06/06/22 - Running MGE and JAM on J0037 cleanly using functions from script slacs_mge_jampy.py

In [1]:

```
################################################################

# import general libraries and modules
import numpy as np
np.set_printoptions(threshold=10000)
import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"] = (8, 6)
import pandas as pd
import warnings
warnings.filterwarnings( "ignore", module = "matplotlib\..*" )
warnings.filterwarnings( "ignore", module = "plotbin\..*" )
from os import path

# astronomy/scipy
from astropy.io import fits
from astropy.wcs import WCS
from scipy.ndimage import rotate
from astropy.cosmology import Planck18 as cosmo  # Planck 2018
from scipy.interpolate import interp1d
from scipy.optimize import fsolve

# mge fit
import mgefit
from mgefit.find_galaxy import find_galaxy
from mgefit.mge_fit_1d import mge_fit_1d
from mgefit.sectors_photometry import sectors_photometry
from mgefit.mge_fit_sectors import mge_fit_sectors
from mgefit.mge_print_contours import mge_print_contours
from mgefit.mge_fit_sectors_regularized import mge_fit_sectors_regula

# jam
from jampy.jam_axi_proj import jam_axi_proj
from jampy.jam_axi_proj import rotate_points
from plotbin.plot_velfield import plot_velfield
from plotbin.sauron_colormap import register_sauron_colormap
from pafit.fit_kinematic_pa import fit_kinematic_pa

# my functions
from slacs_mge_jampy import crop_center_image
from slacs_mge_jampy import import_center_crop
from slacs_mge_jampy import try_fractions_for_find_galaxy
from slacs_mge_jampy import convert_mge_model_outputs
from slacs_mge_jampy import plot_contours_321
from slacs_mge_jampy import load_2d_kinematics
from slacs_mge_jampy import bin_velocity_maps
from slacs_mge_jampy import rotate_bins
from slacs_mge_jampy import osipkov_merritt_model
from slacs_mge_jampy import find_half_light
from slacs_mge_jampy import calculate_minlevel

################################################################
# some needed constants
kcwi_scale = 0.147  # arcsec/pixel
hst_scale = 0.050 # ACS/WFC
```

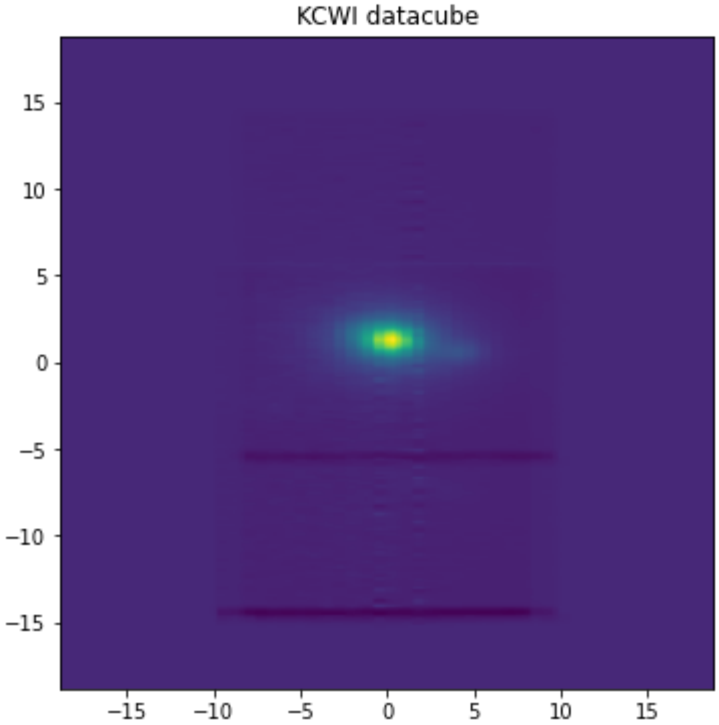
In [2]:

```
# specify object directory and name

data_dir = '/home/shawn/data/' # data directory
obj_name = 'SDSSJ0037-0942' # e.g. SDSSJ0037-0942
obj_abbr = obj_name[4:9] # e.g. J0029
file_dir = f'{data_dir}CF_mosaics/{obj_name}/' # directory with all f
```

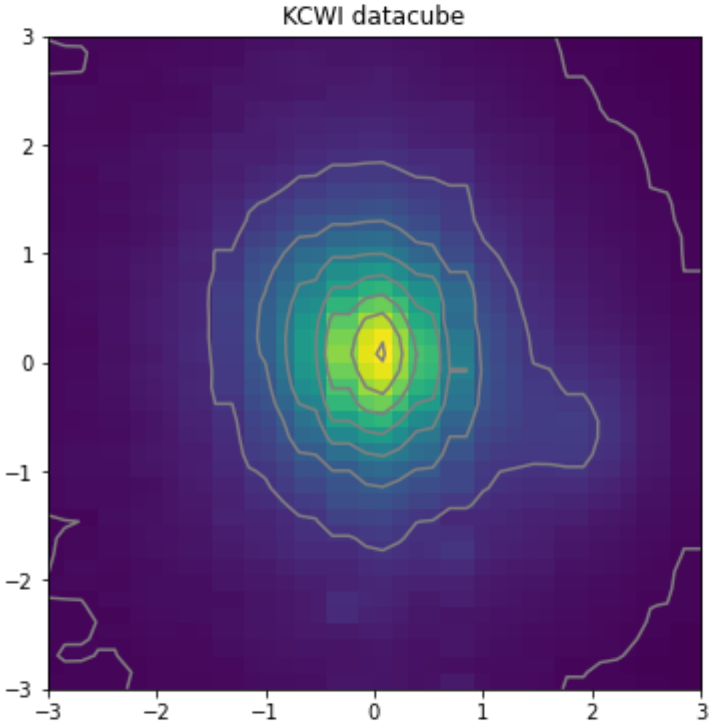
First look at the KCWI integrated datacube and HSTF435W image, crop to 3 arcsec

In [3]:

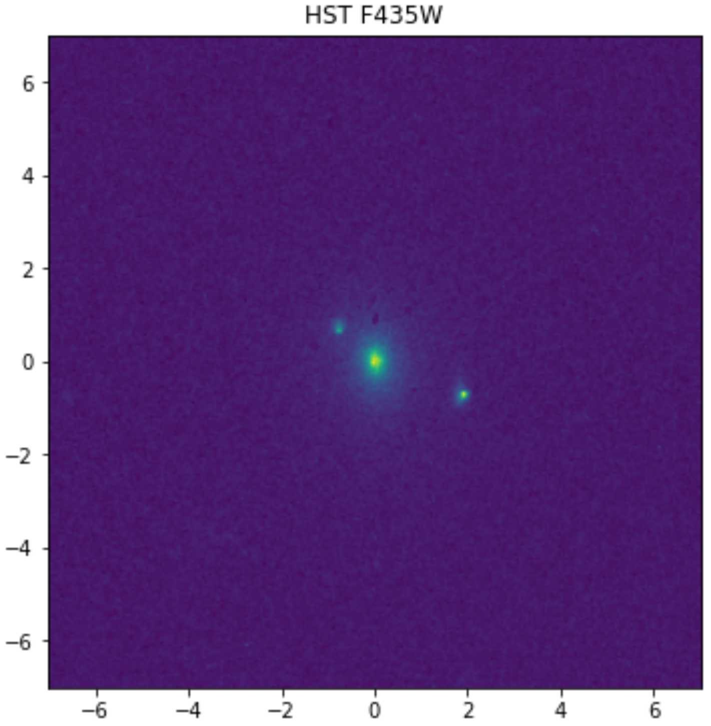
```
# import image, center, and crop

#####################################################################
# kcwi datacube

kcwi_img, kcwi_3arc_img, kcwi_header, \
    kcwi_central_pix_x, kcwi_central_pix_y = import_center_crop(file_
                                                         data_source

#####################################################################
# F435W cutout

hstF435_img, hstF435_3arc_img, hstF435_header, \
    central_pix_x, central_pix_y = import_center_crop(file_dir, obj_n
                                                     data_source='F4
```

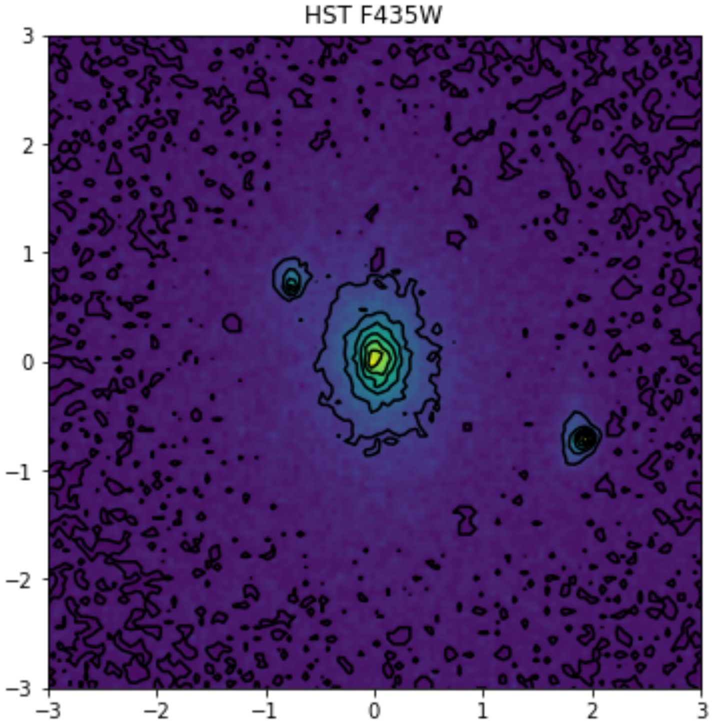


KCWI datacube



KCWI datacube

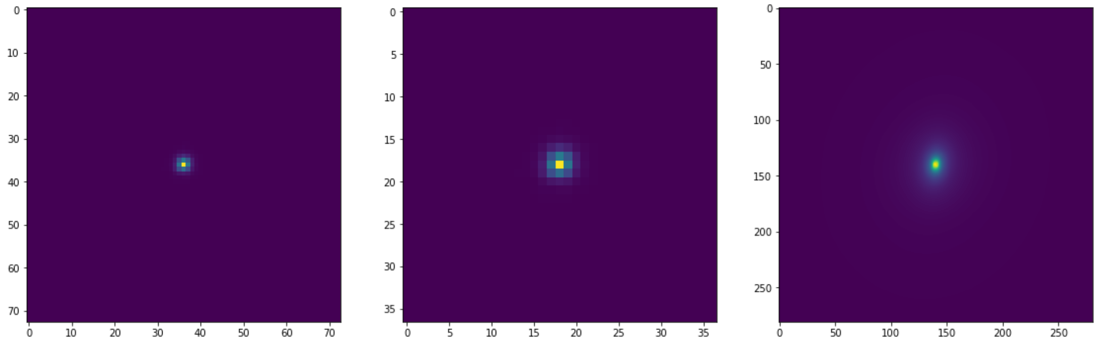




Estimate PSF using Gaussian MGE for each image...

In [4]:

```
# Let's look at the HST image first
file = f'{file_dir}{obj_name}_F435W.fits'
hdul = fits.open(file)

plt.figure()
plt.tight_layout()

#####################################################################
# 4th hdu is psf
psf_hdu = hdul[3]
hst_psf_model = psf_hdu.data
hst_psf_header = psf_hdu.header
#print(hst_psf_header) # header is not useful
plt.subplot(131)
plt.imshow(hst_psf_model)

#####################################################################
# 10th hdu is smaller psf
psf_small_hdu = hdul[9]
hst_psf_small_model = psf_small_hdu.data
hst_psf_small_header = psf_small_hdu.header
#print(psf_header) # header is not useful
plt.subplot(132)
plt.imshow(hst_psf_small_model)


#####################################################################
# 8th hdu is bspline model
bspline_hdu = hdul[7]
hst_bspline_model = bspline_hdu.data
hst_bspline_header = bspline_hdu.header
#print(bspline_header)
plt.subplot(133)
plt.imshow(hst_bspline_model)
plt.subplots_adjust(bottom=0.1, right=2, top=0.9)
plt.pause(1)
```



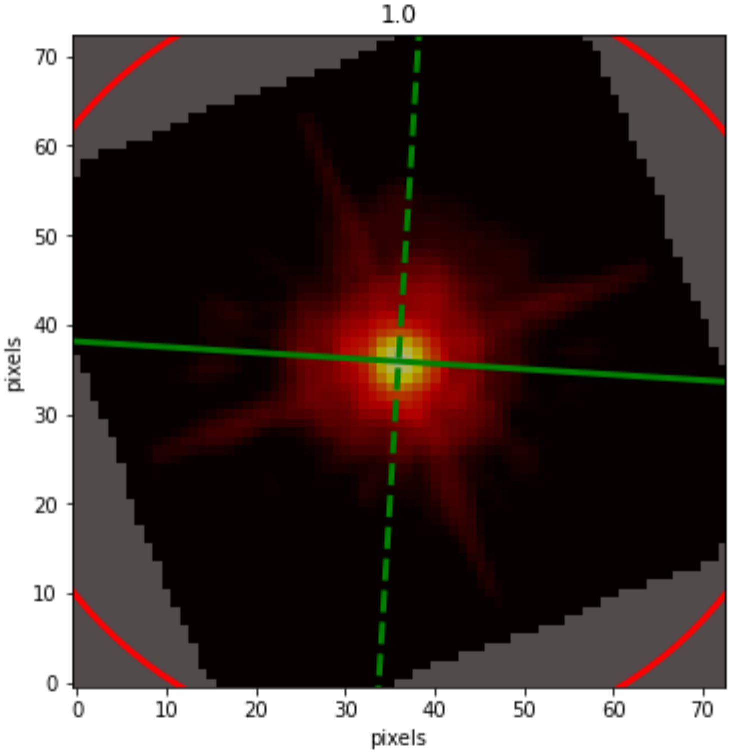
Fit the PSF model with Gaussian MGE.

In [5]:

```
# find the right fraction to use

#find_galaxy
#try_fractions_for_find_galaxy(hst_psf_model)
```

In [6]:

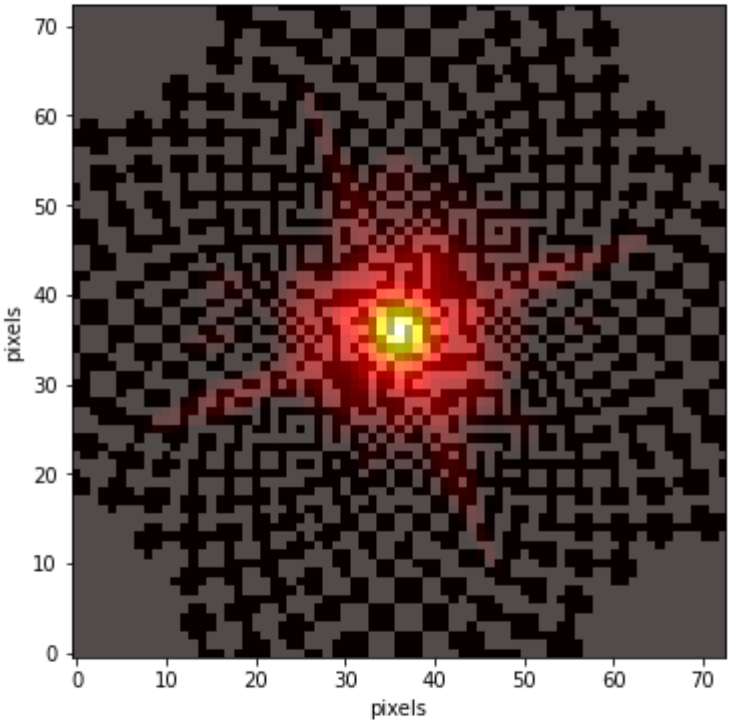
```
# Just use the whole image.
frac = 1.0
#

#################################################
# Model the central light ellipse

plt.clf()
#plt.clf()
f = find_galaxy(hst_psf_model, fraction=frac, plot=1, quiet=True)
eps = f.eps
theta = f.theta
cen_y = f.ypeak
cen_x = f.xpeak
plt.title(f'{frac}')
plt.pause(1)

print(eps, theta)
```



```
0.05042878760754688 3.5415987526017574
```

In [7]:

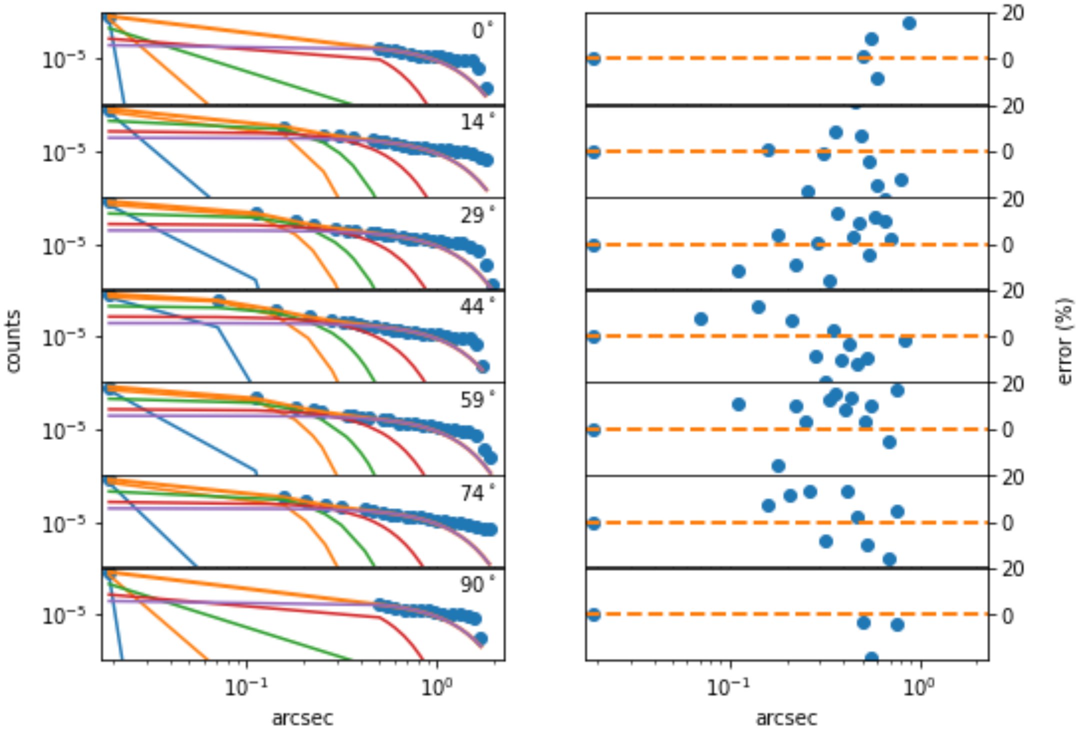
```
# It's basically circular, but I don't know how to do this as 1D.
# run sectors photometry
plt.clf()
s = sectors_photometry(hst_psf_model, eps, theta, cen_x, cen_y, plot=
plt.pause(1)  # Allow plot to appear on the screen
```



In [8]:

```
#####################################################################

# select number of gaussians to fit

scale = hst_scale
ngauss = 12

#####################################################################
# fit and plot

plt.clf()
m = mge_fit_sectors(s.radius, s.angle, s.counts, eps,
                    ngauss=ngauss, #sigmapsf=sigmapsf, #normpsf=normp
                    scale=scale, plot=1, bulge_disk=0, linear=0)
plt.pause(1)

#####################################################################
# take the output weights and sigmas for each Gaussian

hst_psf_weights = m.sol[0] # unnormalized weights in counts of each Ga
hst_normpsf = hst_psf_weights / np.sum(hst_psf_weights) # normalized 
hst_sigmapsf = m.sol[1] # sigma of each Gaussian

print('How good is the fit? Should be low (~ 0.02)... ' + str(m.absde
```

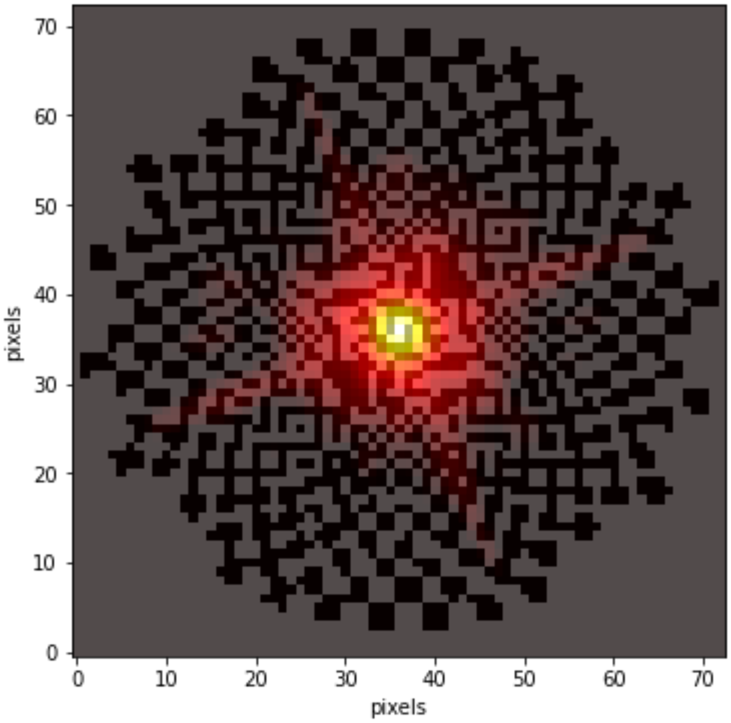
```
Iteration:1  chi2: 155.5  Nonzero: 7/12
Nonzero Gaussians: 6/12
Eliminating not useful Gaussians...
ngauss: 5          chi2: 134.2
Starting nonlinear fit...
Iteration:1  chi2: 134.2  Nonzero: 5/5
Nonzero Gaussians: 5/5
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 0.27 seconds
  Total Iterations:  10
 Nonzero Gaussians:  5
  Unused Gaussians:  7
 Sectors used in the fit:  19
 Total number of points fitted:  424
 Chi2: 134.1
 STDEV: 0.465
 MEANABSDEV: 0.4111
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  1.176437e-01     0.380000     0.932621
  5.616214e-01     0.966834     1.000000
  1.850400e-01      1.59761     1.000000
  6.106347e-02      3.22752     0.955569
  7.608733e-02      7.46306     1.000000
++++++++++++++++++++++++++++++++++++++++++++
```



```
How good is the fit? Should be low (~ 0.02)... 0.4111019369597713
```

Datapoints that drop off sharply at large radii worsen the fit and should be removed as skylevel.

In [9]:
```
#################################################
# Set the minlevel by eye for this one
minlevel = 2*1e-6

# run sectors photometry
plt.clf()
s = sectors_photometry(hst_psf_model, eps, theta, cen_x, cen_y, minle
plt.pause(1)  # Allow plot to appear on the screen
```

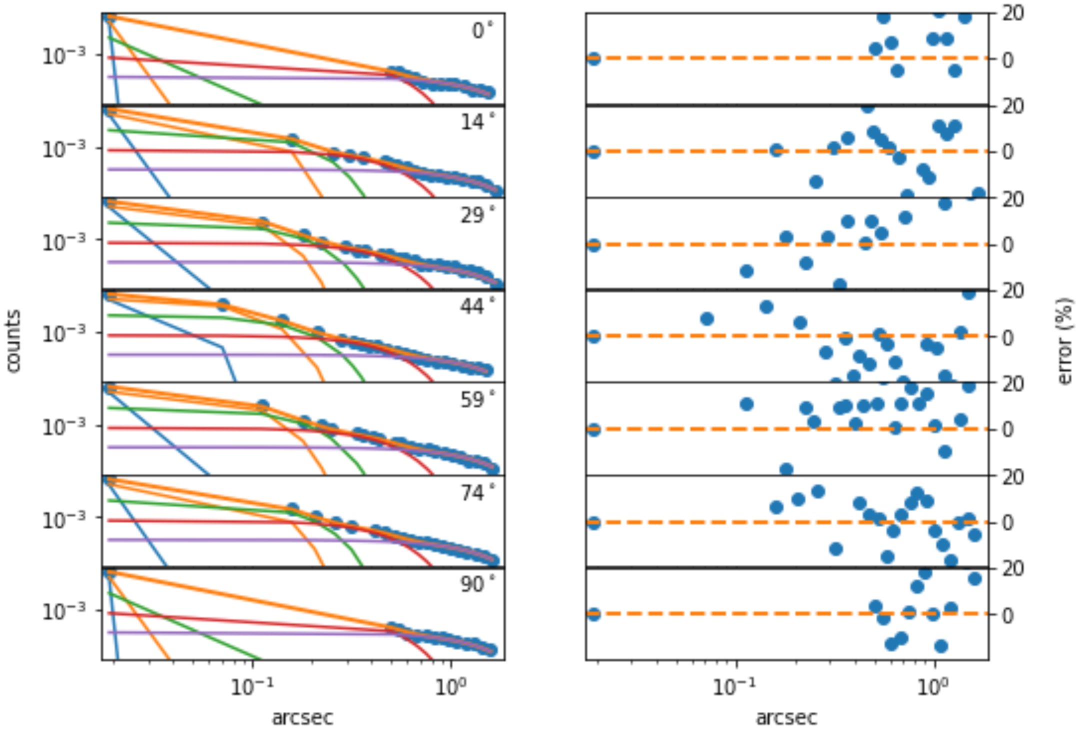


In [10]:

```
#######################################################################
# select number of gaussians to fit (max of 20... penalty in uncertai
ngauss = 12

# pixel scale is hst scale
scale = hst_scale

# set qbound to force Gaussians to be nearly circular
qbounds = [0.98, 1.0]


#######################################################################
# fit and plot

plt.clf()
m = mge_fit_sectors(s.radius, s.angle, s.counts, eps,
                    ngauss=ngauss, qbounds=qbounds,
                    scale=scale, plot=1, bulge_disk=0, linear=0)
plt.pause(1)

#######################################################################
# take the output weights and sigmas for each Gaussian

hst_psf_weights = m.sol[0] # unnormalized weights in counts of each G
hst_normpsf = hst_psf_weights / np.sum(hst_psf_weights) # normalized 
hst_sigmapsf = m.sol[1] # sigma of each Gaussian

print('How good is the fit? Should be low (~ 0.02)... ' + str(m.absde
```

```
Iteration:1  chi2: 12.20  Nonzero: 8/12
```

```
Iteration:11  chi2: 11.65  Nonzero: 6/12
Nonzero Gaussians: 6/12
Eliminating not useful Gaussians...
ngauss: 5          chi2: 11.64
Starting nonlinear fit...
Iteration:1  chi2: 11.64  Nonzero: 5/5
Nonzero Gaussians: 5/5
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 0.28 seconds
  Total Iterations:  15
 Nonzero Gaussians:  5
  Unused Gaussians:  7
 Sectors used in the fit:  19
 Total number of points fitted:  387
 Chi2: 11.62
 STDEV: 0.1706
 MEANABSDEV: 0.1303
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  1.219898e-01     0.380000     0.980000
  5.791940e-01     0.976113     1.000000
  1.847795e-01      1.72293     1.000000
  8.123488e-02      4.61518     0.980000
  5.516521e-02      14.2546     0.980000
++++++++++++++++++++++++++++++++++++++++++++
```



```
How good is the fit? Should be low (~ 0.02)... 0.1303200854422108
```

What about the KCWI PSF? The notes I took from our meeting are not clear.

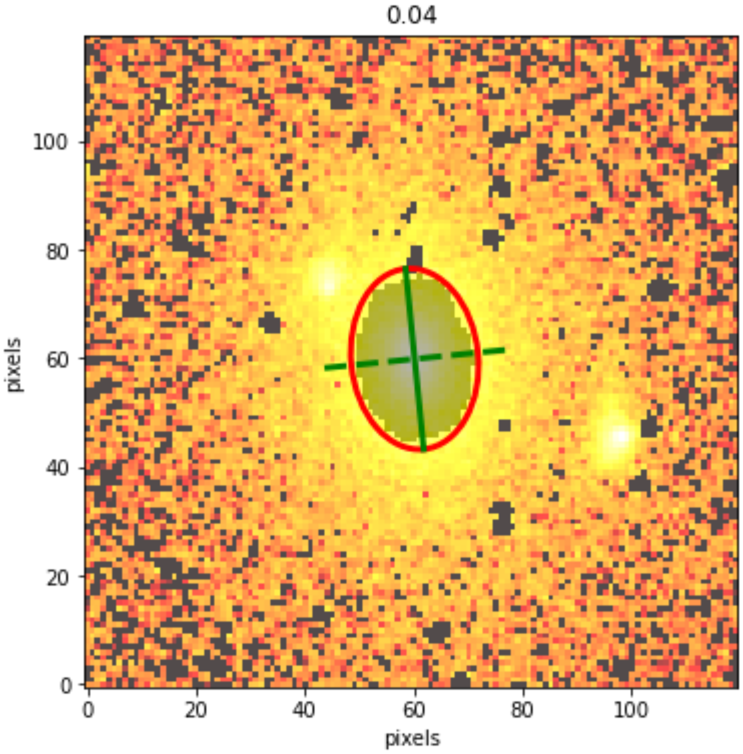
Now go ahead and fit the HST image at 3 arcsec

In [11]:

```
# take 3 arcsec hst image for find_galaxy initial estimates of PA and

img = hstF435_3arc_img

#####################################################################
# figure out the pixel fraction best to use

#try_fractions_for_find_galaxy(img)
```

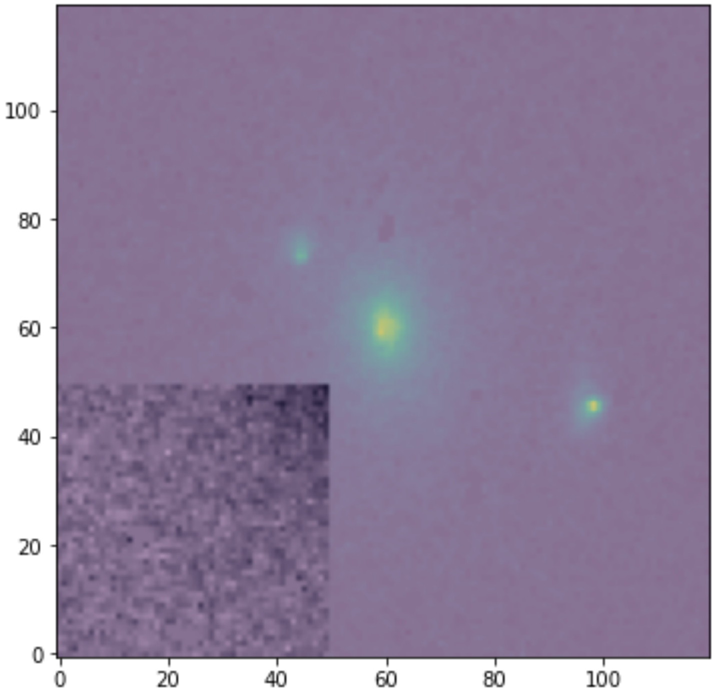
In [13]:

```
# set the fraction to be used as frac

frac = 0.04

#################################################
# Model the central light ellipse

plt.clf()
#plt.clf()
f = find_galaxy(img, fraction=frac, plot=1, quiet=True)
eps = f.eps
theta = f.theta
cen_y = f.ypeak
cen_x = f.xpeak
plt.title(f'{frac}')
plt.pause(1)
```



Get minlevel for this image by taking ~1/2 std of the background

In [14]:
```
# calculate the minimum level for inclusion in the photometry fitting

size=50
minlevel, noise = calculate_minlevel(img, size)
```



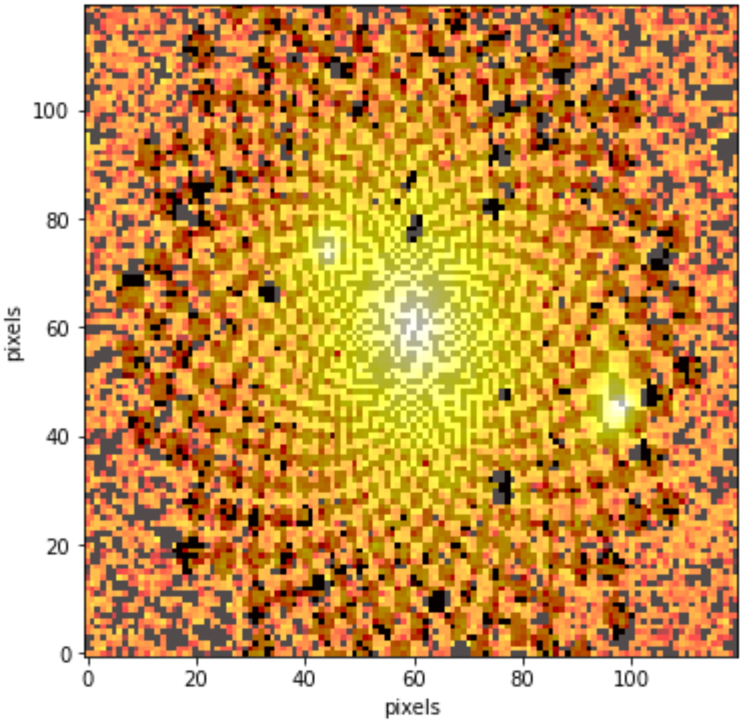
In [15]:

```
print(minlevel)
```

```
0.007076466293540728
```

In [16]:

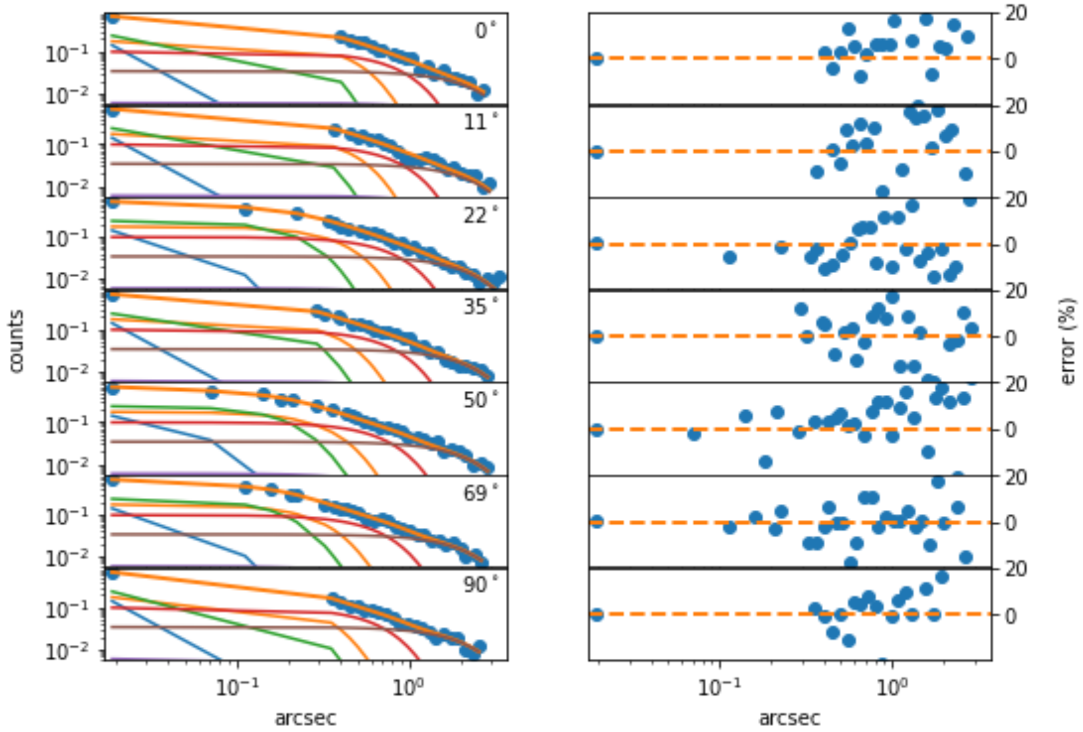
```
#################################################
# Perform galaxy photometry with full image

plt.clf()
s = sectors_photometry(img, eps, theta, cen_x, cen_y, minlevel=minlev
plt.pause(1)  # Allow plot to appear on the screen
```


0
20
40
60
80
100
0
20
40
60
80
100
pixels
pixels

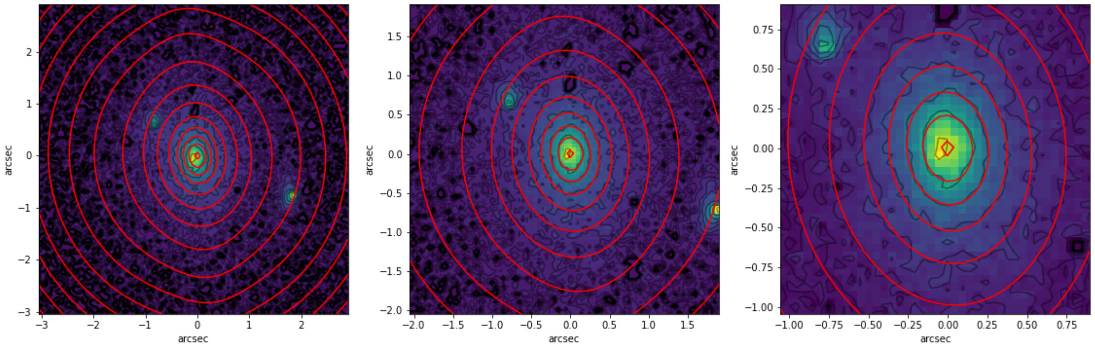
In [17]:

```
#####################################################################

# Do the actual MGE fit
# *********************** IMPORTANT *********************************
# For the final publication-quality MGE fit one should include the li
# "from mge_fit_sectors_regularized import mge_fit_sectors_regularize
# at the top of this file, rename mge_fit_sectors() into
# mge_fit_sectors_regularized() and re-run the procedure.
# See the documentation of mge_fit_sectors_regularized for details.
# *******************************************************************

# select number of gaussians to fit
ngauss = 12

# pixel scale
scale = hst_scale

# psf - take from the MGE psf model above
sigmapsf = hst_sigmapsf
normpsf = hst_normpsf
#seeing_fwhm = 0.1 # arcsec
#sigmapsf = seeing_fwhm / scale / 2.355 # pixels, 2.355 is fwhm/sigma

# exposure time
exp_time = hstF435_header['EXPTIME']

#####################################################################
# fit and plot

plt.clf()
m = mge_fit_sectors(s.radius, s.angle, s.counts, eps,
                    ngauss=ngauss, sigmapsf=sigmapsf, normpsf=normpsf
                    scale=scale, plot=1, bulge_disk=0, linear=0)
plt.pause(1)
```

```
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 7.600  Nonzero: 7/12
Iteration:21  chi2: 7.430  Nonzero: 7/12
Nonzero Gaussians: 7/12
Eliminating not useful Gaussians...
ngauss: 6          chi2: 7.432
Starting nonlinear fit...
Iteration:1  chi2: 7.432  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 1.16 seconds
  Total Iterations:  25
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.43
 STDEV: 0.1215
 MEANABSDEV: 0.08722
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.238904e-01     0.380000     0.692549
  1.575472e+01      3.29439     0.739303
  3.111699e+01      6.14274     0.645795
  7.288732e+01      12.0297     0.761856
  1.249031e+01      33.2094     0.289728
  2.202044e+02      33.2094     0.919459
++++++++++++++++++++++++++++++++++++++++++++
```



In [18]:
```
plot_contours_321 (img, f, m, sigmapsf, normpsf, contour_alpha=0.4, d
```



Looks good! Everything from here will have to be done with the mge_fit_sectors_regularized. Just do all the same with _regularized at the end.

In [21]:
```
#####################################################################

# Do the actual MGE fit
# *********************** IMPORTANT *********************************
# For the final publication-quality MGE fit one should include the li
# "from mge_fit_sectors_regularized import mge_fit_sectors_regularize
# at the top of this file, rename mge_fit_sectors() into
# mge_fit_sectors_regularized() and re-run the procedure.
# See the documentation of mge_fit_sectors_regularized for details.
# *******************************************************************

# select number of gaussians to fit
ngauss = 12

# pixel scale
scale = hst_scale

# psf - take from the MGE psf model above
sigmapsf = hst_sigmapsf
normpsf = hst_normpsf
#seeing_fwhm = 0.1 # arcsec
#sigmapsf = seeing_fwhm / scale / 2.355 # pixels, 2.355 is fwhm/sigma

# exposure time
exp_time = hstF435_header['EXPTIME']

#####################################################################
# fit and plot

plt.clf()
m = mge_fit_sectors_regularized(s.radius, s.angle, s.counts, eps,
                    ngauss=ngauss, sigmapsf=sigmapsf, normpsf=normpsf
                    scale=scale, plot=1, bulge_disk=0, linear=0)
plt.pause(1)

#####################################################################
# take the outputs and convert to the needed units

# convert sigma from pixels to arcsec and surface brightness to surfa
sigma, surf_density, q = convert_mge_model_outputs (m, exp_time, exti
```

```
Iteration:1  chi2: 21.44  Nonzero: 6/12
```

```
Iteration:11  chi2: 7.600  Nonzero: 7/12
Iteration:21  chi2: 7.430  Nonzero: 7/12
Nonzero Gaussians: 7/12
Eliminating not useful Gaussians...
ngauss: 6          chi2: 7.432
Starting nonlinear fit...
Iteration:1  chi2: 7.432  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 1.08 seconds
  Total Iterations:  25
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.43
 STDEV: 0.1215
 MEANABSDEV: 0.08722
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.238904e-01     0.380000     0.692549
  1.575472e+01      3.29439     0.739303
  3.111699e+01      6.14274     0.645795
  7.288732e+01      12.0297     0.761856
  1.249031e+01      33.2094     0.289728
  2.202044e+02      33.2094     0.919459
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.0500 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 8.052  Nonzero: 6/12
Iteration:21  chi2: 7.524  Nonzero: 7/12
Nonzero Gaussians: 6/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 7.430  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 1.43 seconds
  Total Iterations:  30
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.43
 STDEV: 0.1215
 MEANABSDEV: 0.08723
############################################
  Total_Counts  sigma_Pixels      q_obs
```

```
############################################
  9.241927e-01     0.380000     0.693090
  1.561661e+01      3.28515     0.740336
  3.118197e+01      6.12745     0.645845
  7.294655e+01      12.0250     0.761712
  1.251226e+01      33.2094     0.290343
  2.201929e+02      33.2094     0.919425
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.1000 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 7.600  Nonzero: 7/12
Iteration:21  chi2: 7.431  Nonzero: 6/12
Nonzero Gaussians: 7/12
Eliminating not useful Gaussians...
ngauss: 6          chi2: 7.433
Starting nonlinear fit...
Iteration:1  chi2: 7.433  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 1.18 seconds
  Total Iterations:  24
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.43
 STDEV: 0.1215
 MEANABSDEV: 0.08722
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.266909e-01     0.380000     0.692510
  1.575544e+01      3.29443     0.739349
  3.111529e+01      6.14362     0.645671
  7.288385e+01      12.0299     0.761748
  1.259858e+01      33.2094     0.291150
  2.201359e+02      33.2094     0.919996
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.1500 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 7.618  Nonzero: 7/12
Iteration:21  chi2: 7.434  Nonzero: 7/12
Iteration:31  chi2: 7.428  Nonzero: 7/12
Nonzero Gaussians: 7/12
Eliminating not useful Gaussians...
ngauss: 6          chi2: 7.433
Starting nonlinear fit...
Iteration:1  chi2: 7.433  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
```

```
All Gaussians are needed!
############################################
 Computation time: 1.81 seconds
  Total Iterations:  33
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.43
 STDEV: 0.1215
 MEANABSDEV: 0.08723
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.246685e-01    0.380000     0.692662
  1.544951e+01     3.27778     0.737448
  3.136838e+01     6.11085     0.647875
  7.295736e+01     12.0335     0.761250
  1.241453e+01     33.2094     0.288971
  2.202538e+02     33.2094     0.919248
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.2000 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 7.607  Nonzero: 7/12
Iteration:21  chi2: 7.432  Nonzero: 7/12
Nonzero Gaussians: 7/12
Eliminating not useful Gaussians...
ngauss: 6          chi2: 7.434
Starting nonlinear fit...
Iteration:1  chi2: 7.434  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 1.21 seconds
  Total Iterations:  24
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.43
 STDEV: 0.1215
 MEANABSDEV: 0.08723
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.341001e-01    0.380000     0.692225
  1.586052e+01     3.30256     0.739029
  3.103789e+01     6.15431     0.645421
  7.284907e+01     12.0301     0.761946
  1.256785e+01     33.2094     0.290699
  2.201527e+02     33.2094     0.919800
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.2500 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 7.601  Nonzero: 7/12
Iteration:21  chi2: 7.430  Nonzero: 7/12
Nonzero Gaussians: 7/12
Eliminating not useful Gaussians...
ngauss: 6          chi2: 7.434
Starting nonlinear fit...
Iteration:1  chi2: 7.434  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 1.08 seconds
  Total Iterations:  23
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.431
 STDEV: 0.1215
 MEANABSDEV: 0.08729
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.369236e-01     0.380000     0.692412
  1.577690e+01      3.29652     0.739441
  3.111275e+01      6.14896     0.645382
  7.282288e+01      12.0299     0.761143
  1.311848e+01      33.2094     0.300000
  2.197433e+02      33.2094     0.921676
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.3000 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 7.618  Nonzero: 7/12
Nonzero Gaussians: 5/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 7.601  Nonzero: 5/5
Nonzero Gaussians: 5/5
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 0.76 seconds
  Total Iterations:  20
 Nonzero Gaussians:  5
  Unused Gaussians:  7
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.6
 STDEV: 0.1228
 MEANABSDEV: 0.08834
```

```
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.350324e-01     0.380000     0.695615
  1.517946e+01      3.29567     0.722424
  3.137871e+01      5.98605     0.668806
  7.591470e+01      12.4498     0.711973
  2.270857e+02      33.2094     0.877186
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.3500 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 7.604  Nonzero: 6/12
Iteration:21  chi2: 7.585  Nonzero: 6/12
Iteration:31  chi2: 7.518  Nonzero: 6/12
Iteration:41  chi2: 7.456  Nonzero: 6/12
Nonzero Gaussians: 6/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 7.456  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 1.61 seconds
  Total Iterations:  41
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.456
 STDEV: 0.1217
 MEANABSDEV: 0.08781
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.453626e-01     0.380000     0.693408
  1.556517e+01      3.28529     0.740014
  3.100344e+01      6.11009     0.646802
  7.312341e+01      12.0494     0.749169
  2.027965e+01      33.2094     0.400000
  2.137084e+02      33.2094     0.944211
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.4000 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 7.577  Nonzero: 7/12
Nonzero Gaussians: 7/12
Eliminating not useful Gaussians...
ngauss: 6          chi2: 7.531
Starting nonlinear fit...
Iteration:1  chi2: 7.531  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
```

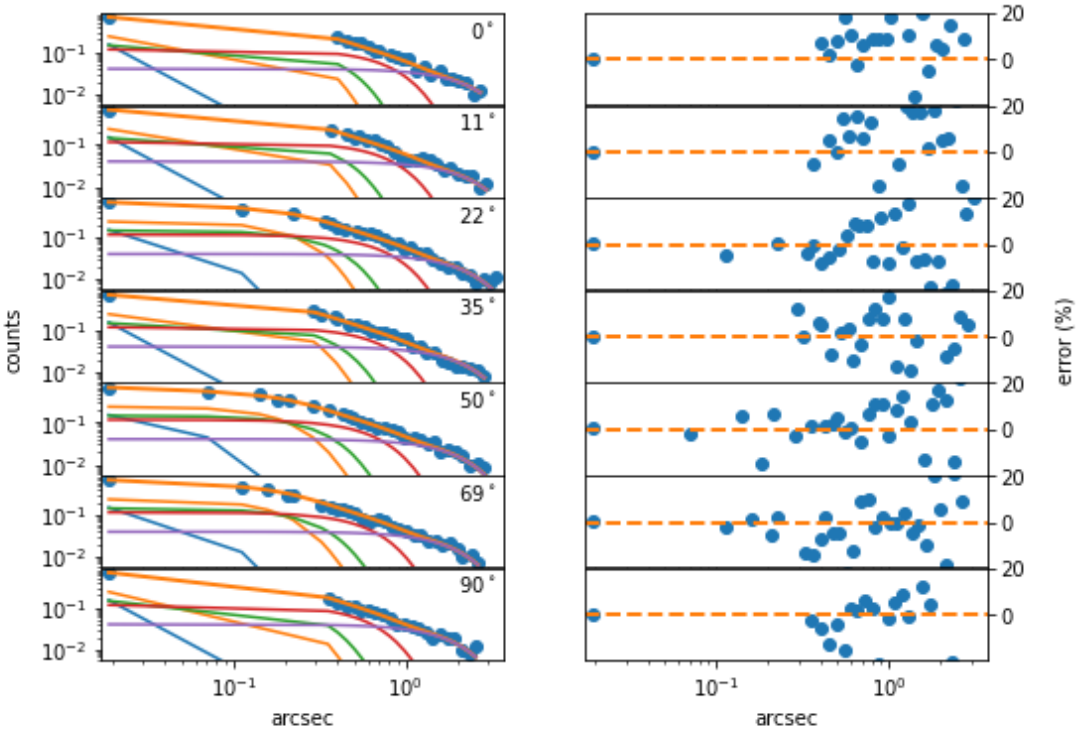
```
All Gaussians are needed!
############################################
 Computation time: 0.70 seconds
  Total Iterations:  19
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.529
 STDEV: 0.1223
 MEANABSDEV: 0.08774
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.558361e-01     0.380000     0.690950
  1.563872e+01      3.29474     0.743163
  2.937414e+01      6.06368     0.638737
  6.388248e+01      11.3112     0.786640
  1.745503e+01      19.9684     0.450000
  2.241089e+02      33.2094     0.896164
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.4500 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 7.614  Nonzero: 6/12
Iteration:21  chi2: 7.546  Nonzero: 6/12
Nonzero Gaussians: 6/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 7.546  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 0.94 seconds
  Total Iterations:  23
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.546
 STDEV: 0.1224
 MEANABSDEV: 0.08794
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.618019e-01     0.380000     0.692447
  1.522166e+01      3.27203     0.745869
  2.932410e+01      6.00215     0.639024
  5.876708e+01      11.0791     0.798215
  2.178094e+01      17.6804     0.500000
  2.250740e+02      33.2094     0.890584
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.5000 1.0000
```

```
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 8.190  Nonzero: 6/12
Iteration:21  chi2: 7.600  Nonzero: 7/12
Iteration:31  chi2: 7.559  Nonzero: 6/12
Nonzero Gaussians: 6/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 7.559  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 1.16 seconds
  Total Iterations:  31
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.559
 STDEV: 0.1225
 MEANABSDEV: 0.08809
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.705565e-01     0.380000     0.692143
  1.526647e+01      3.27838     0.745390
  2.912209e+01      5.99808     0.638897
  5.192010e+01      10.8360     0.810196
  2.804132e+01      16.1332     0.550000
  2.256348e+02      33.2094     0.887056
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.5500 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 7.601  Nonzero: 7/12
Iteration:21  chi2: 7.576  Nonzero: 6/12
Nonzero Gaussians: 6/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 7.571  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 1.19 seconds
  Total Iterations:  28
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.571
 STDEV: 0.1226
 MEANABSDEV: 0.08822
```

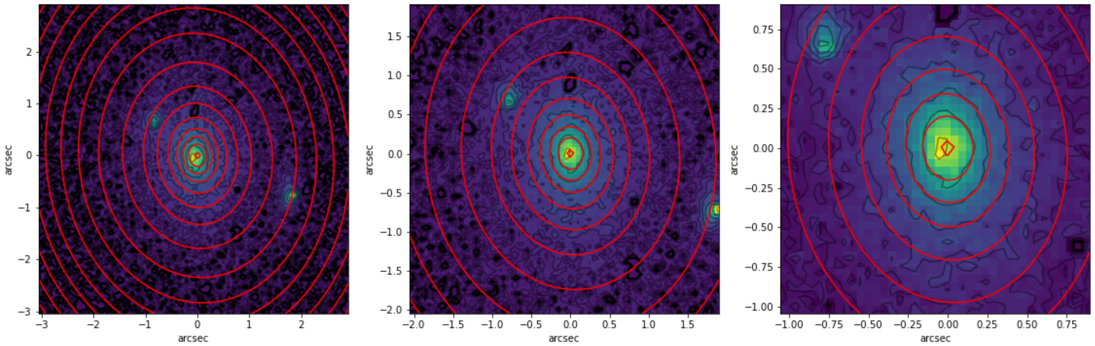
```
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.767159e-01     0.380000     0.692832
  1.505333e+01      3.26925     0.744351
  2.925318e+01      5.96912     0.641313
  4.223316e+01      10.5494     0.824029
  3.715976e+01      14.8335     0.600000
  2.261366e+02      33.2094     0.883935
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.6000 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 7.645  Nonzero: 6/12
Iteration:21  chi2: 7.589  Nonzero: 6/12
Nonzero Gaussians: 6/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 7.583  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 1.21 seconds
  Total Iterations:  27
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.583
 STDEV: 0.1227
 MEANABSDEV: 0.08833
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.809047e-01     0.380000     0.693269
  1.493058e+01      3.27113     0.736539
  2.987051e+01      5.95844     0.650000
  3.021743e+01      10.3564     0.825642
  4.819861e+01      13.8119     0.650000
  2.264659e+02      33.2094     0.881013
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.6500 1.0000
############################################
Iteration:1  chi2: 21.44  Nonzero: 6/12
Iteration:11  chi2: 19.31  Nonzero: 6/12
Iteration:21  chi2: 11.67  Nonzero: 5/12
Iteration:31  chi2: 8.337  Nonzero: 6/12
Iteration:41  chi2: 7.612  Nonzero: 6/12
Nonzero Gaussians: 6/12
Eliminating not useful Gaussians...
ngauss: 5          chi2: 7.612
Starting nonlinear fit...
Iteration:1  chi2: 7.612  Nonzero: 5/5
```

```
Nonzero Gaussians: 5/5
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 1.63 seconds
  Total Iterations:  41
 Nonzero Gaussians:  5
  Unused Gaussians:  7
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.611
 STDEV: 0.1229
 MEANABSDEV: 0.08844
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  9.902545e-01     0.380000     0.700000
  1.625653e+01      3.40079     0.700000
  3.124172e+01      6.00881     0.700000
  7.513424e+01      12.5884     0.704734
  2.269512e+02      33.2094     0.878347
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.7000 1.0000
############################################
Iteration:1  chi2: 17.38  Nonzero: 6/12
Iteration:11  chi2: 7.864  Nonzero: 7/12
Nonzero Gaussians: 6/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 7.811  Nonzero: 6/6
Nonzero Gaussians: 6/6
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 0.74 seconds
  Total Iterations:  20
 Nonzero Gaussians:  6
  Unused Gaussians:  6
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.811
 STDEV: 0.1245
 MEANABSDEV: 0.09153
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  1.055877e+00     0.380000     0.750000
  1.867290e+01      3.46366     0.750000
  2.740208e+01      5.88342     0.750000
  7.497311e+01      12.0065     0.750000
  1.297776e+02      33.2094     1.000000
  1.006785e+02      33.2094     0.750000
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.7500 1.0000
```

```
############################################
Iteration:1  chi2: 15.17  Nonzero: 6/12
Iteration:11  chi2: 8.592  Nonzero: 5/12
Nonzero Gaussians: 5/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 8.589  Nonzero: 5/5
Nonzero Gaussians: 5/5
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 0.64 seconds
  Total Iterations:  14
 Nonzero Gaussians:  5
  Unused Gaussians:  7
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 8.575
 STDEV: 0.1303
 MEANABSDEV: 0.09878
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  1.117806e+00     0.380000     0.800000
  2.173327e+01      3.52606     0.800000
  2.257135e+01      5.78136     0.800000
  7.583658e+01      11.4725     0.800000
  2.278123e+02      33.2094     0.861114
++++++++++++++++++++++++++++++++++++++++++++
(minloop) qbounds=0.8000 1.0000
############################################
Iteration:1  chi2: 17.38  Nonzero: 6/12
Iteration:11  chi2: 7.885  Nonzero: 5/12
Nonzero Gaussians: 5/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 7.874  Nonzero: 5/5
Nonzero Gaussians: 5/5
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 0.64 seconds
  Total Iterations:  16
 Nonzero Gaussians:  5
  Unused Gaussians:  7
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.874
 STDEV: 0.125
 MEANABSDEV: 0.0919
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  1.054501e+00     0.380000     0.750000
  1.838250e+01      3.45093     0.750000
  2.763896e+01      5.85545     0.750000
  7.558601e+01      12.0334     0.750000
  2.272962e+02      33.2094     0.870258
++++++++++++++++++++++++++++++++++++++++++++
(maxloop) qbounds=0.7500 0.9500
############################################
Iteration:1  chi2: 17.38  Nonzero: 6/12
Iteration:11  chi2: 7.884  Nonzero: 5/12
Nonzero Gaussians: 5/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 7.874  Nonzero: 5/5
Nonzero Gaussians: 5/5
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 0.62 seconds
  Total Iterations:  15
 Nonzero Gaussians:  5
  Unused Gaussians:  7
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.874
 STDEV: 0.125
 MEANABSDEV: 0.0919
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  1.052733e+00     0.380000     0.750000
  1.814040e+01      3.43931     0.750000
  2.783086e+01      5.83407     0.750000
  7.565234e+01      12.0335     0.750000
  2.272854e+02      33.2094     0.870319
++++++++++++++++++++++++++++++++++++++++++++
(maxloop) qbounds=0.7500 0.9000
############################################
Iteration:1  chi2: 17.38  Nonzero: 6/12
Iteration:11  chi2: 7.922  Nonzero: 5/12
Nonzero Gaussians: 5/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 7.911  Nonzero: 5/5
Nonzero Gaussians: 5/5
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 0.64 seconds
  Total Iterations:  16
 Nonzero Gaussians:  5
  Unused Gaussians:  7
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 7.911
 STDEV: 0.1253
 MEANABSDEV: 0.09225
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  1.058461e+00     0.380000     0.750000
  1.829224e+01      3.45895     0.750000
  2.621937e+01      5.74896     0.750000
  7.521277e+01      11.7907     0.750000
  2.273715e+02      33.2094     0.850000
++++++++++++++++++++++++++++++++++++++++++++
(maxloop) qbounds=0.7500 0.8500
############################################
Iteration:1  chi2: 17.38  Nonzero: 6/12
Iteration:11  chi2: 8.532  Nonzero: 4/12
Iteration:21  chi2: 8.401  Nonzero: 5/12
Nonzero Gaussians: 5/12
Eliminating not useful Gaussians...
Starting nonlinear fit...
Iteration:1  chi2: 8.401  Nonzero: 5/5
Nonzero Gaussians: 5/5
Eliminating not useful Gaussians...
All Gaussians are needed!
############################################
 Computation time: 0.89 seconds
  Total Iterations:  24
 Nonzero Gaussians:  5
  Unused Gaussians:  7
 Sectors used in the fit:  19
 Total number of points fitted:  496
 Chi2: 8.401
 STDEV: 0.129
 MEANABSDEV: 0.09454
############################################
  Total_Counts  sigma_Pixels      q_obs
############################################
  1.068813e+00     0.380000     0.750000
  1.731079e+01      3.45767     0.750000
  2.341815e+01      5.40175     0.750000
  7.452514e+01      11.2280     0.750000
  2.270038e+02      33.2094     0.800000
++++++++++++++++++++++++++++++++++++++++++++
(maxloop) qbounds=0.7500 0.8000
############################################
############################################
Final Regularized MGE Solution:
  Total_Counts  Sigma_Pixels      qObs
############################################
      1.06881         0.38         0.75
      17.3108      3.45767         0.75
      23.4182      5.40175         0.75
      74.5251       11.228         0.75
      227.004      33.2094          0.8
++++++++++++++++++++++++++++++++++++++++++++
Final qbounds=0.7500 0.8000
############################################
```



In [22]:
```
plot_contours_321 (img, f, m, sigmapsf, normpsf, contour_alpha=0.4, d
```



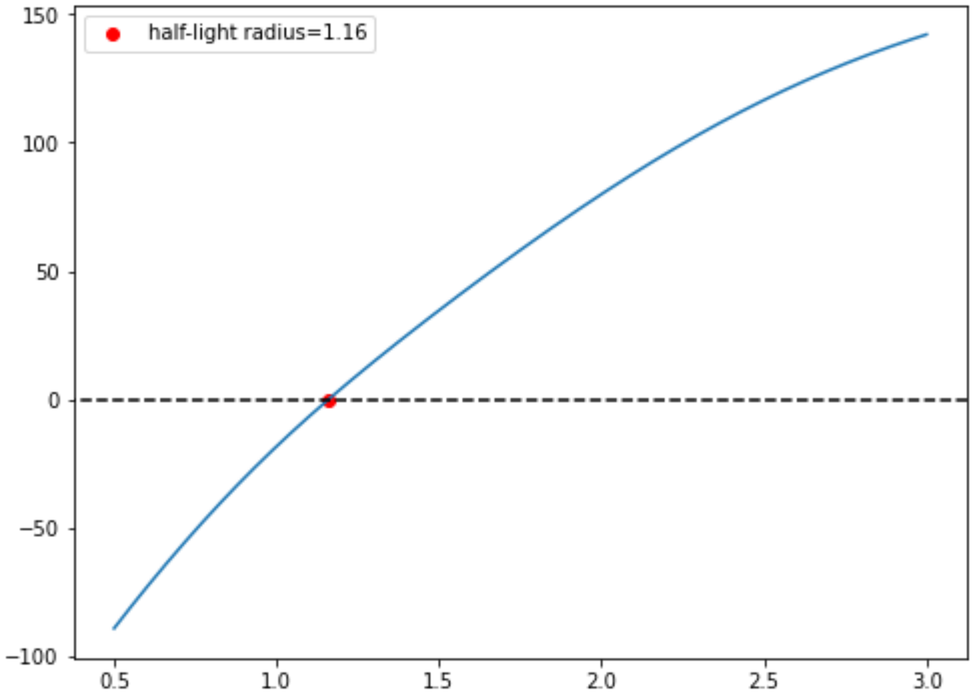
Need to get effective (half-light) radius from this... i.e. radius at which half the total light is enclosed. So... That will be where half the sum of the total counts of the Gaussian components are enclosed.

In [23]:

```
# take values from the mge expansion
lum_ks = m.sol[0]
lum_tot = np.sum(lum_ks) # total lumninosity is sum of the components
sigma_ks = m.sol[1] * hst_scale
q_ks = m.sol[2]
```

In [24]:

```
half_light_radius = find_half_light(lum_ks, sigma_ks, q_ks)
```



Michele has a routine to calculate the half light radius... I don't know why it wasn't showing up in my directory... :(

In [25]:

```
from jampy.mge_half_light_isophote import mge_half_light_isophote

half_light_radius_mich, _, _, _ = mge_half_light_isophote(surf_densit
print( (half_light_radius - half_light_radius_mich) / half_light_radi
```

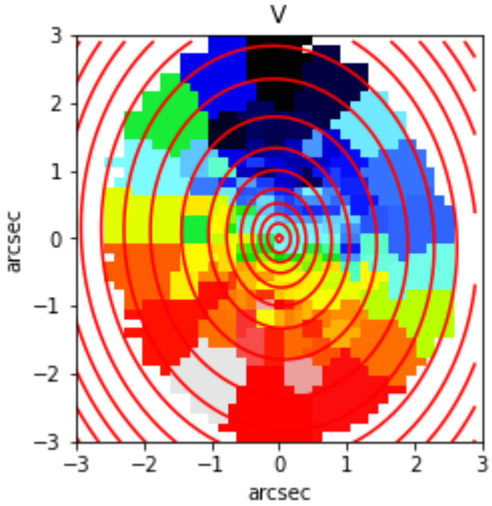
```
-0.0002151210009278633
```

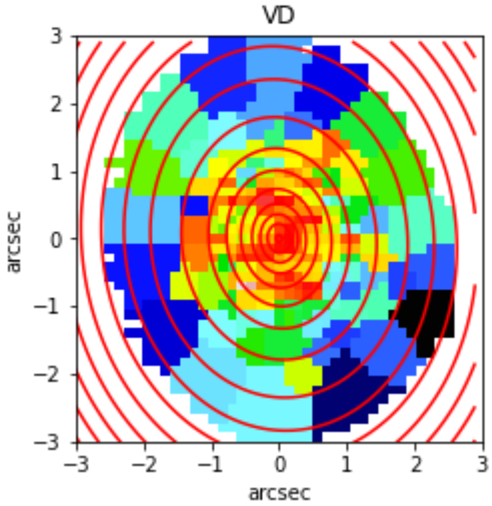
Okay, so my estimate is within 0.02% of Michele's

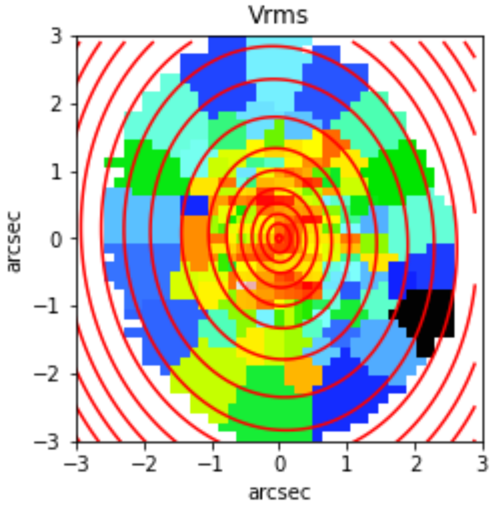
Look at kinematics.

In [26]:
```
from plotbin.sauron_colormap import register_sauron_colormap
```

In [27]:
```
V, VD, Vrms, dV, dVD, dVrms, Vbary, center_axis_index = load_2d_kinem
```





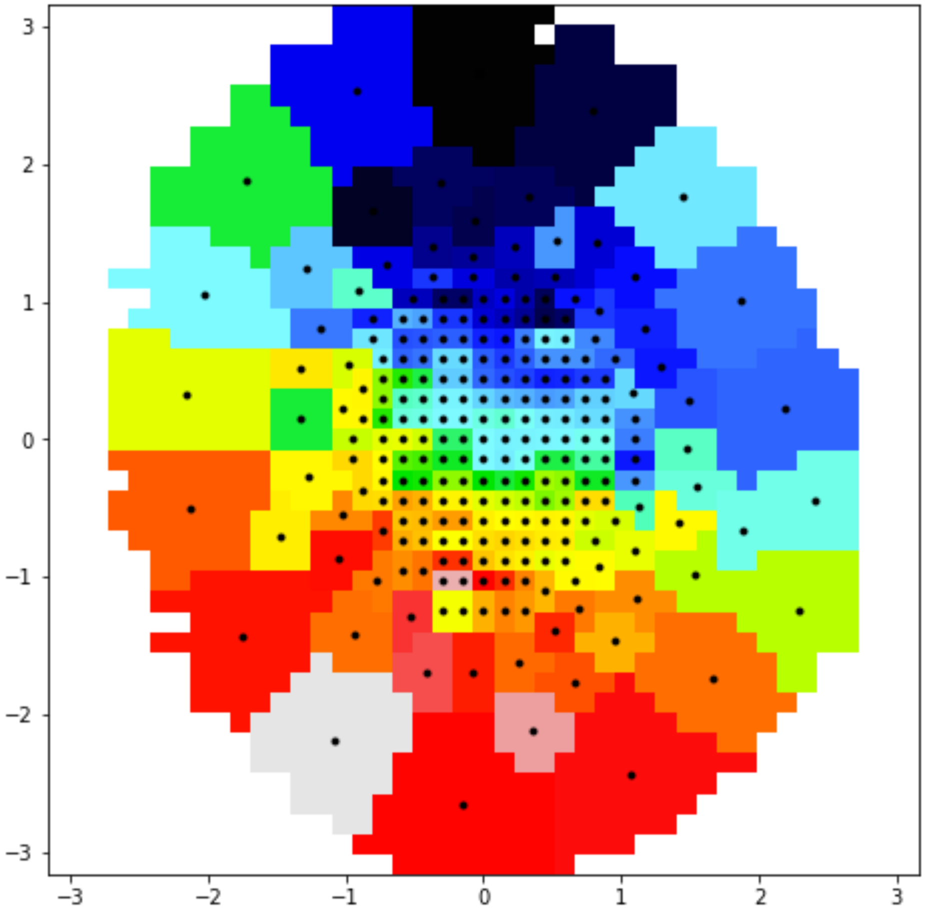


This is pretty, but we need it by bins, not by pixel.

In [28]:
```
V_bin, VD_bin, Vrms_bin, dV_bin, dVD_bin, dVrms_bin, xbin_arcsec, ybi
```

In [29]:
```
# Check binning

# plot with arcsec
width =  V.shape[0]/2 * kcwi_scale
extent = [-width,width,-width,width]
plt.figure(figsize=(8,8))
plt.imshow(V, origin='lower', extent=extent, cmap='sauron')
plt.scatter(xbin_arcsec, ybin_arcsec, color='k', marker='.')
```

Out[29]:
```
<matplotlib.collections.PathCollection at 0x7fba7a611ed0>
```

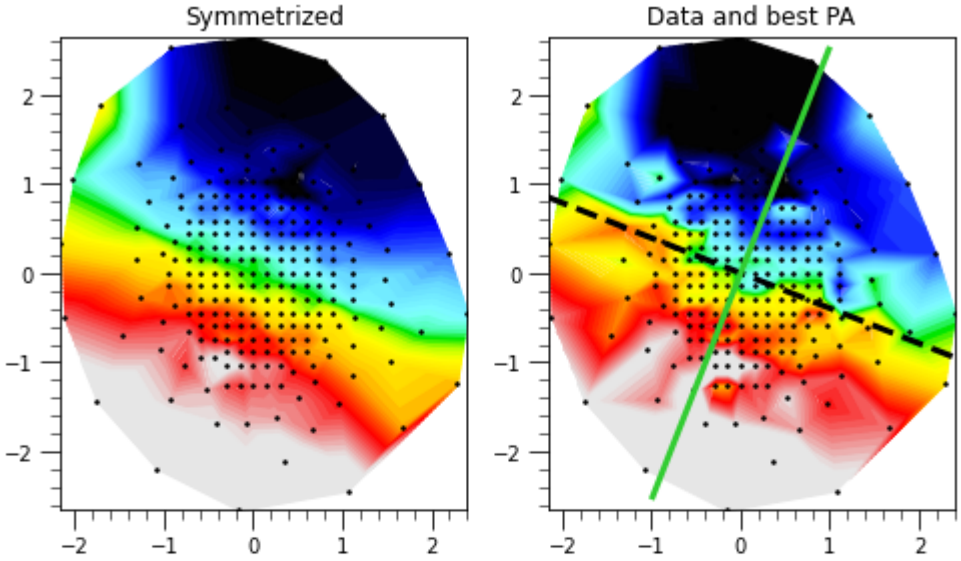


Determine correction to intrinsic (barycenter) velocity and kinematics PA with PAFit.

In [30]:
```
from pafit.fit_kinematic_pa import fit_kinematic_pa
```

In [31]:
```
PA_kin, dPA_kin, velocity_offset = fit_kinematic_pa(xbin_arcsec, ybin
```

```
  Kin PA: 158.5  +/-  10.2  (3*sigma error)
Velocity Offset: 8.84
```



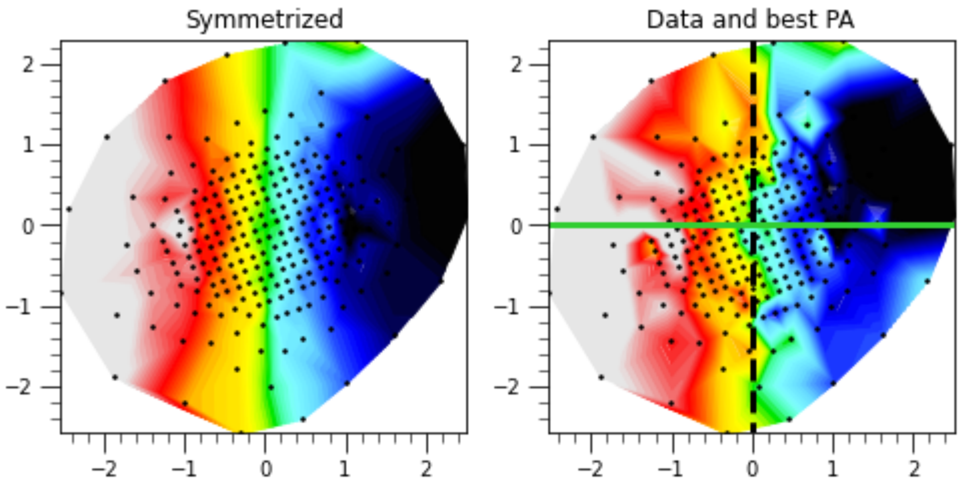
In [32]:
```
# Vbary_new = Vbary+correction
# V_new = V - correction
V_bin = V_bin - velocity_offset

# set kinematic PA from the negative x-axis
PA_kin = 270 - PA_kin

# PA_phot is theta from find_galaxy, measure from negative x-axis
PA_phot = theta
```

In [33]:
```
# Whoa, these are very different
print(PA_kin) # PA from kinematics
print(f.theta) # PA from photometry
```

```
111.5
84.27748713543393
```

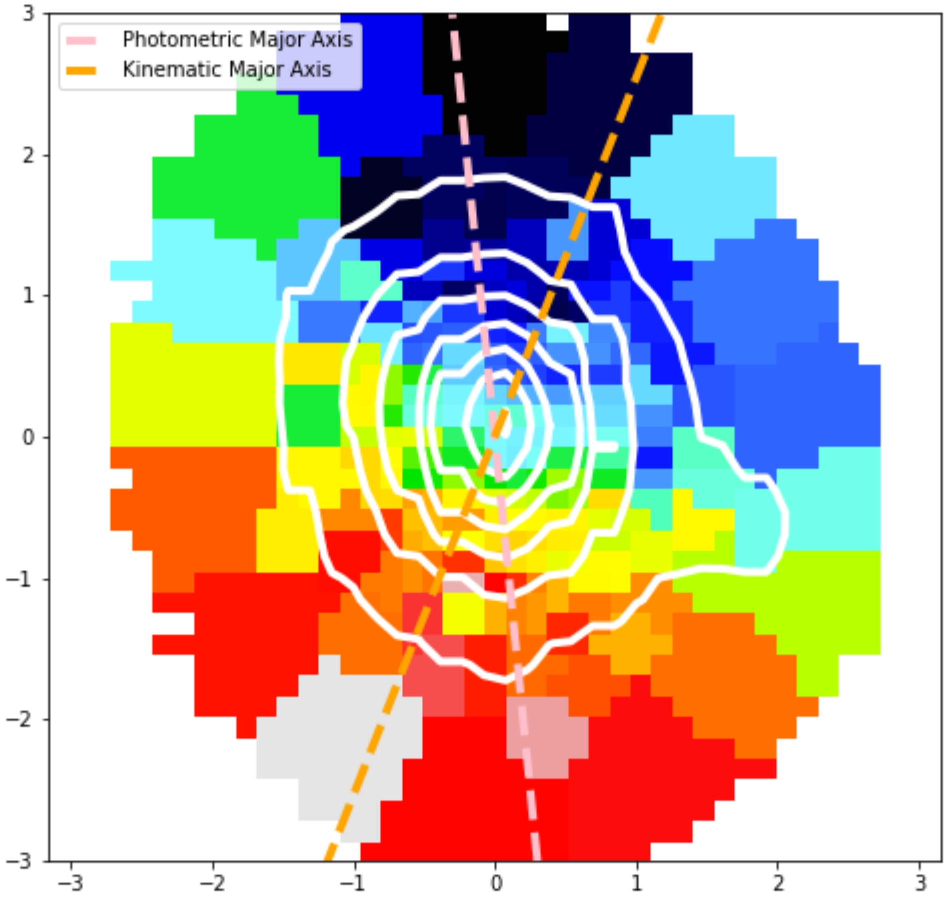
In [34]:
```
# rotate the bins by the PA from photomety # No, we should use the ki
# plot the rotation with the "non-symmetrized velocity field"

xbin, ybin = rotate_bins (PA_kin+180, xbin_arcsec, ybin_arcsec, V_bin
```

```
  Kin PA:  90.0  +/-   9.5  (3*sigma error)
Velocity Offset: 0.00
```



```
In [35]: # Look at photometric and kinematic major axis offsets

         # plot with arcsec
         width =  V.shape[0]/2 * kcwi_scale
         extent = [-width,width,-width,width]
         plt.figure(figsize=(8,8))
         plt.imshow(V, origin='lower', extent=extent, cmap='sauron')
         #plt.scatter(xbin_arcsec, ybin_arcsec, color='k', marker='.')
         plt.contour(kcwi_3arc_img,
                     extent=[-3,3,-3,3],
                     linewidths=4,
                     colors='white')

         # plot the major axes
         # photometric
         x = np.linspace(-3, 3, 1000)
         yph = -np.tan(np.radians(PA_phot))*x
         plt.plot(x,yph,
                  label='Photometric Major Axis',
                  c='pink',
                 linestyle='--',
                 linewidth=4)
         # kinematic
         ykin = -np.tan(np.radians(PA_kin))*x
         plt.plot(x,ykin,
                  label='Kinematic Major Axis',
                  c='orange',
                 linestyle='--',
                 linewidth=4)

         plt.ylim(-3,3)
         plt.legend()
```

```
Out[35]: <matplotlib.legend.Legend at 0x7fba7c827f10>
```



First try JAM with constant anisotropy (without radial dependece)

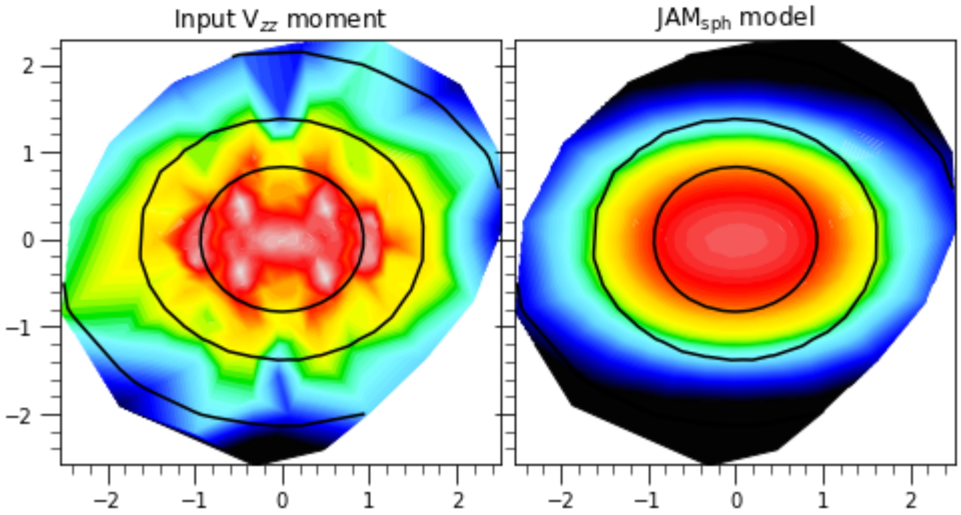
In [36]:

```
#############################################################
# JAM Parameters

#####################################################################

'''
What do I do for inclination?
'''
inc = 80 #np.arange(30,95,10)          # Assumed galaxy inclination

# take the surface density, etc from mge
surf = surf_density
sigma = sigma
qObs = q

'''
beta
'''
# constant beta - Do opsikov-merritt later
beta = np.full_like(surf, 0.2)

# redshift, convert to angular diameter dist in Mpc
z = 0.195
distance = cosmo.angular_diameter_distance(z).value

'''
What do I do for black hole mass? - According to CF, he thinks it pro
'''
mbh = 0# 1e8 # Black hole mass in solar masses # not sure what to do

# Below I assume mass follows light, but in a real application one
# will generally include a dark halo in surf_pot, sigma_pot, qobs_pot
# See e.g. Cappellari (2013) for an example
# https://ui.adsabs.harvard.edu/abs/2013MNRAS.432.1709C
surf_lum = surf_pot = surf
sigma_lum = sigma_pot = sigma
qobs_lum = qobs_pot = qObs

'''
PSF is wrong, should be done with MGE
'''
# kinematics sigmapsf
seeing_fwhm = 1.0 # arcsec, typical of KCWI small slicer https://www2
# pixel scale
sigmapsf = seeing_fwhm / 2.355

#normpsf = [0.7, 0.3]
'''
Is pixsize just the pixel scale?
'''
pixsize = kcwi_scale #0.8
goodbins = None
```

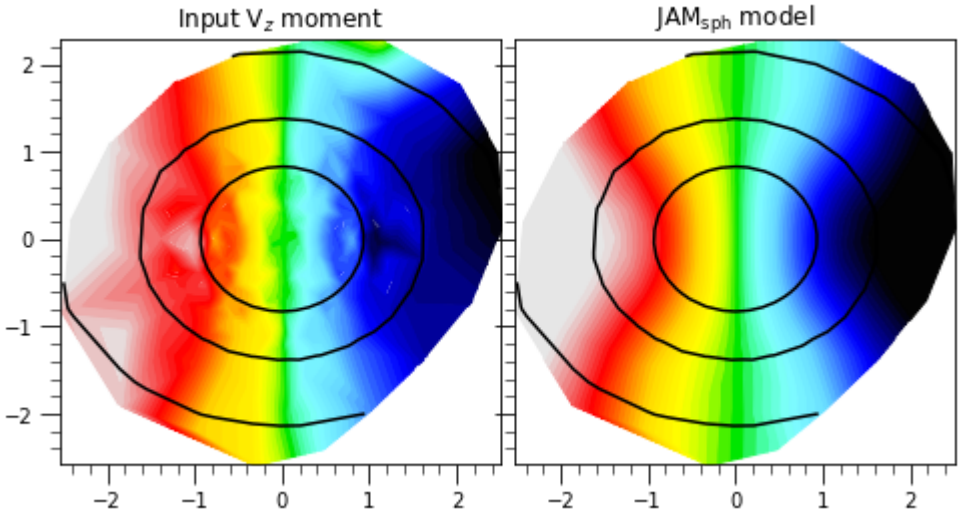
In [37]:

```
#############################################################
# It's time to JAM now!

# I use a loop below, just to higlight the fact that all parameters
# remain the same for the two JAM calls, except for 'moment' and 'dat
plt.figure(1)

for moment, data, errors in zip(['zz', 'z'], [Vrms_bin, V_bin], [dVrm

    print('####################################################')
    print('####################################################')
    print(f'Modeling moment {moment}')

    inc_rad = np.radians(inc)
    qintr_lum = qobs_lum**2 - np.cos(inc_rad)**2
    if np.any(qintr_lum <= 0):
        print('This inclination does not work')

    # The model is by design similar but not identical to the adopted
    m = jam_axi_proj(surf_lum, sigma_lum, qobs_lum, surf_pot, sigma_p
                     inc, mbh, distance, xbin, ybin, plot=True, data=
                     sigmapsf=sigmapsf, #normpsf=normpsf,
                     #flux_obs=flux_obs,
                     beta=beta, pixsize=pixsize,
                     moment=moment, goodbins=goodbins,
                     align='sph', ml=None, nodots=True)
    plt.pause(3)
    plt.figure(2)
    #surf_pot *= m.ml  # Scale the density by the best fitting M/L fr
    reduced_chi_squared = m.chi2
```

```
####################################################
####################################################
Modeling moment zz
jam_axi_proj_sph_zz (analytic_los=False) elapsed time sec: 3.26
inc=80.0; beta[0]=0.20; kappa=1.00; M/L=2.82e+04; BH=0.0; chi2/DOF=1.
89
Total mass MGE (MSun): 4.783e+11
```



```
#####################################################
#####################################################
Modeling moment z
jam_axi_proj_sph_z (analytic_los=False) elapsed time sec: 3.18
inc=80.0; beta[0]=0.20; kappa=-152.; M/L=1.00; BH=0.0; chi2/DOF=2.15
Total mass MGE (MSun): 1.696e+07
```



```
<Figure size 576x432 with 0 Axes>
```

JAM-time! Start with assigning beta_k to each Gaussian with Osipkov-Merritt profile

$$Beta(r) = \frac{r^2}{r_{ani}^2 + r^)} = \frac{1}{a_{ani}^2 (r_{eff}/r)^2 + 1}$$$$a_{ani} = r_{ani} / r_{eff}$$

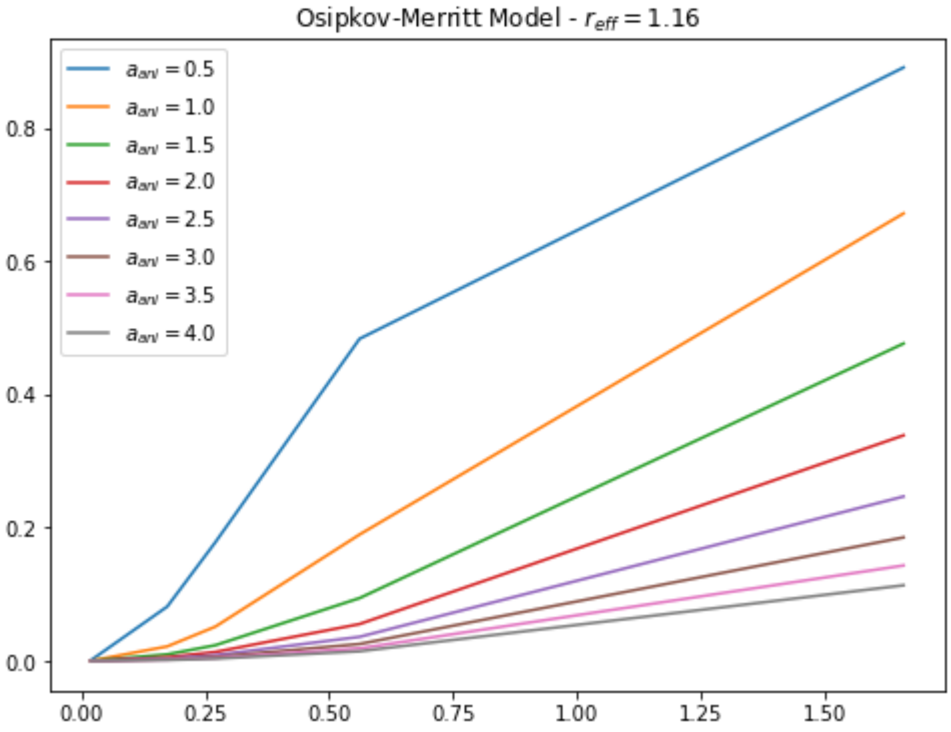
For r << r_ani, Beta ~ 0 For r >> r_ani, Beta ~ 1

In [38]:

```
# take a range of values for a_ani (from TDCOSMO IV something between
a_ani = np.arange(0.5, 4.5, 0.5)

# take half-light radius from MGE
#r_eff_V = 2.68 # arcsec
r_eff = half_light_radius

# take the sigma values for each Gaussian k for R
R = sigma # arcsec

# create array of Beta values for each Gaussian k
Beta = np.zeros(len(R))


for a_ani in a_ani:

    for i in range(len(R)):
        r = R[i]
        Beta[i] = osipkov_merritt_model(r, a_ani, r_eff)

    #plt.clf()
    plt.plot(R, Beta, label=r'$a_{ani}=$'+str(a_ani))

plt.legend()
plt.title('Osipkov-Merritt Model - ' r'$r_{eff}=$'+f'{np.around(r_eff
    #plt.pause(1)
```

Out[38]:

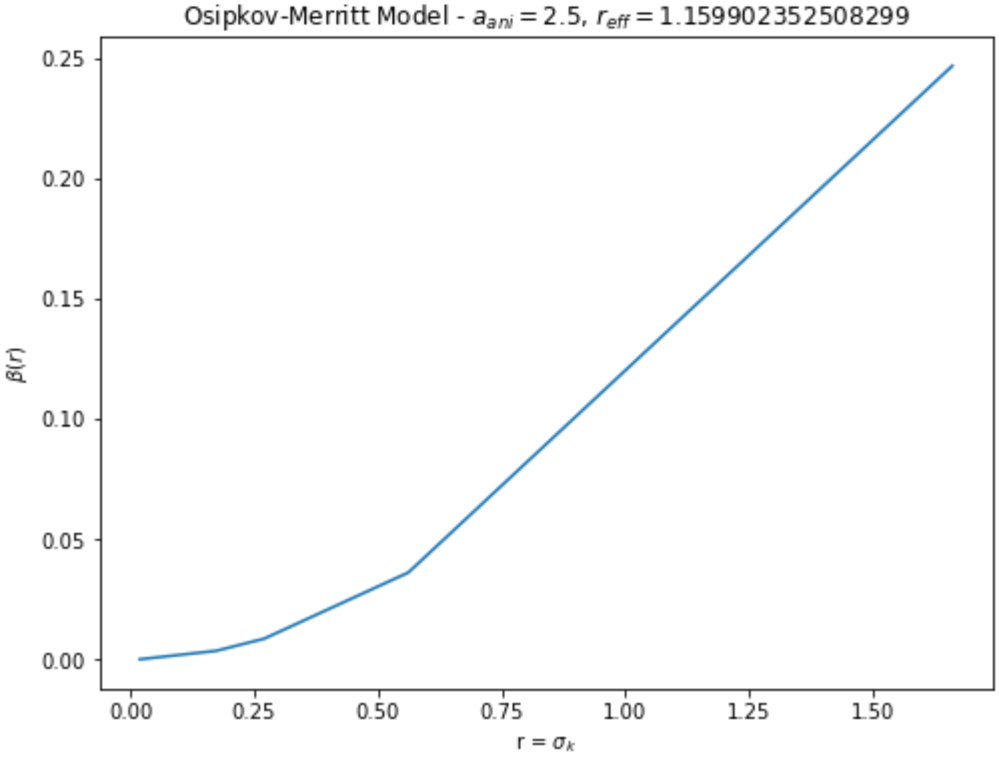
```
Text(0.5, 1.0, 'Osipkov-Merritt Model - $r_{eff}=$1.16')
```



Let's take a_ani = 1.5 for now

In [185…

```
# try different beta # TBD
#beta_list = [0.1,0.2,0.3,0.4,0.5]
#beta = np.full_like(surf, 0.2)

# anisotropy ratio 1.5
a_ani = 2.5

# take the V-band effective radius? What should I actually do?
#r_eff_V = 2.68 # arcsec
r_eff = half_light_radius

# take the sigma values for each Gaussian k for R
sigma # arcsec

# create array of Beta values for each Gaussian k
beta = np.zeros(len(sigma))

# calculate Beta at each sigma
for i in range(len(sigma)):
        r = sigma[i]
        beta[i] = osipkov_merritt_model(r, a_ani, r_eff)

plt.clf()
plt.plot(sigma, beta)
plt.xlabel(r'r = $\sigma_k$')
plt.ylabel(r'$\beta(r)$')
plt.title('Osipkov-Merritt Model - '+r'$a_{ani}=$'+f'{a_ani}, '+r'$r_
plt.pause(1)
```


Osipkov-Merritt Model - $a_{ani} = 2.5$, $r_{eff} = 1.159902352508299$
$\beta(r)$
0.25
0.20
0.15
0.10
0.05
0.00
0.00
0.25
0.50
0.75
1.00
1.25
1.50
r = σ_k

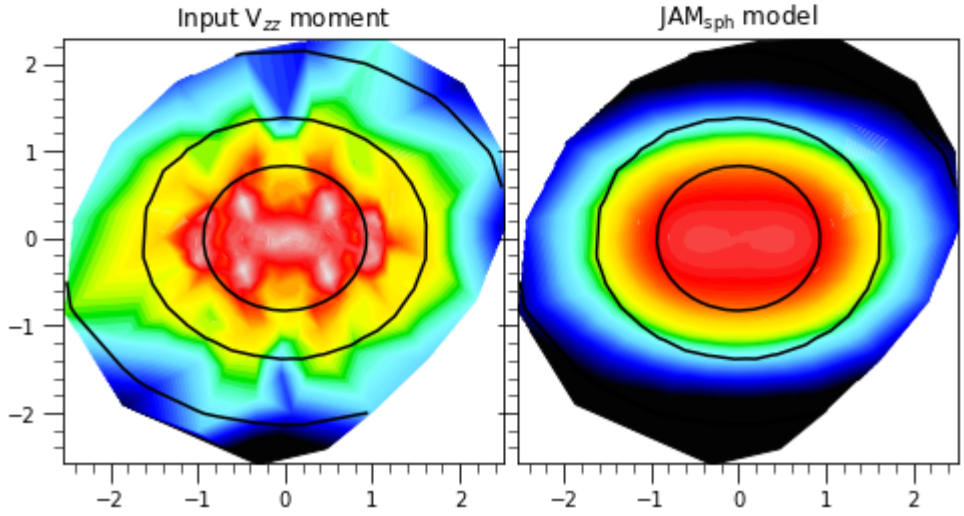
In [186…

```
#############################################################
# JAM Parameters

######################################################################


'''
What do I do for inclination?
'''
inc = 80 #np.arange(30,95,10)          # Assumed galaxy inclination

# take the surface density, etc from mge
surf = surf_density
sigma = sigma
qObs = q

# redshift, convert to angular diameter dist in Mpc
z = 0.195
distance = cosmo.angular_diameter_distance(z).value

'''
What do I do for black hole mass? - According to CF, he thinks it pro
'''
mbh = 0# 1e8 # Black hole mass in solar masses # not sure what to do 

# Below I assume mass follows light, but in a real application one
# will generally include a dark halo in surf_pot, sigma_pot, qobs_pot
# See e.g. Cappellari (2013) for an example
# https://ui.adsabs.harvard.edu/abs/2013MNRAS.432.1709C
surf_lum = surf_pot = surf
sigma_lum = sigma_pot = sigma
qobs_lum = qobs_pot = qObs

'''
PSF is wrong, should be done with MGE
'''
# kinematics sigmapsf
seeing_fwhm = 1.0 # arcsec, typical of KCWI small slicer https://www2
# pixel scale
sigmapsf = seeing_fwhm / 2.355

#normpsf = [0.7, 0.3]
'''
Is pixsize just the pixel scale?
'''
pixsize = kcwi_scale #0.8
goodbins = None
```

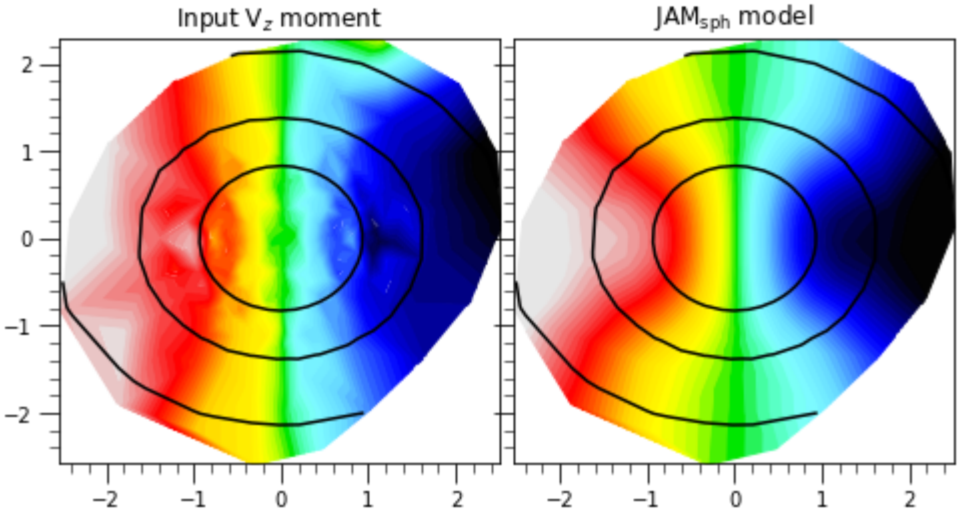
In [187…

```
#############################################################
# It's time to JAM now!

# I use a loop below, just to higlight the fact that all parameters
# remain the same for the two JAM calls, except for 'moment' and 'dat
plt.figure(1)

for moment, data, errors in zip(['zz', 'z'], [Vrms_bin, V_bin], [dVrm

    print('####################################################')
    print('####################################################')
    print(f'Modeling moment {moment}')

    inc_rad = np.radians(inc)
    qintr_lum = qobs_lum**2 - np.cos(inc_rad)**2
    if np.any(qintr_lum <= 0):
        print('This inclination does not work')

    # The model is by design similar but not identical to the adopted
    m = jam_axi_proj(surf_lum, sigma_lum, qobs_lum, surf_pot, sigma_p
                     inc, mbh, distance, xbin, ybin, plot=True, data=
                     sigmapsf=sigmapsf, #normpsf=normpsf,
                     beta=beta, pixsize=pixsize,
                     moment=moment, goodbins=goodbins,
                     align='sph', ml=None, nodots=True)
    plt.pause(3)
    plt.figure(2)
    #surf_pot *= m.ml  # Scale the density by the best fitting M/L fr
    reduced_chi_squared = m.chi2
```

```
####################################################
####################################################
Modeling moment zz
jam_axi_proj_sph_zz (analytic_los=False) elapsed time sec: 2.78
inc=80.0; beta[0]=4.3e-05; kappa=1.00; M/L=649.; BH=0.0; chi2/DOF=1.9
6
Total mass MGE (MSun): 4.760e+11
```



```
####################################################
####################################################
Modeling moment z
jam_axi_proj_sph_z (analytic_los=False) elapsed time sec: 2.75
inc=80.0; beta[0]=4.3e-05; kappa=-22.3; M/L=1.00; BH=0.0; chi2/DOF=2.
34
Total mass MGE (MSun): 7.330e+08
```



```
<Figure size 576x432 with 0 Axes>
```

```
In [ ]:
```