simulation tools

Philipe Mota

CBPF

June 5, 2020

goal

- it is essencial to have a good control of the noise and dark current of the images since they affect directly detection threshold
- a good understanding of the behavior of the dark current is paramount for an efficient event extraction focusing in reducing the fake events
- hardware approach using skipper CCDs is underway which promises to decrease significantly the readout noise allowing the proper identification of the dark current
- I have advanced in a parallel direction: developping tools to extract the maximum information from the existing images both 1x1 and 1x5

goal

- analytically, I have derived the following relations

$$m_1[\mathrm{DC} + \mathrm{Noise}] = g\lambda + \mu[\mathrm{Noise}]$$

$$m_2[\mathrm{DC} + \mathrm{Noise}] = g^2\lambda + \sigma[\mathrm{Noise}]^2$$

$$m_3[\mathrm{DC} + \mathrm{Noise}] = g^3\lambda$$

  where m_i are the i-th moments of the Poisson-Norm distribution corrected by its first moment, μ and σ are the parameters of the Norm noise, λ is the Poisson rate and g is the convolution shift which physically controls how many ADUs the distribution is shifted by the presence of one electron (dark current), g is the charge gain
- these relations are redundant and can be used to attest the quality of the estimations

goal

- in practice, we have to relate these quantities with sample estimations of the data
- The straightforward associations are proposed in a first glance

$$\mu[\text{Noise}] = \text{mean}[\text{OS}]$$

$$\sigma[\text{Noise}]^2 = \text{var}[\text{OS}]$$

$$m_1[\text{DC} + \text{Noise}] = \text{mean}[\text{AC}]$$

$$m_2[\text{DC} + \text{Noise}] = \text{var}[\text{AC}]$$

$$m_3[\text{DC} + \text{Noise}] = \text{mean}\big(X - \text{mean}[\text{AC}]\big)^3$$

- however, the presence of modulations and other features of the image make weaken the quality of these estimations
- furthermore, the presence of the data requires a careful analysis since all these quantites will be shifted by the data

goal

I focus in 3 parallel ways to tackle the problem

- correcting the image
- using robust estimators
- energy-independent separation of data and dark current

tool for image analysis

I have developed an user-friendly – ask Carla ;-) – tool for this analysis in fact, this presentation is can be generated by the tool sample call

```
python Image.py analyse ImagePresentation/median
    "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --plot-sections --
    plot-spectrum
```

other functionalities

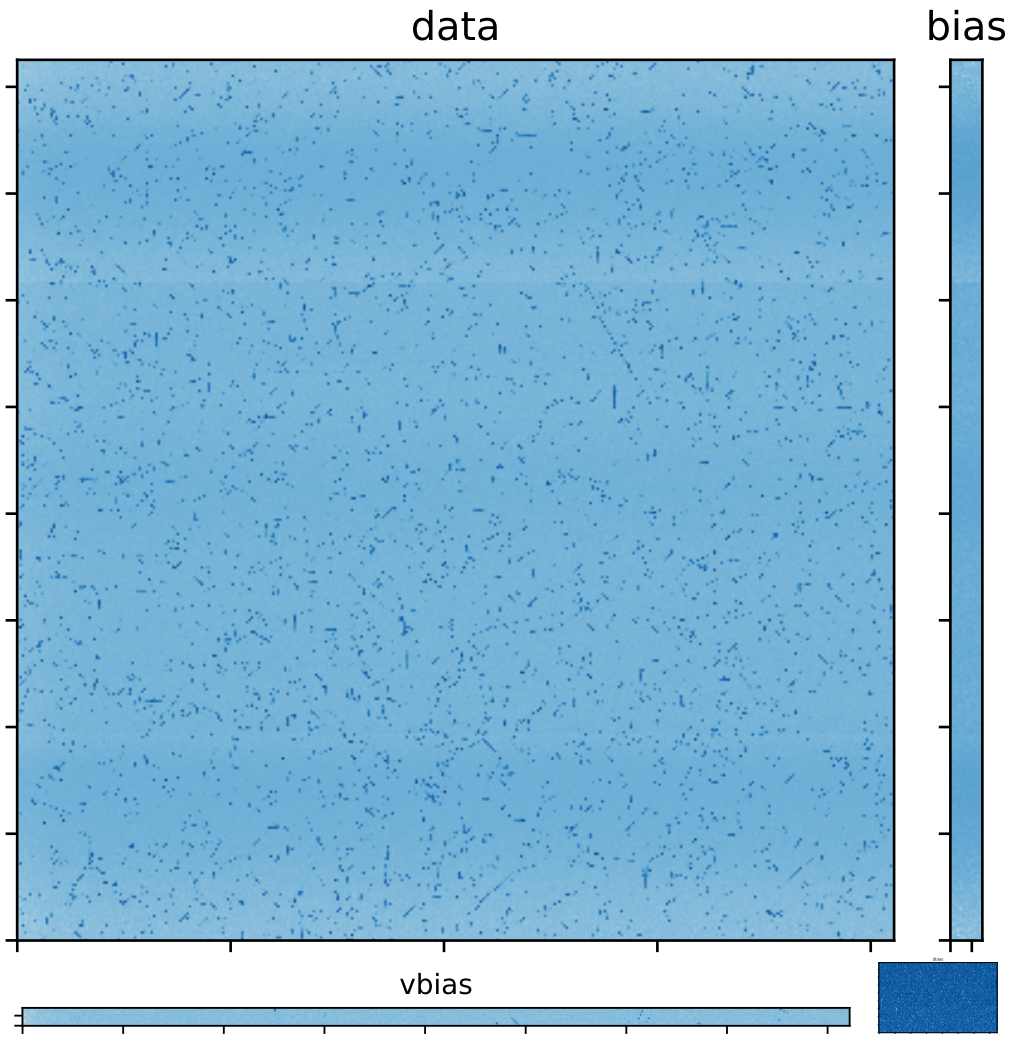
- read header
- simulate and get params
- extract hits (next week: comparison with offical extraction)

run locally

```
git init
git pull https://github.com/PhMota/CONNIEtools
python ImagePresentation.py
```

image imperfections

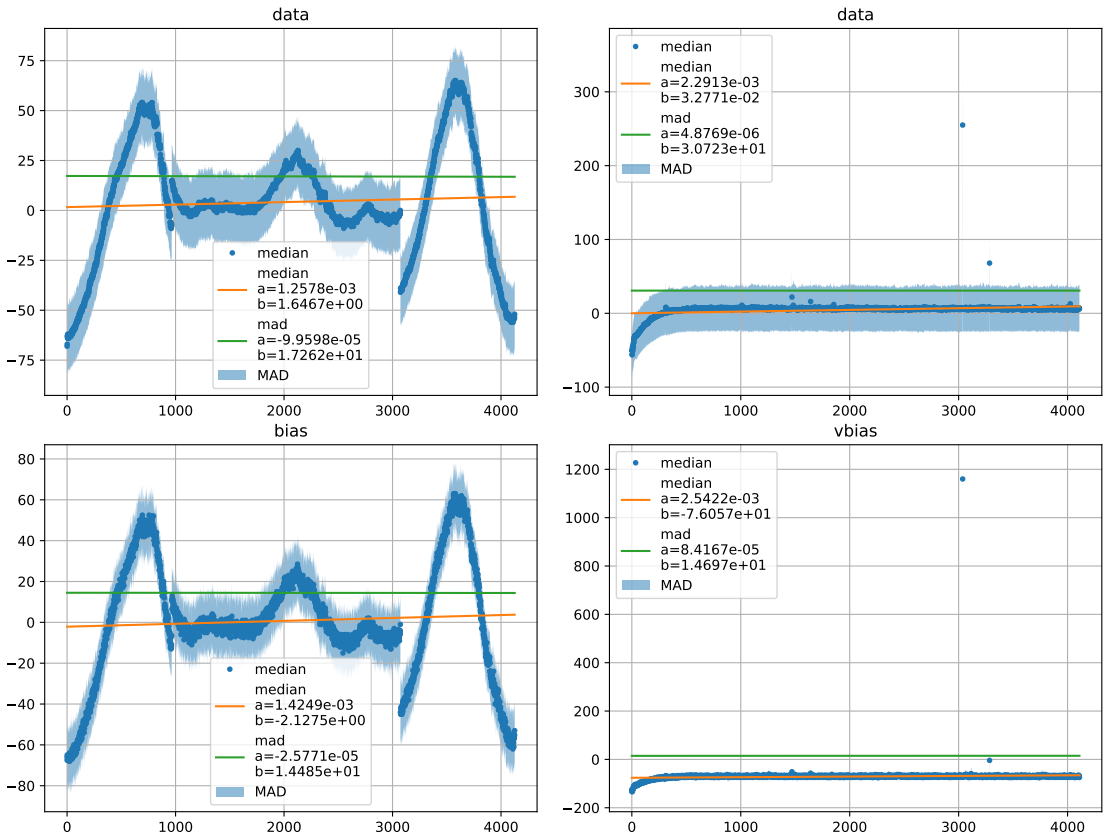
```
1   python Image.py analyse ImagePresentation/median
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --plot-sections --
        plot-spectrum
```



1x1 raw sections
vertical modulation is clearly visible

projections of the raw image

```
1   python Image.py analyse ImagePresentation/median
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --plot-sections --
        plot-spectrum
```



vertical modulation, horizontal modulation, hot columns

projections of the raw image

the MonitorViewer tool attempts to circumvent these imperfections, by estimating the quantities independently for each line and taking the mean over these results

$$\sigma = \mathrm{mean}\big(\mathrm{MAD}[\mathrm{OS}]_i\big)$$

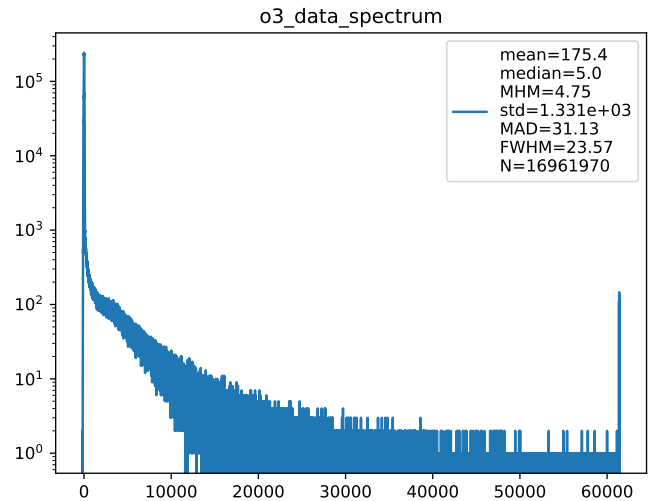
$$g\lambda = \mathrm{mean}\big(\mathrm{median}[\mathrm{AC}]_i - \mathrm{median}[\mathrm{OS}]_i\big)$$

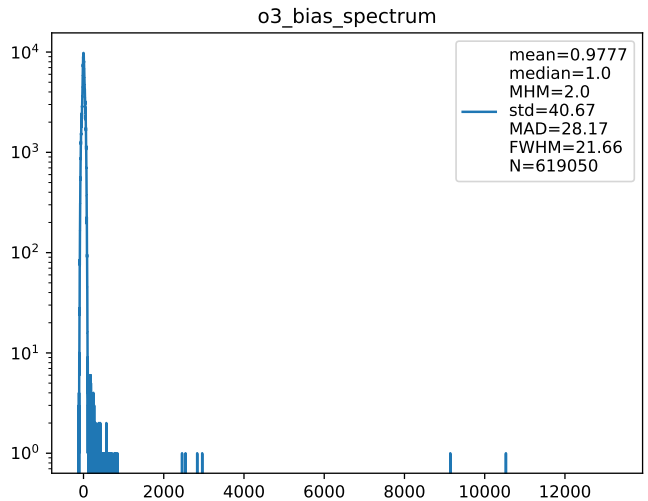
$$g^2\lambda = \mathrm{mean}\big(\mathrm{MAD}[\mathrm{AC}]_i - \mathrm{MAD}[\mathrm{OS}]_i\big)$$

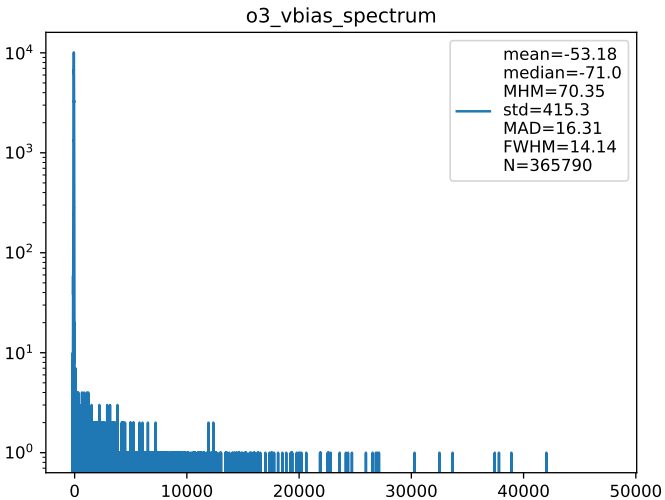
these quantities give consistent estimations for simulations generated with no(!) data

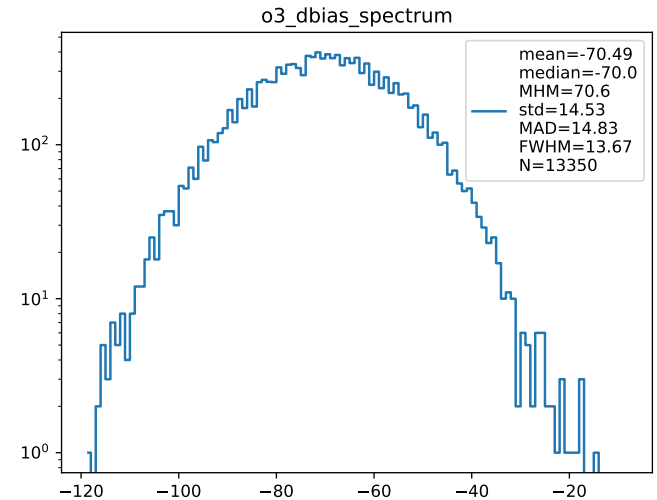
spectra of the raw image

```
1   python Image.py analyse ImagePresentation/median
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --plot-sections --
        plot-spectrum
```





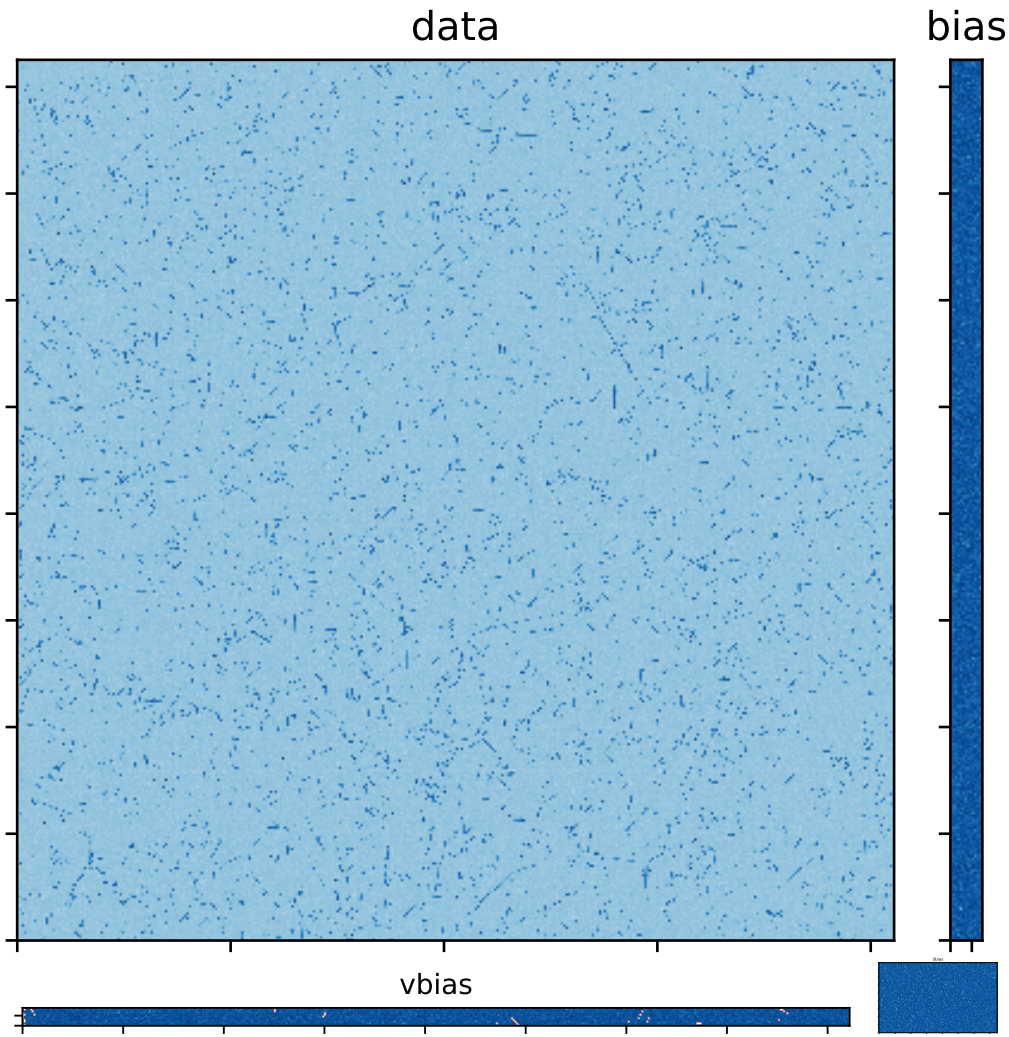




however, real distributions are crowded with outliers which heavily impair the capability of accurately estimating the parameters of the distribution

sections with line and col corrections

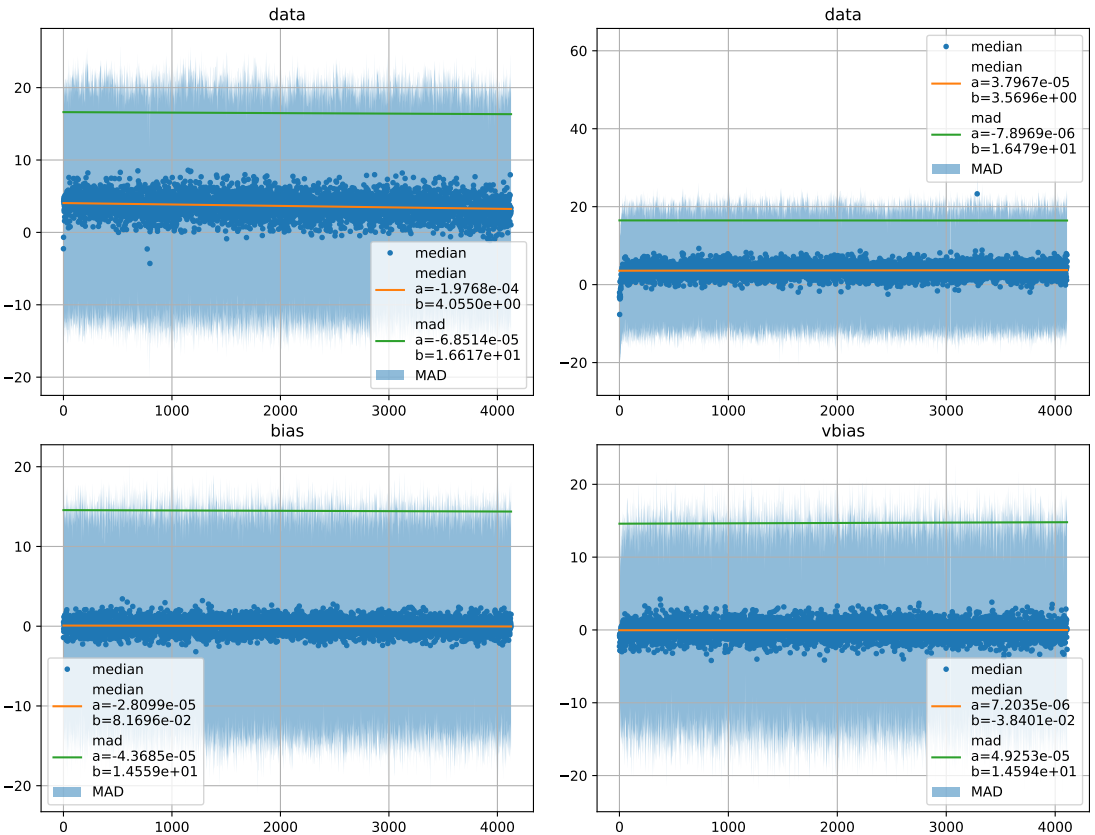
```
1   python Image.py analyse ImagePresentation/mean
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        plot-sections --plot-spectrum
```



perform overscan subtraction by estimating the mean of the distribution line by line after removing the outliers then remove the vertical overscan modulation

projections with line and col corrections

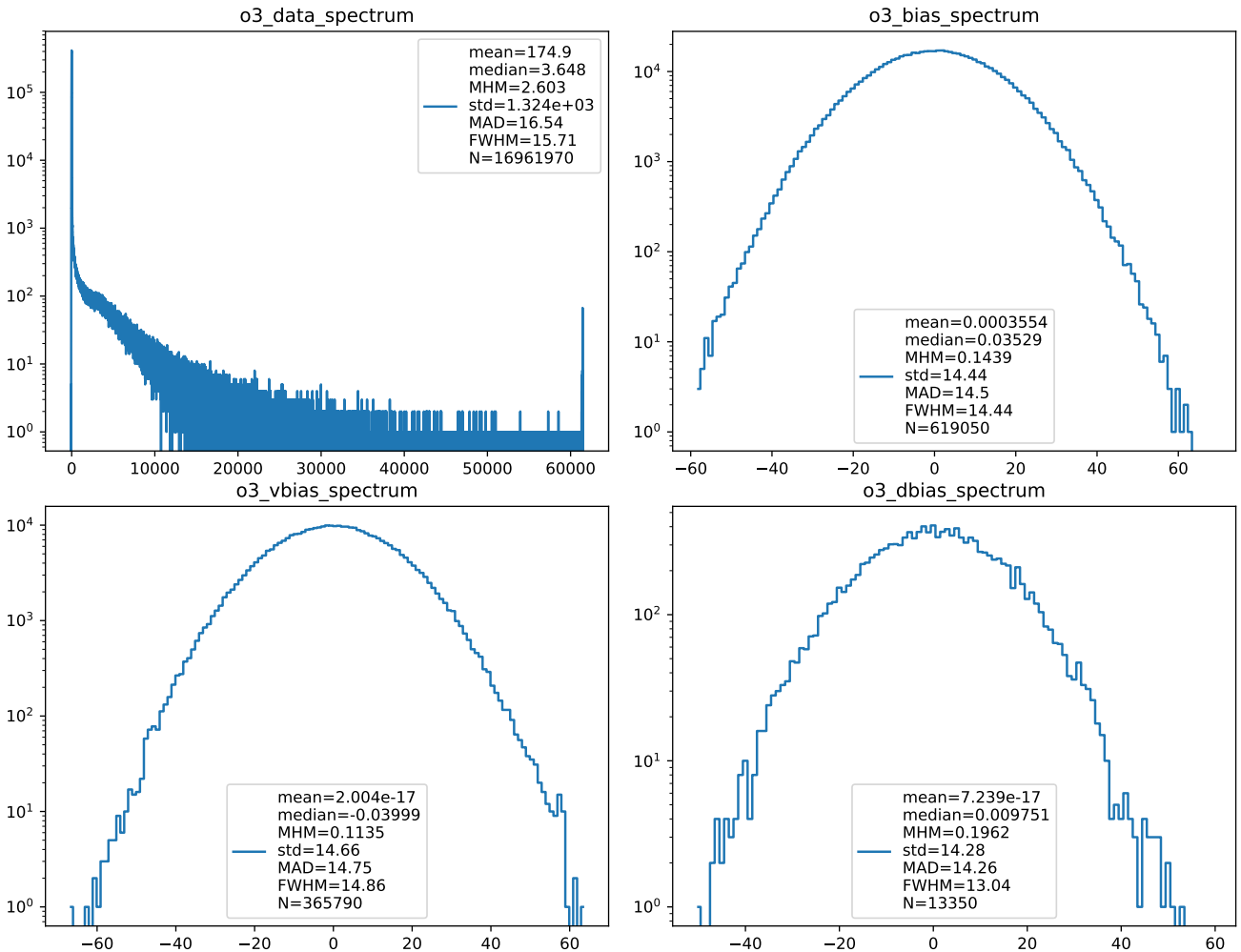
```
1  python Image.py analyse ImagePresentation/mean
       "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
       plot-sections --plot-spectrum
```



data becomes quite stable on both lines and columns

spectra with line and col corrections

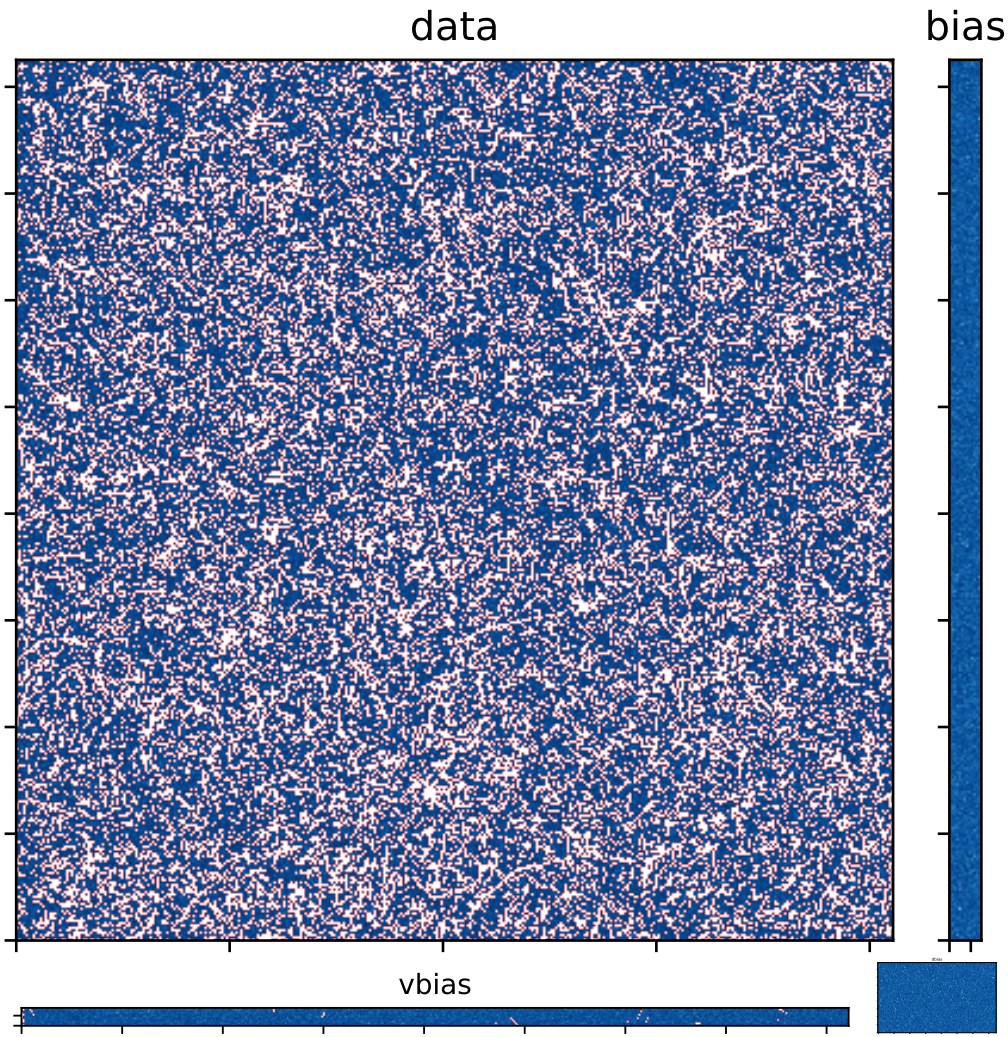
```
1   python Image.py analyse ImagePresentation/mean
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        plot-sections --plot-spectrum
```



this proceedure over the overscans succesfully removes their outliers allowing for the use of redundant estimators to control the estimations

removed above 40ADU with border 3

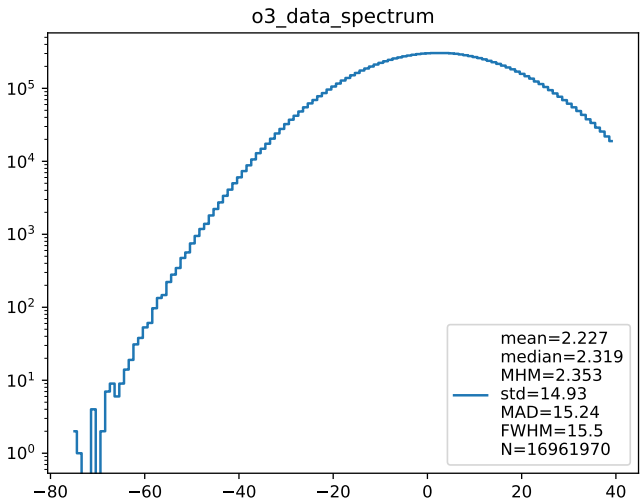
```
1   python Image.py analyse ImagePresentation/mean
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --remove-hits 40 3 --
        params-mode mean --plot-sections --plot-spectrum
```

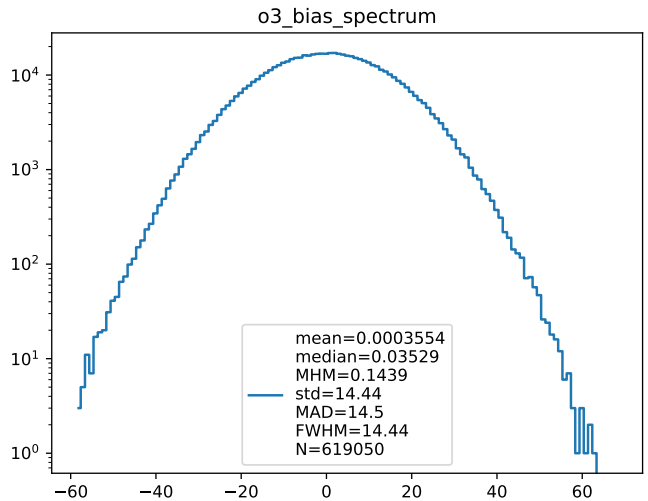


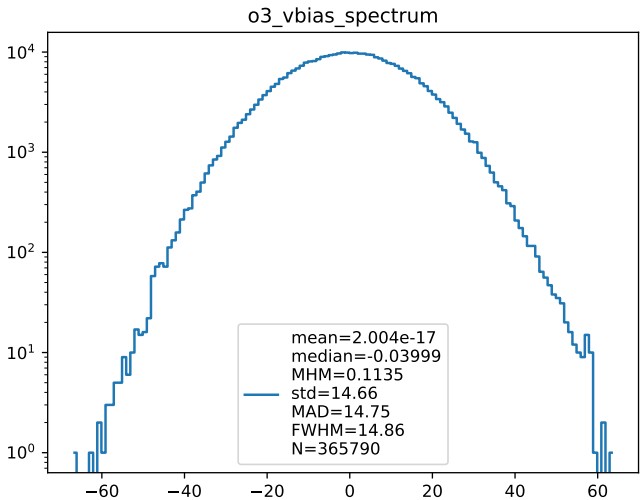
in the data region, a cluster removal algorithm is applied

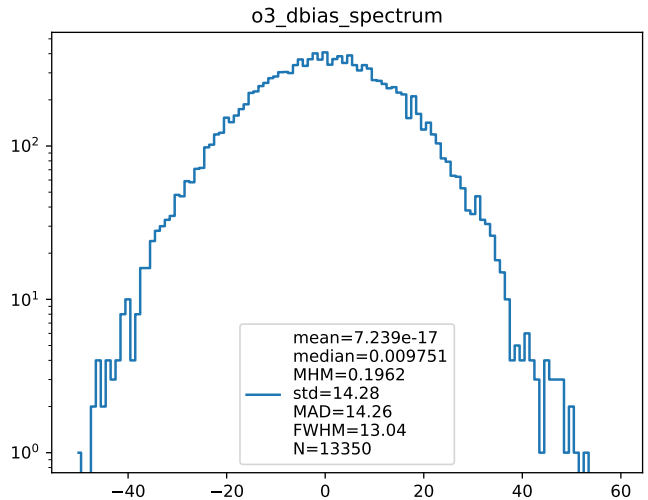
removed above 40ADU with border 3 spectra

```
1  python Image.py analyse ImagePresentation/mean
       "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --remove-hits 40 3 --
       params-mode mean --plot-sections --plot-spectrum
```









$$\sigma = 14.44$$

$$g\lambda = 2.32$$

$$g^2\lambda = 23.74$$

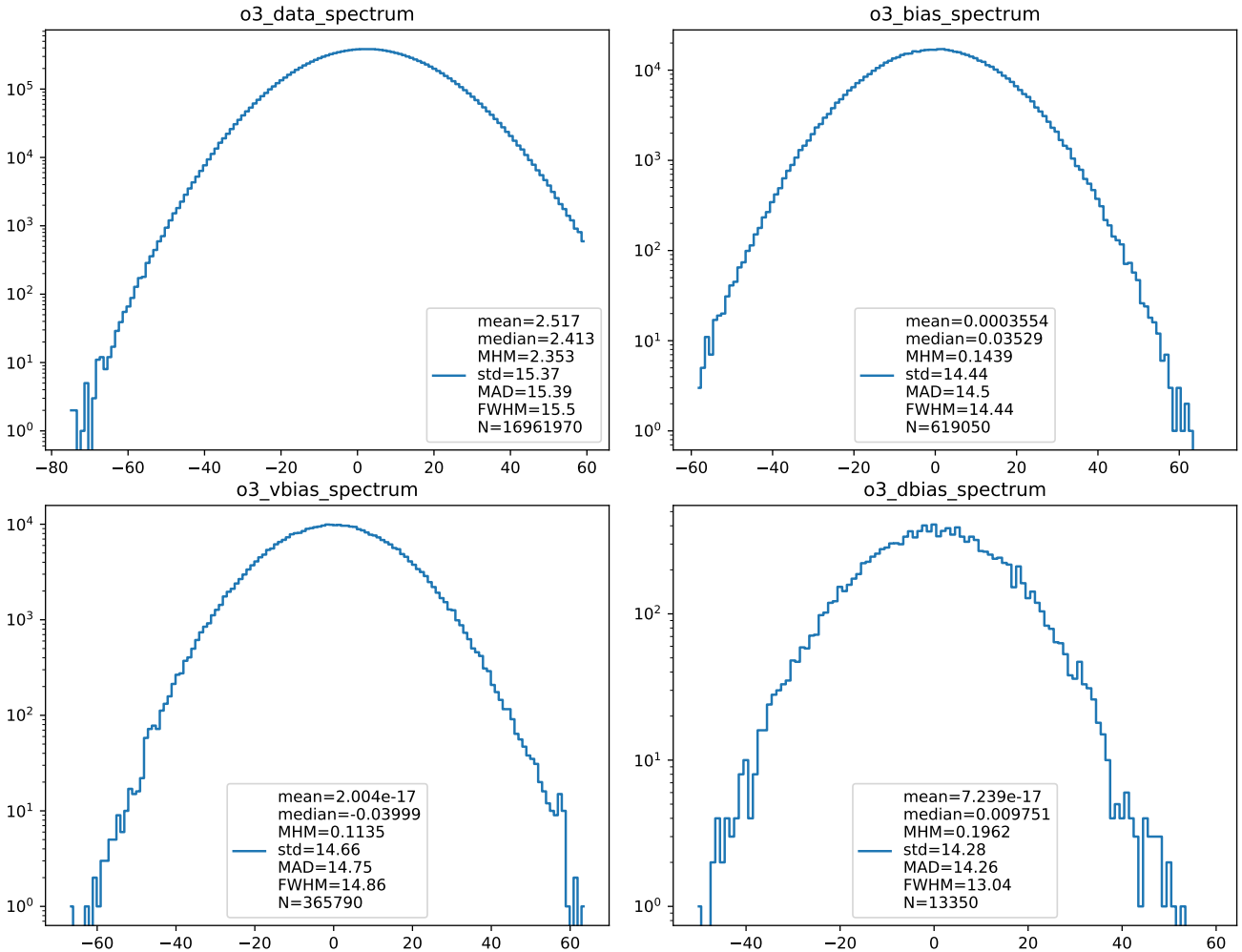
$$g = 10.23$$

$$\lambda = 0.23$$

perhaps 40ADU was too agressive since a large chunck of the Poisson-Norm distribution was also removed

$E < 60$ADU (+3) spectra

```
1   python Image.py analyse ImagePresentation/mean
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 60.0 3.0 --plot-spectrum
```



estimations

$$\sigma = 14.4$$

$$g\lambda = 2.41$$

$$g^2\lambda = 26.60$$

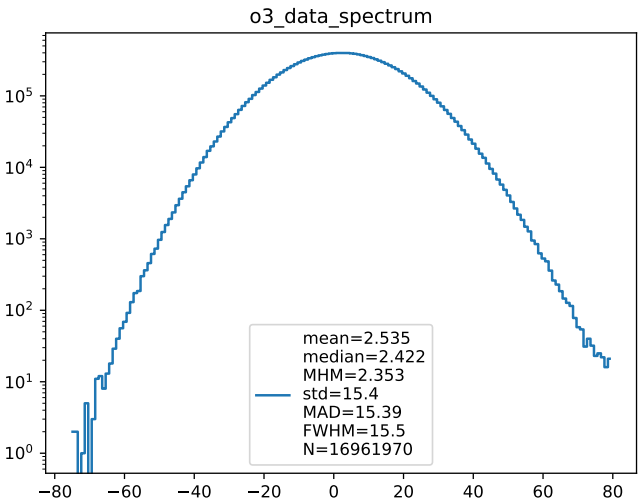
$$g = 11.04$$

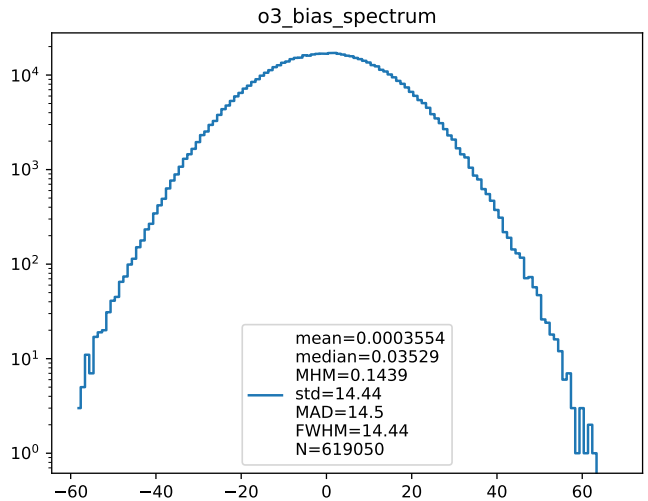
$$\lambda = 0.22$$

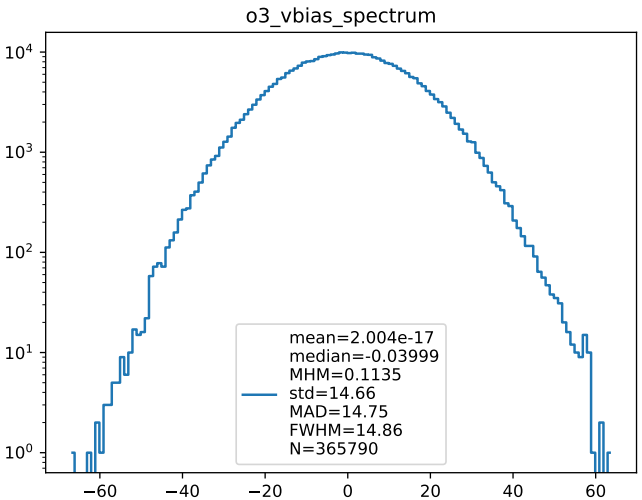
values are remarkably stable even though the value for the gain is higher than ~ 7 expected from the independent Cu estimation

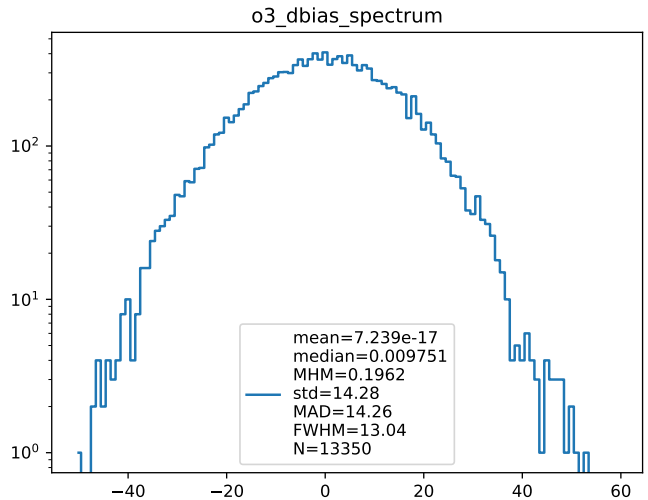
$E < 80$ADU (+2) spectra

```
1  python Image.py analyse ImagePresentation/mean
       "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
       remove-hits 80.0 2.0 --plot-spectrum
```









estimations

$$\sigma = 14.4$$

$$g\lambda = 2.42$$

$$g^2\lambda = 26.60$$

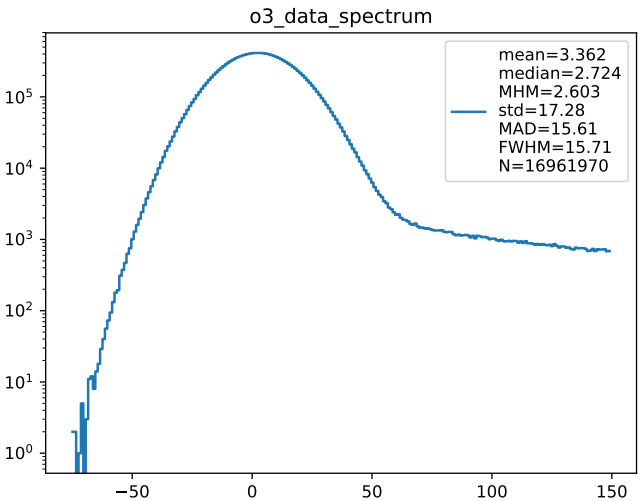
$$g = 10.99$$

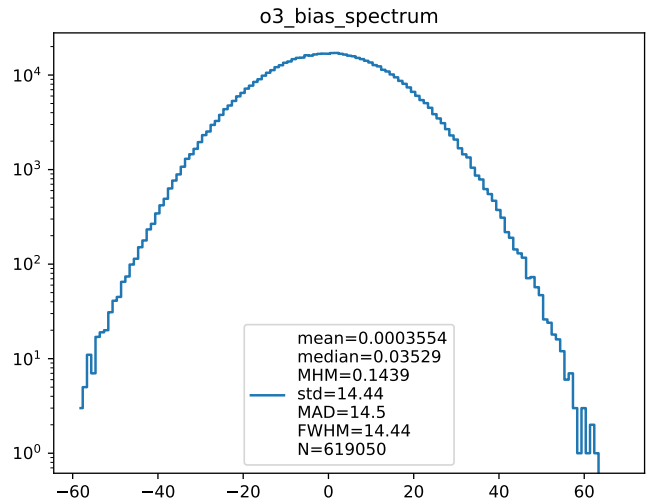
$$\lambda = 0.22$$

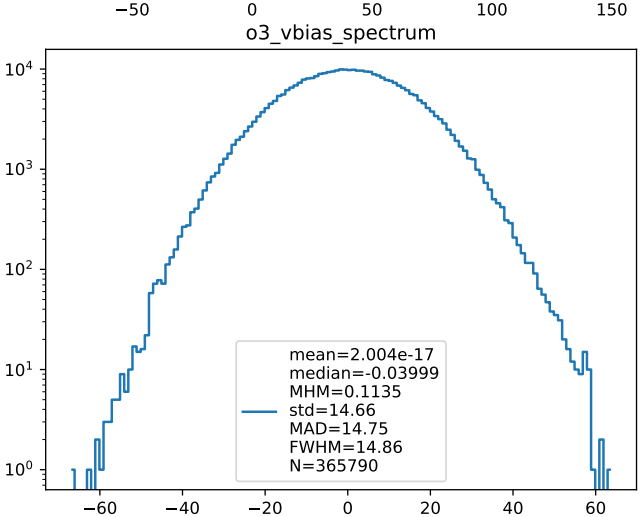
values are remarkably stable, although stable does not mean correct

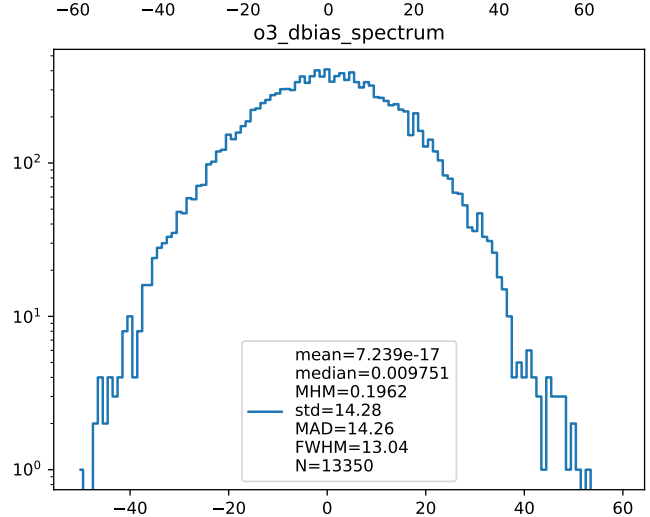
$E < 150$ADU (+0) spectra

```
1  python Image.py analyse ImagePresentation/mean
       "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
       remove-hits 150.0 0.0 --plot-spectrum
```









estimations

$$\sigma = 14.4$$

$$g\lambda = 2.72$$

$$g^2\lambda = 33.11$$

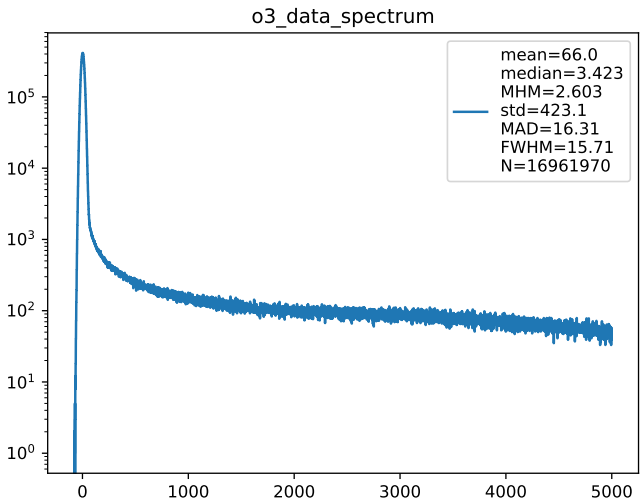
$$g = 12.17$$

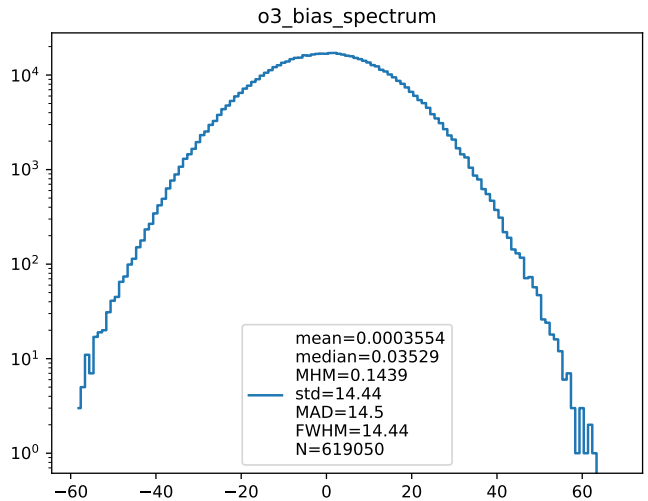
$$\lambda = 0.22$$

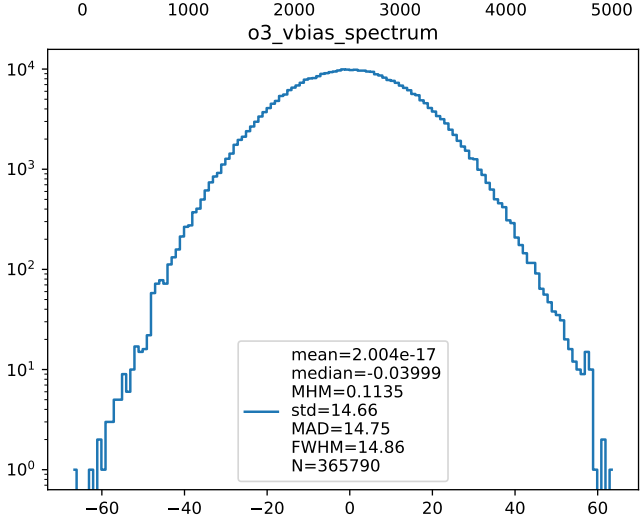
values are remarkably stable

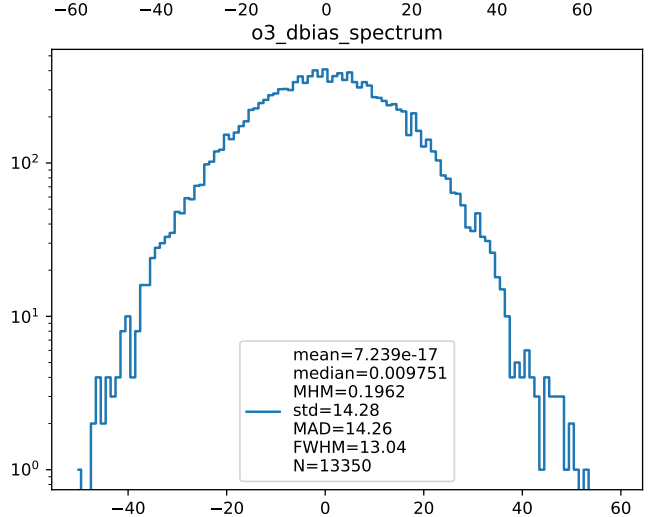
$E < 5000$ADU (+0) spectra

```
1  python Image.py analyse ImagePresentation/mean
       "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
       remove-hits 5000.0 0.0 --plot-spectrum
```









estimations

$$\sigma = 14.4$$

$$g\lambda = 3.42$$

$$g^2\lambda = 55.44$$

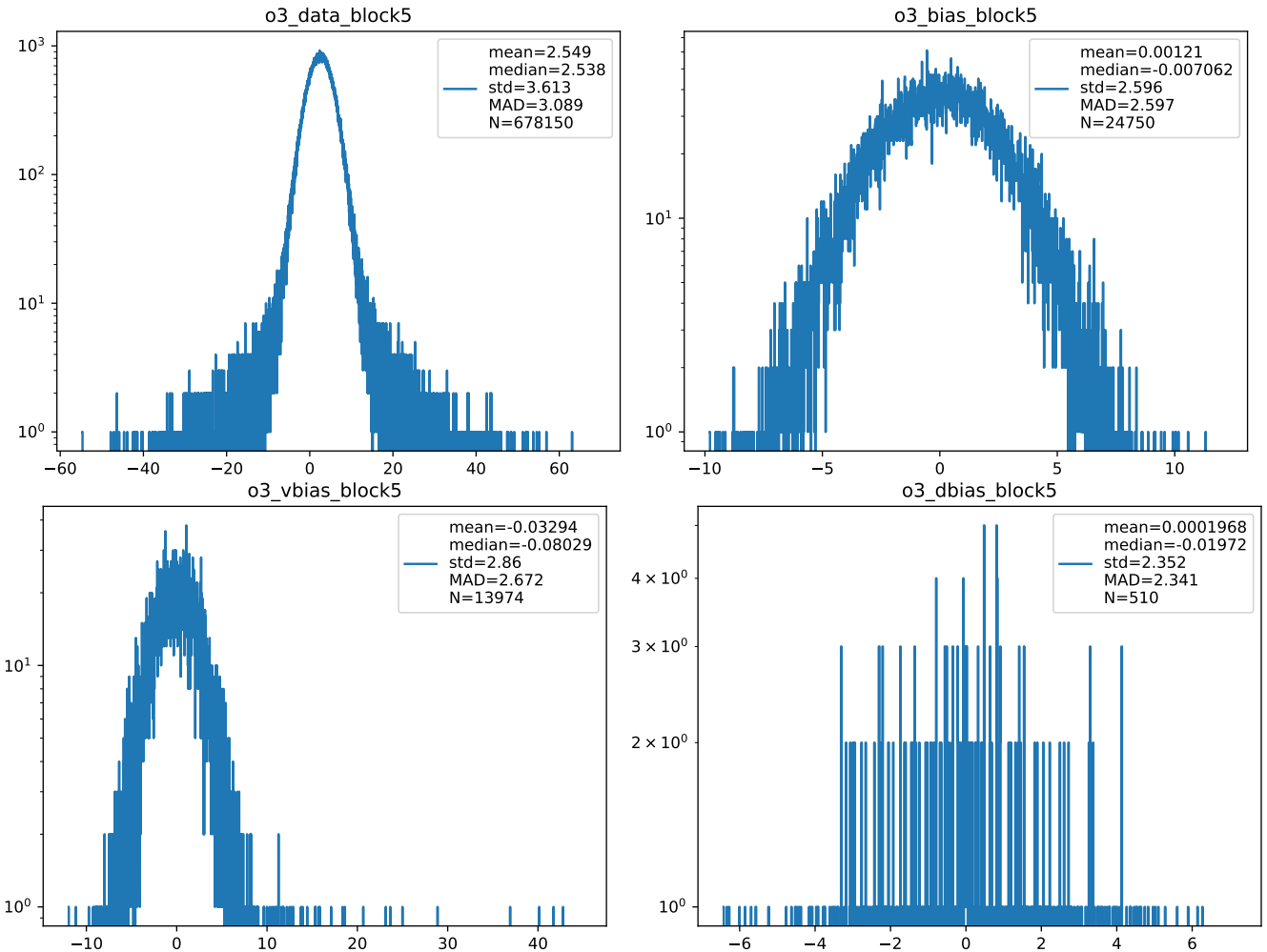
$$g = 16.21$$

$$\lambda = 0.21$$

sanity check: yes, it breaks! yupi

mean block 5x5 $E < 100$ADU (+3)

```
1  python Image.py analyse ImagePresentation/blockmean
       "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
       remove-hits 100.0 3.0 --plot-block-spectrum --block-function "np.nanmean(x)"
```

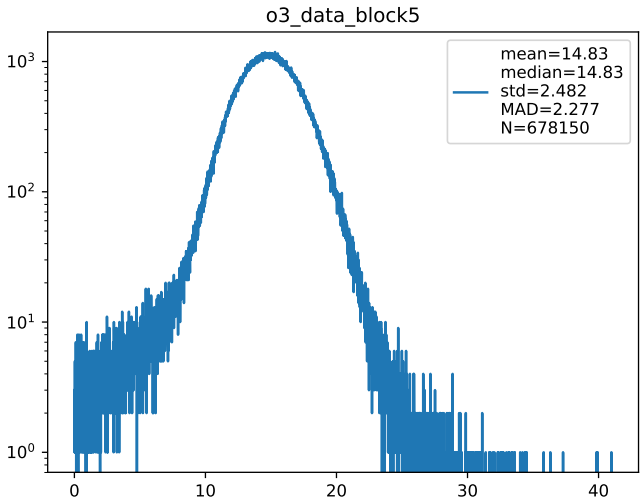


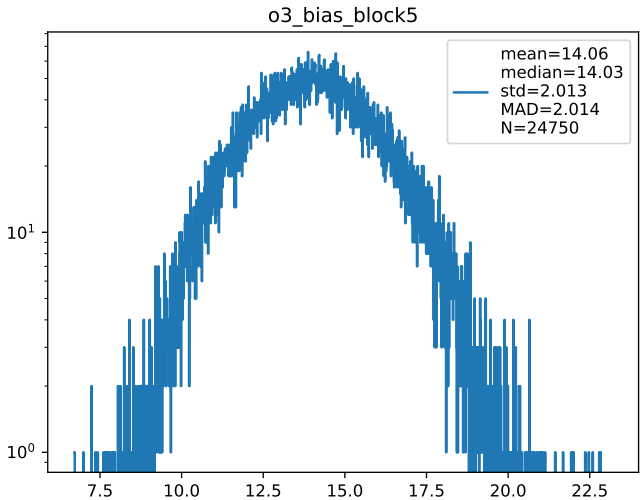
the previous estimation was global and relies on the homogeneity of the estimation throught the CCD. This plots show the distribution of the mean estimation for partitions of 5x5 of the image

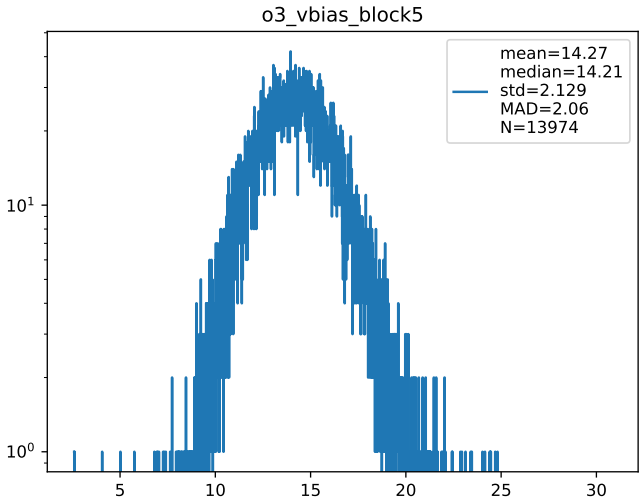
$$g\lambda = 2.54$$

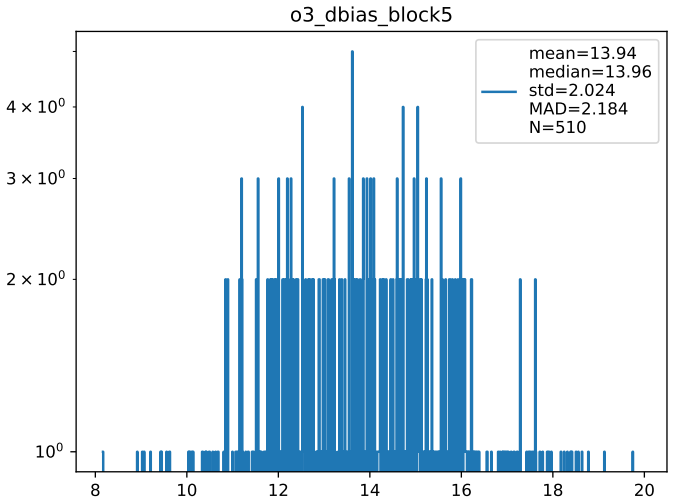
std block 5x5 $E < 100$ADU (+3)

```
1   python Image.py analyse ImagePresentation/blockstd
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 100.0 3.0 --plot-block-spectrum --block-function "(lambda y: np.nan if y==0 else
        y)(np.nanstd(x))"
```









distribution of the stds

$$\sigma = 14$$

$$g^2\lambda = 23$$

$$g = 9.07$$

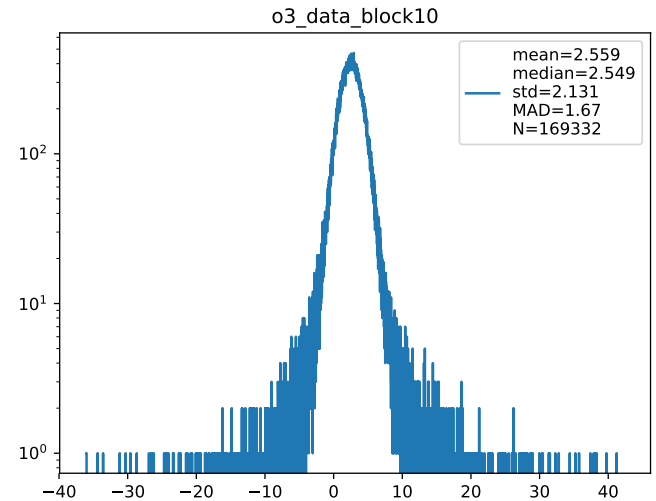
$$\lambda = 0.28$$

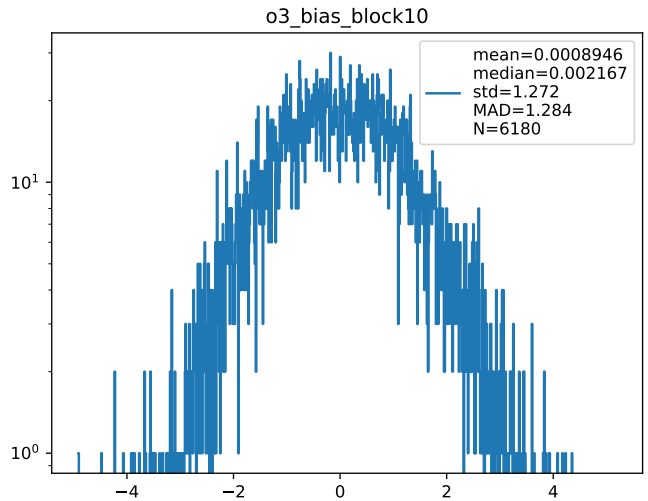
this result can be estimated with properly calculated errorbars!

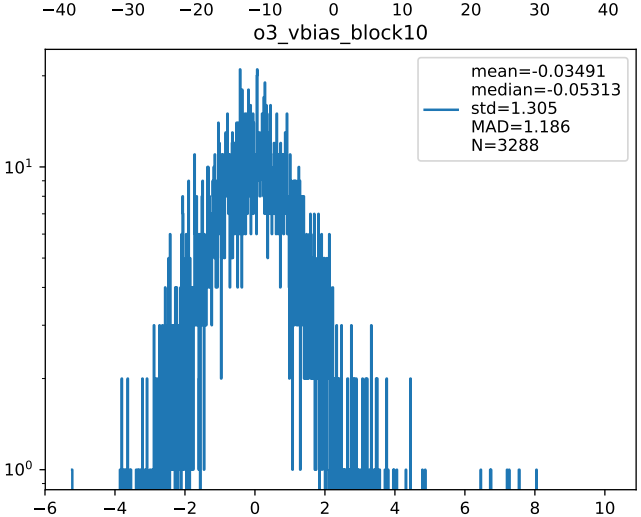
dramatic pause for appreciation

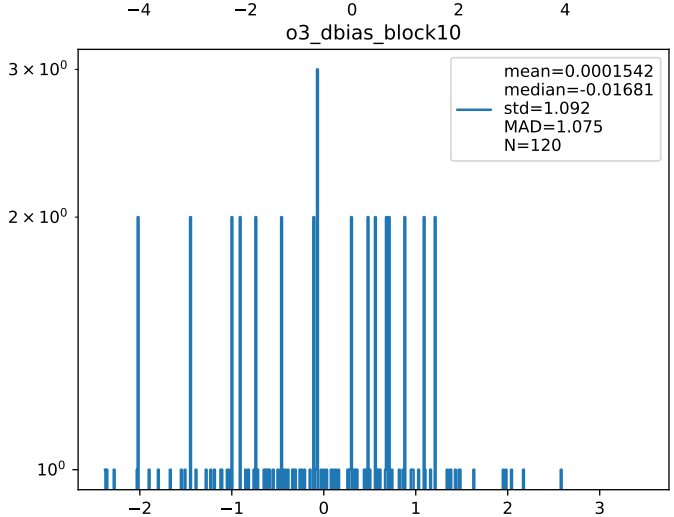
mean block 10x10 $E < 100$ADU (+3)

```
1   python Image.py analyse ImagePresentation/blockmean
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 100.0 3.0 --plot-block-spectrum --block-function "np.nanmean(x)"
```







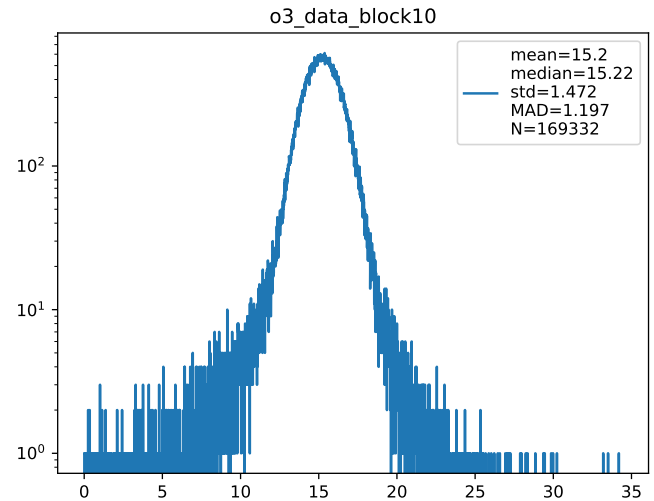


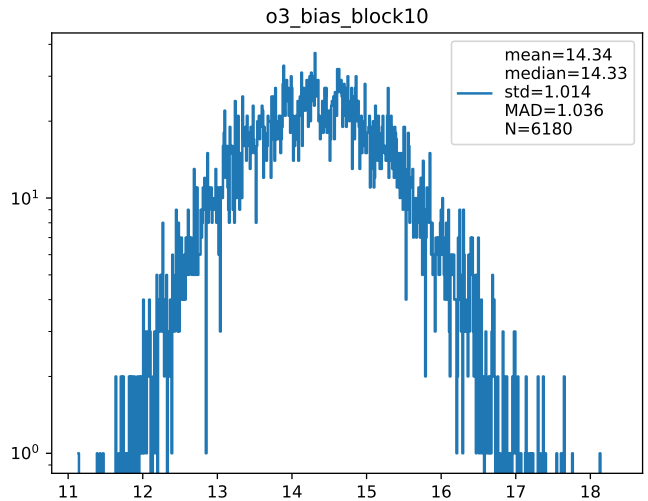
sanity check, repeat the calculations for 10x10 partitions

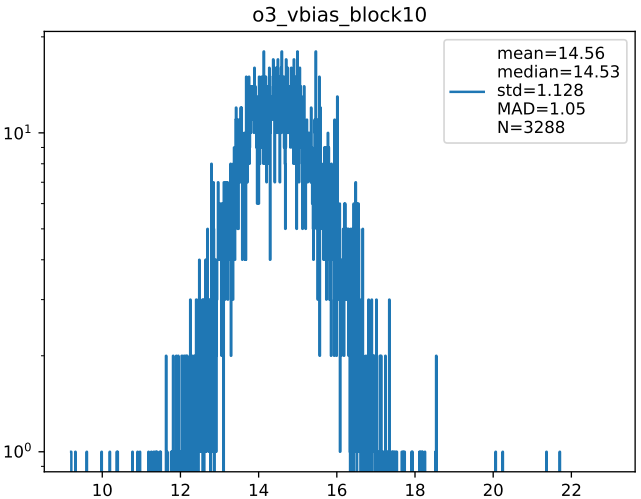
$$g\lambda = 2.55$$

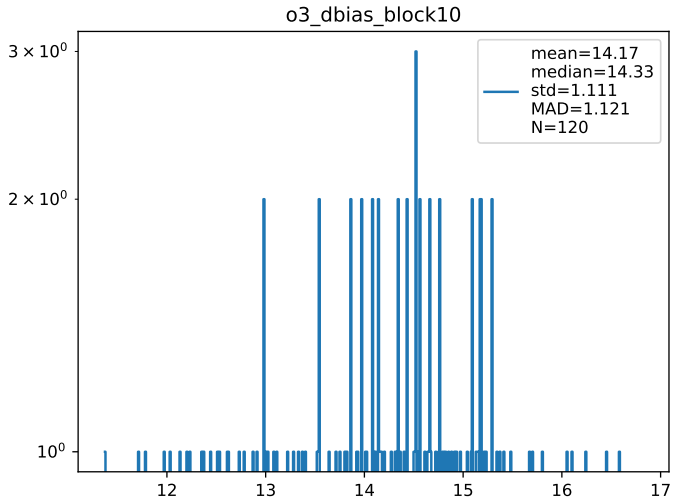
std block 10x10 $E < 100$ADU (+3)

```
1  python Image.py analyse ImagePresentation/blockstd
       "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
       remove-hits 100.0 3.0 --plot-block-spectrum --block-function "(lambda y: np.nan if y==0 else
       y)(np.nanstd(x))"
```









$$\sigma = 14.3$$

$$g^2\lambda = 27$$

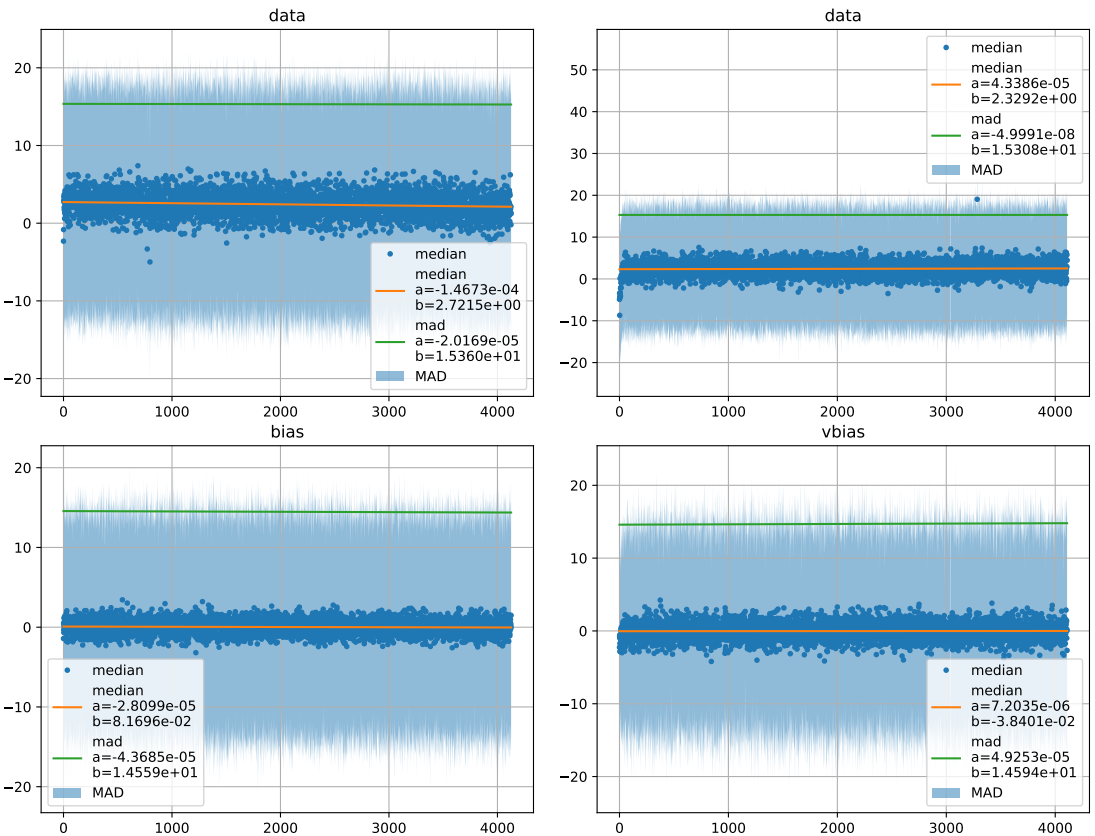
$$g = 10.41$$

$$\lambda = 0.24$$

there is a dependence on the size of the partition

evolution of DC through time $E < 100.0$ADU $(+3.0)$

```
1  python Image.py analyse ImagePresentation/dcevo
       "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
       remove-hits 100.0 3.0 --plot-sections
```



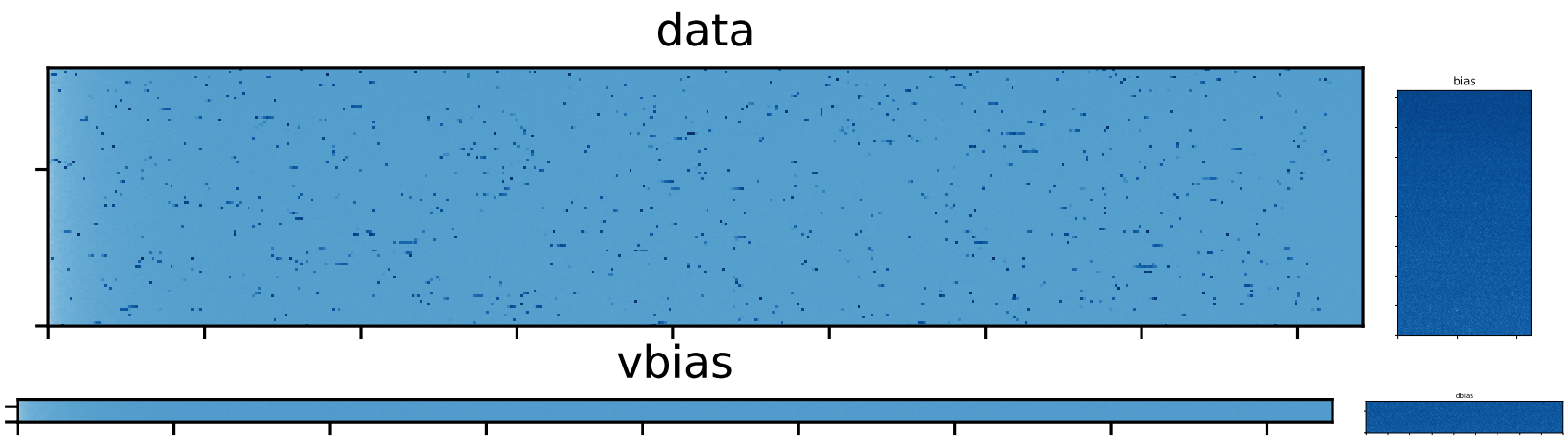
the obvious question you might be making me now is: can I plot the DC for time?

YES! there is a shift of $g\lambda = 0.5$ comparing the first read line to the last and it is one order of magnitude higher than the shifts in the overscan sections. Promising!

1x5

before you ask, here is the analysis on rebinned images

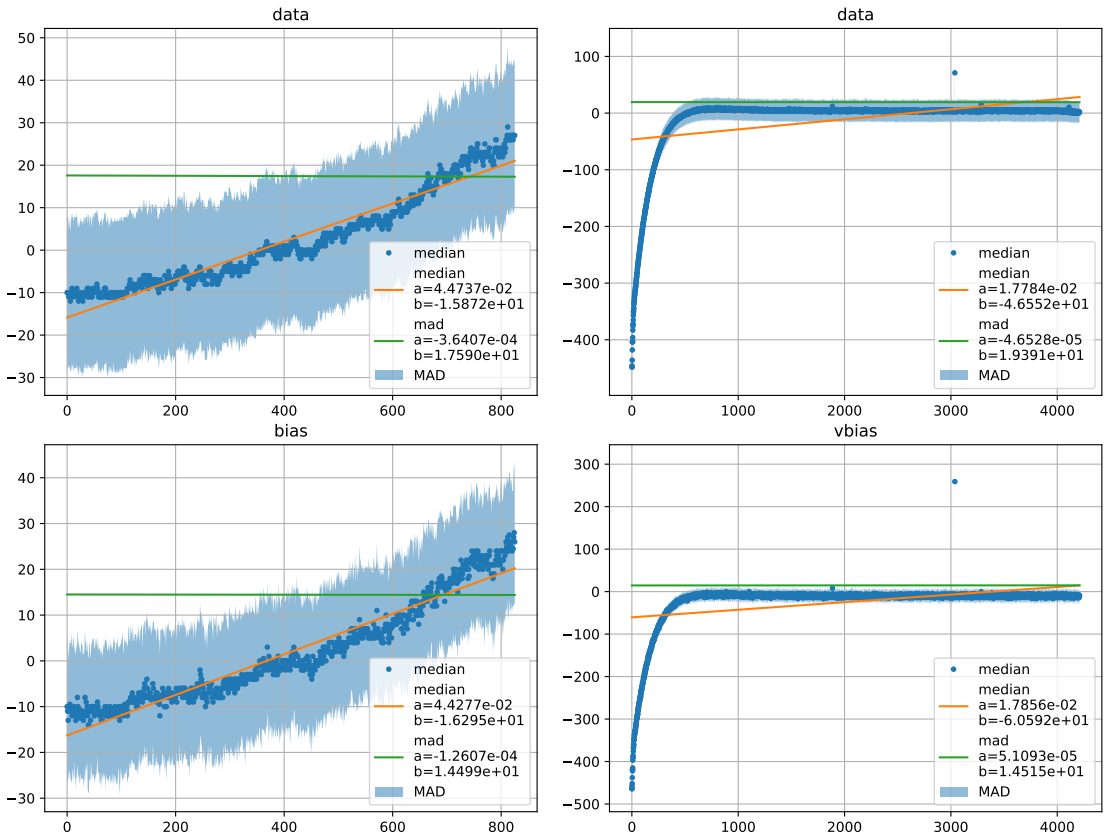
```
1  python Image.py analyse ImagePresentation/median1x5
       "/share/storage2/connie/data/runs/*/runID_*_12000_*_p*.fits.fz" --ohdu 3 --plot-sections --
       plot-spectrum
```



raw image

projections of the raw image

```
1  python Image.py analyse ImagePresentation/median1x5
       "/share/storage2/connie/data/runs/*/runID_*_12000_*_p*.fits.fz" --ohdu 3 --plot-sections --
       plot-spectrum
```

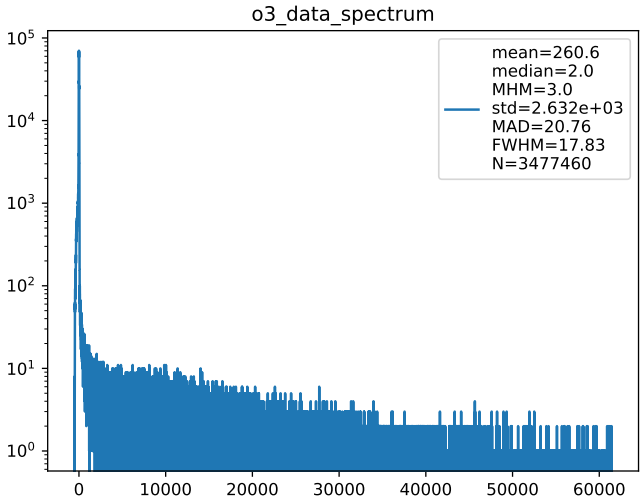


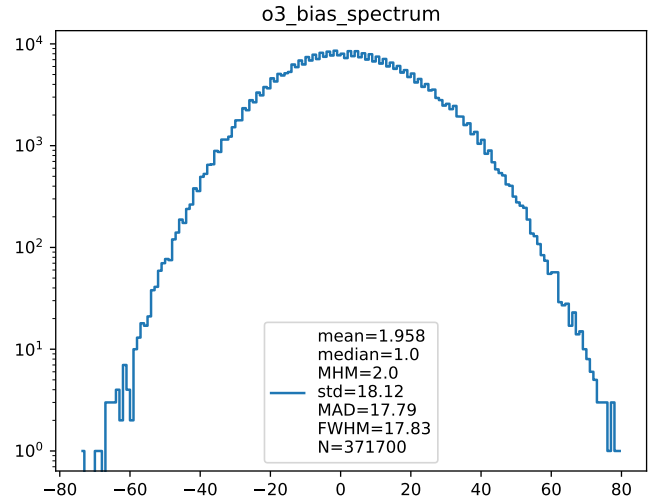
before the line and column corrections data show am interesting modulation on time and a sharp rise of baseline in the beginning of each line (this is also present in 1x1, but here it is more evident)

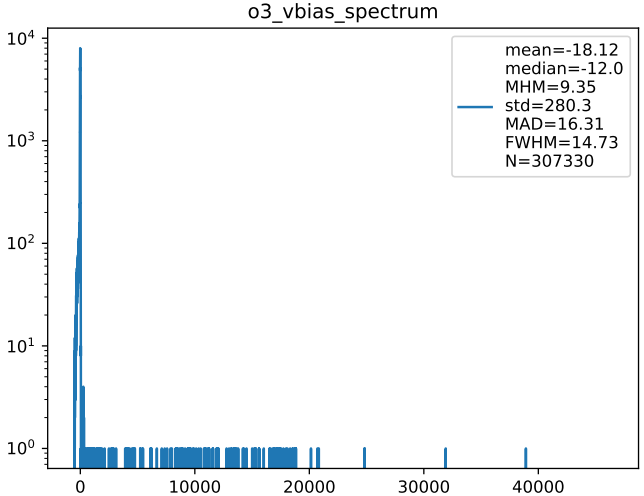
spectra of the raw image

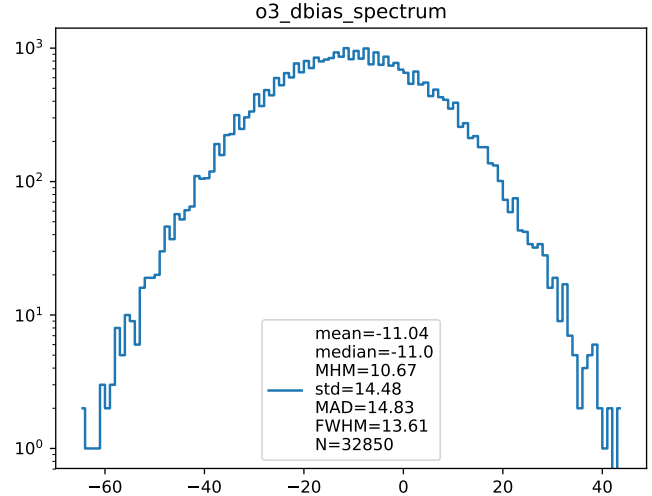
distributions are crowded with outliers

```
1  python Image.py analyse ImagePresentation/median1x5
       "/share/storage2/connie/data/runs/*/runID_*_12000_*_p*.fits.fz" --ohdu 3 --plot-sections --
       plot-spectrum
```





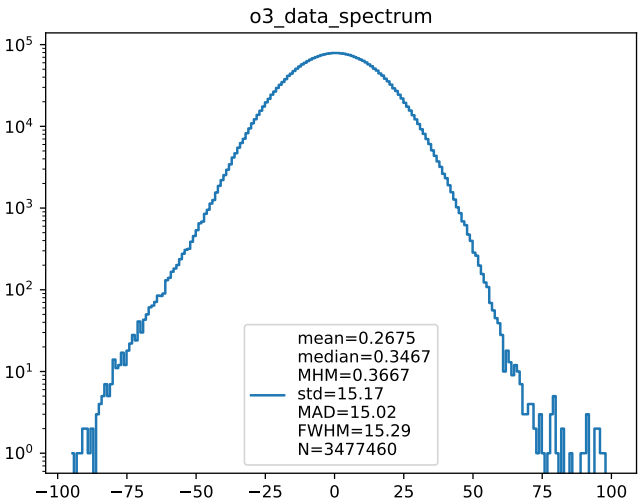


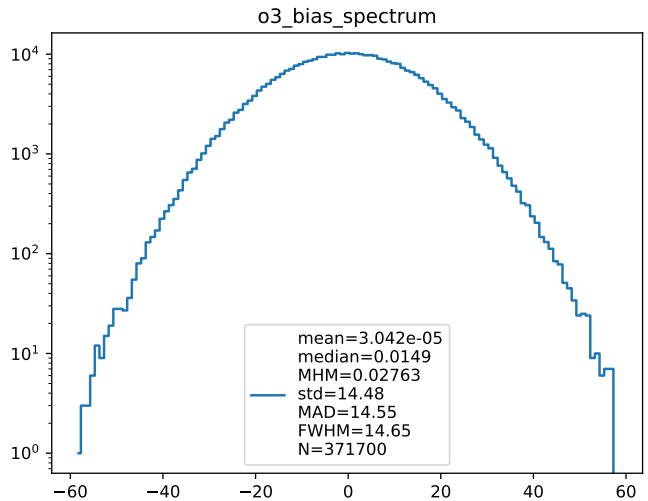


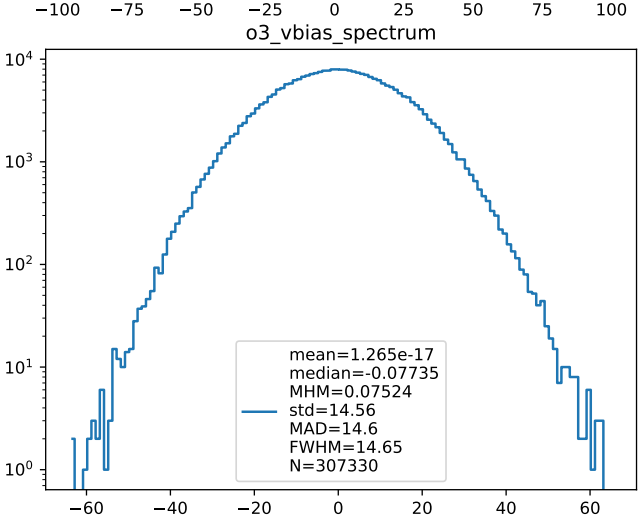
which leads to terrible outliers in the vertical overscan. The overscan looks clean, but it is actually convoluted with a sharp modulation

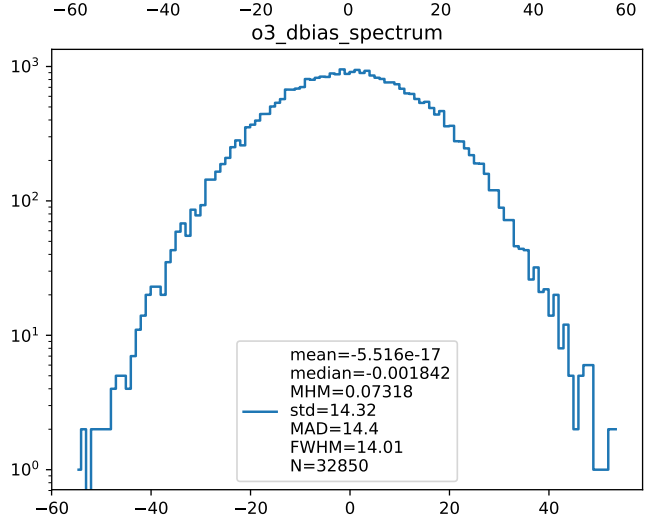
1x5 spectra with line and col corrections 1x5

```
1   python Image.py analyse ImagePresentation/dcevo1x5
        "/share/storage2/connie/data/runs/*/runID_*_12000_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 100.0 3.0 --plot-sections --plot-spectrum
```





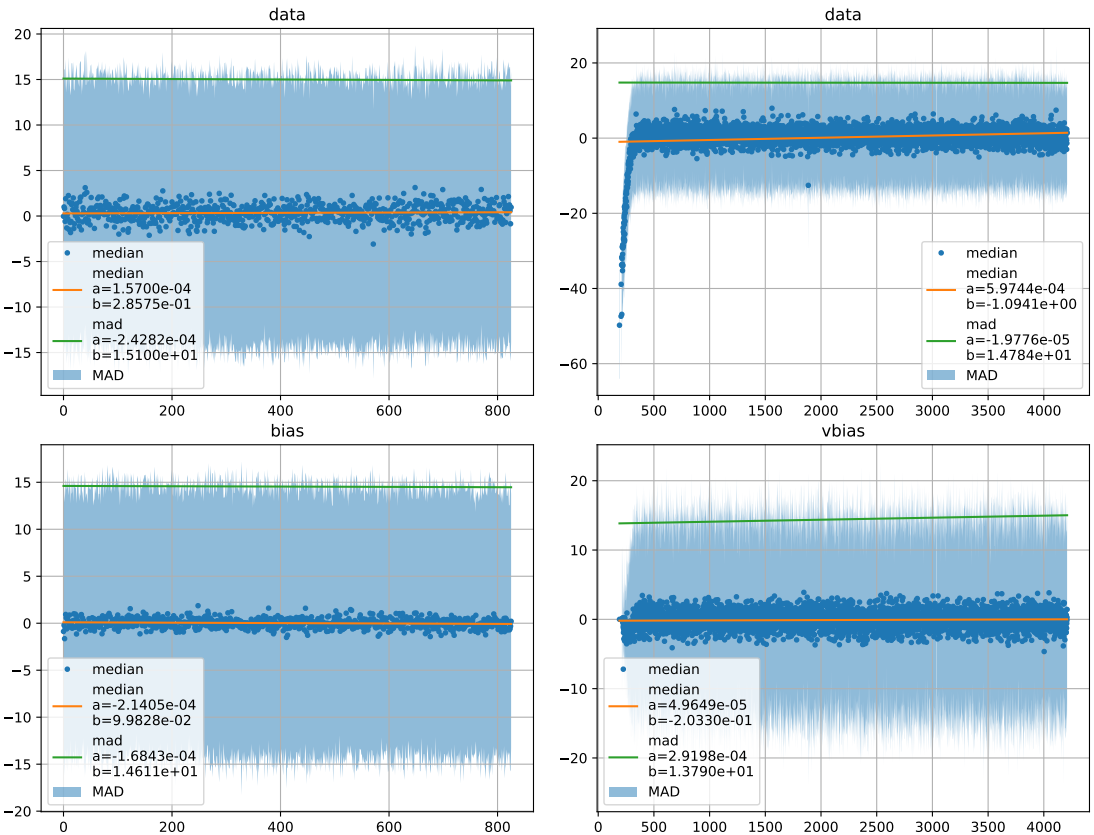




the algorithm manages to succesfully correct for the vertical overscan outliers, but there is a visible bump at negative energies of the data distribution. Let us investigate!

1x5 evolution of DC $E < 100.0$ADU (+3.0)

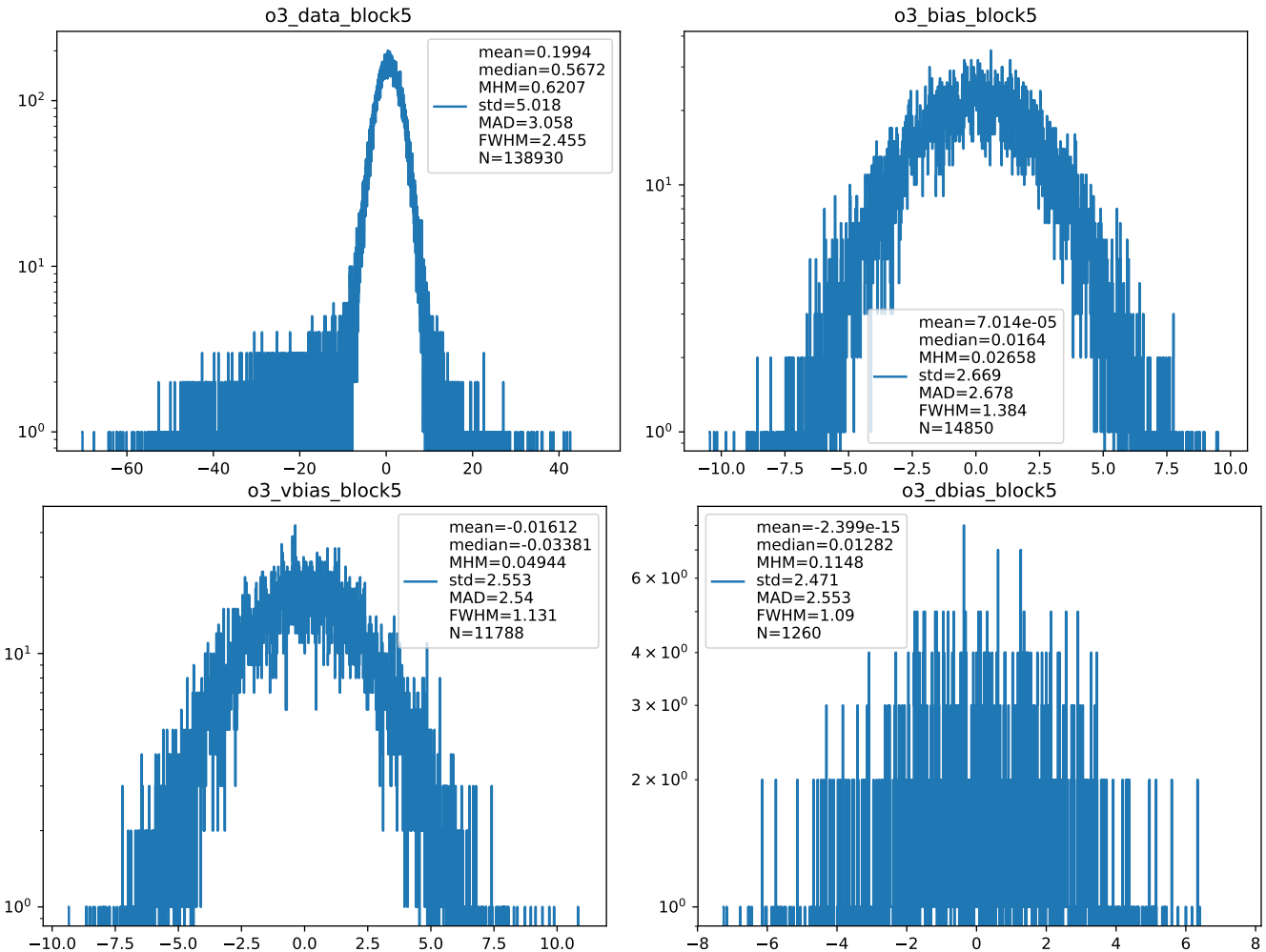
```
1   python Image.py analyse ImagePresentation/dcevo1x5
        "/share/storage2/connie/data/runs/*/runID_*_12000_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 100.0 3.0 --plot-sections --plot-spectrum
```



although the horizontal
overscan, the data still
fluctuations collapse clo
compatible with a simpl

1x5 mean block 5x5 $E < 100.0$ADU (+3.0)

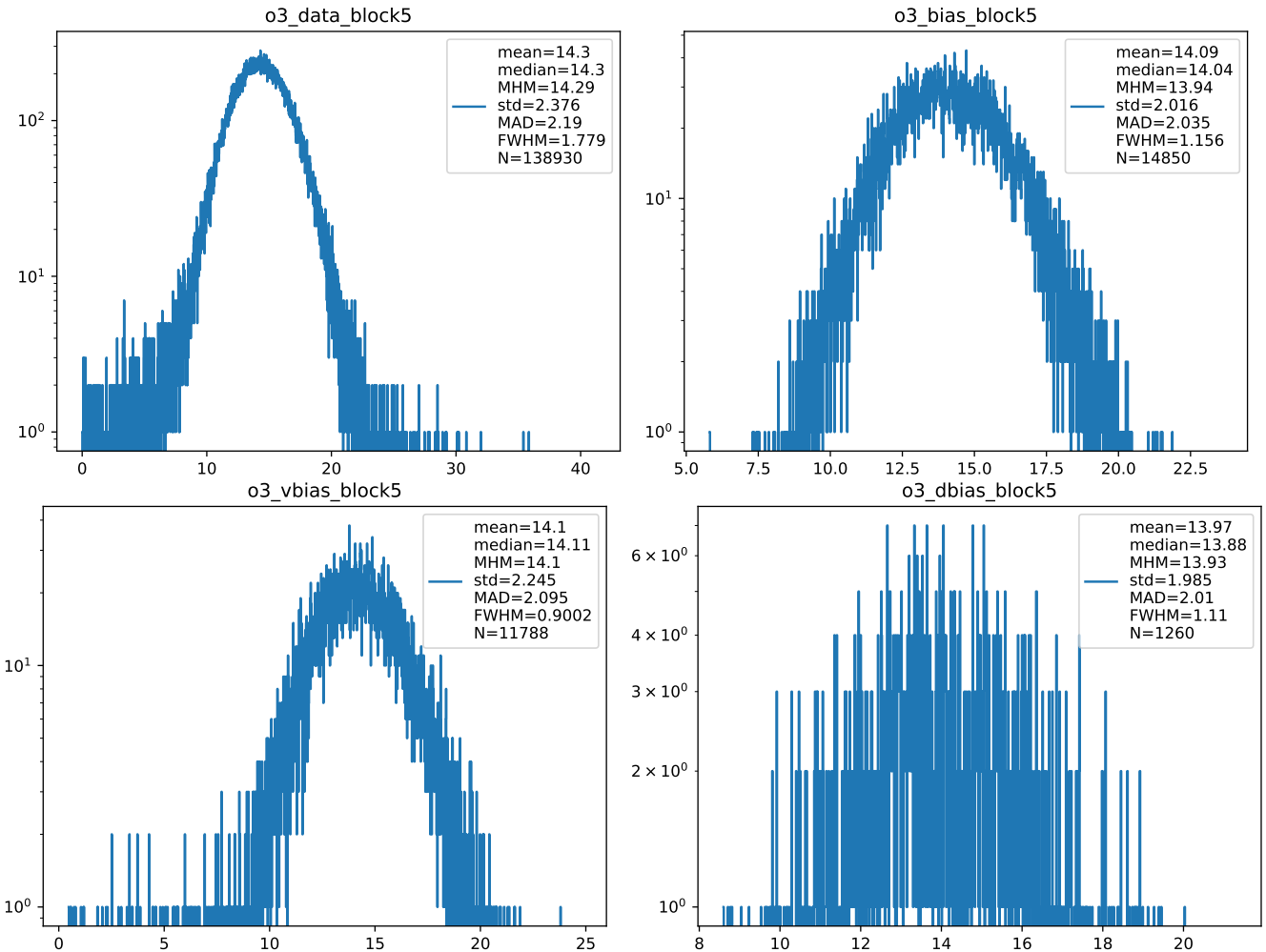
```
1   python Image.py analyse ImagePresentation/blockmean1x5
        "/share/storage2/connie/data/runs/*/runID_*_12000_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 100.0 3.0 --plot-block-spectrum --block-function "np.nanmean(x)"
```



the spatial partition algorithm is robust enough to ignore the modulation

1x5 std block 5x5 $E < 100.0$ADU $(+3.0)$

```
1   python Image.py analyse ImagePresentation/blockstd1x5
        "/share/storage2/connie/data/runs/*/runID_*_12000_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 100.0 3.0 --plot-block-spectrum --block-function "(lambda y: np.nan if y==0 else
        y)(np.nanstd(x))"
```



estimation

$$g\lambda = 0.58$$

$$\sigma = 14.1$$

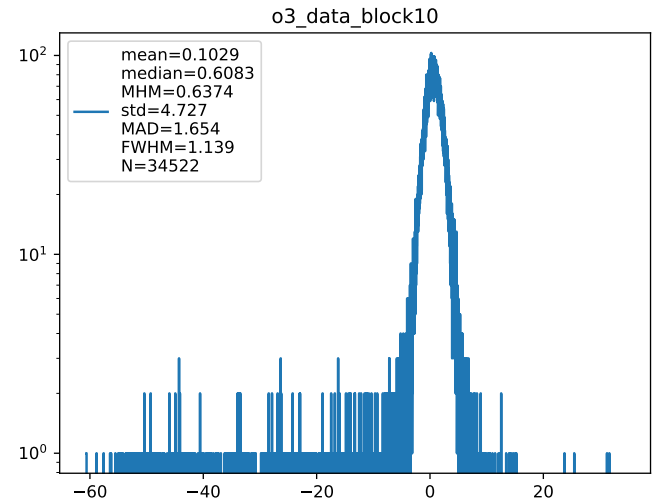
$$g^2\lambda = 5.68$$

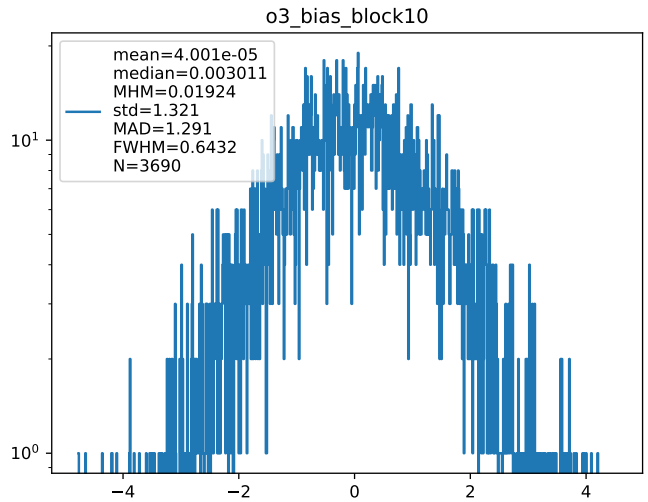
$$g = 9.79$$

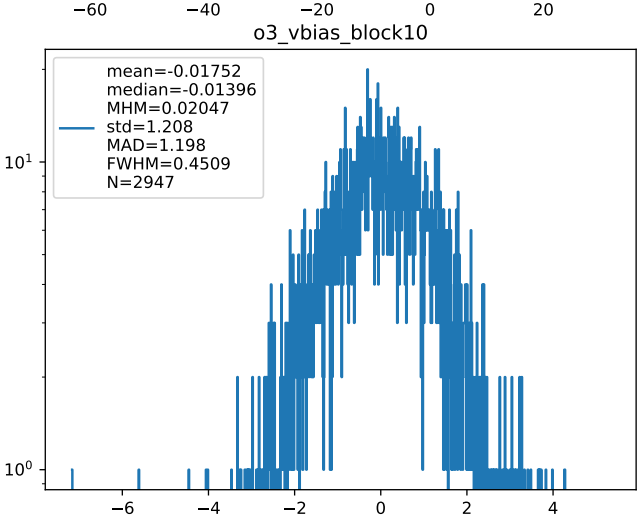
$$\lambda = 0.06$$

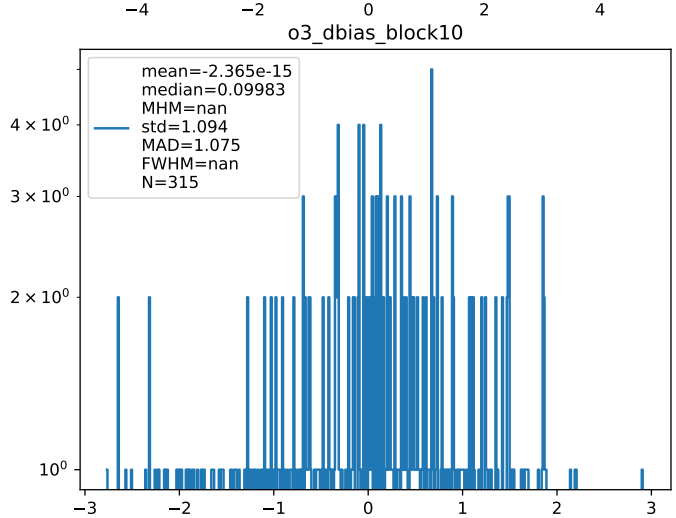
1x5 mean block 10x10 $E < 100.0$ADU $(+3.0)$

```
1  python Image.py analyse ImagePresentation/blockmean1x5
       "/share/storage2/connie/data/runs/*/runID_*_12000_*_p*.fits.fz" --ohdu 3 --params-mode mean --
       remove-hits 100.0 3.0 --plot-block-spectrum --block-function "np.nanmean(x)"
```





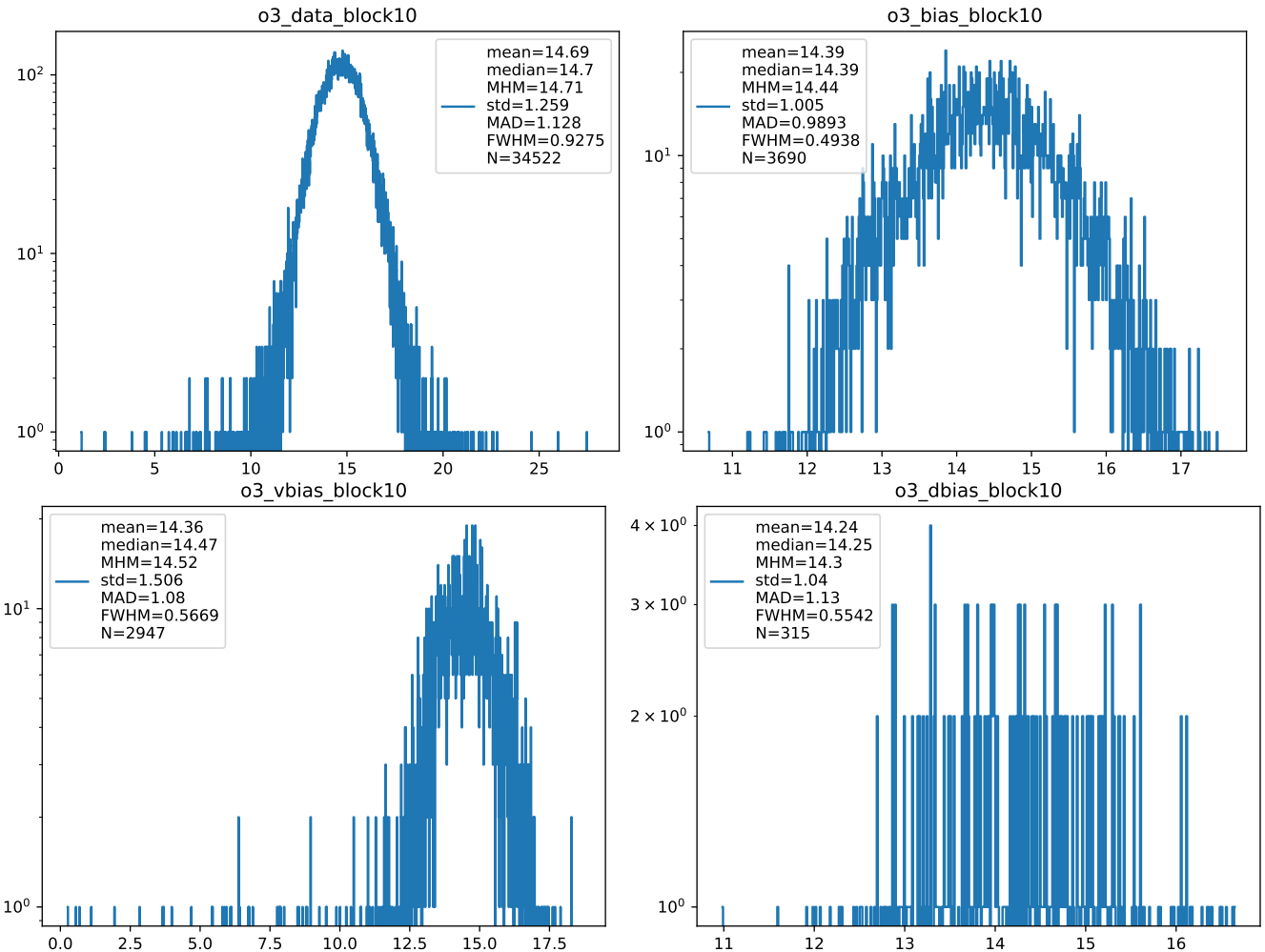




repeating the a analysis for larger partitions

1x5 std block 10x10 $E < 100.0$ADU (+3.0)

```
1   python Image.py analyse ImagePresentation/blockstd1x5
        "/share/storage2/connie/data/runs/*/runID_*_12000_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 100.0 3.0 --plot-block-spectrum --block-function "(lambda y: np.nan if y==0 else
        y)(np.nanstd(x))"
```



estimation

$$g\lambda = 0.62$$

$$\sigma = 14.4$$

$$g^2\lambda = 8.73$$

$$g = 14.08$$

$$\lambda = 0.04$$

summary

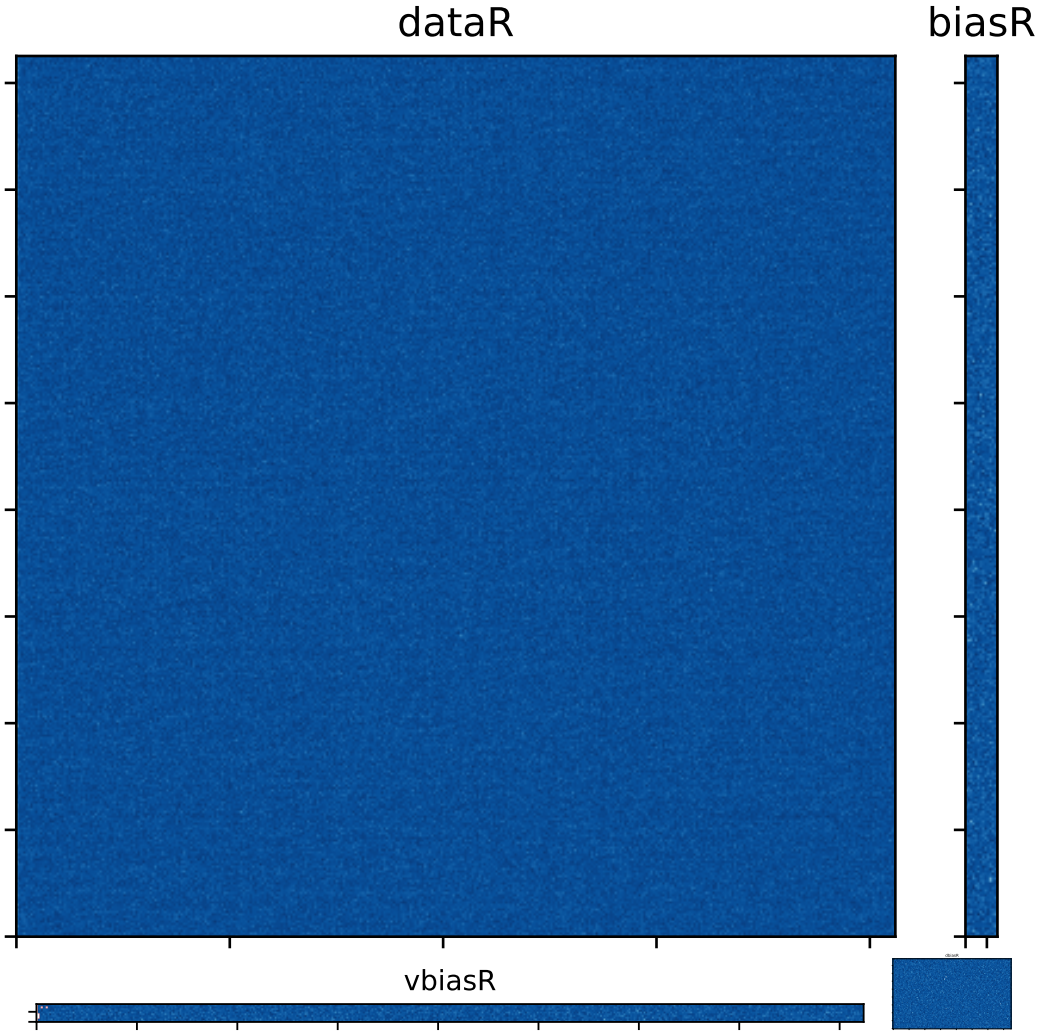
- yet another tool for analysing the data and performing simulations
- I believe that redundancy of tools for analysis is needed in order to reduce the chance of a major analysis mistake not to mention the constructive competition
- testing against simulations is zero-th order and I am working on it
- testing agains the official tools is next in line – Carla's share of duty
- promising insights that are being tested against simulations, presentation next week

summary

- hypothesis: the gain overestimation is related to the vertical overscan subtraction (which contains small dark current) that ends up shifting the center of the data distribution, therefore underevaluating $g\lambda$. On the other hand, $g^2\lambda$ is unaffected by this shift and thus although a small shift for the data mean, it is a great shift for the gain!
- verification: not to lose the advantage of (partially) removing the horizontal modulation, I will estimate the vertical overscan shift in relation to the horizontal overscan using the half of the section not affected by the modulation. Then after the subtraction, I will reintroduce this shift
- there is also the correlated noise which was not treated (yet) here. The correlated noise affects the terms in different amounts and may be a source of incertainty, I plan to attempt subtracting the mirrored right side image by requiring that it maximally reduces the noise in the overscan.the partitioning analysis opens the possibility of performing spatial dc/data discrimination by estimating the spread of the partition. This

right-side with line and col corrections

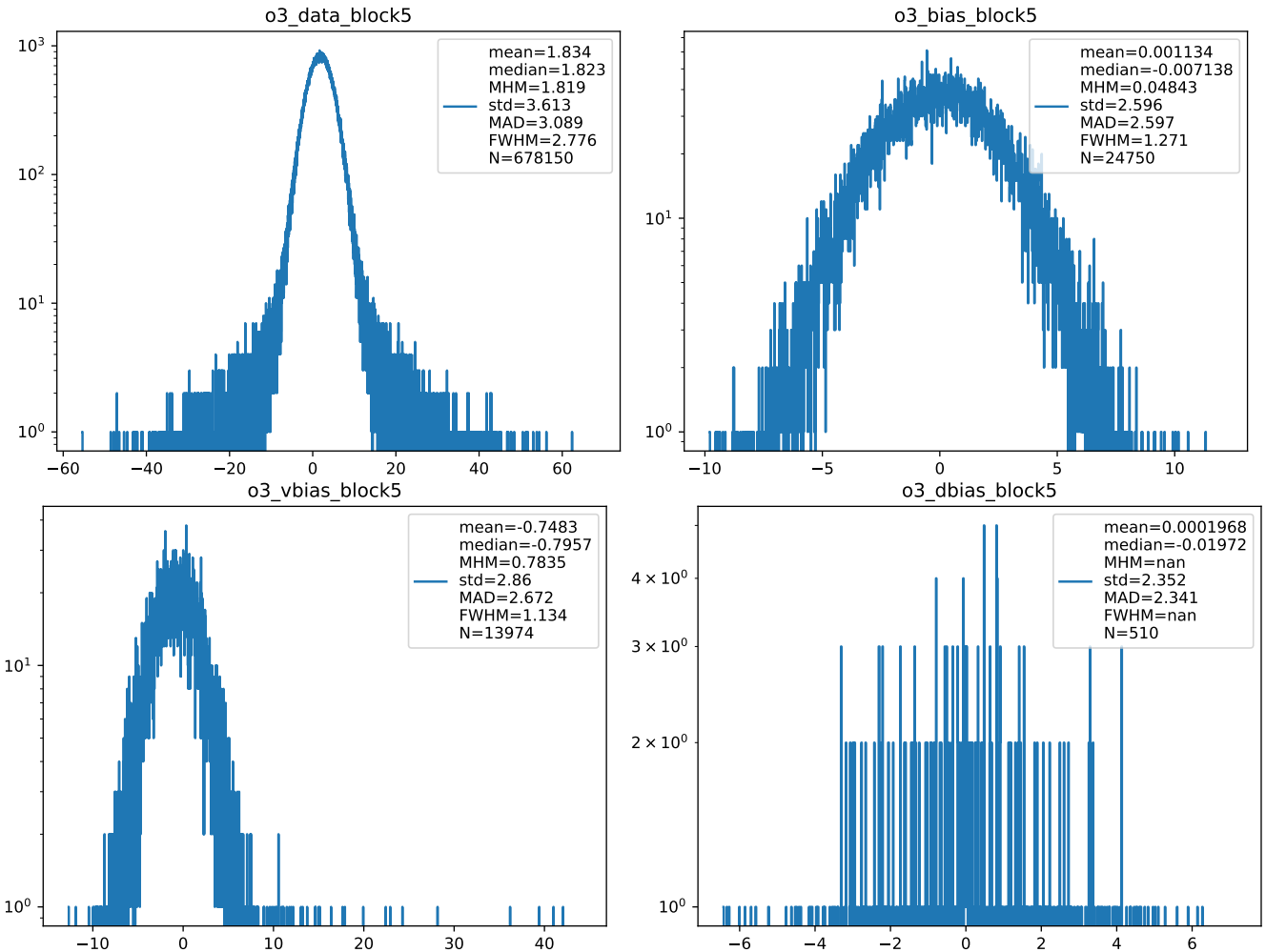
```
1   python Image.py analyse ImagePresentation/blockstd1x5
        "/share/storage2/connie/data/runs/*/runID_*_12000_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 100.0 3.0 --plot-block-spectrum --block-function "(lambda y: np.nan if y==0 else
        y)(np.nanstd(x))"
```



right side

vfix mean block 5x5 $E < 100$ADU $(+3)$

```
1  python Image.py analyse ImagePresentation/blockmean
       "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
       remove-hits 100.0 3.0 --plot-block-spectrum --block-function "np.nanmean(x)" --fix-vbias
```

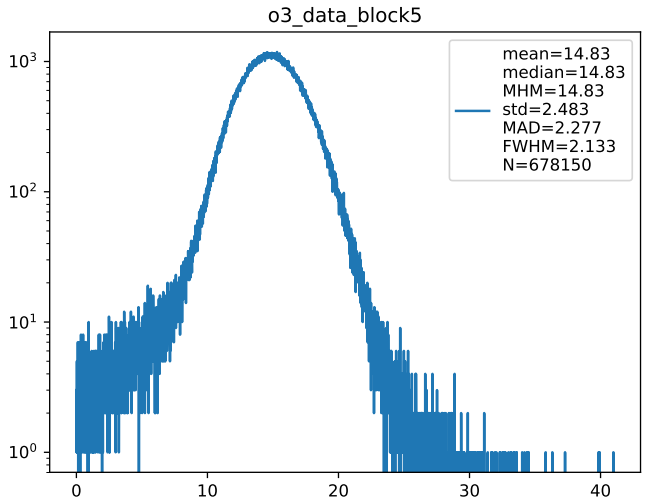


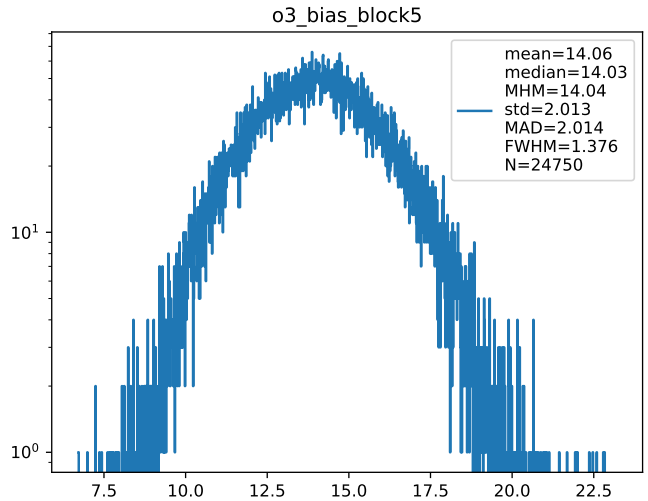
the previous estimation was global and relies on the homogeneity of the estimation throught the CCD. This plots show the distribution of the mean estimation for partitions of 5x5 of the image

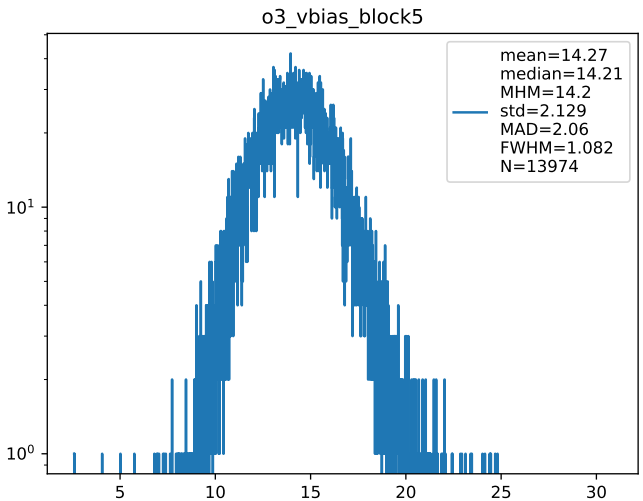
$$g\lambda = 2.54$$

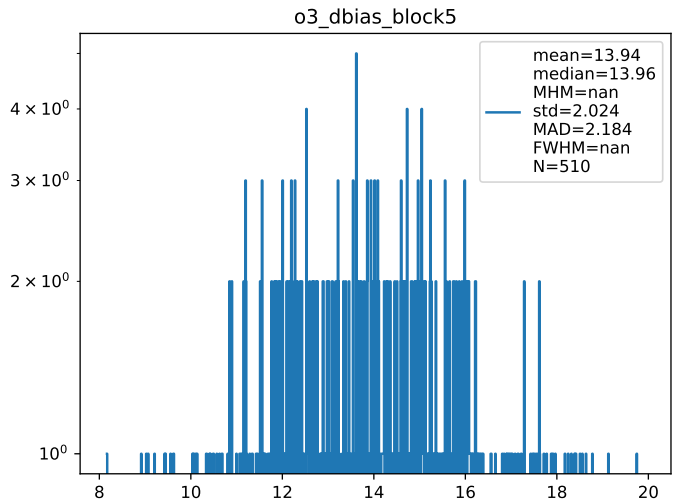
vfix std block 5x5 $E < 100$ADU (+3)

```
1   python Image.py analyse ImagePresentation/blockstd
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 100.0 3.0 --plot-block-spectrum --block-function "(lambda y: np.nan if y==0 else
        y)(np.nanstd(x))" --fix-vbias
```









distribution of the stds

$$\sigma = 14$$

$$g^2\lambda = 23$$

$$g = 9.07$$

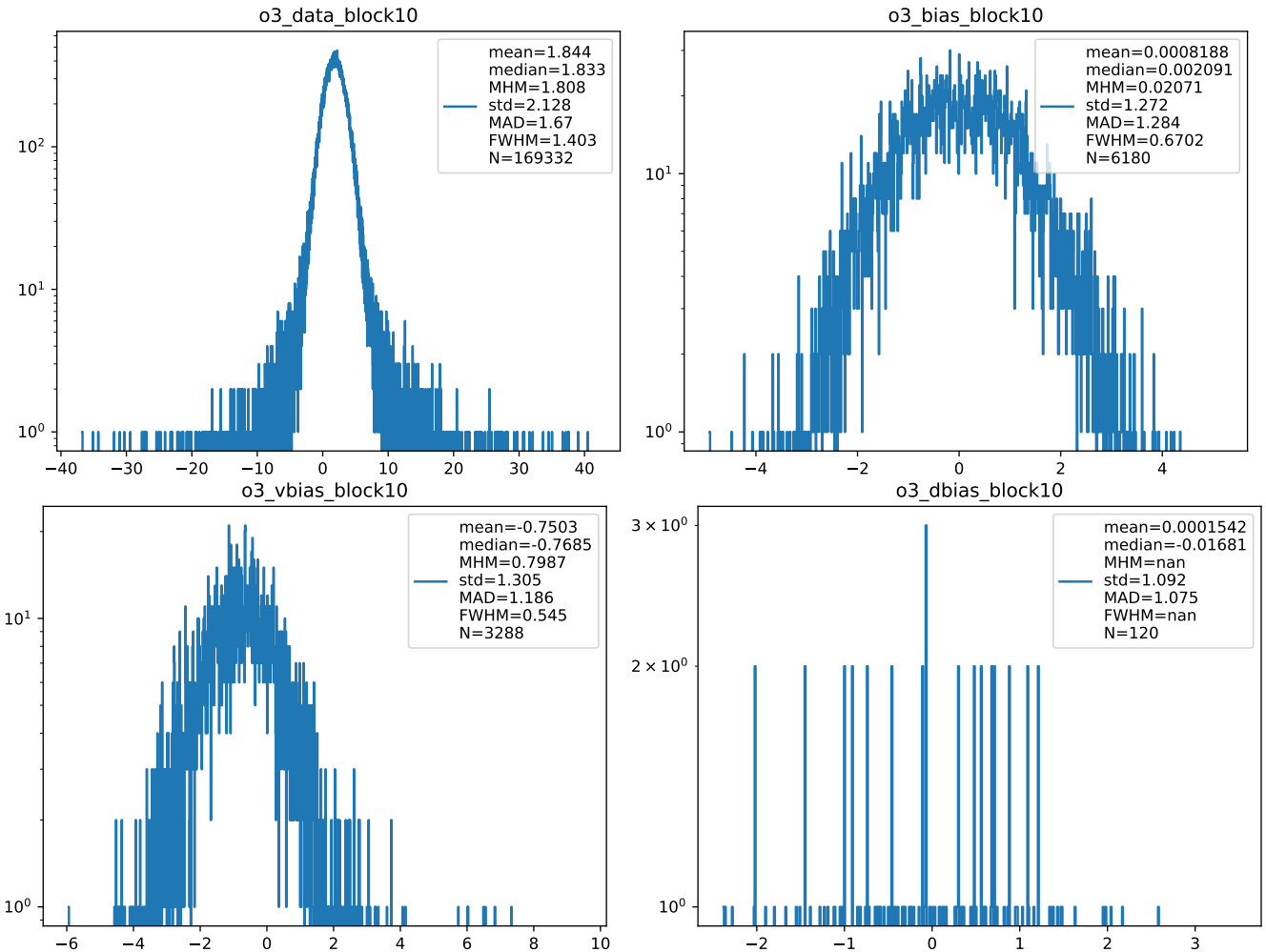
$$\lambda = 0.28$$

this result can be estimated with properly calculated errorbars!

dramatic pause for appreciation

vfix mean block 10x10 $E < 100$ADU (+3)

```
1   python Image.py analyse ImagePresentation/blockmean
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 100.0 3.0 --plot-block-spectrum --block-function "np.nanmean(x)" --fix-vbias
```

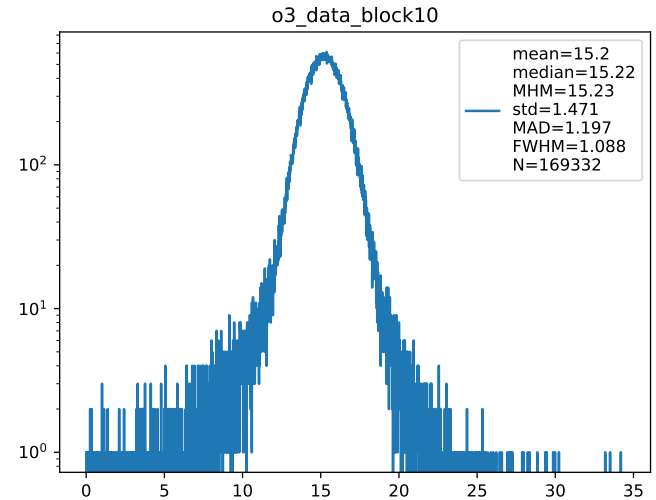


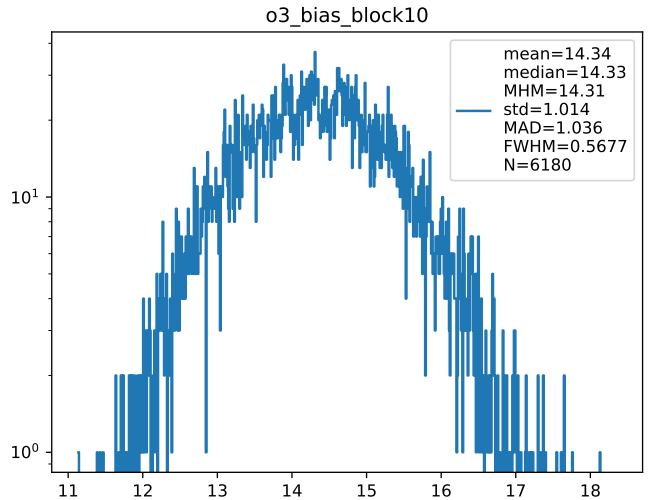
sanity check, repeat the calculations for 10x10 partitions

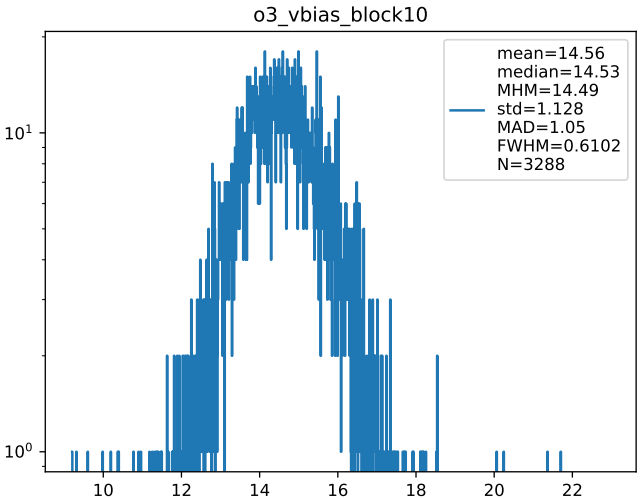
$$g\lambda = 2.55$$

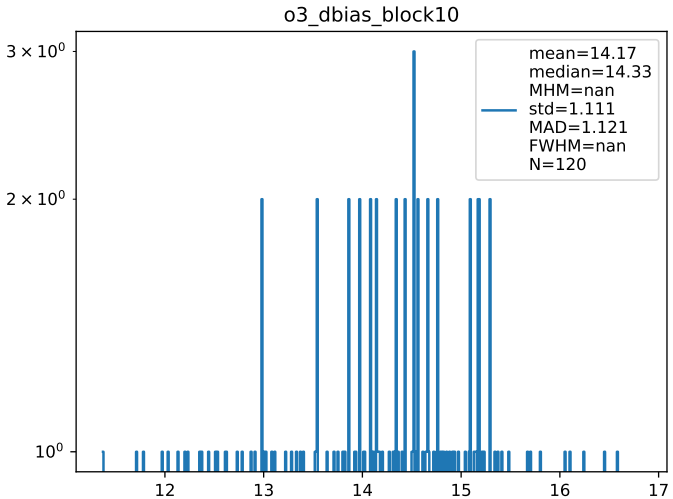
vfix std block 10x10 $E < 100$ADU (+3)

```
1   python Image.py analyse ImagePresentation/blockstd
        "/share/storage2/connie/data/runs/*/runID_*_03326_*_p*.fits.fz" --ohdu 3 --params-mode mean --
        remove-hits 100.0 3.0 --plot-block-spectrum --block-function "(lambda y: np.nan if y==0 else
        y)(np.nanstd(x))" --fix-vbias
```









$$\sigma = 14.3$$

$$g^2\lambda = 27$$

$$g = 10.41$$

$$\lambda = 0.24$$

there is a dependence on the size of the partition