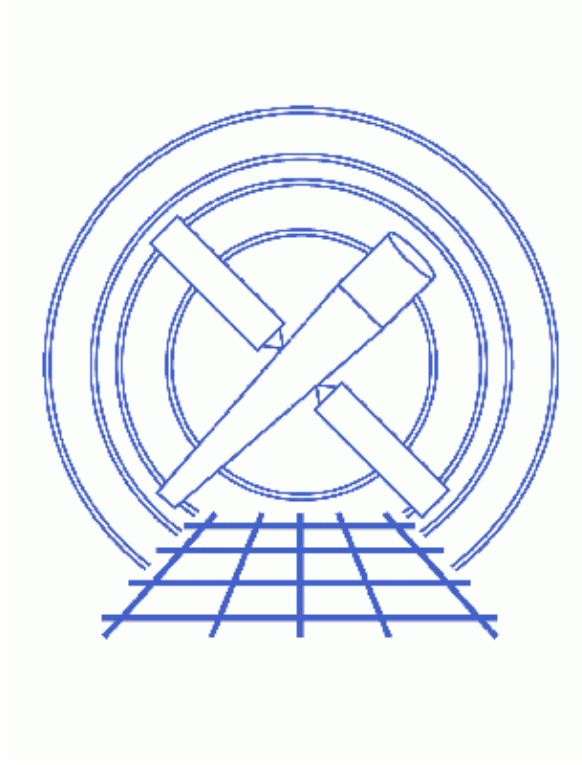


Using specextract to Extract ACIS Spectra and Response Files



CIAO 3.4 Science Threads

Table of Contents

- **How specextract Works**
 - ◆ Creating RMFs: mkrmf vs mkacisrmf
 - ◆ The ACIS dead area correction
- **Get Started**
 - ◆ Choosing a background file
- **Single Spectrum**
 - ◆ Build Source and Background Regions
 - ◆ Run specextract
 - ◆ Examining the Output Files
- **Extracting Multiple Spectra**
 - ◆ Build Source Regions and Background Regions
 - ◆ Run specextract
 - ◆ Examining the Output Files
- **More Information on the Output Files**
 - ◆ Which RMF tool was used?
 - ◆ Header keywords
- **Running mkacisrmf Independently**
- **Fitting**
- **Caveats**
 - ◆ Using the ACIS "Blank–Sky" Background Files
 - ◆ Analysis at the edges of ACIS CCDs
- **Parameter files:**
 - ◆ specextract
 - ◆ specextract
- **History**
- **Images**
 - ◆ Single Spectrum: extraction regions
 - ◆ Multiple Spectra: extraction regions

Using specextract to Extract ACIS Spectra and Response Files

CIAO 3.4 Science Threads

Overview

Last Update: 31 Mar 2008 – updated for CALDB 3.4.3: use `mkacisrmf` for –110 BI chips if TGAIN calibration has been applied

Synopsis:

Recommended use of the `specextract` script: It has been determined that the `mkwarf` tool, which is used for ARF generation by `specextract`, may not produce accurate results for point sources. The `mkarf` tool, used by `psextract`, is preferred for point source extraction. `specextract` should be used for the analysis of extended sources only. Users working with point sources should instead use the [psextract script](#). ***Analysis of point sources that was done with `mkwarf/specextract` should be redone with `mkarf/psextract` for the most accurate results***, e.g. taking bad columns into account.

`specextract`, a new script for creating ACIS spectra for extended sources, was released in CIAO 3.3. It allows the user to create source and background PHA or PI spectra and the associated ARF and RMF files. `specextract` can take a stack of input files and generate many spectra in one run of the script. Read [the help file](#) for full details on how the script works.

This script is designed to replace the `acisspec` script currently distributed in [the CIAO Scripts package](#). The major improvement over its predecessor is the ability to determine when `mkacisrmf` should be used in place of `mkrmf` (see the ["Creating RMFs: `mkrmf` vs. `mkacisrmf`"](#) section of this thread). ***Users are encouraged to start using `specextract` in place of `acisspec` for new extended source analyses***. Note that `specextract` does not currently offer the option to coadd spectra as `acisspec` does.

Purpose:

To generate source and background PI (PHA) spectra of an extended ACIS source and build the proper RMFs and ARFs.

Read this thread if:

you are working with an ACIS observation of an extended source, whether imaging or grating data.

Calibration Updates:

- **[CALDB v3.3.0](#) (18 Dec 2006):** The version 6 "phase 2" response file (`acisD2000-01-29p2_respN0006.fits`) was added to CALDB 3.3.0. Only calibration for the BI chips (S1, S3) has changed in this file; calibration for the FI chips is identical to the v5 file. The [How CIAO 3.4 and CALDB 3.3.0 Affect Your Analysis](#) section of the CIAO release notes explains how the file will affect your analysis.
- **[CALDB v3.2.1](#) (15 Dec 2005):** The gain for the back-illuminated (BI) chips – ACIS-S1 and S3 – for –120 GRADED mode observations has been upgraded. It is now possible to use `mkacisrmf` when

creating responses for these chips only in GRADED mode. See [How CALDB 3.2.1 Affects Your Analysis](#) for details.

- **CALDB v3.2.0 (21 Nov 2005):** The version 5 "phase 2" response file (`acisD2000-01-29p2_respN0005.fits`) was added to CALDB 3.2.0, released on 21 November 2005. Information on the changes in CALDB 3.2.0 is in the [How CIAO 3.3 and CALDB 3.2.0 Affect Your Analysis](#) section of the CIAO release notes.

Proceed to the [HTML](#) or [hardcopy \(PDF: \[A4\]\(#\) / \[letter\]\(#\)\)](#) version of the thread.

How specextract Works

specextract runs the `dmextract`, `mkwarf`, `mkrmf` or `mkacisrmf` (see the "[Creating RMFs: mkrmf vs. mkacisrmf](#)" section), `dmgroup` and `dmhedit` tools.

- `dmextract`: to extract source and (optionally) background spectra. This tool also creates the WMAP used as input to `mkwarf` and `mkacisrmf`; see [ahelp mkwarf](#) for details on why a WMAP is necessary.
- `mkwarf`: to create weighted ARF(s).
- `mkrmf` or `mkacisrmf`: to build the RMF(s); also see the "[Creating RMFs: mkrmf vs. mkacisrmf](#)" section.
- `dmgroup`: to group the source spectrum and/or background spectrum.
- `dmhedit`: to update the BACKFILE, RESPFILE and ANCRFILE keys in the source and background spectrum files.

Creating RMFs: mkrmf vs mkacisrmf

The tool `mkacisrmf` is used to create RMFs for:

- all -120 ACIS data taken in (V)FAINT mode that has the time-dependent gain adjustment and CTI correction applied
- -120 ACIS GRADED mode data on the back-illuminated chips (ACIS-S1 and S3) only
- -110 ACIS data taken on the back-illuminated chips (ACIS-S1 and S3) only

All new analyses with these types of data should be done with `mkacisrmf` instead of `mkrmf`. Although this script runs `mkrmf`, it can still be used to create the spectrum and ARF files for the data. Then follow the [Creating ACIS RMFs with mkacisrmf thread](#) to generate new RMFs.

It is important that the calibration applied to the event file is consistent with the RMF tool chosen, as explained in the "[Using Consistent Calibration](#)" section of the [why](#) topic. *If necessary, reprocess your data with the correct gain file before beginning this thread.*

How specextract chooses the RMF tool to use

specextract reads the source and background file header keywords to determine when the observation was taken and what calibration has been applied. The script then does a CALDB lookup to compare the calibration applied to the event file and the most recent file available in the CALDB. This means that the data needs to have been processed with very newest calibration, not just a "good enough" version; that is, even if the new calibration will have a minimal effect on your data, specextract requires that it be applied for the "use `mkacisrmf`" switch to be triggered.

If all the requirements are met, the script uses `mkacisrmf` to generate the source and background RMF(s) for the data. If the observation was not done at -120 or has not been reprocessed with the newest calibration available, `specextract` creates the RMF(s) with `mkrmf`. At higher verbosity, a message of this form will be printed to the screen:

```
Cannot use mkacisrmf because last applied gainfile does not match current
CALDB gainfile: acisD2000-01-29gain_ctiN0004.fits != acisD2000-01-29gain_ctiN0006.fits
Please re-run <your_filename> with acis_process_events if you wish to use mkacisrmf
```

Users who don't wish to reprocess can still use `specextract`, and run `mkacisrmf` afterwards to create new RMF files.

The ACIS dead area correction

There is a fractional area loss per unit time due to cosmic ray flux incident on the ACIS detector. Calibration to account for this ACIS "dead area" was included in CALDB 3.3.0 on 15 December 2006. The correction is non-zero for the 8 front-illuminated ACIS chips; the effect is not detectable for the BI chips, so the nominal calibration value is 0.0. The resulting chipy-dependent reduction in the EA will be approximately 2.2% at the readout, and 4.0% at the top of the chip. Refer to the [ACIS Dead Area Correction why topic](#) for technical details.

In CIAO 3.4, it is not possible to apply the ACIS dead area correction while running `specextract`. Users who wish to apply this calibration should rerun `mkwarf` independently to include the correction, e.g. as shown in [Example 3 of the `mkwarf` help file](#).

Get Started

Sample ObsIDs used: 869 (ACIS-S, ARP 220); 1842 (ACIS-I, G21.5-09); 1843 (ACIS-I, G21.5-09)

File types needed: `evt2`

Please ensure that you have set up `ardlib` to use the [bad pixel file for your observation](#) before following this thread.

One of the event files used in this thread – `acis_869_evt2.fits` – has been reprocessed with the newest calibration available by following the [Create a New Level=2 Event File thread](#).

Choosing a background file

There are several options when choosing the background file for use with `specextract`:

- define the background region from a source-free region of the same event file
- use a different event file when defining the background
- omit the background completely (i.e. leave the [bk`q`file](#) blank)

If you plan on using one of the "blank-sky" background files from the CALDB with `specextract`, read the [Using the ACIS "Blank-Sky" Background Files caveats](#) before continuing.

Single Spectrum

A simple example: extracting spectra with one source region and one background region.

Build Source and Background Regions


We need to define two regions, one for the source and another for the background. To do this, first display the file:

```
unix% ds9 acis_869_evt2.fits &
```

Refer to the [Using CIAO Regions](#) thread for information on creating region files. The files for this example look like:

```
unix% cat simple.reg
# Region file format: CIAO version 1.0
ellipse(4026,4138.9,50,40,0)

unix% cat bg_simple.reg
# Region file format: CIAO version 1.0
circle(3975,4233,20)
```

The regions are shown displayed on the event file in [Figure 1](#) ; the source is white and the background is green.

Make sure that you save the regions in CIAO format (Regions → File Format → CIAO) so that they are fully compatible with the analysis tools.

Run specextract

Input the event file with the appropriate region file for the source and background:

```
unix% punlearn specextract
unix% pset specextract infile="acis_869_evt2.fits[sky=region(simple.reg)]"
unix% pset specextract outroot=simple
unix% pset specextract bkgfile="acis_869_evt2.fits[sky=region(bg_simple.reg)]"
unix% pset specextract grouptype=NUM_CTS binspec=15
```

We choose to use the default grouping values: the source spectrum will be grouped to a minimum number of 15 counts per new channel and the background spectrum will be ungrouped. The [grouptype](#) and [binspec](#) parameters are used to specify the source spectrum grouping, and the [bkg_grouptype](#) and [bkg_binspec](#) parameters specify the background spectrum grouping.

Running the tool with `verbose=1` shows what it is doing:

```
unix% specextract verbose=1
Source events file(s) (acis_869_evt2.fits[sky=region(simple.reg)]):
Output directory path + root name for output files (simple):
Running: specextract

Checking initial status and initializing variables...

Extracting src spectra for item 1 of 1 in input list

-----
                        Input Parameters
-----
VERBOSE: infile          = acis_869_evt2.fits[sky=region(simple.reg)][bin PI]
```

specextract: ACIS Spectrum/RMF/ARF – CIAO 3.4

```
VERBOSE: outfile      = simple.pi
VERBOSE: Creating output file simple.pi
VERBOSE: Write WMAP using acis_869_evt2.fits[sky=region(simple.reg)][energy=300:2000][bin det=8
```

Creating src ARF for item 1 of 1 in input list

Parameters for mkwarf:

```
infile      = simple.pi[WMAP]
outfile     = simple.warf
weightfile  = simple.wfef
feffile     = CALDB
egridspec   = 0.3:11.0:0.01
spectrumfile =
threshold   = 0
mirror      = HRMA
ardlibpar   = ardlib
clobber     = no
verbose     = 1
```

Creating src RMF for item 1 of 1 in input list

Using mkacisrmf...

```
*** mkacisrmf parameter inputs ***
input file: CALDB
output file: simple.wrmf
wmap file: simple.pi[WMAP]
energy: 0.3:11.0:0.01
channel: 1:1024:1
chantype: PI
ccd_id:
chipx:
chipy:
contour lvl: 100
log file:
gain file: CALDB
pixlib param. file: geom
threshold: 1.00e-06
clobber(1=yes, 0=no): 0
verbose level: 1
```

CALDB -> /soft/ciao/CALDB/data/chandra/acis/cpf/p2_resp/acisD2000-01-29p2_respN0006.fits

CALDB -> /soft/ciao/CALDB/data/chandra/acis/bcf/gain/acisD2000-01-29gain_ctiN0006.fits

Total 16 regions to be processed:

```
1> reg# 2287 processed
2> reg# 2288 processed
3> reg# 2289 processed
4> reg# 2290 processed
5> reg# 2319 processed
6> reg# 2320 processed
7> reg# 2321 processed
8> reg# 2322 processed
9> reg# 2351 processed
10> reg# 2352 processed
11> reg# 2353 processed
12> reg# 2354 processed
13> reg# 2383 processed
14> reg# 2384 processed
15> reg# 2385 processed
16> reg# 2386 processed
```

Grouping src spectrum for item 1 of 1 in input list

Parameters for dmgroup:

```
infile = simple.pi[SPECTRUM]
outfile = simple_grp.pi
grouptype= NUM_CTS
xcolumn = channel
binspec =
ycolumn = counts
tabspec =
tabcolumn=
stopspec=
stopcolumn=
errcolumn=
grouptypeval= 15.000000
maxlength= 0.000000
clobber = no
verbose = 1
```

Updating header of simple.pi with RESPFILE and ANCRFILE keywords.

```
dmhedit: verbose set to 1
dmhedit: Input file = simple.pi
dmhedit: file list = none
dmhedit: key RESPFILE will be added.
```

```
dmhedit: verbose set to 1
dmhedit: Input file = simple.pi
dmhedit: file list = none
dmhedit: key ANCRFILE will be added.
```

Updating header of simple_grp.pi with RESPFILE and ANCRFILE keywords.

```
dmhedit: verbose set to 1
dmhedit: Input file = simple_grp.pi
dmhedit: file list = none
dmhedit: key RESPFILE will be added.
```

```
dmhedit: verbose set to 1
dmhedit: Input file = simple_grp.pi
dmhedit: file list = none
dmhedit: key ANCRFILE will be added.
```

Extracting bkg spectra for item 1 of 1 in input list

Input Parameters

```
VERBOSE: infile      = acis_869_evt2.fits[sky=region(bg_simple.reg)][bin PI]
VERBOSE: outfile     = simple_bkg.pi
VERBOSE: Creating output file simple_bkg.pi
VERBOSE: Write WMAP using acis_869_evt2.fits[sky=region(bg_simple.reg)][energy=300:2000][bin det=8]
```

Creating bkg ARF for item 1 of 1 in input list

Parameters for mkwarf:

```
infile      = simple_bkg.pi[WMAP]
outfile     = simple_bkg.warf
weightfile  = simple_bkg.wfef
feffile     = CALDB
egridspec   = 0.3:11.0:0.01
spectrumfile =
threshold   = 0
mirror      = HRMA
ardlibpar   = ardlib
```


specextract: ACIS Spectrum/RMF/ARF – CIAO 3.4

```
clobber          = no
verbose          = 1

Creating bkg RMF for item 1 of 1 in input list

Using mkacisrmf...

    *** mkacisrmf parameter inputs ***
        input file: CALDB
        output file: simple_bkg.wrmf
        wmap file: simple_bkg.pi[WMAP]
        energy: 0.3:11.0:0.01
        channel: 1:1024:1
        chantype: PI
        ccd_id:
        chipx:
        chipy:
        contour lvl: 100
        log file:
        gain file: CALDB
        pixlib param. file: geom
        threshold: 1.00e-06
        clobber(1=yes, 0=no): 0
        verbose level: 1

CALDB -> /soft/ciao/CALDB/data/chandra/acis/cpf/p2_resp/acisD2000-01-29p2_respN0006.fits
CALDB -> /soft/ciao/CALDB/data/chandra/acis/bcf/gain/acisD2000-01-29gain_ctiN0006.fits

Total 5 regions to be processed:
1> reg# 2417 processed
2> reg# 2418 processed
3> reg# 2449 processed
4> reg# 2450 processed
5> reg# 2481 processed

Updating header of simple_bkg.pi with RESPFILE and ANCRFILE keywords.

dmhedit: verbose set to 1
dmhedit: Input file = simple_bkg.pi
dmhedit: file list = none
dmhedit: key RESPFILE will be added.

dmhedit: verbose set to 1
dmhedit: Input file = simple_bkg.pi
dmhedit: file list = none
dmhedit: key ANCRFILE will be added.
Updating header of simple.pi with BACKFILE keyword.

dmhedit: verbose set to 1
dmhedit: Input file = simple.pi
dmhedit: file list = none
dmhedit: key BACKFILE will be added.
Updating header of simple_grp.pi with BACKFILE keyword.

dmhedit: verbose set to 1
dmhedit: Input file = simple_grp.pi
dmhedit: file list = none
dmhedit: key BACKFILE will be added.
```

The contents of the parameter file may be checked with [plist specextract](#).

Examining the Output Files

The number of files created depends on if a background event file was provided and if source and/or background grouping was specified. In this case, the output files are:

```

Source:
simple.pi      ungrouped spectrum
simple.warf    weighted ARF
simple.wfef    FEF weight file
simple.wrmf    weighted RMF
simple_grp.pi  grouped spectrum

Background:
simple_bkg.pi  ungrouped spectrum
simple_bkg.warf  weighted ARF
simple_bkg.wfef  FEF weight file
simple_bkg.wrmf  weighted RMF

```

The FEF weight files (.wfef) are required as input to `mkrmf`; they are created by `mkwarf` before the script determines whether to use `mkrmf` or `mkacisrmf`. After the RMFs are created, these files are no longer needed.

Extracting Multiple Spectra

In this example, we show how `specextract` can create multiple output spectra from a single run of the script.

Build Source Regions and Background Regions

This example uses the two observations of G21.5–09. Both observations will be processed by `specextract` at the same time, producing two sets of output files; this is explained further in the [Run specextract section](#).

We define have defined a source and background region for each observation:

```

unix% cat 1842_src.reg
# Region file format: CIAO version 1.0
circle(2249.5,4221.5,102.0092)

unix% cat 1842_bg.reg
# Region file format: CIAO version 1.0
circle(2565.5,4129.5,40)

unix% cat 1843_src.reg
# Region file format: CIAO version 1.0
circle(1635.5,4113.5,135.11408)

unix% cat 1843_bg.reg
# Region file format: CIAO version 1.0
circle(2129.5,4007.5,40)

```

The regions are shown displayed on the event files in [Figure 2 !\[\]\(6a9b39b98eb945faa14c645ec99e4eaa_img.jpg\)](#); ObsID 1842 is on the left and ObsID 1843 is on the right. The source region is in white and the background region is in green in each frame.

Run specextract

The event files are input to the script as a stack; this syntax means that a separate spectrum will be created for each of the file/region pairs:

```
unix% cat multi_src.lis
1842_evt2.fits[sky=region(1842_src.reg)]
1843_evt2.fits[sky=region(1843_src.reg)]
```

When working with stack inputs to `specextract`, the source and background stacks must contain the same number of items, unless you are not extracting background spectra. Make sure that the background file definitions are in the same order as the source files:

```
unix% cat multi_bg.lis
1842_evt2.fits[sky=region(1842_bg.reg)]
1843_evt2.fits[sky=region(1843_bg.reg)]
```

As of CIAO 3.4, it is also possible to supply a stack of output root names:

```
unix% cat multi_out.lis
1842
1843
```

If you prefer, you may still just give a string as the outroot and `specextract` will create output files designated as "src1", "src2", "bkg1", "bkg2", etc.

```
unix% pset specextract outroot=multi
```

Set the parameters:

```
unix% punlearn specextract
unix% pset specextract infile=@multi_src.lis
unix% pset specextract outroot=@multi_out.lis
unix% pset specextract bkgfile=@multi_bg.lis
unix% pset specextract grouptype=BIN binspec=10
```

The source spectra will be grouped into bins of 10 counts each.

Note that this method allows you to input as many event file/region file pairs as you like, but the same grouping will be applied to all of them. The tool is run with `verbose=0` for no screen output:

```
unix% specextract
Source events file(s) (@multi_src.lis):
Output directory path + root name for output files (multi):
```

The contents of the parameter file may be checked with `plist specextract`.

Examining the Output Files

The number of files created depends on if a background event file was provided and if source and/or background grouping was specified. In this case, the output files are:

```
Source 1 (1842_src.reg):
1842.pi      ungrouped spectrum
1842.warf    weighted ARF
1842.wfef    FEF weight file
1842.wrmf    weighted RMF
1842_grp.pi  grouped spectrum

Source 2 (1843_src.reg):
1843.pi
1843.warf
```

```

1843.wfef
1843.wrmf
1843_grp.pi

Background 1 (1842_bg.reg):
1842_bkg.pi
1842_bkg.warf
1842_bkg.wfef
1842_bkg.wrmf

Background 2 (1843_bg.reg):
1843_bkg.pi
1843_bkg.warf
1843_bkg.wfef
1843_bkg.wrmf

```

If one string is provided for the `outroot` parameter, the script simply numbers the output to match the order in which the files were input: `src1 = 1842_src.reg` and `src2 = 1843_src.reg`. Similarly, the background files are `bkg1` and `bkg2`.

More Information on the Output Files

Which RMF tool was used?

When `specextract` is run with `verbose` greater than zero, the status messages printed to the screen report which tool was used to create the RMF, `mkacisrmf` or `mkrmf`. This information is also recorded in the FITS file that is created. If you have verbosity set to zero, or simply don't want to parse all that screen output, use the `dmhistory` tool:

```

unix% dmhistory simple.wrmf all
mkacisrmf infile="CALDB" outfile="simple.wrmf"
wmap="simple.pi[WMAP]" energy="0.3:11.0:0.01" channel="1:1024:1"
chantype="PI" ccd_id="0" chipx="0" chipy="0" gain="CALDB" logfile=""
contlvl="100" geompar="geom" thresh="1e-06" clobber="no" verbose="1"

unix% dmhistory 1842.wrmf all
mkrmf infile="CALDB" outfile="1842.wrmf"
axis1="energy=0.3:11.0:0.01" axis2="pi=1:1024:1" logfile=""
weights="1842.wfef" thresh="1e-05" outfmt="legacy" clobber="no"
verbose="0" axis3="none" axis4="none" axis5="none"

```

The [single spectrum example](#) created the RMF with `mkacisrmf`, while the [multiple spectra example](#) used `mkrmf`. Refer back to the [Creating RMFs section](#) for information on how the script decides which tool to use.

If `specextract` used `mkrmf`, but you know the calibration is good enough for `mkacisrmf`, read the [Running mkacisrmf Independently section](#).

Header keywords

The `RESPFILE` and `ANCRFILE` header keywords in the source and background spectra have been updated, as well as the `BACKFILE` in the source spectra only. For example, in the single source output files:

```

unix% dmlist simple_grp.pi header | grep FILE
0110 BACKFILE          simple_bkg.pi          String
0111 CORRFILE          none                    String
0112 RESPFILE          simple.wrmf             String
0113 ANCRFILE          simple.warf             String

```

```
unix% dmlist simple_bkg.pi header | grep FILE
0112 BACKFILE          none                      String
0113 CORRFILE          none                      String
0114 RESPFILE          simple_bkg.wrmf          String
0115 ANCRFILE          simple_bkg.warf          String
```

If stack inputs were used, the source and background file header values are matched up appropriately:

```
unix% dmlist 1842_grp.pi header |grep FILE
0098 BACKFILE          1842_bkg.pi             String
0099 CORRFILE          none                     String
0100 RESPFILE          1842.wrmf                String
0101 ANCRFILE          1842.warf                String

unix% dmlist 1843_grp.pi header |grep FILE
0098 BACKFILE          1843_bkg.pi             String
0099 CORRFILE          none                     String
0100 RESPFILE          1843.wrmf                String
0101 ANCRFILE          1843.warf                String
```

This means that when the spectra are read into *Sherpa*, all the supporting files will automatically be read as well; the background (if available) will be defined, as will the source and background response files.

Running mkacisrmf Independently

Users with "good enough" calibration who wish to run `mkacisrmf` to create new RMF response files should now follow the [Using mkacisrmf with the specextract script section](#) of the [Creating ACIS RMFs with mkacisrmf thread](#).

This also applies to users with GRADED mode data on chips S1 or S3 (the back-illuminated chips), as explained in the [ACIS GRADED Mode Data section](#) of that thread.

Fitting

If you would like to fit the background-subtracted source spectrum using a common RMF and ARF for source and background, simply read the source spectrum FITS file into *Sherpa*, subtract the background, and fit it. See the [Introduction to Fitting PHA Spectra](#) thread for details.

To fit source and background spectra simultaneously with proper and distinct RMFs and ARFs, load the source and background as different datasets. This procedure is discussed in the [Independent Background Responses](#) thread.

Caveats

Using the ACIS "Blank-Sky" Background Files

Subtracting the Background

If you intend to subtract the background spectrum (i.e. not fit it), then you do not need to create a background RMF and ARF. You may simply use `dmextract` to create the spectrum. In this case, leave the `bkgfile`

parameter blank so that `specextract` will only create the source spectrum and responses.

Header Keyword Issue

The older ACIS "blank-sky" background files in the CALDB do not work correctly when input to this script "as-is". Users will see a warning of this form:

```
# mkwarf (CIAO 3.4): WARNING: No files found matching CALDB search:
  tel=CHANDRA
  inst=ACIS
  det=-
  filt=-
  product=FEF_PI
  start_date=1998-01-01T00:00:00
  start_time=00:00:00
  stop_date=1998-01-06T18:53:20
  stop_time=18:53:20
  query=CTI_CORR.eq.NO

# mkwarf (CIAO 3.4): WARNING: No files found matching CALDB search:
  tel=CHANDRA
  inst=ACIS
  det=-
  filt=-
  product=FEF_PHA
  start_date=1998-01-01T00:00:00
  start_time=00:00:00
  stop_date=1998-01-06T18:53:20
  stop_time=18:53:20
  query=CTI_CORR.eq.NO
```

Follow the workaround in the [specextract bugs page](#) in order to use these files with the script.

Analysis at the edges of ACIS CCDs

Users should be cautious about analyzing the data for sources near the edges of the ACIS CCDs.

1. For X-rays passing through the mirrors, the very bottom of each CCD is obscured by the frame store. As a result, some of the events in rows with `CHIPY <= 8` are not detected. (The set of rows affected varies from CCD to CCD.) Since the CIAO tools do not compensate for this effect, the ARFs and exposure maps for sources in these regions may be inaccurate.
2. For sources within about thirty-two pixels of any edge of a CCD, the source may be dithered off the CCD during part of an observation. The aspect histogram, which is used to create ARFs and exposure maps, is designed to compensate for this effect.
3. A contaminant has accumulated on the optical-blocking filters of the ACIS detectors, as described in the [ACIS OE Degradation why topic](#). Since there is a gradient in the temperature across the filters (the edges are colder), there is a gradient in the amount of material on the filters. (The contaminant is thicker at the edges.) Within about 100 pixels of the outer edges of the ACIS-I and ACIS-S arrays, the gradient is relatively steep. Therefore, the effective low-energy (< 1 keV) detection efficiency may vary within the dither pattern in this region. The ARF and instrument map tools are designed to read a calibration file which describes this spatial dependence.

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

```
infile = acis_869_evt2.fits[sky=region(simple.reg)] Source events file(s)
```

specextract: ACIS Spectrum/RMF/ARF – CIAO 3.4

```
outroot = simple          Output directory path + root name for output files
(bkgfile = acis_869_evt2.fits[sky=region(bg_simple.reg)]) Background events file(s)
  (ptype = PI)            PI or PHA
(grouptype = NUM_CTS)    Spectrum grouping type (same as grouptype in dmgroup)
  (binspec = 15)         Spectrum grouping specification (NONE,1:1024:10,etc)
(bkg_grouptype = NONE)   Background spectrum grouping type (NONE, BIN, SNR, NUM_BINS, NU
  (bkg_binspec = )       Background spectrum grouping specification (NONE,10,etc)
  (energy = 0.3:11.0:0.01) Energy grid
  (channel = 1:1024:1)   RMF binning attributes
(energy_wmap = 300:2000) Energy Range for WMAPs
  (binwmap = det=8)      Binning factor for WMAPs
  (clobber = no)         OK to overwrite existing output file?
  (verbose = 0)          Debug Level(0-5)
  (mode = ql)
```

Parameters for /home/username/cxcds_param/specextract.par

```
infile = @multi_input.lis Source events file(s)
outroot = multi          Output directory path + root name for output files
(bkgfile = @multi_bkg.lis) Background events file(s)
  (ptype = PI)            PI or PHA
(grouptype = BIN)       Spectrum grouping type (same as grouptype in dmgroup)
  (binspec = 10)         Spectrum grouping specification (NONE,1:1024:10,etc)
(bkg_grouptype = NONE)   Background spectrum grouping type (NONE, BIN, SNR, NUM_BINS, NU
  (bkg_binspec = )       Background spectrum grouping specification (NONE,10,etc)
  (energy = 0.3:11.0:0.01) Energy grid
  (channel = 1:1024:1)   RMF binning attributes
(energy_wmap = 300:2000) Energy Range for WMAPs
  (binwmap = det=8)      Binning factor for WMAPs
  (clobber = no)         OK to overwrite existing output file?
  (verbose = 0)          Debug Level(0-5)
  (mode = ql)
```

History

- 01 Feb 2006 new for CIAO 3.3
- 15 Feb 2006 created [Running mkacisrmf Independently section](#)
- 31 Mar 2006 specextract use update added to Overview
- 05 Apr 2006 In light of the specextract usage change, the thread has been rewritten to use extended sources in the examples
- 14 Apr 2006 added [Analysis at the edges of ACIS CCDs caveat](#)
- 24 May 2006 added new information to [Using the ACIS "Blank-Sky" Background Files caveat](#)
- 14 Jun 2006 corrected link in "Calibration Updates"; clarified information on GRADED mode data
- 18 Dec 2006 updated for CIAO 3.4: new calibration files in CALDB 3.3.0; [Extracting Multiple Spectra section](#) uses a stack of output file roots (new feature in CIAO 3.4); output files in [one-output case](#) no longer have "src1" or "bkg1" included in the filename; mkrmf no longer prints messages at verbose=0; CIAO version in warnings
- 06 Mar 2007 added [ACIS dead area correction section](#)
- 16 May 2007 The [header keyword problem with the ACIS background files](#) should be eliminated with the new ACIS blank-sky background files released in CALDB 3.4.0 (16 May 2007)

specextract: ACIS Spectrum/RMF/ARF – CIAO 3.4

31 Mar 2008 updated for CALDB 3.4.3: use `mkacisrmf` for -110 BI chips if TGAIN calibration has been applied

URL: <http://cxc.harvard.edu/ciao/threads/specextract/>

Last modified: 31 March 2008

Image 1: Single Spectrum: extraction regions

The source region is shown in white and the background region is green.

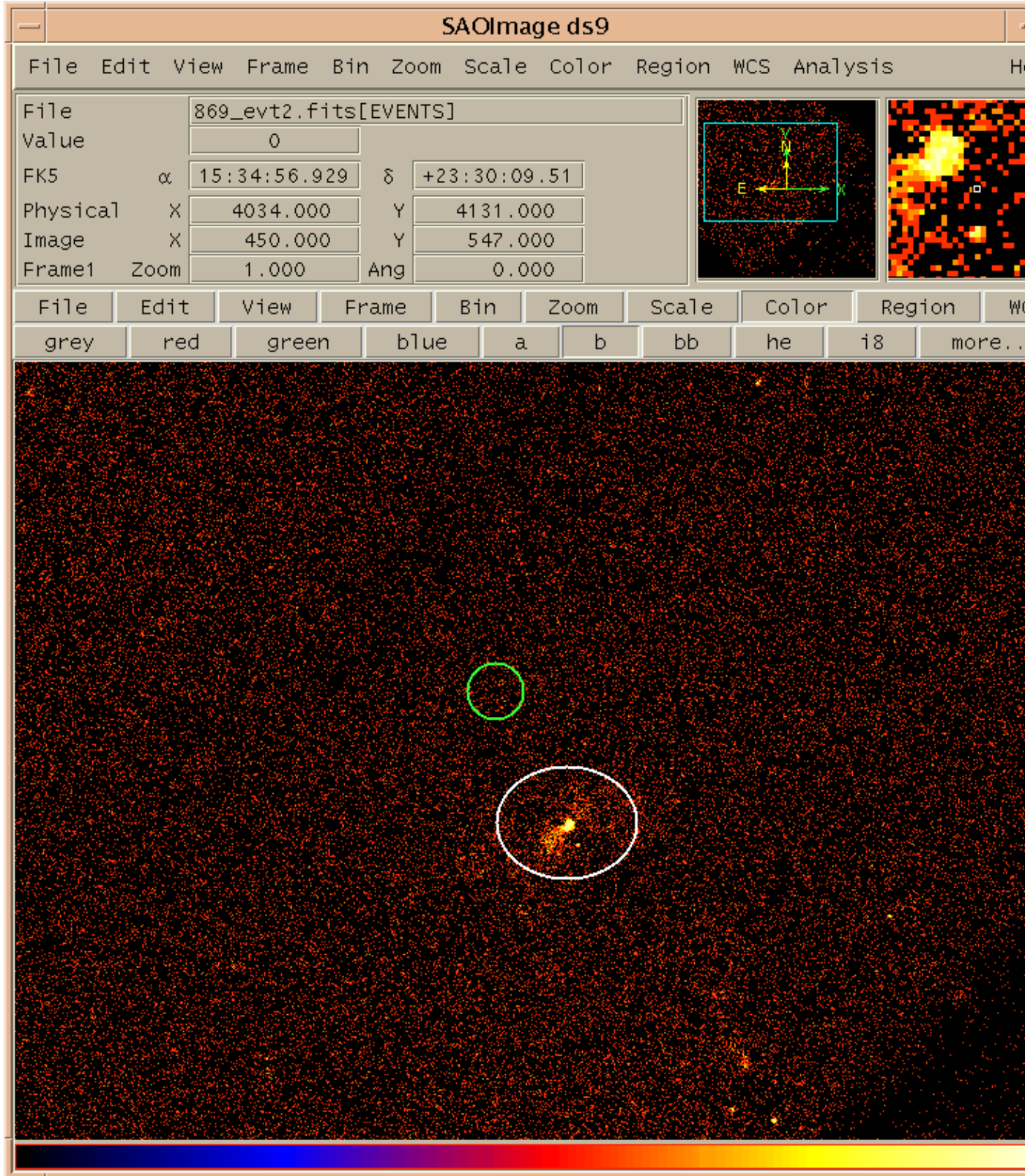


Image 2: Multiple Spectra: extraction regions

ObsID 1842 is on the left and ObsID 1843 is on the right. The source region is in white and the background region is in green in each frame.

