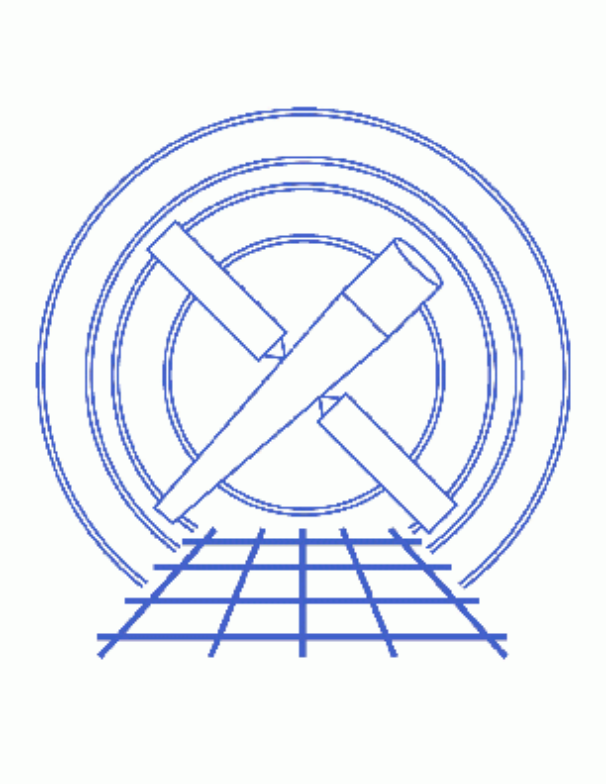
Obtain Grating Spectra from LETG/HRC−I Data



CIAO 3.4 Science Threads

Table of Contents

- ***Data Preparation***
- ***Get Started***
- ***Generate A New Level=1.5 Event File***
    1. Get position of zero−order image (tgdetect)
    2. Get region mask (tg_create_mask)
    3. Run tg_resolve_events
- ***Generate a New Level=2 Event File***
    1. Filter on status (dmcopy)
    2. Apply GTI filter (dmcopy)
- ***Extract a Grating Spectrum (tgextract)***
- ***Summary***
- ***Parameter files:***
    - ♦ tgdetect
    - ♦ tg_create_mask
    - ♦ tg_resolve_events
    - ♦ tgextract
- ***History***
- ***Images***
    - ♦ Image with region file overlaid

Obtain Grating Spectra from LETG/HRC−I Data

CIAO 3.4 Science Threads

Overview

Last Update: 1 Dec 2006 − updated for CIAO 3.4: change to wording of `tgdetect`/`dmcopy` warning

Synopsis:

Generate a new PHA2 spectrum file for any LETG/HRCS−I grating observation.

Purpose:

To ensure that consistent calibration is used throughout the analysis.

Read this thread if:

you are working with an LETG/HRC−I dataset.

Related Links:

- Analysis Guide for Chandra High Resolution Spectroscopy: an in−depth discussion of grating analysis.

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

Data Preparation

This analysis thread starts with the level 1 event file. Before beginning, users may wish to reprocess the data to create a new event file with the most recent calibration applied. Instructions on how to reprocess your data are available in the HRC Data Preparation Analysis Guide.

Get Started

Sample ObsID used: 1801 (LETG/HRC−I, PKS2155−304)

File types needed: evt1; flt1; asol1

In this thread, we assume that all relevant files are in the same working directory.

Generate A New Level=1.5 Event File

1. Get position of zero−order image (tgdetect)

To find the zero−order location, the tool `tgdetect` is run:

```
unix% punlearn tgdetect
unix% pset tgdetect infile=hrcf01801_000N003_evt1.fits
unix% pset tgdetect outfile=hrc_1801_evt1_src1a.fits
unix% tgdetect
Input L1 event file (hrcf01801_000N003_evt1.fits):
Input source position(s) file from previous OBI or NONE (NONE):
Output source position(s) file name (hrc_1801_evt1_src1a.fits):

# DMCOPY (CIAO 3.4): Bad data type in filter string formatting
```

The warning may be ignored; it is due to a minor bug in the Data Model and does not affect the output of `tgdetect`.

The contents of the parameter file may be checked using plist tgdetect.

The source list may be viewed over the event file using ds9:

```
unix% ds9 hrcf01801_000N003_evt1.fits &
```

Overlay the source list: Region−> Load Regions−> hrc_1801_evt1_src1a.fits[SRCLIST].

If the zero order of the source is outside of the default search area (e.g. far from the aimpoint), `tgdetect` will not find it. ***If this problem affects your data, it will be obvious when the source list is displayed on the event file.*** In this case, run the Correcting a Misplaced Zero−order Source Position thread to identify the correct source position.

2. Get region mask (tg_create_mask)

The location of the spectrum needs to be found next, via the tool `tg_create_mask`, which creates a region file that will be used to mask the image:

```
unix% punlearn tg_create_mask
unix% pset tg_create_mask infile=hrcf01801_000N003_evt1.fits
unix% pset tg_create_mask outfile=hrc_1801_evt1_L1a.fits
unix% pset tg_create_mask input_pos_tab=hrc_1801_evt1_src1a.fits
unix% tg_create_mask
Input event file or stack (hrcf01801_000N003_evt1.fits):
Output region file or stack (hrc_1801_evt1_L1a.fits):
Input table with zero order positions or stack (hrc_1801_evt1_src1a.fits):
Observed grating type (header_value|HETG|HEG|MEG|LETG) (HETG|HEG|MEG|LETG|header_value|HEADER_VALUE) (head
```

The contents of the parameter file may be checked using plist tg_create_mask.

3. Run tg_resolve_events

The tool `tg_resolve_events` is now used to assign grating events to spectral orders:

```
unix% punlearn tg_resolve_events
unix% pset tg_resolve_events infile=hrcf01801_000N003_evt1.fits
unix% pset tg_resolve_events outfile=hrc_1801_evt1a.fits
unix% pset tg_resolve_events regionfile=hrc_1801_evt1_L1a.fits
unix% pset tg_resolve_events acaofffile=pcadf082337011N002_asol1.fits
unix% pset tg_resolve_events eventdef=")stdlev1_HRC"
unix% pset tg_resolve_events osipfile=none
unix% tg_resolve_events
Input event file or stack (hrcf01801_000N003_evt1.fits):
Input region file or stack (hrc_1801_evt1_L1a.fits):
Output event file or stack (hrc_1801_evt1a.fits):
Input aspect offset file (pcadf082337011N002_asol1.fits):
```

It is important to note several things here:

- In some cases there will be more than one asol1.fits file for an observation. ***All*** the files must be input to the `acaofffile` parameter ***in chronological order*** (the time is in the filename, so "`ls`" lists them in order), either as a comma−separated list or as a stack (see stack for more information).
- The unusual syntax of the `eventdef` parameter; the tool will not access the predefined string if the leading ")" is missing.

The contents of the parameter file may be checked using plist tg_resolve_events.

The created region file, which has been appended to the event file as a block, may be viewed over the event file using ds9:

```
unix% ds9 hrc_1801_evt1a.fits &
```

Overlay the region file that was created by `tg_create_mask` (Region−> Load Regions−> hrc_1801_evt1a.fits[REGION]) and you should see something like Figure 1 .

Generate a New Level=2 Event File

1. Filter on status (dmcopy)

Next we apply the status filter that is specific to HRC−I observations; a value of 0 demands that the bit be flagged as "good", a value of x indicates that either status (0/1) is acceptable:

```
unix% punlearn dmcopy
unix% dmcopy "hrc_1801_evt1a.fits[status=xxxxxx00xxxx0xxx0000x000x0000000]" \
      hrc_1801_flt1_evt1a.fits opt=all
```

2. Apply GTI filter (dmcopy)

Finally, the Good Time Intervals (GTIs) supplied by the pipeline are applied:

```
unix% punlearn dmcopy
unix% dmcopy \
      "hrc_1801_flt1_evt1a.fits[EVENTS][@hrcf01801_000N003_std_flt1.fits]" \
      hrc_1801_evt2.fits opt=all
```

Be sure to include the ***@ symbol*** in the filter expression; the command will not be executed properly if it is omitted.

Extract a Grating Spectrum (tgextract)

The CIAO tool `tgextract` produces a PHA2 spectrum file from the level=2 data file:

```
unix% punlearn tgextract
unix% pset tgextract infile=hrc_1801_evt2.fits
unix% pset tgextract outfile=hrc_1801_pha2.fits
unix% tgextract
Input event file (output event file from L1.5 processing) (hrc_1801_evt2.fits):
If typeII, enter full output file name or '.'; if typeI, enter output rootname (hrc_1801_pha2.fits):
Input ancillary response file name (none):
Input redistribution file name (none):
Source ID's to process: 'all', comma list, @file (all):
Grating parts to process: HETG, HEG, MEG, LETG, header_value (HETG|HEG|MEG|LETG|header_value) (header_valu
Grating diffraction orders to process: 'default', comma list, range list, @file (default):
Ouput file type: typeI (single spectrum) or typeII (multiple spectra) (pha_typeI|pha_typeII) (pha_typeII):
```

The contents of the parameter file may be checked using plist tgextract.

Summary

This thread is now complete; the PHA2 grating spectrum file is named hrc_1801_pha2.fits. You should now proceed to the Create Grating RMFs for HRC Observations thread.

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


##
## TGDETECT -- Create filter; run celldetect; narrow down detected
##             'zero order' source list; set source id's; match
##             sources to previous OBI source list.
##
## Note: if either "infile" or "OBI_srclist_file" are @lists, only
## the first item on the list is read in;  this tool only works on
## one set of input files;  if more than one file is listed,
## everything but the first are ignored.
##
        infile = hrcf01801_000N003_evt1.fits    Input L1 event file
OBI_srclist_file = NONE             Input source position(s) file from previous OBI or NONE
```

```
       outfile = hrc_1801_evt1_src1a.fits Output source position(s) file name
#
#   output file naming
#
     (temproot = )               Path and root file name to be given to temporary files
     (keeptemp = no)             Keep temporary files?
     (keepexit = no)             Keep exit status file?
#
#
#
     (zo_pos_x = default)        Center GZO filter sky X position (default=pixel(ra_nom))
     (zo_pos_y = default)        Center GZO filter sky Y position (default=pixel(dec_nom))
 (zo_sz_filt_x = default)        Size of GZO filter in X pixels (ACIS=400; HRC=1800)
 (zo_sz_filt_y = default)        Size of GZO filter in Y pixels (ACIS=400; HRC=1800)
   (snr_thresh = 40)             SNR threshold to select the detected sources
#
#   celldetect parameters
#
       (expstk = none)           list of exposure map files
       (thresh = )celldetect.thresh -> 3) celldetect source threshold
     (ellsigma = 3.0)            Size of output source ellipses (in sigmas)
     (expratio = 0)              cutoff ratio for source cell exposure variation
    (findpeaks = yes)            find local peaks for celldetect
(celldetect_log = )celldetect.log -> no) make a celldetect log file?
     (psftable = )celldetect.psftable -> /soft/ciao/data/psfsize20010416.fits) table of PSF size dat
    (fixedcell = 15)             celldetect fixed cell size to use
(fixedcell_cc_mode = 15)             celldetect fixed cell size to use for CC mode ACIS data
      (bkgfile = none)           background file, for celldetect
     (bkgvalue = )celldetect.bkgvalue -> 0) background count/pixel, for celldetect
  (bkgerrvalue = )celldetect.bkgerrvalue -> 0) background error, for celldetect
        (eband = )celldetect.eband -> 1.4967) energy band, for celldetect
      (eenergy = )celldetect.eenergy -> 0.8) encircled energy of PSF, for celldetect
      (snrfile = none)           celldetect snr output file (for convolution only)
     (convolve = )celldetect.convolve -> no) use convolutions for celldetect
      (xoffset = INDEF)          celldetect offset of x axis from optical axis
      (yoffset = INDEF)          celldetect offset of y axis from optical axis
     (cellfile = none)           output cell size image file
     (centroid = yes)            compute source centroids in celldetection?
#
#   tgidselectsrc parameters
#
(snr_ratio_limit = )tgidselectsrc.snr_ratio_limit -> 1) Value of SNR ratio to use as lower limit
     (setsrcid = )tgidselectsrc.setsrcid -> yes) Set src ids in output file?
#
#   tgmatchsrc parameters
#
(max_separation = )tgmatchsrc.max_separation -> 3) Maximum allowed separation (arcsec) for sources t
#
#
      (clobber = no)             OK to overwrite existing output file(s)?
      (verbose = 0)              Verbosity level (0 = no display)
         (mode = ql)
```

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


##
## TG_CREATE_MASK -- Calculates the mask regions of the grating arms
```

```
##    for AXAF flight L1 grating data files.  The output is a region
##    file(s) in sky coordinates.
##
        infile = hrcf01801_000N003_evt1.fits    Input event file or stack
       outfile = hrc_1801_evt1_L1a.fits Output region file or stack
 input_pos_tab = hrc_1801_evt1_src1a.fits Input table with zero order positions or stack
   grating_obs = header_value     Observed grating type (header_value|HETG|HEG|MEG|LETG)
     sA_zero_x = 1                Source A − x position of zero order
     sA_zero_y = 1                Source A − y position of zero order
     sB_zero_x = 1                Source B − x position of zero order
     sB_zero_y = 1                Source B − y position of zero order
     sC_zero_x = 1                Source C − x position of zero order
     sC_zero_y = 1                Source C − y position of zero order
     sD_zero_x = 1                Source D − x position of zero order
     sD_zero_y = 1                Source D − y position of zero order
     sE_zero_x = 1                Source E − x position of zero order
     sE_zero_y = 1                Source E − y position of zero order
     sF_zero_x = 1                Source F − x position of zero order
     sF_zero_y = 1                Source F − y position of zero order
     sG_zero_x = 1                Source G − x position of zero order
     sG_zero_y = 1                Source G − y position of zero order
     sH_zero_x = 1                Source H − x position of zero order
     sH_zero_y = 1                Source H − y position of zero order
     sI_zero_x = 1                Source I − x position of zero order
     sI_zero_y = 1                Source I − y position of zero order
     sJ_zero_x = 1                Source J − x position of zero order
     sJ_zero_y = 1                Source J − y position of zero order
(input_psf_tab = CALDB)           Calibration file with mirror psf vs off−axis angle
     (detector = header_value)    Detector type: ACIS | HRC−I | HRC−S | header_value
(radius_factor_zero = 50)              A scale factor which multiplies the app. calculation of the one−sigm
(width_factor_hetg = 35)              A scale factor which multiplies the one−sigma width of the heg/meg ma
(width_factor_letg = 40)              A scale factor which multiplies the one−sigma width of the letg mask
(r_astig_max_hetg = 0.5600000000000001) Max grating r coord (deg, along the dispersion) for HETG astigmatis
(r_astig_max_letg = 1.1)             Max grating r coord (deg, along the dispersion) for LETG astigmatism c
(r_mask_max_hetg = 0.992)           Max grating r coord (deg) for HETG mask (to support offset pointing)
(r_mask_max_letg = 2.1)             Max grating r coordinate (deg) for LETG mask (to support offset pointin
# ------------------------------------------------------------------------
# The parameters below are to be set ONLY if the user wants to use their
# own grating mask sizes instead of having the masks automatically generated.
# Only ONE input file, with up to 10 soures, can be processed using the user
# params.  @ lists of multiple files can only be done with automated mask
# processing, or by running each file individually with hand set mask sizes.
# To start, you MUST set the following parameters:
#
# > pset tg_create_mask use_user_pars=yes last_source_toread=[letter A −> J]
#
# The parameter last_source_toread should be set to the last source letter
# for which you will enter parameters.  If you want to input 2 sources
# (regardless of their source id's), the last_source_toread=B. Sections
# A −> J are for (upto) 10 user specified sources.  In each sections,
# each source must have an ID, a zero order center position specified,
# as well as the grating mask width(s).  An example with 2 HETG sources,
# with src_id's 6 and 3:
#
# > pset tg_create_mask use_user_pars=yes last_source_toread=B
# > pset tg_create_mask sA_id=6 sA_zero_x=4762.34 sA_zero_y=2344.29
# > pset tg_create_mask sA_zero_rad=35 sA_width_heg=25 sA_width_meg=28
# > pset tg_create_mask sB_id=3 sB_zero_x=4063.54 sB_zero_y=6346.62
# > pset tg_create_mask sB_zero_rad=45 sB_width_heg=50 sB_width_meg=75
#              (units are all in sky pixels)
#
# NOTE: for Continuous Clocking data (CC mode), the HETG mask does not
```

```
# require the s#_width_heg, since the meg mask will encompase the entire
# data set.  HEG event processing in CC mode is done using the next
# tool tg_resolve_events.
# ------------------------------------------------------------------------
(use_user_pars = no)              Use the user defined mask parameters below: yes or no?
(last_source_toread = A)               Last source name to be read; character A->J.
# ------------------------------------------------------------------------
#                       Source A parameters
# ------------------------------------------------------------------------
        (sA_id = 1)               Source A − source id number
  (sA_zero_rad = )                Source A − radius of zero order mask
 (sA_width_heg = )                Source A − width of heg mask in sky pixels
 (sA_width_meg = )                Source A − width of meg mask in sky pixels
 (sA_width_leg = )                Source A − width of leg mask in sky pixels
# ------------------------------------------------------------------------
#                       Source B parameters
# ------------------------------------------------------------------------
..(through Source J)..
      (geompar = geom)            Parameter file for Pixlib Geometry files
      (verbose = 0)               Verbose level: 0 − no output, 5 − max verbosity
      (clobber = no)              Clobber existing outfile?
         (mode = ql)
```

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


#-------------------------------------------------------------------------
#
#   tg_resolve_events.par: Parameter file for the tg_resolve_events program
#
#-------------------------------------------------------------------------
        infile = hrcf01801_000N003_evt1.fits    Input event file or stack
       outfile = hrc_1801_evt1a.fits   Output event file or stack
    regionfile = hrc_1801_evt1_L1a.fits Input region file or stack
    acaofffile = pcadf082337011N002_asol1.fits Input aspect offset file
(alignmentfile = )acaofffile -> pcadf082337011N002_asol1.fits) Input sim offset file
      (logfile = stdout)          Output log (NONE|<filename>|stdout)
# The osipfile contains position dependent energy limits based on
# the CCD resolution, used for order-sorting.
# A value of "NONE" means that the file will not be used, and
# that the parameters, osort_hi and osort_lo will be used.
     (osipfile = none)            Lookup table for order resolving (for acis data only)
#sort_hi, osort_lo specify fractional deviations from the integer
#order which will be included in order-sorting via CCD ENERGY values (PHA).
#eg. osort_lo-0.3, osort_hi=0.2 means that photons with real-valued
#orders between 0.7 < order <= 1.2 will be included in first order,
#1.7 < order <= 2.2 will be second order, etc.
     (osort_lo = 0.3)             Order-sorting lower bound fraction; order > m − osort_lo
     (osort_hi = 0.3)             Order-sorting high bound fraction; order <= m + osort_hi
  (grating_obs = header_value)    Observed grating type (header_value|HETG|HEG|MEG|LETG)
     (detector = header_value)    Detector type: ACIS | HRC-I | HRC-S | header_value
(energy_lo_adj = 1.0)             Lower Energy limit factor
(energy_hi_adj = 1.0)             Upper Energy limit factor
  (time_offset = 0)               Offset to add to event time to synch w/ alignment data
    (rand_seed = 1)               Random seed (for pixlib), 0 = use time dependent seed
(rand_pix_size = 0.0)             pixel randomization width (-size..+size), 0.0 = no randomization
     (eventdef = )stdlev1_HRC -> {d:time,f:rd,s:chip,l:tdet,f:det,f:sky,s:chip_id,s:pha,s:pi,s:tg_m,
f:tg_mlam,s:tg_srcid,s:tg_part,s:tg_smap,x:status}) Output format definition
```

```
      (stdlev1 = )eventdef -> {d:time,f:rd,s:chip,l:tdet,f:det,f:sky,s:chip_id,s:pha,s:pi,s:tg_m,f:tg_lam,
f:tg_mlam,s:tg_srcid,s:tg_part,s:tg_smap,x:status})
 (stdlev1_ACIS = {d:time,i:expno,f:rd,s:chip,s:tdet,f:det,f:sky,s:ccd_id,l:pha,s:pi,f:energy,s:grade,
s:fltgrade,s:node_id,s:tg_m,f:tg_lam,f:tg_mlam,s:tg_srcid,s:tg_part,s:tg_smap,x:status})
ACIS event format definition string
  (stdlev1_HRC = {d:time,f:rd,s:chip,l:tdet,f:det,f:sky,s:chip_id,s:pha,s:pi,s:tg_m,f:tg_lam,
f:tg_mlam,s:tg_srcid,s:tg_part,s:tg_smap,x:status}) HRC event format definition string
# ------------------------------------------------------------------------
      (geompar = geom)           Parameter file for Pixlib Geometry files
      (verbose = 0)              Verbosity level of detail (0=none, 5=most)
      (clobber = no)             Clobber outfile if it already exists?
         (mode = ql)
```

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


##
## TGEXTRACT -- create 1D spectrum(a) table file(s) from the
##              L1.5 output event list
##
        infile = hrc_1801_evt2.fits    Input event file (output event file from L1.5 processing)
       outfile = hrc_1801_pha2.fits    If typeII, enter full output file name or '.'; if typeI, enter outpu
#
# tg_srcid_list parameter explanation...
#  - "all" will process all the sources id's found in the event list
#  - a comma list is a comma separated string list of all the
#    sources to process, ie:
#       "1,2,5,7"
#  - @file is a pointer to an ascii file which contains a comma
#    separated list of the id's to process
#
 tg_srcid_list = all             Source ID's to process: 'all', comma list, @file
  tg_part_list = header_value    Grating parts to process: HETG, HEG, MEG, LETG, header_value
#
# tg_order_list parameter explanation...
#  - "default" is set to process the following:
#       if ACIS:  1, 2, 3, -1, -2, -3
#       if HRC:   -1, 1
#  - a comma list is a comma separated string list of the orders
#    the user wants to process, ie:
#       "-5, -1, 1, 3"
#  - a range list sets the min and max of the orders to process;
#    all the orders in between, will be processed, ie:
#       "-1..5"  will do orders from -1 to +5th order
#    a range list can be mixed with comma separated list
#  - @file is a pointer to an ascii file which contains a comma
#    separated list and/or range list of the orders to process
#
 tg_order_list = default         Grating diffraction orders to process: 'default', comma list, range list,
      ancrfile = none            Input ancillary response file name
      respfile = none            Input redistribution file name
  outfile_type = pha_typeII      Ouput file type: typeI (single spectrum) or typeII (multiple spectra)
(inregion_file = none)           Input region file.
     (backfile = none)           Input background file name
        (rowid = )               If rowid column is to be filled in, enter name here
    (bin_units = angstrom)       Bin units (for bin parameters below): angstrom, eV, keV
  (min_bin_leg = compute)        Minimum dispersion coordinate for LEG, or 'compute'
  (max_bin_leg = compute)        Maximum dispersion coordinate for LEG, or 'compute'
```

```
 (bin_size_leg = compute)        Bin size for binning LEG spectra, or 'compute'
 (num_bins_leg = compute)        Number of bins for the output LEG spectra, 'compute'
  (min_bin_meg = compute)        Minimum dispersion coordinate for MEG, or 'compute'
  (max_bin_meg = compute)        Maximum dispersion coordinate for MEG, or 'compute'
 (bin_size_meg = compute)        Bin size for binning MEG spectra, or 'compute'
 (num_bins_meg = compute)        Number of bins for the output MEG spectra, or 'compute'
  (min_bin_heg = compute)        Minimum dispersion coordinate for HEG, or 'compute'
  (max_bin_heg = compute)        Maximum dispersion coordinate for HEG, or 'compute'
 (bin_size_heg = compute)        Bin size for binning HEG spectra, or 'compute'
 (num_bins_heg = compute)        Number of bins for the output HEG spectra, 'compute'
     (min_tg_d = default)        Minimum tg_d range to include in histogram, or use 'default'
     (max_tg_d = default)        Maximum tg_d range to include in histogram, or use 'default'
(extract_background = yes)            Extract the local background spectrum?
(min_upbkg_tg_d = default)         Minimum value of tg_d for the background up spectrum.
(max_upbkg_tg_d = default)         Maximum value of tg_d for the background up spectrum.
(min_downbkg_tg_d = default)         Minimum value of tg_d for the background down spectrum.
(max_downbkg_tg_d = default)         Maximum value of tg_d for the background down spectrum.
      (geompar = geom)          Parameter file for Pixlib Geometry files
      (clobber = no)            OK to overwrite existing output file(s)?
      (verbose = 0)             Verbosity level (0 = no display)
         (mode = ql)
```

History

16 Dec 2004	updated for CIAO 3.2: minor changes to parameter files
05 Dec 2005	updated for CIAO 3.3: output filenames include ObsID; parameter file change (`kernel` parameter removed from all "tg" tools)
05 Jan 2006	created Data Preparation section
01 Dec 2006	updated for CIAO 3.4: change to wording of `tgdetect`/`dmcopy` warning

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

Image 1: Image with region file overlaid