Using the OM Fast mode SAS

Simon Rosen(MSSL, srr@mssl.ucl.ac.uk)
Vladimir Yershov (MSSL, vny@mssl.ucl.ac.uk)

October 30 2001

Abstract

This document describes, with examples, the step-by-step use of the OM Fast mode SAS processing software to extract light curves from Fast mode observations.

1 Introduction

The OM SAS currently comprises 2 processing chains (perl scripts), one to reduce imaging mode data, the other, the fast (timing) mode observations, together with their constituent tasks.

The tasks that make up these chains (or pipelines) can be run individually. However, a given task will usually need to operate on the output results of a preceding task, so in general it is necessary for files from a given OM exposure to pass through all the chain tasks in sequence. For example, it is not possible to run the flat-fielding algorithm on a raw ODF image file because at the flat-fielding stage, the file should contain extra or modified header keywords and a quality array which are inserted by a routine at an earlier stage in the imaging chain.

The following sections describe the files required and the tasks used to process fast mode data from raw ODF files to finished products.

2 Calibration set-up

Before processing OM data, it is necessary to create a calibration index file (e.g. ccf.cif file) using the SAS task, *cifbuild* - it is probably best to create it in the ODF directory (see contribution by *gvacanti_2*).

3 Flat-field

One file required by both the Imaging and Fast chains is the flat-field map. It was originally envisaged that flat-field images, generally obtained during spacecraft slews or periods when the OM is inactive, would be stored in an archive. This was intended, partly to monitor the evolution of the detector response and also to allow users to accumulate a statistically good flat-field from flats taken close in time to their observation. However, for a number of reasons this has so far proved unnecessary and a single OM flat-field is to be stored as a component of an OM Calibration file (OM_PIXTOPIXSENS). Due to the complexity of analysing the OM flat-field, at the current time the stored flat is a unit image (which is a reasonable approximation to 5%). This will be superceded by a 'real' flat once better understanding of the flat-field variations and the gross response are attained.

The following SAS command will create the flat-field file from the calibration data.

omflatgen **outset**=flat.fit

- At present a warning will be issued noting that a template (ie a unit flat-field) is being used (see above).
- CAL usage: Reads the flat-field image from OM_PIXTOPIXSENS_vvvv.CCF (vvvv being the version number of the file).

This flat-field file is created automatically within the chain *omfchain*.

4 Fast data types

Fast mode data is taken in a single format, arriving as an event list. In a given exposure up to two fast mode windows may be defined, each giving rise to a separate event list. One should be aware that fast mode data may be accompanied by image mode windows from the same exposure.

5 Fast chain capabilities

The fast mode chain, which becomes available from SAS V5.2.1, processes only event lists. It internally builds tracking shifted and flat-field corrected images of the fast mode windows. These are searched for sources and light curves for any detected objects are extracted from the event list files. If more than one source is detected within the window, the output data set will contain a light curve for each source.

6 Required ODF files

Before attempting to use the *omfchain* or individual OM fast mode tasks, first run the *odfingest* SAS task. This will create a *SUM.SAS file from *SUM.ASC file that comes with the ODF. It is better to keep this in the same directory as the ODF. With this, and the other files listed below, which should already be in the ODF, it should then be possible to process the OM fast mode event list files.

rrrr_iiiiiiiii_SCX00000ATS.FIT - spacecraft attitude history file

rrrr_iiiiiiiii_SCX00000TCS.FIT - spacecraft time correlation file

rrrr_iiiiiiiii_SCX00000SUM.SAS - modified ascii summary file

rrrr_iiiiiiiii_OMX00000NPH.FIT - non-period housekeeping file

rrrr_iiiiiiiii_OMX00000PEH.FIT - periodic housekeeping file

rrrr_iiiiiiiii_OMX00000FFX.FIT - flat-field from omflatgen (mv from flat.fit)

For each exposure to be processed we also need

rrrr_iiiiiiiii_OMSeeewwFAE.FIT - fast event list file(s)

rrrr_iiiiiiiii_OMSeeewwWDX.FIT - auxiliary priority window data file

rrrr_iiiiiiiii_OMSeeewwTHX.FIT ** - OM tracking history auxiliary file

where **rrrr** is the 4 digit XMM rev number, **iiiiiiiii** is the 10 digit observation id, **eee** is the exposure number (e.g. 006 etc) and **ww** is the window identifier (00 or 01).

** - if these are not present, no tracking corrections can be applied but this is often not critical since XMM's tracking appears to be good to around 1 arc-second. If THX files are absent, a dummy file needs to be created (see *omprep* usage below) - this is the default action of *omfchain* (section 9).

Often a raw image FIT-file, in which the OM fast mode window is embedded, is also present in the input data set, as well as a larger image corresponding to the OM imaging window for the same exposure but these files are not used for the fast mode chain:

rrrr_iiiiiiiii_OMSeeewwIMI.FIT

7 Approaches to OM fast data reduction

OM fast mode data can be reduced either by use of the 'pipeline' task, *omfchain*, which encompasses all the processing tasks in one 'chain' and can act on a directory of data in one go, or by application of the separate tasks that constitute *omfchain* to individual files.

Two points should be borne in mind when using the current version (1.20 or earlier) of *omfchain*.

1. It does not provide user control of parameters affecting tasks within it - they are fixed at the values used in the pipeline processing system (PPS) at Leicester.
2. Some of the intermediate files are modified by tasks making it difficult to run one specific task, if necessary intermediate files were not properly prepared by the previous stage task.

8 Using the omfchain

With the ODF data stored in a working directory, and with the SAS_ODF environment variable pointing to this directory, one can process data automatically, creating output files in the same directory, simply by typing

omfchain

from that directory. Alternatively, one should be able to pick up data from an inpdirectory (which overrides any setting of SAS_ODF) and output to an outdirectory by typing

omfchain **inpdirectory** =........ **outdirectory**=........

Where the .... are the directory paths - but note, please, this does not work properly in versions prior to, and including, SAS V5.2. It should also be noted that *omfchain* does not strictly adhere to the SAS standard for the use of SAS_ODF - this will be corrected in the future version.

Apart from the initial automatic creation of the flat-field image, the *omfchain* can be considered as being divided into 4 subsequent stages for processing of each window/event list (see figure 1).

1. tracking history information is turned into graphic products and a file used by the event-list analysis part of the chain.
2. A tracking-shift and flat-field corrected image is created from the event list
3. the image is source-searched and appropriate extraction regions defined.
4. background-subtracted light curves are derived (on a per-exposure basis) for each source from the event list and graphical products are generated.

These four processes run inside a loop over all exposures and all windows. In addition, the last step is executed for all detected sources within each window (see figure 1).

If the chain runs to completion, a number of important output files will exist (for naming conventions, refer to [1]). One can access varying levels of information at run-time by changing the SAS_VERBOSITY environment variable. At SAS_VERBOSITY=5, for example, *omdetect* will output messages explaining where it has got to. Higher levels of verbosity yield greater (sometimes rather opaque) details.

9 Using separate tasks

The following describes the separate use of each task in the *omfchain*. Note that most tasks can make use of more parameters than are shown but default values are generally satisfactory, or the resulting output files may be of interest but are not essential (see task descriptions for details).

In the following we illustrate the command line use of each task with the files from the data provided for the workshop. Where a given file name is used in the command line invocation of different tasks, this indicates that the same output file from an earlier routine is used by the later task (e.g., image.fit is created by *omfastflat* and is then supplied to both *omdetect* and *omatt*).

It will be seen that several tasks used in the fast mode chain are also employed in the OM imaging chain (see *srosen_1*) - they perform common functions, e.g. the tasks *omprep*, *omdrifthist* and *omthconv* perform tracking analysis which is applied to both observing modes. Some tasks are also repeatedly used within each pipeline. For example, inbuilt functionality in *omprep* allows it to extract tracking information in one mode of operation and insert header keywords into science files, in another. *evselect* is used in the fast mode pipeline to create a raw image and then later to extract filtered event lists, first for a source region and then for an associated background region.

The task sequence is as follows (note - the first 3 tasks form the tracking analysis part of the fast mode pipeline, tasks 4-7 add header information, and build the shift- and flat-field- corrected image of the window, tasks 8-9 perform the source searching, compute sky coordinates and produce a sky map, and the last tasks, 11-15, extract and build light curves corresponding to the detected sources).

1) *omprep* **set**=0202_0099010101_OMS00600THX.FIT
nphset=0202_0099010101_OMX00000NPH.FIT
pehset=0202_0099010101_OMX00000PEH.FIT
wdxset=0202_0099010101_OMS00600WDX.FIT
outset=tracking.fit

- In its first use, *omprep* derives a tracking file for later application. It is important to note that if there is no THX file for the given exposure, here *006*, you must make set=DUMMYTHX.FIT. *omprep* will then generate a dummy tracking file needed by the rest of the chain with zero drift in it.

2) *omdrifthist* **set**=tracking.fit bf plotfile=tracking.ps

- Extracts information about the tracking history during the exposure and constructs plots of the tracking vector and tracking statistics. When used in *omfchain* a PS to PDF conversion utility generates a `tracking.PDF file.`

Figure 1: OM Fast Mode Pipeline

3) *omthconv* **nphset**=0202_0099010101_OMX00000NPH.FIT **thxset**=tracking.fit **outset**=tracking_star_timeseries.FIT

- Generates time series for the 'tracking' stars in the field.

4) *omprep* **modeset**=1 **set**=0202_0099010101_OMS00600FAE.FIT **nphset**=0202_0099010101_OMX00000NPH.FIT **pehset**=0202_0099010101_OMX00000PEH.FIT **wdxset**=0202_0099010101_OMS00600WDX.FIT **outset**=event_list00600.fit

- Used again, here in the event-list preparation part of the pipeline, where it augments the header contents of the event-list file with information derived from summary and housekeeping files (e.g. incorporating filter, exposure, attitude and some window information). The modified event-list file is an intermediate file which becomes an input for the *evselect* task.

5) *evselect* **set**=event_list00600.fit **xcolumn**=RAWX **ycolumn**=RAWY **withimageset**=true **imageset**=raw_image.fit

- *evselect* extracts all the photon events from the event-list file and builds an image corresponding to the fast mode window. This image is not corrected for tracking history or for the flat field.

6) *omfastshift* **set**=event_list00600.fit **thxset**=tracking.fit **nphset**=0202_0099010101_OMX00000NPH.FIT

- *omfastshift* uses the tracking history file in order to correct the **X**- and **Y**- coordinates of the photon events for drifts of the spacecraft during the exposure. The input event-list table is expanded (CORR_X and CORR_Y columnd are added).

7) *omfastflat* **set**=event_list00600.fit **slewflatset**=0202_0099010101_OMX00000FFX.FIT **oswflatset**=flat_field.fit **fstimgset**=image.fit

- tracking shifts the flat-field (obtained from *omflatgen* and builds a flat-field corrected OSW image (image.fit file). The output flat_field.fit file corresponds to the given fast-mode OSW, i.e. it is the tracking shifted version of the flat-field. Note that *omfastflat* can not be run prior to *omfastshift* as the CORR_X and CORR_Y columns will not exist in the event table.

8) *omdetect* **set**=image00600.fit **outset**=oswsourcelist00600.fit **outputregionfile**=True **regionfile**=reg.dat **boxscale**=2

- Source detection and parameterisation algorithm. The main output file is the source list for the image (OSW). This contains filled values for the image coordinates of each source, source count rate, background rate, significance, source ellipse parameters and flags for sources that are potentially confused, extended or affected by bad pixels. Optional parameters allow output of an SAOimage-type region file. Other useful options include output of the background image and significance map. The **boxscale**=2 parameter is optimised for better detection of sources in the small fast mode OSW. Note that due to the small size of the fast mode OSWs, often only one source is present.
- CAL usage: Makes use of the PSF function stored in OM_PSF1DRB_vvvv.CCF, and the platescale available from OM_ASTROMET_vvvv.CCF.

9) *omatt* **set**=image00600.fit **sourcelistset**=oswsourcelist00600.fit **pposwset**=sky_image00600.fit **usecat**=F

- Performs astrometry conversions (pixel positions $\rightarrow \alpha$, δ, b_{II}, l_{II}) and produces a rotated skymap image. You will see a warning that the usecat option is not implemented - this is not relevant for the fast mode data.
- CAL usage: Uses distortion map parameters from OM_LINCOORDS_vvvv.CCF to linearise the pixel coordinates. Boresight details for the OM are derived from XMM_BORESIGHT_vvvv.CCF.

10) *omregion* **set**=oswsourcelist00600.fit **srcnumber**=1 **srcradius**=-12 **nfwhm**=3 **bkginner**=1.2 **bkgouter**=1.8 **bkgfile**=back_region00600.fit **srcfile**=source_region00600.fit

- Produces source and background extraction region files needed for *evselect*. A fixed radius (12) is recommended for the `srcradius` parameter in order to achieve uniformity of the output count rates for different exposures. The negative value of this parameter suppresses the automatic reduction of the source region when a neighbouring source is present within the 3-FWHM zone around the target (for the same reason).
- The background extraction region is an annulus with the inner radius of **bkginner** times the source extraction radius and the outer boundary is defined by **bkgouter** times the source extraction radius. Here 1.1 - 1.2 value is recommended for **bkginner** in order to minimise contamination of background light curve by the source photons.

11) and 12) *evselect* **table**=eventlist00600.fit **expression**="((WIN_FLAG .eq. 0) .and. (region(source_region00600.fit, CORR_X, CORR_Y)))" **xcolumn**=CORR_X **ycolumn**=CORR_Y **maketimecolumn**=T **withrateset**=T **timebinsize**=10 **rateset**=source_rates00600.fit

- Filters the photon event lists and produces count rate tables with the time binning interval corresponding to the **timebinsize** parameter. Here the time binning interval is set to 10 seconds, but any other reasonable values could be chosen for this parameter (>0.5s).
- The task *evselect* should be called twice in order to extract both source and background rates. In its second call the background_region00600.fit file should be given in the **expression** parameter argument. A different file name (say, back_rates00600.fit) should be used for the **rateset** parameter.

13) *omlcbuild* **srcregionset**=source_region00600.fit **bkgregionset**=background_region00600.fit **srcrateset**=source_rates00600.fit **bkgrateset**=back_rates00600.fit **sourcelistset**=oswsourcelist00600.fit **wdxset**=0202_0099010101_OMS00600WDX.FIT **outset**=lightcurve00600.fit

- Generates coincidence-loss and dead-time corrected source timeseries using source and background rate files. Makes necessary photometric corrections (dead-time, coincidence-loss), subtracts the background (appropriately scaled for area) from the source rates, and calculates the minimum, maximum and average source magnitudes for the given exposure (these data are added to the header of the output FITS-file). If the source is offset from the centre, some source photons will have arrived outside the window so a correction is included for the missing part of the PSF function.
- CAL usage: Exploits the PSF function stored in OM_PSF1DRB_vvvv.CCF, coincidence loss information in OM_PHOTTONAT_vvvv.CCF, and obtains instrumental magnitude zero points from OM_COLORTRANS_vvvv.CCF.

14) *lcplot* **set**=lightcurve00600.fit **binsize**=1 **plotfile**=lightcurve00600.ps

- Plots the lightcurve (for both the source and background) and runs Chi-square and Kholmogorov-Smirnov tests on the light curve to gauge possible source variability. The resulting postscript plot can be transformed in to a PDF-file by using the following command:
- *ps2pdf* lightcurve00600.ps ligthtcurve00600.pdf

10 Sanity checks

A few output files (P* files) from *omfchain* should be checked

A useful first step is to inspect the output PDF light curve plots to check that the data are sensible - watch out for frequent drop-outs, isolated high or substantial negative values. Establish that the background light curve is reasonable.

One should also examine the image of the fast mode window, for example, with the ftool, fv. Check whether the source is well centred or close to an edge. If the latter, inspect the light curves, especially from different exposures, to see if the separate exposure based light curves are plausibly consistent (bearing in mind any intrinsic source variability). If the image appears blurred or odd in any way, look at the tracking history file to see whether tracking was reliable. You may also want to overlay the image source list onto the skymap using *implot*, e.g.

implot **set**=sky_image00600.fit **withsrclistset**=y **srclistset**=oswsourcelist00600.fit **device**=yourdevice **itf**=1 **radius**=5 **maxsrc**=10 **colour**=7

While often there will be only one object in the image, one should check for spurious detections which could affect the analysis of the target source and also for missed sources or prominent objects in the window which may contaminate the target. At the current time, bright contaminating sources are not treated properly and, if present, affect the light curve of the target (and vice versa) - this will be rectified in a future version of the relevant SAS tasks. It is possible to change the detection efficiency via the **boxscale**, **smoothsize**, **nsigma**, and **contrast** parameters of *omdetect*.

References

[1] Julian Osborne. Data products specifications. Technical Report SSC-LUX-SP-0004, SSC, 2000.