eimsimprep

February 1, 2016

Abstract

This is part of the package **eimsim**, which is a collection of routines used to make simulated XMM-Newton x-ray images, to perform source detection on them, and to assess the results.

1 Which documents to read

At last count, there are 25 tasks within the package **eimsim**, each with its own documentation (well, they will have, eventually). However, you don't need to read 25 sets of documentation! I recommend that you read the documentation for only 4 of the tasks, in the following order:

1. **eimsim**
2. **eimsimprep** (ie, the present document)
3. **eimsimbatch**
4. **eimsimreduce**

The documentation for **eimsim** contains all generic information relating to the package as a whole, such as the change history.

2 Description

As described in the **eimsim** documentation, the functionality of package **eimsim** is divided into things that need to be done only once and things which must be done N times for N simulation runs. The 'done once' things can be further divided into those things which must be done before the simulation runs, and those things which must be done afterward. The 'before' things are gathered into the present task **eimsimprep**; the 'N simulation runs' things are to be found within **eimsimbatch**; and the 'afterward' things are performed within **eimsimreduce**.

The task **eimsimprep** does 6 things in sequence, which are described individually in the following subsections.

2.1 Detection-specific preparations

This function may be performed alone by calling the script with `entrystage` and `finalstage`='detprep'.

Task **eimsim** is designed to allow the user to supply their own source-detection script. (See the **eimsim** documentation for a description of the requirements of such a script.) As with other parts of the **eimsim** functionality, there may be source-detection functions or jobs which only need to be done once and others (eg to do with the actual detection) which need to be done anew during each simulation run. The 'once-only' jobs can be separated and placed into another script which is called at the present stage of **eimsimprep**. At present, the **eimsim** package contains two example det prep scripts: **eimsimdetprep1xmm** and **eimsimdetprep2xmm**, corresponding to the respective detection scripts and template sets.

A user-written det prep script must fulfil the following requirements:

- The name of the detection-preparation script should be supplied via parameter `detpreptask`.
- The detection-preparation script should accept command-line parameters
  - obsidroots
  - prdssubdir
  - simopsubdir
  - astest
  - refband
  - srcspecset

  The values passed to these parameters by **eimsimprep** are the same values supplied to **eimsimprep** itself via the parameters of the same names - **eimsimprep** just feeds them through.
- As described in the **eimsim** task documentation, the user should supply **eimsimprep** (and **eimsimbatch**) with a list of directory names via the parameter `obsidroots`. The **eimsim** tasks expect to find SSC product files in subdirectories whose name is supplied in parameter `prdssubdir` off each of these `obsidroots`, and write their output files to subdirectories whose name is supplied in parameter `simopsubdir` off the `obsidroots`. The detection-preparation script should do the same.

It is helpful to construct the script so that it has an optional entry point named 'cleanup'. The function of this portion of the det prep script should be just to delete all intermediate files. In other words, the det prep script should be so designed, that it deletes all its intermediate files if invoked as follows:

```
<det prep task> entrystage=cleanup
```

2.2 Make sky masks

This function may be performed alone by calling the script with `entrystage` and `finalstage`='makeskymasks'. The actual processing is performed by tasks **mosaicprep** and **padmask**. The basic idea of this function is to limit the area of sky which is to be simulated, just to save time and memory. Limits are imposed in two separate stages, as described below.

2.2.1 Sky masks for individual exposures

Creation of count-rate images from many superposed PSFs has been found to be the part of the simulations procedure which takes the longest time. It is therefore desirable not to attempt to add sources to the image unless they are likely to make a significant contribution to it - ie there is no point in simulating an image of a source at -30 dec when the 1/2-degree field of view is centred on +20 dec; one may as well save oneself the trouble. This source-screening is done by preparing a mask image and only processing those sources which lie within 'live' areas of the mask.

Each mask is made from the exposure map for that instrument and energy band. Exposure maps are required inputs to **eimsimprep** (see section 5) in order to delineate the edges of the CCDs and of the field of view. If the exposure value is non-zero at the location of a source, then this source should clearly be among those used to make the count-rate image. The initial step in making the mask is therefore to set all pixels for which the exposure is non-zero to TRUE. However, one is to consider that a source which lies only a little way off the edge of the exposure may still make some contribution to the count-rate image due to the wings of its PSF. The TRUE area of the mask is therefore extended some little distance into the zero-exposure region. The width of this padding is set via the parameter `padsizearcsec`.

The algorithm to add the pad is as follows. First, the initial mask (from the non-zero pixels of the exposure map) is made. Then all edge pixels are identified - ie, all those for which the central pixel of a 9x9 square is TRUE but at least one of the eight neighbours is not. Finally, for each of these edge pixels, a circular blob of `padsizearcsec` radius is added to the final mask.

NOTE! These masks are NOT the same as the detection masks made by such tasks as **emask** during source detection.

2.2.2 Sky coverage of the source list

One often wants to detect sources in parallel in several variously-aligned images, made perhaps using several EPIC cameras, or observed at different times. If the images are not spatially aligned, a mask created for one exposure map will not necessarily be adequate for another. However, one naturally wants each simulated image in the set to draw from the same population of source positions and fluxes. For this reason, a super-region is created which is at least as large as the intersection of all the masks created in the previous section. The most convenient way to do this is to calculate the direction and width of that cone which just encompasses the set of masks.

The algorithm first calculates the position and radius of the cone which is just large enough to include the pointing directions of all the exposures; then adds 15 arcmin to the cone radius to account for the approximate radius of the EPIC fields; and finally adds a final radius pad of size specified via parameter `padsizearcsec`.

The RA and dec of the cone central ray, and the cone half-width or angle, are written to the `SRCSPECS` extension of the source template set (parameter `srcspecset`) in the keywords `CONE_RA`, `CONE_DEC` (both in decimal degrees) and `CONE_RAD` (arcsec). The angular area in deg^2 of the cone is also given in the keyword `CONEAREA`. This last is done just for convenience - this keyword is not required in any later stage of **eimsim**.

It is proposed at some later stage to add other alternative shapes of boundary region, to cater for mosaics which are extended in one direction for example.

2.3 Make non-vignetted exposure maps

This function may be performed alone by calling the script with `entrystage` and `finalstage`='makenonvig'. It is performed using the task **eexpmap** with –withvignetting set to 'no'. As well as acting as general image templates, the exposure maps are also used as background templates (see next subsection). A great part of the background is non- or negligably-vignetted, hence the need for non-vignetted exposure maps.

Because this is a quite lengthy procedure, the function checks to see if all required non-vignetted maps are already present: only if this is not the case are the missing maps generated. If you want to force **eimsimprep** to make them again you will have to delete the files yourself beforehand.

2.4 Make maps of expected background counts

This function may be performed alone by calling the script with `entrystage` and `finalstage`='makebkgmap'. Processing is performed by the task **bkgtemplategen**. Note that background maps are expected to be supplied by the user if parameter `bkgstyle` is set to 'products'; In this case, of course, the present function is skipped.

A realistic image simulation includes background counts. The XMM EPIC background has not yet been fully characterised but it seems clear that it consists primarily of one component from cosmological x-ray background, which is vignetted to the same degree as x-rays from discrete sources, plus a second component which is a combination of instrumental and particle background and which exhibits, in comparison, only a small amount of spatial variation. It ought therefore to be possible to produce quite a good background model by adding some scaled combination of normal (ie, vignetted) and non-vignetted exposure maps. This is what is done by the present function of **eimsimprep**.

One background map is produced per band, per exposure, per observation. There are at present two ways of supplying the background rates, controlled by the parameter `bkgstyle`.

If `bkgstyle` is set to 'srcspecset', the *total* (ie, vignetted plus non-vignetted) background rates (in counts per second per square arcsec) are read for the appropriate band, instrument and filter from the source template set (name as supplied to parameter `srcspecset`). At present the fraction of these rates which is to be considered non-vignetted is hard-wired at 0.2. This will no doubt become more sophisticated in the future.

If `bkgstyle` is set to 'user' on the other hand, the respective amounts of vignetted vs. non-vignetted component are supplied via list parameters `bkgrates` and `vigfractions`. Each of these parameters should be supplied with a list with N elements, 1 per each of N energy bands listed in the `FLUX_SCALES` table of the source template set, in the same order as the band IDs occur in that table. The maximum total background rates (in counts arcsec^{-2} sec^{-1}) should be supplied via `bkgrates`; the fractions of these which are to be considered vignetted are supplied via `vigfractions`. The units of the output image are counts pixel^{-1}.

As mentioned above, if parameter `bkgstyle` is set to 'products', the user is expected to supply maps for all necessary instruments and bands. It is up to the user to make sure that such maps are consistent with the remainder of the product templates (exposure maps in particular). At present there is no parameter by which the user can point **eimsimprep** to these files; instead one just names them correctly and places them in the correct directory. (This rather rudimentary interface may change in future.) The easiest way to determine the correct file names for these is to run **eimsimprep** with `astest`='yes' and `bkgstyle`='user' or 'srcspecset'. In 'astest' mode, the command lines are not sent to the shell, just printed to standard output; the appropriate file names will be found in the invocations of task **bkgtemplategen**.

Note that the list of simulated sources should contain sources which are too faint to stand any significant chance of being detected - otherwise it would not be possible to assess the sensitivity of the detection scheme. These faint sources will therefore contribute to the vignetted component of the background. This is in accordance with the actual x-ray sky: on statistical and other grounds it is presently believed that a significant fraction of the cosmological x-ray background is comprised of unresolved point sources (ref????*****). It may therefore be desirable to adjust the faint-end cutoff in the list of simulated sources so as to generate a background which mimics as nearly as possible the (somewhat lumpy) statistical qualities of the real x-ray background. A disadvantage in doing this is that the spectrum of the cosmological background is known to be somewhat harder than that of populations of (so far) detectable sources (ref [**?**]). Representing it in part by sources and in part by a smooth vignetted map hopefully will allow us to overcome this difficulty.

**** Use Ueda et al, Ap J 518, 656-671 (1999) for the ref?

2.5 Make sensitivity and reciprocal-sensitivity maps

******* should be part of the user-supplied det-prep script??? But does not sensy need an estimate of the background?

This function may be performed alone by calling the script with `entrystage` and `finalstage`='sensmaps'.

Theoretically, given an exposure map, a background map and a knowledge of the source detection algorithm, one ought to be able to calculate the sensitivity limit of the source detection algorithm at each point in the sky, in terms of a map of the minimum flux of a point source which would be expected to be detected about half of the time. (Since this flux is obviously infinite for all points at which the exposure is zero, it is in practice often more convenient to work with the reciprocal map). Unfortunately, in both the 1XMM and the 2XMM detection algorithms, source candidates are accepted as valid as a function of not one but two independent, but in practice interfering, parameters (**eboxdetect**'s `LIKE` and **emldetect**'s `DETML`). The present author has shrunk from the task of disentangling this procedure. Since neither **esensmap** nor **esensitivity** provides a sensitivity recipe which is formally correct in respect to these two detection algorithms, we have at this time no way within the sas to make such maps.

However, useful as a sensitivity map may be, it is not essential in the context of simulations. At present, exposure maps are used as a stop-gap reciprocal sensitivity map. This results in incorrect non-zero values of the `INV_SENSY` column of the list of simulated sources. However, only the zero values are important for the simulation machinery, so it functions correctly nevertheless.

2.6 Add bits and bobs

This function may be performed alone by calling the script with `entrystage` and `finalstage`='addskyarea'. At present, all that is done is that the total sky area of non-zero exposure of the mosaic of images is calculated and stored as keyword `SKY_AREA` in the 'reciprocal sensitivity map'.

3 Parameters

This section documents the parameters recognized by this task (if any).

Parameter	Mand	Type	Default	Constraints

obsidroots	no	string	.	

A list of directory names. For each member of the list, the task constructs subdirectory names by appending '/' followed respectively by the strings in `prdssubdir` and `simopsubdir`. Please see the respective parameter descriptions for further information.

entrystage	no	string	makeskymasks	detprep- makeskymasks- makenonvig- makebkgmap- sensmaps-addskyarea- cleanup

This allows the user to enter the **eimsimprep** script at one of several places in its processing sequence.

finalstage	no	string	detprep	detprep- makeskymasks- makenonvig- makebkgmap- sensmaps-addskyarea- cleanup

This allows the user to exit the **eimsimprep** script at one of several places in its processing sequence.

refband	no	int	1	

Where only one file is required out of a set spanning several energy bands, for example an exposure map to be used to create a detection mask, the band used is specified by this parameter.

bkgstyle	no	string	srcspecset	user-products- srcspecset

Permits several ways to generate background maps.

bkgrates	no	real	5.0e-7	

This parameter is read if `bkgstyle`='user'. This is a list parameter, which must either have a single entry, or 1 entry for each energy band in the `FLUX_SCALES` table of the source template set `srcspecset`. The units are counts per image pixel per second.

vigfractions	no	real	0.2	

This parameter is read if `bkgstyle`='user'. It is a list parameter, which must have the same number of elements as `bkgrates`. The elements, which must be fractions between 0 and 1, represent the fraction of the background amplitude which is to be multiplied by the vignetting function for that band. This vignetted component represents that part of the instrumental background which comprises real x-rays. The non-vignetted remainder represents that part of the instrumental background which is caused by unvignetted cosmic rays and weakly vignetted soft protons.

prdssubdir	no	string	product	

For each member of the list `obsidroots`, the task constructs subdirectory names by appending '/' followed by the string in `prdssubdir`. The task expects to find input files in the `prdssubdir` subdirectory, namely a set of template files in the form of XMM products (see section **??** for a detailed description). Product templates from only 1 observation may be present in any one `prdssubdir` subdirectory.

simopsubdir	no	string	sim_output	

For each member of the list `obsidroots`, the task constructs subdirectory names by appending '/' followed by the string in `simopsubdir`. The task writes observation-specific outputs to this directory.

simgensubdir	no	string	sim_generic	

The task writes non-observation-specific output to this directory.

pseudoprodsubdir	no	string	pseudo_product	

This parameter has a similar function to `simopsubdir`, but is intended solely to contain the non-vignetted exposure maps which are created by the present task.

padsizearcsec	no	real	100	

See section 2.2 for a description of this parameter.

srcspecset	no	dataset	srcspec.fits	

This is the name of a FITS dataset which contains specification of the source probability distributions and also band-related specifications. See section 5 for a detailed description. Example files can be found in `$SAS_DIR/lib/data/eimsimdata/` .

dettaskstyle	no	string	auto	user—auto

The user is able to specify which source detection program or script should be used by the simulations. The user may either choose between alternatives provided in the package, or write their own source-detection script. Matters are somewhat complicated by the need also to provide a 'detection preparation' script to **eimsimprep**. The preparation and detection scripts must match. Some guidance for coordinating these choices at package level is given in the **eimsim** package documentation. If `dettaskstyle` is left at its default of 'auto', **eimsimprep** reads the detection style code from the parameter `dettype`. This code, which dictates which preparation script to run, is written to a file in the PWD called 'eimsim_config', for later reading by **eimsim**. If on the other hand `dettaskstyle` is set to 'user', **eimsimprep** reads the name of the preparation task directly from the parameter `detpreptask`.

dettype	no	string	2xmm	1xmm—2xmm

A code which indicates which source-detection scheme the simulation is to use.

detpreptask	no	string	eimsimdetprep2xmm	

This parameter is read if `dettaskstyle` is set to 'user'. It gives the name of the task or script which is to perform any necessary once-only preparation for the source detection task. The user may either choose between alternatives provided in the package, or write their own task. For users wishing to 'roll their own', a description of the 'handshaking' required of such a task is given in section 2.1.

astest	no	bool	no	

If 'yes', no tasks are called except the `detpreptask` script, which is also passed this parameter value to allow it to do the same.

4 Errors

This section documents warnings and errors generated by this task (if any). Note that warnings and errors can also be generated in the SAS infrastructure libraries, in which case they would not be documented here. Refer to the index of all errors and warnings available in the HTML version of the SAS documentation.

code *(error)*
message

code *(warning)*
message
corrective action: action

5 Input Files

1. A FITS dataset which acts as a template file to allow first the list of simulated sources and then the simulated EPIC images to be constructed. Its structure is as follows:

   - Binary table extension `SRCSPECS` containing:
     - real-valued keyword `HI_SLOPE`
     - 4-byte real column `FLUX` in erg cm^{-2} s^{-1}.
     - 4-byte real column `DENSITY` in cumulative sources deg^{-2}.
   - Binary table extension `INSTRUMENTS`, listing all instruments recognized as valid for **eimsim** purposes. The row numbers (the first row being numbered 1) correspond to the 'Potsdam' integer ID scheme appropriate to source lists created by **eboxdetect** or **emldetect**. Contents:
     - integer keyword `I_SUMMRY` giving the integer code, in the 'Potsdam' scheme, of the 'summary' instrument.
     - 10-character string column `NAME`, giving the standard 'XMM'-style name of the instrument. These strings are recognized by the **caloalutils** function instId().
     - 2-character string column `ABBREV`, giving an upper-case abbreviation of the instrument name.
   - Binary table extension `FILTERS`, listing all filters recognized as valid for **eimsim** purposes. Contents:
     - 10-character string column `NAME`.
   - Binary table extension `FLUX_SCALES`, listing all energy bands recognized as valid for **eimsim** purposes. Contents:
     - String keyword `DET_TYPE`
     - integer keyword `I_SUMMRY` giving the integer code, in the 'Potsdam' scheme, of the 'summary' band.
     - 8-byte real keyword `SPECINDX`
     - 8-byte real keyword `HI` in cm^{-2}.
     - String keyword `EUNIT` ('keV' is currently the only accepted value).
     - 8-byte real keyword `E_MIN` in keV.
     - 8-byte real keyword `E_MAX` in keV.
     - 8-byte real keyword `FLUX` in erg cm^{-2} s^{-1}.
     - 2-byte integer column `ID_BAND` (integer code of the band in the 'Potsdam' scheme).
     - 4-byte real column `E_LO`
     - 4-byte real column `E_HI`
     - 4-byte real column `FLUX`
     - For each instrument listed in the `INSTRUMENTS` table, and each filter listed in the `FILTERS` table, a 4-byte real column with a name after the pattern `ECF_<inst abbrev>_<filter>`
     - For each instrument listed in the `INSTRUMENTS` table, a 4-byte real column with a name after the pattern `BG_<inst abbrev>_THIN`. These are currently unused.

2. Various XMM product files (all these are FITS datasets too) which are again used mainly as templates for the construction of the simulated images. These must be found in a subdirectory whose name is constructed by appending '/' followed by `prdssubdir` to one of the base directories supplied in the string `obsidroots`. These files must have the standard XMM product file names. An example set is given in the 'cookbook' section near the start of the present document. Note that the 27-character-long filename prefixes contain information as follows:

- Characters 2 through 11 give the 10-digit observation ID.
- Characters 12 and 13 give the 2-character instrument ID.
- Characters 14 through 17 give the 4-character exposure ID.
- Characters 18 through 23 give the 6-character file type code.
- Character 24 is the energy band ID character.
- Characters 25 through 27 are generally 0 but can be used for source number or spectral order number for example.

These files must obey the following rules:

- Only 1 observation ID is permitted per subdirectory.
- Permitted instrument codes are either 'OB', for files which have significance for the whole observation, or from the list 'PN', 'M1' and 'M2', which represent the three XMM-Newton EPIC cameras. The two required 'OB' files have type codes 'CALIND' (the list of calibration constituents) and 'ATTTSR' (the attitude time series created by task **atthkgen**). At least one of the EPIC instruments must be present. Each instrument may be present with more than 1 exposure ID. For each combination of EPIC instrument and exposure ID, the following files are required:
  - An event list (file type 'PIEVLI' for 'PN' or 'MIEVLI' for the mos).
  - N exposure maps (file type 'EXPMAP'), one for each of N energy bands.

  N must be the same for all observations, instruments and exposures. For the present, the band ID characters (character 24 of the file name) must be digits from 1 to 9. Each digit which is represented by an 'EXPMAP' file name must also be found in the `FLUX_SCALES` table of the template file `srcspecset`.

3. If `bkgstyle`='products', the user should supply, for each observation, instrument, exposure and band, a real-valued image of the expectation value, in average counts per image pixel, of the instrumental background.

6 Output files

These files are all written to the appropriate `simopsubdir` subdirectory.

1. For each observation, instrument, exposure and band: a mask image dataset which specifies those parts of the sky on which simulated sources are to be placed.
2. If `bkgstyle` is set to any value but 'products': for each observation, instrument, exposure and band: a real-valued image of the expectation value, in average counts per image pixel, of the instrumental background.
3. A single map which is supposed to be a mosaic of all reciprocal-sensitivity maps relevant to the simulation in hand. In fact at present the exposure maps at the `refband` are used instead of reciprocal-sensitivity maps.

References