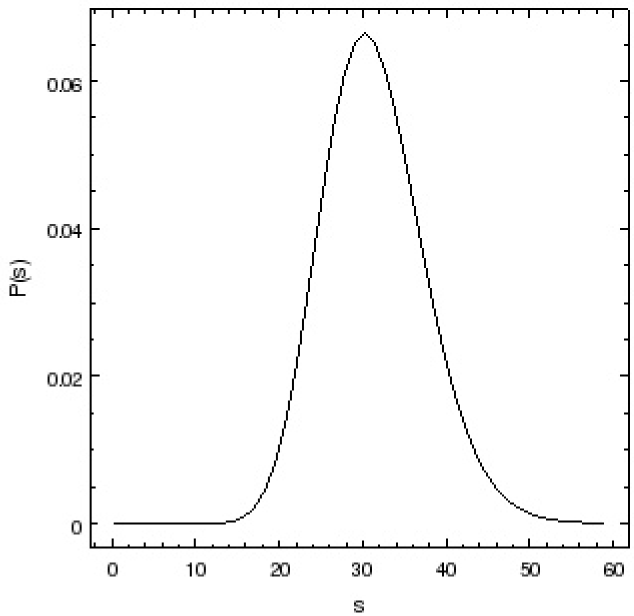
Grouping OBSIDs into Bayesian Blocks on the Basis of Flux

— Aperture Photometry algorithm uses counts $n_s,\ n_b$ in source and background regions, psf fractions $f_s,\ f_b$, average exposure map values $E_s,\ E_b$, and region areas $A_s,\ A_b$ to compute posterior probability distribution for source photon flux.

$$
\begin{aligned}
P(s,b\,|\,n_s,n_b,f_s,f_b,E_s,E_b,A_s,A_b) &= K\,P(s)\,P_{Poisson}(n_s\,|\,f_s,E_s,A_s)\,P(b)\,P_{Poisson}(n_b\,|\,f_b,E_b,A_b)\\
P(s\,|\,\ldots) &= \int_0^\infty P(s,b\,|\,\ldots)db\\
\int_0^\infty P(s\,|\,\ldots)ds &= 1\\
P(s) &= constant\,(flat\,,or\,non-informative,\,prior)
\end{aligned}
$$

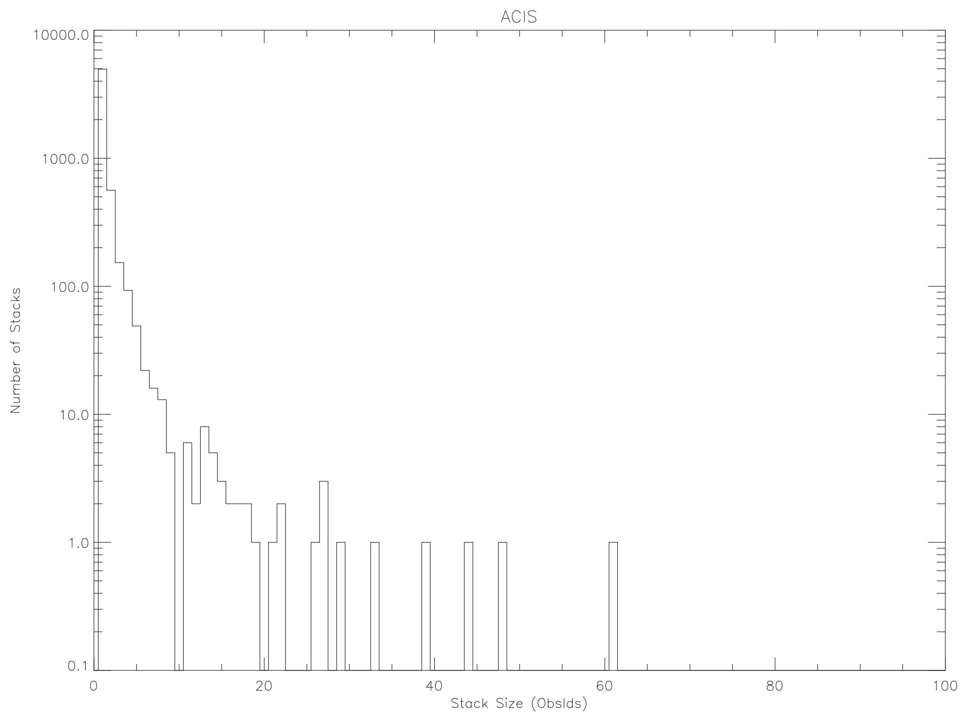
— For each obsid, do this for each energy band (5 for ACIS) and for source and 90% ECF regions (aperture types).



— Report mode and 68 % percentiles for each band, region, obsid in database

Grouping OBSIDs into Bayesian Blocks on the Basis of Flux

— Many sources observed more than once



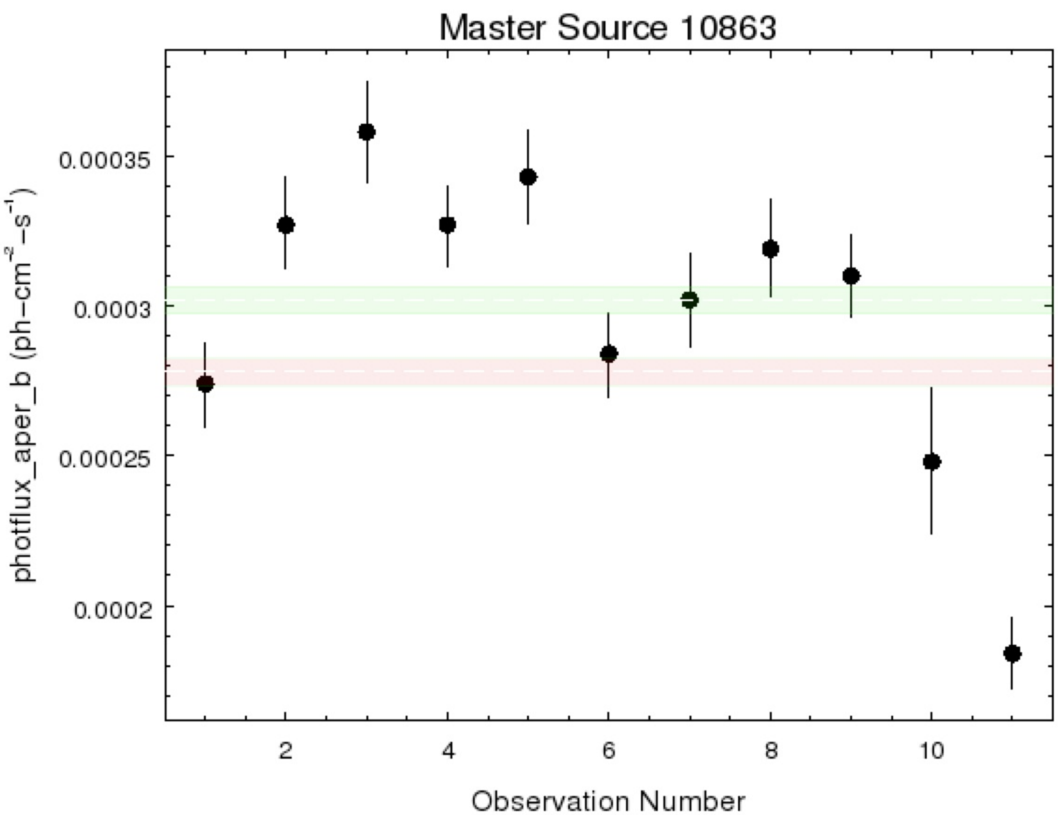
— Want to combine results from multiple obsids into a single “master source” flux, using the individual obsid results (or at least the same formalism)

Grouping OBSIDs into Bayesian Blocks on the Basis of Flux

— In Release 1, combined data from all the apertures

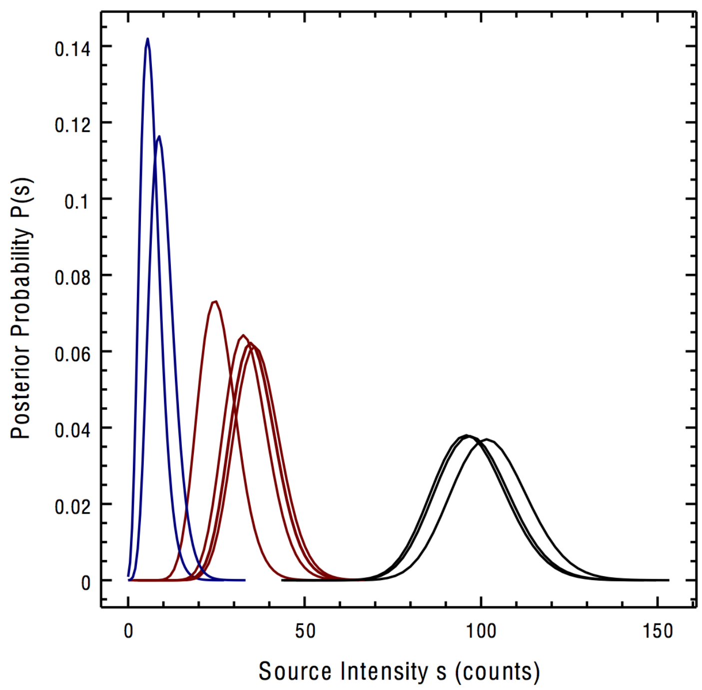
$$N_s = \sum n_{s_i} \; F_s = \sum f_{s_i} \; E_s = \sum E_{s_i} \; etc.$$

— Worked OK, but had some disadvantages
   — didn't incorporate upper limits
   — difficult to combine data on aperture areas
   — for variable sources, didn't match intuitive variance-weighted mean



Grouping OBSIDs into Bayesian Blocks on the Basis of Flux

— For Release 2, use different approach - use posterior $P(s|\ldots)$ from obsid 1 as prior $P(s)$ for obsid 2
— Advantages
  — Upper limits easily incorporated, as long as aperture photometry results are available for non-detections
  — No approximations about 'average' apertures needed
— Disadvantages
  — Where to start in combining obsids? Need to decide how to order results from individual obsids.
  — For variable sources, posterior pdf for one obsid may not be a good candidate for prior for another obsid.



Grouping OBSIDs into Bayesian Blocks on the Basis of Flux

— Order individual obsid results by time. Choose this rather than flux to ensure results represent an actual physical state of the source.
— Divide obsids into blocks, within which a constant source flux is consistent with photometry results from individual obsids.
— Use Bayesian Blocks algorithm of Scargle et al. 2013, “STUDIES IN ASTRONOMICAL TIME SERIES ANALYSIS. VI. BAYESIAN BLOCK REPRESENTATIONS”, (2013ApJ...764..167S).

$$\begin{aligned} P(\{B_i\}\,|\,O_j) &= P(N_{blocks}) \prod_{i=1}^{N_{Blocks}} F(B_i\,|O_j \in B_i) \\ P(N_{Blocks}) &\sim \gamma^{N_{Blocks}} \\ F(B_i\,|O_j \in B_i) &= \int_0^\infty ds \left[\prod_{j\,|\,O_j \in B_i} P(s_j\,|\,\ldots) \right] \end{aligned}$$

— Assumptions:
  — In any given block, obsids are sequential in time, although they may be separated by arbitrary gaps
  — $N_{Blocks} < N_{Obsids}$ $(0 < \gamma < 1)$
  — Data from different energy bands sum in computing F

— Select $\{B_i\}$ that maximizes $log[P(\{B_i\}\,|\,O_j)]$:

$$\begin{aligned} log[P(\{B_i\}\,|\,O_j)] &= N_{Blocks} log\gamma + \sum_{i=1}^{N_{Blocks}} log[F(B_i)] \\ &= \sum_{i=1}^{N_{Blocks}} [log[F(B_i)] - ncprior] \\ ncprior &= |log(\gamma)| \end{aligned}$$

— Parameter *ncprior* determined from simulations.
— Details in routines *get_blocks.py* and *get_Fitness.py.*

Grouping OBSIDs into Bayesian Blocks on the Basis of Flux

— Example: Simulate same source in 10 obsids, with times randomly sampled from uniform distribution between 0 and 1.0, and source counts randomly sampled from Poisson distribution with means given by the following profile:

t	Counts
$0 \leq t < 0.333$	10
$0.333 \leq t < 0.667$	100
$0.667 \leq t < 1.0$	30

```
Number of Cells:        10
Background Density:     0.025000
Source Position:        (57.548635,66.938363)

Cell    Start Time      Counts          PDF Mode        CL lo           CL hi           Actual CL
1       0.495572        100.00          95.808          85.591          106.737         0.686
2       0.466260        100.00          101.537         91.151          112.903         0.686
3       0.883630        30.00           24.534          19.465          30.467          0.687
4       0.323591        10.00           5.487           3.027           8.865           0.698
5       0.953232        30.00           34.743          28.564          41.464          0.685
6       0.699711        30.00           35.816          29.631          42.731          0.685
7       0.912493        30.00           32.604          26.765          39.255          0.685
8       0.881968        30.00           34.619          28.564          41.464          0.685
9       0.180941        10.00           8.679           5.621           12.709          0.697
10      0.412023        100.00          96.814          86.510          108.137         0.694
0.180941        10
0.323591        10
0.412023        100
0.46626        100
0.495572        100
0.699711        30
0.881968        30
0.88363        30
0.912493        30
0.953232        30
Evaluating ncprior
ncprior:        0       change_points:   0   1   2   3   4   5   6   7   8   9
ncprior:        1       change_points:   0   1   2   3   4   5   6   7   8   9
ncprior:        2       change_points:   0   2   5
ncprior:        3       change_points:   0   2   5
ncprior:        4       change_points:   0   2   5
ncprior:        5       change_points:   0   2   5
ncprior:        6       change_points:   0   2   5
ncprior:        7       change_points:   0   2   5
ncprior:        8       change_points:   0   2   5
ncprior:        9       change_points:   0   2   5
```

Grouping OBSIDs into Bayesian Blocks on the Basis of Flux

Draft Specification

— For each master source:
1. For each aperture type (source or ecf90):
   a) collect posterior probability distributions for all contributing obsids (from all contributing cohorts), together with aperture data (counts, psf fractions, expmaps, etc.)
   b) order pdfs by time of obsid
   c) compute blocks and report time of first obsid in each block
   d) Within each block:
      i. For each band
         A. re-order obsids by net source counts
         B. use pdf for lowest net count obsid as prior probability distribution for next lowest obsid, and re-compute pdf
         C. iterate until all obsids in block are used
   e) Select one block as representative and report in DB
      i. mode and percentiles of resultant pdfs from previous step as master source flux and confidence bounds (per band)
      ii. some (TBD) number or numbers to describe block (total exposure, duration, etc.) (per block)
      iii. obsids that contribute to representative block
   f) output to fits file pdfs from step 1 d (intensity array and pdf array, per band, per block)

Grouping OBSIDs into Bayesian Blocks on the Basis of Flux

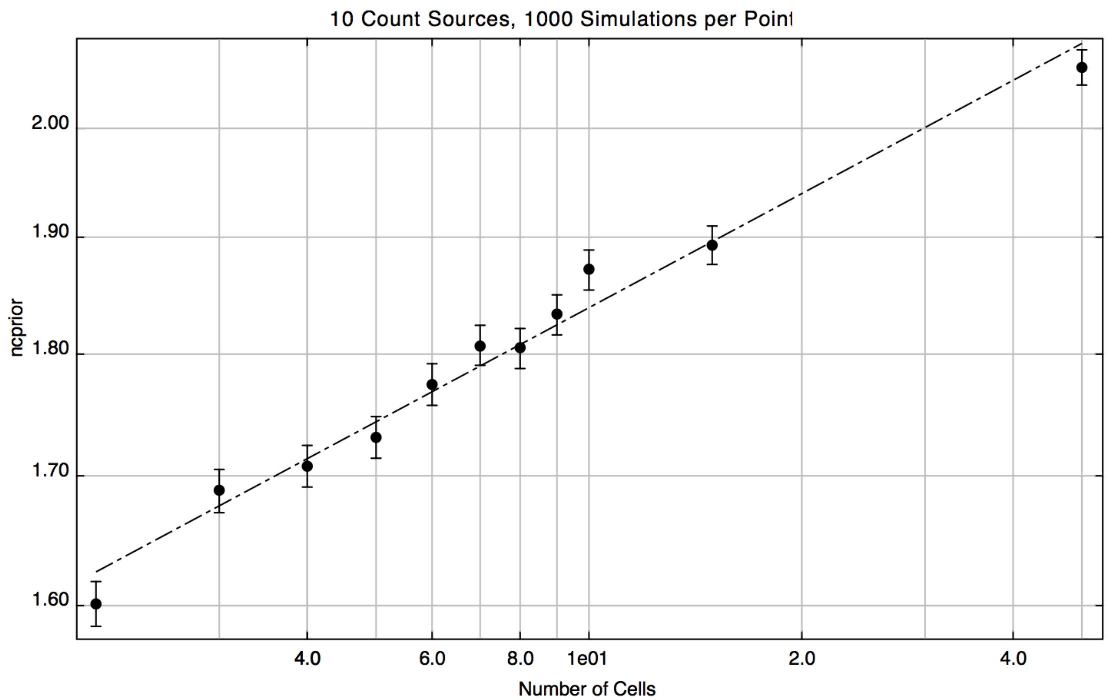
Impact on CSC Products

— Master source flux and 68% percentiles are already reported per band per aperture type - no additional burden
— At least 1 additional (double) column for block description for each aperture type
— Either 1 additional column containing variable-sized array of contributing obsids for each aperture type, or some other way of describing the relation, similar to master source - per obsid source association
— Inter-observation light curves are already provided per band; no additional burden on number of files (unless each aperture type is saved in separate file), but now each file will include intensity and pdf arrays per block; arrays are typically 50 doubles each; also should include start time and total exposure of each block.

Grouping OBSIDs into Bayesian Blocks on the Basis of Flux

Outstanding Issues

— Determine $ncprior$
  — Simulations running; preliminary results available for null case (no variability)



— need to explore how $ncprior$ depends on source counts, number of obsids, different variability profiles
— need to verify that multiple bands can be analyzed together (one block rules all the bands)
— need to define fall-back option in case step 1.d.i. in draft spec fails
— need to decide how to select "representative" block (longest exposure, longest duration, brightest, etc.) and what data are used to describe them in the DB
— need to define actual structure of output data files (assumed 1 file per band, 1 row per block with variable-length arrays)
— need to decide whether both aperture types need to be included

Grouping OBSIDs into Bayesian Blocks on the Basis of Flux

get_Fitness.py

```
def get_Fitness(pdfs,nsteps=1000):
    """
    Determine cumulative fitness functions for a range of pdfs, input as a list with

    pdfs[i][0] = start time of cell
    pdfs[i][1] = array of intensity values s at which pdf is evaluated
    pdfs[i][2] = array of pdf values for that cell

    The pdf is normalized such that sum(pdf)*(s[1]-s[0]) = 1

    """

    import numpy as np

    mins=[]
    maxes=[]
    npdfs = len(pdfs)
    F = np.zeros(npdfs)

    # First, find range of all the s arrays

    for i in range(0,len(pdfs)):
        mins.append(pdfs[i][1][0])
        maxes.append(pdfs[i][1][-1])

    # and use that to define new intensity grid s


    s0 = np.min(mins)
    s1 = np.max(maxes)
    ds = (s1-s0)/nsteps
    sint = np.arange(s0,s1,ds)

    # Build fint, the integrand of F, which is the product of the regridded pdfs. Start with fint set to 1
    # and work backwards, so that F includes the last pdf only the first time through the loop, then the
    # last two, etc. until it includes them all.

    fint = np.ones(len(sint))

    for i in range(0,npdfs):
        j = npdfs - i - 1
        pint=np.exp(interpolate(pdfs[j][1],np.log(pdfs[j][2]),sint))
        pint /= (sum(pint)*ds)
        fint *= pint
        F[j] = sum(fint) * ds

    return np.log10(F)
```

Grouping OBSIDs into Bayesian Blocks on the Basis of Flux

get_blocks.py

```
def get_blocks(pdf_list,ncprior):
    """
    Determine change-points for Bayesian Blocks
    Input:
    pdf_list: List of pdf data for each cell, [start time of cell, array of intensity bins, array of pdfs]
    ncprior:  Penalty factor. log10 of prior probability of having N blocks. Input ncprior is assumed >0
              and is subtracted from the fitness function for each block.
    Output:
    change_points: list of start times of cells that begin new blocks in optimum partition
    """

    import numpy as np

    # Make sure pdf list is time-sorted

    pdf_list.sort()

    ncells = len(pdf_list)

    # The optimal partition for the starting case of the first cell only has a best fitness function of
    # -ncprior, since the marginalized likelihood is 1 for  single normalized pdf. The location of the
    # first change-point is the beginning of the list, or index 0

    best = np.array(-ncprior)
    last = np.array(0)

    # Now need to construct A(r)

    for R in range(1,ncells):        # Skip the first cell since we already know the results for it
        F = get_Fitness(pdf_list[0:R+1])
        A = np.append(0,best) + F - ncprior
        best = np.append(best,A.max())
        last = np.append(last,A.argmax())

    # Once all ncells have been considered, reconstruct change-points from 'last' array:

    change_points = []
    cpindex = last[-1]

    while cpindex > 0 :
        change_points.insert(0,cpindex)
        cpindex = last[cpindex-1]

    # above gets everything except the first one

    change_points.insert(0,last[0])

    return change_points
```