DESI Commissioning Instrument Off-line Image Reduction/Analysis Pipeline

Aaron Meisner (NOAO)

Overview

The DESI Commissioning Instrument (CI) off-line image reduction/analysis package is implemented in Python. The code is being developed through a publicly accessible GitHub repository on my personal account. “ci_reduce” is the name I gave the Python package. The ci_reduce Python package is intended to be run at NERSC.

Dependencies always play an important role in the development and maintenance of software. I have taken the following approach to ci_reduce dependencies: any Python module available via the standard "DESI at NERSC" environment can be considered available for use by ci_reduce, but no dependencies beyond the DESI at NERSC environment are permitted. In particular, this means that the general-purpose Python utilities available from Dustin Lang’s astrometry.net package are off-limits.

The ci_reduce code has object oriented features and is organized into submodules, e.g. one can do “import ci_reduce.analysis.util as util”. The code design was in part modeled after aspects of legacysurvey pipelines including “legacypipe”, “obsbot” and “legacyzpts”. There is a main “driver” script called ci_proc.py that launches the production pipeline from the command line (with various optional flags available). This can be thought of as the ci_reduce counterpart to legacypipe’s “runbrick.py”. The ci_reduce/py/scripts directory houses various relevant pieces of Python analysis code that are not part of the production ci_proc pipeline to be run on every exposure. The ci_proc production pipeline uses auxiliary files such as master flats, master biases, etc. The directory containing these files is accessed through an environment variable called "CI_REDUCE_ETC".

Simulated CI Images

At present, there are no on-sky CI observations available, and only a very limited set of engineering data has been provided to me (the so-called "forDK.tar.gz" samples"). Thus, to test and validate the ci_reduce pipeline, it is necessary to created a set of simulated CI images. There are several ingredients that enable the creation of simulated CI images with the correct properties.

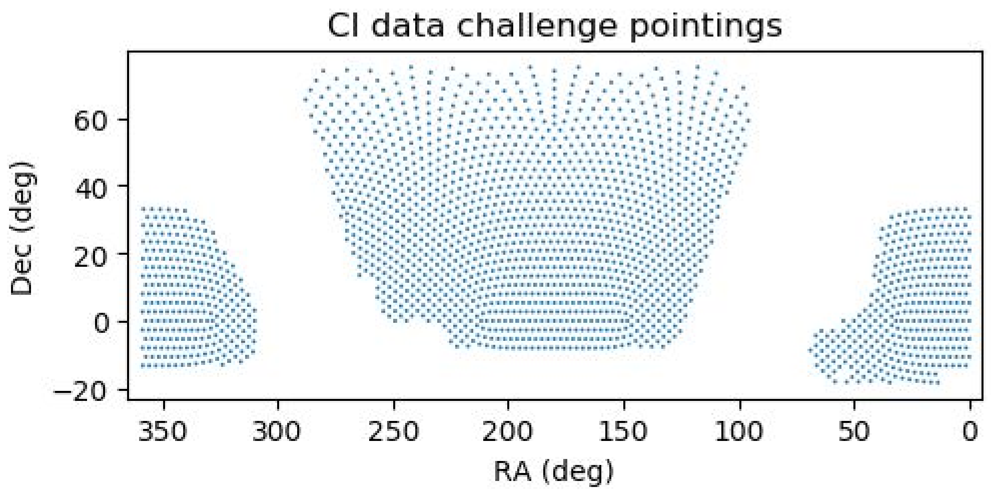
For one, we must know the zeropoint in order to determine how many electrons per second will be detected for a source of a given AB magnitude. I have investigated this in substantial detail, with my "ci_throughput" GitHub repository providing all of the code, auxiliary files, output files, and associated plots. I find a zeropoint whereby a source with total detected flux of 1 electron per second corresponds to an AB magnitude of 26.56. This zeropoint is used to scale the summed flux of fake sources injected into my simulated images. My simulations use the WCS solutions I created for Dustin Lang's online CI viewer tool.

The first step taken by my image simulation code is to calculate the number of measured electrons per pixel. To do this, I take into account many factors/effects: bias, flat field variation, dark current (including temperature dependence), read noise, sky background, injected compact sources, and Poisson noise. Once the image has been fully simulated in units of e-, division by the gain converts to ADU, and the resulting array is cast to an integer data type matching that of the forDK.tar.gz samples.

My CI image simulation software is roughly a thousand lines of throwaway IDL code, located in my "ci_sim" GitHub repo. This is intentional: it would be less informative/useful to test the ci_reduce pipeline on inputs generated with the very same Python package that performs the reductions, as doing so would be prone to circularity.

CI “Data Challenge”

Using the positions and brightnesses of real sources, I simulated a CI exposure for each of the 2,010 IN_DESI=1, PASS=0 desi-tiles.fits pointings. The plot below shows the locations of these pointings.



I then reduced all 2,010 simulated exposures using the ci_reduce package (actually I did this dozens of times while iterating on the ci_reduce codebase). No failures are encountered with the current version of the code. All of the forthcoming validation plots are drawn from CI data challenge outputs.

Pixel Level Calibrations

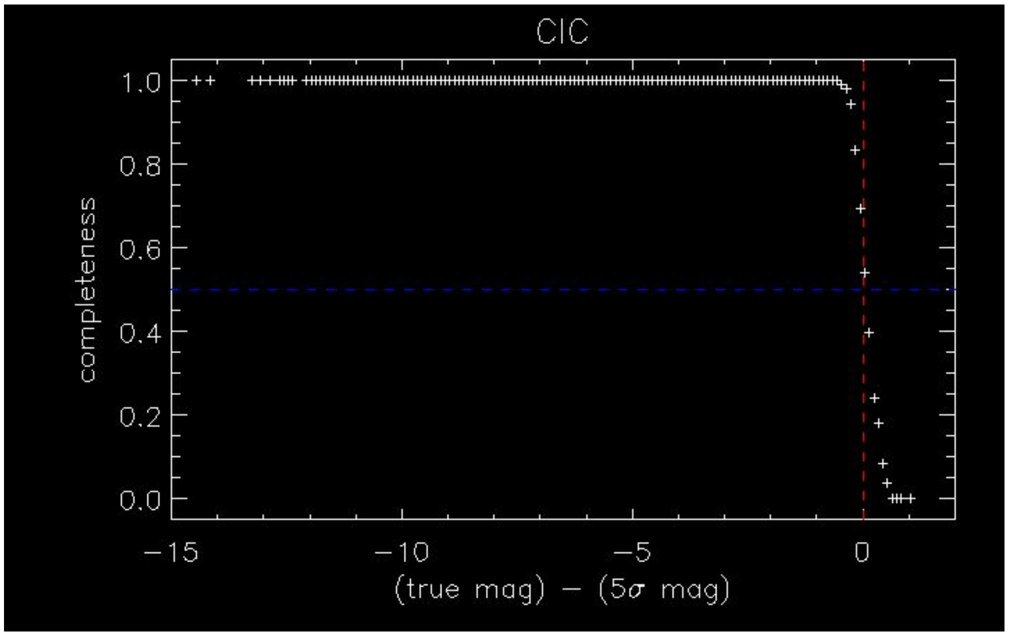
Please refer to the ci_reduce.exposure.calibrate_pixels() method if interested in details. The steps undertaken are: subtract bias, subtract dark current estimate, and divide by flat field. Additionally, an image-level inverse variance “weight map” is also created. All ci_reduce functionality validated in the remainder of this write up (source detection, flux measurement, centroid measurement, sky mag estimation) is based upon downstream analysis of the calibrated pixels delivered by ci_reduce. Therefore, those same analyses also validate the pixel-level calibration.

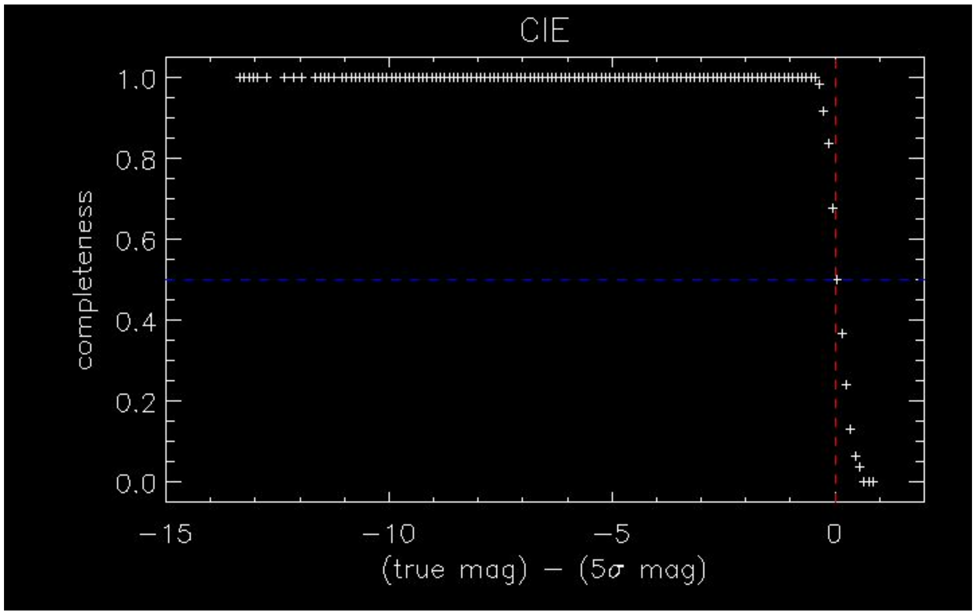
Source Detection

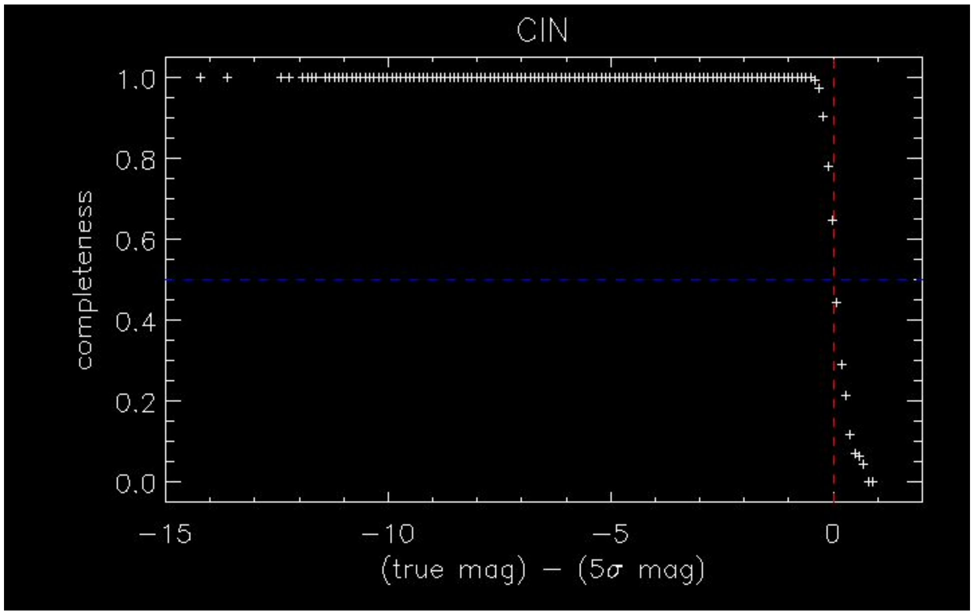
Source detection is performed by calculating a detection significance map, **not** by using a SExtractor-like approach that requires a certain number of contiguous pixels above a certain threshold.

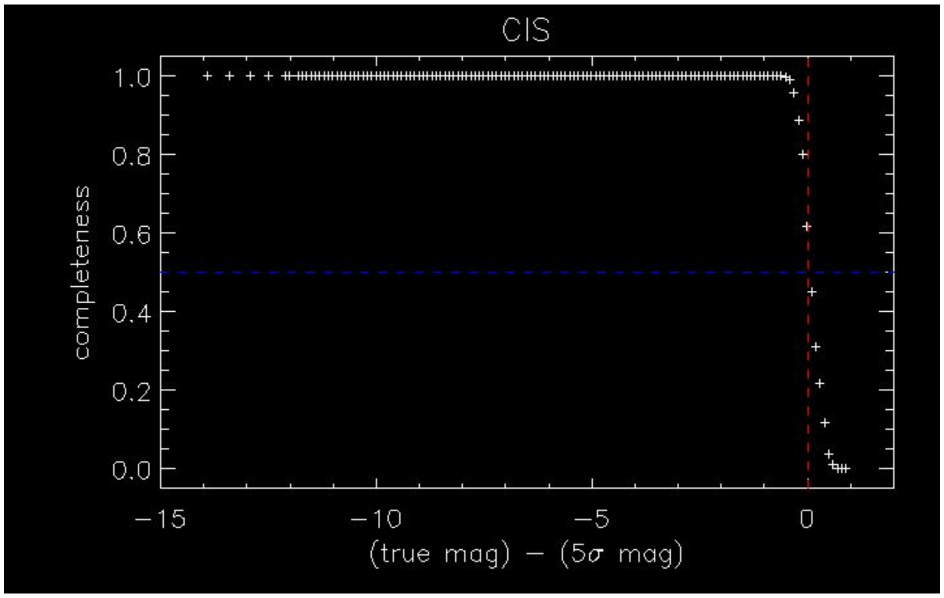
Detailed Validation Plots/Descriptions

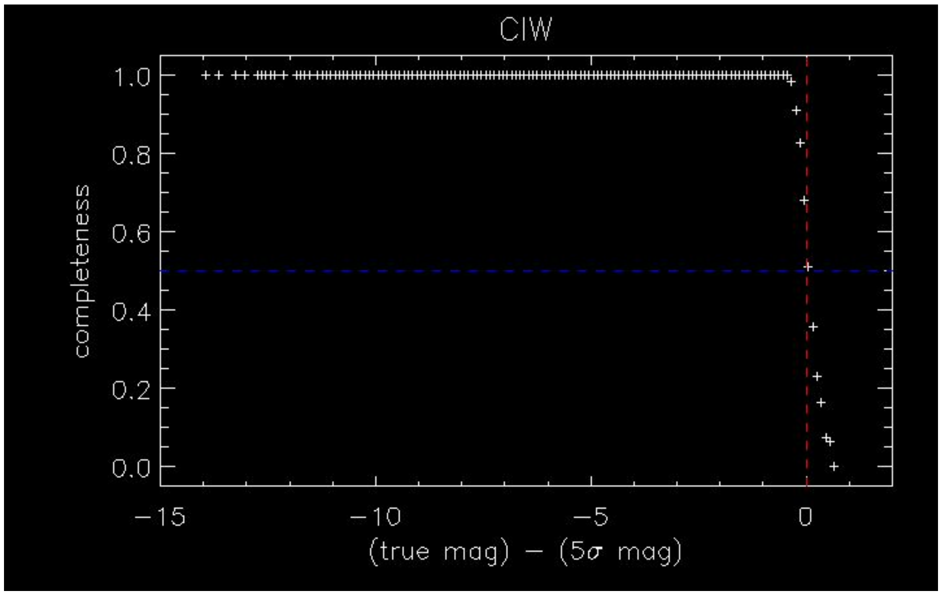
CI data challenge completeness plots:




CIE
completeness
(true mag) − (5σ mag)
1.0
0.8
0.6
0.4
0.2
0.0
−15
−10
−5
0


CIN
completeness
(true mag) − (5σ mag)
1.0
0.8
0.6
0.4
0.2
0.0
−15
−10
−5
0

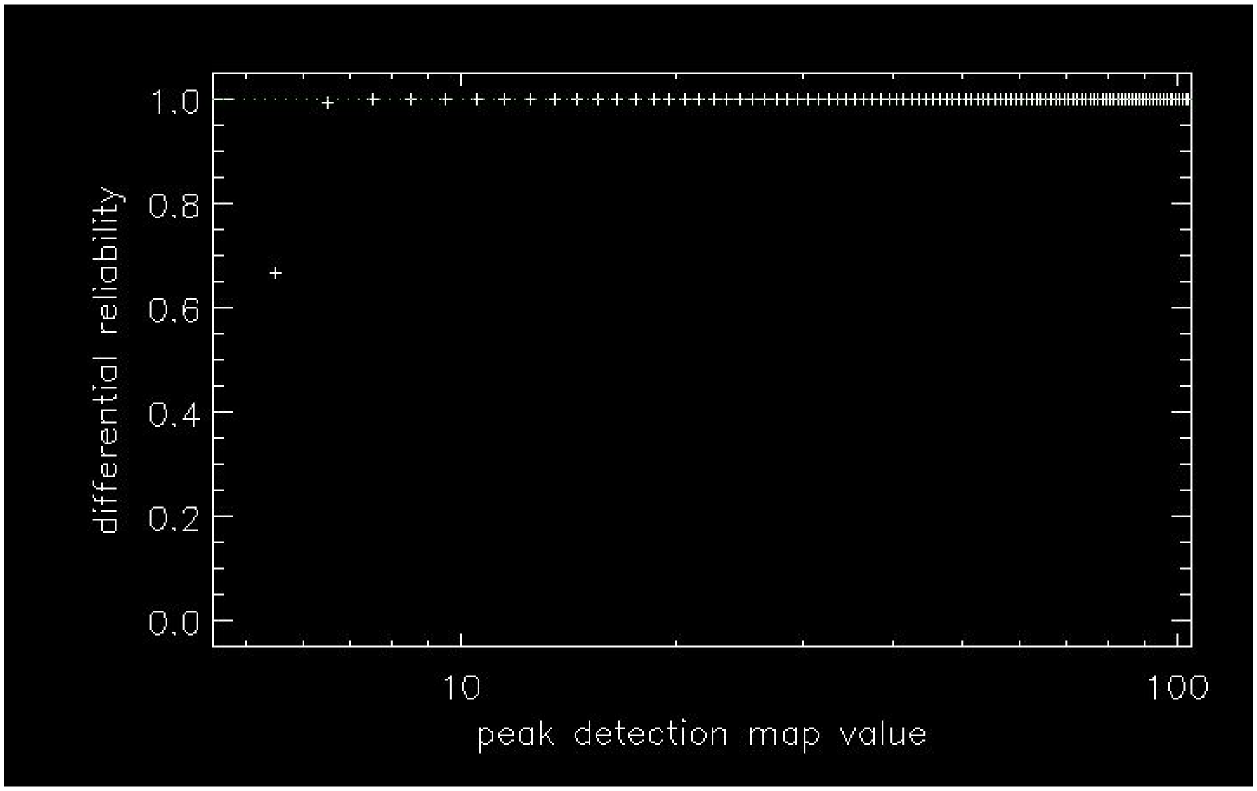




description of CI data challenge completeness plots:

I restricted to injected point sources at least 20 pixels from any image edge and lacking a neighboring injected source within 30 asec. These plots show the results of performing source detection with a "detection map" significance threshold of 5, i.e. 5 sigma source detection. The plots incorporate results from all ~2,000 simulated exposures worth of data challenge reductions. The blue dashed line is 50% completeness, meant only to guide the eye. The red dashed line is the predicted 5 sigma point source sensitivity based on calculating the n_eff value of the injected source profiles, the zeropoint (mag AB corresponding to total flux of 1 ADU), and the per-pixel background noise in the reduced images. The roll-off of the measured completeness coincides well with the predicted 5 sigma detection limit. All completeness plots show differential completeness.

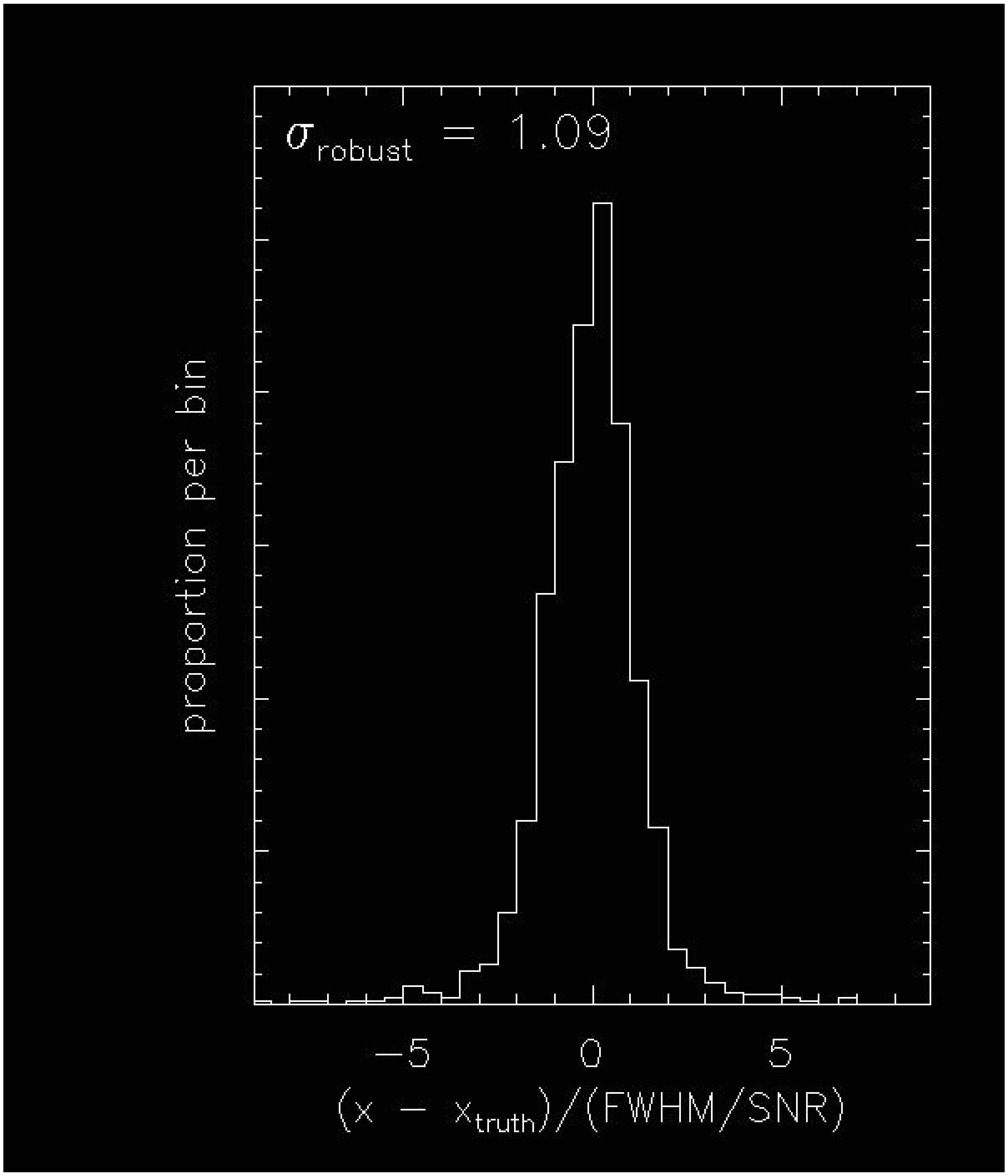
CI data challenge reliability plot:

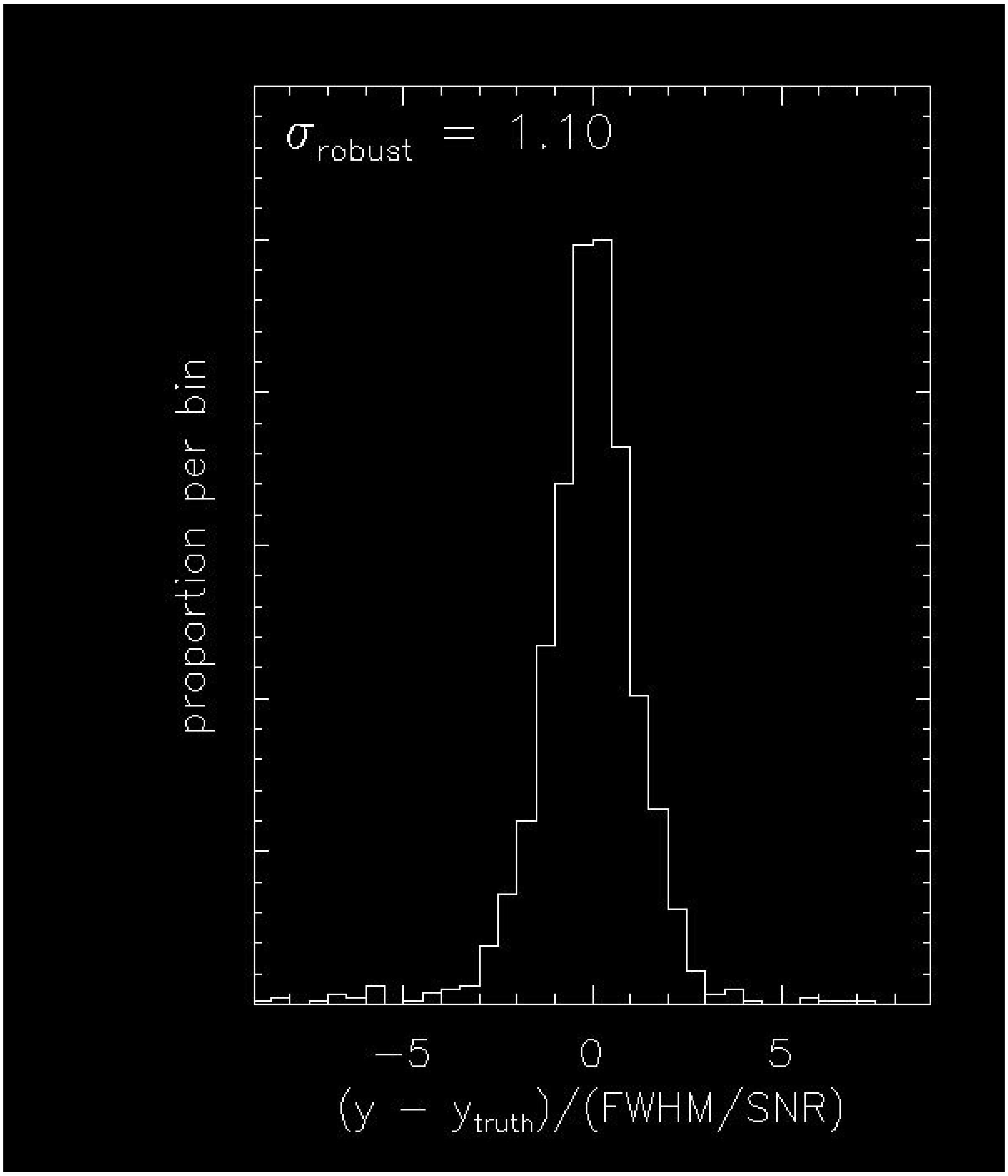


description of CI data challenge reliability plot:

I restricted to detected sources at least 20 pixels from any image edge, and also removed detections having more than 1 injected source within 30" (proxy for crowding/blending). These plots show the results of performing source detection with a "detection map" significance threshold of 5, i.e. 5 sigma source detection. The plots incorporate results from all ~2,000 simulated exposures worth of data challenge reductions. The green dotted line is 100% reliability. The sharp drop-off in the single bin with 5 < SNR < 6 is a bit surprising, but overall this plot demonstrates essentially perfect reliability for high S/N sources that aren't blended/confused or near image boundaries. Note the logarithmic scale on the x axis. Data points are independent bins of width 1 in terms of detection map peak value.

CI data challenge astrometry plots:


σ_{robust} = 1.09
proportion per bin
−5
0
5
$(x - x_{truth})/(\mathrm{FWHM}/\mathrm{SNR})$

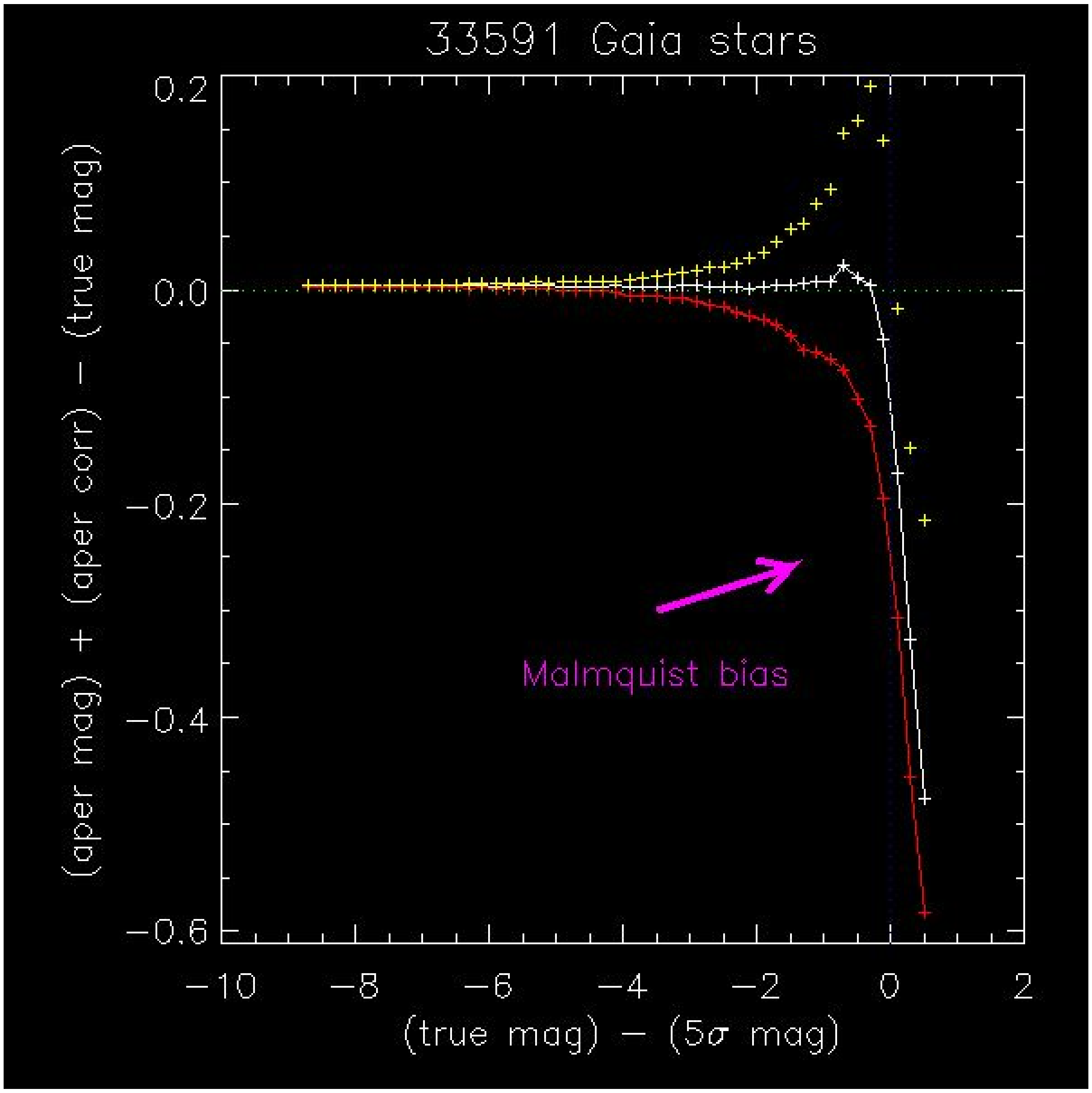


description of CI data challenge astrometry plots:

The CI data challenge outputs include extracted source catalogs for ~10,000 simulated CI images (~2,000 exposures, with 5 per-camera images for each exposure). I randomly chose a subset of 200 of the corresponding ci_reduce catalogs and plotted pull distributions of the source centroids (in each coordinate) relative to truth, which is simply taken from the catalogs of injected sources. These pull distributions normalize the pixel coordinate differences to the quantity (FWHM/SNR), with FWHM expressed in pixels and SNR coming from the detmap_peak column in my ci_reduce output catalogs. Under various assumptions, the optimal centroid uncertainty is FWHM/(2*SNR). These assumptions include an optimal PSF-based modeling of the sources, which is not currently being done by the ci_reduce pipeline. So we do not expect to saturate the FWHM/(2*SNR) bound based on the current ci_reduce implementation. A relevant point of comparison is the widely used IDL routine "djs_photcen", which returns reliable flux-weighted centroid measurements despite not performing an optimal PSF-fitting analysis. Running djs_photcen on the ci_reduce "_reduced" image outputs drawn from my CI data challenge, I find that the robust standard deviation of the per-coordinate pull distributions is very nearly FWHM/SNR. The ci_reduce centroid residuals relative to truth also display pull distributions with similar robust standard deviations of very nearly FWHM/SNR. Therefore I consider the ci_reduce centroid measurements to be acceptably valid.

The above pull distributions do not include all sources for all 200 catalogs analyzed. I made various cuts in attempt to restrict to isolated, unsaturated point sources not nearby any image boundary. These cuts include: a minimum distance from any image edge of at least 40 pixels, exactly one injected source catalog match within 30 asec, a nearest neighbor interpolated ci_reduce "_bitmask" value of zero (no bitmask data quality flags set at the location of the centroid), SNR < 1000, and no neighboring extracted sources in the ci_reduce "_catalog" output within 30 asec. The SNR < 1000 cut is implemented to avoid large pull values in cases of huge SNR, where the FWHM/SNR becomes a tiny number that may be comparable to the centroid estimation convergence criterion. These cuts could potentially have been further tuned/refined but I did not bother to do so.

CI data challenge photometry plots:

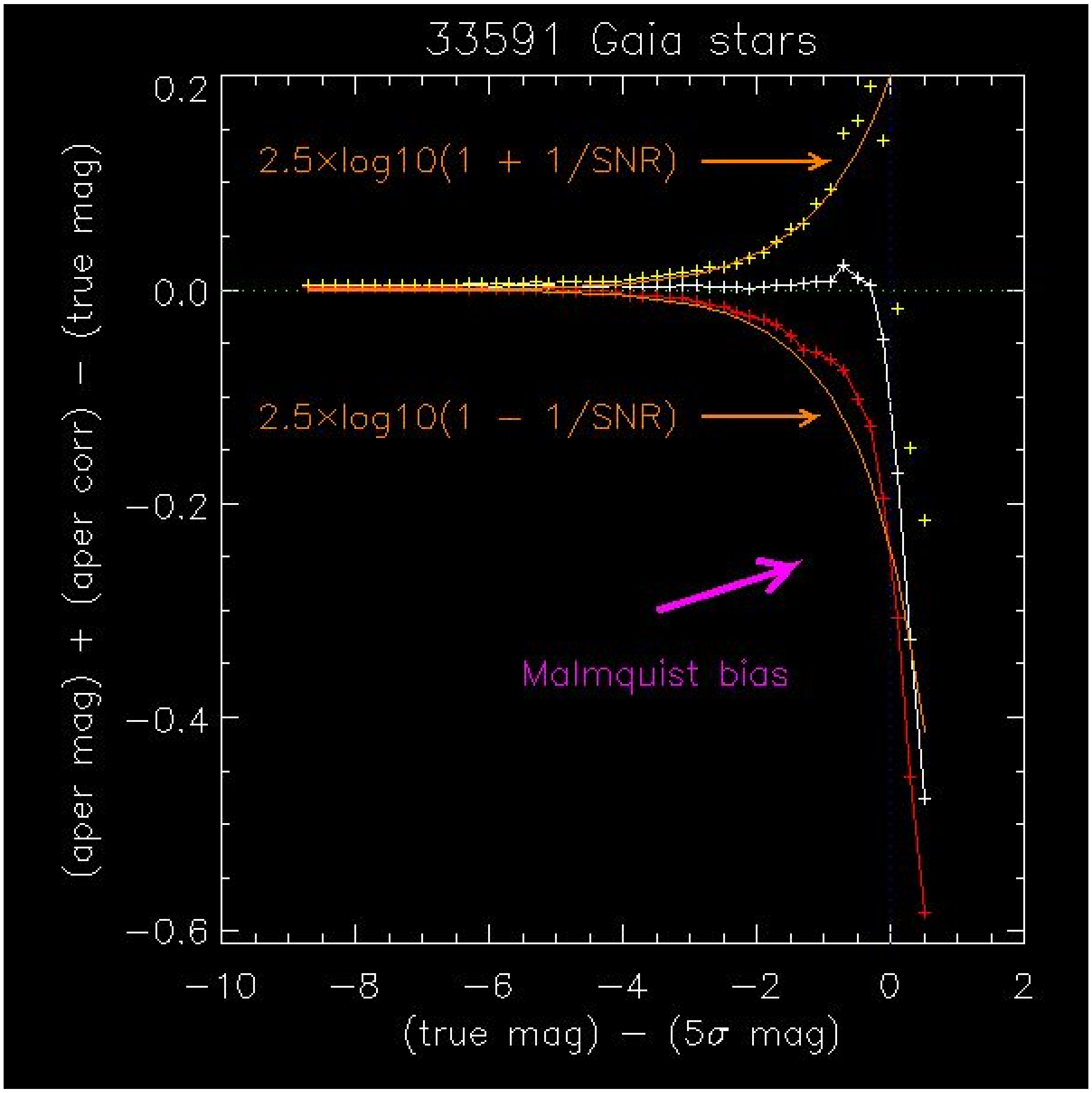


The above plot validates the ci_reduce aperture photometry. For this plot, I chose to use the aperture with diameter most closely matching the injected source FWHM. I then corrected the measured aperture fluxes to be total fluxes. Note that there is no uncertainty or systematic error associated with this correction, since I know the aperture size used and I know the profile of each source that I injected. The vertical axis is the difference between the aperture-corrected magnitudes measured by ci_reduce and the true magnitudes. The horizontal axis is the true source magnitude, shifted so that the 5 sigma limit is always at x = 0. This plot combines measurements from all CI cameras.

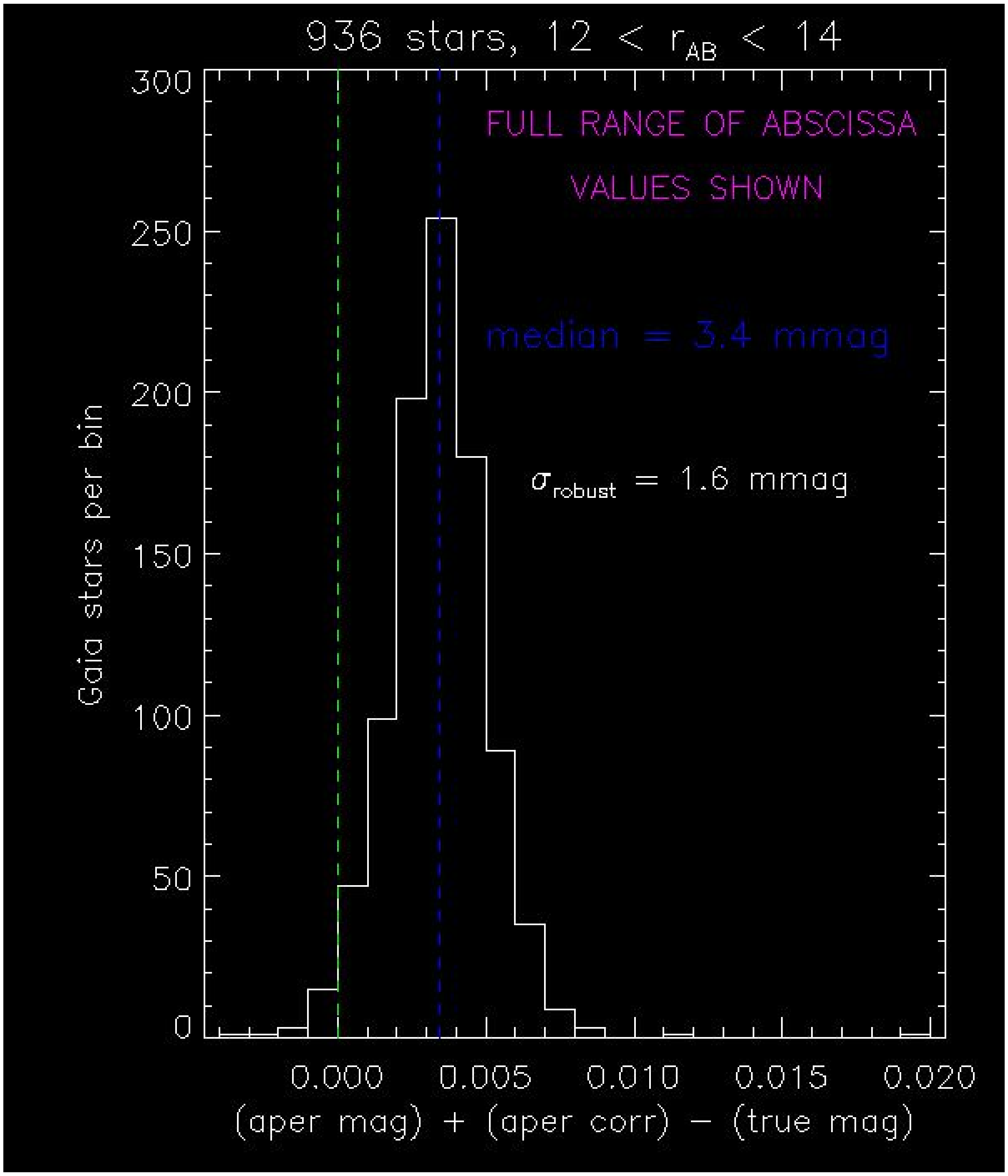
The white plus marks and connecting white lines show the median differences between aperture-corrected measured magnitudes and true magnitudes, in bins 0.2 magnitudes wide. The red plus marks and connecting red lines are the corresponding 25th percentile values. The yellow plus marks and connecting yellow lines are the corresponding 75th percentile values. The horizontal dotted green line indicates perfect agreement between aperture-corrected measured magnitudes and truth. The vertical dotted blue line denotes the SNR = 5 magnitude.

In total 33,591 injected stars drawn from a large number of exposures worth of ci_reduce outputs contribute to this plot. This comparison source sample is a subset of the full list of ci_reduce detections. Notably, I restricted to isolated stars at least 50 pixels from any image edge (using the ci_reduce catalog column min_edge_dist_pix) and required that the ci_reduce catalog column dq_flags have a value equal to zero, i.e. no data quality flags set. This latter cut removes saturated sources.

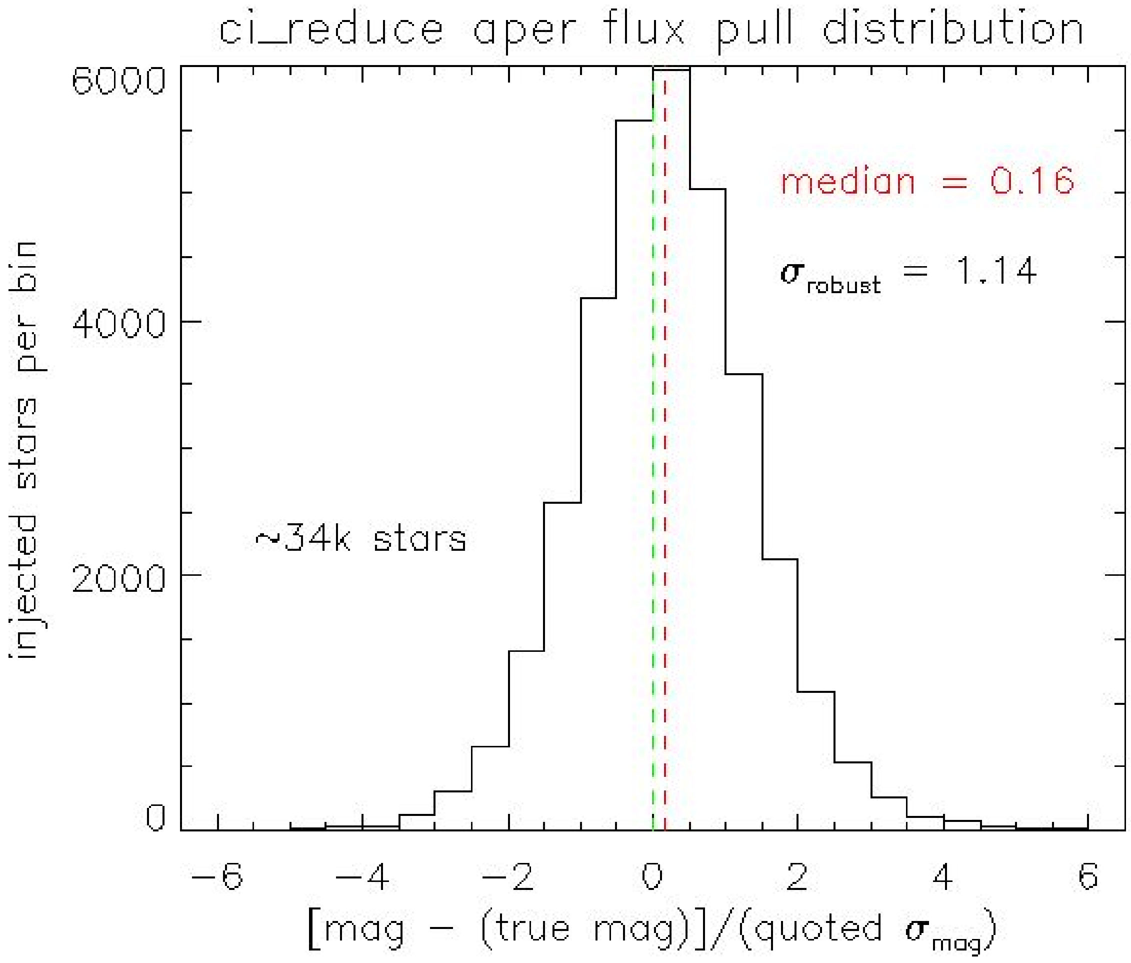
As indicated by the magenta arrow and associated annotation, the effect of Malmquist bias can be seen to set in near the 5 sigma detection limit, as expected. Sources near the detection limit are preferentially detected when they coincide with positive valued random noise, leading the measured magnitudes to be preferentially bright, i.e. lower in value than truth, which is why the red, white and yellow curves dip negative at the faint end.



This is a yet more heavily annotated version of the prior plot. Now I have added in the orange lines, which show the +/- 1 sigma uncertainty envelopes based on the SNR as a function of abscissa value. The very close matching of these orange envelopes with the measured 25th and 75th percentile curves is a coincidence -- with optimally measured fluxes, the 16th and 84th percentile curves should coincide with the orange lines. Therefore we can see that, as expected, the ci_reduce aperture fluxes are slightly suboptimal -- the 16th percentile curve for this set of data challenge outputs would presumably fall below the lower orange envelope, and the 84th percentile curve for this set of data challenge outputs would presumably fall above the upper orange envelope.

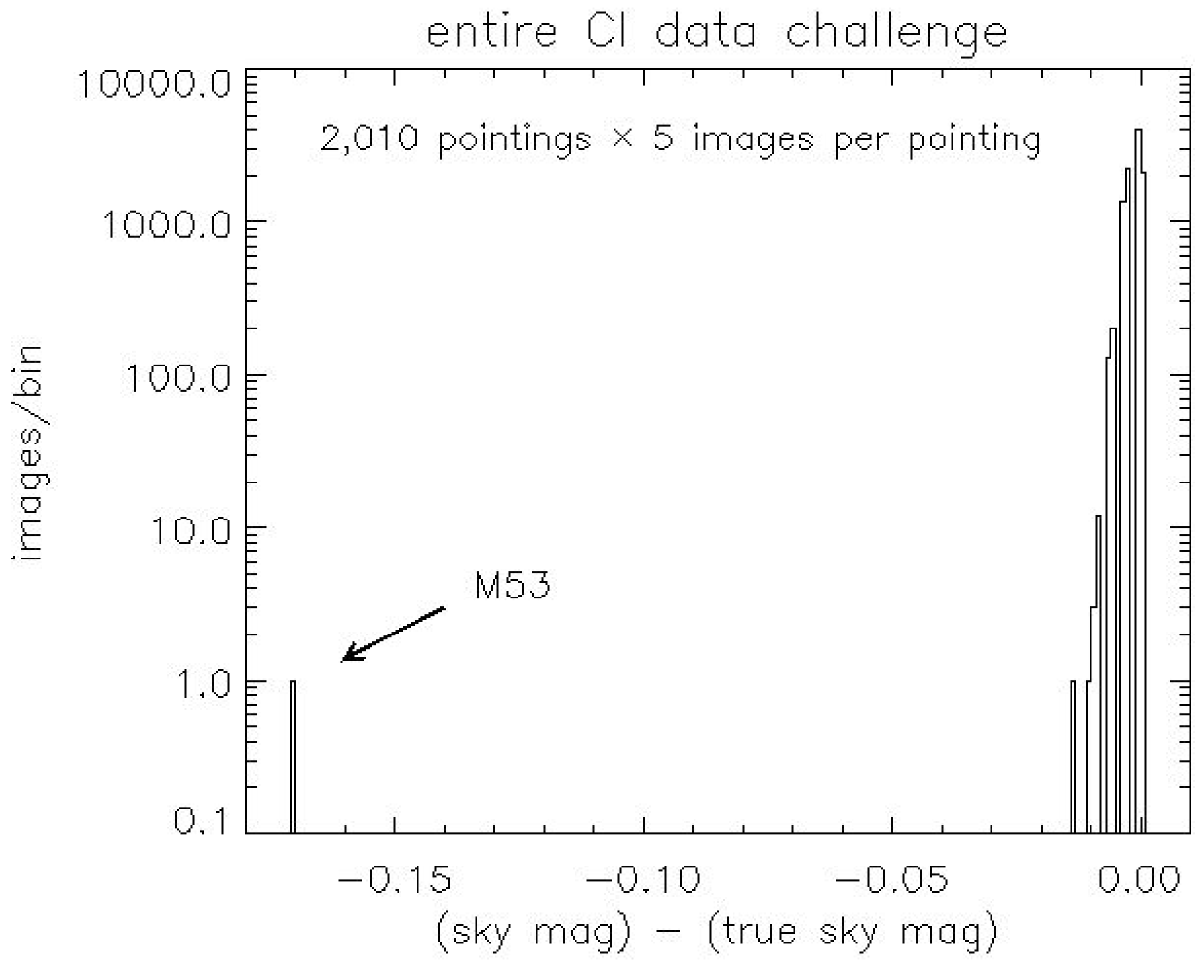


The prior two magnitude comparison plots are crowded with annotations, making it difficult to perceive the bright end accuracy and scatter. The above plot shows a bright-end histogram of the aperture-corrected measured magnitude minus true magnitude difference. The robust standard deviation indicates a bright end scatter of just 1.6 mmag. The full range of abscissa values is shown -- there are no outliers beyond the bounds of the plot. There is a very clear bright end offset of 3.4 mmag, in the sense that the aperture-corrected measured magnitudes are very slightly fainter than truth. One hypothesis that may be able to explain this phenomenon is that small astrometric scatter in the measured centroids adopted for aperture photometry causes a small net loss of flux from within the aperture. This hypothesis could be tested by subsetting the 936 stars in the above histogram based on how close/far their ci_reduce measured centroids are from truth, but I have not performed this test. Another way to test this hypothesis would be to feed the ci_reduce aperture photometry module a list of the exact centroid positions of the injected sources (ci_reduce does not currently offer such forced aperture photometry functionality).



The above plot shows a pull distribution based on the aperture fluxes and their uncertainties reported by ci_reduce. The same sample of ~34k isolated stars as discussed in previous figure descriptions was also used for this plot. The robust standard deviation of the aperture flux pulls is very close to unity, indicating that the aperture flux uncertainties quoted by ci_reduce are reasonable. The small positive bias may again be related to the small bright end bias discussed previously, and has the correct sign to arise from net loss of flux within the aperture due to imperfect centroid determination.

CI data challenge sky mag:



Although sky magnitude estimation is not formally on the checklist of image analysis items required for this review, I have implemented it anyway. The above plots shows a histogram of the difference between the simulated "true" sky level in AB magnitudes per square arcsecond and the sky level in AB magnitudes per square arcsecond computed by the ci_reduce pipeline. The agreement is excellent overall. The sky estimation is contaminated slightly by the presence of compact sources, which is why the deviations from zero are essentially all negative (sky flux per pixel estimate is slightly elevated by flux from compact sources). The most strongly discrepant point alone at left is the globular cluster M53:

http://legacysurvey.org/viewer?ra=198.2310&dec=18.1716&zoom=13&layer=decals-dr7

The ci_reduce pipeline offers a command line option --careful_sky which attempts to use image segmentation to remove the influence of compact sources on overall sky background level estimation. Running the M53 image with this option makes the sky magnitude agreement 2x better (within ~0.1 mag of truth), but still clearly discrepant relative to the distribution of the other ~10,000 images analyzed. In any case, the affected pointing has 4 other CI cameras worth of correct sky mag estimates.