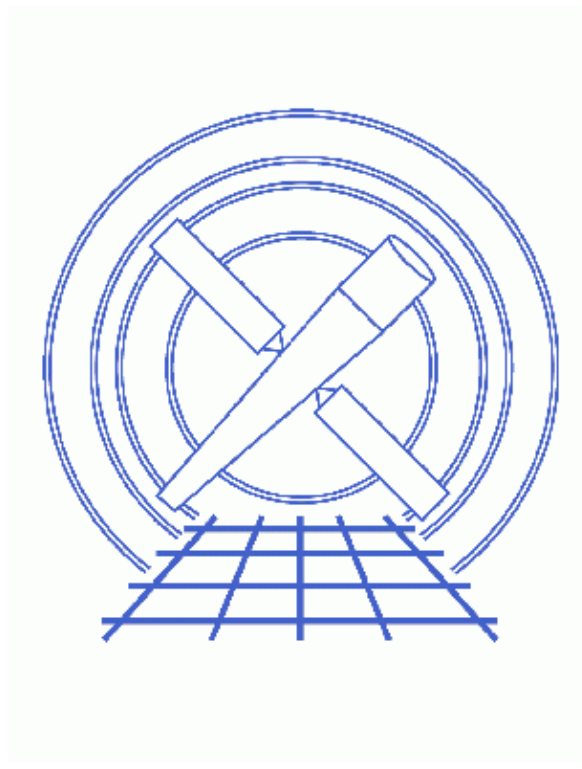


## Weighting ARFs and RMFs: multiple sources



### ***CIAO 3.4 Science Threads***

# Table of Contents

- **Get Started**
  - ◆ CALDB 3.3.0.1 patch
  - ◆ Using Consistent Calibration: mkrmf vs mkacisrmf
  - ◆ The ACIS dead area correction
- **Create the WMAP**
  - A. Using dmextract
  - B. Using dmcoppy
- **Create the weighted ARF (mkwarf)**
  - A. Using the WMAP from a PHA file
  - B. Using an image as a WMAP
  - C. Which FEF was Used by mkwarf
- **Examine the weights**
- **Create the weighted RMF (mkrmf)**
- **Updating the header (dmhedit)**
- **Improving the weights**
- **Caveats**
- **Parameter files:**
  - ◆ dmextract
  - ◆ mkwarf
  - ◆ mkrmf
- **History**
- **Images**
  - ◆ A look at the weights file
  - ◆ Manipulating the graph

---

# Weighting ARFs and RMFs: multiple sources

*CIAO 3.4 Science Threads*

## Overview

**Last Update:** 6 Mar 2007 – added ACIS dead area correction section and example of setting the `pbkfile` and `dafile` parameters

### *Synopsis:*

If you want to extract the spectrum of a large region, or combine data from multiple regions (either from the same or different observations), then you *may* need to use a weighted ARF and RMF for the spectral analysis. This is because the RMF and ARF vary with detector location; the RMF is defined as constant across FEF tiles, whose size depends on both the position on and focal-plane temperature of the ACIS chip, while the ARF varies with position in the detector plane.

### *Purpose:*

To create a spectrum that represents a number of point sources present in the field, and a corresponding ARF and RMF that can be used to analyze it.

### *Read this thread if:*

you are working with an ACIS observation and need to create weighted response files.

### *Calibration Updates:*

- **CALDB v3.3.0.1** (2 Feb 2007): The CALDB 3.3.0.1 patch corrects an indexing problem that may affect users of this thread. Read the Caveat: ACIS –120C FEF for CTI-corrected ACIS data for details.
- **CALDB v2.26** (2 Feb 2004): New FEF files containing updated information for ACIS S3 were added to the CALDB; there are CTI-corrected and uncorrected versions available.
- **CALDB v2.21** (14 Feb 2003): The new CTI-corrected FEF file (`acisD2000-01-29fef_pha_ctiN0002.fits`) contains updated information for chips S0, S1, and S3–5, eliminating the calibration problem announced in December 2002. Applying the CTI correction to these chips will no longer have a negative effect on the data.

### *Related Links:*

- Analysis Guide: Extended Sources
- Caveats: please read this section before using this thread to analyze your data.
- specextract: a script that can be used to create weighted response files for extended sources or multiple sources in a field. Refer to the specextract thread for instructions.

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

---

## Get Started

*Sample ObsID used:* 578 (ACIS–S, 3C 295)

*File types needed:* evt2

This thread uses the `show_wgt.sl` script. The most recent version of `show_wgt.sl` is v1.2 (22 October 2002):

```
unix% grep Id $ASCDS_CONTRIB/share/slsh/local-packages/show_wgt.sl
% $Id: show_wgt.sl,v 1.2 2001/10/22 16:17:22 dburke Exp $
```

Note that `$ASCDS_CONTRIB/share/slsh/local-packages/` is the default path in the standard CIAO scripts installation; see the [Scripts page](#) for more information. **Please check that you are using the most recent version before continuing.** If you do not have the script installed or need to update to a newer version, please refer to the [Scripts page](#).

A region file created by [wavdetect](#) is used as well; you should therefore have completed the [Running wavdetect](#) thread (or copy the region file listed below). We edit the source list to remove the two sources associated with the cluster – the easiest way to do this is to use `ds9` – and call the file `sources.reg`:

```
unix% cat sources.reg
# Region file format: CIAO version 1.0
ellipse(3868.625,4004.125,4.272175,3.60101,147.60056)
ellipse(3928.3611,4282.3333,6.024213,4.747131,118.04667)
ellipse(3960.9444,4035.3222,3.761969,2.861008,38.675421)
ellipse(3998.0455,3944.4091,3.818624,2.833669,70.26377)
ellipse(4035.2564,4279.8551,4.377677,3.239413,17.127037)
ellipse(4085.9444,4360.9,6.624345,4.571682,128.5409)
ellipse(4109.0991,4338.5377,4.447523,2.962107,29.539664)
ellipse(4142.4333,4301.0333,4.299932,2.170389,61.937297)
ellipse(4218.1338,4298.8521,4.953961,3.579325,6.339128)
```

To save time, we filter the event list using this list:

```
unix% punlearn dmcop
unix% dmcop "acisf00578N002_evt2.fits[sky=region(sources.reg)]" sources.evt2
```

### CALDB 3.3.0.1 patch

The [CALDB 3.3.0.1 patch](#), released on 02 February 2007, corrects an indexing problem that may affects users of this thread. Read the [Caveat: ACIS –120C FEF for CTI-corrected ACIS data](#) for details.

Make sure that this patch has been installed in your CALDB before continuing:

```
unix% dmlist "$CALDB/docs/chandra/caldb_version/caldb_version.fits[cols caldb_ver,ciao_ver]" data
...
 52 3.2.4      CIAO3.3.0.1
 53 3.3.0      CIAO3.4
 54 3.3.0.1    CIAO3.4
```

This file is automatically updated each time the CALDB is upgraded on your system, so the final row always indicates the current version.

---

## Using Consistent Calibration: mkrmf vs mkacisrmf

The tool `mkacisrmf`, available since CIAO 3.2, represents an entirely new method for creating ACIS imaging response matrices. Details on the tool are available in the [Creating ACIS RMFs why topic](#). This thread uses the older tool `mkrmf` to create the RMF files.

Users who wish to take advantage of the improvements in `mkacisrmf` may use it in place of `mkrmf`. The [Creating ACIS RMFs with mkacisrmf thread](#) shows how to run the tool.

It is especially important to read the caveat on [Matching the number of energy bins](#) if you intend to use `mkacisrmf`.

## 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, the application of the dead area correction is *turned off* by default. However, users may opt to include it in the analysis by setting the `pbkfile` and `dafile` parameters in the [mkwarf step](#). Refer to the [mkwarf help file](#) for details on these parameters.

If you wish to include the correction when running this thread, set the `pbkfile` and `dafile` parameters:

```
unix% pset mkwarf pbkfile=acisf052379707N002_pbk0.fits dafile=CALDB
```

## Create the WMAP

The `mkwarf` tool takes as input an image of the field – in detector coordinates – which we refer to as the WMAP. This image can either be stored in the WMAP block of a spectrum created by `dmextract` or as an image by using the [DM binning syntax](#) with `dmcopy`.

### A. Using dmextract

To create a WMAP block in a PHA file, `dmextract` must be given a binning specification for its `wmap` option. As the WMAP must be in detector coordinates, and the suggested binning factor is 8, we have

```
unix% punlearn dmextract
unix% pset dmextract infile="sources.evt2[bin pi]"
unix% pset dmextract outfile=sources.pi
unix% pset dmextract wmap="[bin det=8]"
unix% dmextract
Input event file (sources.evt2[bin pi]):
Enter output file name (sources.pi):
```

The file `sources.pi` contains both the source spectrum and WMAP. You can check the parameter file that was used with [plist dmextract](#).

Note that `dmextract` allows additional filters to be specified for the `wmap` parameter. This means that you can filter the WMAP on energy – e.g. the 0.5 to 2.0 keV band – without having to have a separate WMAP:

```
unix% pset dmextract wmap="[energy=500:2000][bin det=8]"
```

Running `dmlist` on the output file shows the WMAP block:

```
unix% dmlist sources.pi blocks
```

---

Dataset: sources.pi

---

Block Name	Type	Dimensions	
Block 1: WMAP	Image	Int2(1024x1024)	
Block 2: SPECTRUM	Table	4 cols x 1024	rows
Block 3: GTI7	Table	2 cols x 1	rows
Block 4: GTI5	Table	2 cols x 2	rows
Block 5: GTI2	Table	2 cols x 7	rows
Block 6: GTI3	Table	2 cols x 7	rows
Block 7: GTI8	Table	2 cols x 4	rows
Block 8: GTI6	Table	2 cols x 5	rows

## B. Using dmcopy

If we use `dmcopy` directly, we are free to choose any energy range for the WMAP. Here we use the soft band (0.5 to 2.0 keV), which effectively count-weights the averaging of the responses, whereas use of harder energies, where the background dominates, would use area-weighting. As with the `dmextract` option, we use the detector coordinate system with a binning factor of 8:

```
unix% dmcopy "sources.evt2[energy=500:2000][bin det=8]" sources.wmap8
```

This command creates a WMAP image in detector coordinates:

```
unix% dmlist sources.wmap8 blocks
```

---

Dataset: sources.wmap8

---

Block Name	Type	Dimensions	
Block 1: EVENTS_IMAGE	Image	Int2(1024x1024)	
Block 2: GTI7	Table	2 cols x 1	rows
Block 3: GTI5	Table	2 cols x 2	rows
Block 4: GTI2	Table	2 cols x 7	rows
Block 5: GTI3	Table	2 cols x 7	rows
Block 6: GTI8	Table	2 cols x 4	rows
Block 7: GTI6	Table	2 cols x 5	rows

## Create the weighted ARF (mkwarf)

Two methods of creating WMAPs were illustrated in the [previous section](#). Here we show how each of them is used in `mkwarf`.

### A. Using the WMAP from a PHA file

Unlike `mkarf`, `mkwarf` must be given an energy grid that lies within the range of energies of the FEF file. An easy way to find the allowed range is to run `mkwarf` with an energy range that definitely extends outside the allowed values – we use 0.01 to 11 keV here. Further information on the FEF files can be found in the

Chandra CALDB pages (in particular the "[Calibration Data](#)" section).

In this run, the WMAP created with `dmextract` (`sources.pi`) is used:

```
unix% punlearn mkwarf
unix% pset mkwarf infile="sources.pi[WMAP]"
unix% pset mkwarf outfile=sources.warf
unix% pset mkwarf weightfile=sources.wgt
unix% pset mkwarf egridspec=0.01:11:0.01
unix% mkwarf
Input detector WMAP (sources.pi[WMAP]):
Output weighted ARF file (sources.warf):
Output FEF weights (sources.wgt):
Input Spectral weighting file (<filename>|NONE) ():
Output energy grid [keV] (0.01:11:0.01):
# mkwarf (CIAO 3.4): The following error occurred 175 times:
    ERROR: Min egridspec energy=0.01 below min FEF energy=0.277

# mkwarf (CIAO 3.4): The following error occurred 175 times:
    ERROR: Max egridspec energy=11 above max FEF energy=9.886
```

Using the energy range 0.28 to 9 keV, re-run `mkwarf`:

```
unix% pset mkwarf egridspec=0.28:9:0.01
unix% mkwarf
Input detector WMAP (sources.pi[WMAP]):
Output weighted ARF file (sources.warf):
Output FEF weights (sources.wgt):
Input Spectral weighting file (<filename>|NONE) ():
Output energy grid [keV] (0.28:9:0.01):
```

The output from `mkwarf` is a weighted ARF (`sources.warf`) and a weights file (`sources.wgt`). You can check the parameter file that was used with `pllist mkwarf`.

Error messages are given if the `infile` is erroneously given without the "[WMAP]" block designation and the tool will fail:

```
...
Couldn't determine chip position for pixel: (1013.000000,0.000000) with value=1.000000. Skipping
Couldn't determine chip position for pixel: (1014.000000,0.000000) with value=1.000000. Skipping
...
# mkwarf (CIAO 3.4): ERROR: No valid response regions could be created.
```

In this case, the "pixel" locations will be the spectral bins of the PHA or PI spectrum to which the WMAP is usually attached.

## B. Using an image as a WMAP

Since we used the WMAP within the PHA file as the value for the `infile` parameter in the previous `mkwarf` run, we had to specify the extension name of the block containing the WMAP data (i.e. the "[WMAP]" `DM filter`). Here we use a similar filter to avoid seeing several warning generated by `ardlib`; it is not required that you specify the block. Remember that now we are using the WMAP created with `dmcopy` (`sources.wmap8`).

Since the energy range was determined in the previous example, it is not necessary to run the tool twice:

```
unix% punlearn mkwarf
unix% pset mkwarf infile="sources.wmap8[EVENTS_IMAGE]"
unix% pset mkwarf outfile=sources_img.warf
unix% pset mkwarf weightfile=sources_img.wgt
unix% pset mkwarf egridspec=0.28:9:0.01
unix% mkwarf
```

## Weighting ARFs and RMFs: multiple sources – CIAO 3.4

```
Input detector WMAP (sources.wmap8[EVENTS_IMAGE]):
Output weighted ARF file (sources_img.warf):
Output FEF weights (sources_img.wgt):
Input Spectral weighting file (<filename>|NONE) ():
Output energy grid [kev] (0.28:9:0.01):
# mkwarf (CIAO 3.4): WARNING: Input image name was "EVENTS_IMAGE" instead of "WMAP". Will attempt to u
```

A few notes on warning messages:

- The warning message seen above can be ignored in this case, since we know that `sources.wmap8` was created as a WMAP image (i.e. the detector coordinate system was used).
- If you *do not* specify the extension for the `infile`, you may see warnings of this form:

```
***ARDLIB warning: Filename sources.wmap8 does not specify an extension.
    Assuming the first "interesting" extension.
```

The warnings may be ignored as they do not adversely affect the output.

- If your source is near a chip boundary, then `mkwarf` may not be able to determine the chip location of every pixel in the WMAP (this has to do with the lack of SIM info in the image); details are available in [this FAQ](#). This will lead to warnings such as:

```
Couldn't determine chip position for pixel: (3884.500000,3588.500000) with value=1.000000. Skipp
Couldn't determine chip position for pixel: (3884.500000,3596.500000) with value=5.000000. Skipp
```

In the vast majority of cases, the number of counts ignored (i.e. the sum of the "value" of each ignored pixel) is much smaller than the total signal in the WMAP. However, this may not be the case for small regions near chip boundaries. `dmstat` may be used to determine the sum of all pixels in the WMAP for comparison to the sum of the ignore pixel values.

---

### C. Which FEF was Used by mkwarf

Unlike `mkarf`, `mkwarf` does not need to be told the FEF file to use, as it can query the CALDB to find the matching file (see "[ahelp caldb](#)" for more information on the syntax of the CALDB tool interface). To find out which FEF file it used, either set `verbose=2` before running the tool (it gets printed out during execution), examine the `FEFFILE` keyword in the file header, or use the [acis\\_fef\\_lookup](#) script (available from the [Scripts page](#)), specifying a `chipid` of none. We show examples of these methods here (the output assumes that the `$CALDB` environment variable is set to `/soft/ciao/CALDB`):

```
unix% mkwarf verbose=2
...
Mapping response regions to FEF regions
FEF File: /soft/ciao/CALDB/data/chandra/acis/cpf/feffs/acisD1999-08-13fef_phaN0002.fits
Mapping response regions to FEF regions. Done
...
```

and

```
unix% dmkeypar sources.wgt FEFFILE echo+
/soft/ciao/CALDB/data/chandra/acis/cpf/feffs/acisD1999-08-13fef_phaN0002.fits
```

and

```
unix% acis_fef_lookup sources.evt2 none 1 1
/soft/ciao/CALDB/data/chandra/acis/cpf/feffs/acisD1999-08-13fef_phaN0002.fits[FUNCTION]
```

---




## Examine the weights

There are two outputs from `mkwarf`: the weighted ARF (`sources.warf`) and a weights file (`sources.wgt`). The weights file is used by `mkrmf` to create the weighted RMF file. We can examine the weights file, using the `show_wgt.sl` S-Lang code, to see how the different FEF regions contribute to the weighted response.

```
unix% chips --slscript show_wgt.sl

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

chips> show_wgt("sources.wgt")
```

creates [Figure 1](#) .

*ChIPS* commands can be used to modify the graph created by the `show_wgt()` S-Lang function: for instance, we can set the y axis of the bottom plot to log scale

```
chips> d 2 log y
chips> quit
```

which creates [Figure 2](#) .

The top plot shows the number of counts falling within each FEF "tile" (i.e. `REGNUM`), and the bottom plot repeats the information, but in terms of the fraction counts. Since `mkrmf` can take a long time to run when given many FEF tiles to process, `mkwarf` contains a threshold parameter which, when set to a value greater than 0, restricts the calculation to only those FEF tiles whose fractional contribution (i.e. the `FRACTION` value) is greater than the input threshold. Since there is no *a priori* correct value for this cut – it depends on both the science requirements of the analysis and the surface brightness profile of the emission – the default value for `threshold` is set to 0.

## Create the weighted RMF (mkrmf)

The `mkrmf` tool uses the weightfile from `mkwarf` to create the weighted RMF. The parameters it needs are slightly different from the "single FEF region" case: the FEF file is given using a CALDB query and the energy axis grid is ignored – as long as a syntactically-correct value is entered – since the grid is read from the weights file.

If you don't want any information printed to the screen, leave the verbosity setting for `mkrmf` at 0.

```
unix% punlearn mkrmf
unix% pset mkrmf infile=CALDB
unix% pset mkrmf outfile=sources.wrnf
unix% pset mkrmf axis1="energy=0:1"
unix% pset mkrmf axis2="pi=1:1024:1"
unix% pset mkrmf weights=sources.wgt
unix% mkrmf verbose=1
name of FEF input file (CALDB):
name of RMF output file (sources.wrnf):
axis-1(name=lo:hi:btype) (energy=0:1):
axis-2(name=lo:hi:btype) (pi=1:1024:1):

*** mkrmf parameter inputs ***
input file: CALDB
```

```

output file: sources.wrmf
weights file: sources.wgt
axis1: energy=0:1
axis2: pi=1:1024:1
log file: STDOUT
axis3: none
axis4: none
axis5: none
threshold: 1.00e-05
output fmt: legacy
clobber(1=yes, 0=no): 0
verbose level: 1

```

```
CALDB->/data/CALDB/test_2/data/chandra/acis/cpf/fefs/acisD1999-08-13fef_phan0002.fits
```

```
Atten: grid input,"energy=0:1", is ignored as a weights file, "sources.wgt", is found.
```

```
Total 32 regions to be processed:
```

```
--- Region #1 processed
```

```
--- Region #2 processed
```

```
(clip)
```

```
--- Region #32 processed
```

The output from `mkrmf` is a weighted RMF (`sources.wrmf`). You can check the parameter file that was used with `plist mkrmf`.

## Updating the header (dmhedit)

The source spectrum (`sources.pi`) can now be analyzed – using the ARF `sources.warf` and RMF `sources.wrmf` – in *Sherpa*, as described in the [Sherpa threads](#). The file header needs updating in order for *Sherpa* to automatically pick up the ARF and RMF when the source is read in:

```

unix% punlearn dmhedit
unix% dmhedit infile=sources.pi filelist= operation=add key=RESPFILE value=sources.wrmf
unix% dmhedit infile=sources.pi filelist= operation=add key=ANCRFILE value=sources.warf

```

## Improving the weights

In the example above we used the detected counts to weight the responses, when what what we really should be using is the incident flux – i.e. the source spectrum before it passes through the telescope/detector system. If we assume an incident spectrum, we can correct the detected counts to give the incident flux, and use that for the weights. Since we do not know the source spectrum, we can use an iterative scheme – where the best-fit spectrum of the source from the previous iteration is used to correct the WMAP for the current iteration, until the results converge. An alternative scheme is to continue to use the detected counts – i.e. make *no* correction for the telescope/detector response – but only use a restricted energy range where the response + model does not vary much; an example being the soft energy band for clusters of galaxies.

The `spectrumfile` parameter of `mkwarf` accepts an ASCII file containing a model of the source spectrum; it will handle either the output of *Sherpa*'s `write source` command or the `spectrum.sl` script (examples of creating both outputs are given in the [Calculating Spectral Weights](#) thread). Note that the energy range of the spectral model should match that used to create the WMAP; the easiest way to ensure this is to edit the output of *Sherpa*'s `write source` command before using it as input to `mkwarf`.

Note that, unlike `mkinstmap`, the input spectrum does not need to be normalized to unit flux.

---

## Caveats

- The algorithm used to calculate the weighted responses requires that there are no spatial variations in the source spectrum.
- If multiple datasets are being combined to create a single WMAP, the observations must have very similar `SIM_Z` values. As a change in `SIM_Z` of 1 mm corresponds to a change of 41.7 pixels (i.e. 1/pixel size) and the shift should be  $\ll 32$  pixels, a suggested tolerance is 0.1 mm (~ 4 pixels).
- The energy bin size is important when creating a weighted ARF, since it is also used by `mkrmf` to create the weighted RMF. Although both `mkwarf` and `mkrmf` will run successfully if the bin size is not specified (and so defaults to 1.0 keV), the results will not adequately describe the instrument response, leading to obviously–incorrect fits. Therefore, when `mkwarf` is run, the bin size should always be specified for the `egridspec` parameter.

## Analysis Caveats

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/dmextract.par`

```
#-----
#
# DMEXTRACT -- extract columns or counts from an event list
#
#-----
    infile = sources.evt2[bin pi] Input event file
    outfile = sources.pi          Enter output file name
    (bkg = )                      Background region file or fixed background (counts/pixel/s) sub
    (error = gaussian)            Method for error determination(poisson|gaussian|<variance file>
    (bkgerror = gaussian)        Method for background error determination(poisson|gaussian|<var
    (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 = phal)                 Output file type: phal
```

## Weighting ARFs and RMFs: multiple sources – CIAO 3.4

```
(defaults = ${ASCDS_CALIB}/cxo.mdb -> /soft/ciao/data/cxo.mdb) Instrument defaults file
  (wmap = [bin det=8])      WMAP filter/binning (e.g. det=8 or default)
  (clobber = no)           OK to overwrite existing output file(s)?
  (verbose = 0)            Verbosity level
  (mode = ql)
```

---

Parameters for /home/username/cxcds\_param/mkwarf.par

```
  infile = sources.pi[WMAP] Input detector WMAP
  outfile = sources.warf     Output weighted ARF file
  weightfile = sources.wgt   Output FEF weights
  spectrumfile =             Input Spectral weighting file (<filename>|NONE)
  egridspec = 0.28:9:0.01   Output energy grid [keV]
  (threshold = 0)           Percent threshold cut for FEF regions
  (feffile = CALDB)        FEF file
  (mskfile = )              Mask file
  (asolfile = )             Stack of aspect solution files
  (mirror = HRMA)           ARDLIB Mirror specification
  (detsubsysmod = )         Detector subsystem modifier
  (pbkfile = )              Parameter block file
  (dofile = NONE)           Dead area file
  (ardlibpar = ardlib)      Parameter file for ARDLIB files
  (geompar = geom)          Parameter file for Pixlib Geometry files
  (clobber = no)            Clobber existing outputs
  (verbose = 0)             Tool chatter level
  (mode = ql)
```

---

Parameters for /home/username/cxcds\_param/mkrmf.par

```
  infile = CALDB            name of FEF input file
  outfile = sources.wrmf    name of RMF output file
  axis1 = energy=0:1        axis-1(name=lo:hi:btype)
  axis2 = pi=1:1024:1       axis-2(name=lo:hi:btype)
  (logfile = STDOUT)        name of log file
  (weights = sources.wgt)   name of weight file
  (thresh = 1e-5)           low threshold of energy cut-off probability
  (outfmt = legacy)         RMF output format (legacy|cxc)
  (clobber = no)            overwrite existing output file (yes|no)?
  (verbose = 0)             verbosity level (0 = no display)
  (axis3 = none)            axis-3(name=lo:hi:btype)
  (axis4 = none)            axis-4(name=lo:hi:btype)
  (axis5 = none)            axis-5(name=lo:hi:btype)
  (mode = ql)
```

---

## History

- 04 Jan 2005 updated for CIAO 3.2: added information about `mkacisrmf` (see [Creating the RMF: mkrmf vs mkacisrmf section](#))
- 20 Dec 2005 updated for CIAO 3.3: default value of `dmextract` error and `bkgerror` parameters is "gaussian"; updated screen output accordingly
- 01 Feb 2006 added information about [specextract thread](#)

## Weighting ARFs and RMFs: multiple sources – CIAO 3.4

- 01 Dec 2006 updated for CIAO 3.4: CIAO and ChIPS versions; set `mkrmf` verbosity  $> 0$  for screen output; parameter file updates for `mkwarf`
- 02 Feb 2007 updated for [CALDB 3.3.0.1 patch](#)
- 06 Mar 2007 added [ACIS dead area correction section](#) and example of setting the `pbkfile` and `dafile` parameters

---

URL: [http://cxc.harvard.edu/ciao/threads/wresp\\_multiple\\_sources/](http://cxc.harvard.edu/ciao/threads/wresp_multiple_sources/)

Last modified: 6 March 2007

Image 1: A look at the weights file

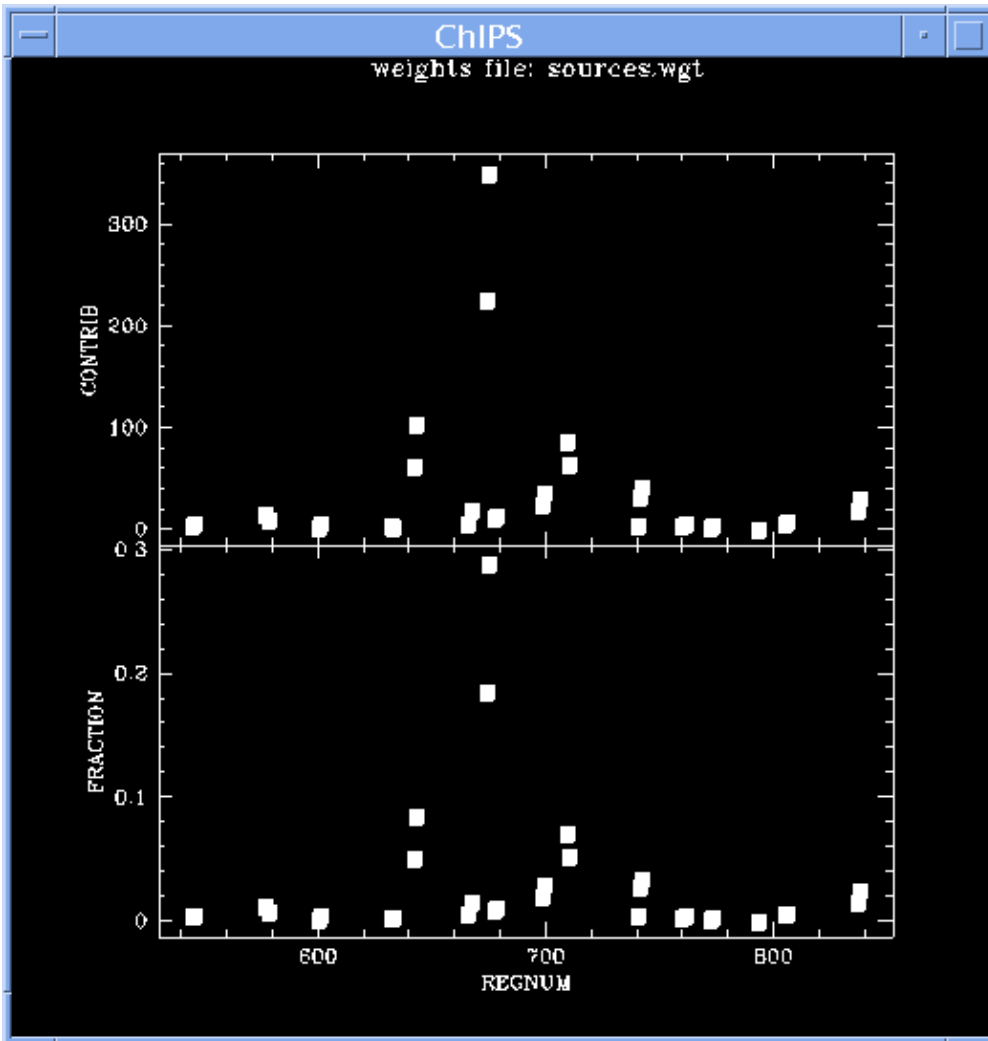


Image 2: Manipulating the graph

