Import Statements

In [1]:
```
import custom_cmap
import os
import sys
import numpy as np
import matplotlib.pyplot as plt
from astropy.io import fits
from astropy.visualization import PercentileInterval
from PIL import Image
from IPython.display import display
import matplotlib.pyplot as plt
import matplotlib as mpl
import pandas as pd
from moria import reduce
from pathlib import Path
from astropy.visualization import PercentileInterval

from astropy.io import fits
from astropy.visualization import LogStretch, ImageNormalize
import plotly.express as px
import numpy as np
from astropy.visualization import ImageNormalize, AsinhStretch, SqrtStretch, LogStretch, PowerStretch

plt.rcParams.update({'font.size':25})
plt.rc('text', usetex=True)
%config InlineBackend.figure_format = 'retina'
%load_ext autoreload
%autoreload 2
```

Understanding the main directory.

The data directory should be organized as follows. You can look at the sample data folder to understand the directory strucutre you need.

00.DATA
01.XYM
02.CMD
03.LOC_TRANS
04.PSF_EXTRACT
05.COORD_TRANS(OPTIONAL)
06.FIT
07.CALIBRATION
All the necessary scripts are included in the sample data directory. The simplest way to run MORIA on your target is to copy the entirety of the "data" folder here, put your exposures in "data/00.DATA", then beginning with the demo.

The target star is modeled using one-star, two-star, or optionally three-star PSF fitting routines to account for the source star, lens star, and any nearby companion stars. Pixels surrounding the target are fit using these PSF models with a Markov Chain Monte Carlo (MCMC) algorithm, which solves for stellar positions and flux fractions.

NOTE: chmod +x program.src is a useful command to use whenever permissions are denied for a script.

In [2]:
```
directory = os.getcwd()
```

Step 0

We assume you ran the output_stacks.ipynbm, cmd_diagram.ipynb, creating_psf.ipynb (in that order) notebooks succesfully.

Step 1

The goal of this penultimate step is to fit the target star pixels using the constructed PSF model in order to determine the best-fit one-star or two-star model.

Optionally, a lens–source separation constraint derived from the Keck analysis can also be applied during the fitting process.

We will begin by performing this analysis for the F814W filter.

We will now design an input file that helps us run the 1star and 2star fit for F814W.

In [8]:
```
reduce.hst_fit_dataprep_onestar(directory) # Input for 1star-fit F814W filter
```

```
File 'IN.uvp2tri_NOscon_fs_asym_mcmc' successfully created.
```

In [9]:
```
reduce.hst_fit_dataprep_twostar(directory) # Input for 2star-fit F814W filter
```

```
File 'IN.uvp2tri_NOscon_fs_asym_mcmc' successfully created.
```

In [10]:
```
reduce.hst_fit_final_F814W(directory) # Run the tri fitting
```

```
File 'IN.uvp2tri_NOscon_fs_asym_mcmc' successfully created.
File 'IN.uvp2tri_NOscon_fs_asym_mcmc' successfully created.
File 'IN.uvp2tri_NOscon_fs_asym_mcmc' successfully created.
File 'IN.uvp2tri_NOscon_fs_asym_mcmc' successfully created.
```

Step 2

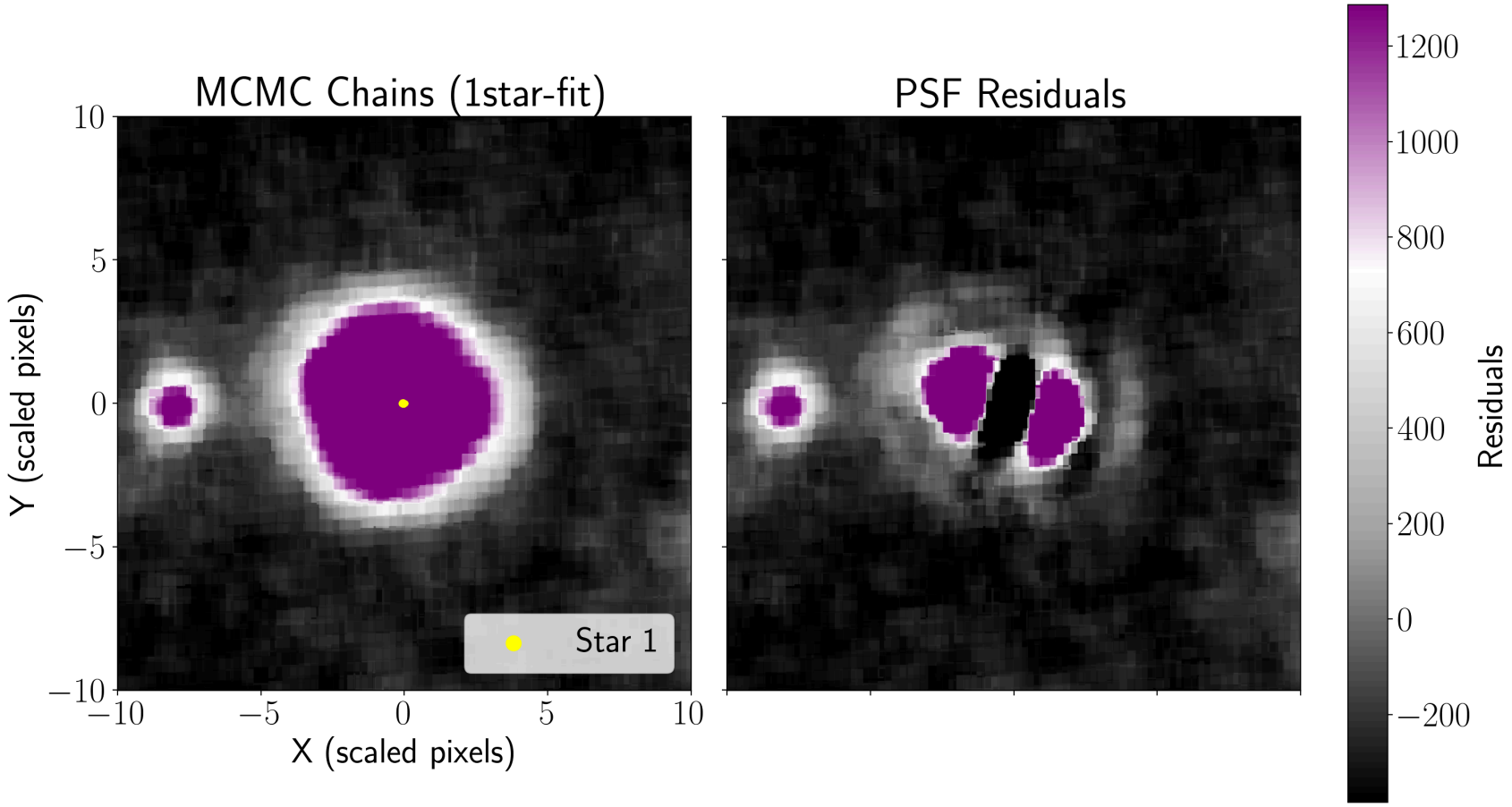
Inspect the residual images for the one-star fit and two-star fit models. The fitting results are stored in 06.FIT/F814W for the F814W filter and in 06.FIT/F606W for the F606W filter.

The coordinate system shown below corresponds to the "outputq" reference frame, where the target star is centered at (0, 0).

Pixel coordinates are scaled relative to this reference frame. If you choose to perform a three-star fit, the results from the one-star and two-star fits can be used to approximately construct the corresponding IN.* input file.

In [14]:
```
################
#OPEN FITS FILE#
################
fit_file = Path(directory).resolve()/f"06.FIT/F814W/1star-fit/uvp2tri_scon_fsky_I_KeckNOcon.06.pix_show.fi
fname = Path(directory).resolve()/f"06.FIT/F814W/1star-fit/expanded_mcmc.txt"


##################
#READ MCMC CHAINS#
##################
xtarg = 3001
```

```
ytarg = 1001
colname1 = ['X1_CENTER', 'Y1_CENTER']
chains1 = pd.read_table(fname, usecols=[0,1], sep=r'\s+', skiprows=1, names=colname1).head(100)

################
#SCALE THE DATE#
################
hdul = fits.open(fit_file)
data = hdul[-1].data
hdul.close()
interval = PercentileInterval(90)
scaled = interval(data)


#######################################
#PLOT THE MCMC CHAINS  + PSF RESIDUALS#
#######################################
# image dimensions
ny, nx = scaled.shape
# scaled physical coordinates
x_extent = ((0 - xtarg)/100, (nx - xtarg)/100)
y_extent = ((0 - ytarg)/100, (ny - ytarg)/100)
fig, ax = plt.subplots(1,2, figsize=(15, 8), constrained_layout=True)
im2 = ax[0].imshow(data, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=
im = ax[0].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cma
ax[0].scatter(chains1['X1_CENTER'], chains1['Y1_CENTER'], s=12, color='yellow', label='Star 1')
ax[0].legend(loc='lower right', markerscale=3)
cbar = fig.colorbar(im2, ax=[ax[0], ax[1]], fraction=0.046, pad=0.04)
cbar.set_label('Residuals')
ax[0].set_title("MCMC Chains (1star-fit)")
ax[0].set_xlabel("X (scaled pixels)")
ax[0].set_ylabel("Y (scaled pixels)")
ax[0].set_xlim(-10, 10)
ax[1].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=cus
ax[1].set_xlim(-30, -10)
ax[1].set_title("PSF Residuals")
ax[1].set_xticklabels([])
ax[1].set_yticklabels([])
plt.show()
```



The panel on the left is the target in the HST images. It is not the PSF!

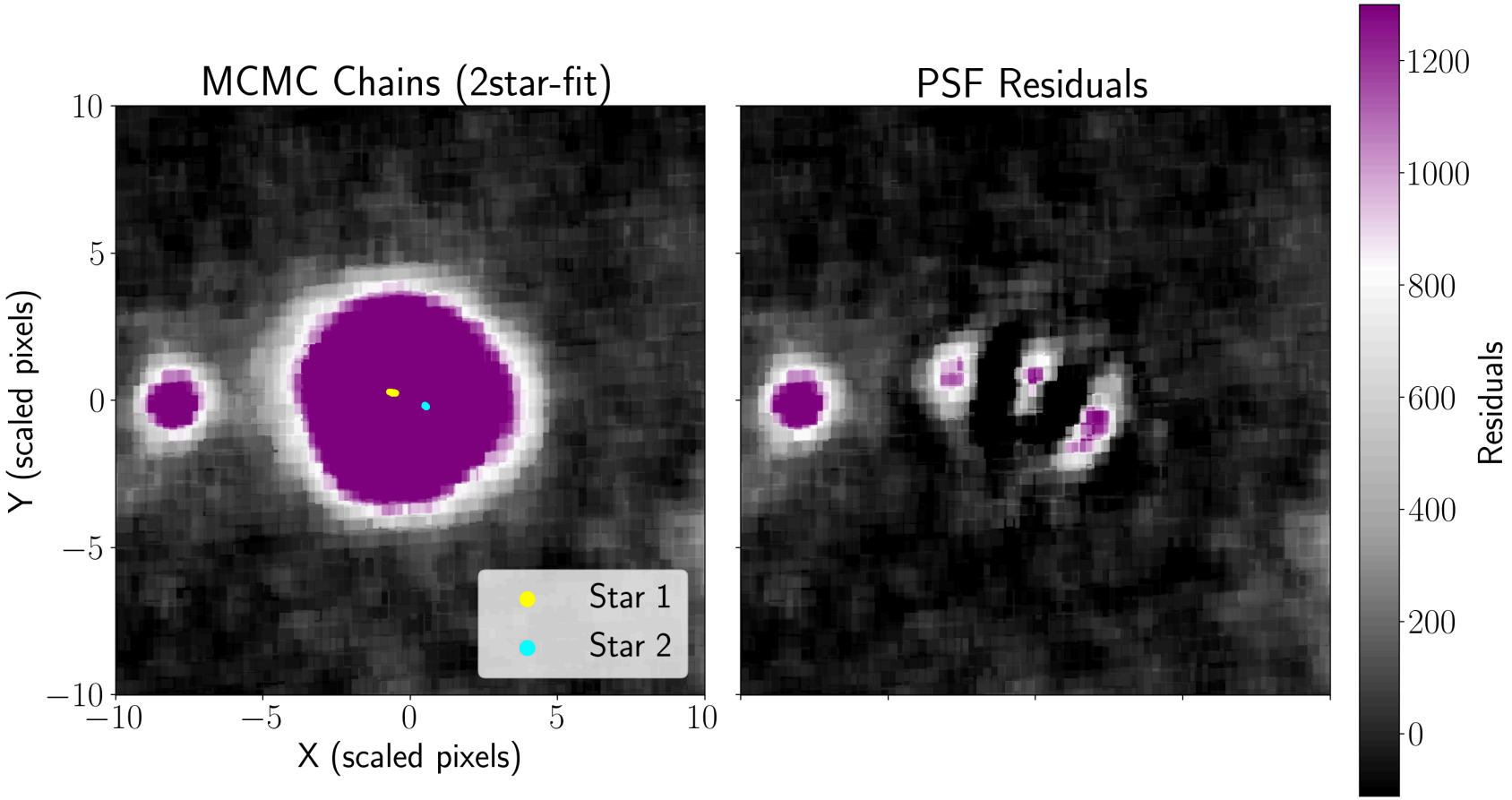
Using the PSF we generated in "creating_psf.ipynb", we try fitting a single PSF on the target. The right panel shows the result of subtracting the PSF from the target. For the 1star PSF fit, we clearly still see big residuals. Ideally, if the residual is smooth on the right panel, the PSF was "well-subtracted" from the target.

Therefore, looking at the dipole structure for the 1star PSF fitting result above, it is clear that running a 2star PSF fitting (or even a 3star PSF fitting) is a good idea!

In [18]:
```
################
#OPEN FITS FILE#
################
```

```
fit_file = Path(directory).resolve()/f"06.FIT/F814W/2star-fit/uvp2tri_scon_fsky_I_KeckNOcon.06.pix_show.fi
fname = Path(directory).resolve()/f"06.FIT/F814W/2star-fit/expanded_mcmc.txt"


##################
#READ MCMC CHAINS#
##################
xtarg = 3001
ytarg = 1001
colname1 = ['X1_CENTER', 'Y1_CENTER']
colname2 = ['X2_CENTER', 'Y2_CENTER']
chains1 = pd.read_table(fname, usecols=[0,1], sep=r'\s+', skiprows=1, names=colname1).head(100)
chains2 = pd.read_table(fname, usecols=[2,3], sep=r'\s+', skiprows=1, names=colname2).head(100)


################
#SCALE THE DATE#
################
hdul = fits.open(fit_file)
data = hdul[-1].data
hdul.close()
interval = PercentileInterval(90)
scaled = interval(data)


#######################################
#PLOT THE MCMC CHAINS  + PSF RESIDUALS#
#######################################
# image dimensions
ny, nx = scaled.shape
# scaled physical coordinates
x_extent = ((0 - xtarg)/100, (nx - xtarg)/100)
y_extent = ((0 - ytarg)/100, (ny - ytarg)/100)
fig, ax = plt.subplots(1,2, figsize=(15, 8), constrained_layout=True)
im2 = ax[0].imshow(data, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=
im = ax[0].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cma
ax[0].scatter(chains1['X1_CENTER'], chains1['Y1_CENTER'], s=12, color='yellow', label='Star 1')
ax[0].scatter(chains2['X2_CENTER'], chains2['Y2_CENTER'], s=12, color='cyan', label='Star 2')
ax[0].legend(loc='lower right', markerscale=3)
cbar = fig.colorbar(im2, ax=ax[1])
cbar.set_label('Residuals')
ax[0].set_title("MCMC Chains (2star-fit)")
```

```
ax[0].set_xlabel("X (scaled pixels)")
ax[0].set_ylabel("Y (scaled pixels)")
ax[0].set_xlim(-10, 10)
ax[1].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=cus
ax[1].set_xlim(-30, -10)
ax[1].set_title("PSF Residuals")
ax[1].set_xticklabels([])
ax[1].set_yticklabels([])
plt.show()
```



If you are not satisfied with these residuals, repeat the step.

If you think a 3star PSF fit might be a better option, continue below

Step 3

If you are still unhappy with the 2 star residuals, consider running a 3star-fit. The 3star-fit does not run by default in MORIA.

In [19]:
```
reduce.hst_fit_dataprep_threestar(directory) # Input for 3star-fit F814W filter
```

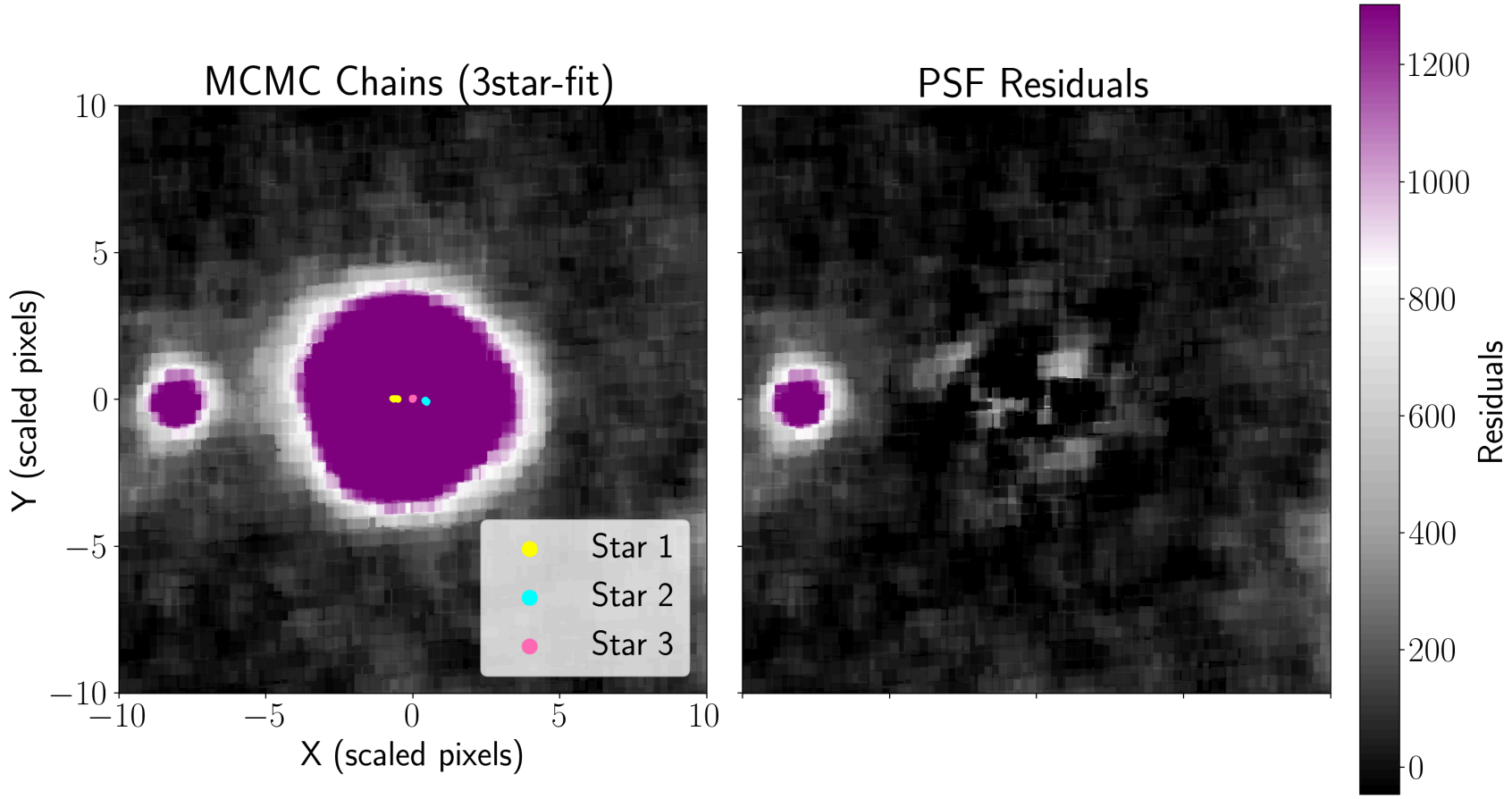
In [20]:
```
reduce.tri_fit_F814W_opt(directory)
```

Step 4

We will inspect the 3star-fit PSF residuals for F814W

In [24]:
```
################
#OPEN FITS FILE#
################
fit_file = Path(directory).resolve()/f"06.FIT/F814W/3star-fit/uvp2tri_scon_fsky_I_KeckNOcon.06.pix_show.fi
fname = Path(directory).resolve()/f"06.FIT/F814W/3star-fit/expanded_mcmc.txt"


##################
#READ MCMC CHAINS#
##################
xtarg = 3001
ytarg = 1001
colname1 = ['X1_CENTER', 'Y1_CENTER']
colname2 = ['X2_CENTER', 'Y2_CENTER']
colname3 = ['X3_CENTER', 'Y3_CENTER']
chains1 = pd.read_table(fname, usecols=[0,1], sep=r'\s+', skiprows=1, names=colname1).head(100)
chains2 = pd.read_table(fname, usecols=[2,3], sep=r'\s+', skiprows=1, names=colname2).head(100)
chains3 = pd.read_table(fname, usecols=[4,5], sep=r'\s+', skiprows=1, names=colname3).head(100)


################
#SCALE THE DATE#
################
```

```
hdul = fits.open(fit_file)
data = hdul[-1].data
hdul.close()
interval = PercentileInterval(90)
scaled = interval(data)


#######################################
#PLOT THE MCMC CHAINS  + PSF RESIDUALS#
#######################################
# image dimensions
ny, nx = scaled.shape
# scaled physical coordinates
x_extent = ((0 - xtarg)/100, (nx - xtarg)/100)
y_extent = ((0 - ytarg)/100, (ny - ytarg)/100)
fig, ax = plt.subplots(1,2, figsize=(15, 8), constrained_layout=True)
im2 = ax[0].imshow(data, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=
im = ax[0].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cma
ax[0].scatter(chains1['X1_CENTER'], chains1['Y1_CENTER'], s=12, color='yellow', label='Star 1')
ax[0].scatter(chains2['X2_CENTER'], chains2['Y2_CENTER'], s=12, color='cyan', label='Star 2')
ax[0].scatter(chains3['X3_CENTER'], chains3['Y3_CENTER'], s=12, color='hotpink', label='Star 3')
ax[0].legend(loc='lower right', markerscale=3)
cbar = fig.colorbar(im2, ax=ax[1])
cbar.set_label('Residuals')
ax[0].set_title("MCMC Chains (3star-fit)")
ax[0].set_xlabel("X (scaled pixels)")
ax[0].set_ylabel("Y (scaled pixels)")
ax[0].set_xlim(-10, 10)
ax[1].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=cus
ax[1].set_xlim(-30, -10)
ax[1].set_title("PSF Residuals")
ax[1].set_xticklabels([])
ax[1].set_yticklabels([])
plt.show()
```



Looks like the 3star PSF fitting gives us smooth residuals!

Step 5

Repeat the tri fitting process for the 606W filter.

In [25]:
```
reduce.hst_fit_dataprep_onestar(directory, f='F606W') # Input for 1star-fit F606W filter
```

In [26]:
```
reduce.hst_fit_dataprep_twostar(directory, f='F606W') # Input for 2star-fit F606w filter
```

In [27]:
```
reduce.hst_fit_dataprep_threestar(directory, f = 'F606W') # Input for 3star-fit F606W filter
```

In [28]:
```
reduce.hst_fit_final_F606W(directory) # Run the tri fitting
```

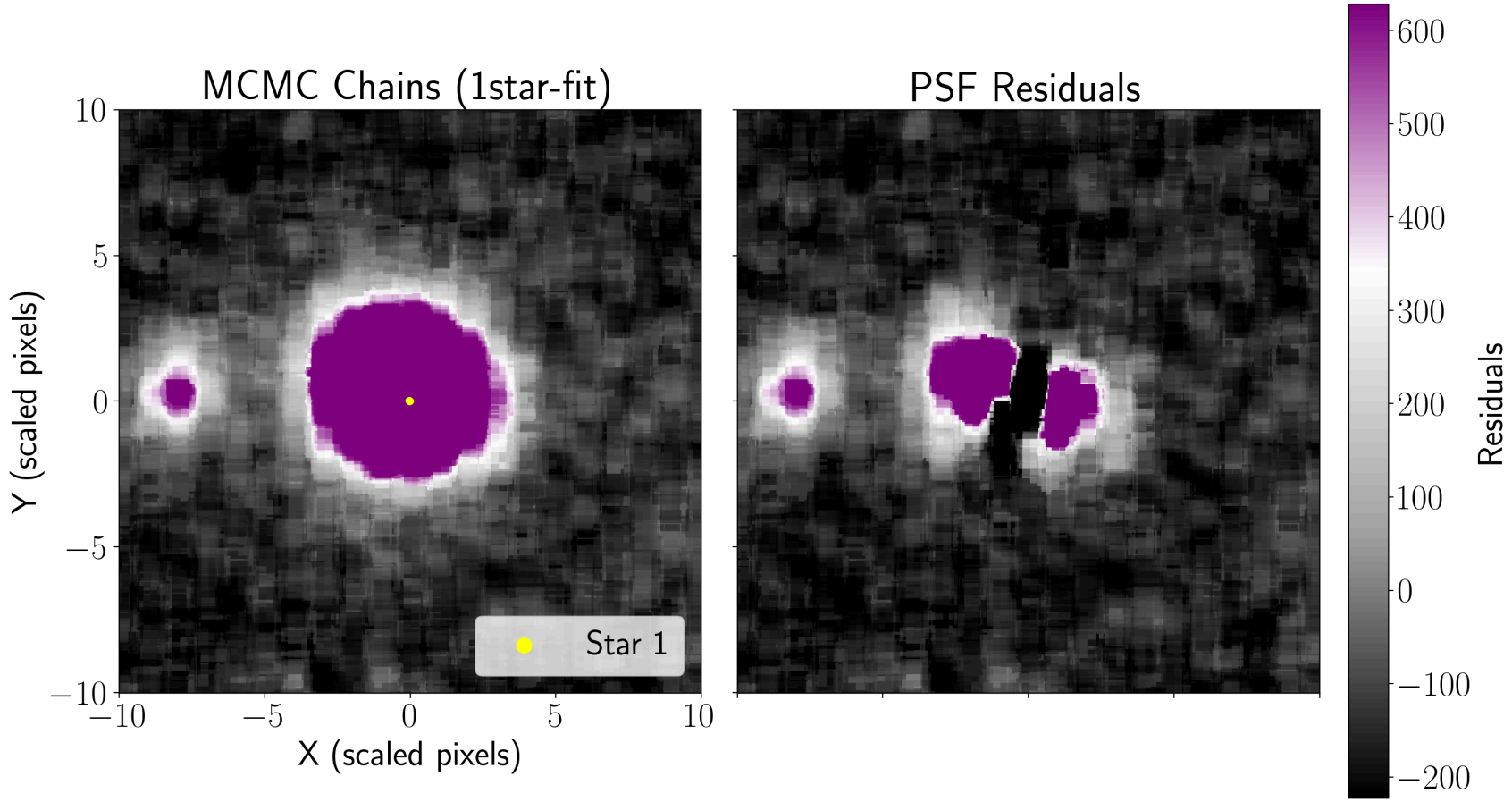
In [30]:
```
reduce.tri_fit_F606W_opt(directory)
```

Step 6

Look at the Residuals for the 1star-fit, 2star-fit and 3star-fit in the F606W filter.

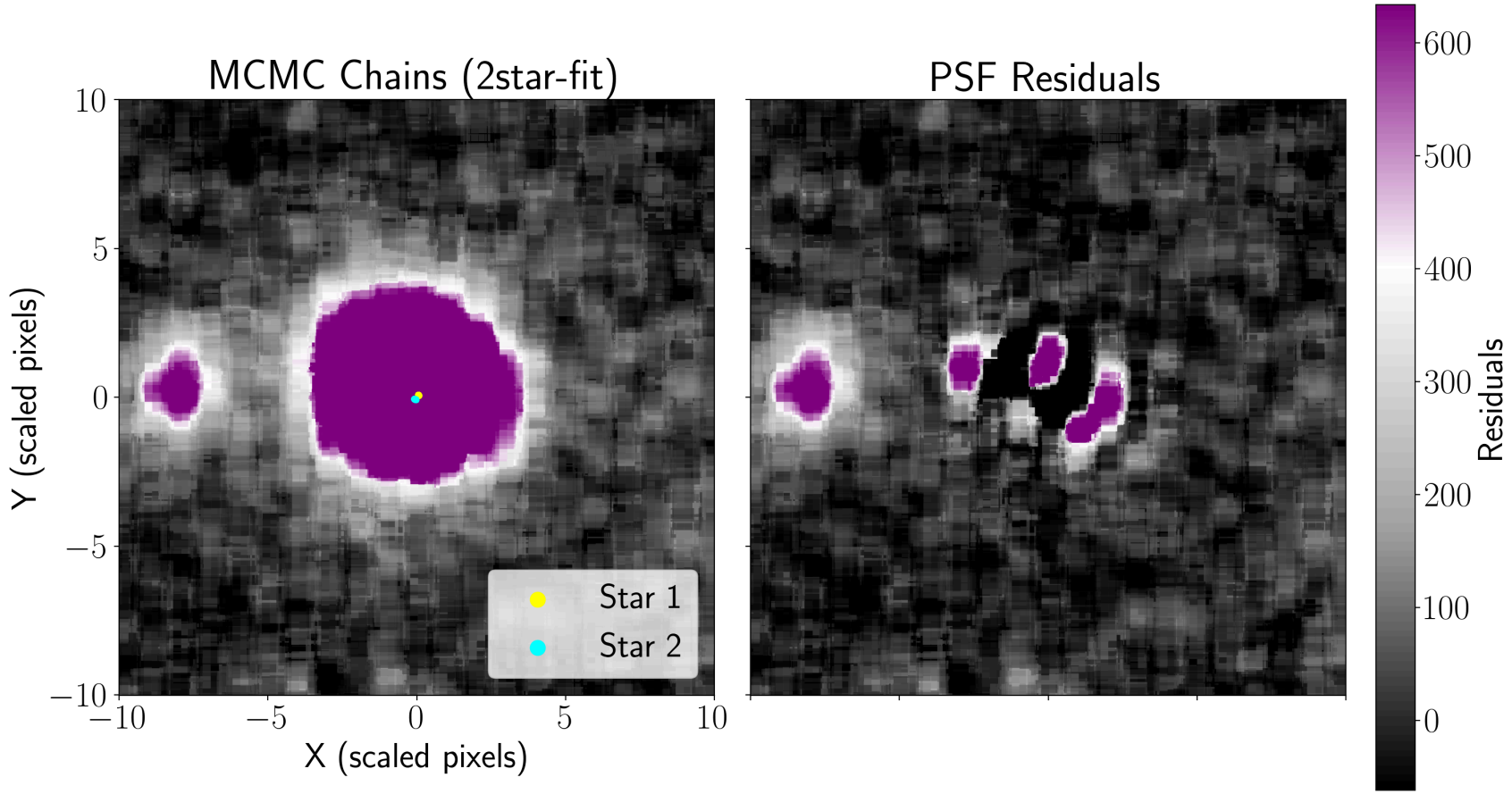
In [32]:
```
################
#OPEN FITS FILE#
################
fit_file = Path(directory).resolve()/f"06.FIT/F606W/1star-fit/uvp2tri_scon_fsky_V_KeckNOcon.06.pix_show.fi
fname = Path(directory).resolve()/f"06.FIT/F606W/1star-fit/expanded_mcmc.txt"


##################
#READ MCMC CHAINS#
##################
xtarg = 3001
ytarg = 1001
colname1 = ['X1_CENTER', 'Y1_CENTER']
chains1 = pd.read_table(fname, usecols=[0,1], sep=r'\s+', skiprows=1, names=colname1).head(100)


################
#SCALE THE DATE#
################
hdul = fits.open(fit_file)
data = hdul[-1].data
hdul.close()
interval = PercentileInterval(90)
scaled = interval(data)


#######################################
#PLOT THE MCMC CHAINS  + PSF RESIDUALS#
```

```
#####################################
# image dimensions
ny, nx = scaled.shape
# scaled physical coordinates
x_extent = ((0 - xtarg)/100, (nx - xtarg)/100)
y_extent = ((0 - ytarg)/100, (ny - ytarg)/100)
fig, ax = plt.subplots(1,2, figsize=(15, 8), constrained_layout=True)
im2 = ax[0].imshow(data, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=
im = ax[0].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cma
ax[0].scatter(chains1['X1_CENTER'], chains1['Y1_CENTER'], s=12, color='yellow', label='Star 1')
ax[0].legend(loc='lower right', markerscale=3)
cbar = fig.colorbar(im2, ax=ax[1])
cbar.set_label('Residuals')
ax[0].set_title("MCMC Chains (1star-fit)")
ax[0].set_xlabel("X (scaled pixels)")
ax[0].set_ylabel("Y (scaled pixels)")
ax[0].set_xlim(-10, 10)
ax[1].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=cus
ax[1].set_xlim(-30, -10)
ax[1].set_title("PSF Residuals")
ax[1].set_xticklabels([])
ax[1].set_yticklabels([])
plt.show()
```



In [33]:
```
################
#OPEN FITS FILE#
################
fit_file = Path(directory).resolve()/f"06.FIT/F606W/2star-fit/uvp2tri_scon_fsky_V_KeckNOcon.06.pix_show.fi
fname = Path(directory).resolve()/f"06.FIT/F606W/2star-fit/expanded_mcmc.txt"


##################
#READ MCMC CHAINS#
##################
xtarg = 3001
ytarg = 1001
colname1 = ['X1_CENTER', 'Y1_CENTER']
colname2 = ['X2_CENTER', 'Y2_CENTER']
```

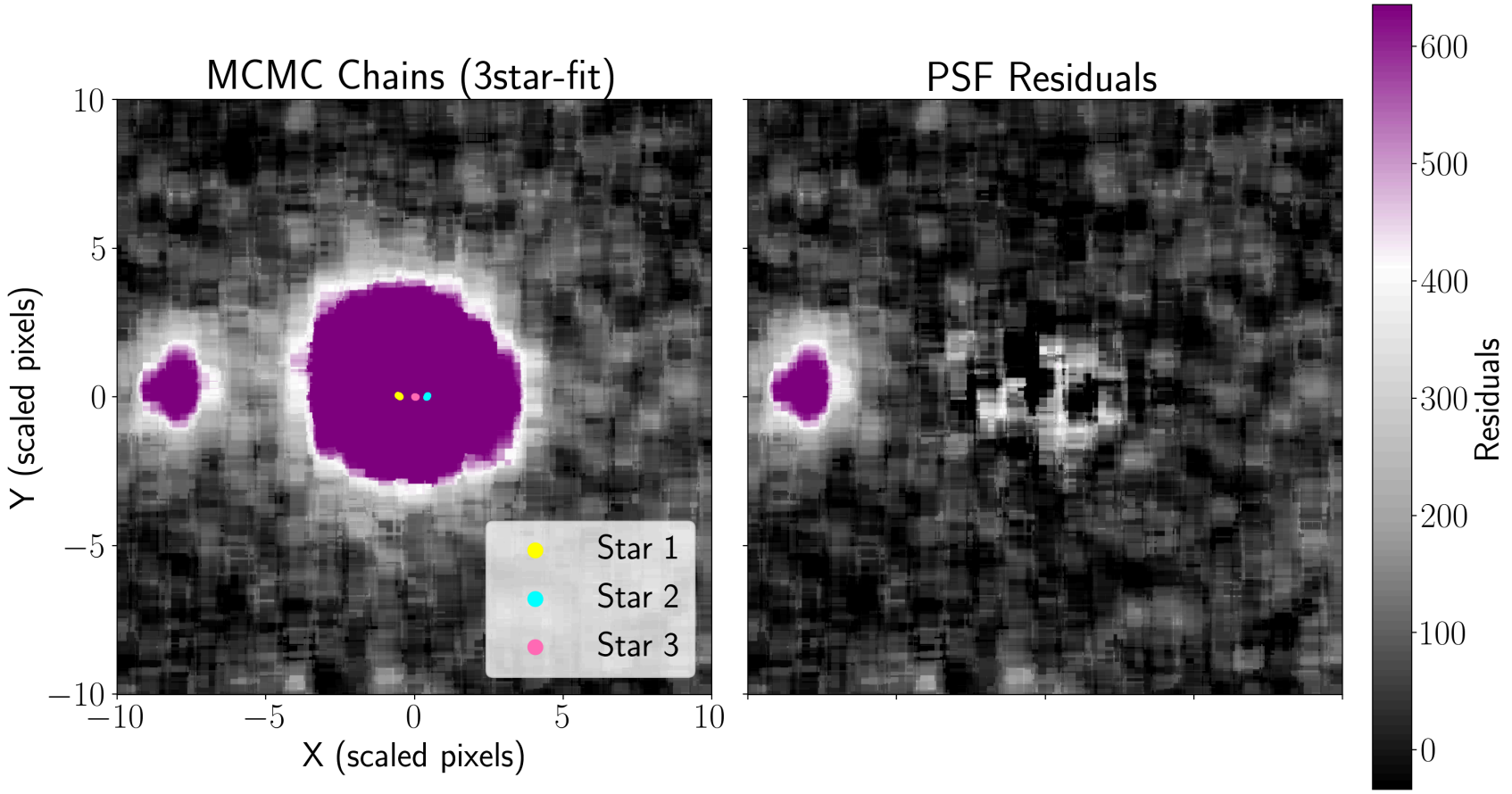
```
chains1 = pd.read_table(fname, usecols=[0,1], sep=r'\s+', skiprows=1, names=colname1).head(100)
chains2 = pd.read_table(fname, usecols=[2,3], sep=r'\s+', skiprows=1, names=colname2).head(100)


################
#SCALE THE DATE#
################
hdul = fits.open(fit_file)
data = hdul[-1].data
hdul.close()
interval = PercentileInterval(90)
scaled = interval(data)


#######################################
#PLOT THE MCMC CHAINS  + PSF RESIDUALS#
#######################################
# image dimensions
ny, nx = scaled.shape
# scaled physical coordinates
x_extent = ((0 - xtarg)/100, (nx - xtarg)/100)
y_extent = ((0 - ytarg)/100, (ny - ytarg)/100)
fig, ax = plt.subplots(1,2, figsize=(15, 8), constrained_layout=True)
im2 = ax[0].imshow(data, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=
im = ax[0].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cma
ax[0].scatter(chains1['X1_CENTER'], chains1['Y1_CENTER'], s=12, color='yellow', label='Star 1')
ax[0].scatter(chains2['X2_CENTER'], chains2['Y2_CENTER'], s=12, color='cyan', label='Star 2')
ax[0].legend(loc='lower right', markerscale=3)
cbar = fig.colorbar(im2, ax=ax[1])
cbar.set_label('Residuals')
ax[0].set_title("MCMC Chains (2star-fit)")
ax[0].set_xlabel("X (scaled pixels)")
ax[0].set_ylabel("Y (scaled pixels)")
ax[0].set_xlim(-10, 10)
ax[1].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=cus
ax[1].set_xlim(-30, -10)
ax[1].set_title("PSF Residuals")
ax[1].set_xticklabels([])
ax[1].set_yticklabels([])
plt.show()
```



In [35]:
```
################
#OPEN FITS FILE#
################
fit_file = Path(directory).resolve()/f"06.FIT/F606W/3star-fit/uvp2tri_scon_fsky_V_KeckNOcon.06.pix_show.fi
fname = Path(directory).resolve()/f"06.FIT/F606W/3star-fit/expanded_mcmc.txt"


##################
#READ MCMC CHAINS#
##################
xtarg = 3001
ytarg = 1001
colname1 = ['X1_CENTER', 'Y1_CENTER']
colname2 = ['X2_CENTER', 'Y2_CENTER']
```

```
colname3 = ['X3_CENTER', 'Y3_CENTER']
chains1 = pd.read_table(fname, usecols=[0,1], sep=r'\s+', skiprows=1, names=colname1).head(100)
chains2 = pd.read_table(fname, usecols=[2,3], sep=r'\s+', skiprows=1, names=colname2).head(100)
chains3 = pd.read_table(fname, usecols=[4,5], sep=r'\s+', skiprows=1, names=colname3).head(100)


################
#SCALE THE DATE#
################
hdul = fits.open(fit_file)
data = hdul[-1].data
hdul.close()
interval = PercentileInterval(90)
scaled = interval(data)


#######################################
#PLOT THE MCMC CHAINS  + PSF RESIDUALS#
#######################################
# image dimensions
ny, nx = scaled.shape
# scaled physical coordinates
x_extent = ((0 - xtarg)/100, (nx - xtarg)/100)
y_extent = ((0 - ytarg)/100, (ny - ytarg)/100)
fig, ax = plt.subplots(1,2, figsize=(15, 8), constrained_layout=True)
im2 = ax[0].imshow(data, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=
im = ax[0].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cma
ax[0].scatter(chains1['X1_CENTER'], chains1['Y1_CENTER'], s=12, color='yellow', label='Star 1')
ax[0].scatter(chains2['X2_CENTER'], chains2['Y2_CENTER'], s=12, color='cyan', label='Star 2')
ax[0].scatter(chains3['X3_CENTER'], chains3['Y3_CENTER'], s=12, color='hotpink', label='Star 3')
ax[0].legend(loc='lower right', markerscale=3)
cbar = fig.colorbar(im2, ax=ax[1])
cbar.set_label('Residuals')
ax[0].set_title("MCMC Chains (3star-fit)")
ax[0].set_xlabel("X (scaled pixels)")
ax[0].set_ylabel("Y (scaled pixels)")
ax[0].set_xlim(-10, 10)
ax[1].imshow(scaled, origin='lower', extent=(x_extent[0], x_extent[1], y_extent[0], y_extent[1]), cmap=cus
ax[1].set_xlim(-30, -10)
ax[1].set_title("PSF Residuals")
ax[1].set_xticklabels([])
```

```
ax[1].set_yticklabels([])
plt.show()
```



In [ ]: