Python Tutorial Series 2019

AstroPy Working with Images

Ph.D. Bruno C. Quint - bquint@gemini.edu

SUSD - GEMINI South, La Serena, Chile

Zen of Python

```
In [1]: import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Previously...

Basics I

What is Python?

What will you need?

Python as a terminal

Python as a script

Types of variables

Basics II

Code Flow

Functions

Classes

Methods

Attributes

Sub-Classes

Basics III

(Re)using code

Modules

Packages

Python2 to Python3

External packages

Virtual Environments

Conda and Anaconda

Data Analysis and Visualization

DataViz 1	DataViz 2
iPython / Jupyter	NumPy Tricks
NumPy	Image Display
MatPlotLib	Image Analysis
2D Plots	Image Manipulation

AstroPy

Coordinated Packages

astropy-core
astroquery
astropy-healpix
ccdproc
photutils
regions
reproject
specutils

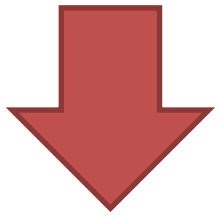
Affiliated Packages

APLpy
Astro-SCRAPPY
astroML
…
ginga
imexam
linetools
PyVO
… and 30 more!

https://www.astropy.org/affiliated/index.html

AstroPy

- Open Source
- Community driven
- Mature



- API Changes
- Limited Support

Coordinated Packages

astropy-core
astroquery
astropy-healpix
ccdproc
photutils
regions
reproject
specutils

Affiliated Packages

APLpy
Astro-SCRAPPY
astroML
…
ginga
imexam
linetools
PyVO
… and 30 more!

https://www.astropy.org/affiliated/index.html

AstroPy

- FITS Input/Output
- Image Display
- Image Analysis with `imexam`
- Photometry with `photutils`

AstroPy

reading data

```
from astropy.io import fits

data = fits.getdata("simple_image.fits")

print(data.__class__)
<class 'numpy.ndarray'>

print(data.shape)
(2149, 3245)
```

getdata

Fits to NumPy Array

Check __class__ attribute

Access shape

AstroPy

reading header

```
from astropy.io import fits

header = fits.getheader("simple_image.fits")

print(header.__class__)
<class 'astropy.io.fits.header.Header'>

print(header["INSTRUME"])
GMOS-S

print(list(header.keys()))
['SIMPLE', 'BITPIX', 'NAXIS', 'EXTEND', ...]
```

getheader
Reads metadata/header

Check `__class__` attribute

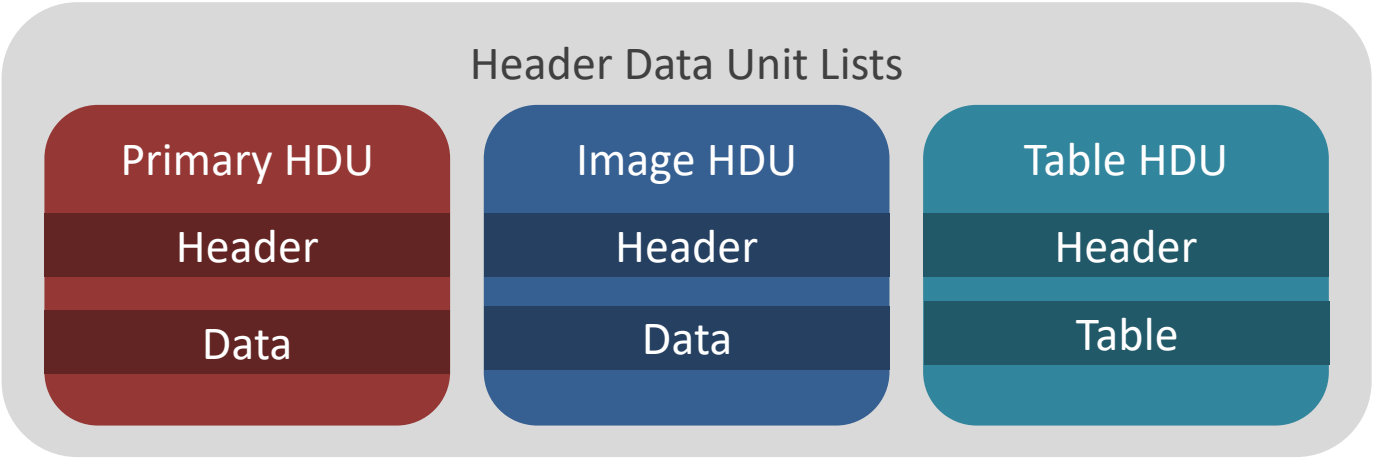
Access info as dictionary

List all keys (cards)

AstroPy

Header **D**ata **U**nit **L**ists

```
from astropy.io import fits

hdul = fits.open("simple_image.fits")

print(hdul.__class__)
<class 'astropy.io.fits.hdu.hdulist.HDUList'>
```



AstroPy

Header Data Unit Lists

```
from astropy.io import fits

hdul = fits.open("simple_image.fits")

hdul.info()
```

```
Filename: simple_image.fits
No.    Name      Ver    Type      Cards   Dimensions   Format
  0  PRIMARY       1 PrimaryHDU     252   ()
  1  SCI           1 ImageHDU       159   (3245, 2149)   float32
  2  VAR           1 ImageHDU       159   (3245, 2149)   float32
  3  DQ            1 ImageHDU       159   (3245, 2149)   int16
  4  OBJMASK       1 ImageHDU       159   (3245, 2149)   uint8
  5  OBJCAT        1 BinTableHDU    128   1756R x 43C   [J,...]
```

AstroPy

Header Data Unit Lists

```
from astropy.io import fits

hdul = fits.open("multi_extension_image.fits")

hdul.info()
```

```
Filename: multi_extension_image.fits
No.    Name      Ver    Type      Cards   Dimensions   Format
  0  PRIMARY       1 PrimaryHDU     252   ()
  1                -1 ImageHDU       324   (288, 2112)   int16
  2                -1 ImageHDU       288   (288, 2112)   int16
  3                -1 ImageHDU       252   (288, 2112)   int16
  4                -1 ImageHDU       252   (288, 2112)   int16
...
```

AstroPy

Show Images

```
from astropy.io import fits

hdul = fits.open("simple_image.fits")

sci = hdul[0]
sci = hdul["SCI"]
```

AstroPy

Show Images

```
from astropy.io import fits

hdul = fits.open("simple_image.fits")

sci = hdul[0]
sci = hdul["SCI"]
```

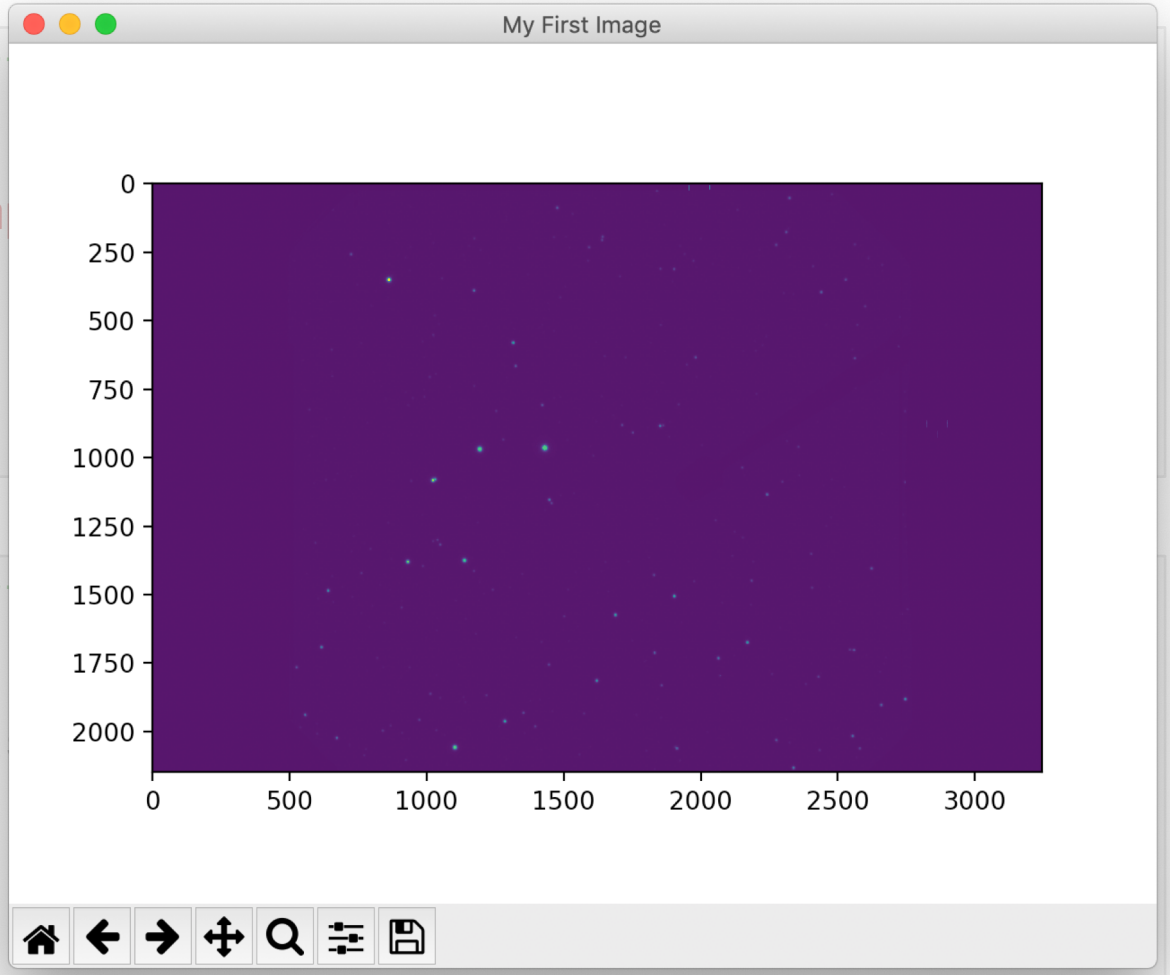
```
from matplotlib import pyplot as plt

fig, ax = plt.subplots(num="My First Image")
ax.imshow(sci.data)

plt.show()
```

AstroPy

Show Images

```
from astropy.io impor

hdul = fits.open("sim

sci = hdul[0]
sci = hdul["SCI"]
```

```
from matplotlib impor

fig, ax = plt.subplot
ax.imshow(sci.data)

plt.show()
```



AstroPy

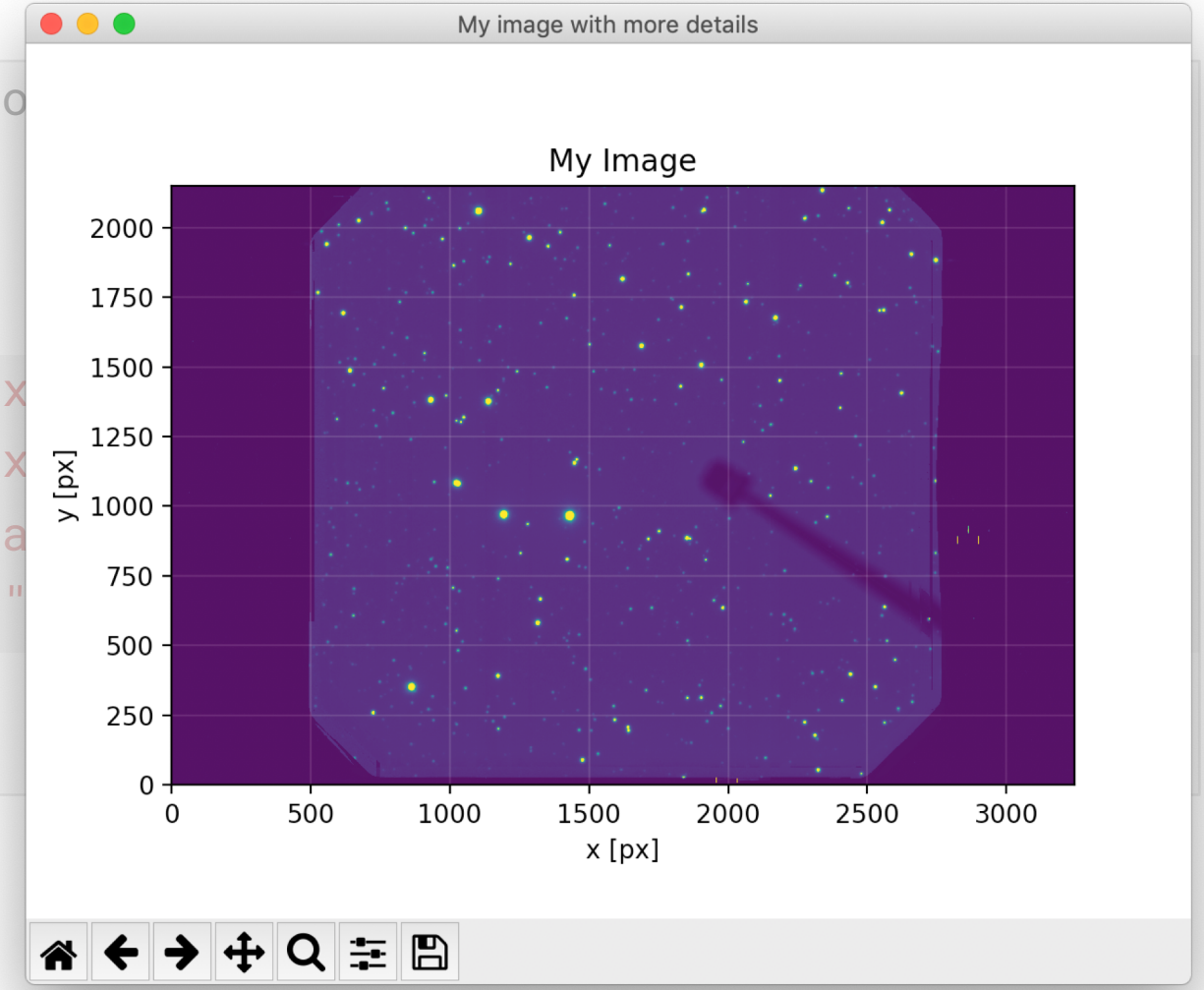
Show Images

```
fig, ax = plt.subplots(num="My image with more details")

ax.imshow(sci.data, origin="lower", vmin=1e1, vmax=1e4)

ax.set_xlabel("x [px]")
ax.set_ylabel("y [px]")
ax.set_title("My Image")
ax.grid("-", color="white", alpha=0.1)

plt.show()
```

AstroPy

Show Images

```
fig, ax = plt.subplo

ax.imshow(sci.data,

ax.set_xlabel("x [px
ax.set_ylabel("y [px
ax.set_title("My Ima
ax.grid("-", color="

plt.show()
```



AstroPy

Image Normalization

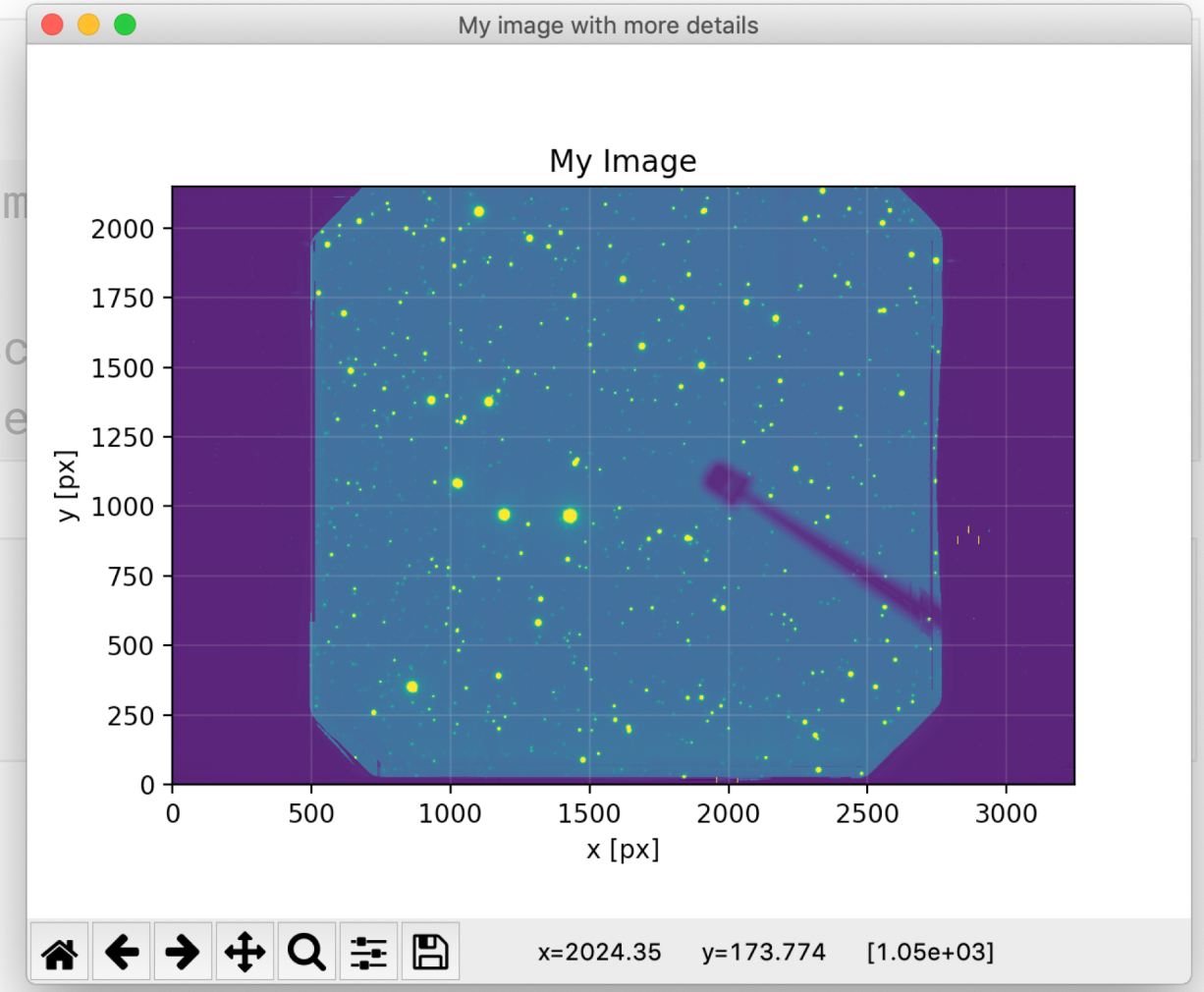
```
from astropy import visualization as vis

norm_factor = vis.ImageNormalize(
    sci.data,
    interval=vis.ZScaleInterval(),
    stretch=vis.LinearStretch())
```

```
...
ax.imshow(sci.data, origin="lower", norm=norm_factor)
...
```

AstroPy

Image Normalization

```
from astropy import

norm_factor = vis.Im
    sci.data,
    interval=vis.ZSc
    stretch=vis.Line

...
ax.imshow(sci.data,
...
```



AstroPy

Bad Data

```
from copy import copy

colormap = copy(plt.cm.viridis)
colormap.set_bad(color="Red")
```

```
dq = hdul["DQ"]
```

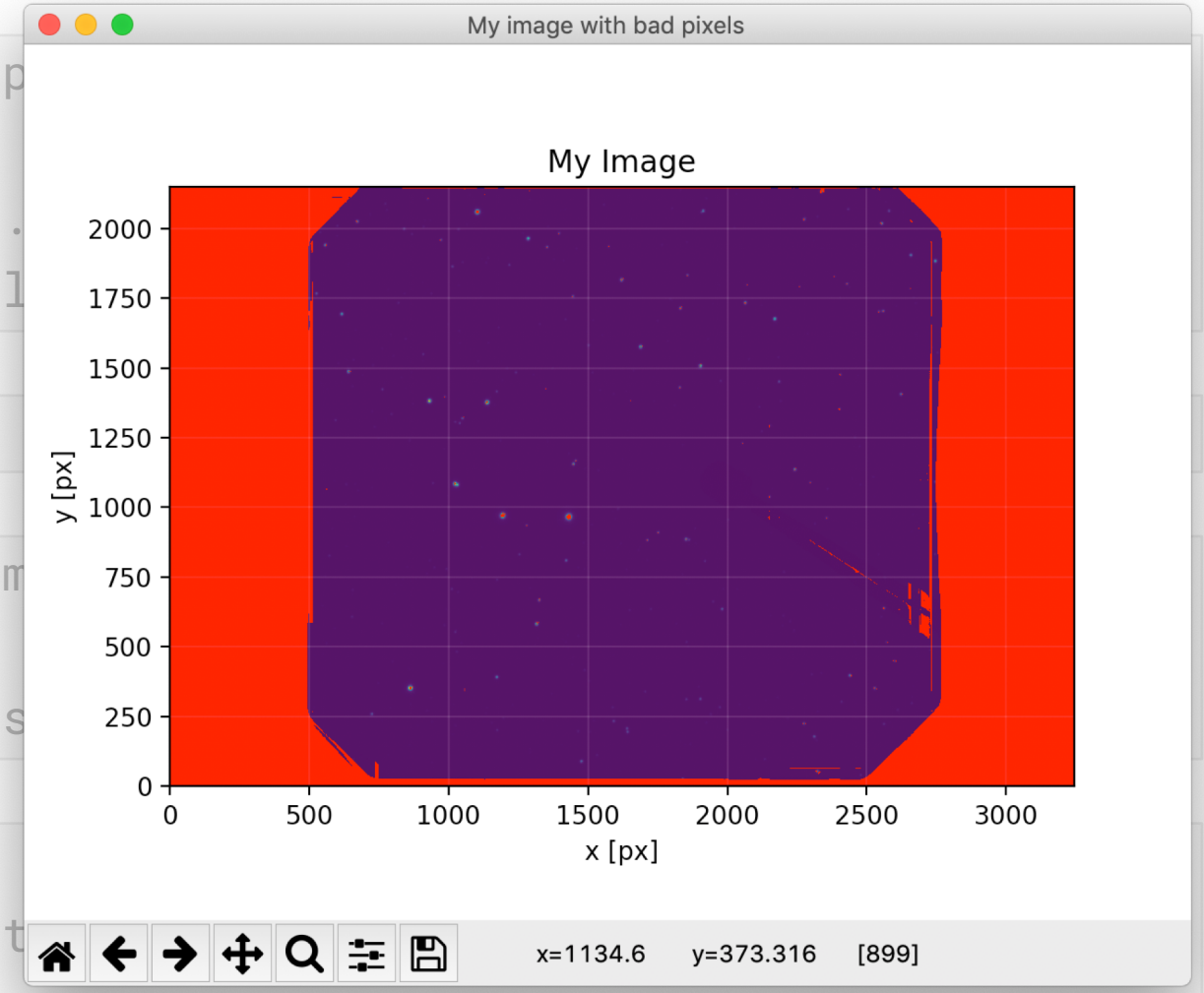
```
import numpy.ma as ma

masked_data = ma.masked_where(dq.data != 0, sci.data)
```

```
...
ax.imshow(masked_data, origin="lower", cmap=colormap)
...
```

AstroPy

Bad Data

```
from copy import cop

colormap = copy(plt.
colormap.set_bad(col

dq = hdul["DQ"]

import numpy.ma as m

masked_data = ma.mas

...
ax.imshow(masked_dat
...
```



AstroPy

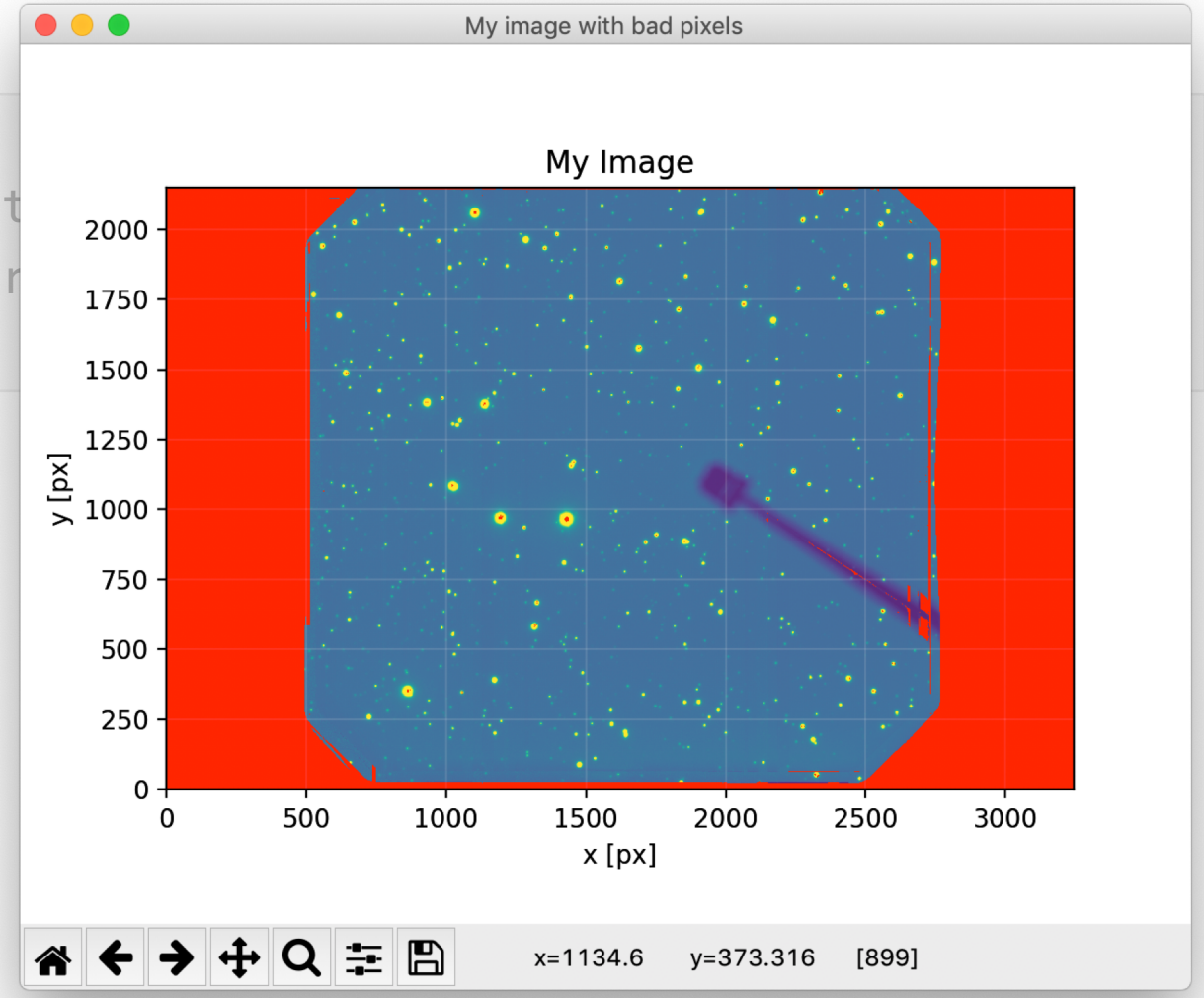
Bad Data and Normalization

```
...
ax.imshow(masked_data, origin="lower", cmap=colormap,
    vmin=norm_factor.vmin, vmax=norm_factor.vmax)
...
```

AstroPy

Bad Data and Normalization

```
...
ax.imshow(masked_dat
    vmin=norm_factor
...
```



AstroPy

World Coordinates

```
from astropy.wcs import WCS

wcs = WCS(sci.header)
```

```
plt.figure(num="Image with WCS")
ax = plt.subplot(projection=wcs)

ax.imshow(sci.data, origin="lower")
ax.coords.grid(True, color="white", ls="solid")
ax.coords[0].set_axislabel("Right Ascention (J2000)")
ax.coords[1].set_axislabel("Declination (J2000)")

plt.show()
```

imexam

Issues and Limitations

- Under heavy development
- DS9 v7.4 - MacOS Mojave

imexam

Issues and Limitations

- Under heavy development
- DS9 v7.4 - MacOS Mojave

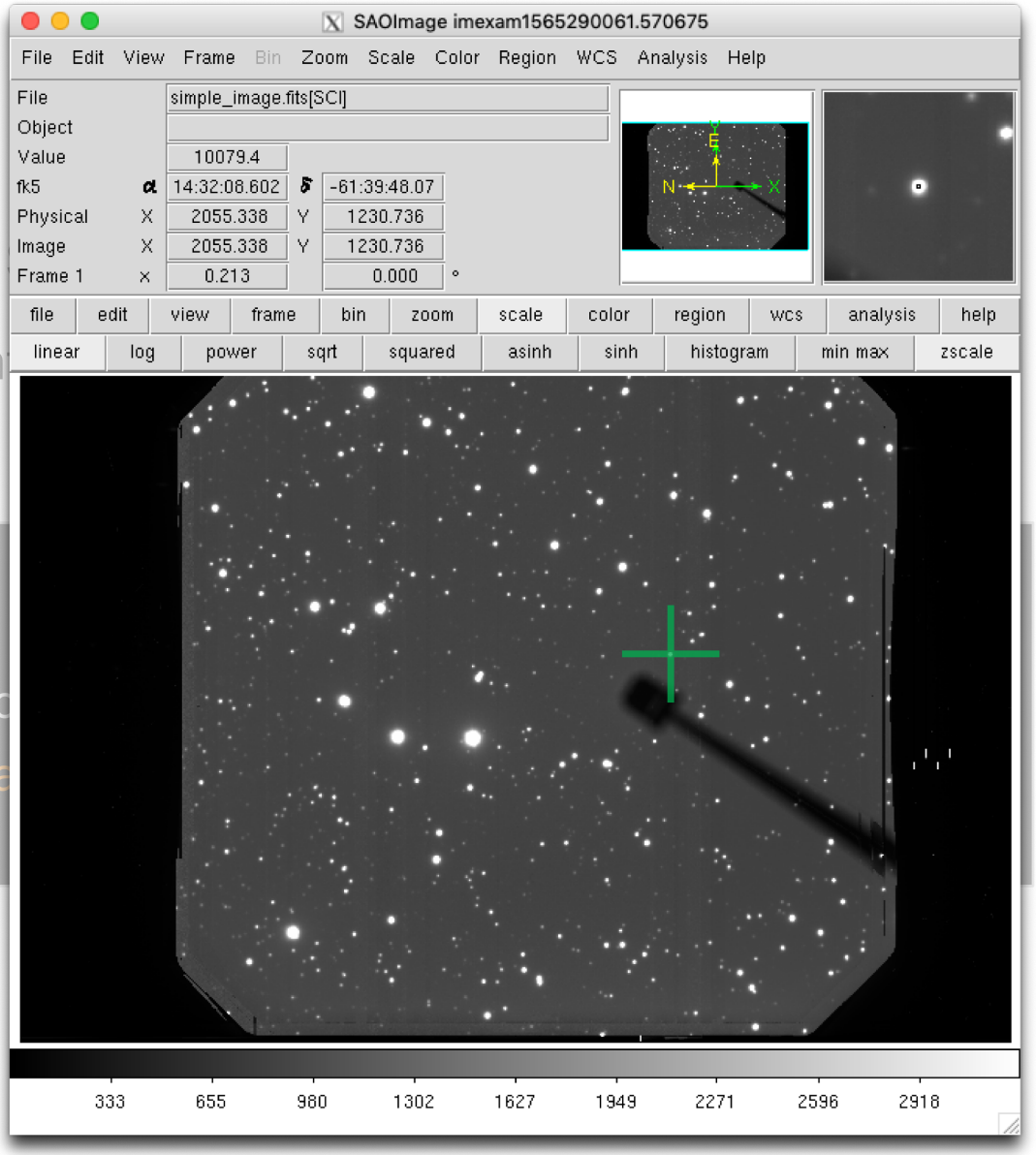
```
In [1]: import imexam
In [2]:
In [2]: w = imexam.connect()
In [3]: w.load_fits("data/simple_image.fits")
In [4]: w.imexam()
```

imexam

Issues and Limitation

- Under heavy developmen
- DS9 v7.4

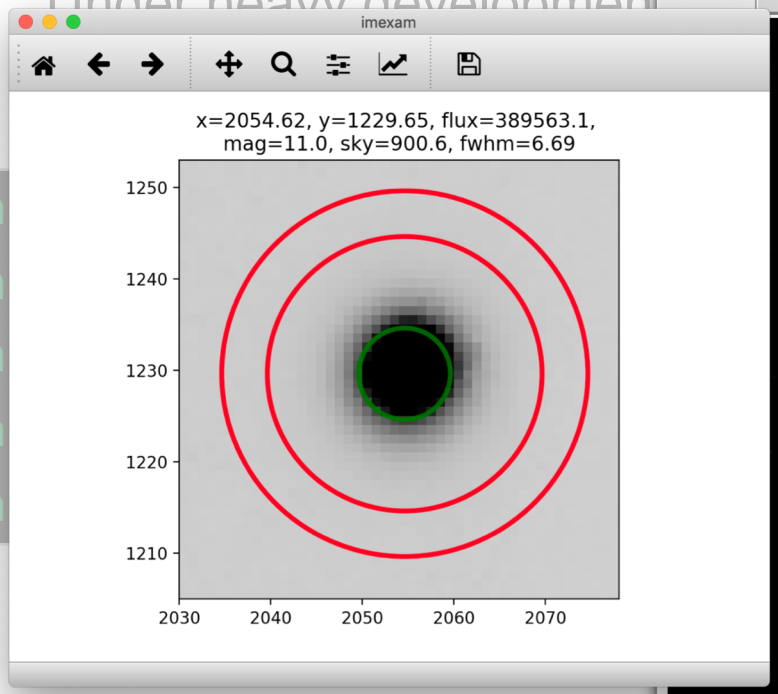
```
In [1]: import imexam
In [2]:
In [2]: w = imexam.connec
In [3]: w.load_fits("data
In [4]: w.imexam()
```

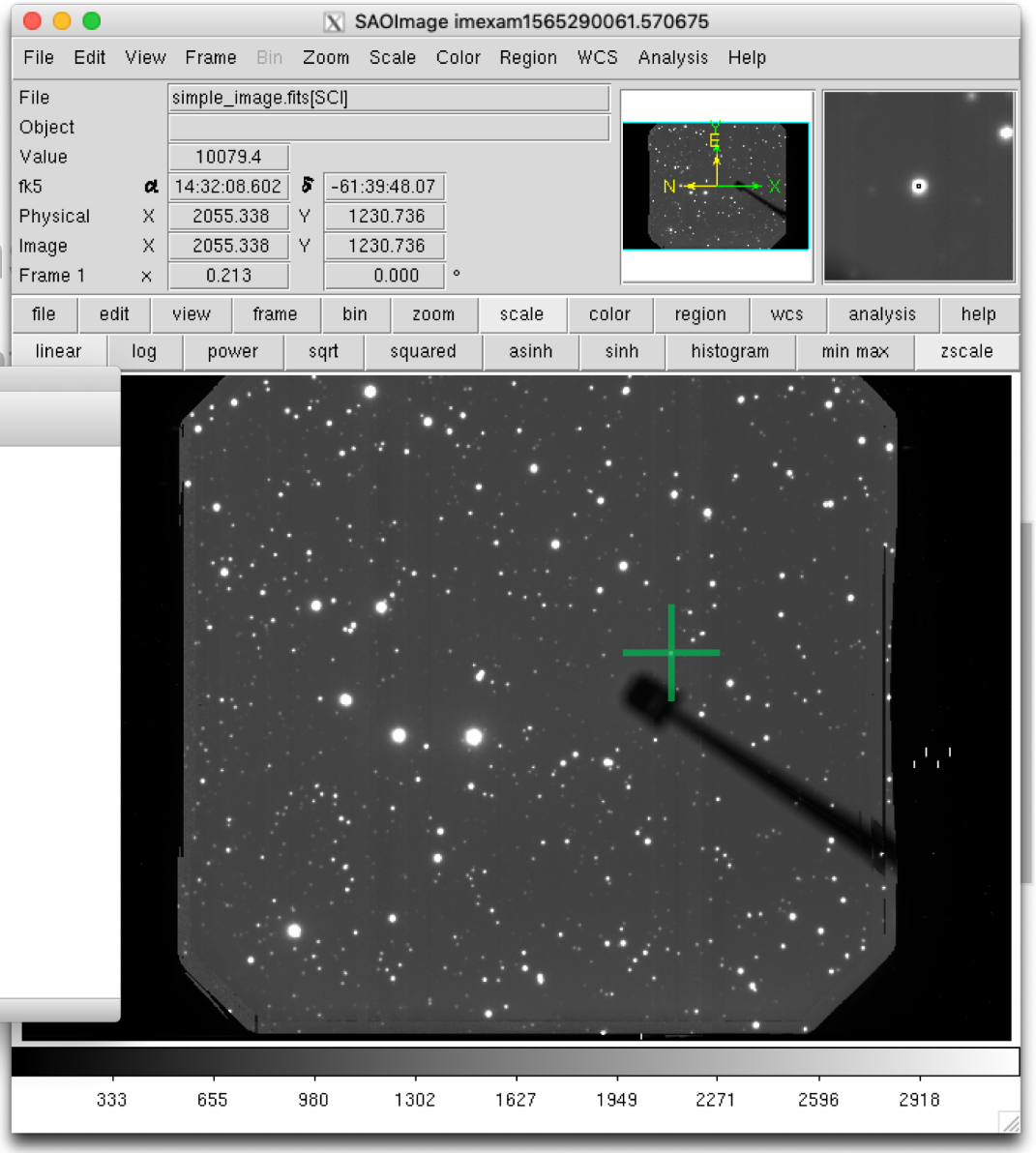


imexam

Issues and Limitation

- Under heavy development
-





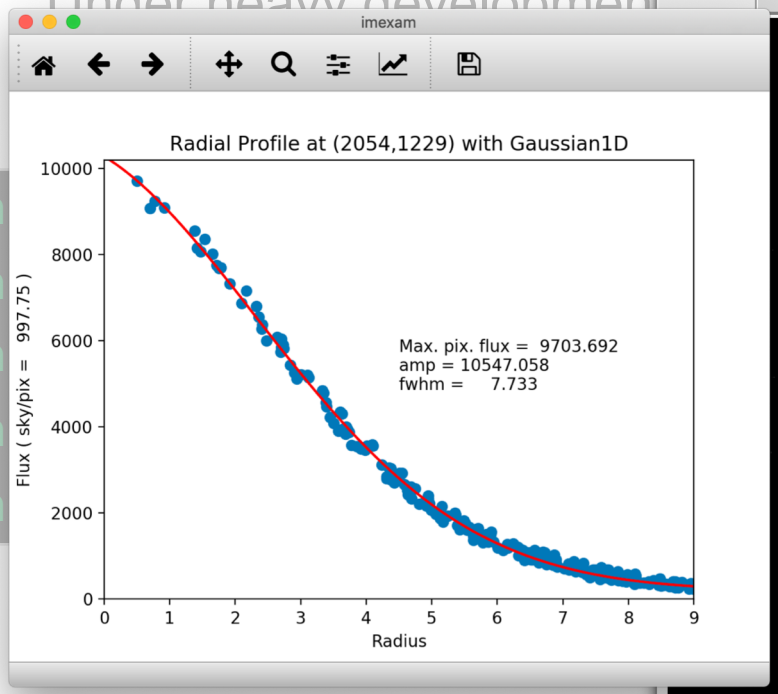
Press **A**

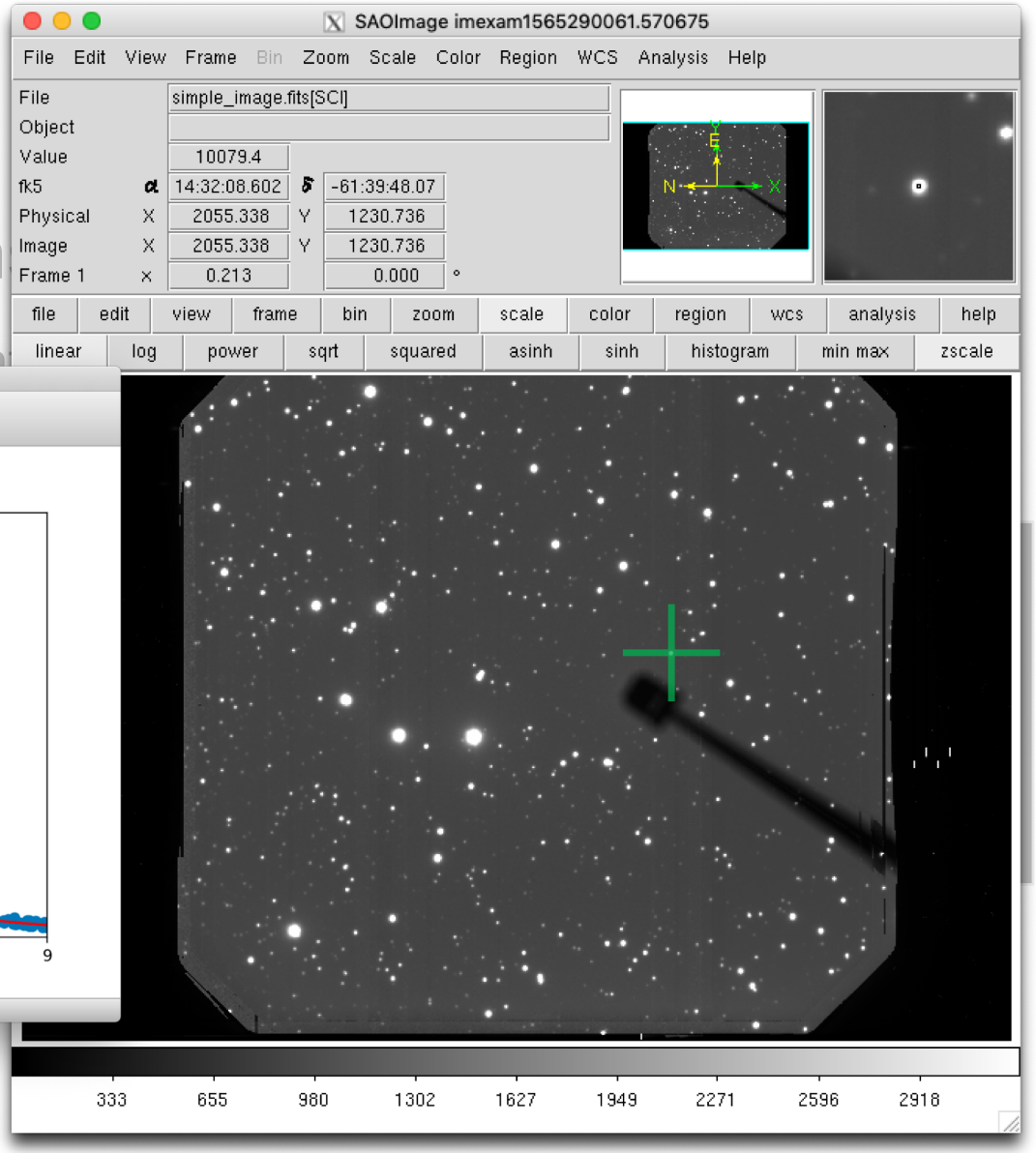
```
Apperture sum
```

imexam

Issues and Limitation

- Under heavy development
-





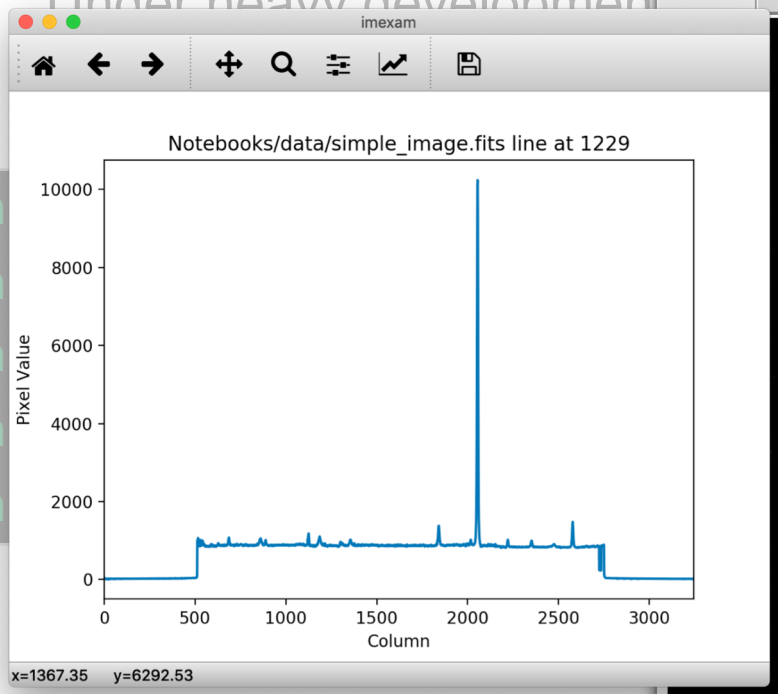
Press **R**

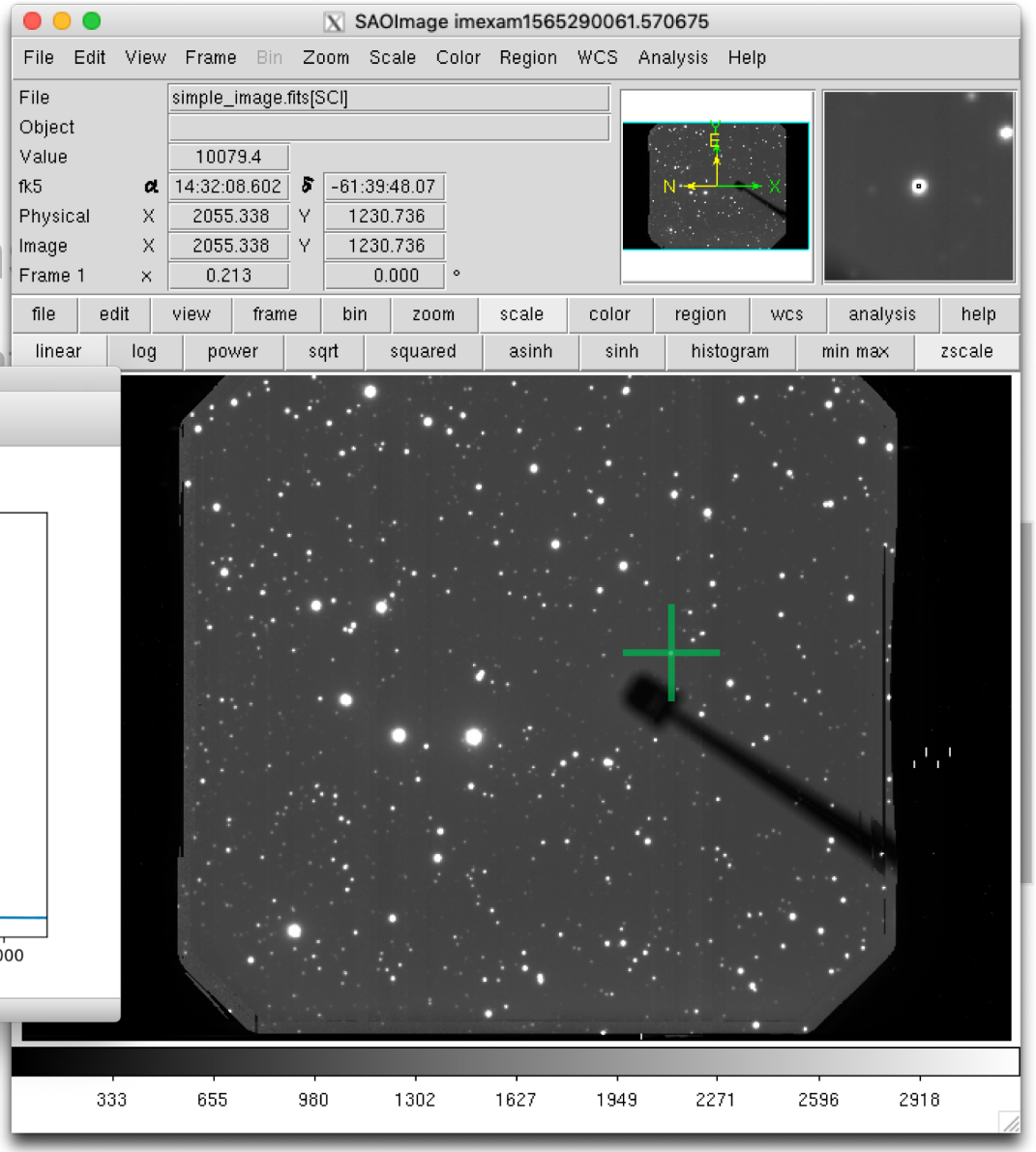
`Radial Profile`

imexam

Issues and Limitation

- Under heavy development
-





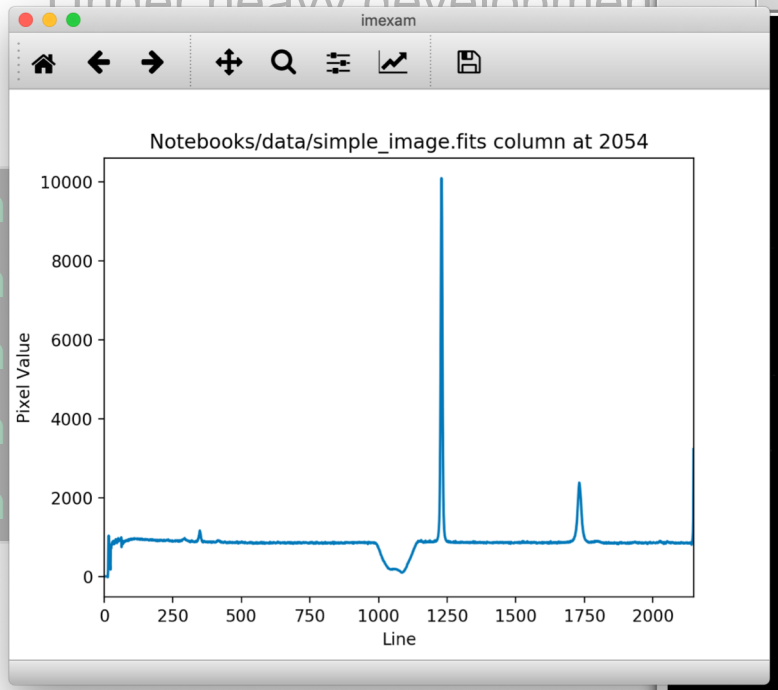
Press L

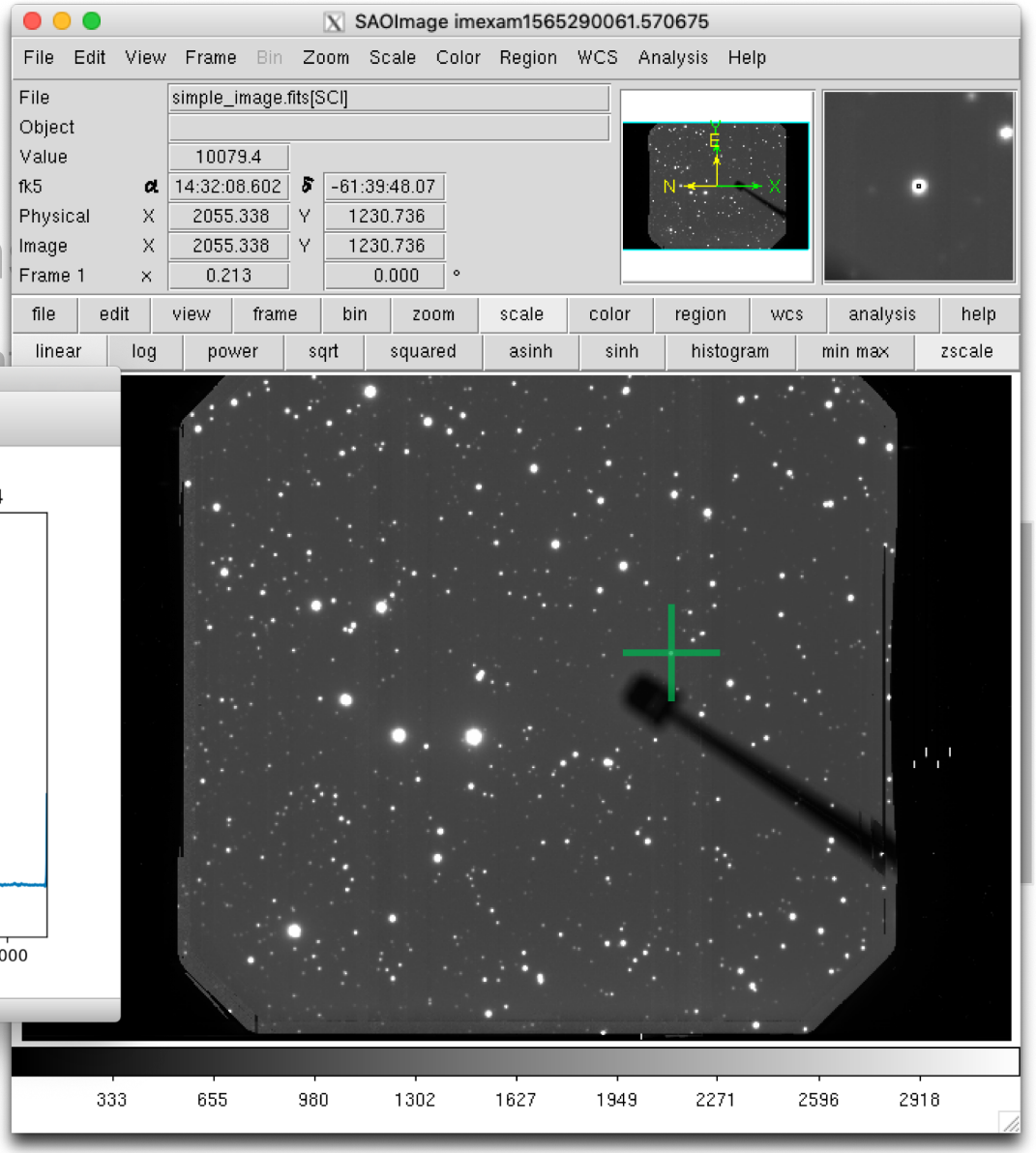
```
Plot Line (row)
```

imexam

Issues and Limitation

- Under heavy development
-





Press `C`

`Plot Column`

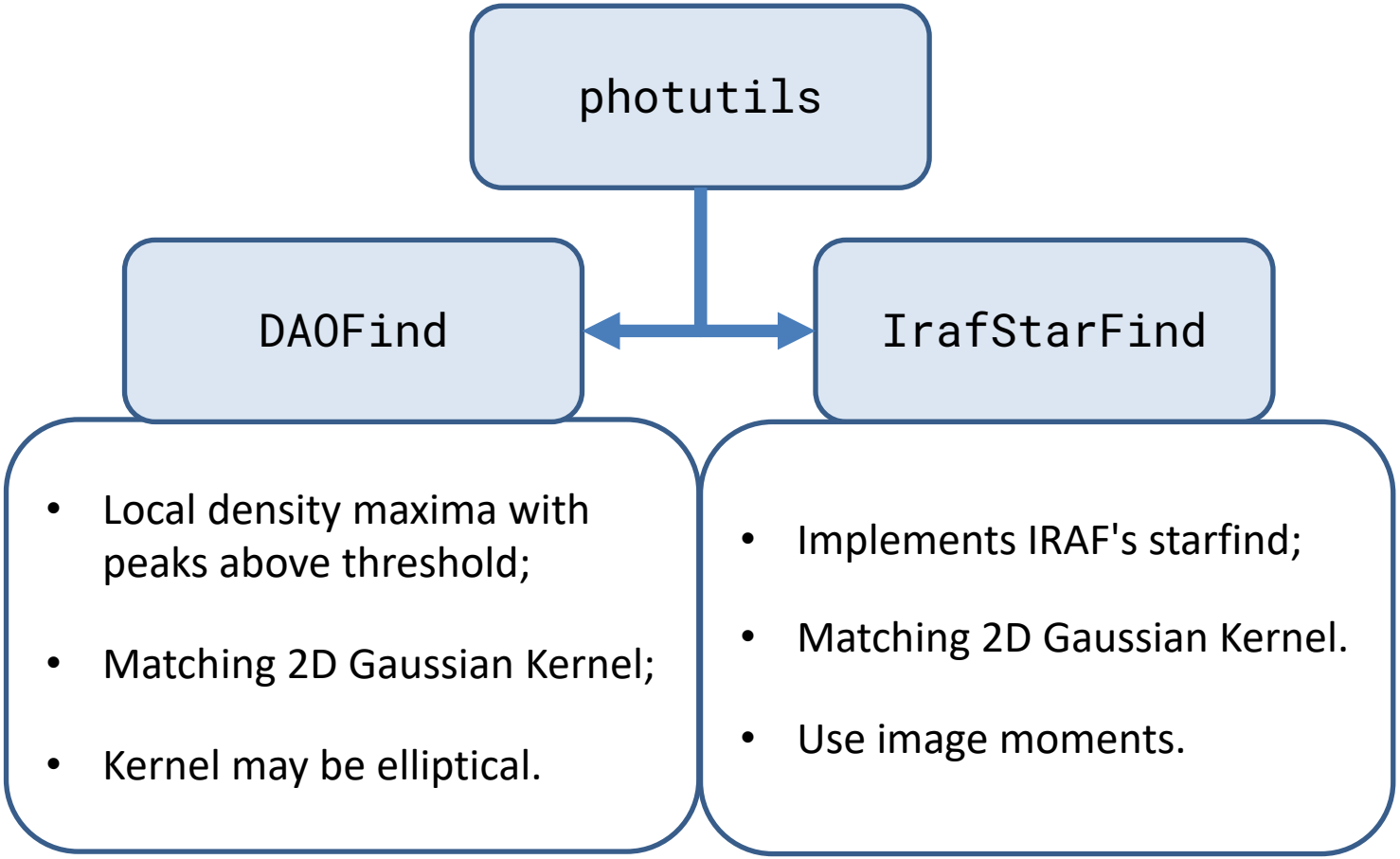
imexam

Issues and Limitations

- Under heavy development
- DS9 v7.4

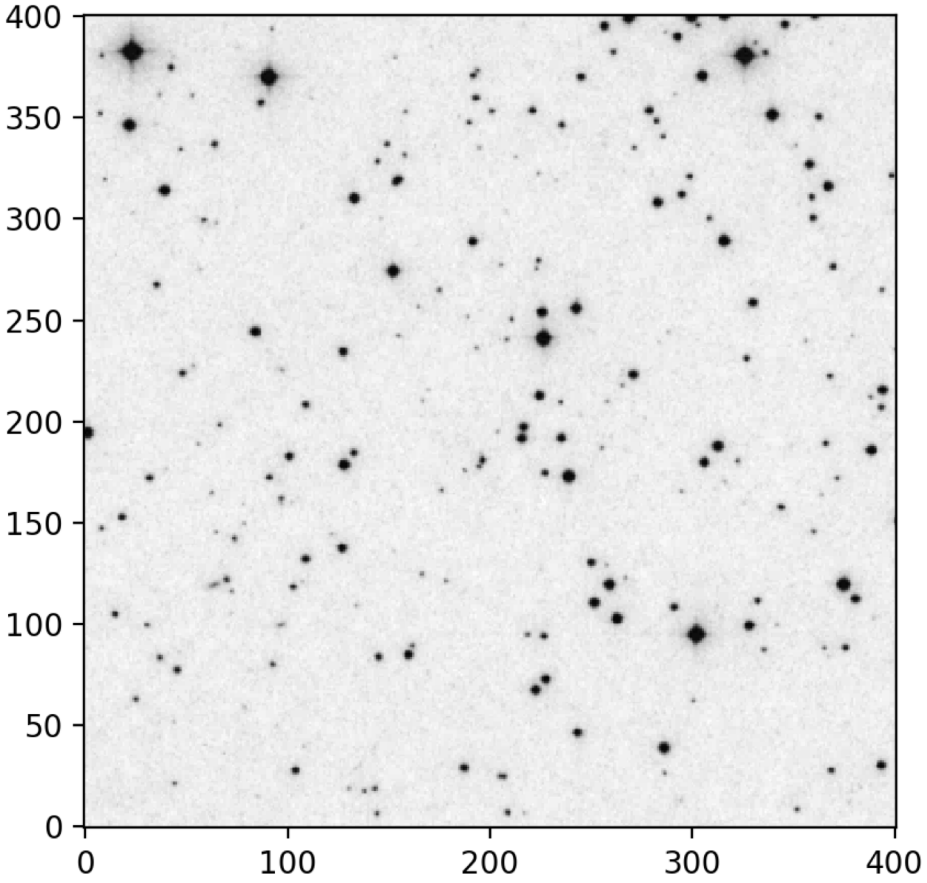
```
In [4]: v.imexam()

Press 'q' to quit

2 Make the next plot in a new window
a Aperture sum, with radius region_size
b Return the 2D gauss fit center of the object
…
```

photutils


photutils
DAOFind
IrafStarFind
• Local density maxima with peaks above threshold;
• Matching 2D Gaussian Kernel;
• Kernel may be elliptical.
• Implements IRAF's starfind;
• Matching 2D Gaussian Kernel.
• Use image moments.

photutils

```
from photutils import da

hdu = datasets.load_star
data = hdu.data[0:401, 0
```



photutils

Getting Stats

```
from photutils import datasets

hdu = datasets.load_star_image()
data = hdu.data[0:401, 0:401]
```

```
from astropy.stats import sigma_clipped_stats

mean, median, std = sigma_clipped_stats(data, sigma=3.0)
print(mean, median, std)
```

```
3668.09661145823 3649.0 204.41388592022315
```

photutils

Detect sources with DAOFind

```
from photutils import DAOStarFinder

daofind = DAOStarFinder(fwhm=3.0, threshold=5.*std)
sources = daofind(data - median)
```

```
# for consistent table output
for col in sources.colnames:
    sources[col].info.format = '%.8g'

print(sources)
```

```
id  xcentroid ycentroid sharpness  ... sky peak flux      mag
--- --------- --------- ---------- ... --- ---- --------- ------------
  1 144.24757 6.3797904 0.58156257 ...   0 6903 5.6976747   -1.8892441
  2 208.66907 6.8205805 0.48348966 ...   0 7896 6.7186388   -2.0682032
...       ...       ...        ... ... ...  ...       ...          ...
286 360.44533 399.52381 0.37315624 ...   0 8079 6.9203438   -2.1003192
Length = 286 rows
```

photutils

Detect sources with DAOFind

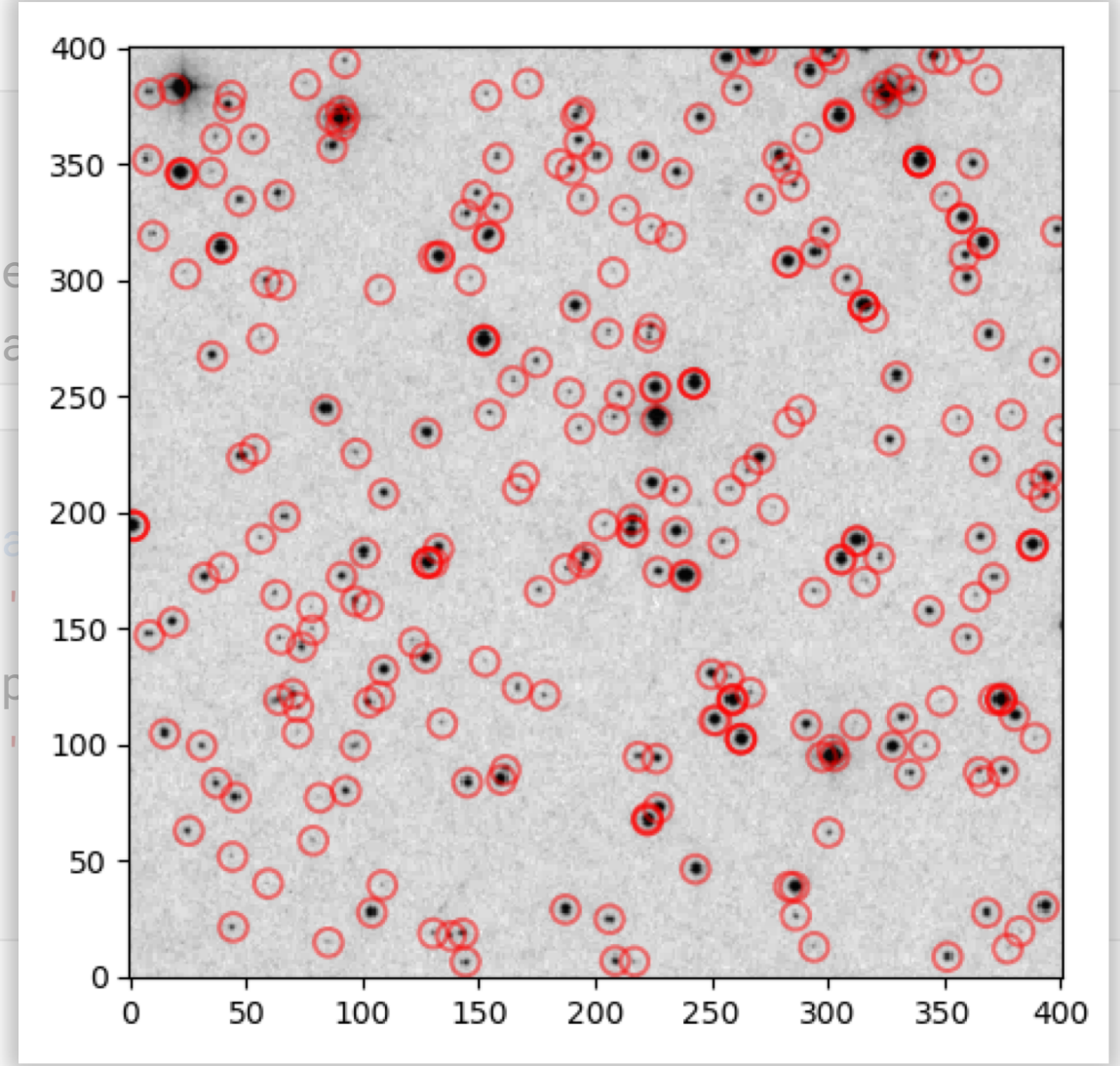
```
from photutils import DAOStarFinder

daofind = DAOStarFinder(fwhm=3.0, threshold=5.*std)
sources = daofind(data - median)
```

```
from photutils import CircularAperture
... # Create fig, ax and display the image first
positions = (sources['xcentroid'], sources['ycentroid'])
apertures = CircularAperture(positions, r=6.)
apertures.plot(color='red', lw=1.5, alpha=0.5, ax=ax)

plt.show()
```

photutils

```
from photutils import

daofind = DAOStarFinde
sources = daofind(data
```

```
from photutils import
... # Create fig, ax a
positions = (sources['
apertures = CircularAp
apertures.plot(color='

plt.show()
```



photutils

Detect sources with IRAFStarFinder

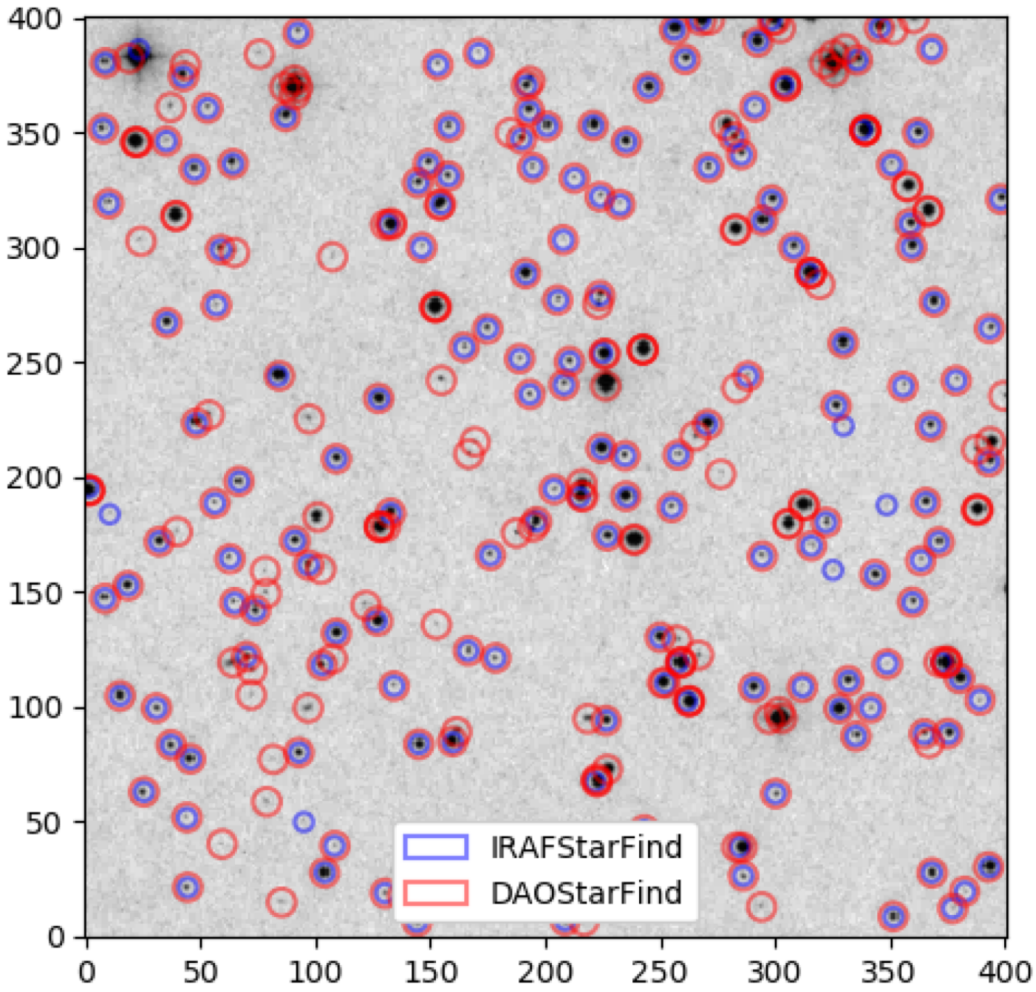
```
from photutils import IRAFStarFinder

iraffind = IRAFStarFinder (fwhm=3.0, threshold=5.*std)
iraf_sources = iraffind(data - median)
```

```
... # Create fig, ax and display the image first
iraf_pos = (sources['xcentroid'], sources['ycentroid'])
iraf_ap = CircularAperture(iraf_pos, r=4.)
iraf_ap.plot(color='blue', lw=1.5, alpha=0.5, ax=ax)

plt.show()
```

photutils

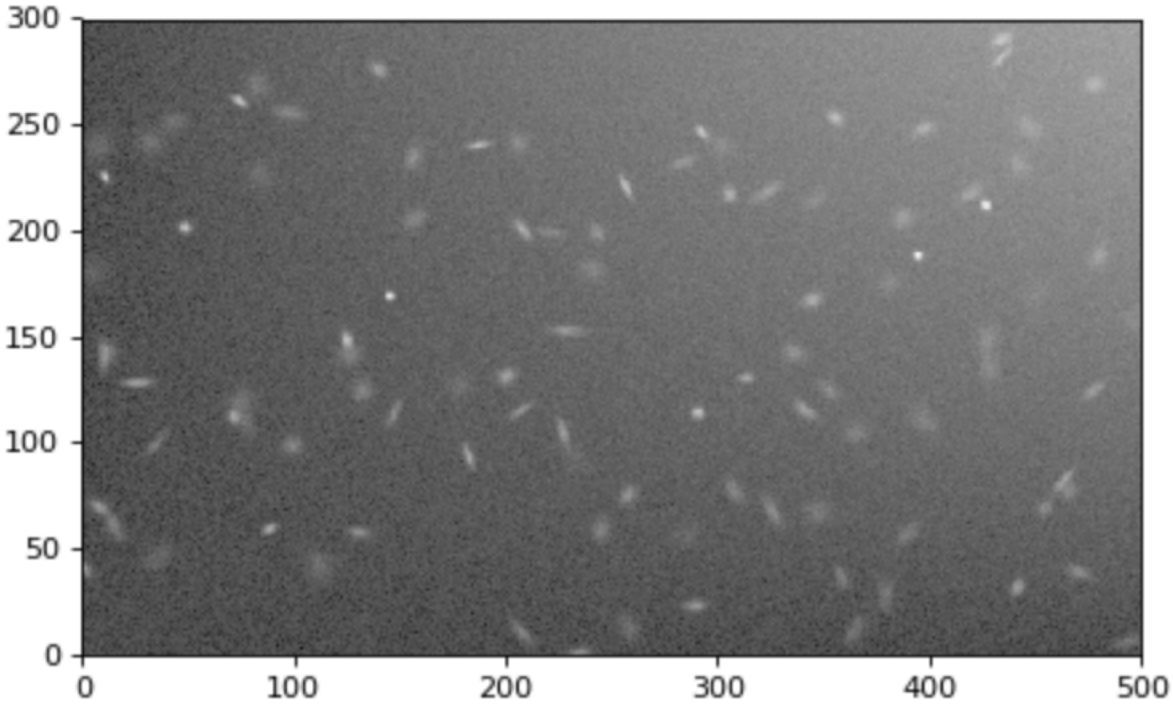
Detect sources with IRAFStarFinder

```
from photutils import IR

iraffind = IRAFStarFinde
iraf_sources = iraffind(

... # Create fig, ax and
iraf_pos = (sources['xce
iraf_ap = CircularApertu
iraf_ap.plot(color='blue

plt.show()
```



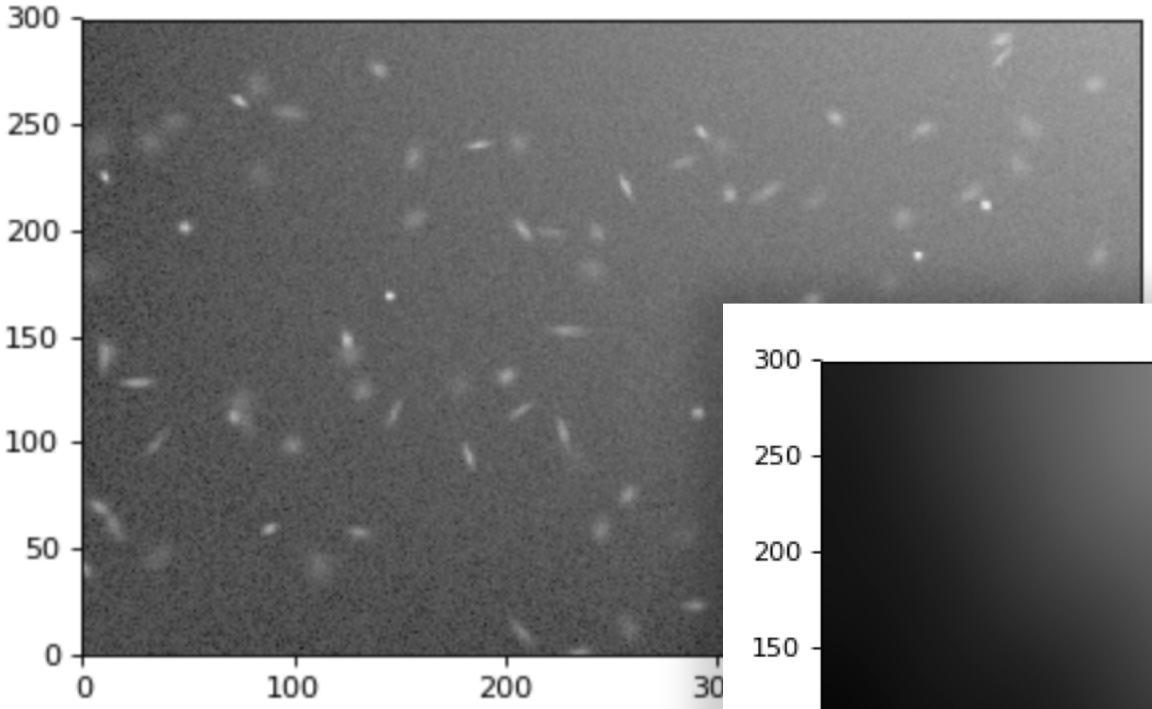
photutils

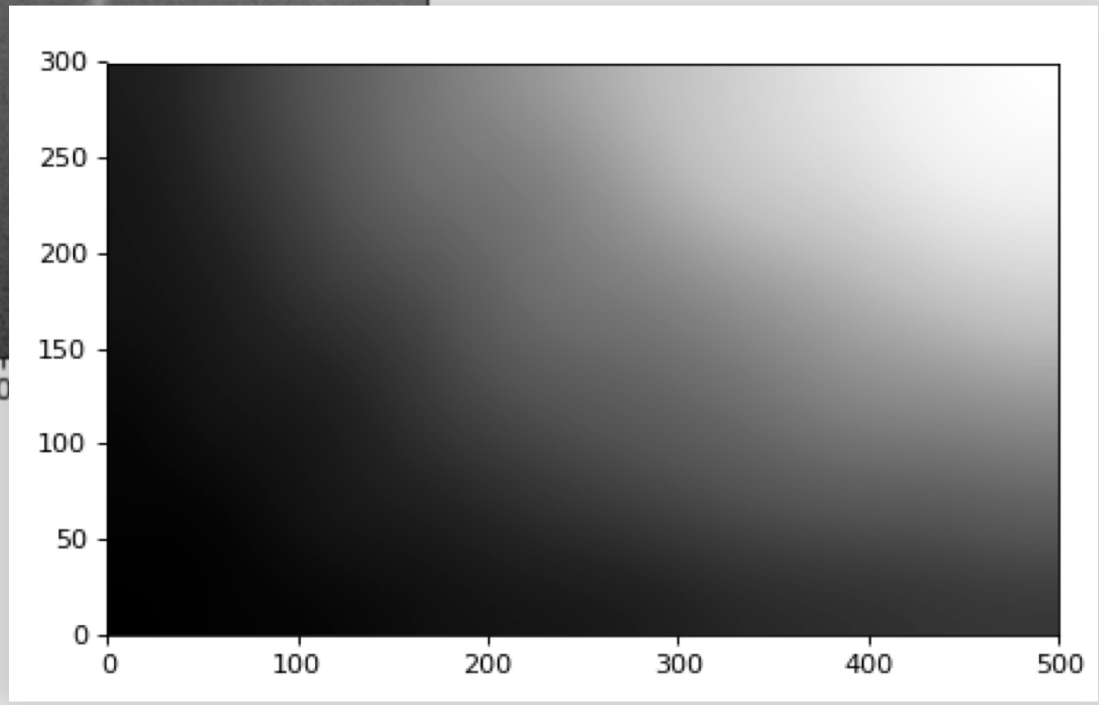
2D Background Subtraction



photutils

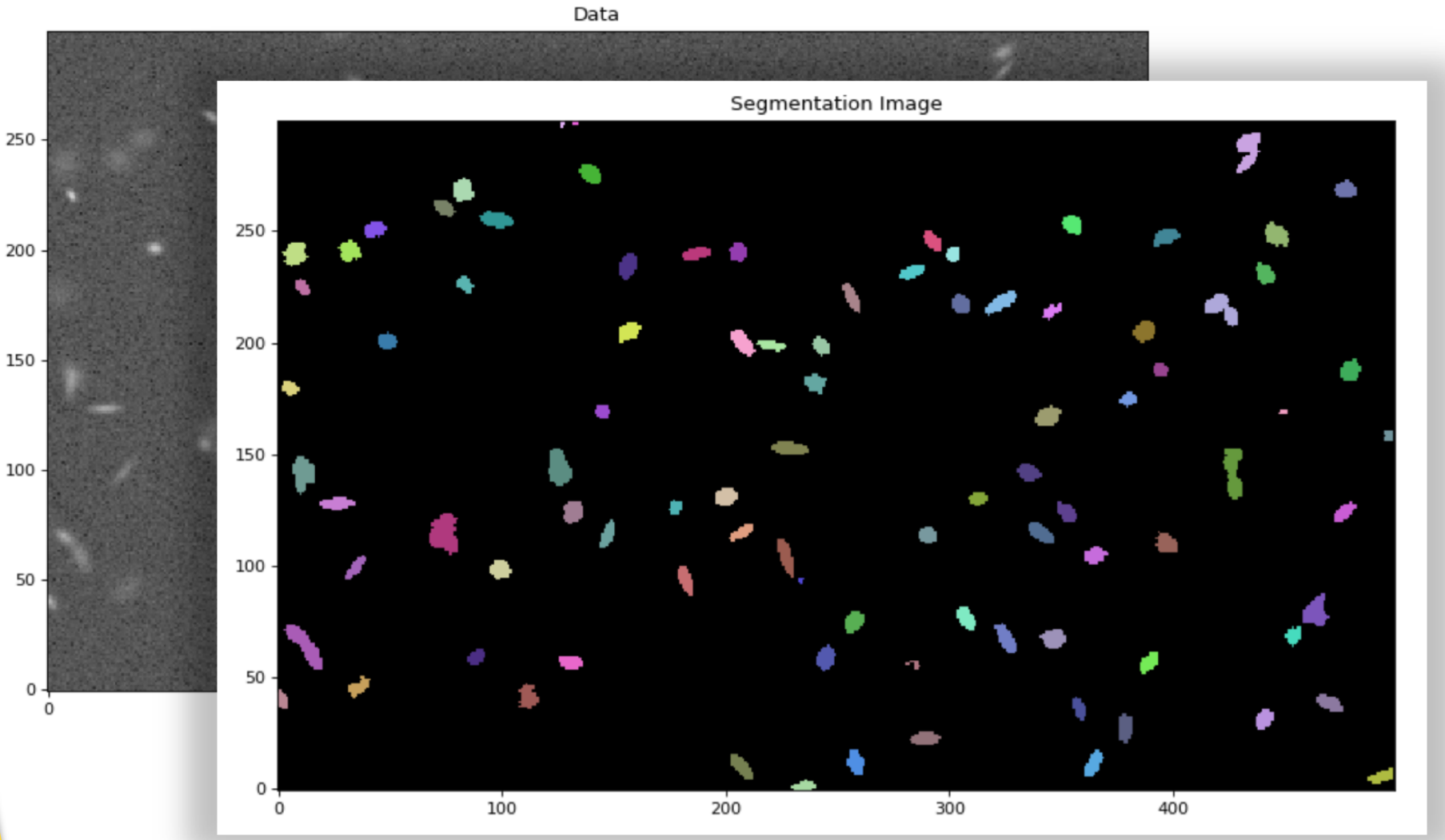
2D Background Subtraction





photutils

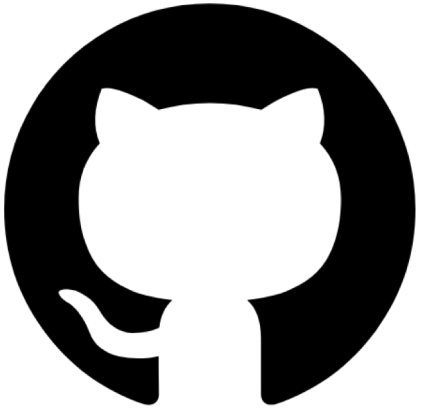
Image Segmentation



Getting Help

Official Documentation
https://www.astropy.org/

Channels on Slack
astropy.slack.com



AstroPy on GitHub
https://github.com/astropy/astropy/issues

Questions?