URL: http://cxc.harvard.edu/ciao3.4/get_src_region.html
Last modified: December 2006

AHELP for CIAO 3.4

get_src_region

Context: tools

Jump to: Description Examples Parameters Bugs See Also

Synopsis

Outputs regions that have counts higher than background

Syntax

```
get_src_region infile outfile [binning] [sigma_factor] [niter] [kernel]
[clobber] [verbose]
```

Description

'get_src_region' determines a background threshold value from an image or event file and outputs regions with counts higher than that threshold.

If the input is an event file, the user must supply a Data Model filter so that get_src_region can create a virtual image; see "ahelp dmsyntax" for more information on DM filtering.

From the bins of the image, the average and standard deviation of background counts are determined. The tool can be made to iterate, excluding pixels more than N−sigma from the mean. The background threshold value is set to be: average background counts + standard deviation of background counts * sigma_factor. sigma_factor is a user input parameter (default=5).

Any region that has counts higher than the background threshold value is classified as source region. A list of source "box" regions will be written to the output file.

* The Algorithm

The following algorithm is used to determine the average and standard deviation of background counts:

1) Using the binning specification, create a histogram of the pixel values in a 2−D image.

2) Sum up the number of counts in the bins of the input image.

3) Divide the sum by the total number of bins in the image to get the average number of counts.

4) Calculate the standard deviation.

5) Set min_value to:

(int)(average number of counts − sigma_factor * standard deviation + 0.5)

(Note: sigma_factor is a user input parameter and its default value is 5) If min_value is smaller than 0, reset it to 0.

6) Set max_value to:

(int)(average number of counts + sigma_factor * standard deviation + 0.5)

If max_value > maximum counts in the bins, set max_value to maximum counts in the bins.

7) If niter=1 (i.e number of iteration, a user input parameter), then we are done calculating the average and standard deviation of the background counts. The average number of counts obtained in step 3) will be considered as the average background counts, while the standard deviation obtained in step 4) will be considered as the standard deviation of the background counts. If niter > 1, go to step 8).

8) Recalculate the average number of counts but this time, only include the bins that have counts greater than or equal to min_value and counts smaller than or equal to max_value. Repeat steps 4 to 6 to obtain a new standard deviation, new min_value and new max_value.

9) Repeat step 8 until the new min_value is equal to the previous min_value and the new max_value is equal to the previous max_value, or when the number of iterations we did is equal to niter. The average background counts will be the last calculated average number of counts, and the standard deviation of background counts will be the last calculated standard deviation.

Example 1

```
get_src_region infile="in_evt.fits[EVENTS][bin x=::14,y=::14]"
outfile=out.fits
```

Run get_src_region on an event file with bin factor equal to 14 and without setting the x or y range.

Example 2

```
get_src_region infile="in_evt.fits[EVENTS][bin x=1:1000:10,bin
y=1:1000:10] outfile=out.fits binning=2
```

Run get_src_region on an event file with x,y range set to 1−1000, and with bin factor set to 10. Create a histogram with a bin size of two from the lowest pixel value to the highest pixel value; use this histogram to calculate the weighted average and standard deviation.

Example 3

```
get_src_region infile=in_img.fits outfile=out.fits
```

Run get_src_region on an image file.

Example 4

```
get_src_region infile=in_img2.fits outfile=out.fits
binning=0.5:120.5:10 sigma=2 niter=2 clobber=yes
```

Run get_src_region on an non−integer image file.

Parameters

name	type	ftype	def	min	max	reqd	autoname
infile	string	input				yes	
outfile	string	output				yes	yes
binning	string		1				
sigma_factor	integer		5				
niter	integer		5				
kernel	string		default				
clobber	boolean		no				
verbose	integer		0	0	5		

Detailed Parameter Descriptions

Parameter=infile `(string required filetype=input)`

Input file

The input file can be an image file, or an event file. If the input is an event file, the user should specify the extension name or extension number in which the data is in, and also specify the x, y range and bin factor for binning the event file into an image. eg. "infile[EVENTS][bin x=1:2000:14,y=1:2000:14]" Where 1:2000 is the x and y range for binning and 14 is the bin factor. Note that the x, y range is an optional parameter, if the user doesn't want to specify the x, y range, he can do: "infile[EVENTS][bin x=::14,y=::14]" If the user doesn't specify the extension name or extension number, the 1st extension that has data will be opened. Currently the program will bin the event file into an image in x and y sky coordinate only. However, if the input file is an image file, there is no need to give the extension name, x, y range and the bin factor.

Parameter=outfile `(string required filetype=output autoname=yes)`

Output file.

The output file contains a list of source regions in physical x and y sky coordinate.

If auto−naming is used (i.e. input to the outfile parameter ends with "."), the output file will have the suffix "_src_region".

Parameter=binning (`string default=1`)

Histogram binning factor

The histogram binning specification is of the form [a]:[b][:c] | c, where a is the low bin, b is the high bin, and c is the bin size (default=::1). Able to accept either integers or floats.

Parameter=sigma_factor (`integer default=5`)

Sigma factor

The sigma factor is used to set the background threshold value:

```
background threshold = average background counts + sigma_factor * standard deviation of background counts
```

Parameter=niter (`integer default=5`)

Number of iterations for determining the average background counts.

Parameter=kernel (`string default=default`)

Data Model creation/copy kernel.

It allows the user to specify the format of the output file (eg. FITS or IRAF file). Currently, the tool can only create a FITS file.

Parameter=clobber (`boolean default=no`)

Remove output if it exists?

Used to specify whether or not to clobber existing file that has the same name as the specified output file (only DM dataset will be clobbered).

Parameter=verbose (`integer default=0 min=0 max=5`)

The tool chatter level

Verbose can be from 0 to 5, generating different amounts of debugging output.

CHANGES FOR CIAO 3.2

get_src_region no longer excludes zero−valued pixels by default in calculating background mean and sigma. That means that for images dominated by off−detector area, the mean and sigma on iteration 1 will be << 1. Further iterations will throw out all non−zero pixels before calculating mean and sigma, leading to a nonsense result (sigma=0, mean=NaN).

The correct thing to do in these cases is to run the program with a restricted subspace that eliminates the off−detector area. You can determine the approximate region using ds9, for example. Using a command such as:

```
unix% get_src_region \
      infile=image.fits"[sky=rotbox(4089.3704,3773.4914,5199.7363,1006.885,306.01803)]" \
      outfile=get_src_region_1.fits sigma_factor='5' niter='5' \
      kernel='default' clobber='yes' verbose='5' mode='ql'
```

yields reasonable results (although not exactly the same as in CIAO 3.1, since the mean and sigma will still be different).

CHANGES IN CIAO 3.0

get_src_region now supports non−integer input images.

Bugs

See the bugs page for this tool on the CIAO website for an up−to−date listing of known bugs.

See Also

concept
: subspace

dm
: dmregions

tools
: dither_region, dmcontour, dmimg2jpg, dmmakereg, reproject_events, tg_create_mask

---