Constructing Radial Profiles in CIAO

The CIAO tool DMEXTRACT now has the ability to compute background-subtracted, exposure-corrected net counts in multiple regions in a single pass. It can thus be used to construct radial profiles. The sequence of operations is:

1. define multiple concentric annular regions using DS9,
2. generate exposure maps if necessary,
3. run DMEXTRACT, inputting the annular regions **as a stack,**
4. run DMTCALC on the output FITS binary table to normalize by region areas,
5. use SHERPA tto display the profile and fit models to it.

CIAO Workshop, Radial Profiles

For this talk, I'll consider two cases:

- A simple case where the annuli fall within a single chip; I'll assume the exposure is uniform over the regions.
- A more complicated case where the regions span multiple chips.

For refererence, see the threads

http://asc.harvard.edu/ciao2.1/threads/getcnts.thread.html#RadProj

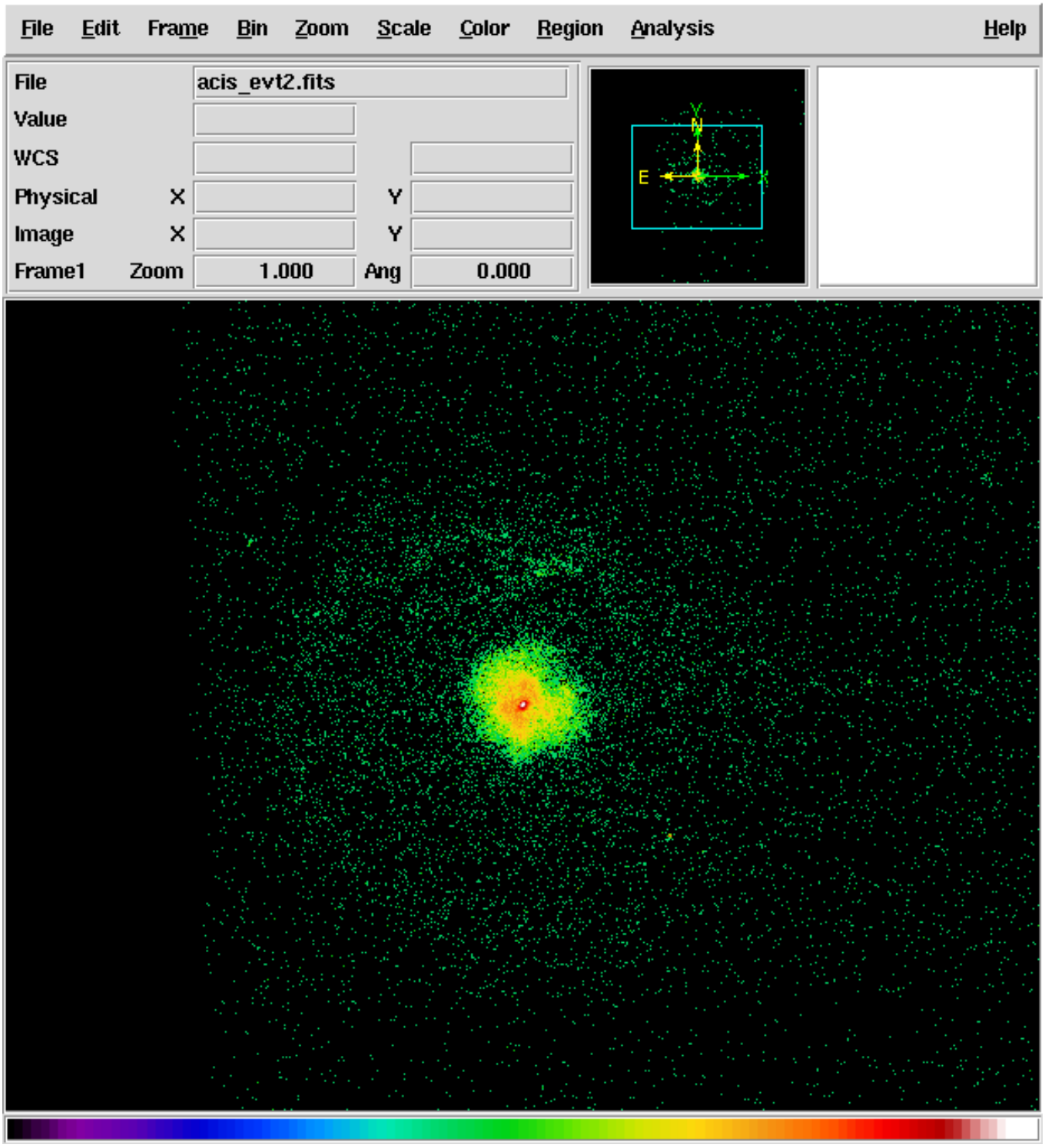
and

http://asc.harvard.edu/ciao2.1/threads/expmap.multi.html

I'll use the same data and in gereal follow those procedures closely.

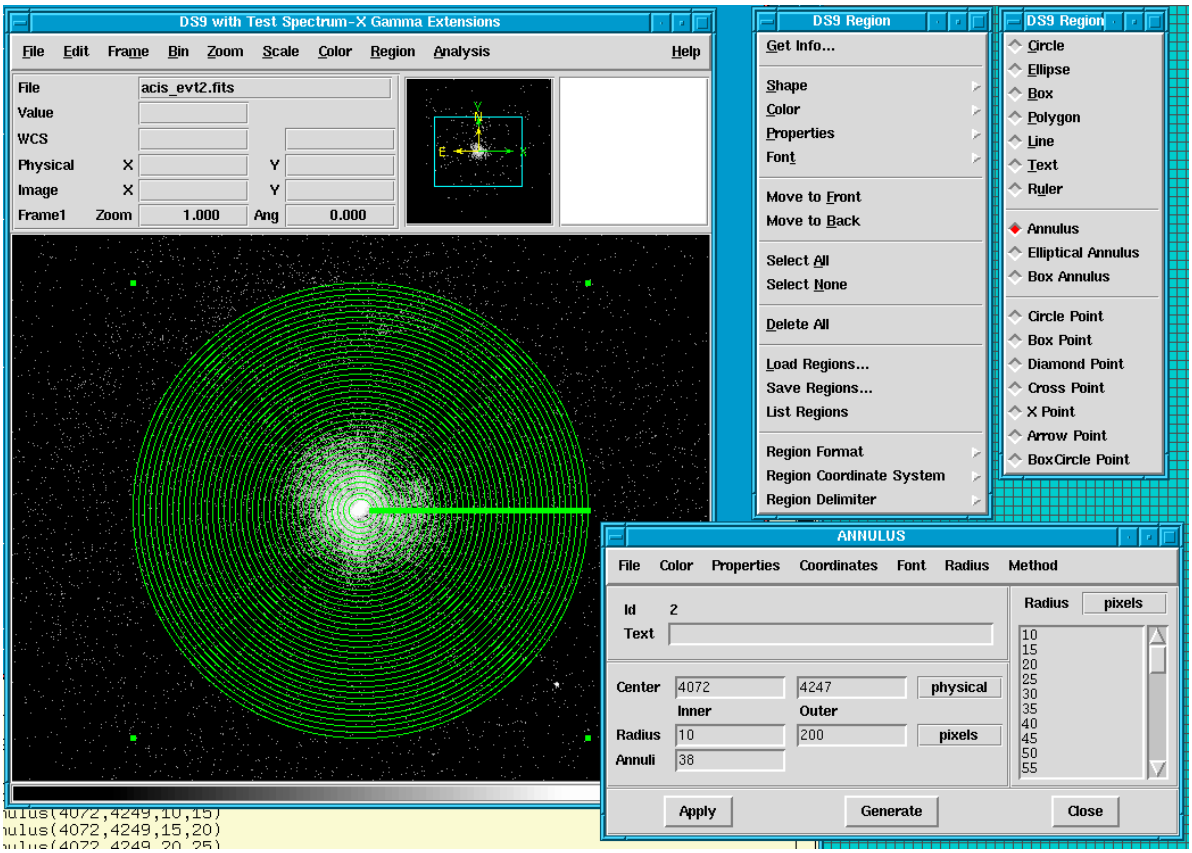
A Simple Case (1)

Consider the ACIS exposure of the Supernova Remnant G21.5-0.9, shown here:



A Simple Case (2)

We use DS9 to define multiple concentric annuli by selecting the "annulus" shape under the **Region** menu and then editing it using the **Get Info...** button. We select 5 pixel wide annuli, from 10-200 pixels, excluding the bright, central source.



A Simple Case (3)

Saving the regions (remember to select CIAO Format) results in the following file

```
moose-28: more annuli.reg
# Region file format: CIAO version 1.0
annulus(4072,4249,10,15) & annulus(4072,4249,15,20) & \
annulus(4072,4249,20,25) & annulus(4072,4249,25,30) & \
annulus(4072,4249,30,35) & annulus(4072,4249,35,40) & \
annulus(4072,4249,40,45) & annulus(4072,4249,45,50) & \
.
.
```

This needs to be converted to a **stack** of annuli using the **fixannuli** script (note, this step will no longer be necessary once DS9 Version 2.1 is incorporated into CIAO):

```
moose-29: fixannuli < annuli.reg
annulus(4072,4249,10,15)
annulus(4072,4249,15,20)
annulus(4072,4249,20,25)
annulus(4072,4249,25,30)
.
.
```

We aslo create a single background annulus:

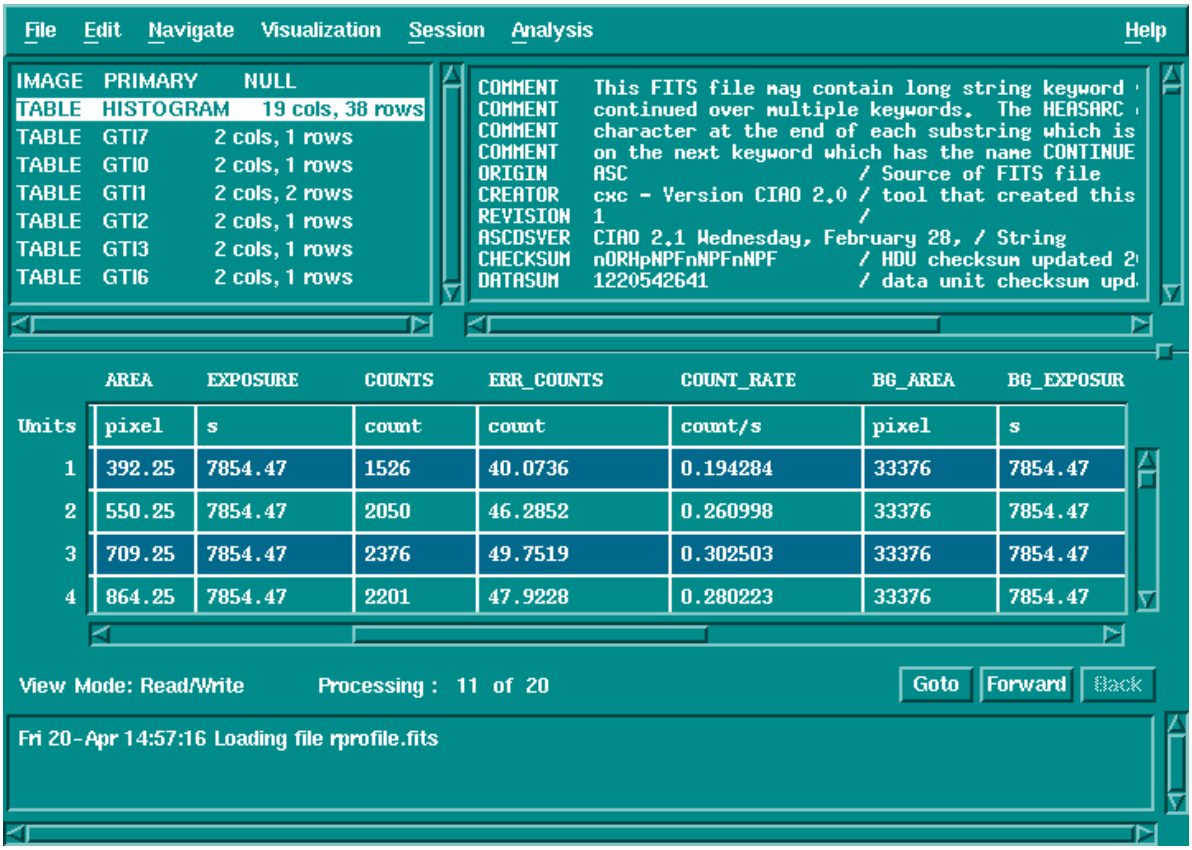
```
moose-31: cat annuli_bgd.reg
# Region file format: CIAO version 1.0
annulus(4071,4249,200,225)
```

A Simple Case (4)

We're now ready to extract counts in the individual annular regions:

```
moose-32: dmextract \
infile="acis_evt2.fits[bin sky=@annuli.fixed.reg]"\
bkg="acis_evt2.fits[bin sky=@annuli_bgd.reg]"\
outfile=rprofile.fits
```

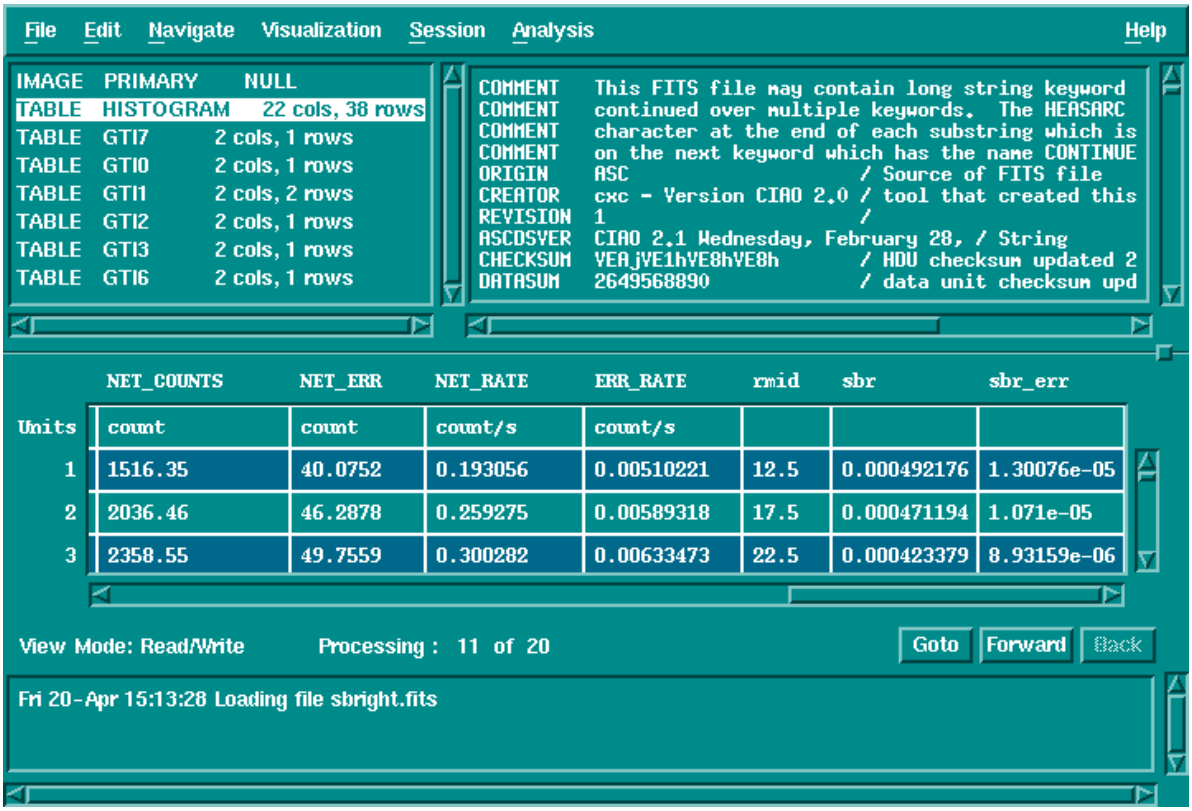
The output can be examined with PRISM:



A Simple Case (5)

Finally, we need to normalize the counts in each annulus by the area, using the DMTCALC expression file rpcolumns.lis:

```
moose-33: cat rpcolumns.lis
.rmid=R[0]+R[1]
rmid=.rmid/2.0
sbr=NET_RATE/AREA
sbr_err=ERR_RATE/AREA
moose-34: dmtcalc infile=rprofile.fits \
outfile=sbright.fits expression=@rpcolumns.lis
```



A Simple Case (6)

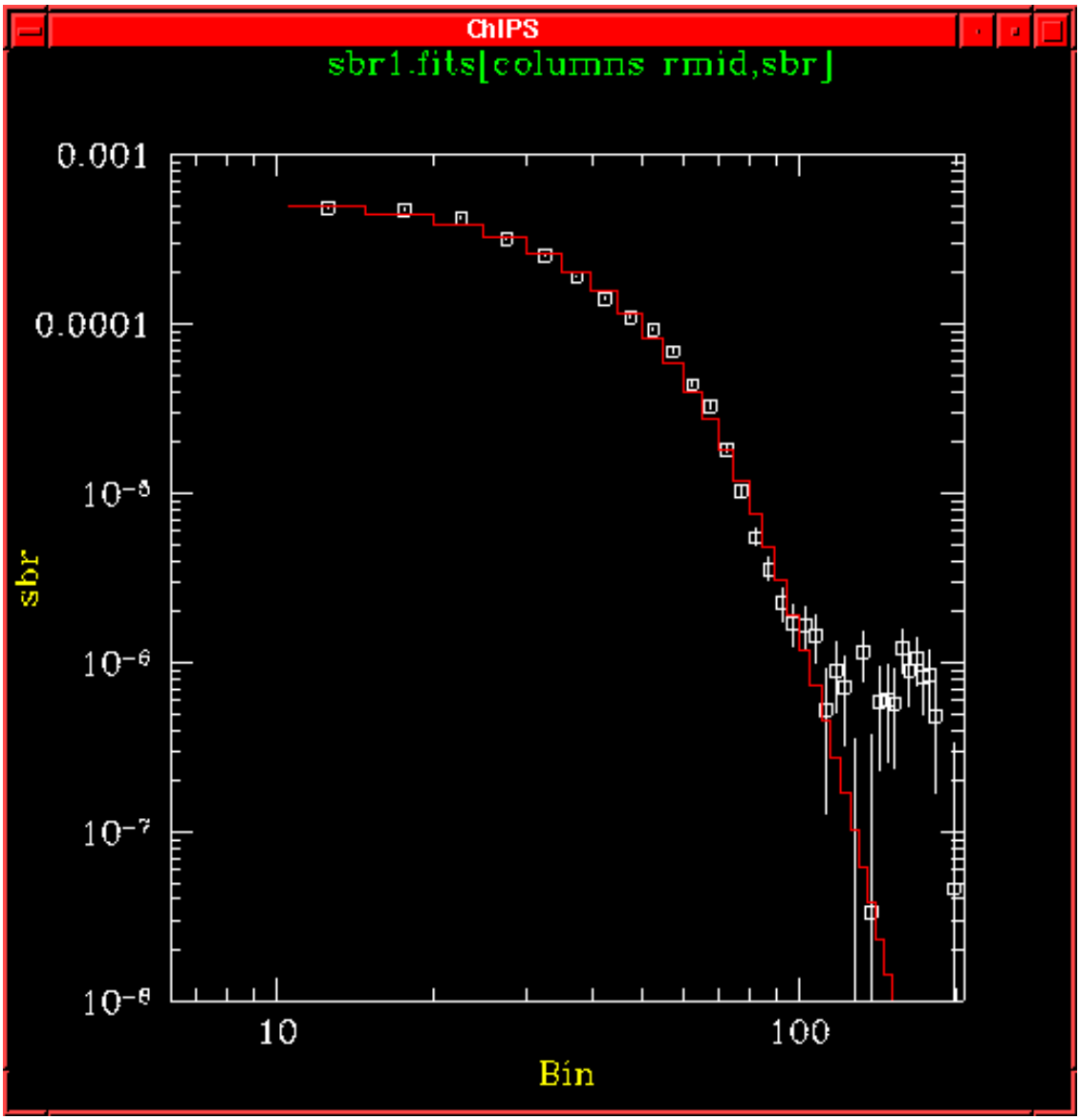
The radial profile can be plotted and fit using SHERPA:

```
moose-35: sherpa
_____________________________________

Welcome to Sherpa: CXC's Modeling and Fitting Program
_____________________________________

Version: 2.1 (1 Dec 2000)
Type HELP for help options.
Type EXIT, QUIT, or BYE to leave the program.
Notes:
Temporary files for visualization will be written to the
    directory:
/tmp
To change this so that these files are not deleted when you
    exit Sherpa,
edit the variable ASCDS_WORK_PATH in your
    $HOME/.cxcds.*sh setup script.
Solar abundances set to Anders & Grevesse
sherpa> read data 1 "sbright_pos.fits[columns rmid,sbr]"
    FITSBIN
sherpa> read errors 1 "sbright_pos.fits[columns
    rmid,sbr_err]" FITSBIN
sherpa> beta1d[sbr1]
sbr1.r0 parameter value [102] 50
sbr1.beta parameter value [0]
sbr1.xpos parameter value [0]
sbr1.ampl parameter value [2.14602e-06] 5.0e-4
```

```
Warning: Parameter is out of range, please reset maximum
    range
if you intend to fit.
Warning: Parameter sbr1.ampl is out of range.
sherpa> sbr1.ampl.max=1.0e-3
sherpa> show sbr1
beta1d[sbr1]  (integrate: off)
   Param  Type     Value      Min      Max          Units
   -----  ------   -------  -------  -------      -----
 1     r0 frozen        50        1   192.5000
  2       beta   thawed                        0-3.4028e+38
    3.4028e+38
  3       xpos   thawed                        0-3.4028e+38
    3.4028e+38
 4   ampl thawed     0.0005 2.1460e-08     0.0010
sherpa> source=sbr1
sherpa> thaw sbr1.r0
sherpa> fit
 powll: v1.2
 powll:   initial function value =     4.66941E+02
 powll:      converged to minimum =     1.44025E+02 at
    iteration =     15
 powll:   final function value    =     1.44025E+02
         sbr1.r0  107.889
         sbr1.beta  3.21701
         sbr1.xpos  0
         sbr1.ampl  0.000581895

sherpa> lplot fit
sherpa> log
```

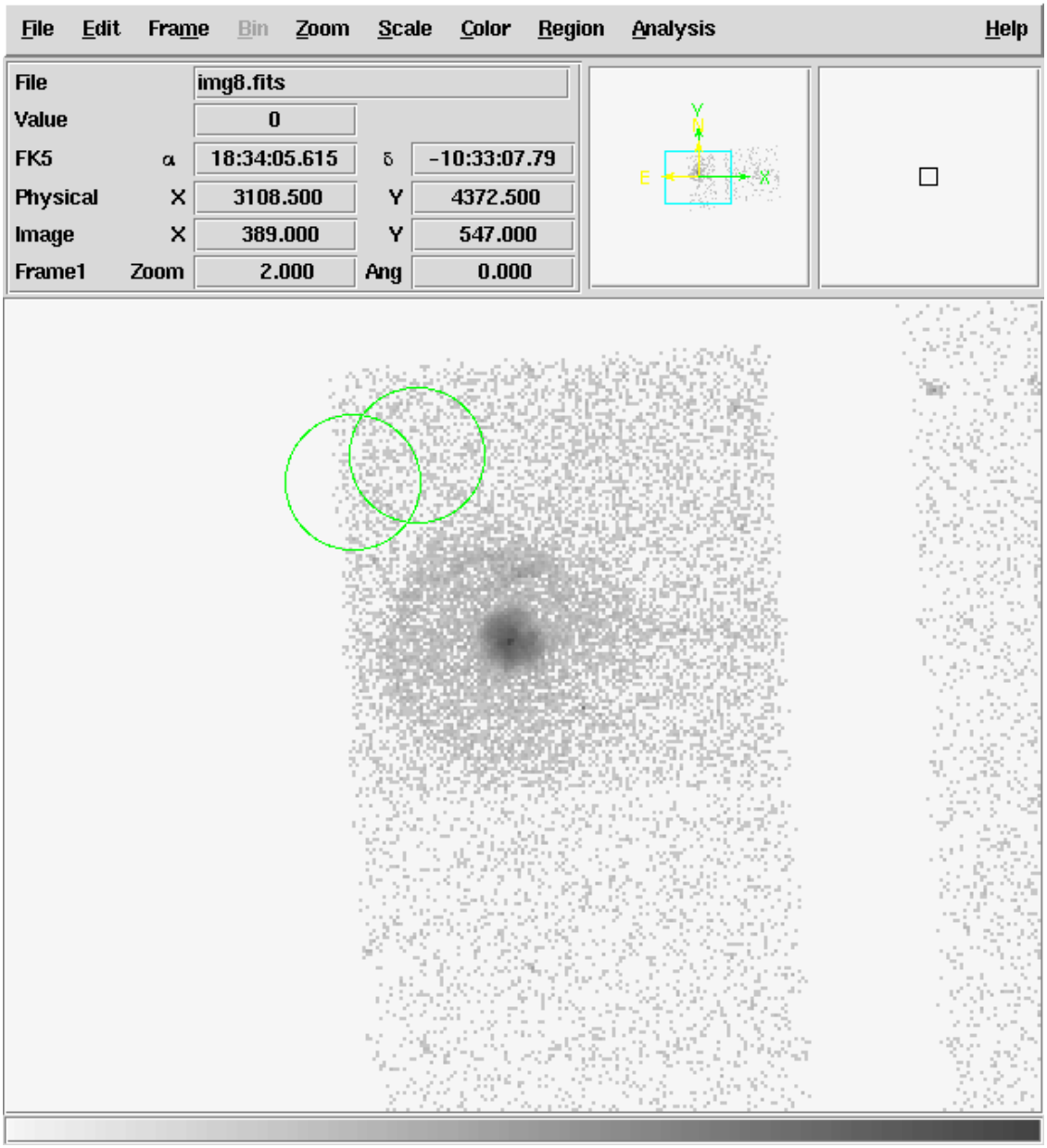
```
sherpa> limits y 1.0e-8 1.0e-3
sherpa> redraw
```

to produce the following plot:



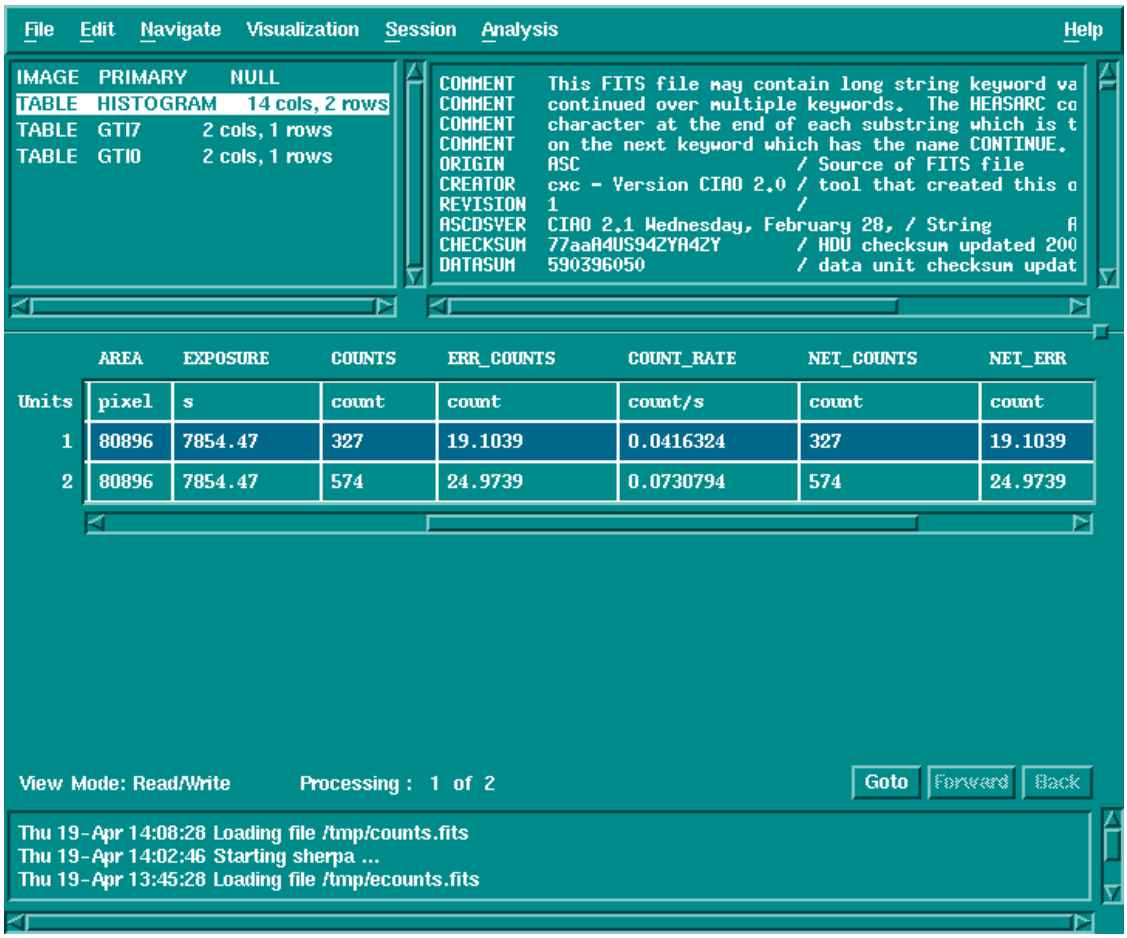
What Happens Near Chip Edges (1)

If you want to extract counts from regions near detector boundaries, you need to be careful. Consider the following case:



Edges (2)

If we simply run dmextract as before, we get the wrong answer, i.e., the counts in the two circles are different by $\sim \times 2$:

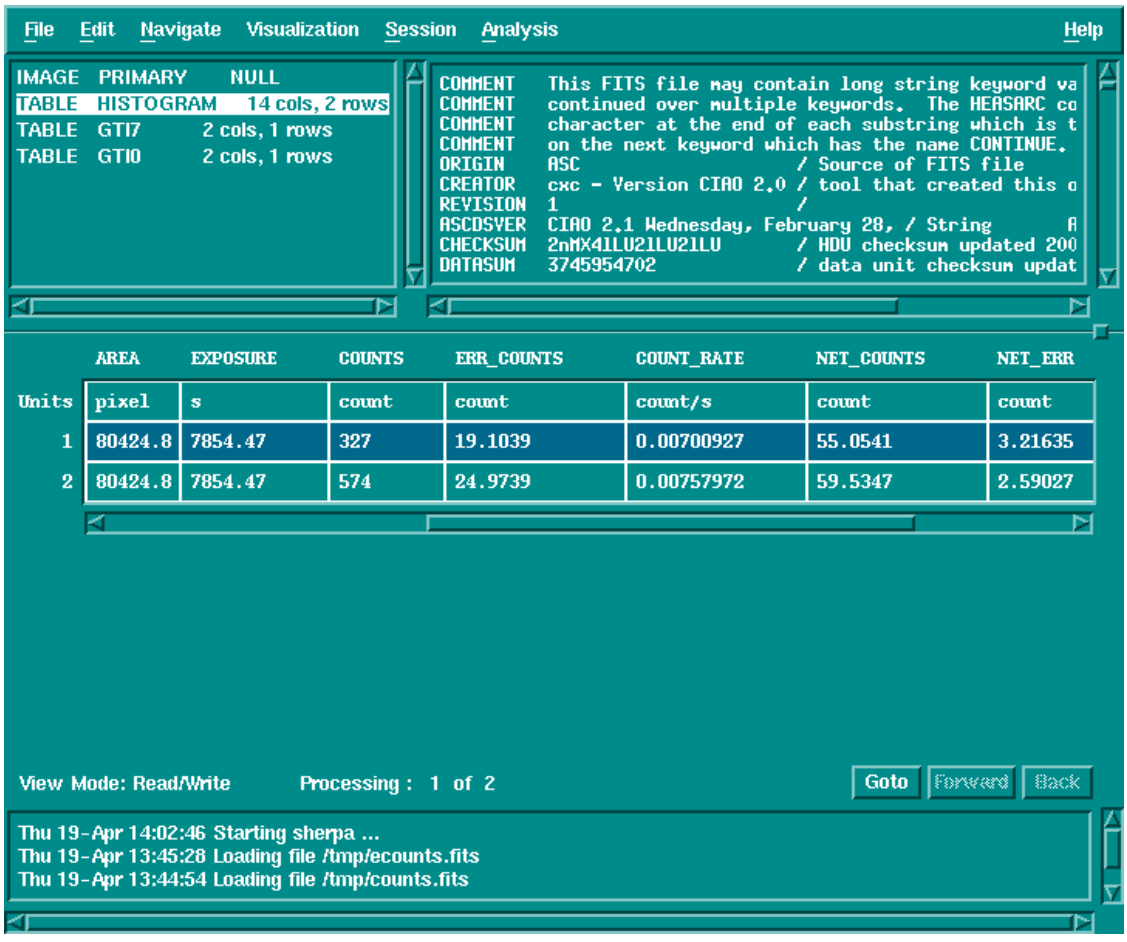


Edges (3)

We need to correct by the exposure, using an exposure map congruent with the image. Here, we use expmap_1.7kev.fits, made for these data, ans in the multi-chip exposure map thread:

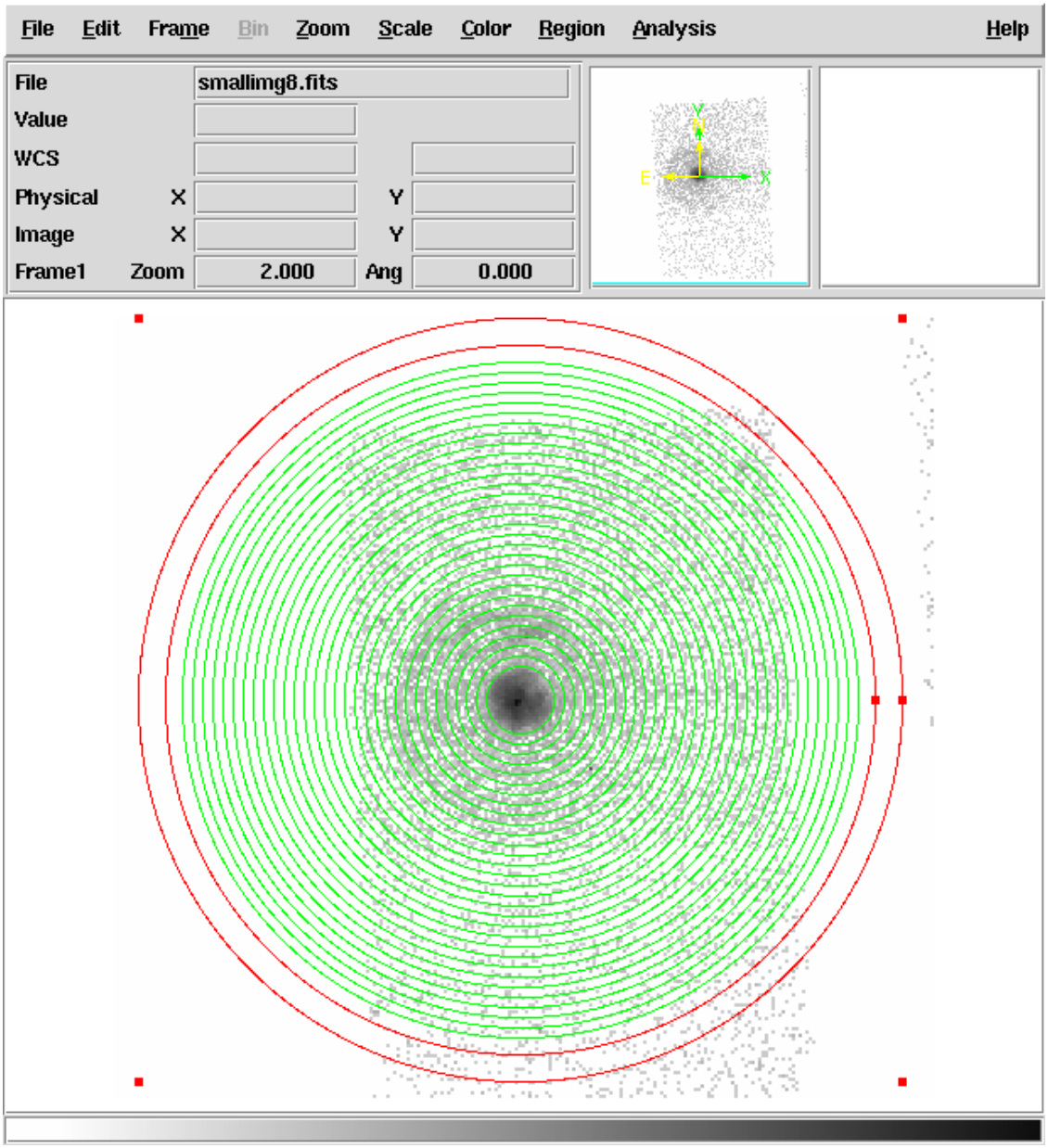
```
moose-16: dmextract \
infile="img8.fits[bin sky=@img8_edge.reg]"\
outfile=counts.fits exp=expmap_1.7kev.fits
# 11897: dmextract:
WARNING: TLMIN and TLMAX not found for column sky
moose-17:
```

which results in the following:



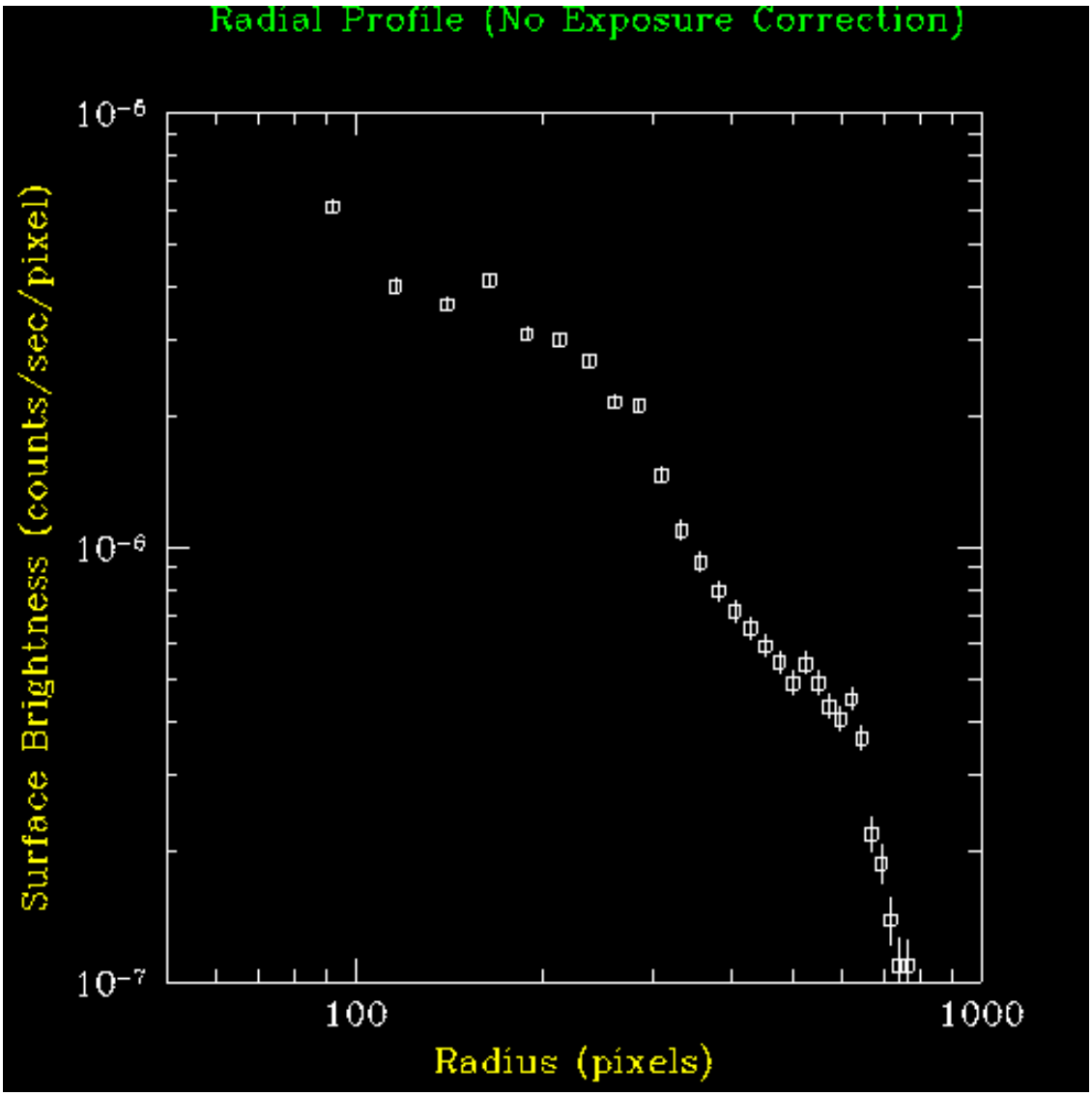
Edges (4)

Of course, what we're really interested in is something like:



Edges (5)

Without exposure corrections, the radial profile looks like:



Edges (6)

With exposure corrections for both source and background:

```
dmextract infile="smallimg8.fits[bin sky=@img8.reg]" \
bkg="smallimg8.fits[bin sky=@img8_bgd.reg]" \
exp=smallexp.fits bkgexp=smallexp.fits \
outfile=counts.fits
```