Constellation School on X-rays from Star forming regions (May 18-22, 2009)

Chandra Analysis basic guide

edited by Marilena Caramazza

This handbook is the simplified version of the CIAO handbook that you can find on CIAO 4.1 Homepage . For any doubt refer to the complete guide or to the CIAO ahelp command that accesses the CIAO on-line documentation. In the following you will find a short description of your data products and the main steps of the data reduction.

Introduction to the Data Products

CIAO 4.1 Science Threads

ObsID Directory

The ObsID directory is the top level in the directory tree created when the tarfile is unpacked. We have already unpacked the data and you may find in your home a directory called data where the pipeline products are stored. It contains two files and two data directories:

```
unix% pwd
~/DATA/data_rho_ophiuchi

unix% ls -1
axaff01843N001_VV001_vv2.pdf
oif.fits
primary/
secondary/
```

Observation index file
`oif.fits`
The observation index file contains a summary of the data products associated with an observation. It is equivalent to a structured directory listing with descriptive information about the observation.

V&V report
`axaff01843N001_VV001_vv2.pdf`
All standard data products are checked by a CXC scientist before release to ensure data quality and to investigate the cause of any exposure losses or other anomalies. This process is known as "verification and validation", or V&V for short. The V&V report is distributed in PDF format with the data files.

This file contains a summary report of the V&V report. The full report is in a second PDF file (`axaff01843N001_VV001_vvref2.pdf`), which is packaged in the secondary data directory.

Users should review this information before beginning the analysis to ensure that there aren't any caveats from the V&V scientist.

In the following sections, we describe the contents of the primary and secondary data directories.

Primary Directory

The data products are arranged such that all products necessary for most analyses (e.g. the CIAO threads) are in the primary directory.

```
unix% cd primary/
unix% pwd
/intro_data/1843/primary

unix% ls -1
acisf01843_000N002_bpix1.fits
acisf01843_000N002_fov1.fits
acisf01843N002_1_sum2.html
acisf01843N002_2_sum2.html
acisf01843N002_3_sum2.html
acisf01843N002_cntr_img2.fits
acisf01843N002_cntr_img2.jpg
```

```
acisf01843N002_evt2.fits
acisf01843N002_full_img2.fits
acisf01843N002_full_img2.jpg
acisf01843N002_src2.fits
acisf01843N002_src_img2.jpg
orbitf084197100N001_eph1.fits
pcadf084271087N002_asol1.fits
[hrc_dtf1.fits]
[acis_plt2.jpg]
[acis_pha2.fits]
```

The file you will need to use in this directory are:

Level 2 event file

`evt2.fits`
The level 2 event file is the most important data product you receive. This file is created from the level 1 event list by filtering on the good time intervals (GTI) and status bits. The result is a list of events that is suitable for use in data analysis.

When new calibration is released, it is often necessary to remake the level 2 event file in order to apply the changes. This process is explained in the Create a New Level 2 Event File thread.

Bad pixels
`bpix1.fits`
A list of pixels identified as "bad"; criteria for flagging a pixel are listed in the Bad Pixels dictionary entry. Any tool that reads this file will exclude the bad pixels from its calculations.

The New ACIS Bad Pixel File: Identify ACIS Hot Pixels and Cosmic Ray Afterglows thread contains more information, including instructions on how to make a new ACIS bad pixel file, if necessary. The New Observation-Specific HRC Bad Pixel File thread explains when and how to make a new HRC bad pixel file.

Aspect solution
`pcad_asol1.fits`
The aspect solution describes the orientation of the telescope as a function of time. The detected position of an event and the corresponding telescope aspect are combined for an accurate determination of the celestial position of that event.

Since an aspect solution file is created for each stable aspect interval, there is often more than one `pcad_asol1.fits` file for an observation. All the files must be used whenever a tool requires the aspect solution as input.

Detailed information on the aspect solution files is available from the the Aspect Solution why topic.

Secondary Directory

If you are interested in reprocessing your data, you will also need the level 1 files, which are in the secondary directory.

```
unix% cd ../secondary/
unix% pwd
/intro_data/1843/secondary

unix% ls -1
acisf01843_000N002_aoff1.fits
acisf01843_000N002_evt1.fits
acisf01843_000N002_flt1.fits
[hrc_std_flt1.fits]
acisf01843_000N002_msk1.fits
acisf01843_000N002_mtl1.fits
acisf01843_000N002_soff1.fits
[hrc_std_dtfstat1.fits]
acisf01843_000N002_stat1.fits
acisf084271178N002_0_bias0.fits
acisf084271178N002_1_bias0.fits
acisf084271178N002_2_bias0.fits
acisf084271178N002_3_bias0.fits
acisf084271178N002_4_bias0.fits
acisf084271178N002_5_bias0.fits
acisf084272477N002_pbk0.fits
aspect/
ephem/
axaff01843N001_VV001_vvref2.pdf
```

The V&V report - `axaff01843N001_VV001_vvref2.pdf` - is explained in the ObsID Directory section.

Aspect offsets
`aoff1.fits`
A set of offsets versus time (e.g. `ra_off=(ra-ra_nom)`) . The one piece of remaining useful info is information about gaps in the aspect records. These are derived from the aspect solution files (`pcad_asol1.fits`, primary directory) for the observation.

Most users will never need this file in their analysis, instead using the aspect solution files directly.

Level 1 event file
`evt1.fits`
The level 1 event file contains *all* the events recorded for the observation. While many of these events have a status bit set to flags them as "bad", none of the information has been removed. This file is filtered on GTIs and status bits to create the level 2 event file.

The `evt1.fits` file is also the starting point for reprocessing your data, as explained in the Create a New Level 2 Event File thread.

Good time intervals
`flt1.fits` / `std_flt1.fits`
The GTI information for the observation, e.g. the start and stop times of all accepted time intervals over the observation. The major contributor to creating GTIs is information about when there is aspect data and when that aspect data is good. When the event file is filtered, the GTIs are stored as extensions of the data file, creating a dynamic record of the time filters applied to the data.

Note that this file is named slightly differently for ACIS (`flt1.fits`) and HRC (`std_flt1.fits`) observations.

Parameter block
`pbk0.fits`
The parameter block file is needed in conjunction with the bias maps when creating a new bad pixel list. It is used to determine observational parameters, such as which CCDs are active, the `READMODE` and `DATAMODE`, etc.

This file is only created for ACIS observations.

Data analysis

Create in your home directory a new directory called ANALYSIS and copy there the files you need.

Inizialize the CIAO environment sourcing the file bin/ciao.sh or bin/ciao.csh.

Setting the Observation-specific Bad Pixel Files

CIAO 4.1 Science Threads

Overview

Purpose:

To set the observation-specific bad pixel file for an analysis session. This ensures that the tools will locate the proper bad pixel file in `ardlib.par` when it is needed.

Related Links:

- Analysis Guide: ACIS Data Preparation
- Analysis Guide: HRC Data Preparation
- HRC calibration pages: information on HRC bad pixel maps. In general, there are very few bad pixels in the HRC, and those that do exist are mostly at the edges of the data.

Reminder: Reset ardlib Between Analysis Sessions

Use caution when analyzing more than one dataset; either "`punlearn`" or delete the `ardlib.par` file before you start processing another dataset.

If you find that tools/scripts complain about being unable to find the bad pixel file for an observation that is different from the one you are observing, chances are that you have not cleaned out your `ardlib.par` file.

Running acis_set_ardlib

```
unix% punlearn ardlib

unix% pwd
/data/ObsID1843

unix% acis_set_ardlib acis1843_new_bpix1.fits
Updated ardlib parameter file: /home/username/cxcds_param4/ardlib.par
  AXAF_ACIS0_BADPIX_FILE -> /data/ObsID1843/acis1843_new_bpix1.fits[BADPIX0]
  AXAF_ACIS1_BADPIX_FILE -> /data/ObsID1843/acis1843_new_bpix1.fits[BADPIX1]
  AXAF_ACIS2_BADPIX_FILE -> /data/ObsID1843/acis1843_new_bpix1.fits[BADPIX2]
```

```
  AXAF_ACIS3_BADPIX_FILE -> /data/ObsID1843/acis1843_new_bpix1.fits[BADPIX3]
  AXAF_ACIS4_BADPIX_FILE -> CALDB
  AXAF_ACIS5_BADPIX_FILE -> CALDB
  AXAF_ACIS6_BADPIX_FILE -> /data/ObsID1843/acis1843_new_bpix1.fits[BADPIX6]
  AXAF_ACIS7_BADPIX_FILE -> /data/ObsID1843/acis1843_new_bpix1.fits[BADPIX7]
  AXAF_ACIS8_BADPIX_FILE -> CALDB
  AXAF_ACIS9_BADPIX_FILE -> CALDB
```

It is recommended that the script be run with `absolutepath=yes` so that the correct bad pixel file is accessed, regardless of the working directory.

The content of the parameter file may be checked using plist acis_set_ardlib.

Filtering Data

CIAO 4.1 Science Threads

[S-Lang Syntax]

Overview

Last Update: 6 Feb 2009 - updated for CIAO 4.1: Python and S-Lang syntax included for ChIPS plotting

Synopsis:

The CIAO Data Model allows powerful filtering of datafiles beyond the standard onboard event filtering. A file may be filtered on any of its columns, e.g. energy, time, position. Any (or none) of the filtering actions described in this thread may need to be performed on your dataset.

Related Links:

- Analysis Guide: ACIS Data Preparation

File types needed: evt2

Restrict The Energy Range

For most ACIS analyses, you will want to include at most the 0.3 kev to 10.0 kev energy range, as explained in the Choosing an Energy Filter why topic.

```
unix% punlearn dmcopy
unix% dmcopy "acisf01843N002_evt2.fits[energy=300:10000]" acis_1843_evt2_0.3-10.fits
opt=all
```

This command creates a new event file that only includes the data within the specified energy range.

CCD Filter

```
unix% dmcopy "acisf01843N002_evt2.fits[ccd_id=0,1,2,3]" acis_1843_evt2_sel.fits
opt=all
```

Applying Time Filters

For simple time filters, e.g. selecting a particular interval, the basic DM syntax may be used:

```
unix% dmcopy "acisf01843N002_evt2.fits[time=60413209:60414209]"
acis_1843_evt2_1000s.fits opt=all
```

Using merge_all to Compute ACIS Exposure Maps and Fluxed Images

CIAO 4.1 Science Threads

Overview

Purpose:

To create a wide variety of exposure maps using the script `merge_all`.

Read this thread if:

Related Links:

- ACIS QE Degradation why topic: correcting for the change in low-energy ACIS QE associated with the deposition of one or more materials on the ACIS detectors or optical blocking filters.
- An Introduction to Exposure Maps (PS, 12pp): a general discussion of exposure maps.

Sample ObsIDs used: 1842 (ACIS-I, G21.5-09); 1843 (ACIS-I, G21.5-09)

File types needed: evt2; asol1

This thread uses the `merge_all` script; for information about the script, consult the help file ("`ahelp merge_all`"):

CIAO 4.1 and CALDB 4.1 require that ACIS event files have a `CTI_APP` header keyword to indicate whether the CTI correction has been applied. The older `CTI_CORR` keyword is no longer used.

To check for `CTI_APP`:

```
unix% dmkeypar input.fits CTI_APP echo+
# dmkeypar (CIAO 4.1): ERROR: Keyword 'CTI_APP' was not found in file 'input.fits'.
```

If `CTI_APP` is not found, follow the instructions in the ACIS CTI_APP Keyword Required section of the ACIS CTI Correction why topic to add the keyword before continuing.

This thread may produce incorrect results **without issuing an error** if the keyword is missing.

Create a Multiple-Chip, Single Observation Exposure Map and Fluxed Image

Sample ObsID used in this section: 1843, all chips:

```
unix% dmkeypar 1843_evt2.fits DETNAM echo+
ACIS-012367
```

Finally, we choose to make the map monoenergetic with an energy of 2 keV (see the Create a Single Chip ACIS Exposure Map Step-by-Step thread for details).

Set the necessary parameters and run the script:

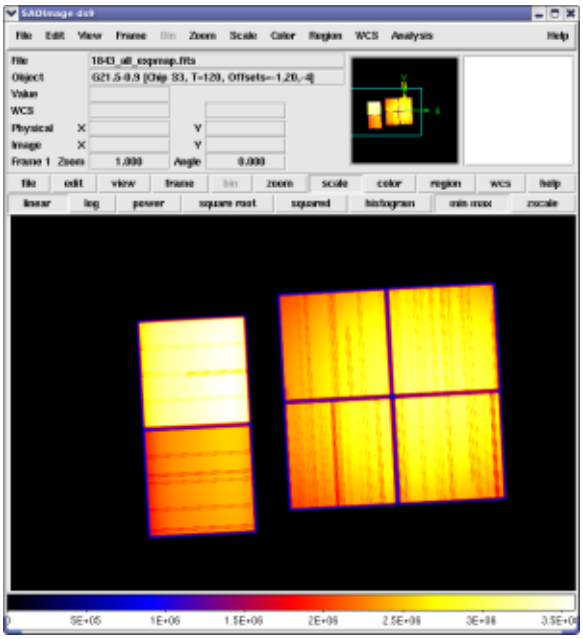
```
unix% punlearn mkinstmap
unix% pset mkinstmap pbkfile=secondary/acisf072038471N003_pbk0.fits
unix% punlearn merge_all
unix% pset merge_all evtfile=1843_evt2.fits
unix% pset merge_all asol=pcadf084271087N002_asol1.fits
unix% pset merge_all chip=0,1,2,3
unix% pset merge_all xygrid=0.5:8192.5:#2048,0.5:8192.5:#2048
unix% pset merge_all energy=2
unix% pset merge_all expmap=1843_expmap.fits
unix% pset merge_all expcorr=1843_expcorr.fits
unix% merge_all clobber=yes mode='h'
Input event file(s) (1843_evt2.fits):
Input asol file(s); time-sorted (pcadf084271087N002_asol1.fits):
Input Live Time Correction list files for HRC-I ():
ACIS CCD ID(s), comma-separated or range; or 'HRC-I' or 'HRC-S' (0,1,2,3,6,7):
Reference coordinates or evt2 file for reproject_events ():
XY grid (e.g., 0.5:8192.5:#1024,0.5:8192.5:#1024)
```

```
(0.5:8192.5:#1024,0.5:8192.5:#1024):
Energy or spectrum file for mkinstmap (1):
Output merged event file ():
Output merged exposure map (1843_expmap.fits):
Output exposure-corrected image (1843_expcorr.fits):

Creating aspect histogram for CCD 0...
Creating aspect histogram for CCD 1...
Creating aspect histogram for CCD 2...
Creating aspect histogram for CCD 3...


Making instrument map for CCD 0...

Making exposure map for CCD 0...
Exposure map limits: 0.000000e+00, 2.846048e+06
Making exposure map for CCD 1...
Exposure map limits: 0.000000e+00, 2.880390e+06
Making exposure map for CCD 2...
Exposure map limits: 0.000000e+00, 2.804751e+06
Making exposure map for CCD 3...
Exposure map limits: 0.000000e+00, 2.774244e+06

Merging exposure maps

Creating image

Normalizing image

All done!
```

In some cases, there will be more than one asol1.fits file for an observation. **All** the files must be input to the `asol` parameter, either as a list or as a stack.

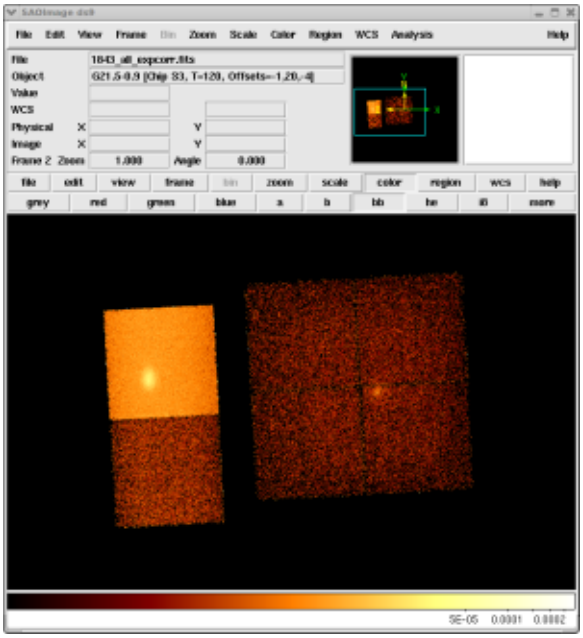
The output exposure map is shown in Figure 1 and the output fluxed image is shown in Figure 2.

Check the parameter file that was used with plist merge_all.



[Version: full-size]

Figure 3: Multiple chip, single observation exposure map



[Version: full-size]

Figure 4: Multiple chip, single observation fluxed image

Source Detection (PWDetect)

Note: this is not a CIAO routine, for details consult the pwdetect handbook

To perform source detection, copy the file ***pwdetect.par***, that you may find in ~/bin/., in the ANALYSIS directory and edit it to set the name of the event file and of the exposure map, and the desired threshold significance and detection scales. The threshold significance (sigthresh2) is set by the number of spurious sources one wished to accept and it depends on the number of background counts (for details consult the pwdetect handbook).

Then run:

```
unix% pwdetect_chandra
Thu May  7 08:45:08 2009

                    Welcome to PWDETECT V1.2
               Palermo Wavelet Detect for CHANDRA data


 Reading exposure map:  635_expmap.fits ...done

 initial threshold =  4.00 sigmas
 final threshold   =  4.80 sigmas

 Reading event list: 635_ccd0123_en05_8_evt2.fits
 Detector is ACIS-I
 Pointing RA  (degrees) = 246.824749  ( 16:27:17.9)
 Pointing Dec (degrees) = -24.573379  (-24:34:24.2)
 Roll angle   (degrees) =  78.299835
 Exposure time    (sec) =  100679.90
 ... successfully read 147591 events

 performing detection at  9 scales,
 from 0.50 to   8.00 arcsec, in logarithmic steps by a factor sqrt(2)

 computing background image... done
     54  sources at scale =  0.50 arcsec
     62  sources at scale =  0.71 arcsec
     75  sources at scale =  1.00 arcsec
    113  sources at scale =  1.41 arcsec
    119  sources at scale =  2.00 arcsec
    126  sources at scale =  2.83 arcsec
    125  sources at scale =  4.00 arcsec
     89  sources at scale =  5.66 arcsec
     75  sources at scale =  8.00 arcsec
 computing best source properties... done
 computing background image update... done
     54  sources at scale =  0.50 arcsec
     63  sources at scale =  0.71 arcsec
```

```
    75  sources at scale =  1.00 arcsec
   113  sources at scale =  1.41 arcsec
   120  sources at scale =  2.00 arcsec
   132  sources at scale =  2.83 arcsec
   126  sources at scale =  4.00 arcsec
    90  sources at scale =  5.66 arcsec
    75  sources at scale =  8.00 arcsec
computing best source properties... done

 Total number of detected sources = 148

 Nominal effective area (cm^2) =  445.720062

Thu May  7 08:46:57 2009
total elapsed time: 109 sec
```

This produces a FITS file called ***det_src.fits*** containing the detected sources, an ASCII version of it called ***det_src***, and a region file ***det_overlay.ds9*** that can be displayed in ds9 with the position of the detected sources (the circle radius corresponds to the scale at which the source was detected with maximum significance). For more details on the detection software and on the produced files, see the pwdetect_chandra_guide.pdf.

The region file det_overlay.ds9 does not contain the identification numbers of the sources. To create an overlay file with source numbers, run the script

```
add_src_num_chandra.com
```

which will create a file ***det_overlay_txt.ds9***.

Basic Lightcurves

CIAO 4.1 Science Threads

Overview

Synopsis:

A simple lightcurve from a point source in ACIS data can be used to get an idea of the variability of the source or to look for background flares that should be filtered out. The CIAO tool `dmextract` is used in this thread as it accurately applies good time interval (GTI) information when creating lightcurves.

Purpose:

To create lightcurves for use in a variety of analyses.

Related Links:

- Why topic: Timing Analysis with Lightcurves: caveats that one should be aware of when doing timing analysis on Chandra data.

Proceed to the HTML or hardcopy (PDF: A4 | letter) version of the thread.

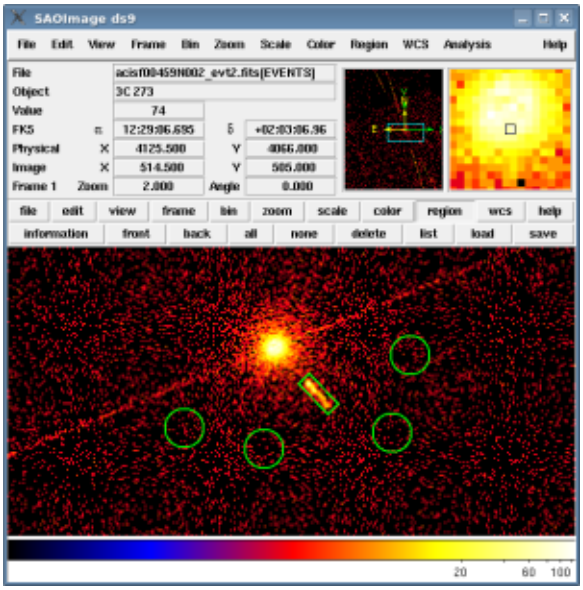
Build Source and Background Regions

We need to define two regions, one for the source and another for the background. To do this, first display the image:

```
unix% ds9 acisf00459N002_evt2.fits &
```

In this example, we define the jet as the source with a rectangle (see this FAQ on how to rotate shapes in ds9) and four 10-pixel radius circles for the background (from source-free parts of the image around the source). All the regions are shown in Figure 1. The background region can also be selected from a different chip or different event file, if desired.

The source and background region(s) must each be confined to a single chip. `psextract` cannot correctly handle the case where a region contains events from two different chips.



[Version: full-size]

Figure 1: Extraction regions on the event file

The background was chosen from a from source-free area of the same chip for this example, but it may also be chosen from a different chip or different event file.

To save the regions, follow these steps:

1. Region → File Format → CIAO
2. Region → File Coordinate System → Physical
3. Region → Save Regions... → Save As "3c273.reg" (source) and "3c273_bg.reg" (background). To select multiple regions for saving, hold down the <SHIFT> key and click on each one.

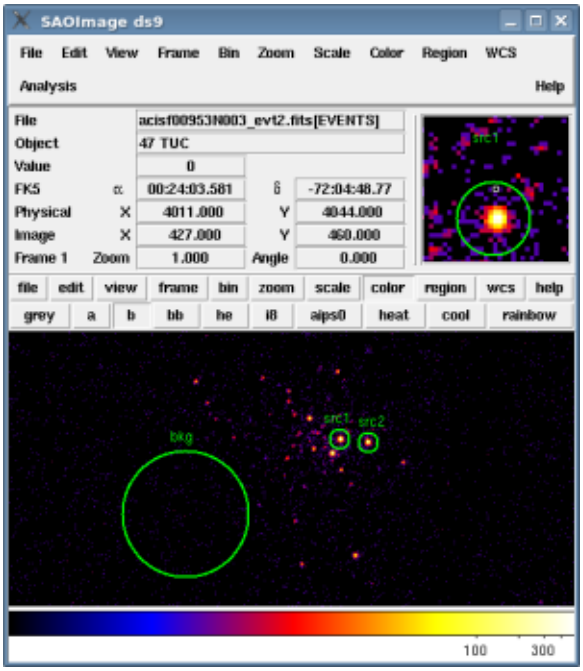
The resulting region files will look something like this:

```
unix% more 3c273.reg
# Region file format: CIAO version 1.0
rotbox(4148.125,4043.625,7.58978,22.338761,44.516094)

unix% more 3c273_bg.reg
# Region file format: CIAO version 1.0
circle(4119,4014.75,10)
circle(4077,4025.75,10)
circle(4186.5,4023.25,10)
circle(4196.25,4064.5,10)
```

ACIS Lightcurves

The most common lightcurve is made from a point source observed with the ACIS detector. This may be done to get an idea of the variability of the source or to help identify periods of high background.

To begin, we define the regions - two source and one background - which will be used to create the lightcurves. For instructions on how to create regions in ds9, see the Using CIAO Region Files thread. The regions used in this example are shown in Figure 1.



[Version: full-size]

Figure 1: Image of 47 Tuc with extraction regions

The source and background regions for creating the lightcurves.

```
unix% more src1.reg
# Region file format: CIAO version 1.0
circle(4010.5,4037.5,8)

unix% more src2.reg
# Region file format: CIAO version 1.0
circle(4035.5,4034.5,8)

unix% more bkg.reg
# Region file format: CIAO version 1.0
circle(3875.5,3972,54.5)
```

Determine which chips are being used

`dmextract` uses a `ccd_id` filter on the input file ensure that the proper GTIs are used. Use `dmstat` to determine the correct chip:

```
unix% punlearn dmstat
```

```
unix% dmstat "acisf00953N003_evt2.fits[sky=region(src1.reg)][cols ccd_id]"
ccd_id
    min:        3               @:        1
    max:        3               @:        1
   mean:        3
  sigma:        0
    sum:        7284
   good:        2428
   null:        0

unix% dmstat "acisf00953N003_evt2.fits[sky=region(src2.reg)][cols ccd_id]"
ccd_id
    min:        3               @:        1
    max:        3               @:        1
   mean:        3
  sigma:        0
    sum:        5850
   good:        1950
   null:        0

unix% dmstat "acisf00953N003_evt2.fits[sky=region(bkg.reg)][cols ccd_id]"
ccd_id
    min:        3               @:        1
    max:        3               @:        1
   mean:        3
  sigma:        0
    sum:        663
```

```
   good:         221
   null:         0
```
The regions with which we are working are all located on chip 3 (ACIS-I3); see Figure 6.1 of the POG for an illustration of the focal plane)

Create a background-subtracted lightcurve

First we extract a background-subtracted lightcurve for "src2":

```
unix% punlearn dmextract
unix% pset dmextract \
      infile="acisf00953N003_evt2.fits[ccd_id=3,sky=region(src2.reg)][bin
time=::2000]"
unix% pset dmextract outfile="src2_sub_lc.fits"
unix% pset dmextract bkg="acisf00953N003_evt2.fits[ccd_id=3,sky=region(bkg.reg)]"
unix% pset dmextract opt="ltc1"
unix% dmextract
Input event file  (acisf00953N003_evt2.fits[ccd_id=3,sky=region(src2.reg)][bin
time=::2000]):
Enter output file name (src2_sub_lc.fits):
```
You can check the parameter file that was used with plist dmextract.

The lightcurve may be plotted using the heasoft tool fv.

Using psextract to Extract ACIS Spectra and Response Files for Pointlike Sources

CIAO 4.1 Science Threads

Overview

Synopsis:

The `psextract` script allows you to compute different RMFs and ARFs for the source and background spectra, if needed. The output files are the source and (ungrouped) background spectra and the source RMF and ARF. If the source and background extraction regions have different average detector coordinates or the source and background event files are different, then the background RMF and ARF and a "linearly grouped" (by a factor of 20) background spectrum are also created.

Purpose:

To generate source and background PI (PHA) spectra of a pointlike ACIS source and build the proper RMFs and ARFs.

Related Links:

- Step-by-Step Guide to Creating ACIS Spectra: the manual version of this thread.

Proceed to the HTML or hardcopy (PDF: A4 | letter) version of the thread.

Run psextract

`psextract` runs the following tools in order:

1. `dmextract`: to extract source and background spectra
2. `dmkeypar`: to find out the start-time of the observation (for source and background)
3. `dmstat`: to get statistics on chipx, chipy, ccd_id, x and y (for source and background)
4. `acis_fef_lookup`: to find the proper FEF file (for source and background)
5. `mkrmf`: to build the proper RMF(s)
6. `asphist`: to create the aspect histogram(s)
7. `mkarf`: to create the proper ARF(s) In the case of grating data, `psextract` reads the `GRATING` keyword from the header of the event file and passes it to `mkarf`
8. `dmgroup`: to group the source spectrum and, if needed, the background spectrum
9. `dmhedit`: to update the BACKFILE, RESPFILE and ANCRFILE keys in the source PHA file, and the RESPFILE and ANCRFILE keys in the "linearly grouped" background PHA file (if needed).

In this case, since we are using an ACIS/HETG observation, `mkarf` will extract the 0th order spectra and ARF.

To extract the spectra in PI space and group the source spectrum to contain the new grouping-scheme; that is, there will be a minimum number of 15 counts per new channel:

```
unix% punlearn psextract
unix% pset psextract events="acisf00459N002_evt2.fits[sky=region(3c273.reg)]"
unix% pset psextract bgevents="acisf00459N002_evt2.fits[sky=region(3c273_bg.reg)]"
unix% pset psextract pbkfile=acisf063875928N002_pbk0.fits
unix% pset psextract dafile=CALDB
unix% pset psextract root=3c273
unix% pset psextract asol=pcad_asol1.fits
unix% pset psextract gtype=NUM_CTS
unix% pset psextract gspec=15
```

Running the tool with `verbose=2` shows what it is doing:

```
unix% psextract verbose=2
Source events specification (acisf00459N002_evt2.fits[sky=region(3c273.reg)]):
Background events specification
(acisf00459N002_evt2.fits[sky=region(3c273_bg.reg)]):
Root name for output files (3c273):
Source aspect solution file (pcad_asol1.fits):
Background aspect solution file ():
NONE, or name of the parameter block file (acisf063875928N002_pbk0.fits):

          events=acisf00459N002_evt2.fits[sky=region(3c273.reg)]
          bgevents=acisf00459N002_evt2.fits[sky=region(3c273_bg.reg)]
          root=3c273
          aoff=pcad_asol1.fits bgaoff=pcad_asol1.fits
          pbkfile=acisf063875928N002_pbk0.fits dafile=CALDB
          ptype=pi
          gtype=NUM_CTS grouping=15
          verbose=2

Extract source spectrum 3c273.pi from acisf00459N002_evt2.fits[sky=region(3c273.reg)]
dmextract infile="acisf00459N002_evt2.fits[sky=region(3c273.reg)][bin pi]"
outfile=3c273.pi
opt=pha1 clobber=no verbose=0

Extract background spectrum 3c273_bg.pi from
acisf00459N002_evt2.fits[sky=region(3c273_bg.reg)]
dmextract infile="acisf00459N002_evt2.fits[sky=region(3c273_bg.reg)][bin pi]"
outfile=3c273_bg.pi
opt=pha1 clobber=no verbose=0
Inspecting events file acisf00459N002_evt2.fits[sky=region(3c273.reg)]
Inspecting background events file acisf00459N002_evt2.fits[sky=region(3c273_bg.reg)]
Running dmstat on source and background events files:

Source Event statistics: CHIP 7 280.28940217 373.73097826   SKY 4146.0548958
4045.9470477
Background Event statistics: CHIP 7 290.40277778 379.9537037   SKY 4144.8603323
4032.6548281

Look up FEF spectral calibration file(s)
Source FEF file is /soft/ciao/CALDB/data/chandra/acis/fef_pha/acisD1999-09-
16fef_phaN0002.fits
[FUNCTION][ccd_id=7,chipx=257:288,chipy=353:384]
Background FEF file is /soft/ciao/CALDB/data/chandra/acis/fef_pha/acisD1999-09-
16fef_phaN0002.
fits[FUNCTION][ccd_id=7,chipx=289:320,chipy=353:384]
Verified Source FEF file is present
Verified background FEF file is present

Source and background FEFs are different
Building different RMFs and ARFs for source and background

Make source RMF file 3c273.rmf with mkrmf
mkrmf infile=/soft/ciao/CALDB/data/chandra/acis/fef_pha/acisD1999-09-
16fef_phaN0002.fits[FUNCTION]
[ccd_id=7,chipx=257:288,chipy=353:384] outfile=3c273.rmf logfile=./.psp.mlog
axis1=energy=0.1:11.0:0.
01 axis2=pi=1:1024:1 clobber=no verbose=2

Created source RMF file 3c273.rmf

Make background RMF file 3c273_bg.rmf with mkrmf
mkrmf infile=/soft/ciao/CALDB/data/chandra/acis/fef_pha/acisD1999-09-
16fef_phaN0002.fits[FUNCTION]
[ccd_id=7,chipx=289:320,chipy=353:384] outfile=3c273_bg.rmf logfile=./.psp.bgmlog
```

```
axis1=energy=0.1:11.
0:0.01 axis2=pi=1:1024:1 clobber=no verbose=2

Created background RMF file 3c273_bg.rmf

Running asphist to create source aspect histogram 3c273.asphist
asphist infile="pcad_asol1.fits" outfile=3c273.asphist
evtfile="acisf00459N002_evt2.fits[ccd_id=7]"
clobber=no dtffile="" verbose=0
Running mkarf to create source 3c273.arf at (4146.0548958,4045.9470477)
mkarf detsubsys=ACIS-S3 outfile=3c273.arf asphistfile="3c273.asphist"
sourcepixelx=4146.0548958 sourc
epixely=4045.9470477 grating=HETG obsfile="3c273.asphist"
pbkfile=acisf063875928N002_pbk0.fits
dafile=CALDB maskfile=NONE
verbose=0 engrid="grid(3c273.rmf[MATRIX][cols ENERG_LO,ENERG_HI])" clobber=no
Running mkarf to create background 3c273_bg.arf at (4144.8603323,4032.6548281)
mkarf detsubsys=ACIS-S3 outfile=3c273_bg.arf asphistfile="3c273.asphist"
sourcepixelx=4144.8603323 so
urcepixely=4032.6548281 grating=HETG obsfile="3c273.asphist"
pbkfile=acisf063875928N002_pbk0.fits
dafile=CALDB maskfile=NONE
verbose=0 engrid="grid(3c273_bg.rmf[MATRIX][cols ENERG_LO,ENERG_HI])" clobber=no

Group source spectrum using NUM_CTS 15
dmgroup infile=3c273.pi outfile=./3c273.tmp binspec= tabspec= grouptype=NUM_CTS
grouptypeval=15 ycolu
mn=counts xcolumn=channel tabcolumn= clobber=no verbose=0
Source spectrum 3c273.pi has been grouped

Group Background spectrum using BIN=1:1024:20
dmgroup infile=3c273_bg.pi outfile=3c273_bg_grp.pi binspec="1:1024:20" tabspec=
grouptype=BIN groupty
peval= ycolumn=counts xcolumn=channel tabcolumn= clobber=no verbose=0
Background spectrum 3c273_bg_grp.pi has been grouped

Updating PHA file 3c273.pi with BACKFILE RESPFILE ANCRFILE keys
Background spectrum (BACKFILE) = 3c273_bg.pi
SOURCE RMF (RESPFILE) = 3c273.rmf
SOURCE ARF (ANCRFILE) = 3c273.arf
Updating PHA file 3c273_bg_grp.pi with RESPFILE ANCRFILE key
BACKGROUND RMF (RESPFILE) = 3c273_bg.rmf
BACKGROUND ARF (ANCRFILE) = 3c273_bg.arf
```

Important: the output source spectrum (in this case, `3c273.pi`) is grouped if a grouping was specified; currently there **is not** an ungrouped spectrum created in this case.

Note: if you need to extract a spectrum in a given temporal interval set:

```
unix% pset psextract
events="acisf00459N002_evt2.fits[sky=region(3c273.reg)][time=6.21e+07:6.234e+07]"
```

The contents of the parameter file may be checked with plist psextract.

IMPORTANT:

The psextract script creates a wrong RMF and ARF files for the data. Therefore you MUST calculate again them as explained in the following threads.

Creating ACIS RMFs with mkacisrmf

CIAO 4.1 Science Threads

Overview

Synopsis:

The tool `mkacisrmf` separates the RMF calculation process into two components: an "ideal" component which describes the CCD spectral response prior to the effects of CTI, and a spatially varying component which incorporates the changes in the response produced by CTI.

Technical details on `mkacisrmf` are available from the Creating ACIS RMFs why topic.

Purpose:

To create an RMF for an ACIS imaging observation (or zeroth-order grating) with the newest calibration available. This thread also explains how **Related Links:**

- mkacisrmf bugs page: a list of all known issues with this tool.
- Why topic: Creating ACIS RMFs

Using mkacisrmf with the psextract script

The psextract script uses the mkrmf tool to create the RMF. Users who have the correct calibration applied to the data may run mkacisrmf independently to create a new RMF.

```
unix% dmkeypar acis_point_evt2.fits GAINFILE echo+
acisD2000-01-29gain_ctiN0005.fits
```

The file has an appropriate CTI-corrected gain applied to it, so we can use mkacisrmf. For the point source case, mkacisrmf is run in "Creating an RMF at a specific location" mode.

Choose the mkacisrmf calibration file (called a "P2_RESP" file). The P2_RESP and gain files are released in pairs of matching versions, so simply choose the file which has the same "N000x" version as the gain:

```
unix% ls -1 $CALDB/data/chandra/acis/p2_resp/
acisD1999-09-16p2_respN0005.fits
acisD2000-01-29p2_respN0005.fits
acisD2000-01-29p2_respN0006.fits
```

For this gain, acisD2000-01-29gain_cti**N0005**.fits, we should use the acisD2000-01-29p2_resp**N0005**.fits file as the infile parameter.

To get the necessary coordinate information (ccd_id, chipx, chipy), convert the source region information to chip coordinates with dmcoords:

```
unix% punlearn dmcoords
unix% pset dmcoords asolfile="pcadf084244404N002_asol1.fits"
unix% dmcoords acis_evt2.fits
dmcoords>: sky 4072.125 4245.625
(RA,Dec):      18:33:33.590    -10:34:08.00
(RA,Dec):       278.38996      -10.56889 deg
THETA,PHI           1.239'           6.09 deg
(Logical):        4072.12        4245.62
SKY(X,Y):         4072.12        4245.62
DETX,DETY         4246.75        4112.53
CHIP ACIS-S3       367.19         383.66
TDET              4284.19        2085.66

dmcoords>: quit
```

Now we have all the information to run the tool:

```
unix% mkacisrmf \
      infile="$CALDB/data/chandra/acis/p2_resp/acisD2000-01-29p2_respN0005.fits" \
      outfile=acis_point_rmf.fits \
      energy=0.3:10.0:0.005 \
      channel=1:1024:1 \
      chantype=PI \
      wmap=none \
      ccd_id=7 chipx=367.19 chipy=383.66 \
      gain=$CALDB/data/chandra/acis/det_gain/acisD2000-01-29gain_ctiN0005.fits
```

The new RMF file is named acis_point_rmf.fits.

Making a new ARF:

Read the "Matching the ARF and RMF energy grids" caveat before continuing.

psextract users who intend to finish the data analysis in XSpec should now remake the ARF file to ensure that the grid matches the RMF. To do so, get the mkarf command from the history in the ARF file. Change the engrid parameter to use the new RMF file, and rerun mkarf.

```
unix% dmhistory 3c273.arf tool=mkarf
mkarf asphistfile="3c273.asphist" outfile="3c273.arf" sourcepixelx="4145.9587485"
sourcepixely="4045.8953985"
engrid="grid(3c273.rmf[MATRIX][cols ENERG_LO,ENERG_HI])"
obsfile="acis_dstrk_evt2.fits"
pbkfile="acisf063875928N002_pbk0.fits" dafile="CALDB" mirror="HRMA" detsubsys="ACIS-
S3"
grating="HETG" maskfile="NONE" ardlibparfile="ardlib.par" geompar="geom" verbose="0"
clobber="no"

unix% dmhistory 3c273.arf tool=mkarf action=pset
```

```
unix% pset mkarf outfile="3c273_new.arf"
unix% pset mkarf engrid="grid(acis_point_rmf.fits[MATRIX][cols ENERG_LO,ENERG_HI])"

unix% mkarf
```
Be sure to update the spectrum file header to contain the name of the new RMF (`RESPFILE` keyword) and ARF (`ANCRFILE` keyword).

Update the Spectrum File Header

Update the `RESPFILE` keyword in the header of the source (and background) spectrum file with the name of the new RMF. For example:

```
unix% dmhedit infile=acis_src1.pi filelist="" operation=add \
      key=RESPFILE value=acis_new_rmf.fits
```
If the spectrum was created with one of the CIAO scripts (e.g. `specextract` or `acisspec`), the header keyword will contain the name of the file created with `mkrmf`. The wrong file could be selected during fitting if the keyword is not updated.

Also Update the `ANCRFILE` keyword in the header of the source (and background) spectrum file with the name of the new ARF. For example:

```
unix% dmhedit infile=acis_src1.pi filelist="" operation=add \
      key=ANCRFILE value=acis_new_arf.fits
```
If the spectrum was created with one of the CIAO scripts (e.g. `specextract` or `acisspec`), the header keyword will contain the name of the file created with `mkrmf`. The wrong file could be selected during fitting if the keyword is not updated.