Since the current flat field is unity, this correction has no effect on the data.

- 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.

6.2.4 Source detection, astrometry and photometry

- omdetect (step8). The task omdetect applies a source detection algorithm to find the sources present in the image. First the background is modelled. Then the algorithm looks for sources matching the criteria defined by the task parameters. Aperture photometry is applied to the detected sources. The point spread function (PSF) of OM is taken into account. The output is a source list (SWSRLI) which contains the source positions in absolute detector pixel and the calculated sources and background count rates (cts/sec). Coincidence loss correction is also applied to the count rates of the sources and the background. An optional output file, called the region file, can be generated with a name defined by the parameter `regionfile`. The region file describes the area associated with each identified source in the image. It is used by the image display tool.

Extended sources are also identified if the parameter `detectextended` is set. The photometry of an extended source is made by adding all the counts received in an elliptical area and the background is measured in an annulus, as for point sources. Coincidence loss correction is applied by considering that every pixel is a point like source.

- `ommag` (step9). The task `ommag` converts count rates into the instrumental OM photometric system. Count rates are corrected for detector deadtime. Time sensitivity degradation of the detector is computed for the epoch of the observation and the count rates are corrected for this effect. For the UV filters the PSF is extrapolated to account for the larger aperture of 35 pixels radius. The corrected count rates are converted into instrumental magnitudes. The task requires input from the source list file SWSRLI. The output is added as extra column to the source list file (SWSRLI). The sensitivity limit of the specific exposure and the time sensitivity degradation are written as keywords in the source list.

- `omatt` (step10). The task `omatt` reads the intermediate image generated by `ommodmap` 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 aligning the image. No other correction is applied to this image, i.e. coincidence losses and deadtime are not corrected. The skyimage provides the most accurate positional information of an image in the OM image data processing chain. Note that IMAGE_ has no distortion correction and therefore its positional information is less accurate than in the SIMAGE.

6.2.5 Final combined results

- `omsrclistcomb` (step11). The source lists of the different OSW's of all exposures contained in an observation are combined into one single observation source list file using `omsrclistcomb`, which is invoked when the analysis of the individual 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. `omsrclistcomb` generates a master source list with RA/Dec coordinates, RA/Dec errors and galactic coordinates. The list is sorted into increasing RA order. 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. The colour transformations are applied and the source brightness is also listed in a standard photometric system.

- `ommosaic` (step 12). In case of multiple OM exposures of the same field, `ommosaic` is used to combine the corresponding skyimages into a single skyimage covering the full field of view. This task is used for the default image mode, to combine the windows of the five exposures obtained with the same OM filter, but it can also be used to mix windows observed with different filters. Note that the final image is not corrected for coincidence losses nor for deadtime. The exposure times of the original images composing the mosaic are normalized to 1000 seconds.

6.2.6 Further notes on processing of OM image data

With the default settings, `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`.

6.2.7 Interactive source detection and photometry

The task `omsource` allows the user to either create or modify an existing source list file by selecting sources on a displayed image. Interactive photometry can be done on the selected sources. The source list file is compatible for use as an input file to the SAS task `omsrclistcomb`. The task can also be used on images and source list files processed prior to SAS 5.4.

`omsource` needs as input an OM image (IMAGE_) which has undergone the corrections explained in Section 6.2.3 and eventually a source list (SWSRLI). A graphic interface and a `ds9` type display will be presented to the user. Then, sources on the list can be analyzed interactively. The photometry can be re-done and the new values will be incorporated into the source list. Sources can be deleted from the list, and new ones can be added. If no input source list is given, a new one will be created. If desired, then `omdetect` can be run on the image and the detected sources can be analyzed.

Since SAS 7.1 there is a new task `omphotom` intended also for interactive photometry. It does not have a graphical user interface, but it runs in the command line.

`omphotom` allows the user to recompute the photometry of one or several sources in a list previously obtained with `omichain`. The aperture and background regions definition can be modified. For crowded fields, a least squares fit to the PSF can be done. This can give better results than the standard mode used in omichain.

The task can accept as input a parameter file that allows specifying several input images and the corresponding source files. So it is very well suited for batch processing of many exposures in the same observation.

Potential users are referred to the on-line documentation for these two interactive tasks `omphotom` and `omsource`.

6.3 Description of the OM fast mode data processing chain `omfchain`

The example input dataset given in Table 10 contains also fast mode data files. To process them with SAS the task `omfchain` has to be used. As for image data, the input data do not change through the fast mode chain.

The process is similar to the one for image data. It can be followed in Fig. 41. For each fast mode window of the different exposures in the observation (ODF) there will be

- **Data preparation: tracking correction.** The tracking data, which are common for image and fast mode data, are treated to derive the corrections to be applied to the events detected in an exposure.
- **Event selection and corrections.** The event list file is scanned through to find the photon events. Then tracking and flat field corrections are applied.
- **Source detection and astrometry.** The source is located within the fast mode window (eventually more than one source) and astrometry conversions are applied.
- **Light curve.** Count rates are derived for the source(s) and the background. Then a light curve is produced giving the net countrate as a function of time for each fast window in each exposure. A graphical output is produced as well. In the future all light curves obtained with the same OM filter will be merged.

6.3.1 Data preparation: tracking correction

The same tasks used in the image chain are repeated here:

- `omprep` (step1). The task `omprep` does not 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 TSTRTS.FIT file, which contains the tracking star time series, i.e the average count rate (cts/s) per tracking frame for each tracking star.

6.3.2 Event selection & corrections

The following tasks are applied to each individual fast mode window

- `omprep` (step4). The task `omprep` applied to the event list file will augment its header contents with house-keeping information. A modified intermediate event list file will be created.

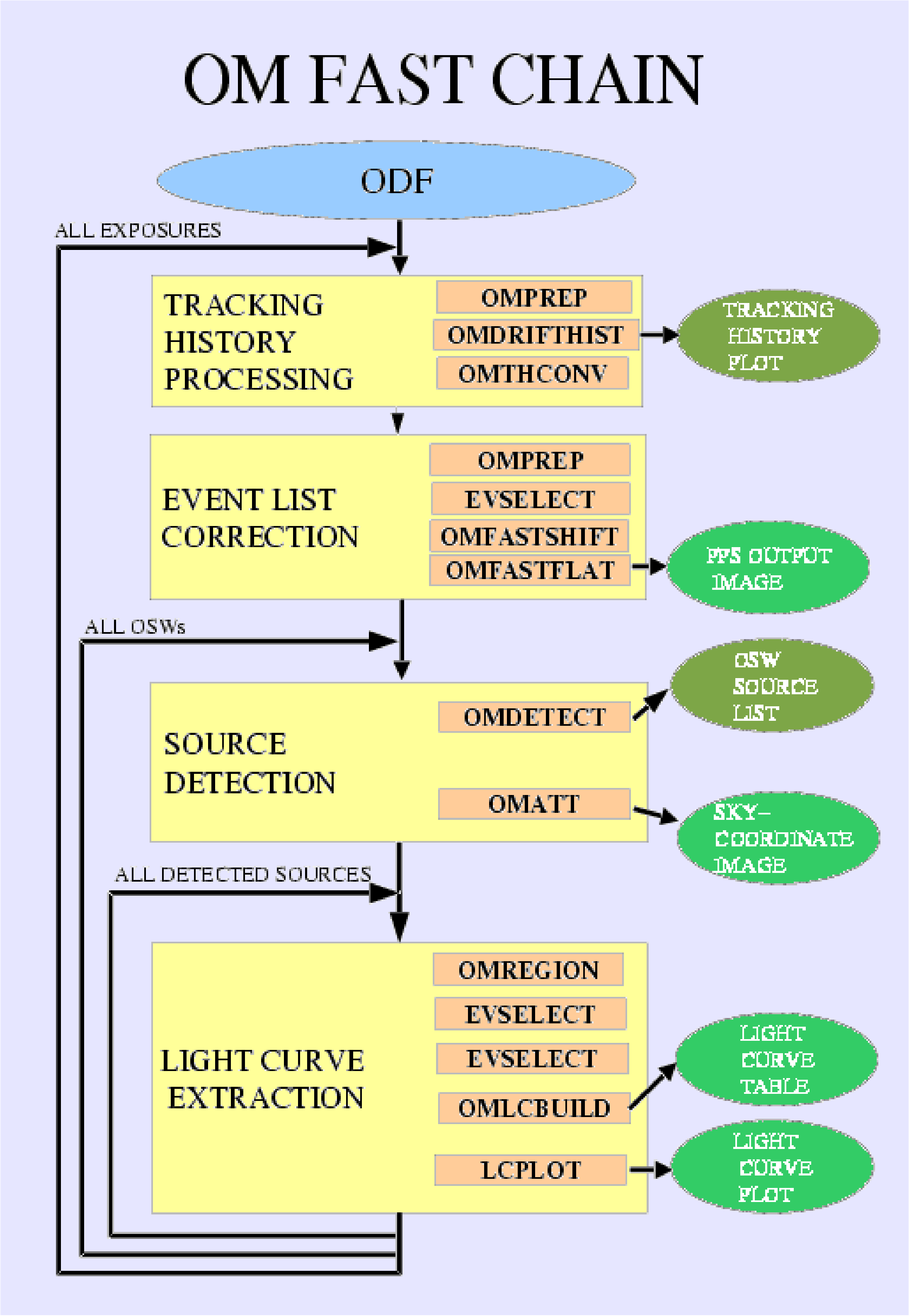


Figure 41: Pipeline processing of data acquired in the OM fast mode.

- `evselect` (step5). The task `evselect` extracts the photon events from the event-list file and builds a pseudo image corresponding to the fast mode window.

- `omfastshift` (step6). The task `omfastshift` uses the tracking history file (as contained in the intermediate processing product of the tracking history task) to correct the position of the photon events for drifts of the spacecraft during the exposure. New columns are added to the event-list file with corrected coordinates.

- `omfastflat` (step7). The task `omfastflat` applies the tracking history file to the OM flatfield file (obtained from `omflatgen`) to build a shifted flat field corresponding to the fast mode pseudo image.

  We point out again the peculiarity of the flat field for OM. As for image data, a unity flat field is applied to fast mode data as well.

6.3.3 Source detection & astrometry

- `omdetect` (step8). The task `omdetect` applies the source detection algorithm to the fast mode pseudo image to locate the existing sources. It builds a source list similar to the image data one (SWSRLI) which contains the source position in absolute detector pixel and the calculated source and background count rates (cts/s).

- `omatt` (step9). The task `omatt` performs astrometry conversions from pixel positions to astronomical coordinates in the source list and rotates the pseudo image to obtain a north aligned skyimage.

- `omregion` (step10). The task `omregion` produces the regions for the source and background needed later for `evselect`

- `evselect` (step11-12). The task `evselect` is run twice consecutively on the event-list file to extract the photon events for the source and the background in the specified time sampling. Intermediate rate files are created for the source and the background.

- `omlcbuild` (step13). The task `omlcbuild` generates coincidence loss and dead-time corrected source timeseries using the source and background rates produced by `evselect`. The background rates are subtracted and the photometry corrections are applied. If the source is offset from the centre, the knowledge of the PSF is used to account for possible missing photons. Time dependent sensitivity degradation is corrected as for image data. A light-curve is obtained as final result for each of the fast mode windows in the exposure.

- `lcplot` (step14). The task `lcplot` reads the ligh-curve file and makes some plots. Some statistics is also applied to assess possible source variability.

6.4 Description of the OM grism data processing chain `omgchain`

The processing of grism data is very similar to the one performed in image data by `omichain`. Some tasks used there are also applied to grism data. The whole process is shown in Fig. 42. It can be divided in:

- **Data preparation.**

  For data obtained in full frame (field spectroscopy), the four files composing a full frame low resolution exposure are combined into a single one using `omcomb`.

- **Image processing.**

  As in all OM image mode data, `omprep` is run to extract information from the tracking history and house-keeping files and to add it to the header. `ommodmap` performs a modulo_8 correction on the grism image to eliminate this fixed pattern noise.

  Then the grism image is corrected from geometric distortion of the detector and then it is rotated so as to have the dispersion direction (the spectra) aligned with the image Y axis. The task `omgprep` is used to achieve this.

- **Source detection, spectral extraction and calibration**

  `omdetect` is used also in grism data to search the image for spectra, both zero and first orders. A source list is generated with all successful detections.

  Then, `omgrism` will analyse the detections in the source list to establish relations between detected zero orders and the corresponding first orders. A boxcar extraction is performed on the successful relations. The extracted spectra are calibrated in wavelength and flux. In case a relation cannot be found for well defined zero orders, a default extraction is used for the zero order. The final spectra are written in a fits file. Finally, `omgrismplot` is used to produce plots of the spectra (net spectrum, background and flux calibrated) in PDF and PS formats.

The complete processing chain extracts by default only the spectrum of the main target, or object placed at the XMM-Newton boresight, and this is what the user will find in the final products. Alternatively, the user may select to extract all objects in the field of view ("extractfieldspectra=yes").

6.4.1 Grism specific tasks

Some of the tasks used in the processing of grism data are the same ones used in `omichain` and they perform the same functions and produce similar output. These tasks are `omcomb`, `omprep` and `ommodmap` (steps 0-1-2).

Other tasks are tools for spectroscopic data processing:

- `omgprep` (step 3) Grism images, as any other OM image, are affected by geometrical distortion which is corrected in the case of grisms by `omgprep` so as to obtain a linearized image. The grisms dispersion direction is inclined with respect to the detector axes. To facilitate the extraction of the spectrum, the image, after correcting it for distortion, is rotated so that the dispersion coincides with the Y axis. The rotation is done also by `omgprep`. The output from this task is a file (RIMAGE) from which spectral extraction can be done with any processing system. As we shall describe later, this file is the input for the interactive `omgsource`.

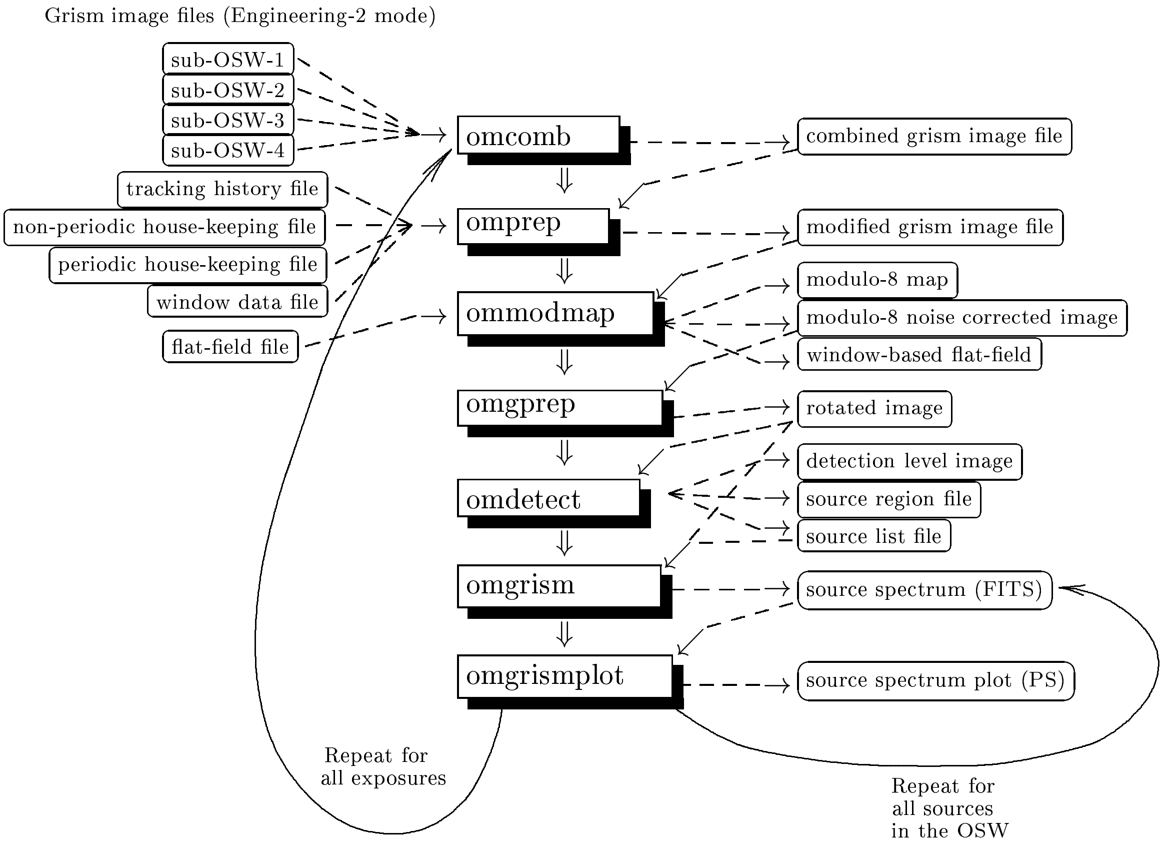


Figure 42: Processing of OM grism data.

- omdetect (step 4). The same detection algorithm used in image mode data is applied to grism images coming from omgprep (RIMAGE) to search for spectral signatures, namely zero and first orders, present in the image. Each source is labeled as zero or first order depending on its shape. The results of the detection are written in a source list (SWSRLI) which contains the detected source positions in detector pixels in the rotated image. A ds9 region file type (REGION) contains all performed detections.

- omatt (step 4.1). The OM astrometry task omatt has been introduced in SAS 7.0. It is applied to the source list (SWSRLI) produced by omdetect to compute the astronomical coordinates of the detected spectra. New columns with RA/Dec positions are added to the source list for all detected zero and first orders.

- omgrism (step 5). Spectral extraction and calibration of the spectra is performed by omgrism. The source list generated by omdetect is analysed to search for the spectrum of the target. If it cannot be found at the proper position, then a default extraction is performed. The positions of the zero and first orders and a flag indicating the nature of the spectrum are written in another source list (SPECLI). A ds9 region file type (SPCREG) depicts the zero and first orders.

  If the parameter "extractfieldspectra=yes" was set, then omgrism is used to establish correlations between the detected zero and first orders. The correlation is declared when the X coordinates of both orders are within a certain range and also their Y coordinates. After finding all the possible relations, the corresponding spectra are extracted. The proper identification of the zero order is essential since its position fixes the wavelength scale. In case a first order cannot be associated with a zero one, or viceversa, a default extraction is used. All detected zero and first orders are included in the SWSRLI source file, as well as the relations found among them. Only the successful ones giving rise to an extracted spectrum and the default extractions are written into the SPECLI source file and the SPCREG region file.

  The performed extraction is by default a box car, with predefined width and positions around the first order (or the Y coordinate of the zero order, if defaulted) for the spectrum and background regions. Signal weighted extraction is optional although its use is not recommended for the moment.

  The extracted net spectrum and the corresponding background are calibrated in wavelength and they are written to the output fits file (SPECTR) as count rate per angstrom in the whole spectral range contained in the image. Then the flux calibration, defined in a restricted wavelength range for each grism, is applied and the fluxed spectrum is written to the SPECTR fits file. The corresponding errors are also written to this file.

- omgrismplot (step 6). The produced spectra are plotted by omgrismplot in PDF and/or Postscript format. Net spectrum, background and fluxed spectrum are plotted.

6.4.2 Astrometry on grism images

As it has been mentioned in the previous section, starting with SAS Version 7.0, the grism processing chain includes the astrometry task omatt so that the sources of all extracted spectra are assigned astronomical coordinates that facilitate their identification. This is particularly useful in case of multiobject spectroscopy (full frame observations).

6.4.3 Interactive spectral extraction

Sometimes spectra are very faint, or the desired spectrum is contaminated by other close zero or first orders, or there can be straylight features, all of these not properly handled by the automatic processing done by omgchain. In this case the user can apply omgsource. This task takes a rotated image coming from omgprep, displays it in the terminal and the user can select with the cursor the zero and corresponding first orders to be extracted. The extraction width and the background region definition can be changed as well so as to avoid contaminating features. The task passes the selected values to omgrism and the extraction and calibration continues. The user names the new source list and spectrum files.

The user selected position for the zero order can be refined by omgsource using a centroiding mechanism. Since the position of the zero order determines the wavelength scale, this centroiding can be turned off by the user in the case that close, contaminating features may perturb its results.

An existing source list can be loaded on top of the rotated image so as to check it and to modify it interactively.

6.4.4 Further notes on processing of OM grism data

OM grism data are complex, and a grism image can contain a lot of sources, zero and first spectral orders, in addition to complicated straylight features affecting the background and the spectra. This makes the automatic extraction process very difficult and prone to spurious detections and extractions. The user must check carefully the produced spectra. The source list of extracted spectra produced by omgrism is also represented in a ds9 type region file (SPCREG) which can be overlaid on the rotated image to verify the correct extraction for the target spectrum or for all spectra in the field.

The SPECLI file tells us whether each extracted spectrum corresponds to the main target, or to a default extraction, or to a field object. The keyword OBJ_TYPE flags this also in the SPECTR file.

In case of problems or doubts, it is recommended to run the interactive omgsource after omgprep.This task uses as input the rotated image already mentioned.

The described rotated image RIMAGE contains all corrections proper to OM grism data. Therefore it can be used within any spectral reduction package or customized reduction system to perform the extraction of the spectrum, which can afterwards be calibrated using the contents of OM_GRISMCAL CCF.

It should be noted that coincidence loss corrections are currently not applied to grism data. Also, there is no correction for flux in the wings of the cross dispersion profile if the user selects interactively an extraction swath narrower than the base width of the spectrum.

Astrometric corrections are applied in grism data. The coordinates of the targets in the field of view corresponding to all extracted spectra (“extractfieldspectra=yes”) are written in the spectra source list and in the headers of the spectra files. It should be noted that these coordinates are less accurate than the ones obtained in normal image data.

6.5 OM SAS processing products

This section provides a general description of the final products obtained when running the SAS chains on OM data.

The chains produce intermediate files that are sometimes overwritten by different tasks. They are recognized by the first letter of their name ("I" in omichain, "F" in omfchain, "g" and "u" in omgchain). Their detailed description can be found in the on-line documentation.

Similar files are generated by the standard pipeline processing run at the SSC, but the user shall be aware that they may have subtle differences between the two sets. All intermediate files are deleted in the pipeline processing. By running the tasks interactively, the user can maintain these intermediate products for a better understanding of the whole process.

OM processing chains and pipeline products are named as "zooooooooooOMueeettttttsxxx.fff", where:

- z denotes final product (P: image or fast, p: grism) or intermediate file (I, F, g, u) as defined above
- oooooooooo stands for the observation ID,
- u is the exposure flag (S:scheduled, U:unsched, X:not applicable),
- eee is the exposure ID,
- tttttt is the data type,
- s is the OM window within the exposure,
- xxx is the source number within the window,
- fff is the file type (FIT, FTZ, PDF, PNG etc.).

In the following we refer to a data type by the tttttt identifier, e.g. such as tttttt= SIMAGE for the OM sky image product. We omit the window and source number identifier (sxxx).

In combined images resulting from `ommosaic` or after processing the full frame image produced by `omcomb`, files with data type IMAGE_ and SIMAGE, "s" indicates the corresponding filter used. The optical filters are noted V, B and U, while L, M and S represent UVW1, UVM2 and UVW2 respectively.

There are two products associated with the tracking history files, namely the TSHPLT.PDF and the TSTRTS.FIT file. There are three data products per OSW, namely IMAGE_.FIT, SIMAGE.FIT and SWSRLI.FIT. If fast mode data are present, there are two products per OSW, namely TIMESR.FIT and its graphics version TIMESR.PDF. There are two products generated on a per observation basis, which are FLAFLD.FIT and OBSMLI.FIT.

It should be noted that the pipeline produces compressed FITS files named .FTZ while the output obtained when the chains are run by the user is not compressed , .FIT.

TSHPLT.PDF provides a visualization 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 s. 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 at the beginning of a 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 systematic drift into one direction. Typically the pointing stability is better than 1 arcsec during one exposure, which corresponds to 2 pixel on the diagrams. There is one TSHPLT PDF file produced for each THX file.

TSTRTS.FIT 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 count rate (cts/s) of a tracking star averaged over the tracking frame duration. Each row indicates a new tracking frame and one column corresponds to a time series 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 WCS parameters applied to the raw image. 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. The pseudo images generated for fast mode windows follow this naming too.

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. Coincidence losses and dead time are not corrected in this image. The north aligned sky pseudo images generated for fast mode windows follow this naming too.

SWSRLI.FIT file: The primary header of an OSW source list file contains, besides the definition of the OSW within the exposure and the observation, the sensitivity limit of the plate expressed in count rates and in instrumental magnitudes. The binary extension holds the list of detected sources. There are the following entries for each detected source: identifier within the OSW, source position in pixel, 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. The last entry is the source identifier in the final combined source list.

FLAFLD.FIT file: This image is 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. It is set to unity.

TIMESR.FIT file: The time series contains source (background subtracted) and background rates and their error estimates as a function of time, within the specified time sampling.

TIMESR.PDF file: The time series is plotted here to produce a graphic light curve, together with some statistics performed on the data.

OBSMLI.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: the observation source identifier, the position in RA/Dec and galactic coordinates, the positional uncertainty, the detection significance and corrected count rates 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 characterization and orientation of the spatial extent (described as in the SWSRLI files by the two semi-axes and the orientation angle with respect to the image x-axis) for each filter, and finally, if a transformation was possible, the source brightness in a standard photometric system, as well as the color indices.

AB magnitudes and fluxes for all filters are included as well.

In SAS version 6.5, all these quantities in the OBSMLI file were vectorized (values for each filter) in order to reduce the number of file columns. In newer versions there is again one simple column for each computed parameter and filter.

IMAGE_.FIT file: This image is similar to the normal image processing one. It is a copy of the original raw image, with completed header information and modulo_8 correction.

RIMAGE_.FIT file: The undistorted and rotated image produced by `omgprep`. It is the basis for the automatic spectral detection and extraction, and for interactive extraction with `omgsource`. This file may also be named **GIMAGE_.FIT** in future pipeline products.

SWSRLI.FIT file: It contains the position and characteristics of the detected zero and first orders and the correlations found between them. The results of the grism astrometry are included in this file.

REGION.ASC file: This is an ASCII version of the SWSRLI source list file for use with `ds9` display.

SPECLI.FIT file: This is similar to the SWSRLI file, but it contains only the zero and first order positions of the main target extracted spectrum, or if "extractfieldspectra=yes" was used, all final extracted spectra. For each extracted spectrum, the nature of the source is also given (Target, Field Object, Default Extraction,...)

SPCREG.ASC file: This is an ASCII version of the SPECLI source list file for use with `ds9` display.

SPECTR.FIT file: The extracted and calibrated spectra are written in this file. For each spectrum a table extension is given. On its columns we can find:

```
    -  wavelength
    -  net spectrum (count rate)
    -  error in net spectrum
    -  background rate
    -  error in background
    -  flux calibrated spectrum
    -  error in flux
```

SPECTR.PDF / SPECTR.PS files: They contain plots of the extracted spectra in the corresponding SPECTR.FIT file.

6.6 Running the OM data processing

The data package received by the investigator contains OM data which normally do not necessitate further processing for the purpose of calibration. However, a user may want to apply the most recent calibration, or to change some default parameters to e.g. improve the source detection on his/her data. It may not be necessary to run the complete chains, but just some tasks. All this can be done interactively.

The SAS_ODF environment variable shall be set to a directory containing the data, or directly to the SAS summary file, and access to calibration files shall be set through `cifbuild` (see § 2.3).

The OM tasks and chains can be run on a working directory different from where the raw data are, providing that the environment variables are properly set.

Invoking `omichain`, `omfchain` or `omgchain` will automatically start the processing of all referred OM data. The duration of the process will depend on the number of exposures and windows and at the end we shall obtain in the working directory the processed files described before. Some intermediate files will be preserved as well.

Currently only `omichain` can accept parameters to limit the processing to a given filter or a given exposure:

```
     omichain filters=filters    ( V, B, U, UVW1, UVM2, UVW2)

     omichain exposures=numbers  (exposure numbers)

  e.g.
          omichain filters=V
          omichain filters="U B V"
          omichain exposures=006
          omichain exposures="006 007 008"
```

Some of the default parameters used by individual tasks can also be tuned. To see all possible parameters run:

```
     omichain -h
```

In the following examples, we show how to run all the tasks that form a chain. We outline the fundamental parameters needed by each task. For a detailed description of all possible parameters, the user is referred to the SAS online documentation at:

http://xmm.esac.esa.int/sas/current/doc/

For a better understanding of the processing done by any of the chains, it is recommended to re-direct their screen output to a log file. This will allow the user to examine the process and to see if there is any error or anomaly.

If a user wants to fully understand the process, then running a chain task by task may be useful. In this case it is recommended to have the raw data to be processed and the auxiliary files in the working directory.

6.6.1 Example of image data processing

Due to the number of input parameters, lengthy file names, etc, it is not recommended to run any OM chain step by step. Only in special cases, some tasks, as `omdetect`, `ommag`, `omatt`, `omsrclistcomb`, `lcplot`, may have to be run so as to fine tune some parameters. In some cases it might be more advantageous to use the interactive `omsource` or `omgsource`.

To process task by task a single exposure in image mode, we need in addition to the housekeeping files listed in Table 10, the following files:

```
0472_0125910501_SCX00000SUM.SAS  - ASCII observation summary file
0472_0125910501_SCX00000TCS.FIT  - Spacecraft Time correlation file
0472_0125910501_SCX00000ATS.FIT  - Spacecraft Attitude file
0472_0125910501_OMS00400WDX.FIT  - Exposure priority window file
0472_0125910501_OMS00400THX.FIT  - Exposure tracking history file
0472_0125910501_OMS00400IMI.FIT  - Exposure image file to process
```

These files can be located in the working directory, or in any other one, providing that the environment variable SAS_ODF has been set accordingly. In our examples we assume that all necessary files are in the working directory.

The process will be run in the following way.

Although, as it was said before, no real flat field correction exists for OM, nor it is necessary, the processing requires such a file which can be generated using the task `omflatgen` as follows.

```
step0> omflatgen outset=P0125910501OMX000FLAFLD0000.FIT
```

The output flatfield (primary extension) will be set to unity.

```
step1> omprep set=0472_0125910501_OMS00400THX.FIT \
              nphset=0472_0125910501_OMX00000NPH.FIT \
              pehset=0472_0125910501_OMX00000PEH.FIT \
              wdxset=0472_0125910501_OMS00400WDX.FIT \
              outset=tmp_tracking modeset=3
```

In case there is no THX file, then `set`=DUMMYTHX.FIT. `omprep` will generate a dummy file needed for the rest of the chain, with zero drift in it.

```
step2> omdrifthist set=tmp_tracking plotfile=P0125910501OMS004TSHPLT0000.ps

step3> omthconv thxset=tmp_tracking nphset=0472_0125910501_OMX00000NPH.FIT \
                outset=P0125910501OMS004TSTRTS0000.FIT  modeset=0

step4> omprep set=0472_0125910501_OMS00400IMI.FIT \
              nphset=0472_0125910501_OMX00000NPH.FIT \
              pehset=0472_0125910501_OMX00000PEH.FIT \
              wdxset=0472_0125910501_OMS00400WDX.FIT \
              outset=tmp_image_1 modeset=0

step5> omcosflag set=tmp_image_1 thxset=tmp_tracking \
                 samplefactor=1 timefactor=1

step6> omflatfield set=tmp_image_1 thxset=tmp_tracking \
                   inorbitflatset=P0125910501OMX000FLAFLD0000.FIT \
                   tsflatset=tmp_flat_field outset=tmp_image_2 \
                   samplefactor=1

step7> ommodmap set=tmp_image_2 mod8product=yes mod8set=tmp_mod8 \
                outset=P0125910501OMS004IMAGE_0000.FIT \
                nsig=3 nbox=16 mod8correction=1

step8> omdetect set=P0125910501OMS004IMAGE_0000.FIT \
                outset=P0125910501_OMS004SWSRLI0000.FIT \
                levelimage=level_image regionfile=region_file \
                nsigma=2 minsignificance=1 detectextended=yes

step9> ommag set=P0125910501_OMS004SWSRLI0000.FIT \

step10> omatt set=P0125910501OMS004IMAGE_0000.FIT \
              sourcelistset=P0125910501_OMS004SWSRLI0000.FIT \
              ppsoswset=P0125910501OMS004SIMAGE0000.FIT
```

The detected sources can be overlaid on the OSW by using `implot`, or `ds9`.

In the standard automatic SSC pipeline processing, the tmp_image files are re−used and thus overwritten. In the task by task processing, they are distinguished so that intermediate stage output can be looked at if desired. 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.

```
step11> omsrclistcomb sourcelistsets=P0125910501OMS004SWSRLI0000.FIT,....... \
                      nsigma=3 outset=P0125910501OMS000OBSMLI0000.FIT
```

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

Finally, the sky images of the multiple exposures corresponding to the default image configuration can be combined in a single sky image of the field of view. Note that this mosaiced image cannot be used for photometric purpose.

```
last step> ommosaic imagesets=‘‘list of sky images’’ \
                    mosaicedset=myfield_in_myfilter.fit
```

6.6.2 Example of fast mode data processing

In a similar way, if an exposure contains fast mode data, the following files will be needed

```
0472_0125910501_OMS00400WDX.FIT  - Exposure priority window file
0472_0125910501_OMS00400THX.FIT  - Exposure tracking history file
0472_0125910501_OMS00401FAE.FIT  - Exposure fast mode data file
```

and the process can be run task by task in the following way

```
step1> omprep set=0472_0125910501_OMS00400THX.FIT \
              nphset=0472_0125910501_OMX00000NPH.FIT \
              pehset=0472_0125910501_OMX00000PEH.FIT \
              wdxset=0472_0125910501_OMS00400WDX.FIT \
              outset=tmp_tracking modeset=3
```

As for image data, if there is no THX file, then `set=DUMMYTHX.FIT`. `omprep` will generate a dummy file needed for the rest of the chain, with zero drift in it.

```
step2> omdrifthist set=tmp_tracking plotfile=P0125910501OMS004TSHPLT0000.ps

step3> omthconv thxset=tmp_tracking modeset=1 \
                nphset=0472_0125910501_OMX00000NPH.FIT \
                outset=P0125910501OMS004TSTRTS0000.FIT

step4> omprep set=0472_0125910501_OMS00401FAE.FIT \
              nphset=0472_0125910501_OMX00000NPH.FIT \
              pehset=0472_0125910501_OMX00000PEH.FIT \
              wdxset=0472_0125910501_OMS00400WDX.FIT \
              outset=event_list.fit modeset=1
```

In its second run, `omprep` is invoked for fast data (modeset=1) and the FAE raw event data is transformed into a modified event list to be used as input for `evselect`

```
step5> evselect table=event_list.fit xcolumn=RAWX ycolumn=RAWY \
                ximagebinsize=1 yimagebinsize=1 \
                withimageset=true imageset=raw_image.fit
```

The pseudo-image corresponding to the fast mode OSW has been created by this first run of `evselect`.

```
step6> omfastshift set=event_list.fit thxset=tmp_tracking \
                   nphset=0472_0125910501_OMX00000NPH.FIT
```

The X- and Y- coordinates of the photon events are corrected for spacecraft drift. New columns are added to the event list with the corrected values.

```
step7> omfastflat set=event_list.fit \
                  slewflatset=P0125910501OMX000FLAFLD0000.FIT \
                  oswflatset=fast_flat.fit \
                  fastimgset=P0125910501OMS002IMAGE_1000.FIT
```

Here again, as for image mode processing, the system is prepared to apply a subset of the omflatgen generated flat field to the fast mode window data, taking into account the spacecraft drift. The flat field is set to one, and therefore this correction has no real effect. The task generates the tracking shifted fast_flat.fit (only for the fast mode window) and the corrected pseudo image image.fit.

```
step8 >omdetect set=P0125910501OMS004IMAGE_1000.FIT \
                regionfile=region_file nsigma=6 \
                outset=P0125910501OMS004SWSRLI1000.FIT
```

The output region will allow the user to check the proper detection of the source in the small fast window. The PSF information is used to parameterize the detected source.

```
step9> omatt set=P0125910501OMS004IMAGE_1000.FIT \
             sourcelistset=P0125910501OMS004SWSRLI1000.FIT \
             ppsoswset=P0125910501OMS004SIMAGE1000.FIT \
             usecat=no
```

Astrometry is applied as for image data. The pseudo-image is north aligned too.

```
step10> omregion set=P0125910501OMS004SWSRLI1000.FIT \
                 srcnumber=1 srcradius=-6 nfwhm=3 bkginner=1.2 bkgouter=2.5 \
                 bkgfile=back_region.fit srcfile=source_region.fit
```

These regions will be used by evselect to filter out the event list, extracting the corresponding photon events for the source and the background. Optional parameters can be used to fine-tune the definition of the regions.

```
step11> evselect table=event_list.fit xcolumn=CORR_X ycolumn=CORR_Y \
                 expression=‘‘((WIN_FLAG .eq. 0) .and. \
                 (region(source_region.fit, CORR_X, CORR_Y)))’’ \
                 maketimecolumn=T withrateset=Y timebinsize=10 \
                 rateset=source_rate.fit

step12> evselect table=event_list.fit xcolumn=CORR_X ycolumn=CORR_Y \
                 expression=‘‘((WIN_FLAG .eq. 0) .and. \
                 (region(back_region.fit, CORR_X, CORR_Y)))’’ \
                 maketimecolumn=T withrateset=Y timebinsize=10 \
                 rateset=back_rate.fit
```

And finally the light curve can be obtained and plotted. The photometric corrections are also applied (coincidence loss, dead-time, PSF, magnitude conversion)

```
step13> omlcbuild srcregionset=source_region.fit bkgregionset=back_region.fit \
                  srcrateset=source_rate.fit bkgrateset=back_rate.fit \
                  sourcelistset=P0125910501OMS004SWSRLI1000.FIT \
                  wdxset=0472_0125910501_OMS00400WDX.FIT \
                  outset=P0125910501OMS004TIMESR1000.FIT

step14> lcplot set=P0125910501OMS004TIMESR1000.FIT \
               binsize=1 plotfile=lightcurve.ps
```

6.6.3 Example of grism data processing

As we have pointed out OM grism data are image mode data. Since tracking corrections are not applied to grism data, we need in addition to spacecraft and summary files:

```
0472_0125910501_OMS00500WDX.FIT  - Exposure priority window file
0472_0125910501_OMS00500IMI.FIT  - Exposure image file with grism data
```

and the process will be run in the following way

```
step1> omprep set=0472_0125910501_OMS00500IMI.FIT \
              pehset=0472_0125910501_OMX00000PEH.FIT \
              nphset=0472_0125910501_OMX00000NPH.FIT \
              wdxset=0472_0125910501_OMS00500WDX.FIT \
              outset=g0125910501OMS005IMAGEI0000.FIT \
              modeset=4
```

The input image corresponds to a user defined window. In case we had a full frame low resolution exposure, then `omcomb` must be run beforehand and its output image be used as input for `omprep`

```
step2> ommodmap set=g0125910501OMS005IMAGEI0000.FIT \
                mod8product=yes \
                mod8set=g0125910501OMS005MOD8MP0000.FIT \
                outset=g0125910501OMS005IMAGE_0000.FIT \
                outflatset=g0125910501OMS005FLAFLD0000.FIT \
                nsig=3 nbox=16 mod8correction=1

step3> omgprep set=g0125910501OMS005IMAGE_0000.FIT \
               outset=p0125910501OMS005RIMAGE0000.FIT
```

We have now the undistorted and rotated image from which the spectrum, or spectra, will be extracted.

```
step4> omdetect set=p0125910501OMS005RIMAGE0000.FIT \
                regionfile=g0125910501OMS005REGION0001.ASC \
                outset=p0125910501OMS005SWSRLI0001.FIT \
                nsigma=2
```

The source spectra have been detected. They can be checked by overplotting the region file on the rotated image, using `ds9`.

```
step5> omatt set=p0125910501OMS005RIMAGE0000.FIT \
             sourcelistset=p0125910501OMS005SWSRLI0001.FIT \
             ppsoswset=g0125910501OMS005SIMAGE0000.FIT
             usecat=no rotateimage=yes tolerance=3 maxradecerr=1 maxrmsres=1.5
```

All detected spectra have been assigned RA and Dec. A sky, north aligned image is also produced.

Spectral extraction will be done now

```
step6> omgrism set=p0125910501OMS005RIMAGE0000.FIT \
               sourcelistset=p0125910501OMS005SWSRLI0001.FIT \
               outset=p0125910501OMS005SPECTR0000.FIT \
               bkgoffsetleft=6 bkgwidthleft=-6 bkgoffsetright=6 \
               bkgwidthright=-6 spectrumhalfwidth=-6 spectrumsmoothlength=0
               extractionmode=0 extractfieldspectra=no
               outspectralistset=p0125910501OMS005SPECLI0000.FIT
               regionfile=p0125910501OMS005REGION0001.ASC
               spectraregionfile=p0125910501OMS005SPCREG0001.ASC
```

Target spectrum, or all spectra in the field of view if

```
extractfieldspectra=yes
```

is used, are extracted and calibrated.

Finally, we can plot the results.

```
step7> omgrismplot set=p0125910501OMS005SPECTR0000.FIT \
                   binsize=1 plotdevice=/PS \
                   plotfile=g0125910501OMS005SPECTR0000.PS \
                   scalebkgplot=no plotflux=2
```

A PDF version of the plot is also produced.

6.7 Analysing OM data

As it has been pointed out already, OM data in image mode and fast mode with the normal filters, are fully processed by the SAS pipeline. All data, including the grisms can be run through the corresponding chain: for each exposure of a given observation all necessary corrections are applied to the data files. Then a source detection algorithm is used to identify the sources present in the image. Standard aperture photometry is applied to obtain the count rates for all detected sources. These rates are corrected for coincidence losses and dead time of the detector and finally OM instrumental magnitudes and standard colour corrections are computed. A final source list is obtained from all exposures and filters. The detector geometric distortion correction and astrometric corrections are applied to each source's position and also the whole image is converted to sky co-ordinates space and rotated so as to have the North on the top. Default image windows are also combined to obtain a mosaic (per filter) of the FOV. In the case of grism data, the spectra are searched for, extracted and calibrated.

However, one has to be aware of some peculiarities of OM data before starting the analysis.

- **Artifacts**

  OM images show several artifacts. These artifacts are generallly only visible when viewing the OM images in a high contrast and in logarithmic display. Artifacts can however affect the accuracy of a measurement, e.g. by increasing the background level. The following artifacts 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 algorithm in 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 image. These loops can degenerate into long streaks depending on the source positions.

- **Grism Data**

  We have already mentioned the complexity of grism data, increased by the eventual presence of the artifacts described before (see § 6.4.4). A great care has to be taken when analysing the output products of grism data processing with SAS.

The following points should also be kept in mind:

1. The OM operates in photon counting mode but images are accumulated on board. Good time intervals (GTI) have no sense in OM data. Either the full exposure is selected or discarded.

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

3. As it has been explained, a flatfield response of unity is assumed.

4. 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.

5. The corrections for coincidence losses and for detector deadtime are applied when aperture photometry is performed on the detected sources. These corrections are not applied to any of the produced images.

6. The OM filters do not form a proper photometric system. However the photometric calibration of the instrument, based on observations with OM and from the ground, allows the SAS to obtain standard U, B, V magnitudes and colours in the Johnson system. This applies to stars. Extended objects should be treated with care.

7. The OM point spread function (PSF) has wide wings. This is taken into account in the application of the calibration with SAS through the proper setting of photometric apertures. A similar approach has to be taken if one wants to make an independent processing and analysis of OM data using any other data reduction package.

8. Straylight features, already mentioned, complicate the background subtraction in some cases, specially when the target star lays in or close to a straylight feature.

9. Imaging data are corrected for time sensitivity degradation, so that data of a given source (count rates, photometry) obtained at different epochs can be compared. Fast mode data processing does not correct for time sensitivity degradation. Grism data are not corrected either.

In principle, after SAS has been run there is no need for further data reduction. The observation source list should contain the calibrated data with their errors. Therefore the user can proceed with the analysis and interpretation of the processed data. However, some checking is convenient to verify the consistency of the data output.

The whole data processing can be repeated easily by a Guest Observer or XSA User, should any calibration file be updated, and what is more important, in case of doubtful results. The processing chains or the pipeline, apply default options in all SAS tasks, which eventually can be changed by the GO in order to improve the quality of the results. In particular, the source detection is very sensitive to the artifacts that are common in OM.

We outline here the checkings that a user should perform on OM processed data (by the standard Pipeline or by running SAS), and the use of one of the tasks, `omdetect`, where the user can modify parameters affecting the source detection and therefore the overall results of the data analysis.

1. Checking `omichain` output products:

    - The first thing to do is to overlay the image source list onto the sky image. This can be achieved with `implot`, which will overplot the detected sources on the corresponding image, and it will allow us to check that all sources are real and that the one(s) we are interested in have been properly identified. If the REGION files produced by `omdetec` have been preserved (in manual run of the chain or the task), then `ds9` can also be used to plot the detections on top of the image.

If the background is strongly affected by straylight features, this check is very important.

Alternatively, the task `omsource` can be used to do the checking. In this case, the obtained results can be modified by re-doing the photometry in an interactive way.

- Inspection of the combined source list (e.g. using `fv`) will allow us to check that the sources of interest have been picked up in all the filters where they are visible and that the combined list contains colours and standard magnitudes for them.
- Check the tracking corrections: although the pointing stability of XMM-Newton is very good, one can verify it by examining the corresponding tracking history PDF file.

2. Checking `omfchain` output products:

   - Inspection of the light curves:

     One can look at the PDF files containing the light curves to check that there is some signal detected and measured (both in the source and in the background).
   - Checking the presence of source(s) in the fast window pseudo-image.

     This can be done easily by displaying this image with SAOImage (`ds9`) or `fv`. Another possibility is to overlay the detected sources onto the pseudo-image using the task `implot` or `ds9`.

3. Checking `omgchain` output products:

   - Checking the detection of zero and first orders

     The correct correlation of the zero and first order of the spectra is essential. It can be verified by displaying the rotated image and overlaying on it the detections from the SPCREG file with `ds9`. The source list SPECLI file indicates also these correlations. If the users wants to analyse all detections, then REGION and SWSRLI files should be examined.
   - Position of zero order

     The position of the zero order (centroid) is the zero point of the wavelength scale. It should be verified, e.g. using `ds9` as pointed out before.
   - Identifying the spectra.

     The final spectra present in the SPECTR.FIT file can be identified as said before by overlaying the SPCREG file on the rotated image. In the future the astronomical coordinates of their corresponding sources will be computed when running `omgchain`.

4. Improving the source detection:

   - For image data, if the source of interest is close to straylight features or to other sources it may not be detected with the default settings of the `omdetect` task. We have to change its parameters: `nsigma` and `minsignificance` (see SAS documentation, or run `omdetect -h` for details)

```
omdetect set=your_image.fit outset=your_sourcelist.fit \
         regionfile=your_region.dat nsigma=p minsignificance=q
```

where `your_image.fit`, should have been produced by `ommodmap`. The default values for `nsigma` and `minsignificance` are 2 and 1, respectively.

Invoking `omdetect` with `regionfile=your_region.dat` will allow us a fast checking by overlaying the newly detected sources positions on the image with `ds9` using the created region file.

It may be easier to use the task `omsource` interactively to improve the photometry of any detected source, to identify new undetected sources and add them to the list, or even to repeat completely the detection and photometry in the whole image.

- In case of fast mode, if there are more than one sources in the fast window, then the detection will be affected and therefore the whole light curve too. We have to change some parameters in `omdetect` as in the case of image mode (see online SAS documentation, or run `omdetect -h` for details).

  Invoking `omdetect` with `regionfile=your_region.dat` will allow us a fast checking by overlaying the newly detected sources positions on the image with `ds9` using the created region file.

5. On grism data, as with normal image data, `omdetect` can be used to improve the detection. However, we recommend using `omgsource` to select the spectra interactively.

6.7.1 Astrometry in OM images

The task `omatt` takes an OM OSW source list, with source positions in pixels, and converts them into sky coordinates. In addition, the whole image is corrected for geometric detector distortion and rotated so as to become north aligned. The task requires the user to enter the name of an OSW source list and its corresponding OSW image. The positional accuracy obtained in this way is of the order of 2 arcsec, this error being due in part to the residuals of the geometric distortion correction.

In its second part, which can be run optionally by the user (it is not used by default in `omichain`), `omatt` employs the USNO SA1 catalogue (it has to be provided by the user). The OM pointing vector and field of view are used as parameters to a search of the USNO, which lists all catalog stars in the field of view. Two synthetic images are constructed from the source list and the catalogue stars. A two-dimensional cross-correlation is performed on the images, and this 2D cross-correlation function is searched for its maximum, which gives x and y offsets in pixels between the OSW source list and the catalogue. This offset is used to correct the catalog tangent plane coordinates. The nearest catalog star to each source in the source list is then found. If this distance is smaller than a tolerance times the positional error, then it is assumed to be that catalog star. If sufficient catalog stars are identified, then a least-squares fit is performed to the catalog positions, to yield more accurate astrometry. The source positions in RA/Dec are then written into the source list, along with the positional error (the pixel coordinates are retained in the source list).

6.7.2 Counts conversion to magnitudes and fluxes

The task `ommag` converts the count rates to magnitudes in the appropriate instrumental bandpasses. The rates are taken from a source-list produced by `omdetect` or from a timeseries produced by `evselect`. The output file will be a FITS file identical to the input source-list or time-series except that the count-rates have been converted into instrumental magnitudes (in the specified filter bandpass: U, B, V, UVW1, UVM2, UVW2) and then written as an extra column in the original FITS file. The program also calculates the limiting magnitude and writes the value in the FITS header.

Since SAS Version 5.4, flux conversion factors for U, B, V, UVW1, UVM2 and UVW2 filters are added in the keyword in the COLORMAG extension of the CCF file, OM_COLORTRANS. These flux conversion factors help users to get a rough direct estimation of flux (expressed in erg/cm^2/s/Å)from countrates.

For each filter, the flux (in erg/cm^2/s/Å) can be obtained multipling the countrates (counts/s) from SAS by the following values.

	V	B	U	UVW1	UVM2	UVW2
effective wavelength (Å)	5430	4500	3440	2910	2310	2120
conversion factors	2.49E-16	1.29E-16	1.94E-16	4.76E-16	2.20E-15	5.71E-15

These flux conversion values have been obtained from observations of white dwarfs standard stars. They reflect the current status of the OM in-flight calibration, and are therefore constantly verified and updated. The users should make sure that the most updated calibration is always employed.

We should also point out that these flux conversion numbers can only provide an approximated measurement of the flux without an *a priori* knowledge of the spectral type. For a more accurate determination of the flux, the users will find in our SAS watchout page updated values of the flux conversion factors for a given spectral type.

6.7.3 Barycentric correction for fast mode data

As it is stated in the header of the TIMESR.FIT files(keyword TIMEREF = LOCAL), the time values in the time series files produced by processing OM data in Fast Mode to the XMM-Newton satellite system.

This time scale can be converted into a solar system barycenter time system by applying the barycentric correction with the SAS task `barycen`:

```
barycen withtable=yes table=tserie.fit:RATE
```

where tserie.fit is the light curve file obtained from `omfchain`.

The `barycen` task overwrites the TIME column in the original file. Therefore it is recommended to copy it beforehand if one wants to keep the non-corrected timing.

It should be noted that for OM, `barycen` is applied to the time stamps of the light curve, and not to the arrival time of each individual photon. `barycen` will not work on OM Fast Mode events list files.

6.7.4 AB magnitude system for OM

The AB Magnitude system (Oke, 1974) has been implemented for OM. The general reprocessing pipeline products and SAS output files contain AB values for all detected sources. This is done by the `omsrclistcomb` and the results are written in the OBSMLI source file as AB magnitudes in all filters.

Conversion from AB magnitudes to flux units is also performed and the results are also written in the OBSMLI source file for all filters.

6.7.5 OM response matrices

The OM response matrices for all filters (UVW2, UVM2, UVW1, U, B, V) have been obtained by computing the effective area of the instrument from ground laboratory measurements of its various components (mirror reflectivity, filter transmission, photocathode quantum efficiency, microchannel plates and detector efficiency,...). In order to match the observed in-flight response, fudge factors were necessary. Those fudge factors were obtained by observing a series of spectrophotometric standard stars.

We have produced as well response matrices for the Visible and UV OM grisms. The OM grisms have an absolute flux calibration in the form of an Inverse Sensitivity Function (ISF). The ISF provides a direct conversion from count rate in the extracted spectrum to absolute flux. This ISF can be easily converted into effective area for both grisms.

The complete set, as well as a concise description and usage guidelines can be obtained from

http://xmm.esac.esa.int/external/xmm_sw_cal/calib/om_files.shtml

These response files will allow the users to make a combined spectral fitting of XMM-Newton Optical Monitor and X-ray data using packages such as XSPEC.

Bibliography and Links

[1] XMM−Newton Users Handbook
http://xmm.esac.esa.int/external/xmm_user_support/documentation/

[2] XMM−Newton Science Archive
http://xmm.esac.esa.int/xsa

[3] Calibration Access and Data Files Handbook
http://xmm.esac.esa.int/external/xmm_sw_cal/calib/

[4] EPIC pile-up: Technical Report XMM-SOC-CAL-TN-0036 (S. Molendi and S. Sembay, 2003)
http://xmm.esac.esa.int/external/xmm_sw_cal/calib/documentation.shtml#EPIC

[5] EPIC background: Technical Report XMM-SOC-CAL-TN-0016 (D. Lumb, 2002)
http://xmm.esac.esa.int/external/xmm_sw_cal/calib/epic_files.shtml

[6] The XMM-Newton View of the Crab, Kirsch et al. 2006
http://xmm.esac.esa.int/docs/documents/CAL-TN-0069-1-0.pdf

[7] EPIC Calibration Issues in Progress:
XMM-SOC-CAL-TN-0018 (Kirsch et al., regularly updated)
http://xmm.esac.esa.int/external/xmm_sw_cal/calib/documentation.shtml#EPIC

[8] XMM Data File Handbook
http://xmm.esac.esa.int/external/xmm_user_support/documentation/

[9] XMM CCF entry page
http://xmm.esac.esa.int/external/xmm_sw_cal/calib/ccf.shtml

[10] Calibration Status
http://xmm.esac.esa.int/external/xmm_sw_cal/calib/documentation.shtml

[11] Data analysis threads
http://xmm.esac.esa.int/sas/current/documentation/threads/

[12] RGS: System Offsets Using Diagnostic Images: XMM-SOC-CAL-TN-0046 (de Vries, 2003)
http://xmm.esac.esa.int/external/xmm_sw_cal/calib/documentation.shtml#RGS

[13] NASA/GSFC Introduction to XMM-NEWTON data analysis
http://heasarc.gsfc.nasa.gov/docs/xmm/abc/

[14] Pipeline product-generation procedure
http://xmmssc-www.star.le.ac.uk/newpages/pipe_top_ext.html

[15] fv : The Interactive FITS File Editor
http://xspec.gsfc.nasa.gov/docs/software/ftools/fv/

[16] ds9 : Astronomical Data Visualization Application
http://hea-www.harvard.edu/RD/ds9/

[17] XSPEC : An X-Ray Spectral Fitting Package
http://xspec.gsfc.nasa.gov/docs/xanadu/xspec/index.html

[18] XRONOS : A Timing Analysis Software Package
http://xspec.gsfc.nasa.gov/docs/xanadu/xronos/xronos.html

[19] XIMAGE : An X-ray Image Package
http://xspec.gsfc.nasa.gov/docs/xanadu/ximage/ximage.html

[20] SPEX : SPEctral X-ray and UV modeling, analysis and fitting
http://www.sron.nl/divisions/hea/spex/

[21] ISIS : Interactive Spectral Interpretation System
http://space.mit.edu/ASC/ISIS/

[22] CIAO: CHANDRA Interactive Analysis of Observations
http://asc.harvard.edu/ciao/

[23] SIMBAD reference astronimical database
http://simbad.u-strasbg.fr/sim-fid.pl

[24] NED : NASA/IPAC Extragalactic Database
http://nedwww.ipac.caltech.edu/

[25] Dennerl, K. et al., *Improving XMM-Newton EPIC-pn data at low energies: Method and application to the Vela SNR*,
http://xmm.esac.esa.int/docs/documents/CAL-TN-0070-0-0.pdf