Part II

CHANDRA X-RAY OBSERVATORY

CIAO Analysis

Nicholas Lee
SAO/*Chandra* X-ray Center
Science Data Systems

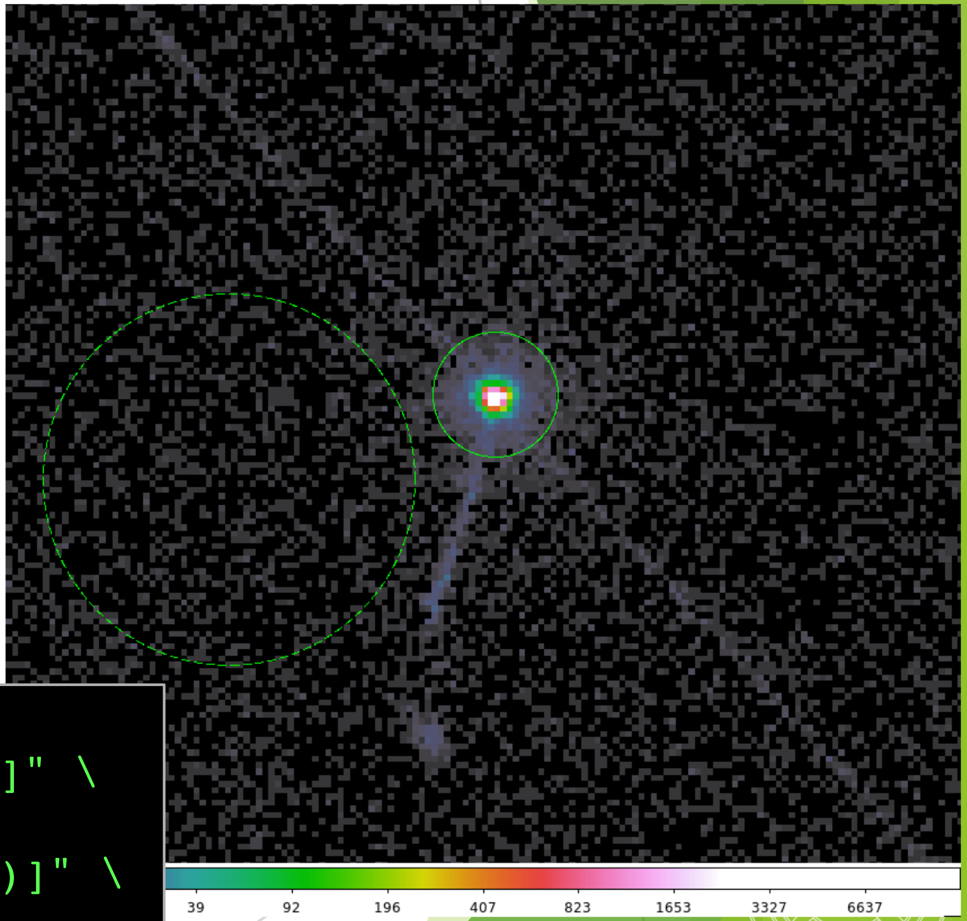
The 2nd AAS Chandra/CIAO Workshop—Honolulu, Hawai'i, January 3-4, 2020

CENTER FOR ASTROPHYSICS
HARVARD & SMITHSONIAN

Extracting Spectrum from an Imaging Spectroscopy Observation

- `specextract` extracts spectrum and calculates corresponding responses
  - background products optional
- extract background or not
  - point source: how much brighter is the source than the local background?
  - extended source and crowded fields: can be critical, but also non-trivial to extract
  - if planning on fitting background spectrum, create background responses

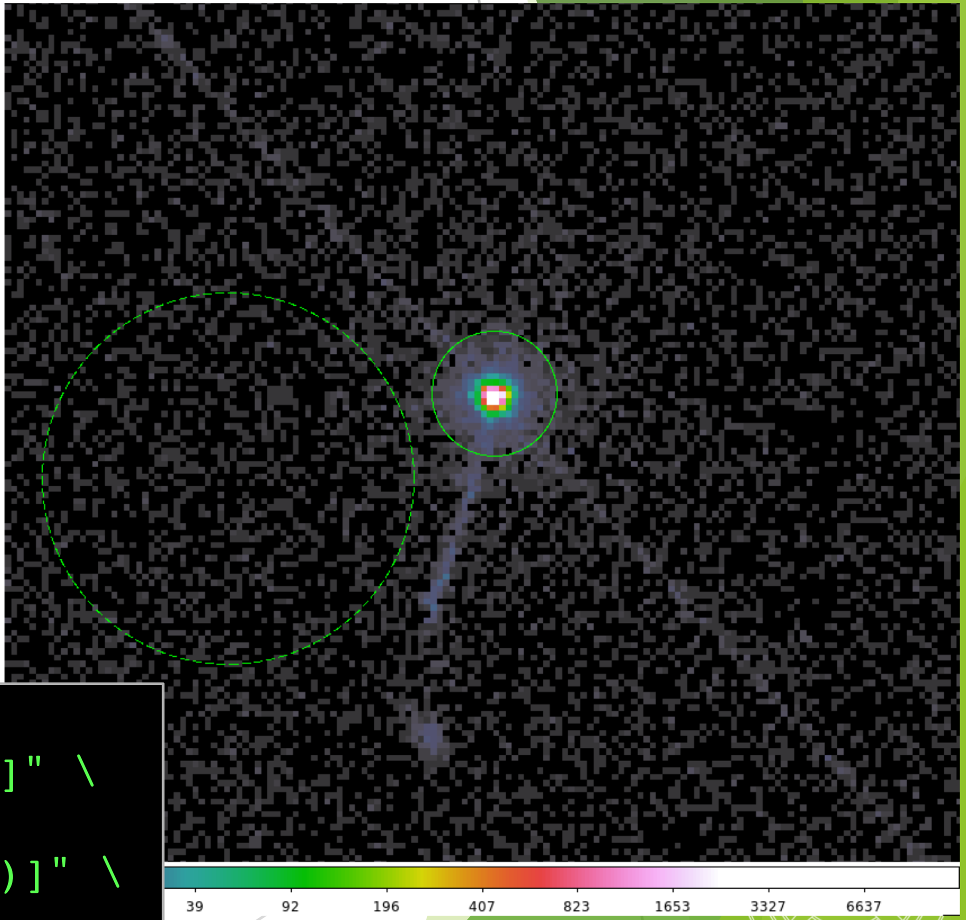
```
unix% specextract
infile="acisf07302_repro_evt2.fits[sky=region(src.reg)]" \
outroot=spec/7302_core \
bkgfile="acisf07302_repro_evt2.fits[sky=region(bkg.reg)]" \
bkgresp=yes weight=no correctpsf=yes grouptype=NONE \
mode=h clobber=yes
```



Extracting Spectrum (cont.)

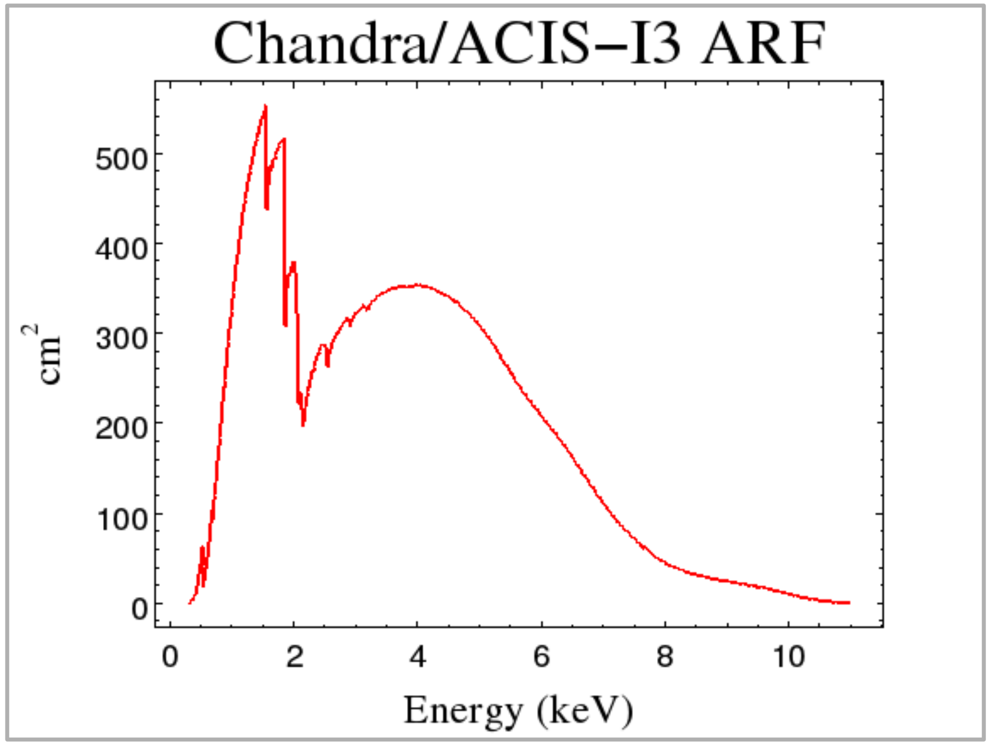
from an Imaging Spectroscopy Observation

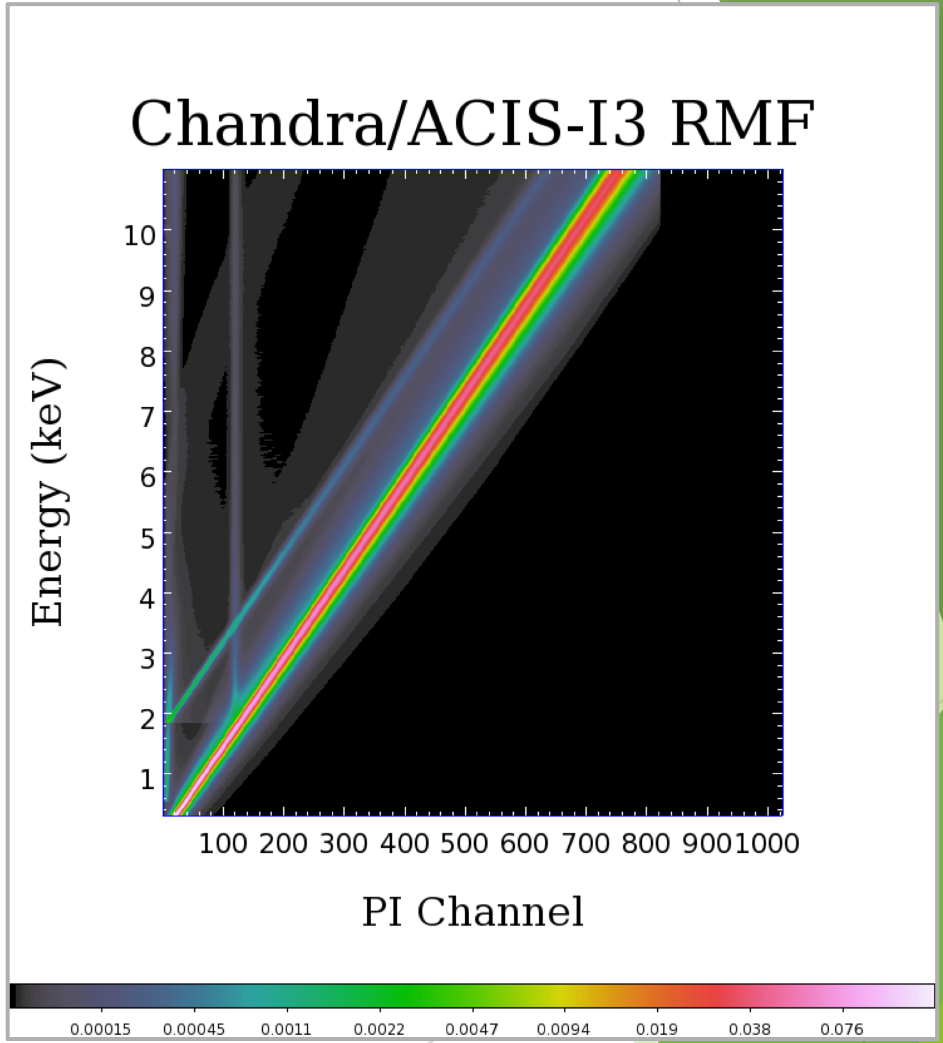
- unweighted vs weighted responses
  - on-axis point sources, unweighted responses
    - correct ARF for events that fall outside the aperture
  - extended and far off-axis point sources, weighted responses
    - weighted ARFs are needed if interested in the spatial variation of the effective area
    - weighted RMFs are computationally expensive, scaling with the number of pixels in the extraction region, but the probability variation with spatial position is small
  - point sources near chip gaps should use weighted responses, since it accounts for affects of source dithering off detector



```
unix% specextract
infile="acisf07302_repro_evt2.fits[sky=region(src.reg)]" \
outroot=spec/7302_core \
bkgfile="acisf07302_repro_evt2.fits[sky=region(bkg.reg)]" \
bkgresp=yes weight=no correctpsf=yes grouptype=NONE \
mode=h clobber=yes
```

Spectral Responses





X-ray Spectral Fitting Packages

- *Sherpa* is the spectral fitting package native to CIAO.
  - **XSpec** is the gold standard in X-ray astronomy for 1D spectral fitting
  - **Sherpa** designed to fit *n*-dimensional data sets and can be used beyond X-ray spectra and 2D image fitting; Python-based and available as a standalone package
  - **ISIS** (*Interactive Spectral Interpretation System*) is optimized for gratings analysis; S-Lang interpreter
  - **SPEX** has many unique non-equilibrium, collisional ionization and plasma models; source code made public in the past year
- All packages designed to solve:

$$C(h) = t\int_0^\infty R(E,h)A(E)S(E)dE + B(h)$$

and in practice, discretized as:

$$C(h) = t\sum_i R_{i,h}A_iS(E_i)\Delta E_i + B(h)$$

where $C(h)$ is the observed counts in a spectrum at detector channel h; t is the exposure time; $R(E,h)$ is the probability of observing a photon of energy E at channel h represented by the dimensionless RMF; $A(E)$ is the effective area and QE encapsulated in the ARF; $S(E)$ is the source model; and $B(h)$ is the observed background counts at channel h.

- Models are fit by the iterative technique of *forward-folding.*

Spectral Fitting: Matrix Inversion

- Directly inverting the integral in $C(h)$ is not mathematically possible due to the non-diagonality of RMFs, so there is no *unique* inversion.
  - simplify the matrix equation, assuming $i = 1$ and $\boldsymbol{R_h}A = \boldsymbol{\mathcal{R}}$, so that:

$$C(h) = t\sum_i R_{i,h}A_i S(E_i)\Delta E_i + B(h) \longrightarrow \boldsymbol{c} = t\boldsymbol{\mathcal{R}}\boldsymbol{s} + \boldsymbol{b}$$

    - Some missions use a response (RSP) file in lieu of the individual ARF and RMF response files, which is a matrix of the product of the RMF and ARF of an observation, $\boldsymbol{\mathcal{R}}$, in this example.
  - invert matrix to solve for $\boldsymbol{s}$?

$$\boldsymbol{s} = \frac{\boldsymbol{\mathcal{R}}^{-1}(\boldsymbol{c}-\boldsymbol{b})}{t}$$

    - there is noise in $(\boldsymbol{c}-\boldsymbol{b})$ and because the form of $\boldsymbol{\mathcal{R}}$ (generally a rather broad redistribution, with significant off-diagonal contribution) does not permit a mathematically unique inversion
    - very unstable to small perturbations (noise), even if there is an unique solution

CHANDRA X-RAY OBSERVATORY

In a Perfect World…

matrix inversion or direct fitting *could* work

- Matrix inversion works if the RMF is perfectly diagonal (where the eigenvalues are non-zero).
  - diagonal RMF $\Rightarrow$ negligible line spread; direct mapping of photons of a given energy being detected in a specific channel
  - sometimes found in high-resolution gratings spectroscopy
- Ignore $\mathcal{R}$ and directly fit spectral features.
  - requires narrow energy band and high-resolution spectra
  - this is what's done in O/IR astronomy. While the detected wavelength/energy is an instrumental quantity, and the spectral lines are broadened by instrumental effects, since the line spread function is narrow, it's possible to neglect the instrument blurring and directly fit a physical model, accepting the introduced uncertainties.
  - typical CCD/CZT spectra has insufficient energy resolution (ACIS: ΔE~14.6 eV$\longrightarrow\Delta\lambda$~850 Å) which would lead to many incorrect fit parameters, since it would be impossible to pin down the energy distribution of the observed photon counts within that energy bin

CENTER FOR ASTROPHYSICS
HARVARD & SMITHSONIAN

In a Perfect World…

matrix inversion or direct fitting *could* work, but…

Since the energy resolution of the detectors are fairly broad and cover a much larger range than optical/IR detectors, spectral modeling in these energy regimes require models to be "forward-fitted" — that is, the theoretical model is convolved with response files calculated from calibration products, and tweaked until the best fit to the observed data is found.

CHANDRA
X-RAY OBSERVATORY

Spectral Fitting: Forward-Folding

- When we forward-fold…
  - make an educated guess for the model
    - convolve with response to get predicted counts per spectral bin:

$$M(h) \propto \int_{E_{lo}(h)}^{E_{hi}(h)} R(E,h)A(E)S(E;\boldsymbol{p})dE$$

    from the earlier integral where $\boldsymbol{p}$ are the source model parameters
  - minimize a statistic (typically χ^2) formed from the difference between the data and model
    - optimization routine will iteratively vary parameters to search for a minimum in the statistic

$$\underbrace{\chi^2}_{\text{statistic to minimize}} = \sum_h \left(\frac{\overbrace{N(h)}^{\text{observed total counts}} - \left(\overbrace{M(h)}^{\text{model counts}} + \overbrace{B(h)}^{\text{background counts}}\right)}{\underbrace{\sigma(h)}_{\text{uncertainties}}}\right)^2$$

    - the uncertainties, σ, may be derived from the statistics of both $N(h)$ and $B(h)$
  - alternately, maximize the probability or likelihood

CENTER FOR ASTROPHYSICS
HARVARD & SMITHSONIAN

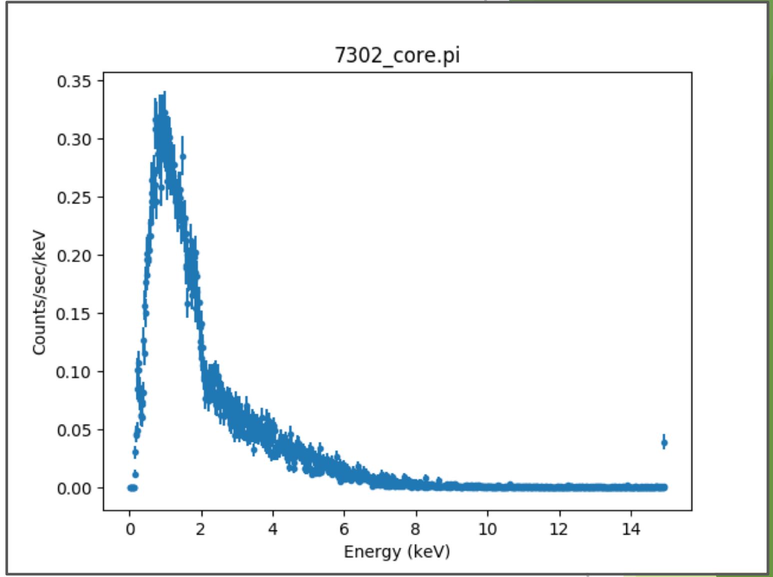
Sherpa: Load and Filter Data

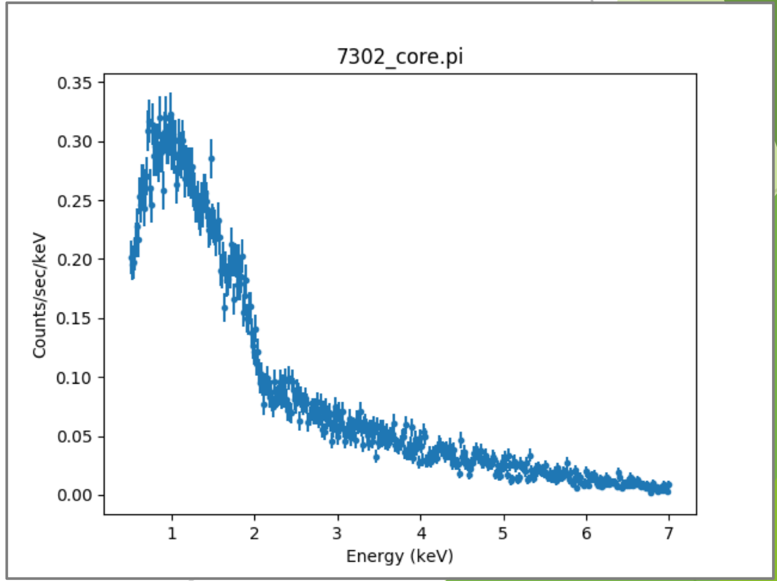
```
sherpa> load_data("7302_core.pi")
```

ARF, RMF, background, and background responses automatically loaded if defined in header keywords and can be found.

```
sherpa> plot_data()

sherpa> show_filter()
Data Set Filter: 1
0.0110-14.9431 Energy (keV)

sherpa> notice(0.5,7.0)
sherpa> show_filter()
Data Set Filter: 1
0.5037-7.0007 Energy (keV)

sherpa> plot_data()
```

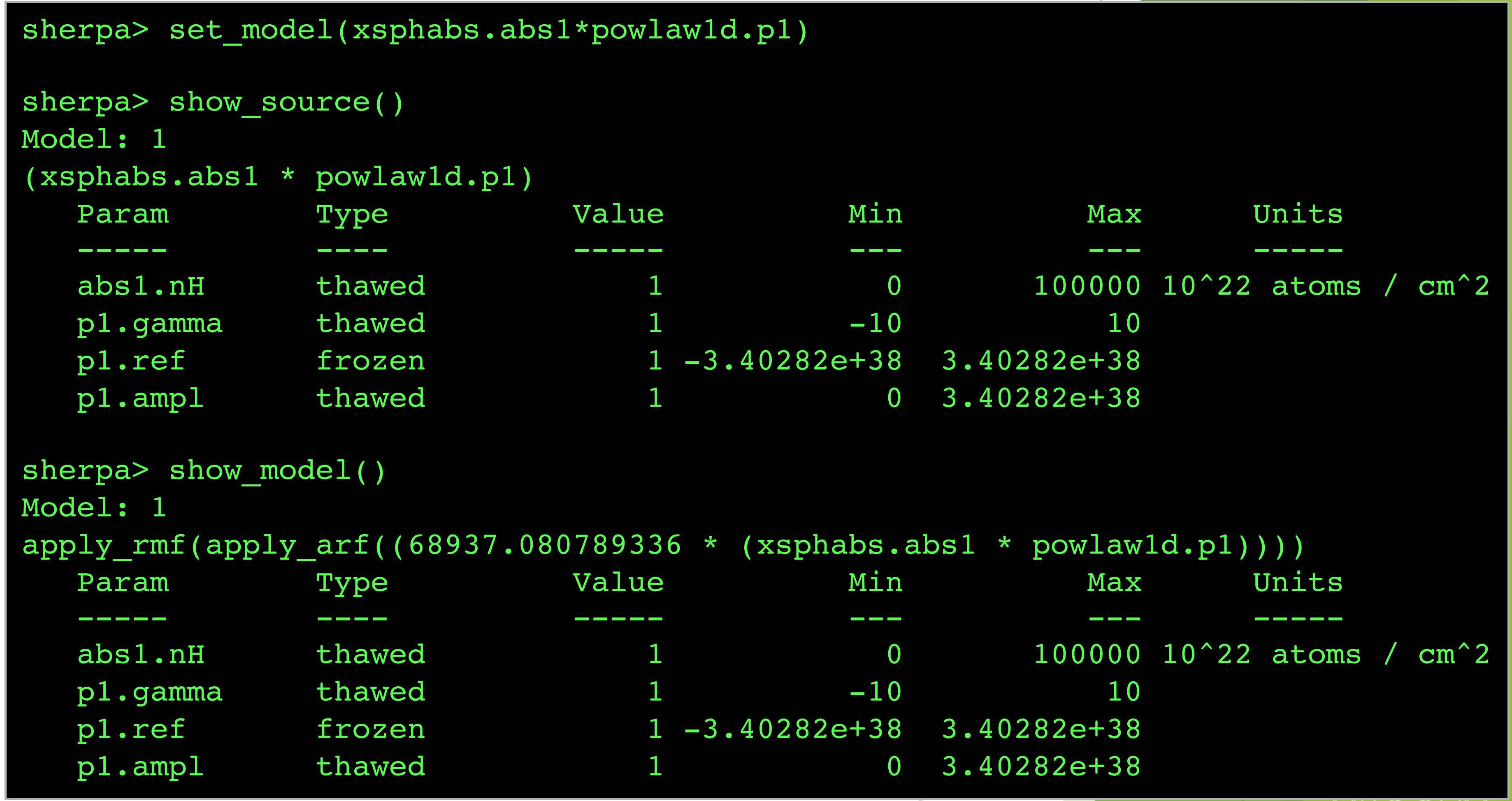
The filter ranges are ultimately determined by the bin edges of the grid that were used to create the response files.





Sherpa: "source" vs. "model"

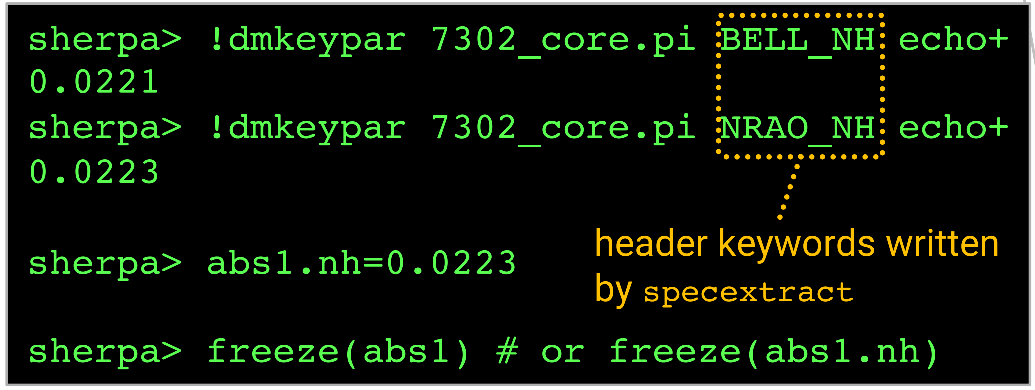
- "source" is the $S(E)$ in the equation solved by software; it is the physical model describing the source.
- "model" refers to the source convolved with the responses and scaled by various terms, including exposure time.



```
sherpa> set_model(xsphabs.abs1*powlaw1d.p1)

sherpa> show_source()
Model: 1
(xsphabs.abs1 * powlaw1d.p1)
   Param        Type          Value          Min          Max      Units
   -----        ----          -----          ---          ---      -----
   abs1.nH      thawed            1            0       100000 10^22 atoms / cm^2
   p1.gamma     thawed            1          -10           10
   p1.ref       frozen            1 -3.40282e+38  3.40282e+38
   p1.ampl      thawed            1            0  3.40282e+38

sherpa> show_model()
Model: 1
apply_rmf(apply_arf((68937.080789336 * (xsphabs.abs1 * powlaw1d.p1))))
   Param        Type          Value          Min          Max      Units
   -----        ----          -----          ---          ---      -----
   abs1.nH      thawed            1            0       100000 10^22 atoms / cm^2
   p1.gamma     thawed            1          -10           10
   p1.ref       frozen            1 -3.40282e+38  3.40282e+38
   p1.ampl      thawed            1            0  3.40282e+38
```

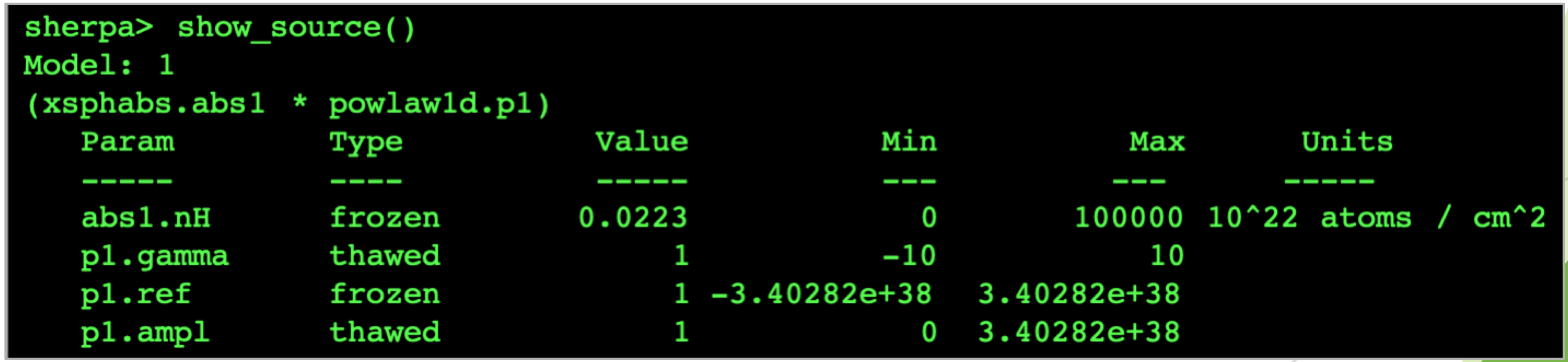
Sherpa: Model Component Parameters

- model components are represented with the model objects **abs1** and **p1**.
- `freeze` and `thaw` entire model component or specific component parameters.
- provide reasonable initial parameter values or use `guess`.



```
sherpa> !dmkeypar 7302_core.pi BELL_NH echo+
0.0221
sherpa> !dmkeypar 7302_core.pi NRAO_NH echo+
0.0223

sherpa> abs1.nh=0.0223

sherpa> freeze(abs1) # or freeze(abs1.nh)
```

header keywords written by `specextract`



```
sherpa> show_source()
Model: 1
(xsphabs.abs1 * powlaw1d.p1)
   Param        Type          Value          Min          Max      Units
   -----        ----          -----          ---          ---      -----
   abs1.nH      frozen       0.0223            0       100000 10^22 atoms / cm^2
   p1.gamma     thawed            1          -10           10
   p1.ref       frozen            1 -3.40282e+38  3.40282e+38
   p1.ampl      thawed            1            0  3.40282e+38
```

Sherpa: Model Component Parameters

Parameter Limits

- "soft" limits restrict the range of parameter-space explored…
- Note: many XSpec models have liberal default limits that are set without any regard to what the model code and documentation claim to allow, which can affect the model behavior, or placing the limits in non-physical regimes.

```
sherpa> p1.gamma.min = -5
sherpa> p1.gamma.max = 5
```

Statistics and Optimization Methods

`cxc.harvard.edu/sherpa/methods/` and `cxc.harvard.edu/sherpa/statistics/`

- χ^2 and [Poissonian] maximum likelihood statistics
- Optimization Methods — minimization of a function
  - Levenberg-Marquardt — quick but very sensitive to initial parameters and easily trapped in local extrema; works well for simple models, but fails to converge on complex models.
  - Nelder-Mead = Simplex — robust exploration of parameter-space, converges with complex models.

  'single shot'

  - Monte Carlo — global search of parameter-space and converges on complex models, very slow.
  - `gridsearch` used for template models, slow.

  'scatter shot'

```
sherpa> list_stats()
['cash’, 'chi2', 'chi2constvar’, 'chi2datavar’, 'chi2gehrels',
 'chi2modvar', 'chi2xspecvar', 'cstat', 'leastsq', 'userstat’, 'wstat’]

sherpa> list_methods()
['gridsearch', 'levmar', 'moncar', 'neldermead', 'simplex']

sherpa> set_stat("wstat")
sherpa> set_method("neldermead")
```

CHANDRA X-RAY OBSERVATORY

Statistics Choice for Forward-Folding

the Conventional Approaches

For the observed net counts in bin h, $C(h)$, then $C(h) = N(h) - B(h)$ where $N(h)$ is the observed total counts and $B(h)$ is the observed background counts in bin h. The convolved source model, $M(h)$, is then iteratively compared with $C(h)$ until the difference is minimized (or alternatively maximizing the probability/likelihood).

- use χ^2 statistics
  - bin the observed spectrum so there are ~10–20 counts per bin (`group_counts`) so that Gaussian statistics apply (i.e., uncertainty in spectral bin h is $\sigma(h) \longrightarrow \frac{1}{\sqrt{N(h)}}$)
  - directly subtract background
- use Poisson statistics
  - unbinned spectrum
  - ignore or model background
- hybrid of the above two
  - include observed background, but as part of the model, $M(h)$
  - assume Poisson statistics

CENTER FOR ASTROPHYSICS HARVARD & SMITHSONIAN

Optimization

to minimize a function

A general function $f(x;p)$ may have many isolated local minima, non-isolated minimum hypersurfaces, or even more complicated topologies. No finite minimization routine is guaranteed to locate the unique, global, minimum of $f(x;p)$ without being fed detailed knowledge about the function by the user.

Good model fits are dependent on the initial model parameters.
An educated, well informed, initial parameter guess can be critical to success!

Therefore:

1. Never accept the result using a single optimization run; always test the minimum using a different method.
2. Check that the result of the minimization does not have parameter values at the edges of the parameter space. If this happens, then the fit must be disregarded since the minimum lies outside the space that has been searched, or the minimization missed the minimum.
3. Get a feel for the range of values of the fit statistic, and the stability of the solution, by starting the minimization from several different parameter values.
4. Always check that the minimum "looks right" using a plotting tool.

Optimization

Method Comparisons

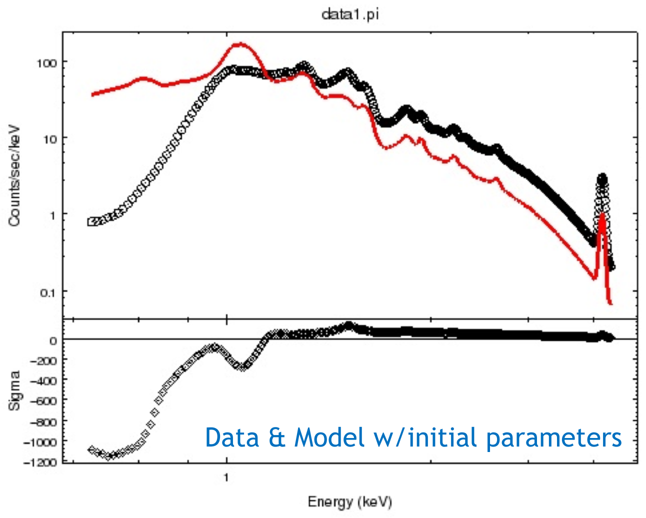
Method	N_{iter}	Final Statistics
LevMar	31	1.55e5
NelderMead	1494	0.0542
MonCar	13045	0.0542

Example: spectral fit with three methods

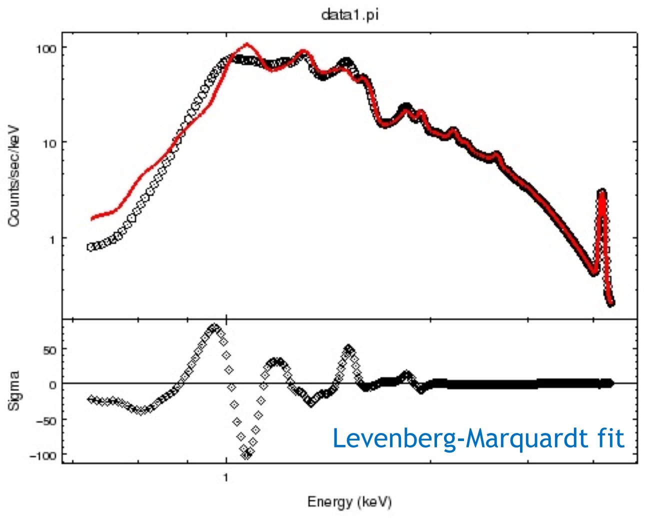
Data—high S/N simulated ACIS-S spectrum of the two temperature plasma

Model—photoelectric absorption plus two MEKAL components (correlated!)

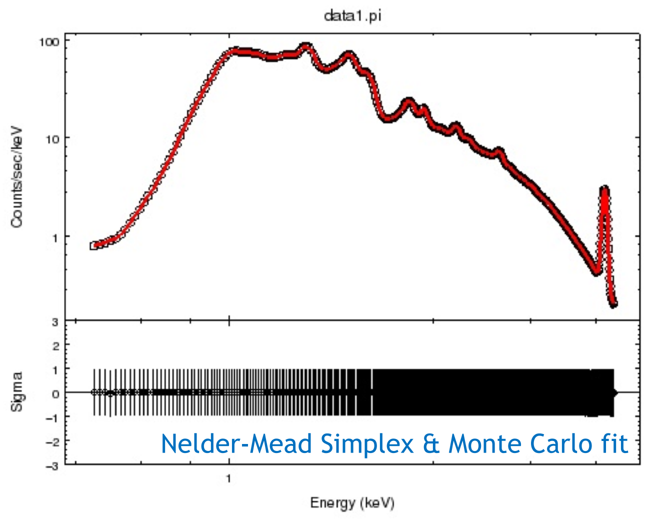
Start fit from the same initial parameters. Figures and Table compares the efficiency and final results



Data & Model w/initial parameters



Levenberg-Marquardt fit

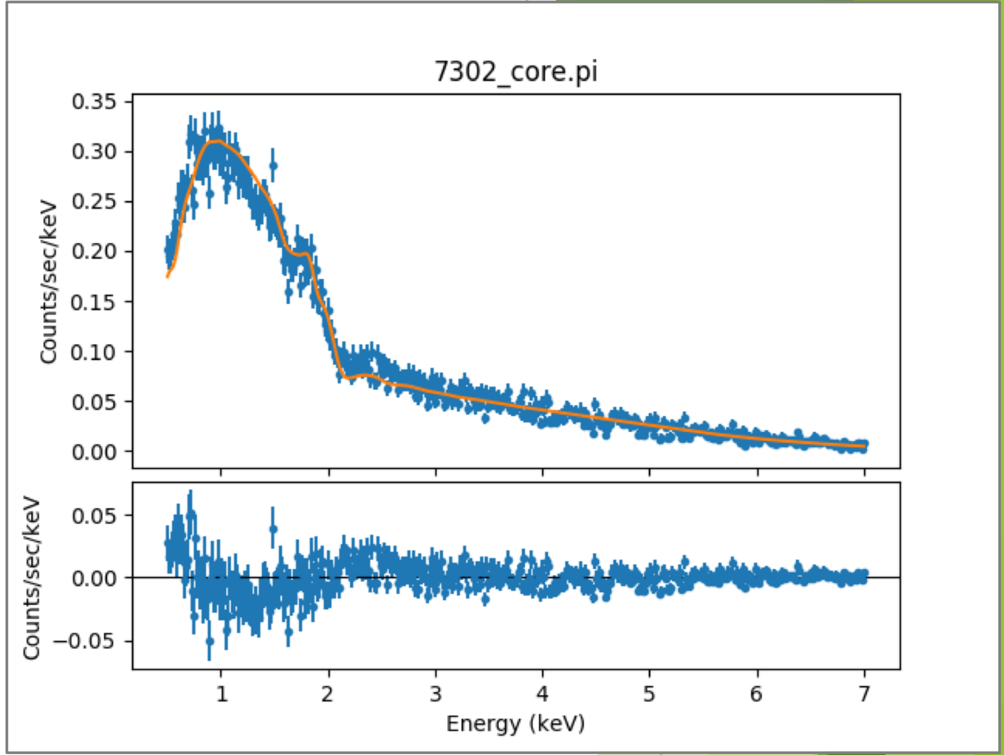


Nelder-Mead Simplex & Monte Carlo fit

Sherpa: Fitting and Residuals

- $\texttt{resid} = data - model$
- $\texttt{delchi} = \delta\chi = \sigma = \frac{data - model}{error}$
  - only available with χ^2 statistics

```
sherpa> fit()
Dataset               = 1
Method                = neldermead
Statistic             = wstat
Initial fit statistic = 1.32552e+08
Final fit statistic   = 641.185 at function evaluation 321
Data points           = 446
Degrees of freedom    = 444
Probability [Q-value] = 2.39751e-09
Reduced statistic     = 1.44411
Change in statistic   = 1.32551e+08
   p1.gamma        1.32099
   p1.ampl         0.000682765

sherpa> plot_fit_resid()
```



reduced statistic $\longrightarrow 1$, good fit
reduced statistic < 1, unexpectedly good fit
reduced statistic $\gg 1$, insufficient data points to believe fit

Sherpa: Final Analysis Steps

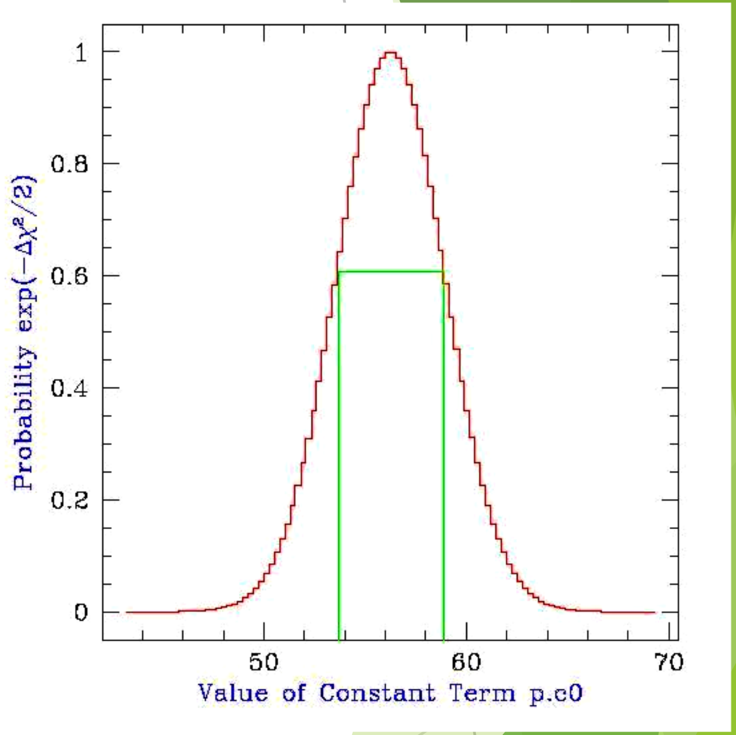
- How well are the model parameters constrained by the data?
- Is this a correct model?
- Is this the only model?
- Do we have definite results?
- What have we learned, discovered?
- How our source compares to the other sources?
- Do we need to obtain a new observation?

Confidence Limits

Essential issue: after the best-fit parameters are found, estimate the confidence limits for them. The region of confidence is given by (Avni 1976):

$$\chi^2_\alpha = \chi^2_{\min} + \Delta(\nu, \alpha)$$

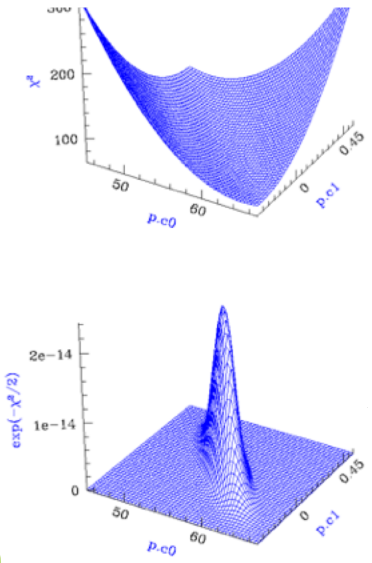
where ν are the degrees of freedom, α is the confidence level, $\chi^2_{\min}$ is the minimum statistics, and $\Delta(\nu, \alpha)$ depends only on the number of parameters involved, not on goodness-of-fit.

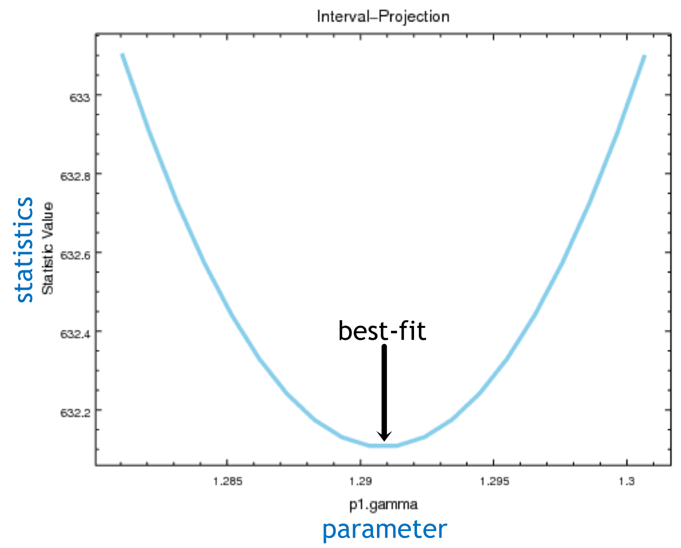


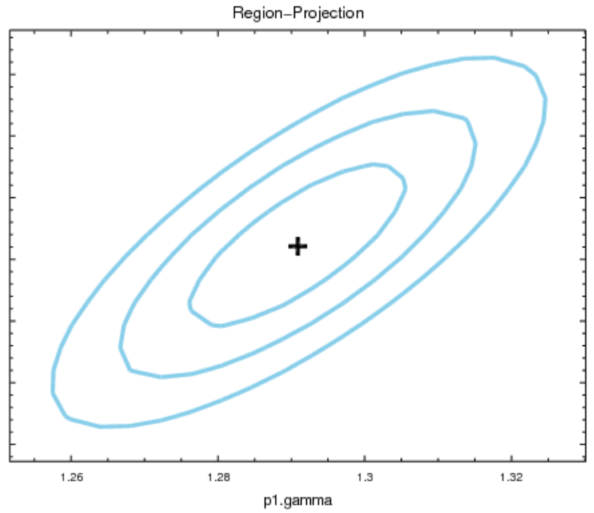
Confidence Regions

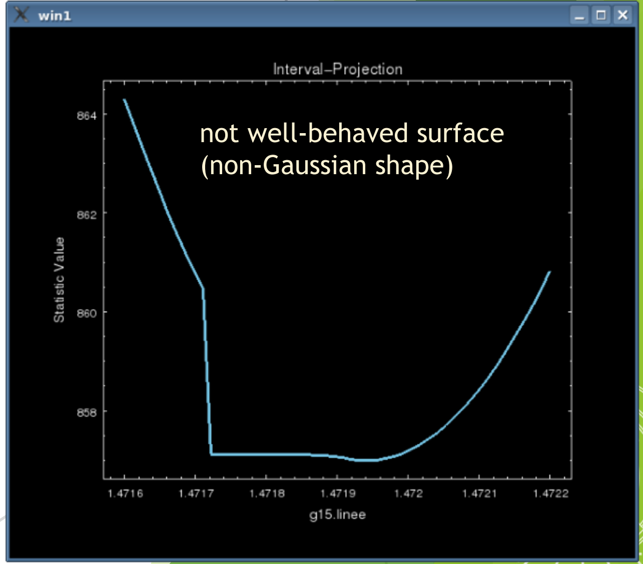
calculating confidence limits means exploring the parameter-space, i.e. the statistical surface

well-behaved statistical surface in parameter-space









not well-behaved surface
(non-Gaussian shape)

Sherpa: Uncertainties on Model Free Parameters and Source Model Fluxes

- Uncertainties on free parameters
  - confidence
  - projection
  - covar
  - reg_proj and reg_unc
- unconvolved model fluxes
  - calc_energy_flux(ID,[lo,hi])
    - $\frac{ergs}{cm^2 \cdot s}$
    - $\frac{ergs}{cm^2 \cdot s \cdot keV}$ or $\frac{ergs}{cm^2 \cdot s \cdot Å}$
  - calc_photon_flux(ID,[lo,hi])
    - $\frac{photons}{cm^2 \cdot s}$
    - $\frac{photons}{cm^2 \cdot s \cdot keV}$ or $\frac{photons}{cm^2 \cdot s \cdot Å}$

```
sherpa> set_conf_opt("sigma",1.6)
sherpa> conf()
p1.gamma lower bound: -0.0133227
p1.ampl lower bound:     -7.18186e-06
p1.gamma upper bound:    0.0133227
p1.ampl upper bound:     7.22921e-06
Dataset               = 1
Confidence Method     = confidence
Iterative Fit Method  = None
Fitting Method        = neldermead
Statistic             = wstat
confidence 1.6-sigma (89.0401%) bounds:
   Param            Best-Fit  Lower Bound  Upper Bound
   -----            --------  -----------  -----------
   p1.gamma          1.32099   -0.0133227    0.0133227
   p1.ampl       0.000682765 -7.18186e-06  7.22921e-06
```

$1.645\sigma \simeq 90\%$ C.I.

```
sherpa> calc_energy_flux(lo=0.5,hi=7.0)
4.943207695012615e-12

sherpa> calc_photon_flux(lo=0.5,hi=7.0)
0.0014460264242070627
```

Model Selection

when competing models fit data equally well

- Hypothesis Testing with nested models [additive components]
  - via goodness-of-fit with *F*-test or likelihood ratio test directly in *Sherpa* and *XSpec*
  - in certain special cases, the chosen test statistic has a probability distribution that is independent of the model being tested so that a *p*-value (significance) can be calculated immediately
    - The *p*-value is defined as the probability that the value of the chosen test statistic be as extreme as observed
      - more generally, the *p*-value is evaluated by performing simulations with synthetic spectra generated assuming the null model and test statistic, *T*, calculated for each simulation
      - the set of *T* values provide an empirical estimate of the probability distribution with the *p*-value being the fraction of the distribution where $T_{\text{simulated}} \geq T_{\text{observed}}$.
  - for *p*-value < 0.05 (the 95% criterion), select more complex model over null model

Model Selection (cont.)

- for non-nested models use "Akaike information criterion", "Bayesian information criterion", or "Deviance information criterion" which are penalized-likelihood criteria methods, the latter being Bayesian approaches.
  - Use AIC for simplicity if using Cash or C-statistic, where given a set of candidate models, the preferred model for the data is the one with the minimum AIC
    - $AIC = 2k - 2\ln\mathcal{L}$, where k is the estimated number of model parameters and $\mathcal{L}$ is the maximum of the likelihood function
    - for small sample size, AIC is modified by correction factor. For univariate model with linear parameter, $AIC_C = AIC + \frac{2k^2+2k}{n-k-1}$; where n is sample size and k is number of parameters
  - For large datasets BIC
    - $BIC = k\ln n - 2\ln\mathcal{L}$, where k is the estimated number of model parameters and $\mathcal{L}$ is the maximum of the likelihood function
- criteria reward goodness-of-fit, but penalizes the number of estimated parameters to discourage over-fitting, since increasing the number of fit parameters in a model will always improve the goodness-of-fit.

Model Selection (cont.)

- hypothesis testing methods (for nested models) directly compares the models to the data
- information criteria compare models relative to one another
  - does not compare models to the data
  - if the set of models are all poor fits to the data, then the test will select the ‘least bad’ bad model

ΔAIC or ΔBIC	Evidence Against Higher Information Criterion
0 to 2	not worth more than a bare mention
2 to 6	positive
6 to 10	strong
>10	very strong

CHANDRA X-RAY OBSERVATORY

What else can `Sherpa` do?

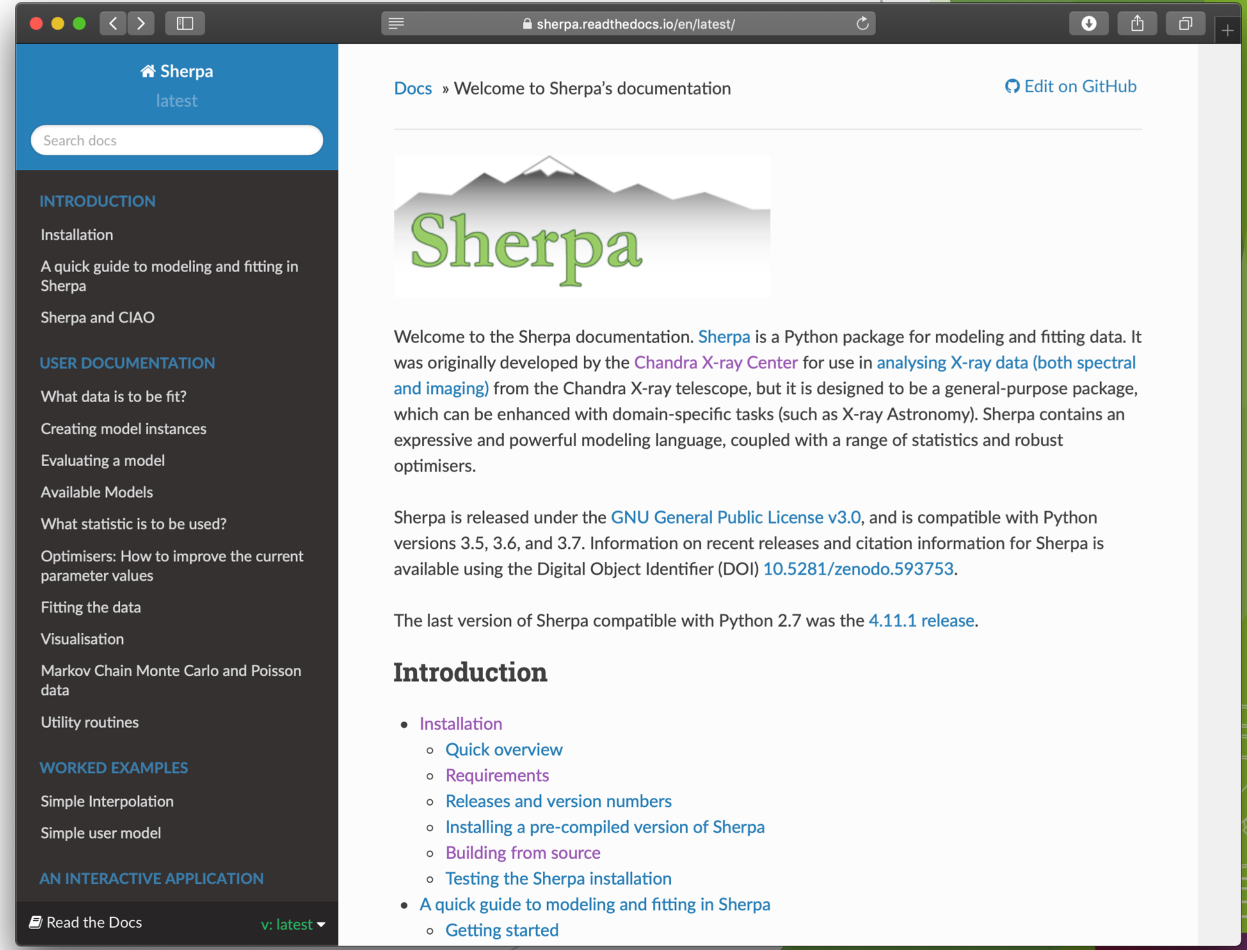
- 2D image fitting with PSF
- Radial profile fitting (e.g. 1D β-function model)
- 1D and 2D ASCII dataset fitting
- Bayesian analysis with priors
  - `get_draws` runs MCMC sampler at a fit's local minimum, with thawed model parameter as priors
    - returns full posterior or posterior profile distribution
    - parameter uncertainties
    - simulate data from the posterior predictive distributions
- extensible to include user-models, -statistics, and -optimizations
- use alongside `pycrates`, `astropy`, and `scipy`
- Source code on GitHub `(https://github.com/sherpa/sherpa)`
  - open to user contributed development

CENTER FOR ASTROPHYSICS HARVARD & SMITHSONIAN

Sherpa: Documentation

- Documentation moving towards `sphinx` for building web documents
  - `https://sherpa.readthedocs.io/`
- Interpreter moving away from `ahelp` and migrating to Python doc strings

```
sherpa> ahelp “toolname”
sherpa> ahelp(“toolname”)
sherpa> help(“docstring”)
```



Welcome to the Sherpa documentation. Sherpa is a Python package for modeling and fitting data. It was originally developed by the Chandra X-ray Center for use in analysing X-ray data (both spectral and imaging) from the Chandra X-ray telescope, but it is designed to be a general-purpose package, which can be enhanced with domain-specific tasks (such as X-ray Astronomy). Sherpa contains an expressive and powerful modeling language, coupled with a range of statistics and robust optimisers.

Sherpa is released under the GNU General Public License v3.0, and is compatible with Python versions 3.5, 3.6, and 3.7. Information on recent releases and citation information for Sherpa is available using the Digital Object Identifier (DOI) 10.5281/zenodo.593753.

The last version of Sherpa compatible with Python 2.7 was the 4.11.1 release.

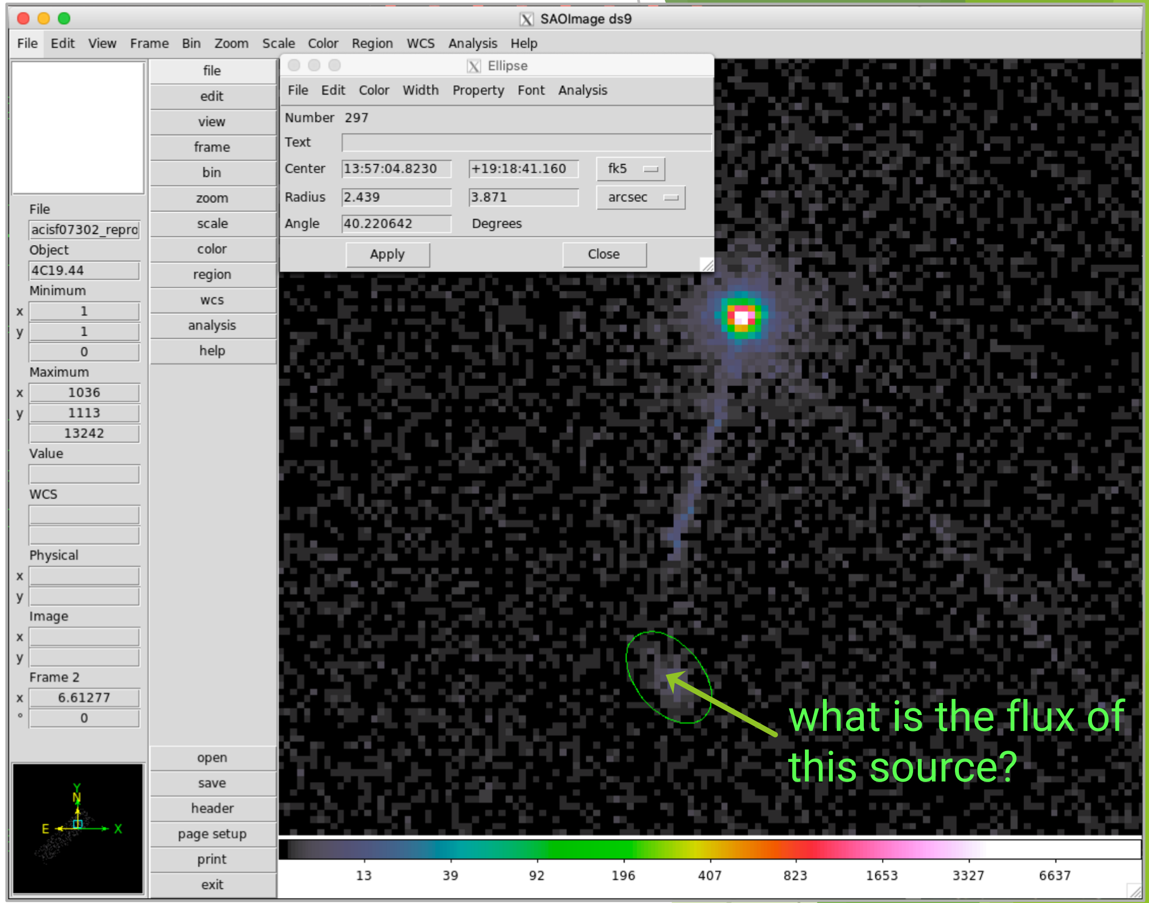
Introduction

- Installation
  - Quick overview
  - Requirements
  - Releases and version numbers
  - Installing a pre-compiled version of Sherpa
  - Building from source
  - Testing the Sherpa installation
- A quick guide to modeling and fitting in Sherpa
  - Getting started

Source Properties

by way of `srcflux`

- Encodes the logic described in six different CIAO threads. Returns count rates, fluxes, and errors with all appropriate corrections.
- `srcflux` capabilities:
  - automatically determines PSF-appropriate extraction region size for source and background, or user-defined
  - uses one of four methods to apply aperture correction
  - runs on multiple energy bands
  - accepts one position or a list
  - calculates count rates using `aprates` method
  - calculates fluxes two different ways (specified spectral model and `eff2evt` method; however, no spectral fit is performed)
  - generates spectral responses for downstream analysis


SAOImage ds9
File Edit View Frame Bin Zoom Scale Color Region WCS Analysis Help
file edit view frame bin zoom scale color region wcs analysis help
open save header page setup print exit
Ellipse
File Edit Color Width Property Font Analysis
Apply Close
N E X Y
13 39 92 196 407 823 1653 3327 6637
what is the flux of this source?

Number 297
Text
Center 13:57:04.8230 +19:18:41.160 fk5
Radius 2.439 3.871 arcsec
Angle 40.220642 Degrees

File acisf07302_repro
Object 4C19.44
Minimum x 1 y 1 0
Maximum x 1036 y 1113 13242
Value
WCS
Physical x y
Image x y
Frame 2 x 6.61277 ° 0

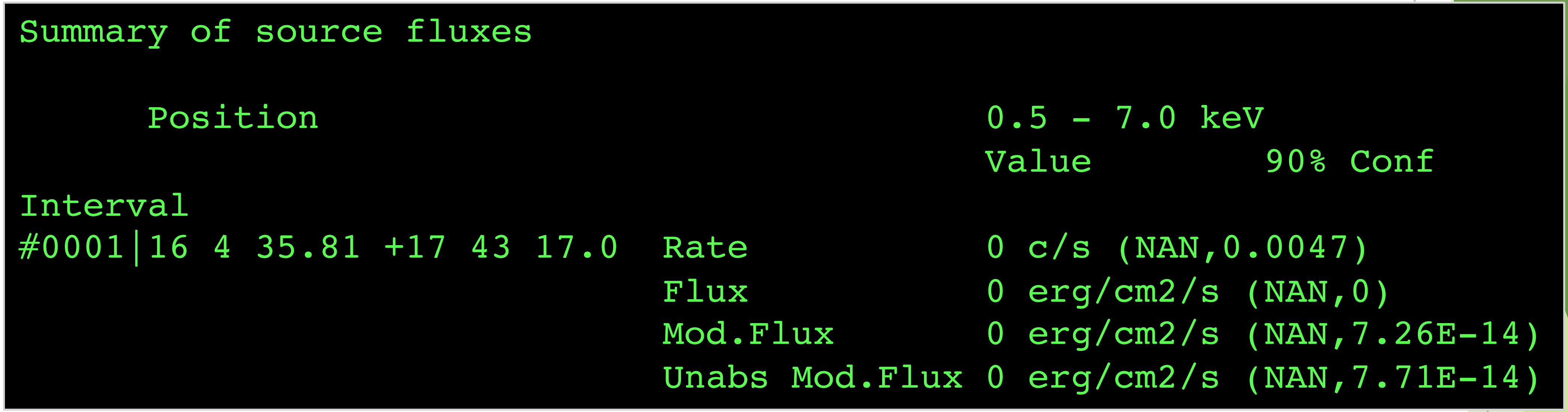
Source Properties (cont.)

by way of `srcflux`

```
unix% srcflux infile=acisf07302_repro_evt2.fits pos="13:57:04.823 +19:18:41.16" \
? outroot=srcflux/lobe mode=h

. . . SCREEN OUTPUT . . .

Summary of source fluxes
      Position                                0.5 - 7.0 keV
                                              Value       90% Conf Interval
#0001|13 57 4.82 +19 18 41.1  Rate           0.000657 c/s (0.000499,0.00084)
                              Flux           6.44E-15 erg/cm2/s (4.9E-15,8.23E-15)
                              Mod.Flux       4.24E-15 erg/cm2/s (3.22E-15,5.42E-15)
                              Unabs Mod.Flux 4.41E-15 erg/cm2/s (3.35E-15,5.64E-15)
```

- `srcflux` has options for PSF corrections, energy bands, confidence intervals (including upper-limits), spectral models, and user supplied regions.
- lower and upper bounds of confidence interval in parentheses.

`srcflux` upper-limits

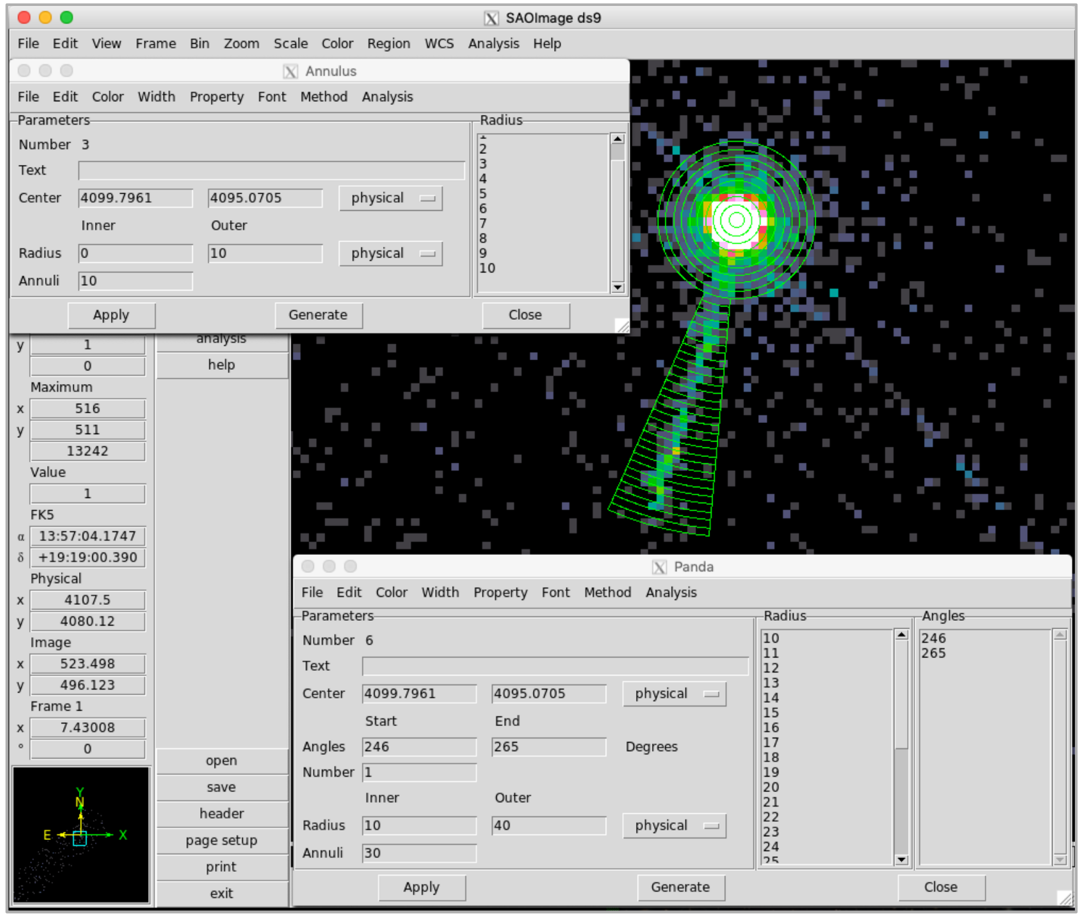


```
Summary of source fluxes

     Position                                 0.5 - 7.0 keV
                                              Value        90% Conf
Interval
#0001|16 4 35.81 +17 43 17.0  Rate            0 c/s (NAN,0.0047)
                              Flux            0 erg/cm2/s (NAN,0)
                              Mod.Flux        0 erg/cm2/s (NAN,7.26E-14)
                              Unabs Mod.Flux  0 erg/cm2/s (NAN,7.71E-14)
```

The upper-limit of a quantity determined by `srcflux` is returned as the upper-bounds of the confidence interval, if the quantity is returned as a zero.

Radial Profiles

- Extract from annular regions with `dmextract`.
  - set `opt=generic`
  - in this example, the background region is the same as the one used for spectral extraction


SAOImage ds9
File Edit View Frame Bin Zoom Scale Color Region WCS Analysis Help
Annulus
File Edit Color Width Property Font Method Analysis
Apply
Generate
Close
Panda
File Edit Color Width Property Font Method Analysis
Apply
Generate
Close
analysis
help
open
save
header
page setup
print
exit
N
E
X
Y

Annulus
Parameters
Number 3
Text
Center 4099.7961 4095.0705 physical
Inner Outer
Radius 0 10 physical
Annuli 10
Radius
2
3
4
5
6
7
8
9
10

y 1
0
Maximum
x 516
y 511
13242
Value
1
FK5
α 13:57:04.1747
δ +19:19:00.390
Physical
x 4107.5
y 4080.12
Image
x 523.498
y 496.123
Frame 1
x 7.43008
° 0

Panda
Parameters
Number 6
Text
Center 4099.7961 4095.0705 physical
Start End
Angles 246 265 Degrees
Number 1
Inner Outer
Radius 10 40 physical
Annuli 30
Radius
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
Angles
246
265

```
unix% cat src.reg
annulus(4099.7961,4095.0705,0,1)
annulus(4099.7961,4095.0705,1,2)
annulus(4099.7961,4095.0705,2,3)
annulus(4099.7961,4095.0705,3,4)
annulus(4099.7961,4095.0705,4,5)
annulus(4099.7961,4095.0705,5,6)
annulus(4099.7961,4095.0705,6,7)
annulus(4099.7961,4095.0705,7,8)
annulus(4099.7961,4095.0705,8,9)
annulus(4099.7961,4095.0705,9,10)
pie(4099.7961,4095.0705,10,11,246,265)
pie(4099.7961,4095.0705,11,12,246,265)
pie(4099.7961,4095.0705,12,13,246,265)
pie(4099.7961,4095.0705,13,14,246,265)
pie(4099.7961,4095.0705,14,15,246,265)
pie(4099.7961,4095.0705,15,16,246,265)
pie(4099.7961,4095.0705,16,17,246,265)
pie(4099.7961,4095.0705,17,18,246,265)
pie(4099.7961,4095.0705,18,19,246,265)
pie(4099.7961,4095.0705,19,20,246,265)
pie(4099.7961,4095.0705,20,21,246,265)
pie(4099.7961,4095.0705,21,22,246,265)
pie(4099.7961,4095.0705,22,23,246,265)
pie(4099.7961,4095.0705,23,24,246,265)
pie(4099.7961,4095.0705,24,25,246,265)
pie(4099.7961,4095.0705,25,26,246,265)
pie(4099.7961,4095.0705,26,27,246,265)
pie(4099.7961,4095.0705,27,28,246,265)
pie(4099.7961,4095.0705,28,29,246,265)
pie(4099.7961,4095.0705,29,30,246,265)
pie(4099.7961,4095.0705,30,31,246,265)
pie(4099.7961,4095.0705,31,32,246,265)
pie(4099.7961,4095.0705,32,33,246,265)
pie(4099.7961,4095.0705,33,34,246,265)
pie(4099.7961,4095.0705,34,35,246,265)
pie(4099.7961,4095.0705,35,36,246,265)
pie(4099.7961,4095.0705,36,37,246,265)
pie(4099.7961,4095.0705,37,38,246,265)
pie(4099.7961,4095.0705,38,39,246,265)
pie(4099.7961,4095.0705,39,40,246,265)

unix% cat radprof_bkg.reg
circle(4057.2756,4081.423,29.742616)
```

CHANDRA X-RAY OBSERVATORY

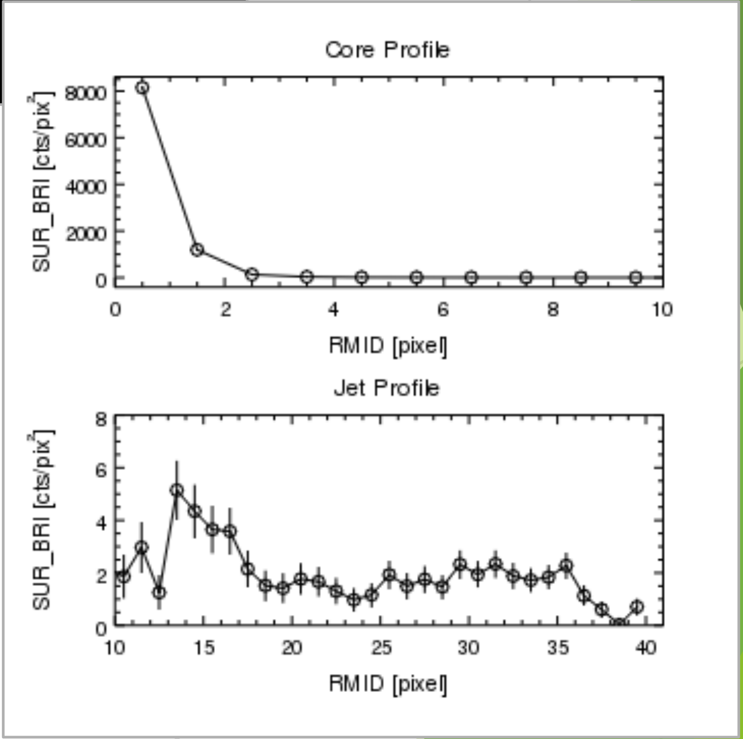
CENTER FOR ASTROPHYSICS HARVARD & SMITHSONIAN

Radial Profiles (cont.)

```
unix% punlearn dmextract

unix% dmextract \
? infile="acisf07302_repro_evt2.fits[bin sky=@radprof.reg]" \
? outfile=7302_corejet.rprof \
? bkg="acisf07302_repro_evt2.fits[bin sky=@radprof_bkg.reg]" \
? opt=generic \
? mode=h clobber=yes
```

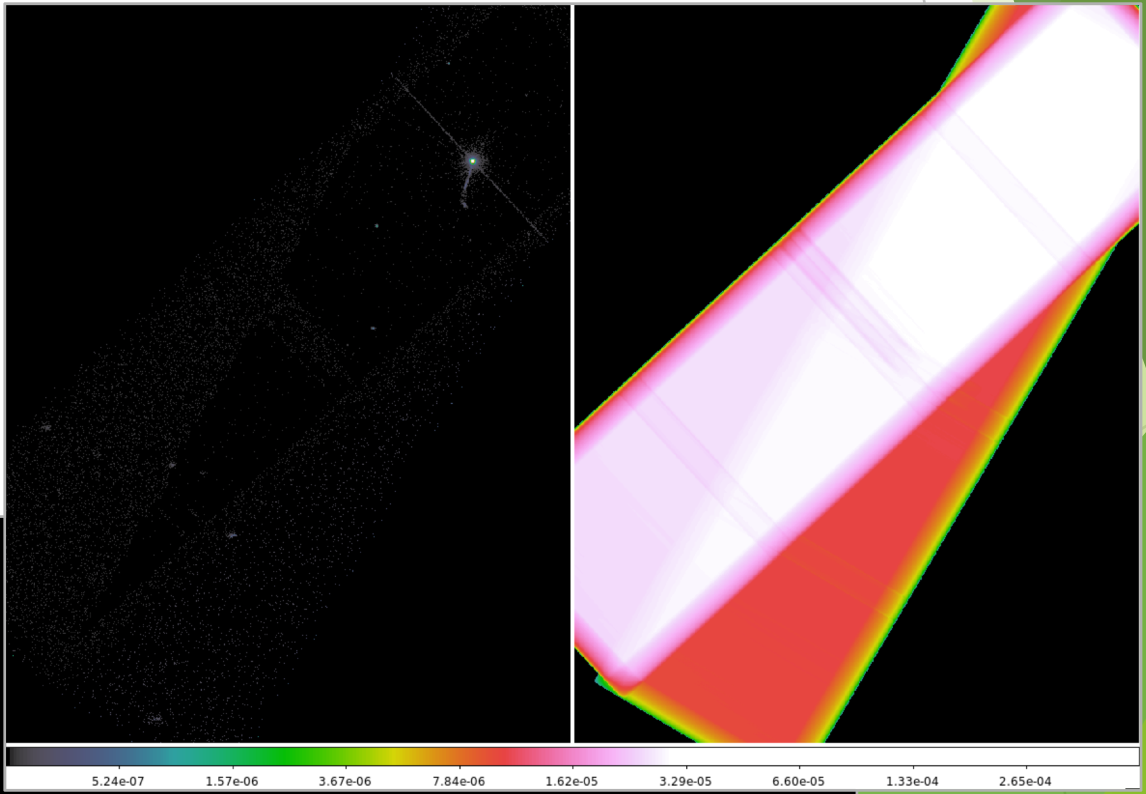
- source and background region files read in as stacks
- prior to CIAO 4.11, would need to calculate `RMID` column with `dmtcalc` which defines the midpoint of the annular regions:

```
unix% punlearn dmtcalc
unix% pset dmtcalc infile=1838_rprofile.fits
unix% pset dmtcalc outfile=1838_rprofile_rmid.fits
unix% pset dmtcalc expression="rmid=0.5*(R[0]+R[1])"
unix% dmtcalc
```



Reprojecting and Co-adding Imaging Data

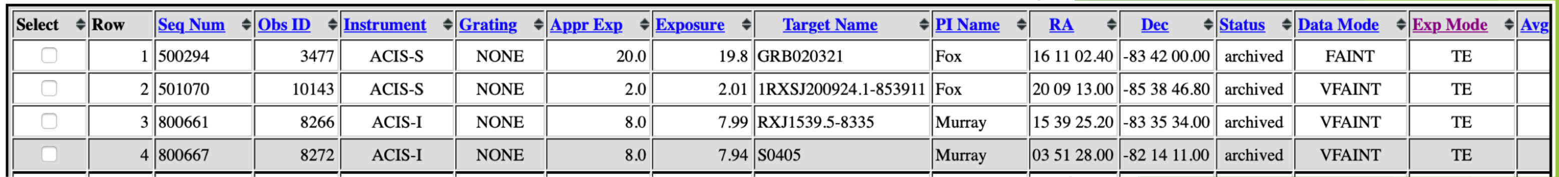
- Combining observations for spatial analysis facilitated by the `merge_obs` script (wrapper around `reproject_obs` and `flux_obs`) using events files.
- Do not use combined events file for spectral extraction.
  - responses vary with time, no calibration products available covering large time spans
  - if observations occur over short period, using the response from a single observation maybe reasonable.
- `dmmerge` used to combine FITS tables.
- `dmimgcalc` used to perform array arithmetic.



```
unix% cat evt2.lis
6903/repro/acisf06903_repro_evt2.fits
6904/repro/acisf06904_repro_evt2.fits
7302/repro/acisf07302_repro_evt2.fits
7303/repro/acisf07303_repro_evt2.fits

unix% merge_obs infiles=@evt2.lis outroot=merged/4C+19.44 bands=broad binsize=1
```

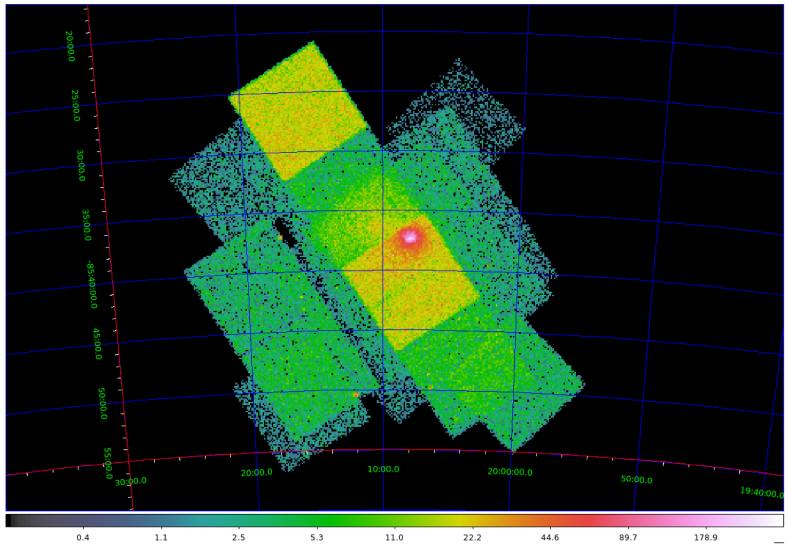
Reprojecting and Co-adding Imaging Data

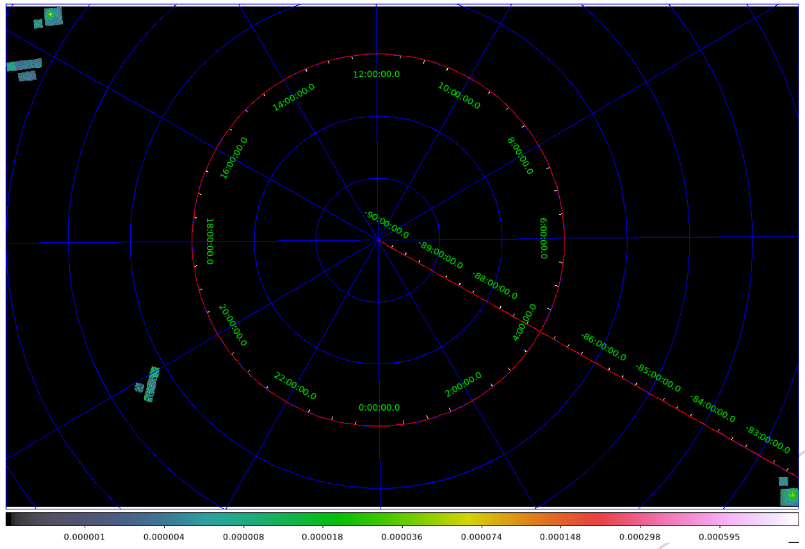
- reprojecting events can be critical to getting correct field location
  - match set of observations to a common tangent point
  - often neglected if observations have similar pointings
- `reproject_image` and `reproject_image_grid` match image pixels between images.



Select	Row	Seq Num	Obs ID	Instrument	Grating	Appr Exp	Exposure	Target Name	PI Name	RA	Dec	Status	Data Mode	Exp Mode	Avg
☐	1	500294	3477	ACIS-S	NONE	20.0	19.8	GRB020321	Fox	16 11 02.40	-83 42 00.00	archived	FAINT	TE	
☐	2	501070	10143	ACIS-S	NONE	2.0	2.01	1RXSJ200924.1-853911	Fox	20 09 13.00	-85 38 46.80	archived	VFAINT	TE	
☐	3	800661	8266	ACIS-I	NONE	8.0	7.99	RXJ1539.5-8335	Murray	15 39 25.20	-83 35 34.00	archived	VFAINT	TE	
☐	4	800667	8272	ACIS-I	NONE	8.0	7.94	S0405	Murray	03 51 28.00	-82 14 11.00	archived	VFAINT	TE	

```
unix% cat evt2.lis
10143/primary/acisf10143N002_evt2.fits.gz
3477/primary/acisf03477N002_evt2.fits.gz
8266/primary/acisf08266N002_evt2.fits.gz
8272/primary/acisf08272N003_evt2.fits.gz

unix% dmmerge infile=@evt.lis outfile=lowlat_bad.fits

unix% merge_obs infiles=@evt2.lis outroot=lowlat_good bands=broad binsize=64
```

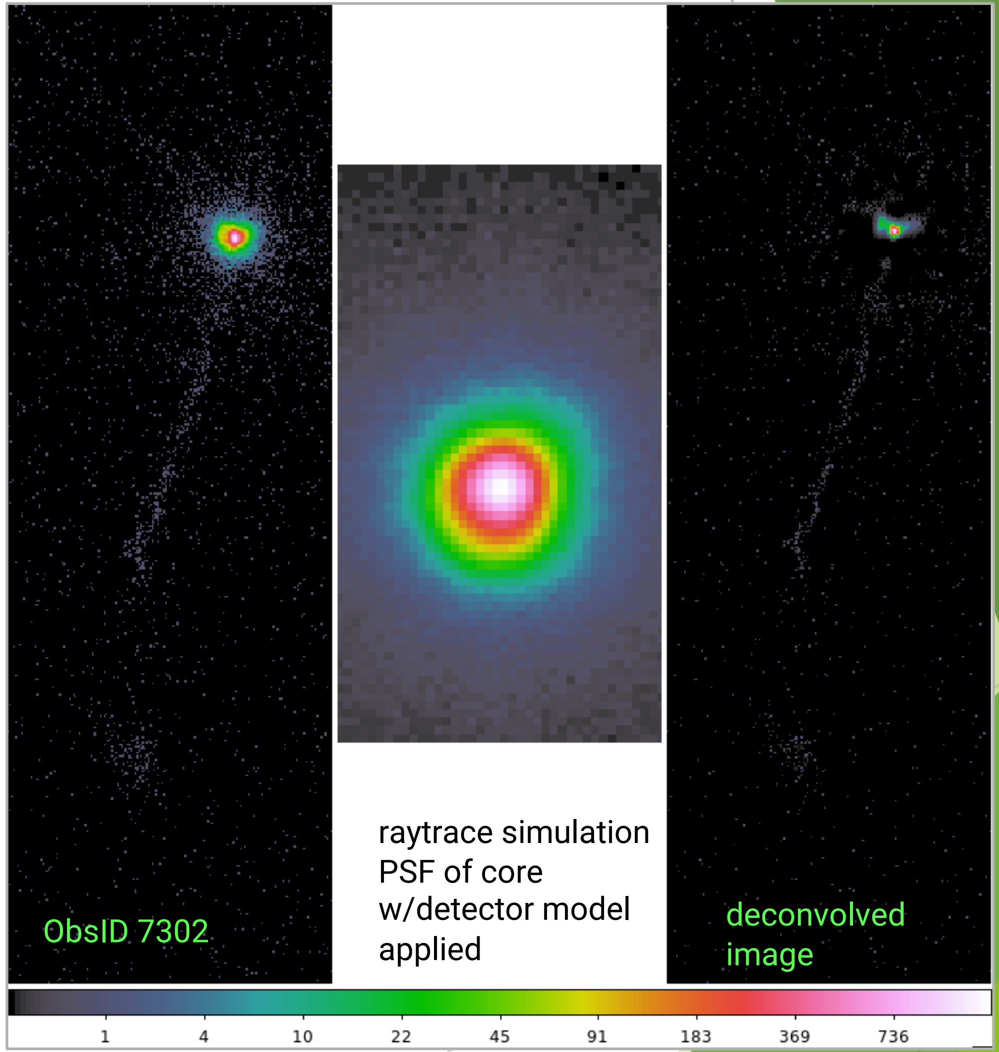




- `reproject_aspect` (wrapper around `wcs_match` and `wcs_update`) used to match source lists and update WCS of images, tables, and asols

Image Smoothing and PSF Deconvolution

- PSF deconvolution
  - Obtain background-subtracted spectrum in ASCII format of the core
  - Use ChaRT/SAOTrace or MARX to simulate PSF
  - Use MARX/`simulate_psf` to project simulated rays on to detector-plane
  - Use `arestore` to deconvolve PSF from observation
- Image smoothing
  - `aconvolve` smooths image with user-defined kernel
  - `csmooth` adaptive image smoothing technique



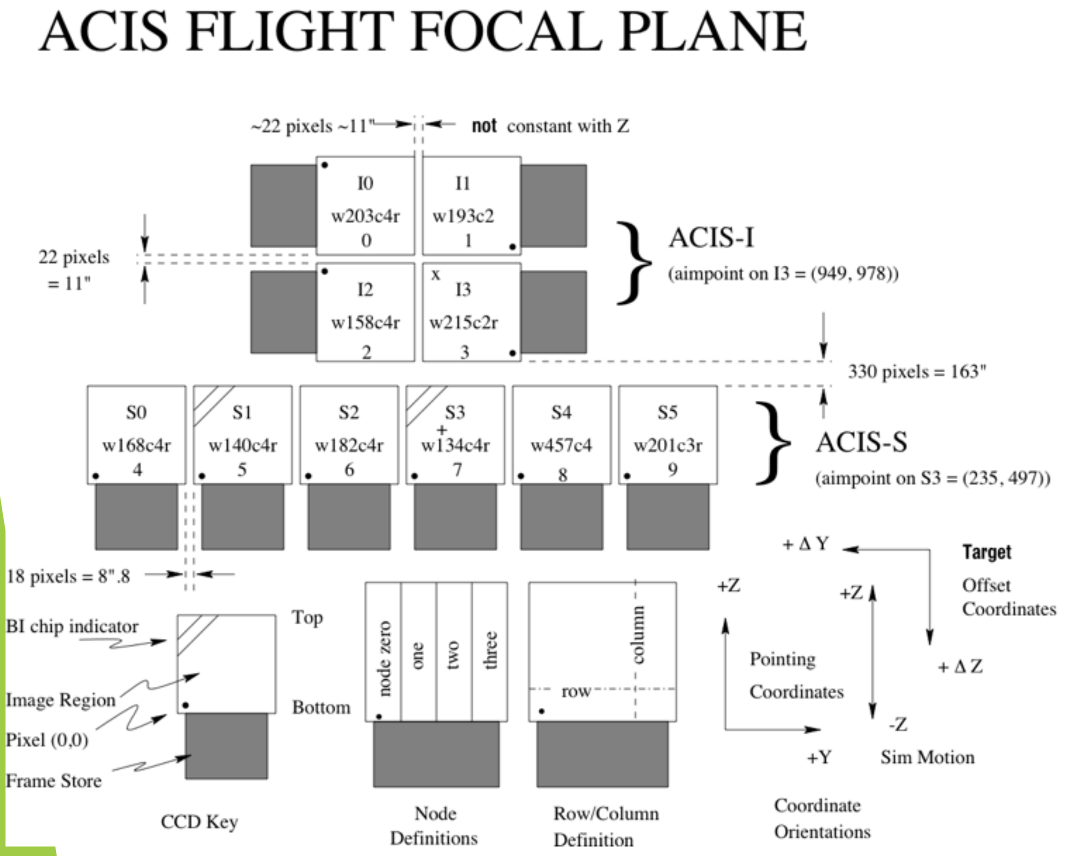
all figures binned to 1/5 an ACIS pixel

Timing Analysis

- light curves
  - `dmextract` with `opt=ltc1` or `opt=ltc2` properly accounts for GTI
  - remember that dither periods are typically 707.1 s and 1000 s for ACIS, 768.6 s and 1087 s for HRC, so beware of variability on those time scales.
- barycentric correction
  - `axbary` corrects all time to a common location, the barycenter
- variability
  - `glvary` is a Bayesian technique based on Gregory-Loredo algorithm that returns an estimate of the most probable light curve from the source, as opposed to what is observed by the telescope and instruments
  - `apowerspectrum` finds $|\mathcal{FFT}|^2$ of a light curve to find the periodicity (or aperiodicity) of variable source by looking for peaks in the power spectrum.

Finally, a gentle reminder:

How can the CCD_ID be mistaken in an observation? It's really easy to, especially if only the ACIS-I array is used…


ACIS FLIGHT FOCAL PLANE
~22 pixels ~11"
not constant with Z
I0 w203c4r 0
I1 w193c2 1
I2 w158c4r 2
I3 w215c2r 3
ACIS-I
(aimpoint on I3 = (949, 978))
22 pixels = 11"
330 pixels = 163"
S0 w168c4r 4
S1 w140c4r 5
S2 w182c4r 6
S3 w134c4r 7
S4 w457c4 8
S5 w201c3r 9
ACIS-S
(aimpoint on S3 = (235, 497))
18 pixels = 8".8
BI chip indicator
Image Region
Pixel (0,0)
Frame Store
Top
Bottom
CCD Key
node zero
one
two
three
Node Definitions
column
row
Row/Column Definition
+ΔY
Target Offset Coordinates
+ΔZ
+Z
Pointing Coordinates
+Y
+Z
-Z
Sim Motion
Coordinate Orientations