Final-Project

March 18, 2024

[2]:

```python
import numpy as np
from astropy.io import fits
import sep
import matplotlib.pyplot as plt
from matplotlib import rcParams
%matplotlib inline
rcParams['figure.figsize'] = [10., 8.]
```

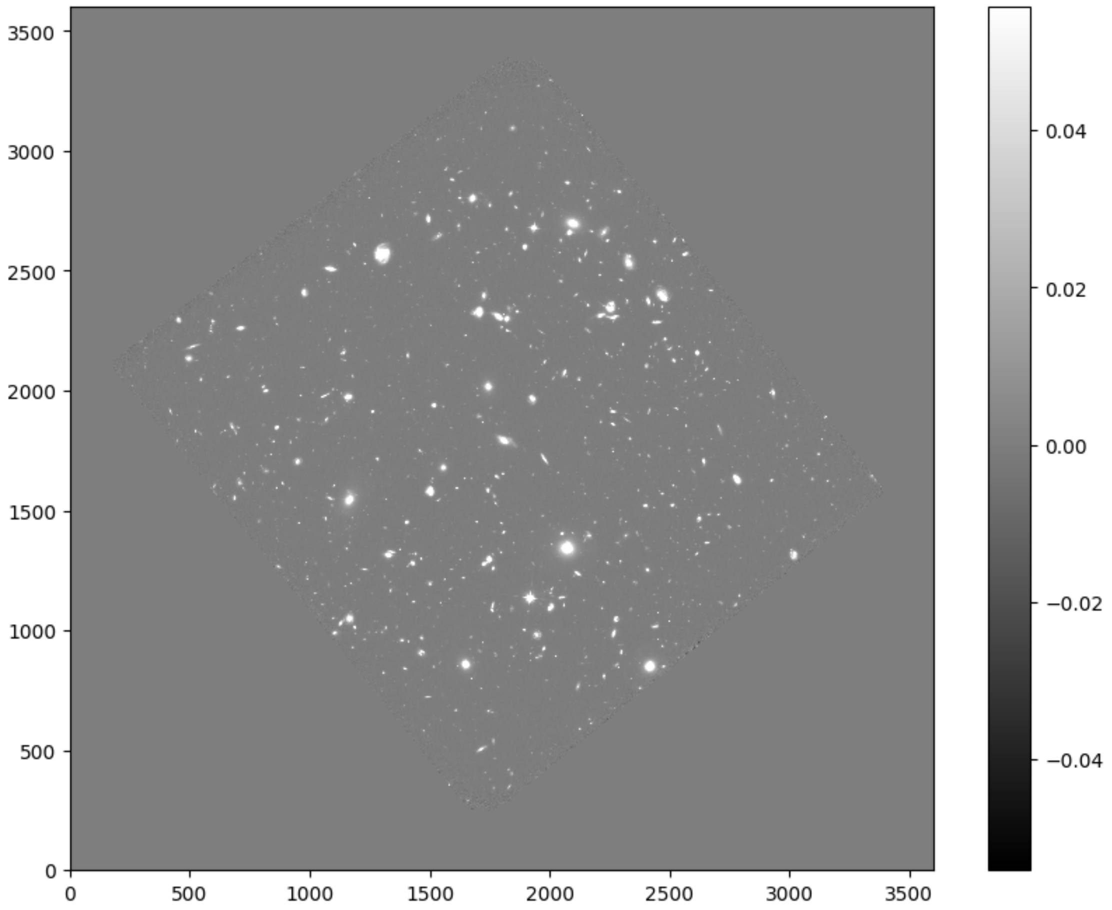
[3]:

```python
# read image into standard 2-d numpy array
image = fits.open("f105.fits")
image.info()
data = image[0].data
print(type(data))
```

```
Filename: f105.fits
No.    Name      Ver    Type      Cards   Dimensions   Format
  0  PRIMARY       1 PrimaryHDU     359   (3600, 3600)   float32
<class 'numpy.ndarray'>
```

[4]:

```python
# show the image
m, s = np.mean(data), np.std(data)
plt.imshow(data, interpolation='nearest', cmap='gray', vmin=m-s, vmax=m+s,␣
 ↪origin='lower')
plt.colorbar();
plt.savefig('image.png',bbox_inches='tight',dpi=600)
```



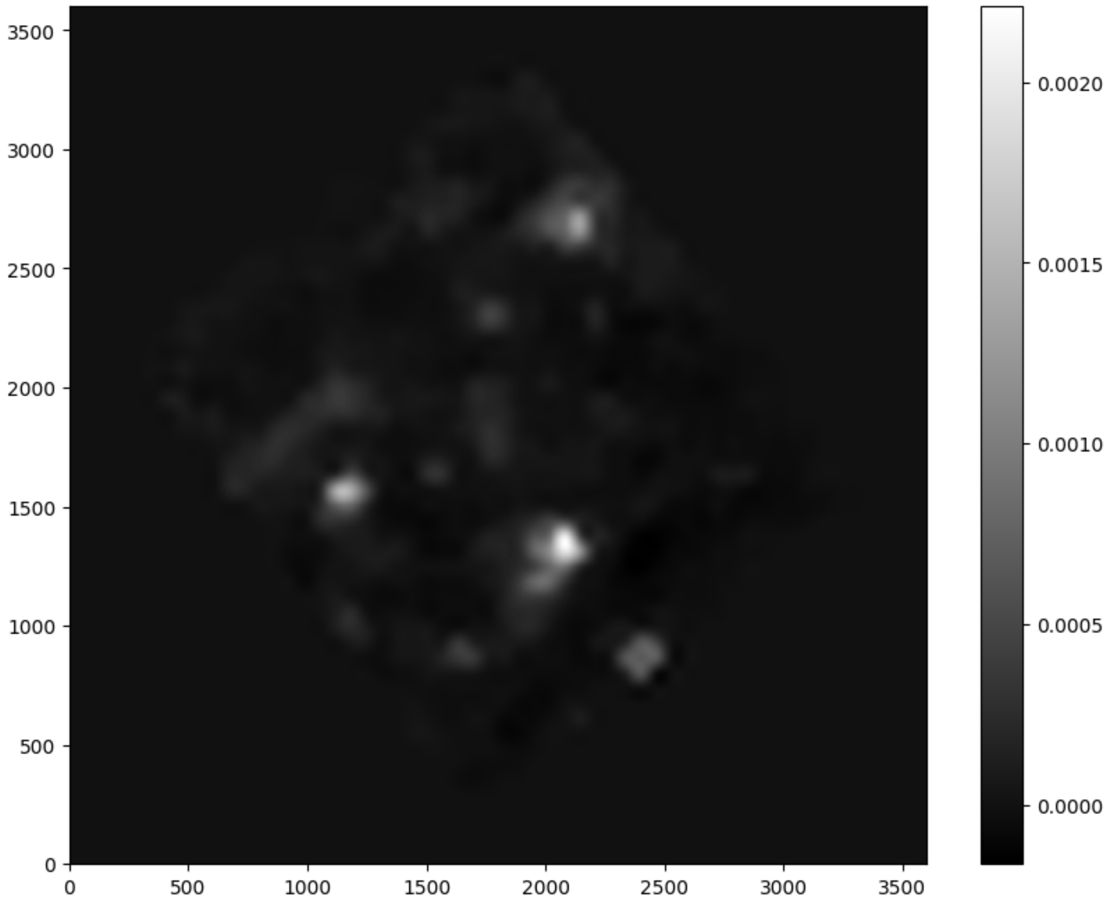
```
[5]: # measure a spatially varying background on the image
     data = data.byteswap().newbyteorder()
     bkg = sep.Background(data)
```

```
[6]: # get a "global" mean and noise of the image background:
     print(bkg.globalback)
     print(bkg.globalrms)
```

```
0.0
0.0005398219218477607
```

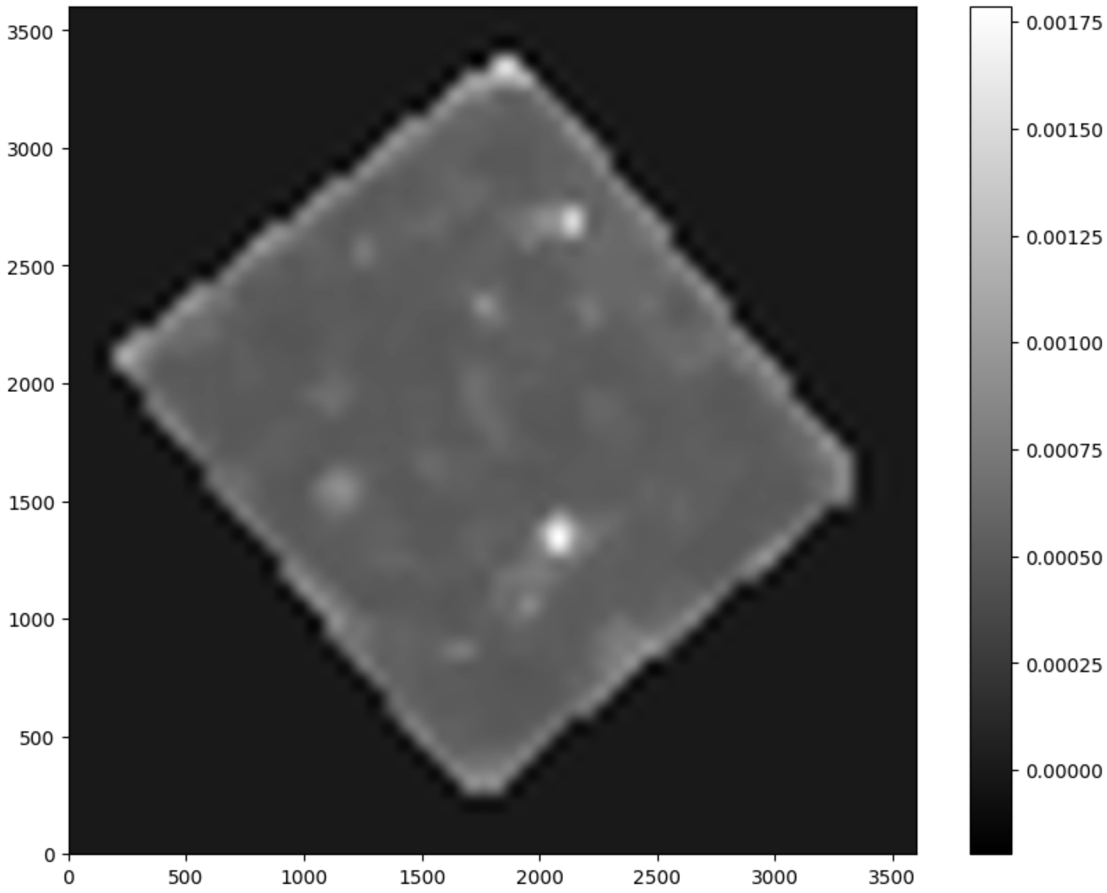
```
[7]: # evaluate background as 2-d array, same size as original image
     bkg_image = bkg.back()
     # bkg_image = np.array(bkg) # equivalent to above
```

```
[8]: # show the background
     plt.imshow(bkg_image, interpolation='nearest', cmap='gray', origin='lower')
     plt.colorbar();
     plt.savefig('background.png',bbox_inches='tight',dpi=600)
```



```
[9]: # evaluate the background noise as 2-d array, same size as original image
     bkg_rms = bkg.rms()
```

```
[10]: # show the background noise
      plt.imshow(bkg_rms, interpolation='nearest', cmap='gray', origin='lower')
      plt.colorbar();
      plt.savefig('backgroundNoise.png',bbox_inches='tight',dpi=600)
```



[11]:

```
# subtract the background
data_sub = data - bkg
```

[12]:

```
objects = sep.extract(data_sub, 1.5, err=bkg.globalrms)
```

[13]:

```
# how many objects were detected
len(objects)
```

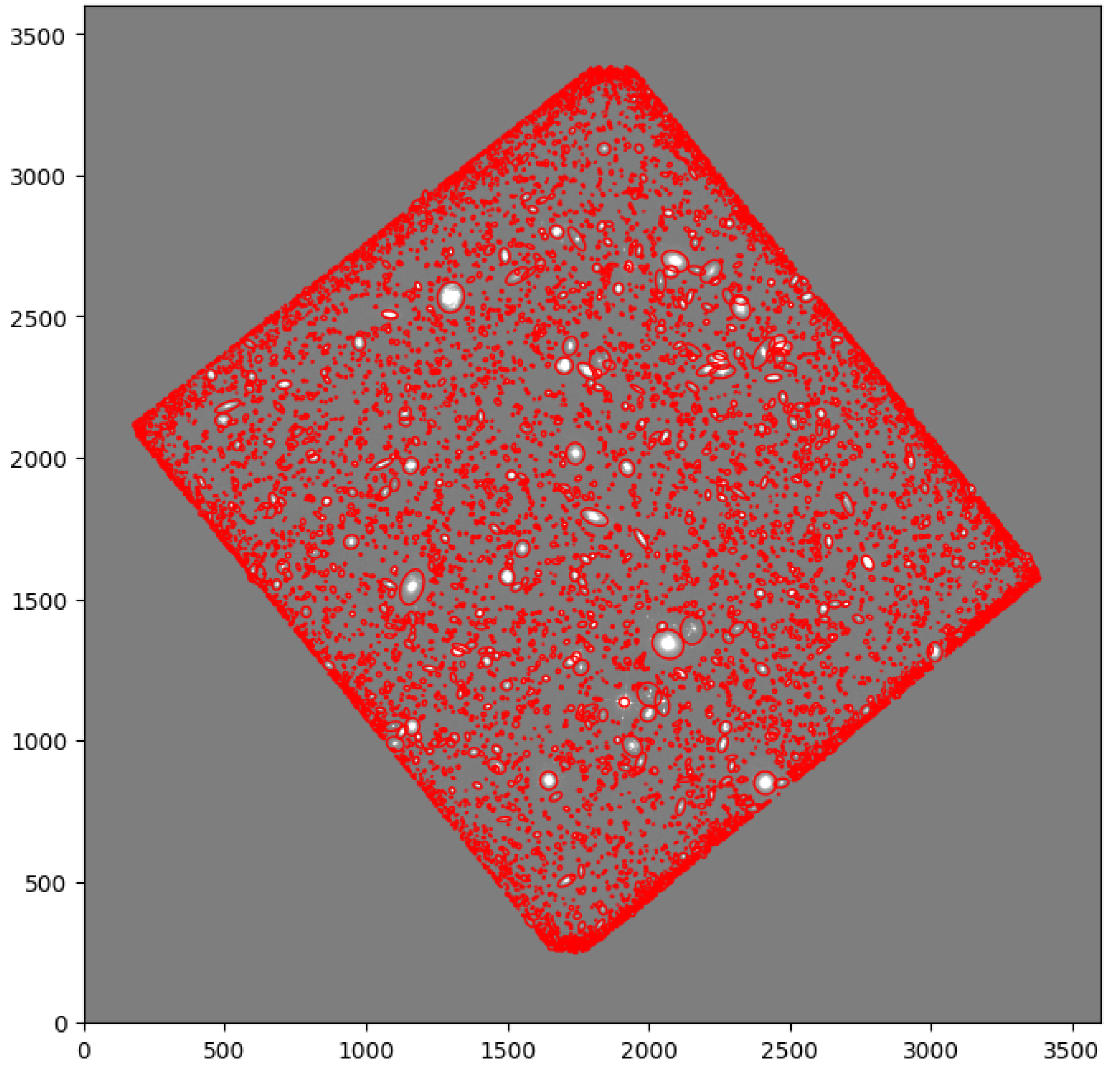
[13]:

```
8640
```

[14]:

```
from matplotlib.patches import Ellipse

# plot background-subtracted image
fig, ax = plt.subplots()
m, s = np.mean(data_sub), np.std(data_sub)
im = ax.imshow(data_sub, interpolation='nearest', cmap='gray',
               vmin=m-s, vmax=m+s, origin='lower')

# plot an ellipse for each object
```

```
for i in range(len(objects)):
    e = Ellipse(xy=(objects['x'][i], objects['y'][i]),
                width=6*objects['a'][i],
                height=6*objects['b'][i],
                angle=objects['theta'][i] * 180. / np.pi)
    e.set_facecolor('none')
    e.set_edgecolor('red')
    ax.add_artist(e)

plt.savefig('objects.png',bbox_inches='tight',dpi=600)
```



[15]:
```
# available fields
objects.dtype.names
```

[15]:
```
('thresh',
 'npix',
 'tnpix',
 'xmin',
 'xmax',
 'ymin',
 'ymax',
 'x',
 'y',
 'x2',
 'y2',
 'xy',
 'errx2',
 'erry2',
 'errxy',
 'a',
 'b',
 'theta',
 'cxx',
 'cyy',
 'cxy',
 'cflux',
 'flux',
 'cpeak',
 'peak',
 'xcpeak',
 'ycpeak',
 'xpeak',
 'ypeak',
 'flag')
```

[16]:
```python
flux, fluxerr, flag = sep.sum_circle(data_sub, objects['x'], objects['y'],
                                     3.0, err=bkg.globalrms, gain=1.0)
```

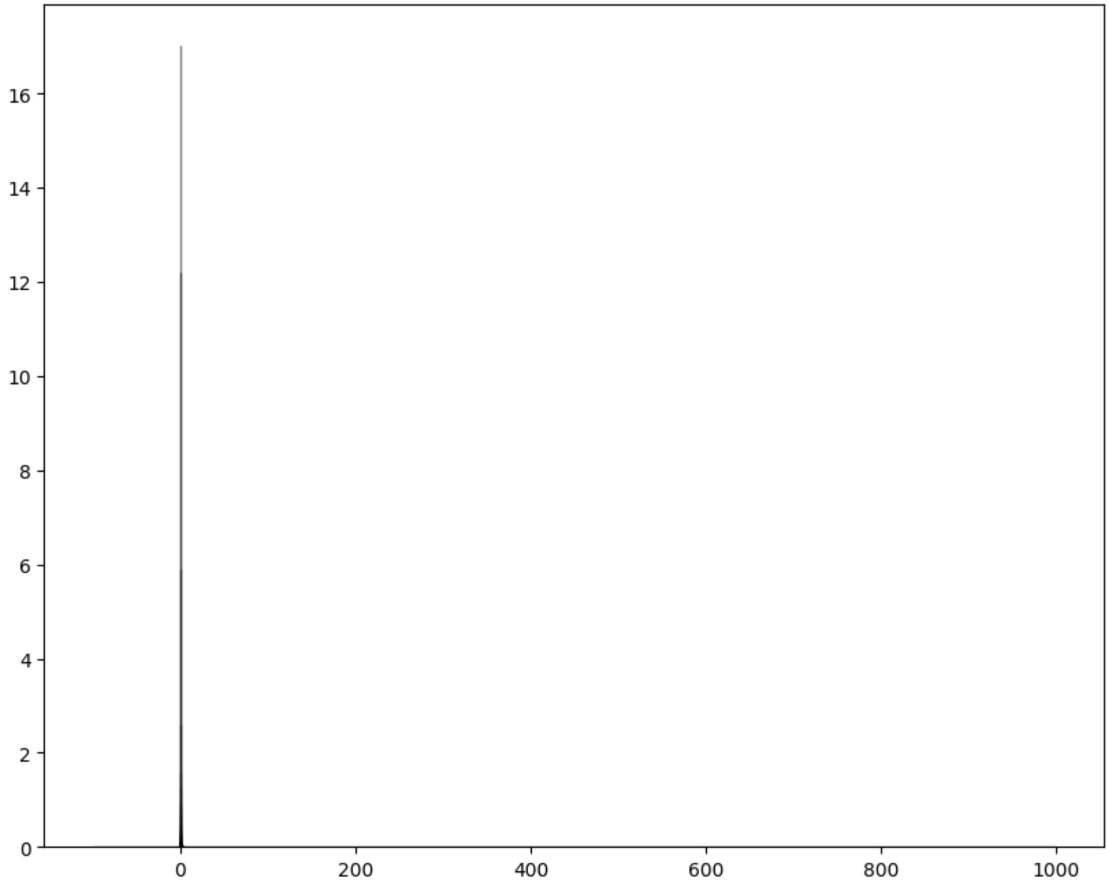
[17]:
```python
# show the first 10 objects results:
for i in range(10):
    print("object {:d}: flux = {:f} +/- {:f}".format(i, flux[i], fluxerr[i]))
```

```
object 0: flux = 0.031282 +/- 0.176890
object 1: flux = 0.031018 +/- 0.176142
object 2: flux = -0.024388 +/- 0.002883
object 3: flux = 0.001947 +/- 0.044219
object 4: flux = 0.012457 +/- 0.111649
object 5: flux = -0.011228 +/- 0.002875
object 6: flux = 0.029368 +/- 0.171394
object 7: flux = -0.009126 +/- 0.002875
object 8: flux = 0.048023 +/- 0.219161
object 9: flux = 0.027840 +/- 0.166877
```

[19]:

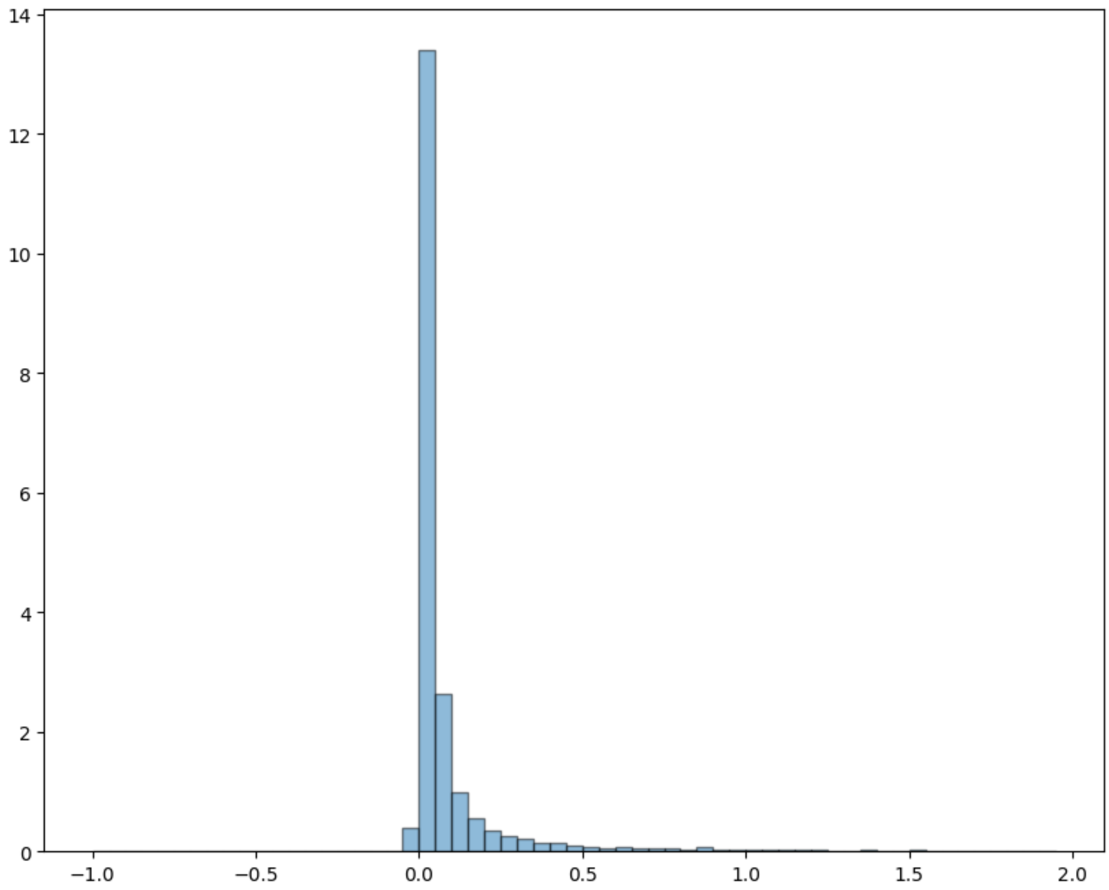
```
# number of sources:
print("The number of sources is:",flux.size)
# histogram of sources
width = 0.02
my_bins = np.arange(-100,1000,width)
plt.hist(flux,bins=my_bins,alpha=0.5,edgecolor='black',density='True')
plt.show()
```

```
The number of sources is: 8640
```



[23]:

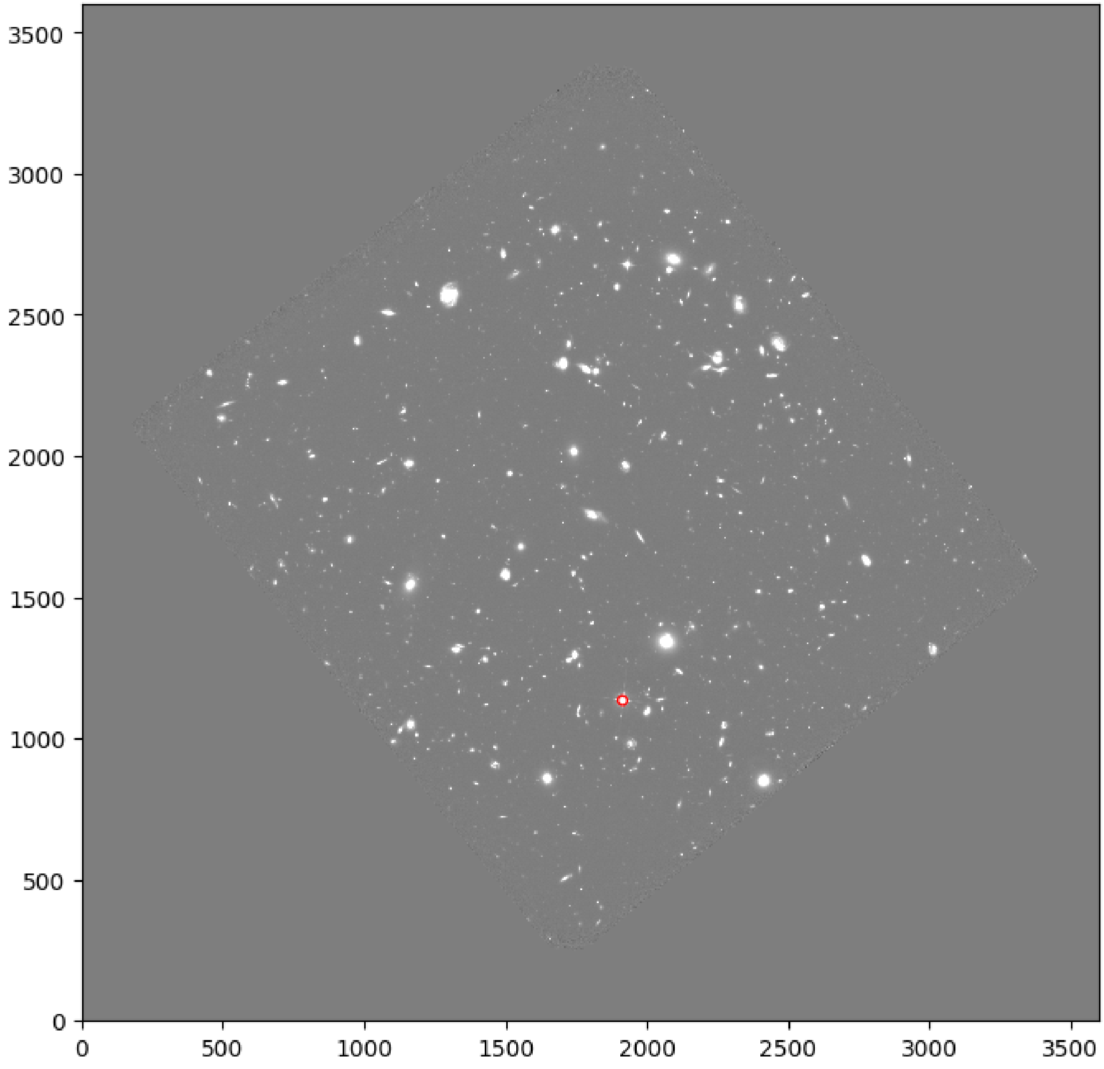
```
# A nicer looking histogram of sources
width = 0.05
my_bins = np.arange(-1,2,width)
plt.hist(flux,bins=my_bins,alpha=0.5,edgecolor='black',density='True')
plt.show()
```



[24]:

```
# mean, median, and std deviation
print("The mean of the fluxes is",np.mean(flux))
print("The median of the fluxes is",np.median(flux))
print("The standard deviation of the fluxes is",np.std(flux))
# outlier
outlier = 0
find = 0
for i in range(flux.size):
    if outlier < flux[i]:
        outlier = flux[i]
        find = i
print("The largest outlier is:",outlier)
# show where the largest outlier is
fig, ax = plt.subplots()
m, s = np.mean(data_sub), np.std(data_sub)
im = ax.imshow(data_sub, interpolation='nearest', cmap='gray',
               vmin=m-s, vmax=m+s, origin='lower')
e = Ellipse(xy=(objects['x'][find], objects['y'][find]),
                width=6*objects['a'][find],
                height=6*objects['b'][find],
                angle=objects['theta'][find] * 180. / np.pi)
e.set_facecolor('none')
e.set_edgecolor('red')
ax.add_artist(e)
# sigma of outlier
sigma = (outlier-np.mean(flux)/np.std(flux))
print("The largest outlier is",sigma,"standard deviations away from the mean")
```

```
The mean of the fluxes is 0.36185728037707154
The median of the fluxes is 0.030960064365426664
The standard deviation of the fluxes is 9.243528029706706
The largest outlier is: 807.2972835731507
The largest outlier is 807.2581364770188 standard deviations away from the mean
```



[25]:

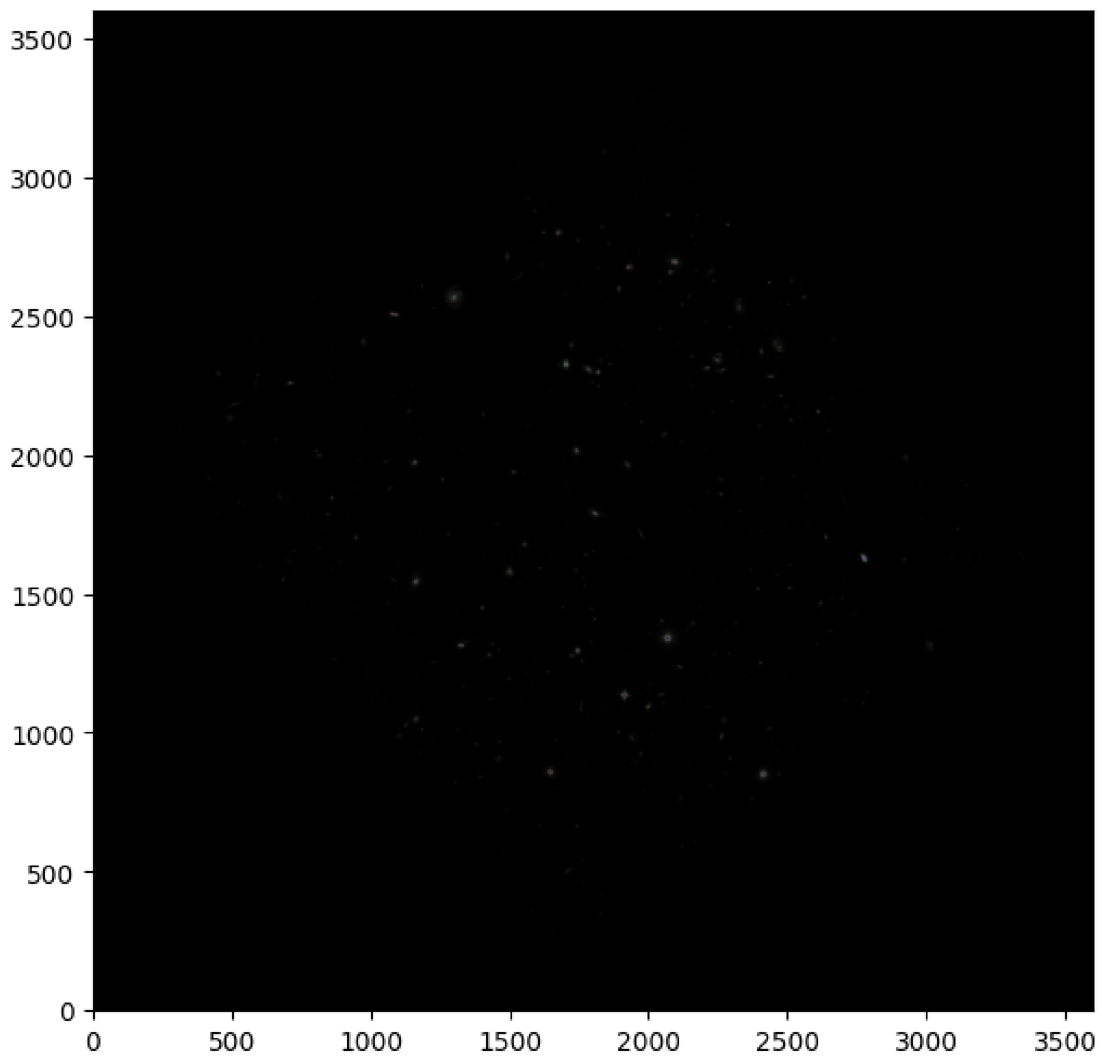
```python
# three color false image
import matplotlib.colors as colors
image2 = fits.open("f125.fits")
image.info()
data2 = image[0].data

image3 = fits.open("f160.fits")
image.info()
data3 = image[0].data

h = data3
s = data2
v = data
hsv_image = np.zeros((3600,3600,3))
hsv_image[:,:,0] = h
hsv_image[:,:,1] = s
hsv_image[:,:,2] = v
rgb_convert = colors.hsv_to_rgb(hsv_image)
f = plt.figure(figsize=(7,7))
plt.imshow(rgb_convert,origin='lower')
```

```
Filename: f105.fits
No.    Name      Ver    Type      Cards   Dimensions   Format
  0  PRIMARY       1 PrimaryHDU     359   (3600, 3600)   float32
Filename: f105.fits
No.    Name      Ver    Type      Cards   Dimensions   Format
  0  PRIMARY       1 PrimaryHDU     359   (3600, 3600)   float32
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers).
```

[25]:

```
<matplotlib.image.AxesImage at 0x154128093d0>
```


3500
3000
2500
2000
1500
1000
500
0
0
500
1000
1500
2000
2500
3000
3500

```
[ ]:
```