Jennifer Vezilj
11 May 2016

Single Star Photometry

Aims

The purpose of this experiment is to analyze images collected by the Clio detector and to determine the effect of the neutral density (ND) filters on the star's flux.

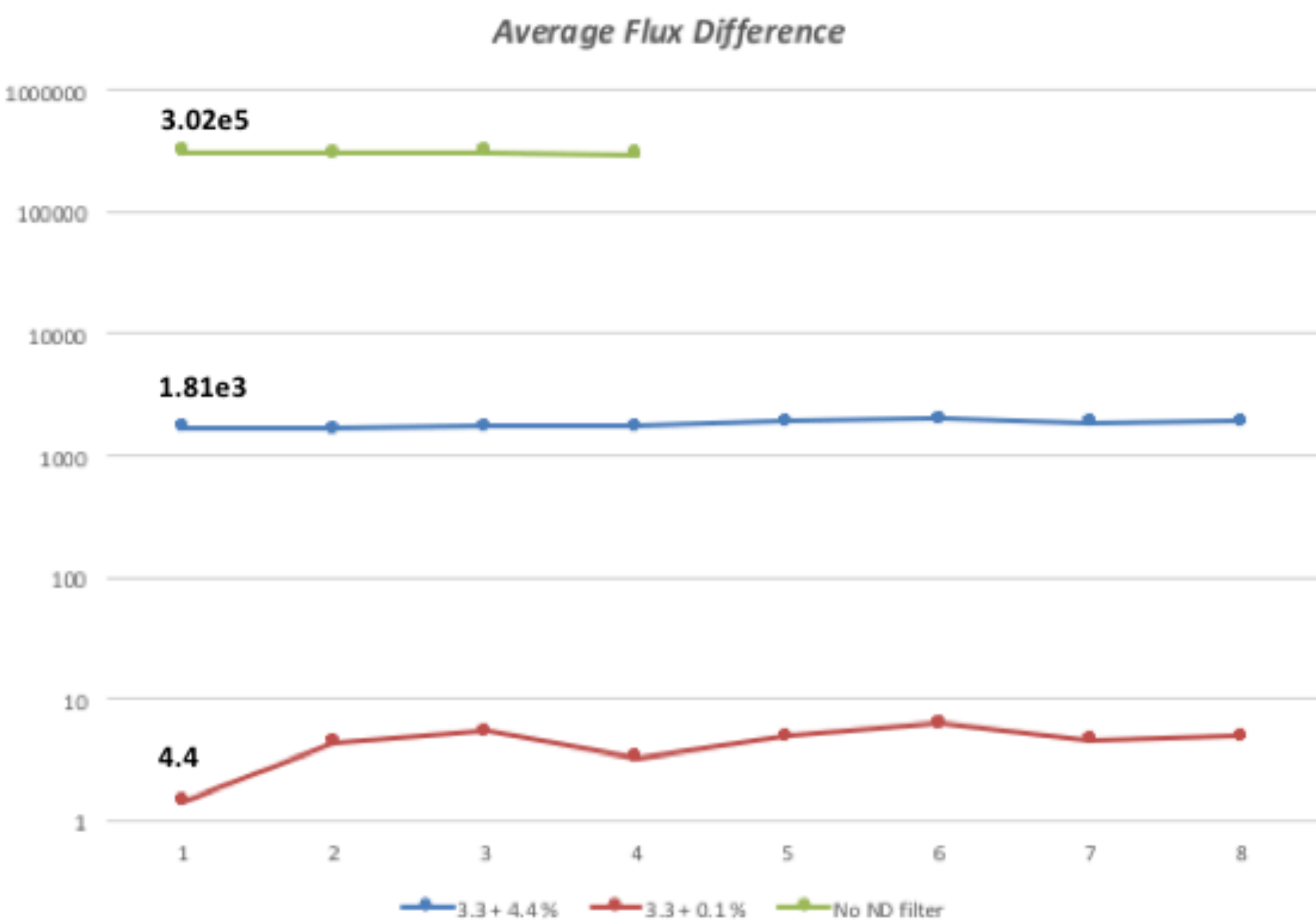
Method

Two sets of imaging data taken on the nights of December 1st and 2nd of 2014 of Beta Pictoris were used to determine flux in the brightness of the observed star. The first of the two data sets contained eight photos taken at an integration (int.) time of three-milliseconds and lacked the use of any neutral density (ND) filters. An ND filter of two different strengths was adapted to the Clio detector to compare the flux through the NDs. The second set of images were grouped more scarcely due to the two different ND filters and the various int. times.
SAOImage DS9 and the programing language, Python were the primary applications used to produce the calculated results. The original images taken from the Clio detector contain too much background noise for the experiment to be properly carried out and in order focus on Beta Pictoris, an image containing Beta Pictoris was subtracted from one of the empty sky. This step, as well as the following two, were completed by creating the appropriate code (a list of the codes used is located below). Following the image subtraction, the parameters full width half max (FWHM), threshold and radius had to be chosen to find the star above. Once it was determined that the inputs were uniform, and SAOImaging and DAO Find was able to correctly identify just Beta Pictoris, the images were put through one last code. The inputs used in the last code, were again used in the last step of finding Beta Pictoris' photometry. As well as locating the star, this code measured the Beta Pictoris' parameters from outputs and wrote them to a separate file. These outputs were: x-centroid, y-centroid, sharpness, roundness 1, roundness 2, npix, sky, peak, flux, mag, aperture sum, x-center and y-center.

Image Name	**Test33_ ...fit**	**Test33_4ND…fit**
Subtraction	test33_sub.py	test33_4nd_sub.py
Parameter Testing	test33_ap.py	test33_4nd_ap.py
Photometry	test33_phot.py	test33_4nd_phot.py

Image	ND Filter	Int (ms)	fwhm	Threshold	Radius
Test33_00001.fir	-	3	10	50	23
Test33_00002.fit	-	3	10	100	23
Test33_00003.fit	-	3	10	50	23
Test33_00004.fit	-	3	10	100	23
Test33_4ND_00001.fit	3.3+4.4%	1000	5	20	23
Test33_4ND_00002.fit	3.3+4.4%	1000	5	20	23
Test33_4ND_00007.fit	3.3+4.4%	1000	15	10	23
Test33_4ND_00008.fit	3.3+4.4%	1000	15	10	23
Test33_4ND_00009.fit	3.3+4.4%	100	15	10	23
Test33_4ND_00015.fit	3.3+4.4%	100	15	10	23
Test33_4ND_00016.fit	3.3+4.4%	100	15	10	23
Test33_4ND_00018.fit	3.3+0.1%	1000	10	10	23
Test33_4ND_00023.fit	3.3+0.1%	1000	10	10	23
Test33_4ND_00025.fit	3.3+0.1%	10000	10	20	23
Test33_4ND_00026.fit	3.3+0.1%	10000	10	20	23
Test33_4ND_00031.fit	3.3+0.1%	10000	10	20	23
Test33_4ND_00032.fit	3.3+0.1%	10000	10	20	23
Test33_4ND_00039.fit	3.3+0.1%	20000	10	20	23
Test33_4ND_00040.fit	3.3+0.1%	20000	10	20	23

Results



The graph above shows the average flux of Beta Pictoris after dividing the aperture sum by the images integration time. The blue line, 4.4% ND filter at 3.3 micro filter, reduces the flux by a factor of **5.99 e-3**, while the 0.1% ND filter, represented in red, reduces the flux by a factor of **1.46 e-5**.

The data below is the collected information from the outputs used in the third and final code.

A	B	C	D	E	F	G	H	I	J	K
	I.D.	**x-centroid**	**y-centriod**	**sharpness**	**roundness 1**	**roundness 2**	**npix**	**sky**	**peak**	**flux**
Test33_4ND00001.fit	1	74.68034058	23.84256085	0.429598425	0.01127439	0.05855935	49	0	20233	16.53597828
Test33_4ND00002.fit	2	74.63592486	23.78528051	0.434305302	0.0122627	0.154048274	49	0	19774	15.73077005
Test33_4ND00007.fit	3	74.80948274	23.58158543	0.730786044	0.06486483	0.073634052	361	0	22203	189.9428878
Test33_4ND00008.fit	4	74.85275991	23.58555938	0.701732986	0.03574947	-0.005899789	361	0	20750	174.9685416
Test33_4ND00009.fit	5	74.95942625	24.21809283	0.420123596	-0.02132531	0.099683112	49	0	2705	2.717840119
Test33_4ND00010.fit	6	74.5427476	23.90332009	0.42345612	-0.12289094	0.083111931	49	0	2536	2.497715594
Test33_4ND00015.fit	7	74.61969859	23.28460696	0.404811677	0.01720431	0.16574252	49	0	2689	2.827056241
Test33_4ND00016.fit	8	74.73588031	23.32999364	0.411259507	0.00408601	0.169659685	49	0	2652	3.045977304
Test33_4ND00018.fit	9	76.1433939	23.83809442	0.601546946	-0.08645173	0.063645187	169	0	70	1.452904746
Test33_4ND00023.fit	10	76.31500037	23.89176821	0.380707979	-0.154244	0.26084963	169	0	58	1.346938377
Test33_4ND00025.fit	11	75.63730508	23.96819603	0.402214295	-0.09288553	0.041736821	169	0	536	5.050358426
Test33_4ND00026.fit	12	75.64631948	23.85728533	0.423147474	0.0121464	0.024437784	169	0	558	5.161361512
Test33_4ND00031.fit	13	75.63424149	24.11414722	0.410093641	-0.06964099	0.021073769	169	0	506	5.076840226
Test33_4ND00032.fit	14	75.76541285	24.00277851	0.393897311	-0.06839177	0.051442134	169	0	572	5.451284864
Test33_4ND00039.fit	15	221.9042318	170.4688464	0.429384144	0.04470307	-0.11054373	49	0	988	1.057296564
Test33_4ND00040.fit	16	222.0746375	170.758298	0.387846795	0.04020607	-0.135899672	49	0	1273	1.489924219
Test33_00001.fit	17	76.19047447	24.55460673	0.438132844	-0.16402653	-0.09019866	169	0	10931	24.25063514
Test33_00002.fit	18	73.9337004	23.62649624	0.487336512	0.07429559	0.103538075	169	0	12903	15.73714233
Test33_00007.fit	19	74.97687957	23.68983195	0.473044164	-0.00446165	0.101721618	169	0	12991	28.90119686
Test33_00008.fit	20	74.13374972	24.18917549	0.494427149	0.04429407	0.087740189	169	0	12610	13.65936959

mag	aperture sum	xcenter	ycenter	int(ms)	peak/int	aperture sum/int
-3.046074733	1686792.263	[74.68034057849569]	[23.84256085481039]	1000	20.233	1686.792263
-2.991874957	1659138.277	[74.63592486123056]	[23.785280513168754]	1000	19.774	1659.138277
-5.696557591	1705967.328	[74.80948273975454]	[23.581585429509854]	1000	22.203	1705.967328
-5.60739993	1729562.233	[74.85275991289939]	[23.585559376921697]	1000	20.75	1729.562233
-1.085559763	191767.9322	[74.95942625107925]	[24.21809282958755]	100	27.05	1917.679322
-0.993857463	196984.0037	[74.54274760110376]	[23.903320086595695]	100	25.36	1969.840037
-1.128336121	186060.6653	[74.61969858962613]	[23.28460696054357]	100	26.89	1860.606653
-1.209316658	189281.1528	[74.73588030676952]	[23.32999363741305]	100	26.52	1892.811528
-0.405592856	1436.689171	[76.14339390232944]	[23.83809441808845]	1000	0.07	1.436689171
-0.323369317	4441.324367	[76.31500036956167]	[23.891768213228236]	1000	0.058	4.441324367
-1.758305503	54922.23398	[75.63730508257733]	[23.96819602766242]	10000	0.0536	5.492223398
-1.781910697	32990.52007	[75.64631948212346]	[23.857285333806942]	10000	0.0558	3.299052007
-1.76398374	50074.45778	[75.6342414854737]	[24.114147216157992]	10000	0.0506	5.007445778
-1.841247193	63759.89621	[75.7654128533899]	[24.002778514252086]	10000	0.0572	6.375989621
-0.060492052	92436.42707	[221.90423176690632]	[170.46884641029024]	20000	0.0494	4.621821353
-0.432910449	98324.98993	[222.0746375475127]	[170.75829795901774]	20000	0.06365	4.916249496
-3.461807794	922746.2278	[76.19047446901945]	[24.55460672921739]	3	3643.67	307582.0759
-2.992314682	903515.9405	[73.93370039964732]	[23.626496242000307]	3	4301	301171.9802
-3.65228957	927116.8762	[74.97687956938745]	[23.689831949343432]	3	4330.33	309038.9587
-2.83857664	881390.0571	[74.13374971717]	[24.18917548608265]	3	4203.33	293796.6857

Image Number	ND	int(ms)	Aperture Sum	Average (ND and int)	Mean	Median	Standard Deviation	Percent Error
Test33_4ND_00001.fit	3.3+4.4%	1000	1686792.263	1.68 e3				
Test33_4ND_00002.fit	3.3+4.4%	1000	1659138.277	1.65 e3				
Test33_4ND_00007.fit	3.3+4.4%	1000	1705967.328	171 e3				
Test33_4ND_00008.fit	3.3+4.4%	1000	1729562.233	1.72 e3				
Test33_4ND_00009.fit	3.3+4.4%	100	191767.9322	1.91 e3				
Test33_4ND_00010.fit	3.3+4.4%	100	196984.0037	1.96 e3				
Test33_4ND_00015.fit	3.3+4.4%	100	186060.6653	1.86 e3				
Test33_4ND_00016.fit	3.3+4.4%	100	189281.1528	1.89 e3	1.81 e3	1.82 e3	1.14 e3	6.29%
Test33_4ND_00018.fit	3.3+0.1%	1000	1436.689171	1.44 e3				
Test33_4ND_00023.fit	3.3+0.1%	1000	4441.324367	4.44				
Test33_4ND_00025.fit	3.3+0.1%	10000	54922.23398	5.49				
Test33_4ND_00026.fit	3.3+0.1%	10000	32990.52007	3.3				
Test33_4ND_00031.fit	3.3+0.1%	10000	50074.45778	5.01				
Test33_4ND_00032.fit	3.3+0.1%	10000	63759.89621	6.38				
Test33_4ND_00039.fit	3.3+0.1%	20000	92436.42707	4.62				
Test33_4ND_00040.fit	3.3+0.1%	20000	98324.98993	4.91	4.44	4.76	0.948462415	33.50%
Test33_00001.fit	**N/A**	3	307582.0759	3.07 e5				
Test33_00002.fit	**N/A**	3	301171.9802	3.01 e5				
Test33_00007.fit	**N/A**	3	309038.9587	3.09 e5				
Test33_00008.fit	**N/A**	3	298796.6857	2.93 e5	3.02 e5	3.04 e5	7.18 e3	2.37%

Appendices

Appendix A: test33_sub.py

Appendix B: test33_ap.py

Appendix C: test33_phot.py

Appendix D: test33_4nd_sub.py

Appendix E: test33_4nd_ap.py

Appendix F: test33_4nd_phot.py

Appendix A:
Subtraction of images containing no neutral density filters

```
import astropy.io.fits
import numpy
import pyds9

read1 = []
write1 = []

starlist = [1,2,7,8]
skylist = [3,4,5,6]

for i in range(4):
   print starlist[i]
   readfile1 =
"/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/raw/Test33_0000%
d.fit" %starlist[i]
   writefile1 =
"/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/sub/Test33_0000%d
.fit" %starlist[i]
   read1.append(readfile1)
   write1.append(writefile1)

read2 = []
write2 = []

for i in range(4):
   print skylist[i]
   readfile2 =
"/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/raw/Test33_0000%
d.fit" %skylist[i]
   writefile2 =
"/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/sub/Test33_0000%d
.fit" %skylist[i]
   read2.append(readfile2)
   write2.append(writefile2)


for i in range(0, 4):
        file1 = read1[i]
```

```
        file2 = read2[i]

        print file1
        print file2

for i in range(0, 4):
        print i, readfile1
        readfile1 = read1[i]
        readfile2 = read2[i]


#          path plus the file :
astropy.io.fits.open(clio_20141102_03/TrapNarrowHunsat00001)
        hdulist1 = astropy.io.fits.open(readfile1)
        hdulist2 = astropy.io.fits.open(readfile2)
        hdulist1.info()
        hdulist2.info()

#Create a new FITS file for the answer
        hdulist1sub = hdulist1
        hdulist2sub = hdulist2
        hdulist1sub[0].data = hdulist1[0].data - hdulist2[0].data
        hdulist2sub[0].data = hdulist2[0].data - hdulist1[0].data
        writefile1 = write1 [i]
        writefile2 = write2 [i]
        hdulist1sub.writeto(writefile1, clobber=True)
```

Appendix B:
Finding the parameters (full width half max, threshold and radius)

```
debugflag = 0
from astropy.io import fits
import numpy as np
file_name= "Test33_00001.fit"
hdulist = fits.open(file_name)
if debugflag == 0:
        hdulist.info
image = hdulist[0].data
if debugflag == 1:
        image.shape
#image = image.astype(float) - np.median(image)
from photutils import daofind
from astropy.stats import mad_std
bkg_sigma = mad_std(image)
if debugflag == 0:
   print (bkg_sigma)
sources = daofind(image, fwhm=10., threshold=100.*bkg_sigma)
print(sources)

from photutils import aperture_photometry, CircularAperture
positions = (sources['xcentroid'], sources['ycentroid'])
apertures = CircularAperture(positions, r=23.)
#print positions
if debugflag == 0:
   from pprint import pprint
   pprint(vars(apertures))
phot_table = aperture_photometry(image, apertures)
print (phot_table)

import matplotlib.pylab as plt
im2 = image
im2[im2<=0]=0.0001
plt.imshow(im2, cmap='gray', origin='lower')
apertures.plot(color='blue', lw=1.5, alpha=0.5)
plt.show()
```

Appendix C:
Finding the photometry

```
from astropy.io import fits
import numpy as np

read1 = []

star=[]

for i in range():
   print star[i]
   path = "/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/sub/"
   file_name = "Test33_0000%d.fit" %star[i]
   read1.append(file_name)

len(read1)

for i in range():
        file_name = read1[i]

        hdulist = fits.open(path+file_name)
        image = hdulist[0].data
        #image = image.astype(float) - np.median(image)
        from photutils import daofind
        from astropy.stats import mad_std
        bkg_sigma = mad_std(image)
        sources = daofind(image, fwhm=10., threshold=100.*bkg_sigma)
        #print_line= (file_name+","+str(sources_2)+"\n")
        sources_2 = np.array(sources["id", "xcentroid", "ycentroid", "sharpness",
"roundness1", "roundness2", "npix", "sky", "peak", "flux", "mag"])
        print_line= (file_name+","+str(sources_2))
        file=
open("/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/code/test33_p
hot.txt", "a")
        file.write(print_line)
        file.close()

        from photutils import aperture_photometry, CircularAperture
        positions = (sources['xcentroid'], sources['ycentroid'])
        apertures = CircularAperture(positions, r=23.)
        phot_table = aperture_photometry(image, apertures)
        phot_table_2 = np.array(phot_table["aperture_sum", "xcenter", "ycenter"])
        print_line= (","+str(phot_table_2)+"\n")
```

```
    file=
open("/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/code/test33_p
hot.txt", "a")
    file.write(print_line)
    file.close()

    import matplotlib.pylab as plt
    im2 = image
    im2[im2<=0]=0.0001
    plt.imshow(im2, cmap='gray', origin='lower')
    apertures.plot(color='blue', lw=1.5, alpha=0.5)
    plt.show()
```

Appendix D:
Subtraction of images containing neutral density filters at two different

```
import astropy.io.fits
import numpy
import pyds9

read1 = []
write1 = []

starlist = []
skylist = []

for i in range():
    print starlist[i]
    if starlist[i] < 10:
        readfile1 =
"/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/raw/Test33_4ND00
00%d.fit" %starlist[i]
        writefile1 =
"/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/sub/Test33_4ND000
0%d.fit" %starlist[i]
        read1.append(readfile1)
        write1.append(writefile1)

    else:
        readfile1 =
"/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/raw/Test33_4ND00
0%d.fit" %starlist[i]
        writefile1 =
"/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/sub/Test33_4ND000
%d.fit" %starlist[i]
        read1.append(readfile1)
        write1.append(writefile1)

read2 = []
write2 = []

for i in range():
    print skylist[i]
    readfile2 =
"/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/raw/Test33_4ND00
0%d.fit" %skylist[i]
```

```
  writefile2 =
"/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/sub/Test33_4ND000
%d.fit" %skylist[i]
   read2.append(readfile2)
   write2.append(writefile2)

for i in range(0, 4):
        file1 = read1[i]
        file2 = read2[i]

        print file1
        print file2

for i in range(0, 4):
        print i, readfile1
        readfile1 = read1[i]
        readfile2 = read2[i]


#           path plus the file :
astropy.io.fits.open(clio_20141102_03/TrapNarrowHunsat00001)
        hdulist1 = astropy.io.fits.open(readfile1)
        hdulist2 = astropy.io.fits.open(readfile2)
        hdulist1.info()
        hdulist2.info()

#Create a new FITS file for the answer
        hdulist1sub = hdulist1
        hdulist2sub = hdulist2
        hdulist1sub[0].data = hdulist1[0].data - hdulist2[0].data
        hdulist2sub[0].data = hdulist2[0].data - hdulist1[0].data
        writefile1 = write1 [i]
        writefile2 = write2 [i]
        hdulist1sub.writeto(writefile1, clobber=True)
```

Appendix E:
Finding the parameters (full width half max, threshold and radius)

```
debugflag = 0
from astropy.io import fits
import numpy as np
file_name= "Test33_4ND00001.fit"
hdulist = fits.open(file_name)
if debugflag == 0:
	hdulist.info
image = hdulist[0].data
if debugflag == 0:
	image.shape
#image = image.astype(float) - np.median(image)
from photutils import daofind
from astropy.stats import mad_std
bkg_sigma = mad_std(image)
if debugflag == 1:
   print (bkg_sigma)
sources = daofind(image, fwhm=10., threshold=20.*bkg_sigma)
print(sources)

from photutils import aperture_photometry, CircularAperture
positions = (sources['xcentroid'], sources['ycentroid'])
apertures = CircularAperture(positions, r=23.)
#print positions
if debugflag == 0:
   from pprint import pprint
   pprint(vars(apertures))
phot_table = aperture_photometry(image, apertures)
print (phot_table)

import matplotlib.pylab as plt
im2 = image
im2[im2<=0]=0.0001
plt.imshow(im2, cmap='gray', origin='lower')
apertures.plot(color='blue', lw=1.5, alpha=0.5)
plt.show()
```

Appendix F:
Finding the photometry

```
from astropy.io import fits
import numpy as np

read1 = []

star=[]

for i in range():
   if star[i] < 10:
      path = "/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/sub/"
      file_name = "Test33_4ND0000%d.fit" %star[i]
      read1.append(file_name)
   else:
      path = "/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/sub/"
      file_name = "Test33_4ND000%d.fit" %star[i]
      read1.append(file_name)

len(read1)

for i in range():
   file_name = read1[i]

   hdulist = fits.open(path+file_name)
   image = hdulist[0].data
         #image = image.astype(float) - np.median(image)
   from photutils import daofind
   from astropy.stats import mad_std
   bkg_sigma = mad_std(image)
   sources = daofind(image, fwhm=10, threshold=10*bkg_sigma)
         #print_line= (file_name+","+str(sources_2)+"\n")
   sources_2 = np.array(sources["id", "xcentroid", "ycentroid", "sharpness",
"roundness1", "roundness2", "npix", "sky", "peak", "flux", "mag"])
   print_line= (file_name+","+str(sources_2))
   file=
open("/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/code/test33_4
nd_phot.txt", "a")
   file.write(print_line)
   file.close()

   from photutils import aperture_photometry, CircularAperture
   positions = (sources['xcentroid'], sources['ycentroid'])
```

```
apertures = CircularAperture(positions, r=23.)
    phot_table = aperture_photometry(image, apertures)
    phot_table_2 = np.array(phot_table["aperture_sum", "xcenter", "ycenter"])
    print_line= (","+str(phot_table_2)+"\n")
    file=
open("/Users/jennifervezilj/mypy/Python/data/clio_20141201_02/code/test33_4
nd_phot.txt", "a")
    file.write(print_line)
    file.close()

    import matplotlib.pylab as plt
    im2 = image
    im2[im2<=0]=0.0001
    plt.imshow(im2, cmap='gray', origin='lower')
    apertures.plot(color='blue', lw=1.5, alpha=0.5)
    plt.show()
```