Using the Output of Detect Tools

CIAO 3.4 Science Threads

Table of Contents

Using the Output of Detect Tools

CIAO 3.4 Science Threads

Overview

Last Update: 1 Dec 2006 − updated for CIAO 3.4: ChIPS version

Synopsis:

All three Detect programs have a common ouput format, the `_src` file; see the Output of Detect: _src Files section of the Detect Overview thread for more information. This thread shows how the source list can be displayed, edited, and used in CIAO data analysis.

Related Links:

- Introduction to Prism thread: many of the tasks featured here may also be accomplished with *Prism's* editing and viewing capabilities.

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

Getting Started

This thread uses products created in the Running celldetect thread. The filtered event files and output source lists are available in detectout.tar.gz (8 MB tarred & gzipped; 14 MB unpacked).

Displaying the Source List

The sources can easily be displayed overlaid on an image:

```
unix%  ds9 s3_evt2.fits &
```

Load the region file from the ds9 menu "Regions" −> "Load Regions..." −> `s3_src.reg`. Alternatively, the FITS filename and extension could be used (`s3_src.fits[SRCLIST]`).

There are two things to note here:

1. FITS files ***must*** end in `.fits` for ds9 to recognize them.

2. `[SRCLIST]` indicates which block of the file the region information is stored in. This extension name is the same for all three detect tools.

Either command will produce Figure 1.

Examining the Source List

The contents of the source list can be viewed with `dmlist`:

```
unix% dmlist s3_src.fits blocks

--------------------------------------------------------------------------------
Dataset: s3_src.fits
--------------------------------------------------------------------------------

     Block Name                          Type         Dimensions
--------------------------------------------------------------------------------
Block    1: PRIMARY                        Null
Block    2: SRCLIST                        Table        31 cols x 17       rows
```

There are 17 sources (one per row) in the file. To see information about the individual sources:

```
unix% dmlist "s3_src.fits[cols RA,DEC,POS]" data

--------------------------------------------------------------------------------
Data for Table Block SRCLIST
--------------------------------------------------------------------------------

ROW    RA                   DEC                  POS(X,Y)

     1       212.8640866629        52.1921374567 (     3961.1449275362,     4035.1449275362)
     2       212.8475950457        52.2255944051 (     4035.1639344262,     4279.9295081967)
     3       212.8362572403        52.2022148086 (     4085.9756097561,     4108.8536585366)
(cut)
    15       212.7874215851        52.2793561164 (     4304.6153846154,     4673.3692307692)
    16       212.7676912429        52.2989312197 (     4392.8109756098,     4816.6707317073)
    17       212.8529855660        52.3233207794 (     4011.1911764706,     4995.0073529412)
```

Here the RA, Dec, and sky position of each source is given. Any of the columns in the file can be used in the DM filter.

Manipulating the Source List

It is easy to delete unwanted sources from or rearrange the order of the ASCII output (region file) of the detect tools via a text editor. Using the Data Model tools, it is just as straightforward to manipulate the FITS source file as well.

Edit the List of Sources

Once the source list is overlaid on the image, you may see some sources that you want to exclude from the analysis. Currently, ds9 does not number the sources in the field of view. However, double−clicking on a source region brings up the region info box, which lists the region "ID". In Figure 2, the source on the left side of the field of view was chosen −− the ID is 14. This number corresponds to the row the region occupies in the FITS file.

NOTE: this technique only works with the first region file loaded into ds9. After that, the imager starts counting up. For example, if you load region_a.fits that has 10 sources, then region_b.fits that has 6, the regions of the second file will be counted from 11 onward. One may, of course, always examine the region file visually to determine which rows to delete.

To create a new region file that doesn't have row 14 in it, `dmcopy` is used:

```
unix% dmcopy "s3_src.fits[#row=1:13,15:17]" s3_1gone_src.fits

unix% dmlist s3_1gone_src.fits blocks

--------------------------------------------------------------------------------
Dataset: s3_1gone_src.fits
--------------------------------------------------------------------------------

     Block Name                          Type         Dimensions
--------------------------------------------------------------------------------
Block    1: PRIMARY                        Null
Block    2: SRCLIST                        Table        31 cols x 16       rows
```

The `dmlist` command shows that there are only 16 rows in the file now. This new file may be displayed in ds9 to ensure that the correct source was removed.

This filter was fairly simple. It is possible to make more complicated filters with the DM syntax. If you wanted to exclude rows (sources) 8, 12, and 14 from the original file:

```
unix% dmcopy "s3_src.fits[#row=1:7,9:11,13,15:17]" s3_3gone_src.fits
```

Figure 4 shows the new source list, indicating where the deleted objects used to be.

Sort on a Particular Column

The tool `dmsort` makes it possible to sort on any column in the FITS file. To order the rows by `NET_COUNTS`:

```
unix% dmlist "s3_src.fits[cols net_counts]" data

--------------------------------------------------------------------------------
Data for Table Block SRCLIST
--------------------------------------------------------------------------------

ROW    NET_COUNTS

     1                 60.0
     2              544.750
     3             48.81250
(cut)
    15                 47.0
    16       112.9583358765
    17              80.8750

unix% dmsort s3_src.fits s3_sorted_src.fits keys=net_counts copyall=yes

unix% dmlist "s3_sorted_src.fits[cols net_counts]" data
```

```

--------------------------------------------------------------------------------
Data for Table Block SRCLIST
--------------------------------------------------------------------------------

ROW    NET_COUNTS

     1        22.7142848969
     2        25.1428565979
     3        33.1428565979
(cut)
    15       112.9583358765
    16       190.2857208252
    17             544.750
```

The default behavior of `dmsort` is to organize the results in ascending order. To sort `NET_COUNTS` in *descending* order, add a dash ("`−`") before the column name:

```
unix% dmsort s3_src.fits s3_descend_src.fits keys=−net_counts copyall=yes

unix% dmlist "s3_descend_src.fits[cols net_counts]" data

--------------------------------------------------------------------------------
Data for Table Block SRCLIST
--------------------------------------------------------------------------------

ROW    NET_COUNTS

     1             544.750
     2       190.2857208252
     3       112.9583358765
(cut)
    15        33.1428565979
    16        25.1428565979
    17        22.7142848969
```

It is also possible to have several sort priorities (i.e. sort by `NET_COUNTS`, then `NET_COUNTS_ERR`); see ahelp dmsort for more information on how to sort files.

Filtering the Source List

If you are interested in only using a certain range of data in your analysis, use the DM filtering syntax to create a new source list with only those items.

To keep the sources that have 40 or more net counts:

```
unix% dmcopy "s3_src.fits[net_counts=40:]" s3_40cts_src.fits
unix% dmlist "s3_40cts_src.fits[cols net_counts]" data

--------------------------------------------------------------------------------
Data for Table Block SRCLIST
--------------------------------------------------------------------------------

ROW    NET_COUNTS

     1                60.0
     2             544.750
     3            48.81250
     4            79.43750

     5             104.6250
     6             44.93750
     7       190.2857208252
     8        54.1428565979
     9                 47.0
    10       112.9583358765
    11              80.8750
```

or those with `NET_COUNTS` between 25 and 80:

```
unix% dmcopy "s3_src.fits[net_counts=25:80]" s3_25−80cts_src.fits
unix% dmlist "s3_25−80cts_src.fits[cols net_counts]" data

−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
Data for Table Block SRCLIST
−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−

ROW    NET_COUNTS

     1                 60.0
     2             48.81250
     3             79.43750
     4             44.93750
     5        39.4285697937
     6        25.1428565979
     7        33.1428565979
     8        54.1428565979
     9        37.2857131958
    10        38.4772720337
    11                 47.0
```

See ahelp dmfiltering for other variations on the filtering syntax.

Converting Coordinates to Sexagesimal

Often it is desirable to have the list of sources in sexagesimal format, i.e. (RA,Dec) = (hms,dms). There are two ways that this may be done − in ds9 or with *S−Lang*.

- ***In ds9:*** load the event file and display the source list on top of it. From the "Region" menu, set "File Format −> CIAO" and "File Coordinate System −> WCS ***and*** Equatorial J2000". Finally, "Save Regions" as `s3_src_ds9.reg`.

  The output (an ASCII region file) still contains the shape definitions, but the coordinates are now given as sexagesimal:

  ```
  unix% more s3_src_ds9.reg
  # Region file format: CIAO version 1.0
  ellipse(14:11:27.380,+52:11:31.70,0.0195851',0.0173948',45)
  ellipse(14:11:23.422,+52:13:32.14,0.0183431',0.0158548',53.264)
  ellipse(14:11:20.701,+52:12:07.98,0.0197903',0.0184924',15.4458)
  ellipse(14:11:20.530,+52:12:10.08,0.0191862',0.0170336',155.484)
  ellipse(14:11:20.387,+52:12:11.51,0.0203776',0.0173993',75.4574)
  ellipse(14:11:13.633,+52:13:41.46,0.0169192',0.0153512',12.7196)
  ellipse(14:11:30.846,+52:14:24.20,0.0390769',0.0242176',31.4398)
  ellipse(14:11:29.124,+52:13:33.01,0.0321857',0.0254928',125.887)
  ellipse(14:11:20.686,+52:14:11.97,0.03602',0.0308016',32.1849)
  ellipse(14:11:19.464,+52:14:00.94,0.0286652',0.0194748',33.3924)
  ellipse(14:10:59.522,+52:11:53.84,0.0395356',0.0245289',151.558)
  ellipse(14:10:57.363,+52:11:31.25,0.0386767',0.0297596',123.502)
  ```

```
ellipse(14:11:14.499,+52:16:11.60,0.0546014',0.0455196',5.8862)
ellipse(14:11:29.153,+52:16:43.48,0.0937733',0.0753438',149.301)
ellipse(14:11:08.981,+52:16:45.68,0.0790725',0.0635626',174.887)
ellipse(14:11:04.245,+52:17:56.15,0.0995391',0.089148',15.4741)
ellipse(14:11:24.716,+52:19:23.96,0.131417',0.107319',167.549)
```

The benefit of this method is that the file may be redisplayed in ds9, as it retains the CIAO region format.

- ***With S−Lang:*** use a small `slang` script, deg2sexag.sl, to convert the FITS source list. Download the script and load it into *ChIPS* with the `evalfile` command:

```
unix% chips

Welcome to ChIPS, version CIAO 3.4
Copyright (C) 1999−2003, Smithsonian Astrophysical Observatory

chips> evalfile("deg2sexag.sl")
1
chips> sources=readfile("s3_src.fits")
chips> (ra,dec)=deg2sexag(sources.RA,sources.DEC)
chips> writeascii(stdout,sources.COMPONENT,ra,dec,sources.X,sources.Y)
1       14:11:27.38     +52:11:31.69    3961.14 4035.14
2       14:11:23.42     +52:13:32.14    4035.16 4279.93
3       14:11:20.70     +52:12:07.97    4085.98 4108.85
4       14:11:20.53     +52:12:10.08    4089.18 4113.13
5       14:11:20.39     +52:12:11.51    4091.84 4116.04
6       14:11:13.63     +52:13:41.46    4217.98 4298.88
7       14:11:30.85     +52:14:24.19    3896.6  4385.79
8       14:11:29.12     +52:13:33.00    3928.69 4281.72
9       14:11:20.69     +52:14:11.96    4086.27 4360.86
10      14:11:19.46     +52:14:00.94    4109.08 4338.45
11      14:10:59.52     +52:11:53.84    4481.75 4080.36
12      14:10:57.36     +52:11:31.25    4522.16 4034.49
13      14:11:14.50     +52:16:11.60    4201.7  4604.05
14      14:11:29.15     +52:16:43.48    3928.33 4668.88
15      14:11:08.98     +52:16:45.68    4304.62 4673.37
16      14:11:04.25     +52:17:56.15    4392.81 4816.67
17      14:11:24.72     +52:19:23.95    4011.19 4995.01

chips> writeascii("s3_src_sexag.reg",sources.COMPONENT,ra,dec)

chips> exit
```

The output (`s3_src_sexag.reg`) is an ASCII source file which contains the positions in the new format:

```
unix% more s3_src_sexag.reg
1       14:11:27.38     +52:11:31.69
2       14:11:23.42     +52:13:32.14
3       14:11:20.70     +52:12:07.97
(cut)
15      14:11:08.98     +52:16:45.68
16      14:11:04.25     +52:17:56.15
17      14:11:24.72     +52:19:23.95
```

This output cannot be loaded into ds9 as it no longer contains the shape definitions. It would, however, be more straightforward to incorporate this output into a latex table than the ds9 output.

For ease in using this frequently, it may be added to the `/home/username/.varmmrc` resource file. See `ahelp configure` for information on using resource files.

Using the Source List in Analysis

Before beginning any analysis with the source list, be aware that the scaling of the size of each region is controlled by the `ellsigma` parameter in each of the detect tools. From the help files:

```
ellsigma is a multiplicative factor applied to sigma ... to scale the major and minor
axes of the ellipses for each source. ellsigma affects both the outfile and the ASCII
region file (regfile). This feature is included so that the graphics overlay will be
more visible and under the user's control.
```

Carefully consider a reasonable value for `ellsigma` before using these regions for analyses.

Creating Individual Spectra

It is possible to create a spectrum from any of the regions in the source list without having to divide up the file. To extract a PI spectrum of the second source in the file and bin it by a factor of 5:

```
unix% punlearn dmextract
unix% dmextract "s3_evt2.fits[sky=region(s3_src.fits[#row=2])][bin pi=::5]" \
      spectrum_2.fits
```

The contents of the parameter file may be checked using plist dmextract. The resultant spectrum can be plotted in *ChIPS*:

```
unix% chips

Welcome to ChIPS, version CIAO 3.4
Copyright (C) 1999−2003, Smithsonian Astrophysical Observatory

chips> curve "spectrum_2.fits[cols channel,counts]"
chips> curve simpleline; symbol none
chips> limits x 0 100
chips> xlabel "channel"; xlabel size 1.5
chips> ylabel "counts"; ylabel size 1.5
chips> exit
```

Figure 6 shows the plot. This data may also be fit in *Sherpa*, once the proper response files have been created; see the Step−by−Step Guide to Creating ACIS Spectra for Pointlike Sources and Independent Background Responses threads for more information.

Creating Multiple Spectra

To create an individual spectrum for each of the sources, use the `lgrid(1:n:steps)` stack syntax, where `n` is the number of sources in the file.

For `s3_src.fits`, which has 17 sources in it:

```
unix% dmextract "s3_evt2.fits[sky=region(s3_src.fits[component=lgrid(1:17:1)])][bin pi=::5]" \
      multi_spectra_phaII.fits
```

The output file is a type II PHA file that has one spectrum per row:

```
unix% dmlist multi_spectra_phaII.fits blocks
```

```

--------------------------------------------------------------------------------
Dataset: multi_spectra_phaII.fits
--------------------------------------------------------------------------------

     Block Name                           Type         Dimensions
--------------------------------------------------------------------------------
Block    1: PRIMARY                         Null
Block    2: SPECTRUM                        Table         7 cols x 17       rows
Block    3: GTI7                            Table         2 cols x 1        rows
```

For more information on working with this type of file, see the Displaying the Spectrum section of the Examining Grating Spectra and Regions: PHA2 files thread (the information in that section applies to imaging files as well).

Creating a Composite Spectrum

It is also possible to extract a composite spectrum of all the (faint) sources in a field for a single spectral analysis. This may be useful in the case where the sources might be of similar characteristics, but there are not enough counts in any given source to do individual spectra.

Here we have deleted the sources associated with 3C295 (IDs 3−5):

```
unix% dmcopy "s3_src.fits[#row=1:2,6:]" s3_nocore_src.fits
```

Now we can use `dmextract` to accept all the regions together and create a spectrum:

```
unix% dmextract "s3_evt2.fits[sky=region(s3_nocore_src.fits)][bin pi=::5]" \
      spectrum_faint.fits
```

Again, the resultant spectrum can be plotted in *ChIPS*:

```
unix% chips

Welcome to ChIPS, version CIAO 3.4
Copyright (C) 1999−2003, Smithsonian Astrophysical Observatory

chips> curve "spectrum_faint.fits[cols channel,counts]"
chips> curve simpleline; symbol none
chips> limits x 0 100
chips> xlabel "channel"; xlabel size 1.5
chips> ylabel "counts"; ylabel size 1.5
chips> exit
```

Figure 7 shows the plot. As mentioned before, this data may also be fit in *Sherpa*, once the proper response files have been created; see the Step−by−Step Guide to Creating ACIS Spectra for Pointlike Sources and Independent Background Responses threads for more information.

Removing Unwanted Point Sources

Often it is desirable to remove point sources from an image for both cosmetic and scientific reasons. For example, if you are analyzing an extended source, an unfortunately located point source may affect the count rates in the object.

Suppose that we want to remove all the point sources from our field of view, except for the three in the core (sources 3−5). First, create a source list that excludes these three regions:

```
unix% dmcopy "s3_src.fits[#row=1:2,6:17]" s3_holes_src.fits
```

Now filter the event file with those regions:

```
unix% dmcopy "s3_evt2.fits[exclude sky=region(s3_holes_src.fits)]" s3_holes_evt2.fits

unix% ds9 s3_holes_evt2.fits &
```

The resulting image, Figure 8, has holes where the sources used to be. The Obtain and Fit a Radial Profile thread illustrates another case in which you might use this syntax (see the Removing Contaminating Point Sources section).

A more sophisticated means of removing point sources is to use `dmfilth`. This tool removes the point source and then fills in the hole where the source was. See ahelp dmfilth for more information and the Identify and Remove Point Sources section of the Create an Image of Diffuse Emission thread for an example.

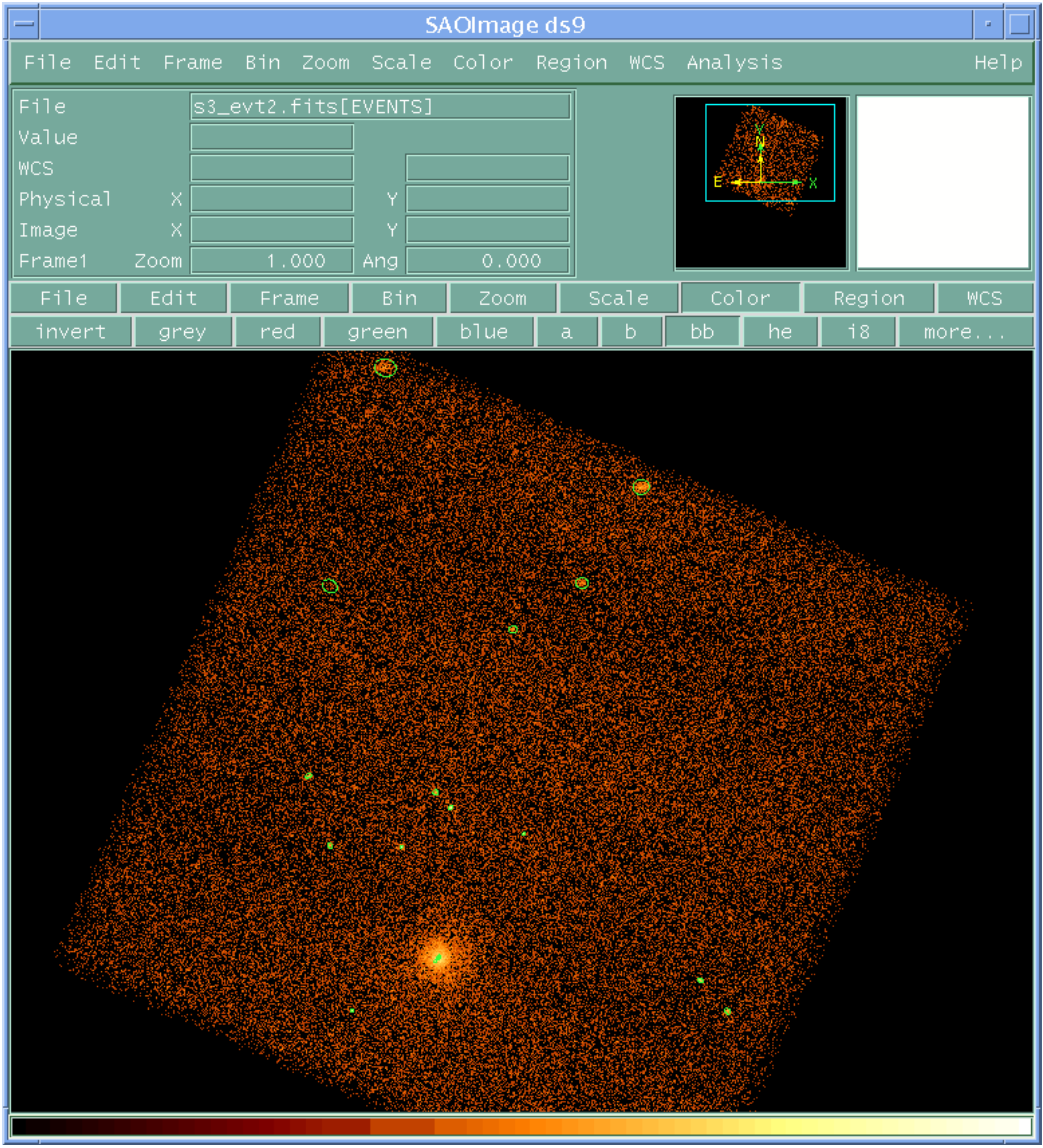
```
Parameters for /home/username/cxcds_param/dmextract.par


#--------------------------------------------------------------------
#
# DMEXTRACT -- extract columns or counts from an event list
#
#--------------------------------------------------------------------
        infile = s3_evt2.fits[sky=region(s3_src.fits[#row=2])][bin pi=::5] Input event file
       outfile = spectrum_2.fits  Enter output file name
          (bkg = )                Background region file or fixed background (counts/pixel/s) subtra
        (error = gaussian)        Method for error determination(poisson|gaussian|<variance file>)
     (bkgerror = gaussian)        Method for background error determination(poisson|gaussian|<varian
      (bkgnorm = 1.0)             Background normalization
          (exp = )                Exposure map image file
       (bkgexp = )                Background exposure map image file
      (sys_err = 0)               Fixed systematic error value for SYS_ERR keyword
          (opt = pha1)            Output file type: pha1
     (defaults = ${ASCDS_CALIB}/cxo.mdb -> /soft/ciao/data/cxo.mdb) Instrument defaults file
         (wmap = )                WMAP filter/binning (e.g. det=8 or default)
      (clobber = no)              OK to overwrite existing output file(s)?
      (verbose = 0)               Verbosity level
         (mode = ql)
```

History

03 Jan 2005	reviewed for CIAO 3.2: no changes
19 Dec 2005	updated for CIAO 3.3: default value of `dmextract` `error` and `bkgerror` parameters is "gaussian", screen output updated accordingly; updated files in detect data tarfile
01 Dec 2006	updated for CIAO 3.4: ChIPS version

URL: http://cxc.harvard.edu/ciao/threads/detect_output/ Last modified: 1 Dec 2006

Image 1: Source list overlaid on image

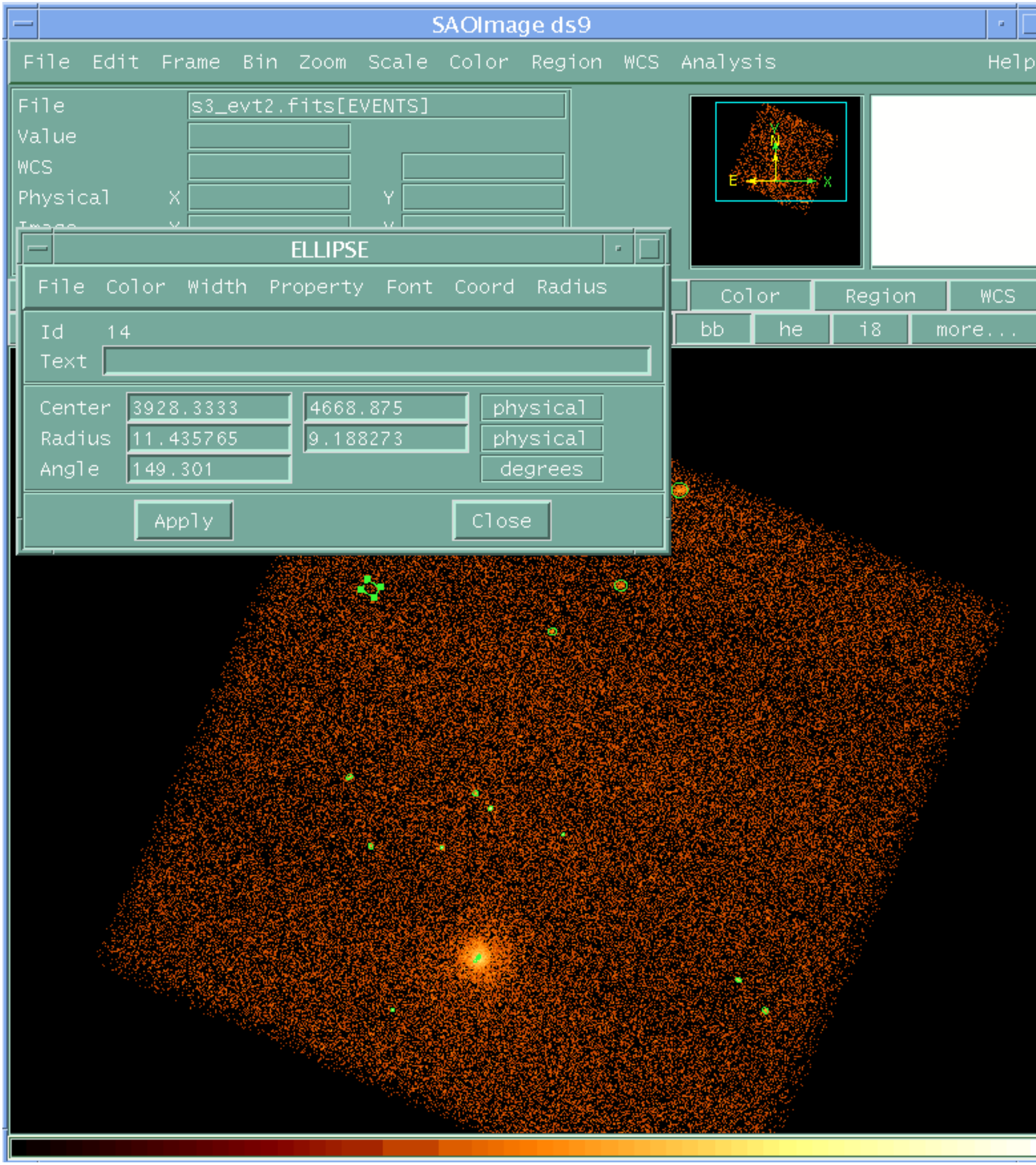


File s3_evt2.fits[EVENTS]

Frame1 Zoom 1.000 Ang 0.000

SAOImage ds9
File Edit Frame Bin Zoom Scale Color Region WCS Analysis Help
File
Value
WCS
Physical X Y
Image X Y
N
E
Y
X
File Edit Frame Bin Zoom Scale Color Region WCS
invert grey red green blue a b bb he i8 more...

Image 2: Finding the ID of a source region


SAOImage ds9
File Edit Frame Bin Zoom Scale Color Region WCS Analysis Help
Color Region WCS
bb he i8 more...
ELLIPSE
File Color Width Property Font Coord Radius
Apply
Close
E
N
Y
X

File s3_evt2.fits[EVENTS]
Value
WCS
Physical X Y
Image X Y

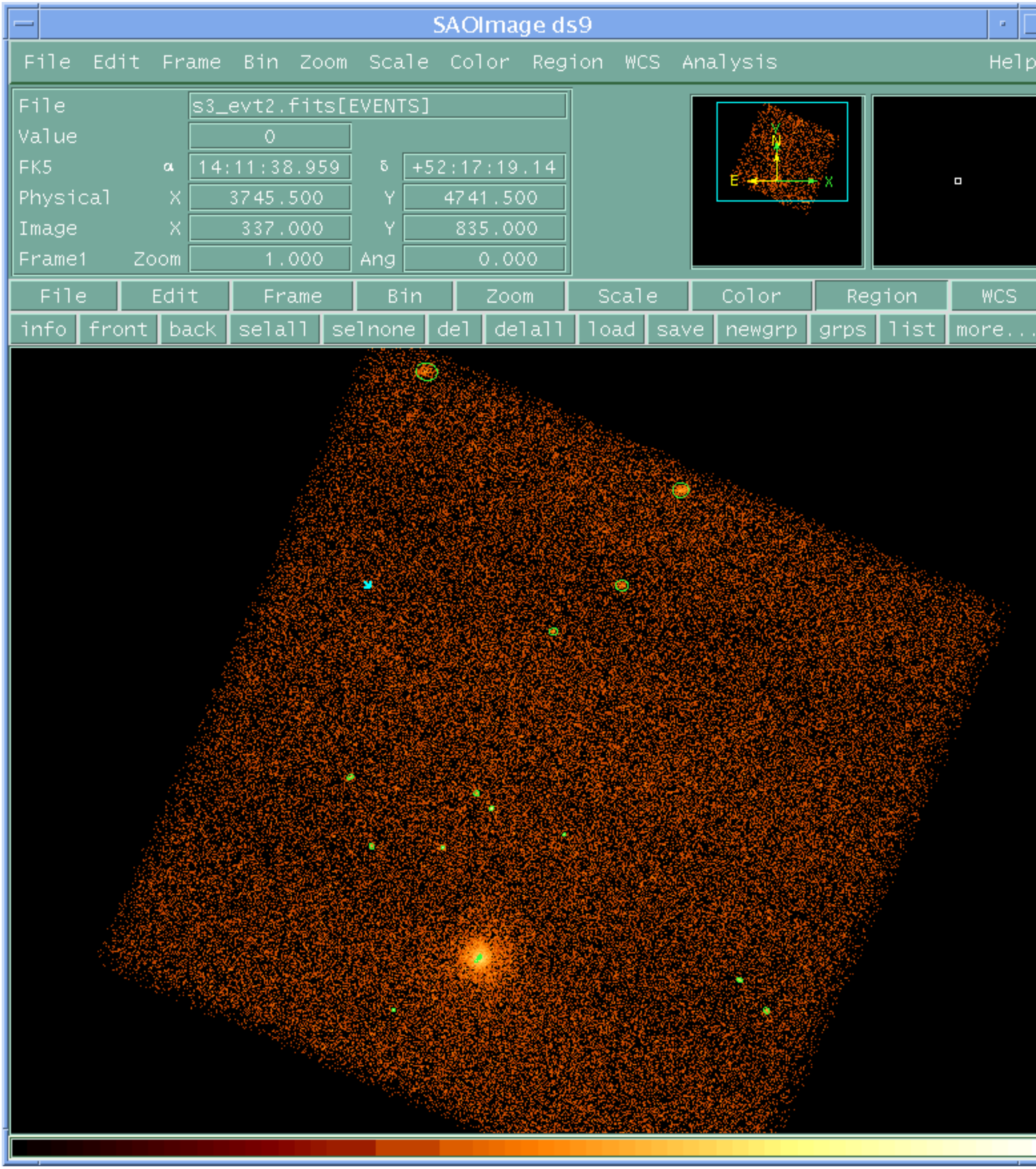
Id 14
Text

Center 3928.3333 4668.875 physical
Radius 11.435765 9.188273 physical
Angle 149.301 degrees

Image 2: Finding the ID of a source region

Image 3: Confirming a removed source

The unwanted source (blue arrow) is no longer present in the region file (green ellipses).

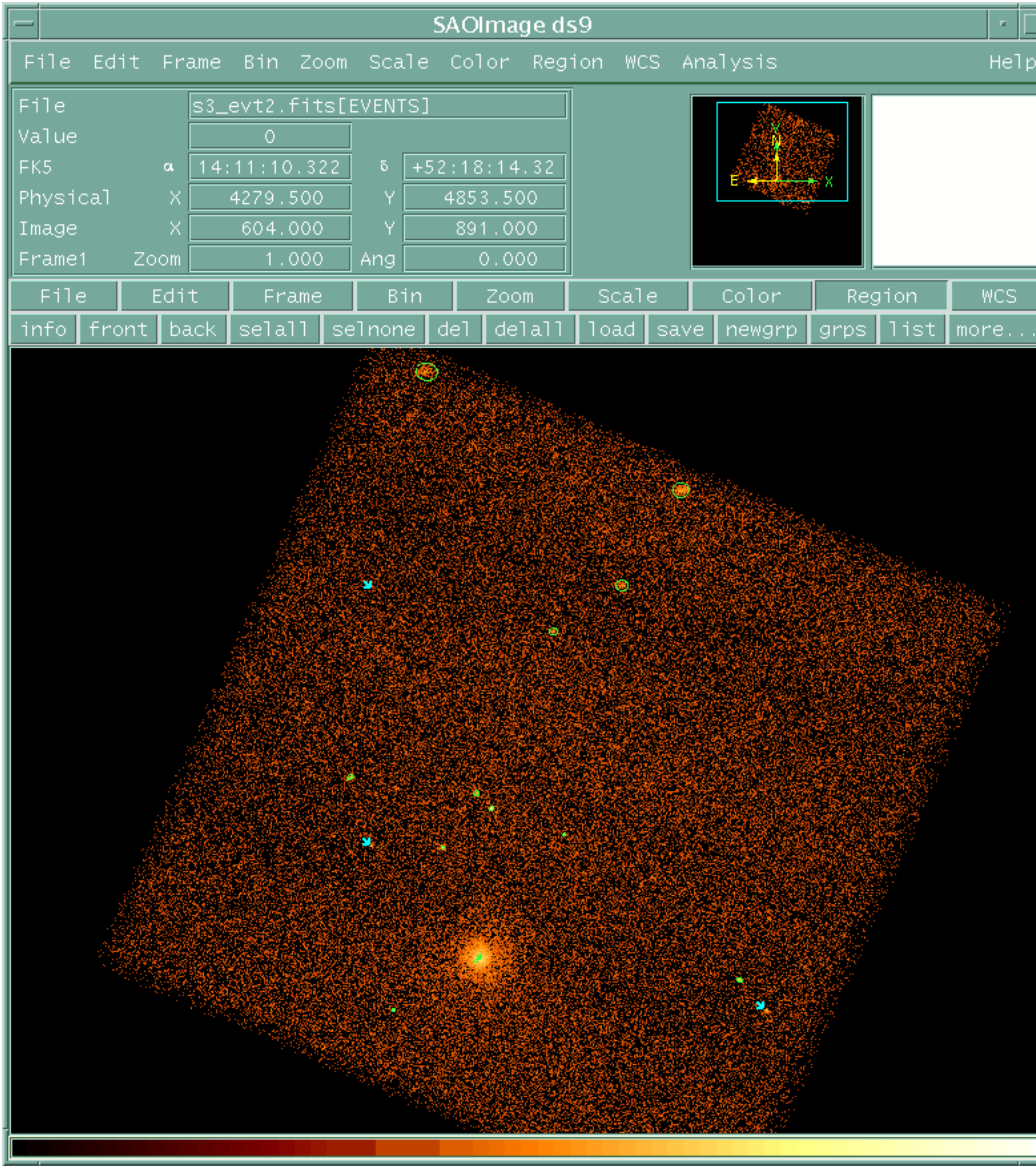

SAOImage ds9
File Edit Frame Bin Zoom Scale Color Region WCS Analysis Help
N
E
X
Y
File Edit Frame Bin Zoom Scale Color Region WCS
info front back selall selnone del delall load save newgrp grps list more...

File s3_evt2.fits[EVENTS]
Value 0
FK5 α 14:11:38.959 δ +52:17:19.14
Physical X 3745.500 Y 4741.500
Image X 337.000 Y 835.000
Frame1 Zoom 1.000 Ang 0.000

Image 3: Confirming a removed source

Image 4: Confirming three deleted sources

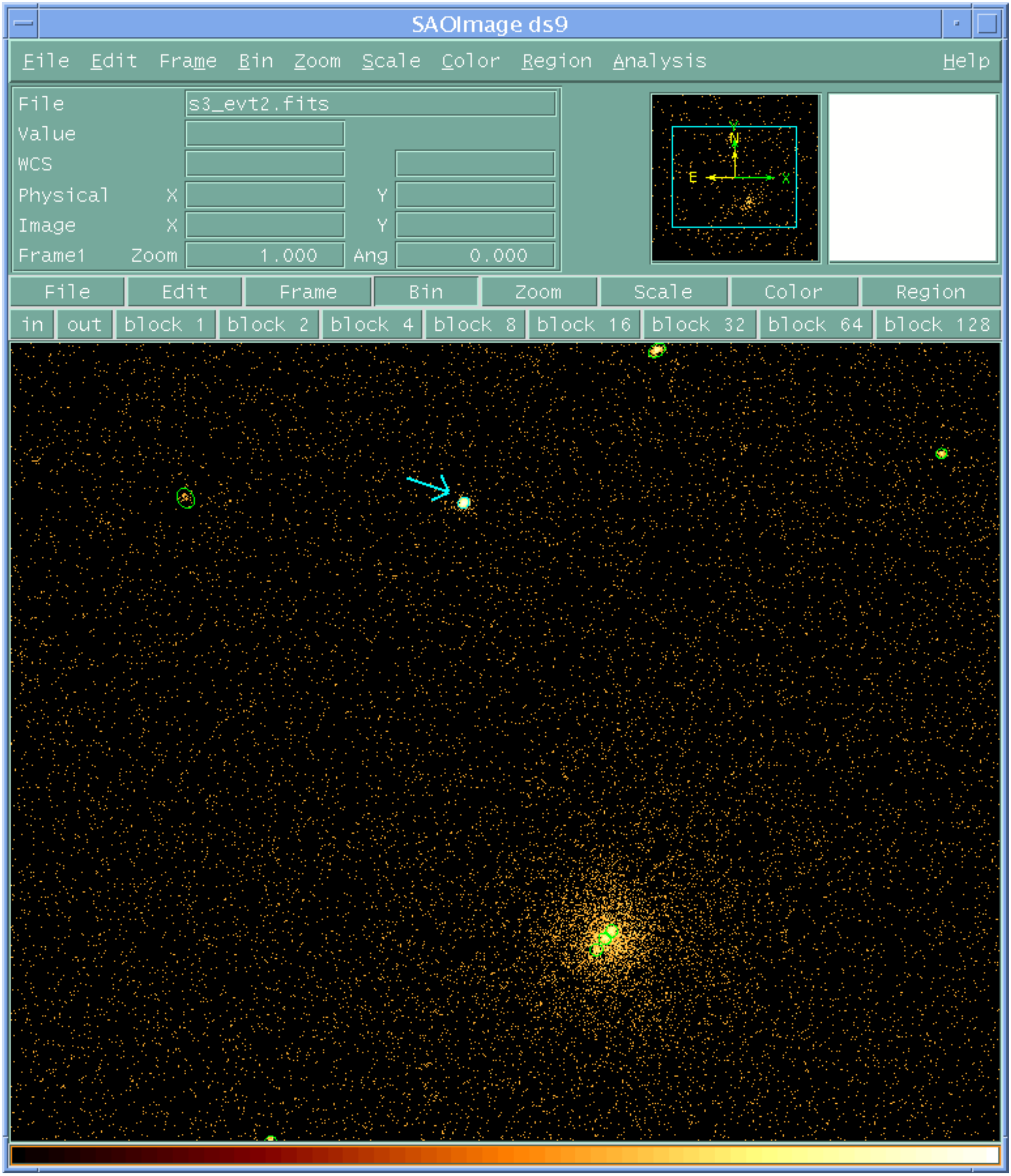
The unwanted sources (blue arrows) are no longer present in the region file (green ellipses).


SAOImage ds9
File Edit Frame Bin Zoom Scale Color Region WCS Analysis Help
File Edit Frame Bin Zoom Scale Color Region WCS
info front back selall selnone del delall load save newgrp grps list more...
N
E
Y
X

File s3_evt2.fits[EVENTS]
Value 0
FK5 α 14:11:10.322 δ +52:18:14.32
Physical X 4279.500 Y 4853.500
Image X 604.000 Y 891.000
Frame1 Zoom 1.000 Ang 0.000

Image 4: Confirming three deleted sources

Image 5: Source 2, used to create the spectrum


SAOImage ds9
File Edit Frame Bin Zoom Scale Color Region Analysis Help
Value
WCS
Physical X Y
Image X Y
N
E
X
Y
File Edit Frame Bin Zoom Scale Color Region
in out block 1 block 2 block 4 block 8 block 16 block 32 block 64 block 128

File s3_evt2.fits

Frame1 Zoom 1.000 Ang 0.000

Image 5: Source 2, used to create the spectrum

Image 6: Plot of the spectrum in chips

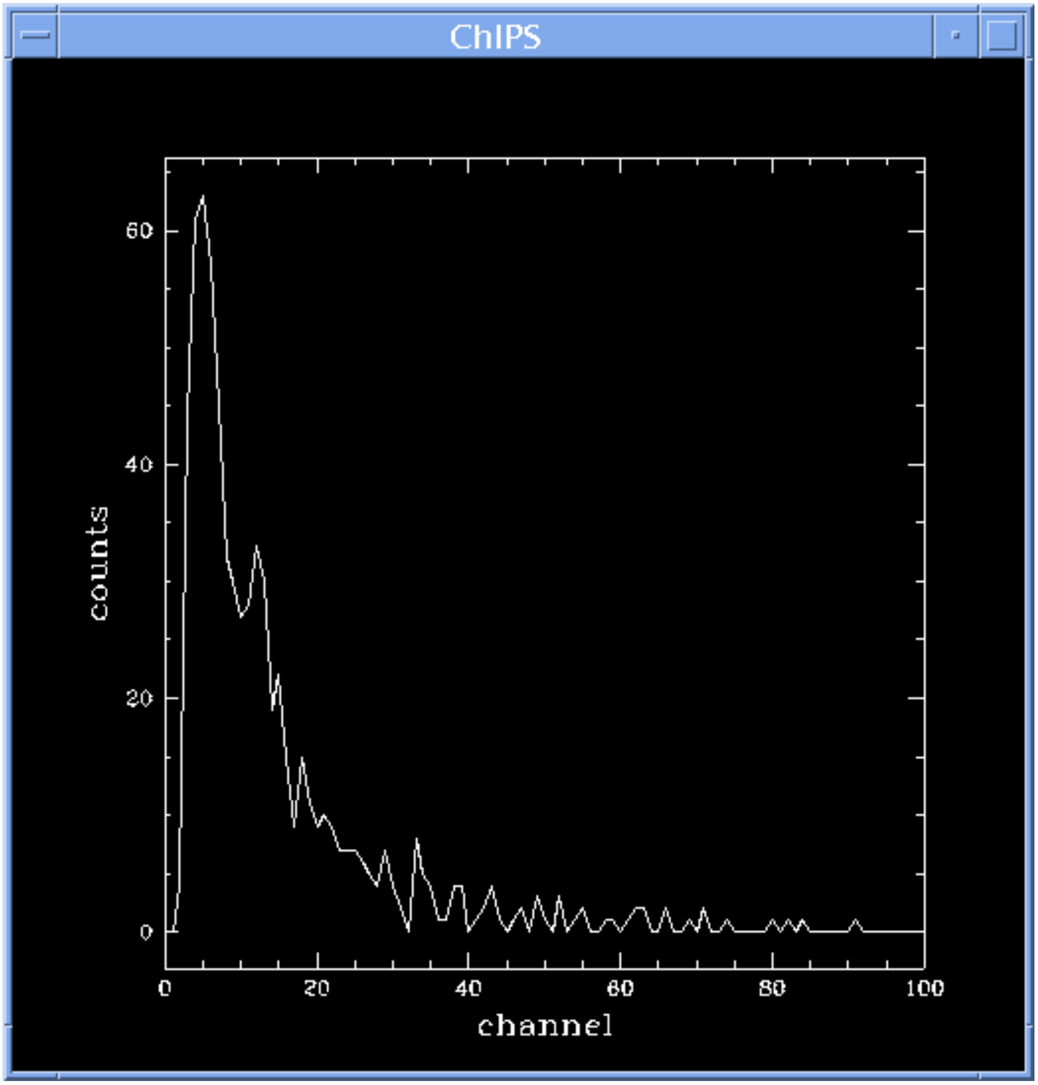


Image 7: Composite spectrum of 14 sources

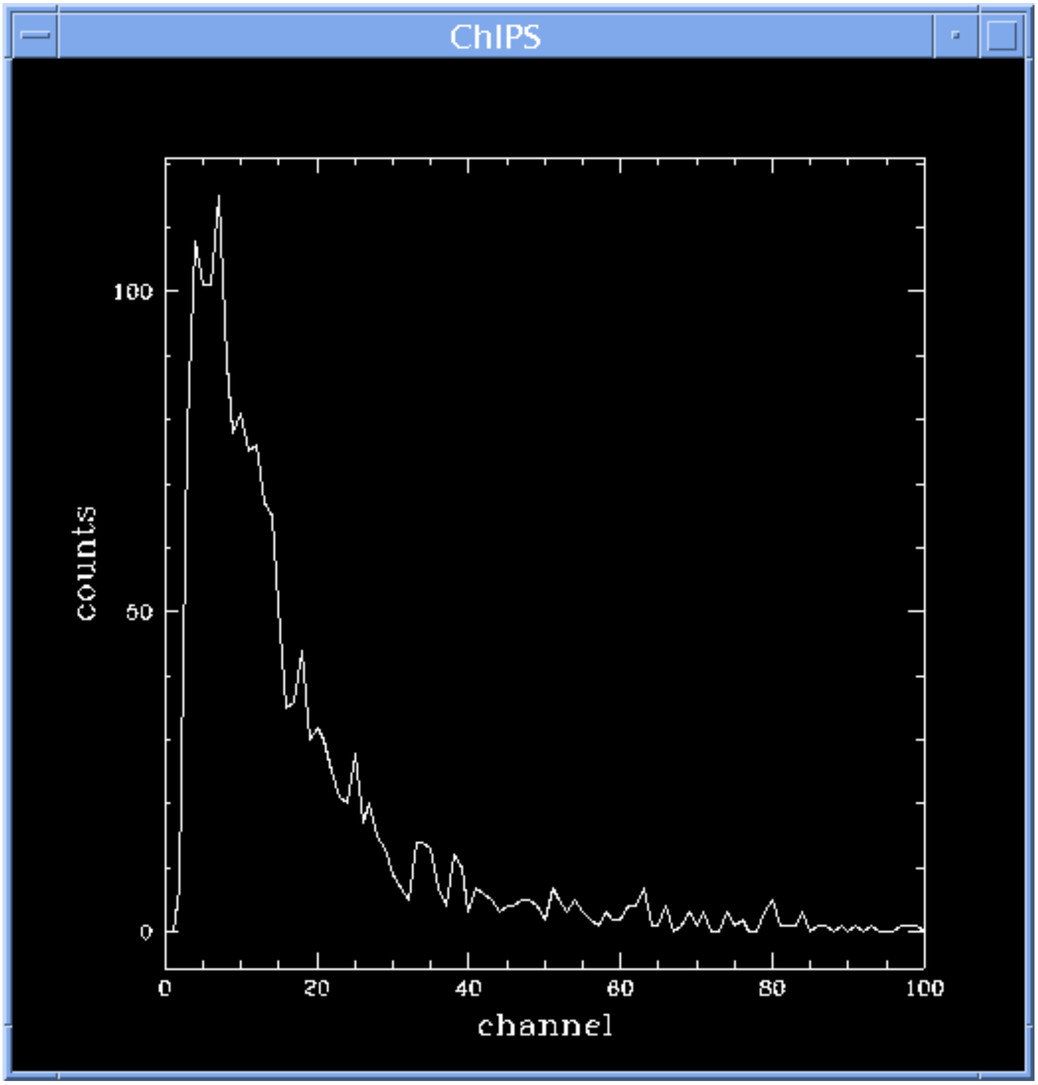
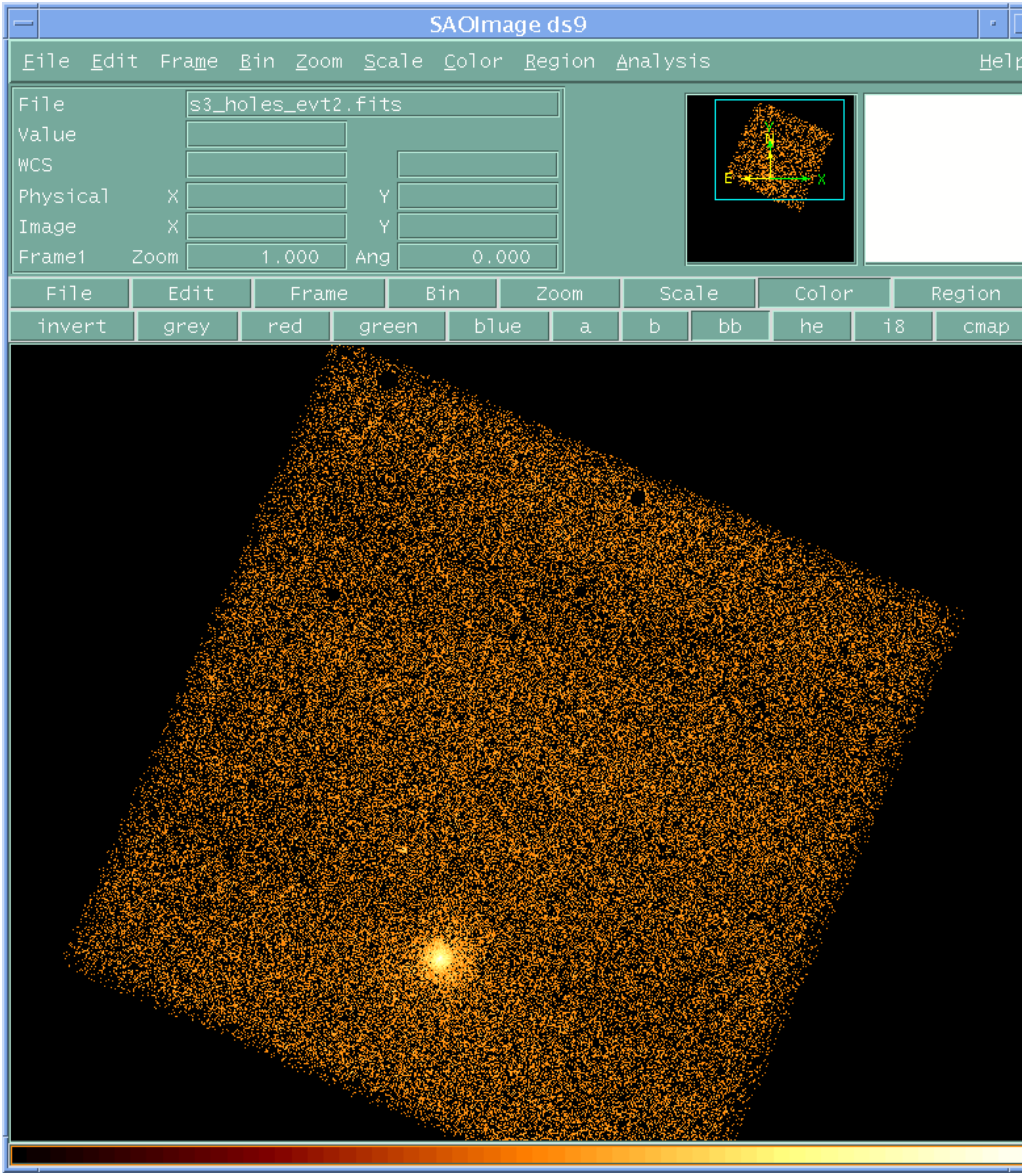


Image 8: Event file with sources removed



File s3_holes_evt2.fits

Value

WCS

Physical X Y

Image X Y

Frame1 Zoom 1.000 Ang 0.000

SAOImage ds9
File Edit Frame Bin Zoom Scale Color Region Analysis Help
File Edit Frame Bin Zoom Scale Color Region
invert grey red green blue a b bb he i8 cmap
Y
N
E
X

Image 8: Event file with sources removed