Import Statements

In [1]:
```
import custom_cmap
import os
import sys
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,

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 directory structure for your data analysis should be organized as follows.

00.DATA
01.XYM
02.CMD
03.LOC_TRANS
04.PSF_EXTRACT
05.COORD_TRANS(OPTIONAL)
06.FIT
07.CALIBRATION
This directory structure has been set up in "MORIA/data".

All necessary scripts are included within the corresponding folders under MORIA/data. Therefore, the simplest way to run MORIA on your target is to copy all eight folders from MORIA/data to the location where you intend to perform your analysis. Begin by placing your exposures in data/00.DATA.

Note: chmod +x program.src is a useful command to run whenever script execution fails due to permission issues

Notebook Goal

Using the MATCHUP files generated in "00.output_stacks.ipynb", a Color–Magnitude Diagram (CMD) can be created.

The CMD is used to identify stars with colors and magnitudes similar to the target. These stars are then used to construct a PSF model with charge transfer efficiency (CTE) distortions similar to those of the target. The MATCHUP files are also used to identify brighter stars near the target that can refine the coordinate transformation and calibrate the HST photometry to the OGLE-III catalog.

The final outputs from this notebook will be found in 02.CMD.

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

Step 0

We assume you ran the 00.output_stacks.ipynb notebook succesfully

Step 1

We assume that you have already identified a target star. The file outputq_F814W.fits inside 01.XYM/F814W can be used to determine the pixel coordinates of the target.

The pixel coordinates identified in "outputq_F814W.fits" should have a corresponding match (to the nearest decimal place) in the MATCHUP files. The MATCHUP files also provide the F814W instrumental magnitude required to place the target on the CMD. To obtain the F606W instrumental magnitude, repeat the same procedure using the files in 01.XYM/F606W.

Run the Python script below to generate the CMD. This step will also move the target star entry to the top of the MATCHUP files in both 01.XYM/F814W and 01.XYM/F606W.

The remaining arguments passed to the cmd_diagram function are optional and only affect the appearance and plotting style of the CMD.

In [3]:
```
reduce.cmd_diagram(directory)
```

If you wish to continue with the pipeline beyond creating this CMD diagram, it is very important that your file 'NEARBY_SIM_STARS' have between 20-40 stars. This file can be found in your 02.CMD folder. If there are more than 40 stars, you should run "reduce.cmd_diagram(directory)" with tighter constraints.

The CMD diagrams were created in the 02.CMD folder.

In this step we also created three important files:

1. NEARBY_SIM_STARS.XYIVB_targ - This contains the stars similar to the target to be used for the PSF model construction.
2. NEARBY_REF_STARS.XYIVB_targ - This contains stars close to the target to be used to refine the coordinate transformations.
3. NOTFAR_CAL_STARS.XYIVB_targ - This contains the stars that will be calibrated to the OGLE-III catalog. Note that there aren't very many stars that won't be blended in the OGLE-III data, so the calibration stars should be selected over a wider range of coordinates than the candidate PSF stars.

Note that the code used to identify PSF model outliers currently only allows up to 40 PSF stars, so the NEARBY_SIM_STARS.XYIVB_targ file should be kept to a maximum of 40 stars (42 lines in the file). Let us open NEARBY_SIM_STARS.XYIVB_targ to ensure that this is the case.

In [4]:
```
filename_sim_stars = Path(directory).resolve()/f"02.CMD/NEARBY_SIM_STARS.XYI
cols = ["xu", "yu", "miu", "mvu", "psf"]
df = pd.read_csv(filename_sim_stars, sep=r"\s+", comment="#", header=None, r
```

In [5]:
```
df
```

Out[5]:

	xu	yu	miu	mvu	psf
0	535.232	623.895	-10.3059	-10.2660	0
1	605.840	347.395	-10.6157	-10.5408	1
2	660.372	375.588	-10.5494	-10.6844	1
3	319.483	447.817	-10.4452	-10.5412	1
4	701.458	453.411	-10.3727	-10.2701	1
5	733.149	467.782	-10.6840	-10.4758	1
6	509.629	500.115	-9.8669	-9.8175	1
7	786.732	510.220	-10.4693	-10.3872	1
8	671.937	574.571	-9.8791	-9.8178	1
9	426.031	583.324	-10.0137	-9.9077	1
10	769.581	617.612	-10.5042	-10.4691	1
11	633.511	632.121	-9.9327	-10.1316	1
12	682.615	658.290	-9.9895	-10.0602	1
13	729.124	690.555	-10.4163	-10.2940	1
14	276.550	701.201	-9.9855	-10.1942	1
15	529.740	739.459	-10.6249	-10.4368	1
16	569.409	742.518	-9.8910	-9.6955	1
17	682.796	745.898	-10.7204	-10.8184	1
18	649.335	825.814	-9.9518	-9.8343	1
19	407.315	854.391	-9.9675	-10.0330	1
20	456.545	855.083	-10.0689	-10.2338	1
21	574.668	887.663	-9.8535	-9.9704	1

Let us also open the file "show_cmd_targ" to inspect the CMD diagram generated around our target, along with panels showing the reference stars selected to create the "NEARBY_REF_STARS.XYIVB_targ" file.

Note: If running this notebook in a non-browser program (like VSCode), the below cell may fail to run or it may prompt you to download the CMD figure to your machine. Either way, the CMD file has already been created in 02.CMD/show_cmd_targ.pdf.

In [ ]:
```
import base64
from IPython.display import IFrame
pdf_path = Path(directory).resolve()/f"02.CMD/show_cmd_targ.pdf"
with open(pdf_path, "rb") as pdf_file:
```

```
        encoded_pdf = base64.b64encode(pdf_file.read()).decode("utf-8")
IFrame(f"data:application/pdf;base64,{encoded_pdf}", width=800, height=800)
```

Lastly, let us open the file "show_cmd_cal" to inspect the CMD diagram generated around our target, along with panels showing the reference stars selected to create the "NEARBY_CAL_STARS.XYIVB_targ" file

In [ ]:
```
import base64
from IPython.display import IFrame
pdf_path = Path(directory).resolve()/f"02.CMD/show_cmd_cal.pdf"
with open(pdf_path, "rb") as pdf_file:
        encoded_pdf = base64.b64encode(pdf_file.read()).decode("utf-8")
IFrame(f"data:application/pdf;base64,{encoded_pdf}", width=450, height=450)
```