Extracting Source and Background Spectra for a Point−Like Source, and Buildind RMF and ARF

1) Selecting and Saving Source and Background Extraction Regions: ds9

2) Extracting Source and Background Spectra (Counts vs PI Histogram): dmextract

3) Inspecting Events File (TSTART, chipx, chipy, ccd_id, x, y): dmlist, dmstat

4) Looking for the proper FEF file: acis_fef_lookup

5) Making the Source RMF: mkrmf

6) Applying SIM Correction and Building Aspect Histogram: asp_apply_sim, asphist

7) Making the Source ARF: mkarf

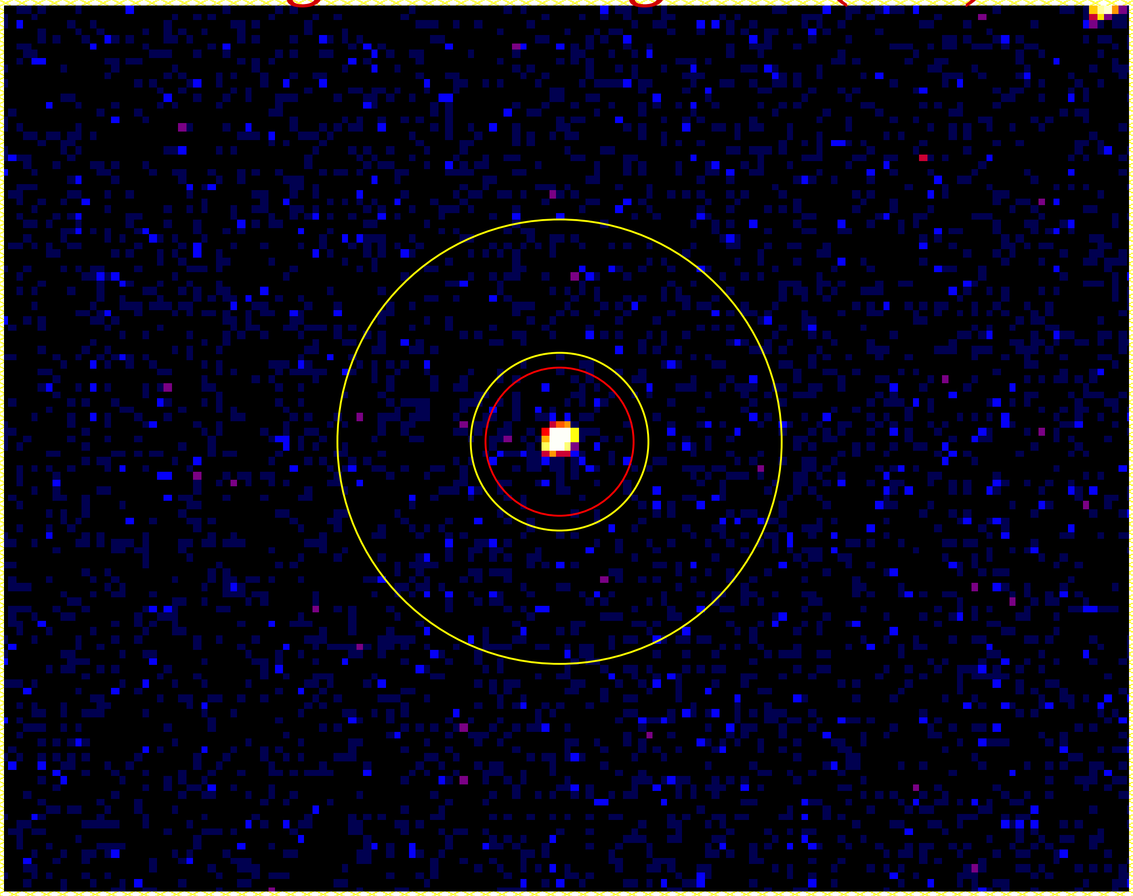
8) Grouping the Output (source) Spectrum: dmgroup

9) Editing Header Keywords BACKFILE, RESPFILE and ANCRFILE: dmhedit

10) The PSEXTRACT script

Fitting the Data in Sherpa

Selecting and Saving Source and Background Regions (ds9)



Source Region: 5" Radius Circle

```
# Region file format: CIAO version 1.0
circle(3874,4157,10)
```

Background Region: 6" and 15" Radii Annulus:

```
# Region file format: CIAO version 1.0
annulus(3874,4157,12,30)
```

The PSEXTRACT script

```
Parameters for /home/nicastro/cxcds_param/psextract.par

        events = evt2.fits[sky=region(sou.reg)] Source events specification
      bgevents = evt2.fits[sky=region(bgd.reg)] Background events specification
          root = test              Root name for output files
          aoff = aoff1.fits        Source aspect offsets file
        bgaoff =                   Background aspect offsets file
        (ptype = pi)               pha or pi
        (gtype = NUM_CTS)          Spectrum grouping type (NONE, BIN, SNR, NUM_BINS, NUM_CTS,
 or ADAPTIVE)
        (gspec = 15)               Spectrum grouping specs (NONE,1:1024:10,etc)
      (verbose = 5)                Debug Level(0-5)
         (mode = ql)
```

events, bgevents:	Virtual Source and Background Events Files, to extract the Spectra From.
aoff, bgaoff:	Source and Background (if Different from Source) Aspect Offsets Files (either SIM−Corrected or Not)
ptype:	PHA or PI Histogram
gtype:	Spectrum Grouping Type, and Grouping Specification. The Grouping type can be (a) NONE: no grouping performed, (b) BIN: the group is performed according to the specification in gspec (e.g. 1:1024:10), (c) SNR: channels are grouped to have a minimum SNR of gspec=N , (d) NUM_CTS: channels are grouped allowing a minimum of gspec=N counts per grouped bin, and (e) ADAPTIVE: channels are grouped adaptively starting with those containing, in the ungrouped scheme, at least gspec=N. Then the remaining channels are grouped into bins of 2 contiguos rows that satisfy the condition NUM_CTS > N, etc. (This can take very long!)

PSEXTRACT: step by step (1)

I) Extract Source and Background Spectra using dmextract:

```
dmextract events[bin ptype] outfile=root.ptype opt=pha1 clobb
er=yes verbose=(verbose-2|0 if verbose<2)

dmextract bgevents[bin ptype] outfile=root_bg.ptype opt=pha1
clobber=yes verbose=(verbose-2|0 if verbose<2)
```

II) Collect Information from Data: dmlist, dmstat:

(i) TSTART, GRATING Header keywords for Source and BGD. e.g.:

```
unix% dmlist events header,clean | grep TSTART |awk '{print $2}'
52377343.2940040
```

(ii) Collect Statistical Information on "mean" X and Y Chip Coordinates, .
X and Y coordinates and CCD−ID for source and Background Regions.

```
unix% dmstat "events[cols chipx, chipy, ccd_id, x, y]" |grep mean:
        mean:   ( 504.566868 548.754901 )       ->       ( 12.
099621 13.160311 )
        mean:   7.000000
        mean:   ( 4071.648088 4228.618097 )     ->       ( 212
.803476 52.235156 )
```

PSEXTRACT: step by step (2)

Look for the FEFs Corresponding to the Centroids of the source and background extraction Regions based on the info on TSTART, CHIPX, CHIPY and CCD_ID: acis_fef_lookup

```
unix% acis_fef_lookup ccdid chipx chipy ptype tstart
```

FEF (Fits Embedded Functions):

contain the functions used to build the Redistribution Matrices (RMFs) in their standard format. These files contain the Positions, the Widths and the Amplitudes of the gasussians used to redistribute photons according to the energy resolution of the detector.
These functions depend on the Chip used and on the particular position of the source (in chip coordinates) on this chip.

If the Background Extraction Region is Symmetric with Respect to the Source Centroid, then the script will find that source and background FEFs are the same, and will build common RMF and ARF.
If not, different RMFs and ARFs for Source and Background Spectra will Be Built.

PSEXTRACT: step by step (3)

Run MKRMF to build Source and Background Redistribution Matrices [BGD RMF is built only if SOU and BGD FEFs are different]:

```
unix% mkrmf infile=fef_file outfile=root.rmf \
        axis1=0.1:11.0:0.01 axis2=pibin clobber=yes \
        verbose=verbose
```

Where:

axis1	is the Energy axis of the output RMF MATRIX block, in keV.
axis2	is the channel axis in PI or PHA space (the value chosen for the parameter ptype of psextract). The defaults (assumed by psextract) are: 1:1024:1 in PI space, and 1:4096:2 in PHA space. These result in blocks EBOUND of the output RMF with 1024 or 2048 channels respectively.

PSEXTRACT: step by step (4)

Build Aspect Histograms for Source and Background (if the parameters "aoff" and "bgaoff" are different).

The scripts checks in the header of the "aoff" (and "bgaoff", if any) file whether the SIM−correction has already been applied to the input file, or not.

*_aoff1.fits files associated to an observation contains the aspect solution for both the optical axis offsets and the SIM (Science Instrument Module) offsets. If this file is input, the tool asp_apply_sim is run, and the SIM offset is added to the optical axis offset. The output is a file containing the "total" aspect offsets.

The output of asp_apply_sim is used as input for the tool asphist to calculate the aspect histogram (duration vs pointing offsets: X, Y and ROLL).

If an already SIM−corrected aspect offsets file is input as value of "aoff" (o0r "bgaoff"), then the script runs directly the tool asphist, to build the aspect histogram.

```
unix% asphist infile=sim_corr_file outfile=root.asphist \
gtifile=events clobber=yes dtffile=""
```

PSEXTRACT: step by step (5)

Run MKARF to build the source (and background) ARFs

```
unix% mkarf detsubsys=ccdid outfile=root.arf asphistfile=\
root.asphist sourcepixelx=x sourcepixely=y grating=grating \
obsfile=root.asphist maskfile=NONE verbose=verb \
engrid=axis1 clobber=yes
```

Where:

ccdid (from dmstat) is the combination of the detector name and the chip number: e.g. ACIS−S3

sourcepixelx and sourcepixely (from dmstat) are the X and Y centroid "physical (i.e. SKY) coordinates.

grating (from dmlist) can be NONE, or a grating configuration (for 0th order spectra), e.g. HETG

engrid (from the parameter axis1 of mkrmf) is the energy axis of the ARF matrix: 0.1:11.0:0.01. This can also be: engrid="grid(root.rmf[MATRIX][cols ENERG_LO ENERG_HI])"

PSEXTRACT: step by step (6)

I) Run DMGROUP on source and background (if different RMFs and ARFs have been created) spectra, if "gtype"!=NONE.

If Background ARF and RMF have been created, then the background spectrum is also grouped according to the scheme: 1:1024:20 (PI) or 1:4096:40 (PHA).

II) Run DMHEDIT to update the header keys of the output source (and background) spectrum:

BACKFILE=	Non−Grouped back_file
RESPFILE=	source RMF filename
ANCRFILE=	source ARF filename

If background ARF and RMF have been created, then also the header of the grouped back file is updated:

RESPFILE=	Back RMF filename
ANCRFILE=	Back ARF filename

Fitting the Spectra in Sherpa

I) Fitting the back−subtracted Source Spectrum:

```
sherpa> data root.pi
The inferred file type is PHA.  If this is not what you want,
 please
specify the type explicitly in the data command.
Background data are being input from:
root_bg.pi
RMF is being input from:
root.rmf
ARF is being input from:
test.arf
sherpa> subtract
sherpa> xswabs[abs](0.1)
sherpa> xspowerlaw[pl](2,0.01:1.e-10:1.e10)
sherpa> source = abs * pl
sherpa> fit
```

II) Fitting simultaneously source and background spectra:

```
sherpa> data 1 sou2.pi
The inferred file type is PHA.  If this is not what you want,
 please
specify the type explicitly in the data command.
Background data are being input from:
sou2_bg.pi
RMF is being input from:
sou2.rmf
ARF is being input from:
sou2.arf
sherpa> data 2 sou2_bg_grp.pi
The inferred file type is PHA.  If this is not what you want,
 please
specify the type explicitly in the data command.
RMF is being input from:
sou2_bg.rmf
ARF is being input from:
sou2_bg.arf
sherpa> xswabs[abs](0.1)
sherpa> xswabs[absbg](0.01)
sherpa> absbg.1.type = free
sherpa> xspowerlaw[pl](2,1.e-2:1.e-10:1.e10)
sherpa> xspowerlaw[plbg](1.4,1.e-4:1.e-10:1.e10)
sherpa> source 1 = abs * pl + absbg * plbg
sherpa> source 2 = absbg * plbg
sherpa> fit
```