MIT Kavli Institute **Chandra X-Ray Center**

MEMORANDUM
August 10, 2019

To: Jonathan McDowell, SDS Group Leader
Raffaele, D'Abrusco, Archive Group Leader
Janet DePonte Evans, DS Software Development Manager
From: David P. Huenemoerder, SDS
Subject: Pipeline enhancements for computation of responses for grating data
Revision: 1.0
URL: `http://space.mit.edu/cxc/docs/docs.html#tgpipe`
File: `cxc_pipe_tg_enhancement.tex`

1 Grating Data Response Computation

The CXC pipeline standard products for all grating observations contain a source position used for computation of diffraction coordinates. The position is stored in a "`REGION`" block attached to the `evt2` file. From Level 1 and Level 2 standard pipeline products, we can compute the source-position-dependent grating responses ("`ARF`" and "`RMF`" files), which are an essential part of standard data analysis and are used in conjunction with the Level 2 standard product `pha` file which containins the binned count spectrum.

Response generation is currently left to the user (or *TGCat*), but there is no reason this could not be done in the pipline for grating data. This differs from imaging-mode data for which there is no source position defined by a detect algorithm in Level 1 or Level 2 processing, since there may be one or more sources of interest, or the primary target may be an extended source.

Providing grating-data responses for the nominal target would be a valuable addition to the archive since products then exist for common data analysis tasks. Custom extractions (e.g., time-filtered, or serendipitous sources in the field) would still be left to users, since such general cases are not defined by the standard pipeline.

Here we specify the requirements for generation of grating response files in the pipeline. [1]

1.1 Supported Instruments and Modes

Only responses for standard modes — those supported by CIAO and for which `CALDB` products exist — will be computed, currently[2]

- HETG/ACIS-S
- LETG/ACIS-S
- LETG/HRC-S

1.2 Scope

Here we only describe the response tool parameters, including their input files and the output files to be archived. We do not discuss pipeline control, logging, errors, or other pipeline-specific details since they are beyond the scope of this document.

New data products to be archived are already in formats defined in the CIAO system. All inputs required to compute the new pipeline products are in the current Level 1 and Level 2 pipeline products.

2 Processing Details

The relevant CIAO programs for generating grating responses are:

`punlearn`**:** initialize parameters of a CIAO program to their defaults;

`pset`**:** set parameters in a CIAO parameter file;

`asphist`**:** construct an aspect histogram from a set of `asol` files;

`mkgrmf`**:** compute a grating response matrix file (RMF);

`mkgarf`**:** compute a grating auxiliary response file (ARF) for a detector subsystem (CCD or chip);

`dmarfadd`**:** combine the detector subsystem ARFs into the final ARF.

We will assume that a `punlearn` (or the equivalent in a non-CIAO-shell implementation) initializes each command's parameters to its defaults, and we will then give differences from those explicitly in the command description:

---

[1] An implementation exists in the "`mk_tgresp`" CIAO contributed script, which could be adapted as appropriate for pipeline use.

[2] Some observations have been made with HETG/HRC-I. But processing requires customization, and some `CALDB` files do not exist. This mode may be added later after CIAO and `CALDB` revisions provide support.

```
punlearn ardlib
punlearn asphist
punlearn mkgrmf
punlearn mkgarf
punlearn dmarfadd
```

While there are large similarities in commands between the instrument modes, we will describe each one separately as needed to make differences explicit. There are also similarities for different orders and different detector elements, and these we will abbreviate with a "repeat for..." description.

For the purposes of illustration, we will assume that files exist in the current directory.

For ACIS-S response generation, there is no difference in commands between timed exposure (TE) or continuus clocking (CC).

The overall procedure is as follows:

1. The zeroth order sky `x,y` coordinates for the extracted spectrum (`pha` file) are read from the `REGION` block of the `evt2` file. (`dmlist`, or equivalent).
2. The list of active detector subsystems is determined from the `DETNAM` header keyword in the `evt2` file.
3. The zeroth-orders's detector subsystem (chip or CCD) is determined from the observational configuration (in the `evt2` header) and the zeroth order sky coordinates. (`dmcoords` or equivalent).
4. An aspect histogram is made for each active detector subsystem (ACIS CCD, or HRC chip) (`asphist`).
5. A grating response matrix is made for each grating type and diffraction order, referenced to the zeroth-orders's subsystem (`mkgrmf`).
6. A grating ARF is made for each active subsystem, for each grating type and diffraction order (`mkgarf`).
7. The subsystem ARFs are then combined to make an ARF for each grating and order (`dmarfadd`).

Intermediate files (sub-system ARFs, aspect histograms, '@"-file lists) can be deleted, and the final ARFs and RMFs (per order, per grating type) will be archived as "primary" Level 2 files.

2.1 Pipeline Standard Product Input Files

We will refer to the pipeline product (Level 1 or 2) input files in shorthand generically by their type, rather than the long form of the name. For example, instead of `acisf01790N005_evt2.fits`, we will simply refer to `evt2`, the File Identifier[3]. Files we use as inputs are:

---

[3] See the Data Products List at `http://asc.harvard.edu/cda/DataProdList.html`

evt2: events;

asol1: aspect solution;

bpix1: bad pixels;

msk1: mask;

pha2: spectrum;

dtf1: live time.

2.2 HETG/ACIS-S

ACIS-S is comprised of 6 CCD subsystem devices, of which 4 to 6 are enabled for grating observations (depending on observatory thermal conditions and the observer's preference). They are referred to as `S0` – `S5`, which also map to `CCD_ID` = 4 – 9. The event file header indicates the active CCDs in the `DETNAM` keyword, such as `ACIS-56789`, which means that the sub-systems named `ACIS-S1` (`CCD_ID = 5`) through `ACIS-S5` (`CCD_ID = 9`) are active, but `ACIS-S0` (`CCD_ID = 4`) is not.

The processing is as follows:

1. Read the zeroth order position from the `REGION` extension of the event file, and save in variable $x0, y0$.

   Example (but not required) implementations are:

   ```
   (x0,y0)=fits_read_col( "evt2[REGION][tg_part==0]", "x",  "y" );
   ```
   or
   ```
   x0=`dmlist "evt2[REGION][tg_part=0][cols x]" data,clean | tail -1`
   y0=`dmlist "evt2[REGION][tg_part=0][cols y]" data,clean | tail -1`
   ```

2. Read the `DETNAM` from the `evt2` header and parse into as list of detector subsystems and `CCD_ID` pairs. The mapping is as follows, but not all need be present:

   ```
   subsys CCD_ID
   ACIS-S0   4
   ACIS-S1   5
   ACIS-S2   6
   ACIS-S3   7
   ACIS-S4   8
   ACIS-S5   9
   ```

3. Determine the zeroth order's subsys from the zeroth order coordinate and the observational configuration defined by the `evt2` header. One way to do this is via:

   ```
   dmcoords mode=hl infile=evt2 option=sky x=$x0 y=$y0
   pget dmcoords chip_id
   ```

and then mapping that ID to the sybsys in the above table. We will call the result `ZO_SUBSYS` for reference later; it is nominally `ACIS-S3`, but could be `ACIS-S2` for some offset pointings (or serendipitous sources).

4. Compute an aspect histograms for each subsystem. Since there may be multiple `asol1` files, first make a list in a "@"-file.

   ```
   ls pcad*asol1.fits > asol.list
   ```

   The following is the repeated for each pair of corresponding `CCD_ID $m` and subsystem index `$k`:

   ```
   asphist \
     infile='@asol.list' \
     evtfile='evt2[CCD_ID=$m]' \
     dtffile='evt2' \
     max_bin='40000' \
     outfile='S$k.asphist'
   ```

   (nominally producing `S0.asphist, S1.asphist, ..., S6.asphist`).

5. Compute grating response matrices (RMFs), given the zeroth order's subsystem, `ZO_SUBSYS`, for each grating type, MEG and HEG, and for each of the diffraction orders, -3 to $+3$ (excluding 0):

   ```
   mkgrmf \
     obsfile='evt2[EVENTS]' \
     regionfile='pha2[region]' \
     detsubsys=$ZO_SUBSYS \
     grating_arm='HEG' \
     order='-3' \
     outfile='heg_-3.rmf'
     ...
   mkgrmf \
     obsfile='evt2[EVENTS]' \
     regionfile='pha2[region]' \
     detsubsys=$ZO_SUBSYS \
     grating_arm='HEG' \
     order='3' \
     outfile='heg_3.rmf'
   ```

   The above is then repeated for MEG, and we produce 12 grating response matrix files: 6 MEG and 6 HEG (6 orders for grating type). (For specification of standardized file names see § 3; the generic names will be used below to make the ARF files.)

6. Compute a set of grating ARFs. First we make a set a files for each subsystem, per order and grating type. Here we also have to first set the bad pixel file name in an auxiliary parameter file:

   ```
   pset ardlib \
     AXAF_ACIS4_BADPIX_FILE='bpix1[BADPIX4]' \
     AXAF_ACIS5_BADPIX_FILE='bpix1[BADPIX5]' \
   ```

```
  AXAF_ACIS6_BADPIX_FILE='bpix1[BADPIX6]' \
  AXAF_ACIS7_BADPIX_FILE='bpix1[BADPIX7]' \
  AXAF_ACIS8_BADPIX_FILE='bpix1[BADPIX8]' \
  AXAF_ACIS9_BADPIX_FILE='bpix1[BADPIX9]'
```

(We need not worry about whether the detector subsystem is active, since that badpix file will not be referenced later.)

To make the ARFs, we first specify the order and grating type, then loop over the relevant detector subsystems. Not all orders fall on all CCDs, we only use those order and subsystems which have non-zero response. Negative orders lie below or equal to `ZO_SUBSYS`, and positive orders are on subsystems equal to or greater than `ZO_SUBSYS`. Nominally, with `ZO_SUBSYS` =ACIS-S3, this means negative orders require subsystems ACIS-S0, ACIS-S1, ACIS-S2, and ACIS-S3, while positive orders require ACIS-S3, ACIS-S4, and ACIS-S5. (There will be warnings for some orders and some subsystems, that the response was zero for all bins; this is expected.)

To compute the ARFs, we need to refer to the zeroth order position retrieved earlier, to the corresponding subsystem aspect histogram, and to the corresponding RMF.

The following example for HEG order -3 wis repeated for a loop over subsystems, incrementing the last 3 parameters appropriately, as shown:

```
mkgarf \
  obsfile='evt2[EVENTS]' \
  maskfile='msk1' \
  pbkfile='none' \
  sourcepixelx=$x0 \
  sourcepixely=$y0 \
  grating_arm='HEG' \
  engrid='grid(heg_-3.rmf)' \
  order='-3' \
  asphistfile='S0.asphist[ASPHIST]' \
  detsubsys='ACIS-S0' \
  outfile='S0_heg_-3.arf'

mkgarf \
  obsfile='evt2[EVENTS]' \
  maskfile='msk1' \
  pbkfile='none' \
  sourcepixelx=$x0 \
  sourcepixely=$y0 \
  grating_arm='HEG' \
  engrid='grid(heg_-3.rmf)' \
  order='-3' \
  asphistfile='S1.asphist[ASPHIST]' \
  detsubsys='ACIS-S1' \
  outfile='S1_heg_-3.arf'

mkgarf \
  obsfile='evt2[EVENTS]' \
  maskfile='msk1' \
```

```
  pbkfile='none' \
  sourcepixelx=$x0 \
  sourcepixely=$y0 \
  grating_arm='HEG' \
  engrid='grid(heg_-3.rmf)' \
  order='-3' \
  asphistfile='S2.asphist[ASPHIST]' \
  detsubsys='ACIS-S2' \
  outfile='S2_heg_-3.arf'

mkgarf \
  obsfile='evt2[EVENTS]' \
  maskfile='msk1' \
  pbkfile='none' \
  sourcepixelx=$x0 \
  sourcepixely=$y0 \
  grating_arm='HEG' \
  engrid='grid(heg_-3.rmf)' \
  order='-3' \
  asphistfile='S3.asphist[ASPHIST]' \
  detsubsys='ACIS-S3' \
  outfile='S3_heg_-3.arf'
```

We then repeat for each of the remaining orders (-2, -1, 1, 2, 3), and then repeat all for grating type MEG.

7. Combine the subsystem ARF components into one ARF per order and grating type (that is, 6 orders for MEG, 6 orders for HEG, in direct correspondence to the set of RMFs). We do this by creating a "@"-file and the running `dmarfadd`. For example, for HEG -1 order:

```
echo 'S0_heg_-1.arf
  S1_heg_-1.arf
  S2_heg_-1.arf
  S3_heg_-1.arf' > heg_-1_garf.list

dmarfadd \
  infile='@heg_-1_garf.list' \
  outfile='heg_-1.arf'
```

The above is repeated for remaining orders, and then all again for MEG.

Temporary files can now be deleted, and the ARFs and RMFs renamed to standard CXC conventions (see § 3).

2.3 LETG/ACIS-S

Processing of LETG/ACIS-S data is very similar to that for HETG/ACIS-S. The main difference from the steps in § 2.2 is that there is only one grating type to consider, "LEG". Steps 1–4 are identical to those in § 2.2. In steps 5 and 6, we require `grating_arm='LEG'` in `mkgrmf` and `mkgarf`, and temporary output files tagged with `leg_*`. In `mkgarf`, we also use “`leg`” in the `engrid` parameter. We still require orders -3 to +3 (excluding 0), and same active detector subsystem handling.

2.4 LETG/HRC-S

The LETG/HRC-S case differs from that for ACIS-S in a few ways: detector subsystems are referred to by `CHIP_ID` (not `CCD_ID`), the zeroth order subsystem is always `CHIP_ID=2`, we require orders -8 to +8 (excluding 0). Only one grating type, `LEG`, is relevant.

To be explicit, we repeat the descriptions as in § 2.2 with appropriate changes for LETG/HRC-S.

1. Read the zeroth order position from the `REGION` extension of the event file (same as in § 2.2).
2. Read the `DETNAM` from the `evt2` header, which should only be `HRC-S`. Define the subsystem names and corresponding `CHIP_ID`:

```
subsys CHIP_ID
HRC-S1   1
HRC-S2   2
HRC-S3   3
```

3. Set the zeroth order's subsystem as `HRC-S2`.
4. Compute an aspect histograms for each subsystem, `HRC-S1`, `HRC-S2`, and `HRC-S3`. Since there may be multiple `asol1` files, first make a list in a “@”-file.

```
ls pcad*asol1.fits > asol.list
```

The following is the repeated for each pair of corresponding `CHIP_ID $m` (which is identical here to the subsystem numerical index):

```
asphist \
  infile='@asol.list' \
  evtfile='evt2[CHIP_ID=$m]' \
  dtffile='dtf1' \
  max_bin='40000' \
  outfile='S$m.asphist'
```

(producing `S1.asphist, S2.asphist, and S3.asphist`).

5. Compute grating response matrices (RMFs) for each of the diffraction orders, -8 to $+8$ (excluding 0):

```
mkgrmf \
  obsfile='evt2[EVENTS]' \
  regionfile='pha2[region]' \
  detsubsys=HRC-S2 \
  grating_arm='LEG' \
  order='-8' \
  outfile='leg_-8.rmf'
  ...
mkgrmf \
  obsfile='evt2[EVENTS]' \
  regionfile='pha2[region]' \
  detsubsys=HRC-S2 \
  grating_arm='LEG' \
  order='8' \
  outfile='leg_8.rmf'
```

This produces 16 grating response matrix files for the set of orders. The generic names will be used below to make the ARF files. (For standardized names, see § 3.)

6. Compute a set of grating ARFs. First we make a set a files for each subsystem, per order. Here we also have to first set the bad pixel file name in an auxiliary parameter file:

   ```
   pset ardlib  AXAF_HRC-S_BADPIX_FILE='bpix1'
   ```

   To make the ARFs, we first specify the order, then loop over the detector subsystems. Not all orders fall on all chips, we only use those order and subsystems which have non-zero response. Negative orders lie on `HRC-S3` and `HRC-S2`, while positive orders are on subsystems `HRC-S2` and `HRC-S1`.

   To compute the ARFs, we need to refer to the zeroth order position retrieved earlier, to the corresponding subsystem aspect histogram, and to the corresponding RMF.

   The following is repeated for a loop over subsystems, incrementing the last 3 parameters appropriately, as shown:

   ```
   mkgarf \
     obsfile='evt2[EVENTS]' \
     maskfile='msk1' \
     pbkfile='none' \
     sourcepixelx=$x0 \
     sourcepixely=$y0 \
     grating_arm='LEG' \
     engrid='grid(leg_-8.rmf)' \
     order='-8' \
     asphistfile='S3.asphist[ASPHIST]' \
     detsubsys='HRC-S3' \
   ```

```
  outfile='S3_leg_-8.arf'

mkgarf \
  obsfile='evt2[EVENTS]' \
  maskfile='msk1' \
  pbkfile='none' \
  sourcepixelx=$x0 \
  sourcepixely=$y0 \
  grating_arm='LEG' \
  engrid='grid(leg_-8.rmf)' \
  order='-8' \
  asphistfile='S2.asphist[ASPHIST]' \
  detsubsys='HRC-S2' \
  outfile='S2_leg_-8.arf'
```

For a representative positive order:

```
mkgarf \
  obsfile='evt2[EVENTS]' \
  maskfile='msk1' \
  pbkfile='none' \
  sourcepixelx=$x0 \
  sourcepixely=$y0 \
  grating_arm='LEG' \
  engrid='grid(leg_1.rmf)' \
  order='1' \
  asphistfile='S2.asphist[ASPHIST]' \
  detsubsys='HRC-S2' \
  outfile='S2_leg_1.arf'

mkgarf \
  obsfile='evt2[EVENTS]' \
  maskfile='msk1' \
  pbkfile='none' \
  sourcepixelx=$x0 \
  sourcepixely=$y0 \
  grating_arm='LEG' \
  engrid='grid(leg_1.rmf)' \
  order='1' \
  asphistfile='S1.asphist[ASPHIST]' \
  detsubsys='HRC-S1' \
  outfile='S1_leg_1.arf'
```

We then repeat for each of the remaining orders,

7. Combine the subsystem ARF components into one ARF per order We do this by creating a “@”-file and the running `dmarfadd`. For example, for -1 order:

```
echo 'S2_leg_-1.arf
```

```
  S1_leg_-1.arf > leg_-1_garf.list

dmarfadd \
  infile='@leg_-1_garf.list' \
  outfile='leg_-1.arf'
```

The above is repeated for remaining orders.

Temporary files can now be deleted, and the ARFs and RMFs renamed to standard CXC conventions (see § 3).

3 File Names and Types

File names for the ARFs and RMFs will follow the CXC conventions for the Chandra Archive Data Products[4]. We will use the optional field to encode the grating type and order and "File Indentifier" to indicate the contents as an ARF or RMF. Hence the generic form of the filename is

```
<instrument>f<obsid>N<rev>_<opt>_<fid>.fits
```

where "instrument" is `acis` or `hrc`, the "obsid" is a 5-digit field for the observation identifier, and "rev" is a 3-digit revision number. The fields defining the response are "opt" which will be given as

```
<Gsm>
```

Where the 1-character fields are:

`G` to have the value `H`, `M`, or `L` to indicate the grating types HEG, MEG, or LEG, respectively;

`s` indicates the sign of the order and is either `p` or `m` to indicate whether the response is for a positive or negative order, respectively; and

`m` is an integer giving the absolute value of the diffraction order.

The `fid` value will be either `rmf2` or `arf2` for the response matrix (RMF), or ancillary response file (ARF; effective area), respectively, in which the digit "2" refers to a Level 2 Pipeline product.

For example, some of the file names for Observation IDs 1843 or 3479 responses would look like:

```
acisf01843N003_Hm1_arf2.fits    HEG -1 order ARF
acisf01843N003_Hp1_rmf2.fits    HEG +1 order RMF
acisf01843N003_Mm3_arf2.fits    MEG -3 order ARF

hrcf03479N003_Lm8_arf2.fits     LEG -8 order ARF
hrcf03479N003_Lp1_rmf2.fits     LEG +1 order RMF
```

---

[4] `http://asc.harvard.edu/cda/DataProdList.html`

For Chandra Archive purposes, the files can be categorized as Primary Products with the following attributes:

Level: L2
Proprietary: Y
Pipeline : TG2

Instrument	Data Product	Content	Identifier
ACIS	Response	SPECRESP	arf2
ACIS	Response	RSP_MATRIX	rmf2
HRC	Response	SPECRESP	arf2
HRC	Response	RSP_MATRIX	rmf2

4 Size and File Number Estimates

The file size and number per observation for each instrumental configuration are as follows:

Instrument	Size	Number
HETG/ACIS-S		Orders -3 to +3 (except 0)
MEG ARF	250K	6
HEG ARF	250K	6
MEG RMF	7M	6
HEG RMF	20M	6
Total	165M	24
LETG/ACIS-S		Orders -3 to +3 (except 0)
LEG ARF	250K	6
LEG RMF	7M	6
Total	43.5M	12
LETG/HRC-S		Orders -8 to +8 (except 0)
LEG ARF	470K	16
LEG RMF	14M	16
Total	231.5M	32