Final-Project-Tutorial

March 18, 2024

[1]:

```
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.]
```

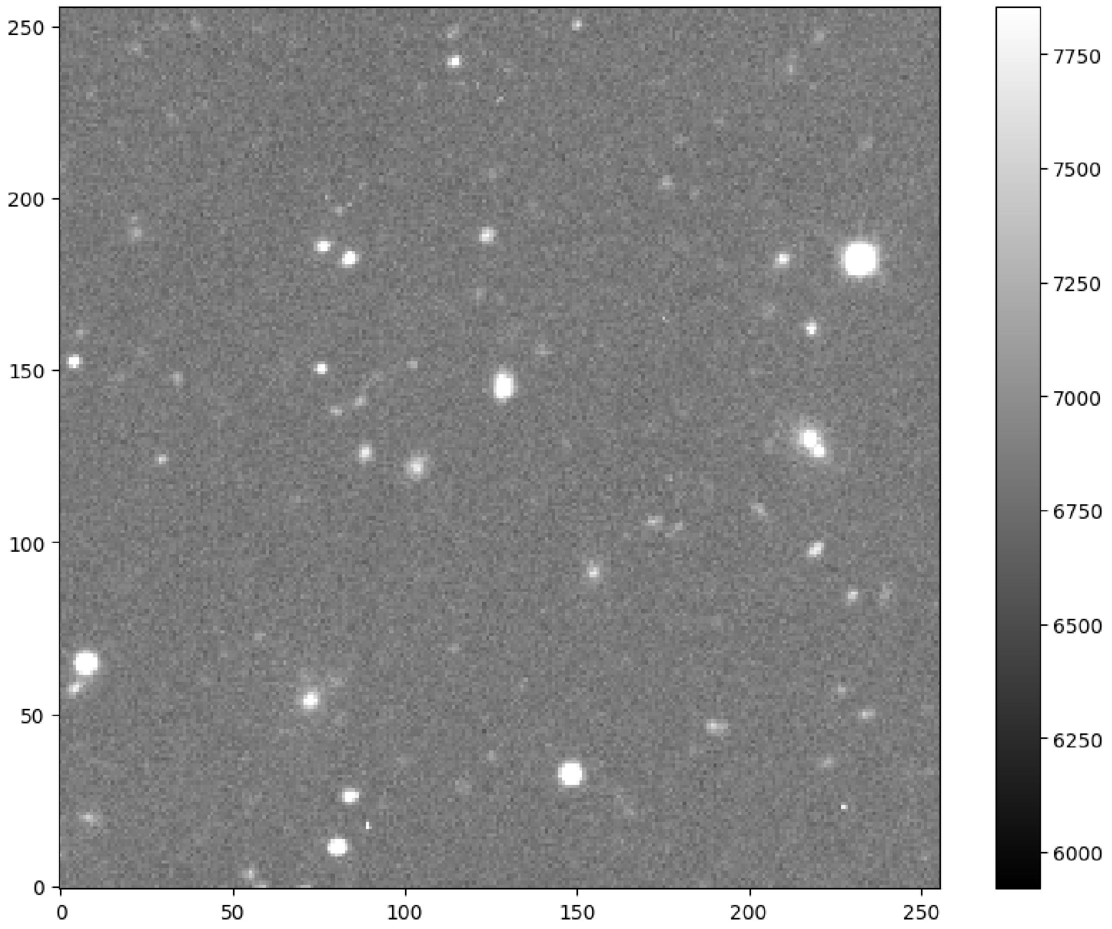
[2]:

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

```
Filename: image.fits
No.    Name      Ver    Type      Cards   Dimensions   Format
  0  PRIMARY       1 PrimaryHDU     337   (256, 256)   int16 (rescales to
float32)
<class 'numpy.ndarray'>
```

```
WARNING: The following header keyword is invalid or follows an unrecognized non-
standard convention:
ESO-LOG 00:00:00> DATE          = '1992-10-26'  / Mon Oct 26, 1992
[astropy.io.fits.card]
WARNING: The following header keyword is invalid or follows an unrecognized non-
standard convention:
ESO-LOG 03:04:08>-START EXPO EMMI RED          / Start exp. on EMMI Red CC
[astropy.io.fits.card]
WARNING: The following header keyword is invalid or follows an unrecognized non-
standard convention:
ESO-LOG 03:04:09> EXPO EMMI RED NO = 24887     / Exp. num. on EMMI Red CCD
[astropy.io.fits.card]
WARNING: The following header keyword is invalid or follows an unrecognized non-
standard convention:
ESO-LOG 03:10:52>-STOP EXPO EMMI RED           / Stop exp. on EMMI Red CCD
[astropy.io.fits.card]
```

[18]:
```
# 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)
```

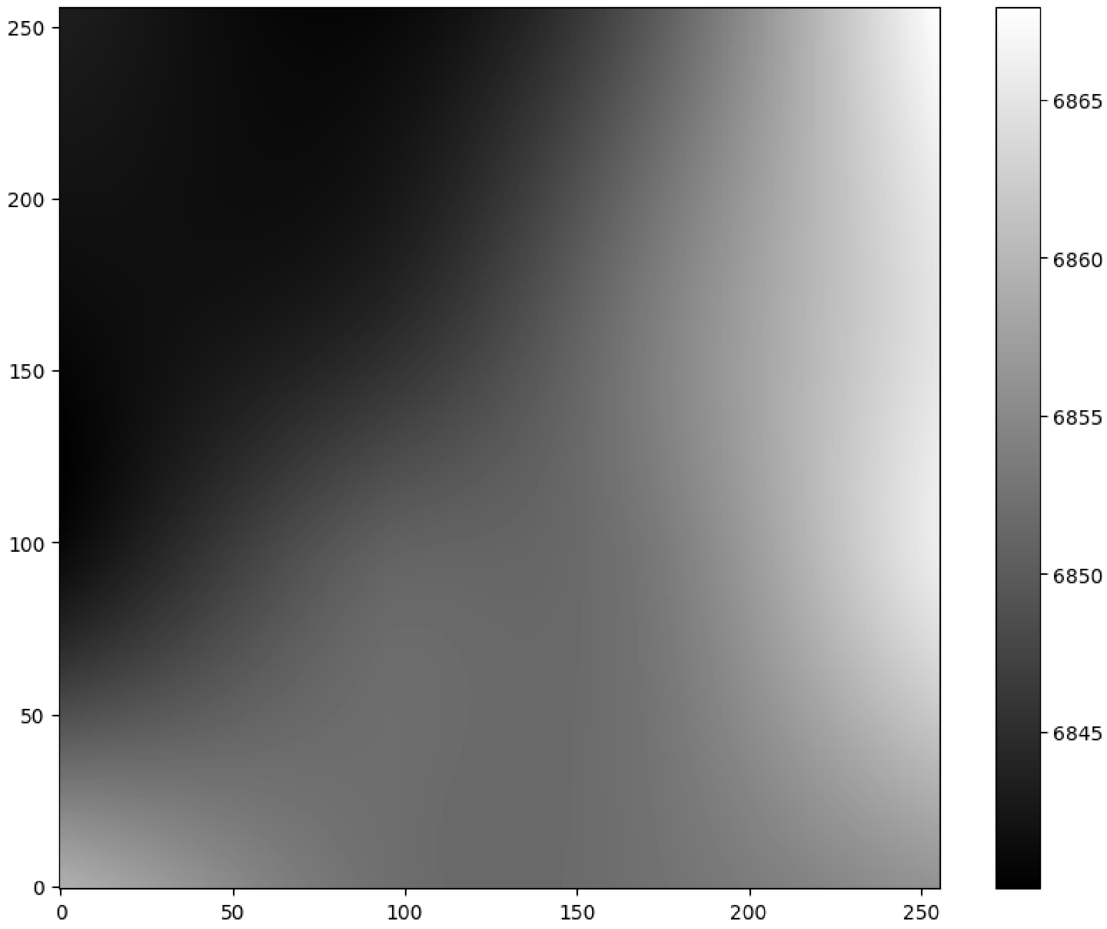
```
6852.04931640625
65.46174621582031
```

[7]:

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

[19]:

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

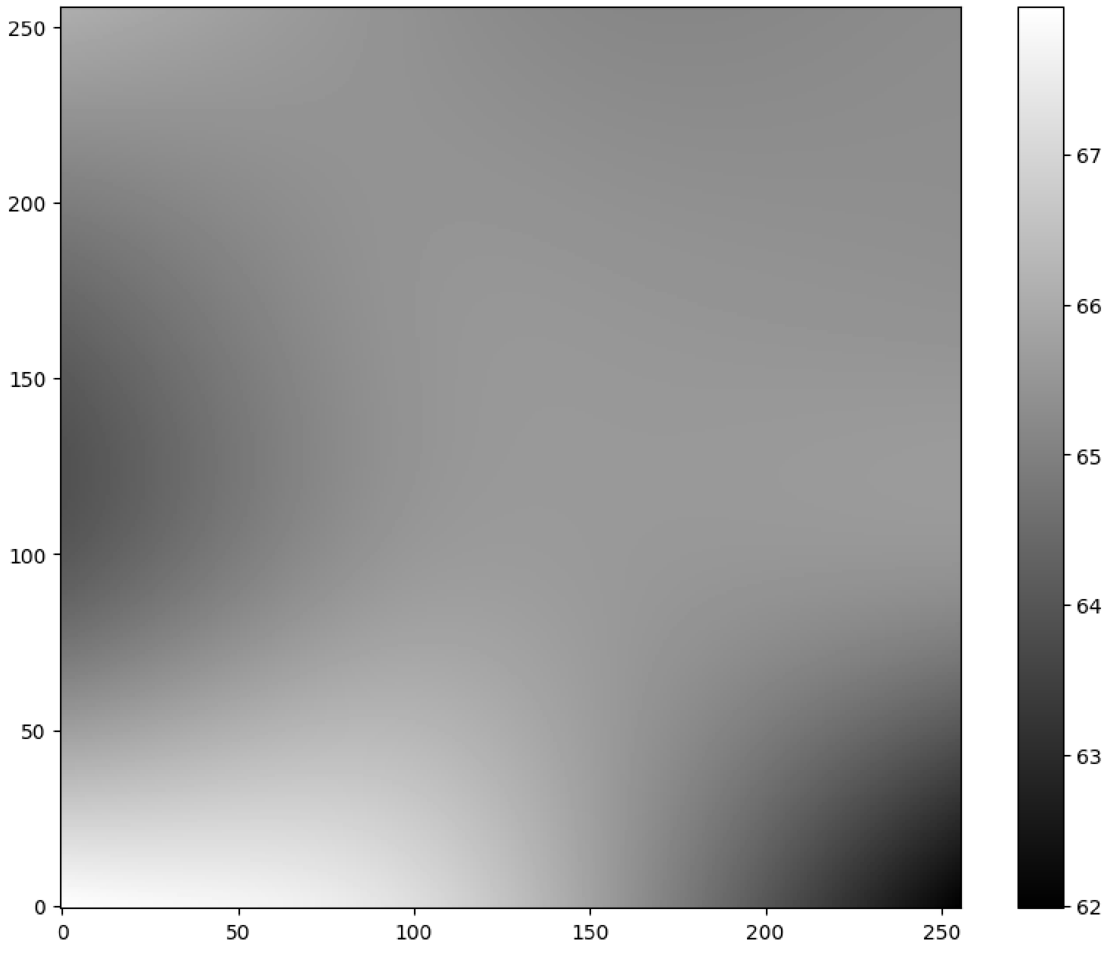


[9]:

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

[20]:

```
# 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)
```

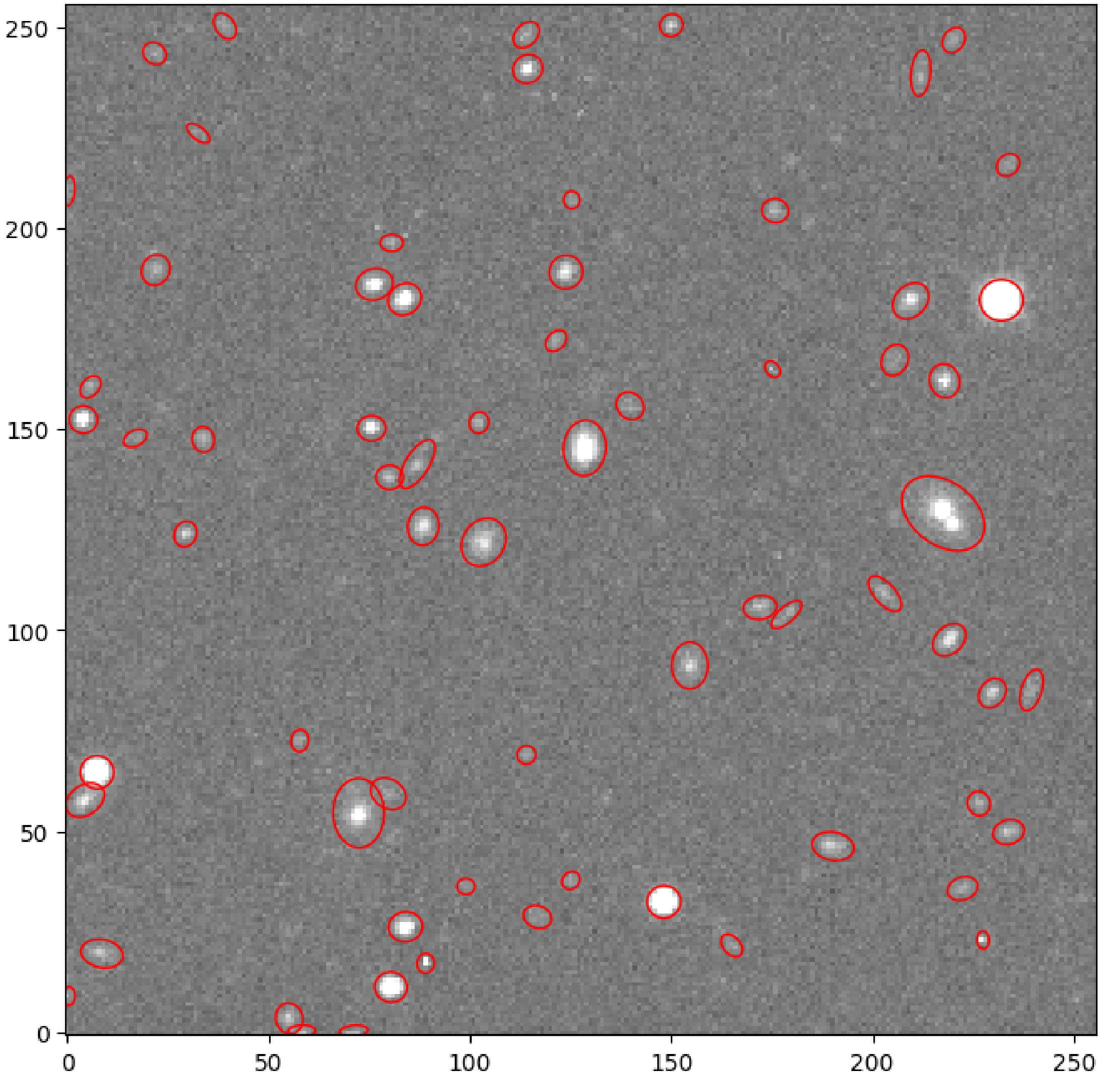
[13]: 68

[21]:
```
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]:

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

[17]:

```
# 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 = 2249.159297 +/- 291.027802
object 1: flux = 3092.220430 +/- 291.592204
object 2: flux = 5949.868379 +/- 356.562003
object 3: flux = 1851.426582 +/- 295.028816
object 4: flux = 72736.386914 +/- 440.172206
object 5: flux = 3860.756152 +/- 352.163162
object 6: flux = 6418.913789 +/- 357.458973
object 7: flux = 2210.707656 +/- 350.791223
object 8: flux = 2741.607227 +/- 352.277746
object 9: flux = 20916.875566 +/- 376.966138
```