eimchip2sky

January 12, 2017

Abstract

The task takes a list of CHIPCOORD images and replots them onto a single image in sky coordinates.

1 Instruments/Modes

Instrument	Mode
EPIC mos or PN	Imaging

2 Use

pipeline processing	maybe
interactive analysis	yes

3 Description

In making a background map or exposure map in sky coordinates it is useful to be able to transform images in CCD coordinates into sky coordinates. This is the function of the present task. The task takes in a list of images in the CHIPCOORD coordinate system (which is described in the **cal** library documentation) and replots them to sky coordinates.

The template for the output sky coordinate image is supplied via the parameter `templateset`. This should contain WCS keywords to define the sky projection plane and the image pixel size; the template image array itself supplies the numbers of pixels in the image X and Y directions. The values contained in this array are ignored, however - either an event image or an exposure map can for example be supplied as the template image. The data type of the template image array is therefore irrelevant. Also, the template image need not be from the same instrument as the chip images, or even an XMM image at all. Note however that the output image contains a copy of all keywords in the template image dataset and primary array headers. It is up to the user to ensure that these are appropriate to the output image.

In order to perform the correct coordinate transform, the task needs to know, for each input CCD image, which CCD of which instrument the image corresponds to. **eimchip2sky** determines the instrument from the `INSTRUME` keyword in each chip image dataset header. (Note that the chip images may come from

more than one instrument.) There are several ways to specify the CCD numbers, including supplying an explicit list of them via the parameter `ccds`.

NOTE! At present the only WCS projection supported by **eimchip2sky** is RA/dec tangent.

3.1 Attitude variation

The task of transforming from CHIPCOORD to sky coordinates is complicated by the fact that the spacecraft attitude is not fixed but rather 'wanders' somewhat over the course of an exposure. The projection from CHIPCOORD to sky coordinates over a period of time can therefore only be estimated by a sequence of transform samples, weighted by the duration of validity of the sample. The user has the facility to supply a dataset table containing a list of spacecraft attitude samples plus the boundary times between them. (The format of this list is described more fully in section 6. See also the task documentation for **attbin** and the library **binned_att**.) Each chip image is then subjected to multiple transforms, one per attitude bin; the output image consists of a weighted sum of these transformed images. There are various ways in which the user can specify the weights.

3.2 `indices` parameter - what it's about

Suppose you have two complete sets of chip images which differ in some way not dependent on either space or time coordinates - for example, they may represent the same exposure but in two different energy bands. It would be simple enough to run **eimchip2sky** twice, one time per set. However this is essentially wasted work because the details of the coordinate transform are identical for the two sets. It is in order to allow the coordinate transform to be done just once for N such sets that the `indices` parameter has been introduced. An example is the best way to illustrate how it works in detail. Suppose there are two sets as above but (for brevity) each set contains just images from chips 1, 2 and 3 of a given instrument. Let the images at band 'a' be named im_1_a, im_2_a and im_3_a, and those at band 'b' be named im_1_b, im_2_b and im_3_b. The user should set parameters (among others as appropriate for other functions) as follows:

```
chipimgsets='im_1_a im_2_a im_3_a im_1_b im_2_b im_3_b'
indices='1 1 1 2 2 2'
outsets='out_a out_b'
```

In this case, the task performs 3 transforms instead of the 6 that would be necessary if the 'a' images were done separate to the 'b's: proper use of the indices as above allows it to group for example im_1_a and im_1_b together and apply the necessary transform just once.

The `indices` parameter thus serves to connect the input and output lists of images; ie it indicates, for each input image, which member of the list of output images it contributes to. Therefore `indices` must have the same number of elements as `chipimgsets`, but its range of values (in this case 1 and 2) must span the number of elements in `outsets`.

Note that, strictly speaking you should also set `withindices`='yes'. However, with the current parameter interface this is unnecessary.

What if you only have chip images from 1 band, and only 1 output image? In this case, do nothing: ignore `indices`, it will automatically have the correct default behaviour.

4 Parameters

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

Parameter	Mand	Type	Default	Constraints
templateset	yes	dataset		

The name of the input template image. This supplies information about the projection plane and pixel size (via WCS keywords) and pixel dimensions (via the array dimensions) of the output image.

chipimgsets	yes	dataset list		

A list of names of input images (in chip coordinates).

ccdstyle	no	string	implicit	user—kwd—implicit

Controls the way in which the task obtains the CCD number for each image in the `chipimgsets` list (in the standard XMM-Newton numbering scheme for the relevant instrument). If `ccdstyle`='user', the user is expected to submit these numbers explicitly via the parameter `ccds`. If `ccdstyle`='kwd', the task looks for a `CCDNR` keyword in the array header of each input image. If `ccdstyle`='implicit', the task assumes that the order of an image in the `chipimgsets` list is the same as the CCD number.

ccds	yes	integer list		$1 \leq$ccds≤ 12

This parameter is only mandatory if `ccdstyle`='user'. The parameter should contain a list of CCD numbers (in the standard XMM EPIC numbering scheme; see **cal** documentation for a description of this). The elements of this list should therefore correspond in number and order to those of `chipimgsets`.

outsets	no	dataset list	outset.ds	

The list of output images (in sky coordinates). See section 3.2 for a description of how they match up with the input images.

withindices	no	bool	no	

Whether to read the `indices` parameter.

indices	yes	integer list	1	

See section 3.2 for a description of how they work.

attstyle	no	string	template	binnedset—template—user

As described in section 3.1, in general the attitude (ie, pointing direction and roll angle) of the spacecraft doesn't stay fixed during the course of an exposure. This 'wandering' of the attitude can be approximated by sampling the attitude at a series of times during the exposure. A separate coordinate transform is computed for each attitude sample and the output image is build up of appropriately weighted contributions from each transform. The present parameter allows the user to choose whether to supply such a list of attitude samples (`attstyle`='binnedset') or to simply compute a single coordinate transform for the whole exposure, either at an attitude read from the _PNT keywords of the `templateset` (`attstyle`='template', the default) or at an attitude submitted by the user (`attstyle`='user').

binnedatttab	yes	table		

This parameter is read (and only mandatory) if `attstyle`='binnedset'. The parameter should give the name of a dataset plus table (eg: 'mySet.ds:ATT_TAB'). The table should contain samples of the XMM spacecraft attitude in time bins (see the task **binned_att** documentation for a description of the format).

weightstyle	no	string	gti	none—binnedset—gti

This parameter is read if `attstyle`='binnedset'. A series of attitude samples is supplied; it remains to decide how to calculate the weight for each sample. `weightstyle`='none' is only valid if there is only 1 attitude sample: the task will fail with an error otherwise. For `weightstyle`='binnedset', the task will seek for weights in a column named `WEIGHTS` in the `binnedatttab` table. If `weightstyle`='gti', the task attempts to obtain a list of Good Time Intervals (GTIs), the exact way these are supplied being dictated via the parameter `gtistyle`; these GTIs are ANDed with the attitude time bins; the net GTI in each bin provides the weight.

gtistyle	no	string	fromimages	user—kwd—fromimages—gtitablist

This parameter is read if `weightstyle`='gti'. In this case the task attempts to obtain a list of Good Time Intervals (GTIs); these GTIs are ANDed with the attitude time bins; the net GTI in each bin provides the weight. Note that matters are complicated by the fact that different CCDs usually have different sets of GTIs. Parameter `gtistyle` specifies the way in which GTIs are supplied. If `gtistyle`='user', the task obtains a single, simple GTI for all chip images from the parameters `tstart` and `tstop`. The same is true if `gtistyle`='kwd', except that the task expects to the start and stop times in keywords `TSTART` and `TSTOP`, (or whatever alternates are specified via parameters `tstartkwd` and `tstopkwd`), in the FITS table specified by the parameter `kwdtab`. Slightly more freedom is afforded by the alternate `gtistyle`='fromimages': in this case the task looks for a separate set of N GTIs for each input image, specified in a table in each image dataset. Only 1 common table name, specified in parameter `gtitabname`, is allowed however. The final alternate, `gtistyle`='gtitablist', affords the most general and flexible specification of GTIs. In this case the task looks for a list of GTI tables in the parameter `gtitablist`. Note that this parameter must have the same number of elements as `chipimgsets`.

tstart	yes	real		

This parameter is read if `gtistyle`='user'. Specifies the start of the single, universal GTI.

tstop	yes	real		

This parameter is read if `gtistyle`='user'. Specifies the end of the single, universal GTI.

kwdtab	yes	table		

This parameter is read if `gtistyle`='kwd'. It gives the FITS table description in which to find the GTI start and stop keywords.

tstartkwd	no	string	TSTART	

This parameter is read if `gtistyle`='kwd'. Name of the keyword in `kwdtab` which specifies the start of the single, universal GTI.

tstopkwd	no	string	TSTOP	

This parameter is read if `gtistyle`='kwd'. Name of the keyword in `kwdtab` which specifies the end of the single, universal GTI.

gtitabname	yes	string		

This parameter is read if `gtistyle`='fromimages'. Gives the common name of an extension containing GTIs in each input image.

gtitablist	yes	table		

This parameter is read if `gtistyle`='gtitablist'. A list of FITS table descriptions, 1 per input image, which should contain GTI lists.

ra	yes	real		

This parameter is read (and only mandatory) if `attstyle`='user'.

dec	yes	real		

This parameter is read (and only mandatory) if `attstyle`='user'.

apos	yes	real		

This parameter is read (and only mandatory) if `attstyle`='user'.

usefast	no	bool	yes	

Whether to use a fast, approximate or slow and accurate regridding algorithm.

conserveflux	no	bool	yes	

A CCD pixel covers, in general, a different amount of sky to a sky-image pixel. With some sorts of quantities (eg background rate per pixel) this change of area matters; for others (eg exposure time) it does not. In the former case you should set `conserveflux` to 'yes', in the latter, 'no'.

withnormalize	no	bool	no	

I am no longer sure what this is about. Best to leave it at default!

tempset	no	dataset	tempset.ds	

Name of a temporary dataset. If you are running more than 1 invocation of the task in the same working directory, you should name this uniquely for each invocation, to avoid crosstalk.

5 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.

badAttStyle *(error)*
The value of `attstyle` was not recognized.

badCcdStyle *(error)*
The value of `ccdstyle` was not recognized.

badGtiStyle *(error)*
The value of `gtistyle` was not recognized.

badNumCcds *(error)*
The number of elements in `ccds` must equal the number in `chipimgsets`.

badNumIndices *(error)*
The number of elements in `indices` must equal the number in `chipimgsets`.

badNumberGtiTabNames *(error)*
The number of elements in `gtitablist` must equal the number in `chipimgsets`.

badWeightStyle *(error)*
The value of `weightstyle` was not recognized.

duplicateCcds *(error)*
More than one CCD was found with the same combination of instrument, CCD number and `indices` value.

noWeightColumn *(error)*
You set `weightstyle` to 'binnedset', but no `WEIGHT` column was found in the binned attitude set.

tooManyAttBins *(error)*
You set `weightstyle` to 'none', but this can't work because there is more than one row in the binned-attitude file.

badIndices *(error)*
The `indices` do not obey the rules: the unique values in this list must enumerate the sequence of output images.

6 Input Files

1. A template image. Its array header must contain WCS keywords (see [**?**] for a recent description of these). The output image(s) will be the same size as the template image and projected onto the same celestial projection plane. Note that the data contained in the template image are not read by **eimchip2sky**.

   If `attstyle`='template', the template image array header must also contain `RA_PNT`, `DEC_PNT` and `PA_PNT` keywords.

2. A list of FITS images in the CHIPCOORD coordinate system. (See the documentation for the **cal** task for a description of the XMM coordinate systems.) These images are supplied via the `chipimgsets` parameter. Each image array must contain the following:

   - An `INSTRUME` keyword, which must be one of the three EPIC instruments. Note that the chip images don't have to come from the same instrument.
   - If `ccdstyle` is set to 'kwd', each chip image array header must also contain a `CCDNR` keyword.
   - The image array must be the correct size for CCDs of the given instrument. Task **eimchip2sky** obtains these from variables (those with names such as EMOS_MAX_X) in module caloaldefs.f90 in SAS package **caloalutils**.
   - The image array may be of any **dal** data type.

   If `attstyle`='binnedset', `weightstyle`='gti' and `gtistyle`='fromimages', each image dataset should also contain a table of GTIs of the name `gtitabname`.

3. If `attstyle`='binnedset': a dataset containing a table of samples of the spacecraft attitude. The table format is described in the documentation to sas library **binned_att**.

4. If `attstyle`='binnedset', `weightstyle`='gti' and `gtistyle`='kwd': A dataset+table which contains `tstartkwd` and `tstopkwd` keywords.

5. If `attstyle`='binnedset', `weightstyle`='gti' and `gtistyle`='gtitablist': a list of datasets, 1 per input image, each of which which contains a GTI table.

7 Output Files

1. Each output file contains an image array, which has the same pixel size and dimensions, and is projected onto the same plane, as the input template image. The data type of the output image is 4-byte real. The output image contains a copy of all the keywords in the template image.

   See section 3.2 for a description of how multiple output images match up with the input images.

References