WFC3/UVIS Time-dependent Photometry

Learning Goals

By the end of this tutorial, you will:

- Compute aperture photometry on FLC frames acquired at three unique epochs and apply the new time-dependent inverse sensitivity (PHOTFLAM) keywords.
- Recompute aperture photometry on modified FLC frames with 'equalized' countrate values.
- Redrizzle the 'equalized' FLC frames and compute aperture photometry on the DRC products.

Table of Contents

Introduction

For UVIS images retrieved after October 15, 2020, new time-dependent photometry keyword values (PHOTFLAM, PHTFLAM1, PHTFLAM2 and PHTRATIO) are populated in the image header and must be applied separately for each observation epoch. This is a change from prior calibration, where a single set of keyword values were provided for each filter, independent of date. For more detail on the new calibration, see WFC3 ISR 2021-04 (https://www.stsci.edu/files/live/sites/www/files/home/hst/instrumentation/wfc3/documentation/instrument-science-reports-isrs/_documents/2021/WFC3_ISR_2021-04.pdf).

In this tutorial, we show how to use the time-dependent calibration to compute aperture photometry on UVIS calibrated, CTE-corrected images (flc.fits, hereafter FLC) obtained at three epochs, spanning a total range of ~8 years and showing a loss in sensitivity of ~2%. The repository includes a CSV file containing a list of FLCs and the centroid of the star in each image, as well as the UVIS Pixel Area Maps to correct for distortion when working with FLC data.

Alternately, the FLC science arrays may be 'equalized' to account for sensitivity changes prior to computing photometry, where a reference set of keywords may be then used for all images. This photometric 'equalization' must be performed before combining any set of FLC images with AstroDrizzle which span multiple epochs in time.

1. Imports

This notebook assumes you have created the virtual environment in WFC3 Library's (https://github.com/spacetelescope/WFC3Library) installation instructions.

We import:

- *os* for setting environment variables
- *glob* for finding lists of files
- *shutil* for managing directories
- *numpy* for handling array functions
- *pandas* for managing data
- *matplotlib.pyplot* for plotting data
- *astroquery.mast Observations* for downloading data from MAST
- *astropy* for astronomy related functions
- *drizzlepac* for combining images
- *photutils* for photometric calculations
- *stwcs* for updating the World Coordinate System

In [1]:

```
%matplotlib inline

import os
import glob
import shutil

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from astropy.io import fits
from astropy.stats import sigma_clipped_stats
from astropy.coordinates import SkyCoord
from astropy import wcs
from astroquery.mast import Observations

from drizzlepac import photeq
from drizzlepac import astrodrizzle

from photutils import aperture_photometry, CircularAperture, CircularAnnulus

from stwcs import updatewcs
```

```
The following task in the stsci.skypac package can be run with TEAL:
                                    skymatch
The following tasks in the drizzlepac package can be run with TEAL:
    astrodrizzle       config_testbed      imagefindpars           map
reg
       photeq            pixreplace           pixtopix            pixt
osky
  refimagefindpars       resetbits          runastrodriz          skyt
opix
     tweakback             tweakreg           updatenpol
```

2. Download the Data

The following commands query MAST for WFC3/UVIS calibrated (FLC) data products in the F606W filter for three epochs of GD153 observations (acquired in 2009, 2013, and 2017) and then downloads them to the current directory.

In [2]:

```
data_list = Observations.query_criteria(obs_id=['IBCDA4010','ICH3040F0','IDBHA6(

Observations.download_products(data_list['obsid'], project='CALWF3',
        mrp_only=False,download_dir='./data',productSubGroupDescription=['FLC','

science_files = glob.glob('data/mastDownload/HST/*/*fits')

for im in science_files:
    root = im.split('/')[-1]
    os.rename(im,'./'+root)
shutil.rmtree('data/')
```

```
Downloading URL https://mast.stsci.edu/api/v0.1/Download/file?uri=mas
t:HST/product/ibcda4010_asn.fits (https://mast.stsci.edu/api/v0.1/Down
load/file?uri=mast:HST/product/ibcda4010_asn.fits) to ./data/mastDownl
oad/HST/ibcda4010/ibcda4010_asn.fits ... [Done]
Downloading URL https://mast.stsci.edu/api/v0.1/Download/file?uri=mas
t:HST/product/ibcda4d0q_flc.fits (https://mast.stsci.edu/api/v0.1/Down
load/file?uri=mast:HST/product/ibcda4d0q_flc.fits) to ./data/mastDownl
oad/HST/ibcda4d0q/ibcda4d0q_flc.fits ... [Done]
Downloading URL https://mast.stsci.edu/api/v0.1/Download/file?uri=mas
t:HST/product/ibcda4d6q_flc.fits (https://mast.stsci.edu/api/v0.1/Down
load/file?uri=mast:HST/product/ibcda4d6q_flc.fits) to ./data/mastDownl
oad/HST/ibcda4d6q/ibcda4d6q_flc.fits ... [Done]
Downloading URL https://mast.stsci.edu/api/v0.1/Download/file?uri=mas
t:HST/product/ibcda4dcq_flc.fits (https://mast.stsci.edu/api/v0.1/Down
load/file?uri=mast:HST/product/ibcda4dcq_flc.fits) to ./data/mastDownl
oad/HST/ibcda4dcq/ibcda4dcq_flc.fits ... [Done]
Downloading URL https://mast.stsci.edu/api/v0.1/Download/file?uri=mas
t:HST/product/ibcda4diq_flc.fits (https://mast.stsci.edu/api/v0.1/Down
load/file?uri=mast:HST/product/ibcda4diq_flc.fits) to ./data/mastDownl
```

For convenience, the file 'GD153_F606W_public.csv' contains information about the images required for the tutorial. For each FLC image, the following is provided: filename, star x-cdt, y-cdt, filter, CCD amplifier, chip, and epoch for the 3 observation dates.

In [3]:

```
df = pd.read_csv('GD153_F606W_public.csv')

df
```

Out[3]:

	FLC	Centx	Centy	Filter	Amp	Chip	Epoch
0	ibcda4d0q_flc.fits	289.940	285.934	F606W	C	2	1
1	ibcda4d6q_flc.fits	325.855	299.742	F606W	C	2	1
2	ibcda4dcq_flc.fits	312.830	321.911	F606W	C	2	1
3	ibcda4diq_flc.fits	276.756	308.288	F606W	C	2	1
4	ich304prq_flc.fits	246.181	331.752	F606W	C	2	2
5	ich304q1q_flc.fits	248.901	334.022	F606W	C	2	2
6	idbha6ntq_flc.fits	262.719	254.764	F606W	C	2	3
7	idbha6nyq_flc.fits	265.020	257.049	F606W	C	2	3

3. Correct for distortion using the Pixel Area Map

FLC frames are not corrected for distortion and pixels therefore do not have equal area on the sky. To correct for this affect, we multiply the FLC frames by the Pixel Area Map (PAM) (https://www.stsci.edu/hst/instrumentation/wfc3/data-analysis/pixel-area-maps). Since the GD153 data are C512C subarrays, the PAM needs to be "cut out" at the region corresponding to the subarray.

In [4]:

```
t1.getdata('UVIS1wfc3_map.fits')
t2.getdata('UVIS2wfc3_map.fits')

A4: pam1[-512:, :513], 'B': pam1[-512:, -513:], 'C': pam2[:512, :513], 'D': pam2[:51
```

4. Compute aperture photometry on the FLC frames

4.1 Calculate countrates

Here we set up the code that computes the photometry. We are using a standard aperture size of 10 pixels, with a sky annulus from 155 to 165 pixels. These are the steps involved in computing the photometry:

1. Loop through each FLC and load the data, MJD date of exposure, exposure time, and PHOTFLAM.

2. Divide each FLC by the exposure time to get countrates in electrons per second (FLC images are in electrons).
3. Correct for distortion by multiplying by the appropriate cutout of the pixel area map (PAM) corresponding to the subarray region.
4. Supply the x,y coordinate of the star in each image from the CSV file/pandas dataframe.
5. Define the aperture, annulus aperture, and annulus mask.
6. Compute the sigma-clipped mean of the sky annulus.
7. Compute the photometry in the aperture.
8. Subtract the sky background from the photometry derived in the previous step.
9. Store all values.

In [5]:

```
phots = []
mjds = []
dates = []
ap = 10
skyrad = [155, 165]
pfl = []
for i, flc in enumerate(df['FLC'].values):
    with fits.open(flc) as f:
        data = f[1].data
        mjd = f[0].header['EXPSTART']
        date = f[0].header['DATE-OBS']
        exptime = f[0].header['EXPTIME']
        pfl.append(f[0].header['PHOTFLAM'])
    data = data / exptime
    data = data * pams[df.at[i, 'Amp']]

    positions = (df.at[i, 'Centx'], df.at[i, 'Centy'])
    aperture = CircularAperture(positions, ap)
    annulus_aperture = CircularAnnulus(positions, r_in=skyrad[0], r_out=skyrad[1
    annulus_masks = annulus_aperture.to_mask(method='center')
    annulus_data = annulus_masks.multiply(data)
    mask = annulus_masks.data
    annulus_data_1d = annulus_data[mask > 0]
    mean_sigclip, _, _ = sigma_clipped_stats(annulus_data_1d)

    apers = [aperture, annulus_aperture]
    phot_table = aperture_photometry(data,apers)

    background = mean_sigclip * aperture.area
    final_sum = phot_table['aperture_sum_0'] - background
    phots.append(final_sum[0])
    mjds.append(mjd)
    dates.append(date)
```

Next, we append the coutrates, MJD's, and PHOTFLAM values to our dataframe.

In [6]:

```
df['Countrate'] = phots
df['MJD'] = mjds
df['DATE-OBS'] = dates
df['PHOTFLAM'] = pfl

df
```

Out[6]:

	FLC	Centx	Centy	Filter	Amp	Chip	Epoch	Countrate	MJD
0	ibcda4d0q_flc.fits	289.940	285.934	F606W	C	2	1	103300.375759	55170.563332
1	ibcda4d6q_flc.fits	325.855	299.742	F606W	C	2	1	103292.263099	55170.569026
2	ibcda4dcq_flc.fits	312.830	321.911	F606W	C	2	1	103661.304094	55170.574721
3	ibcda4diq_flc.fits	276.756	308.288	F606W	C	2	1	103448.275067	55170.620196
4	ich304prq_flc.fits	246.181	331.752	F606W	C	2	2	102472.013533	56629.866744
5	ich304q1q_flc.fits	248.901	334.022	F606W	C	2	2	102744.141944	56629.877033
6	idbha6ntq_flc.fits	262.719	254.764	F606W	C	2	3	101822.670079	58068.856082
7	idbha6nyq_flc.fits	265.020	257.049	F606W	C	2	3	101552.953214	58068.861359

4.2 Calculate magnitudes

We now convert countrates into ST magnitudes using the PHOTFLAM value and the following equation. The EE_{r10} is the encircled energy term for an aperture radius of r=10 pixels (0.4 arcseconds). For F606W with UVIS2, this is 0.91. This value can be computed using `stsynphot` as described in the 'Photometry Examples' notebook in this WFC3 Library repository.

In [7]:

```
EE_r10 = 0.91
df['STMags'] = -21.1 -2.5*np.log10(df['PHOTFLAM']) -2.5*np.log10(df['Countrate']
```

In [8]:

```
1 df
```

Out[8]:

	FLC	Centx	Centy	Filter	Amp	Chip	Epoch	Countrate	MJD	DATE-OBS	PHOTFL
0	ibcda4d0q_flc.fits	289.940	285.934	F606W	C	2	1	103300.375759	55170.563332	2009-12-05	1.1539
1	ibcda4d6q_flc.fits	325.855	299.742	F606W	C	2	1	103292.263099	55170.569026	2009-12-05	1.1539
2	ibcda4dcq_flc.fits	312.830	321.911	F606W	C	2	1	103661.304094	55170.574721	2009-12-05	1.1539
3	ibcda4diq_flc.fits	276.756	308.288	F606W	C	2	1	103448.275067	55170.620196	2009-12-05	1.1539
4	ich304prq_flc.fits	246.181	331.752	F606W	C	2	2	102472.013533	56629.866744	2013-12-03	1.1638
5	ich304q1q_flc.fits	248.901	334.022	F606W	C	2	2	102744.141944	56629.877033	2013-12-03	1.1638

4.3 Plot countrate vs date

We first plot the photometric countrates (electrons per second) vs time in MJD. The decline in the observed countrate is due to sensitivity loss in F606W at a rate of ~0.2% per year.

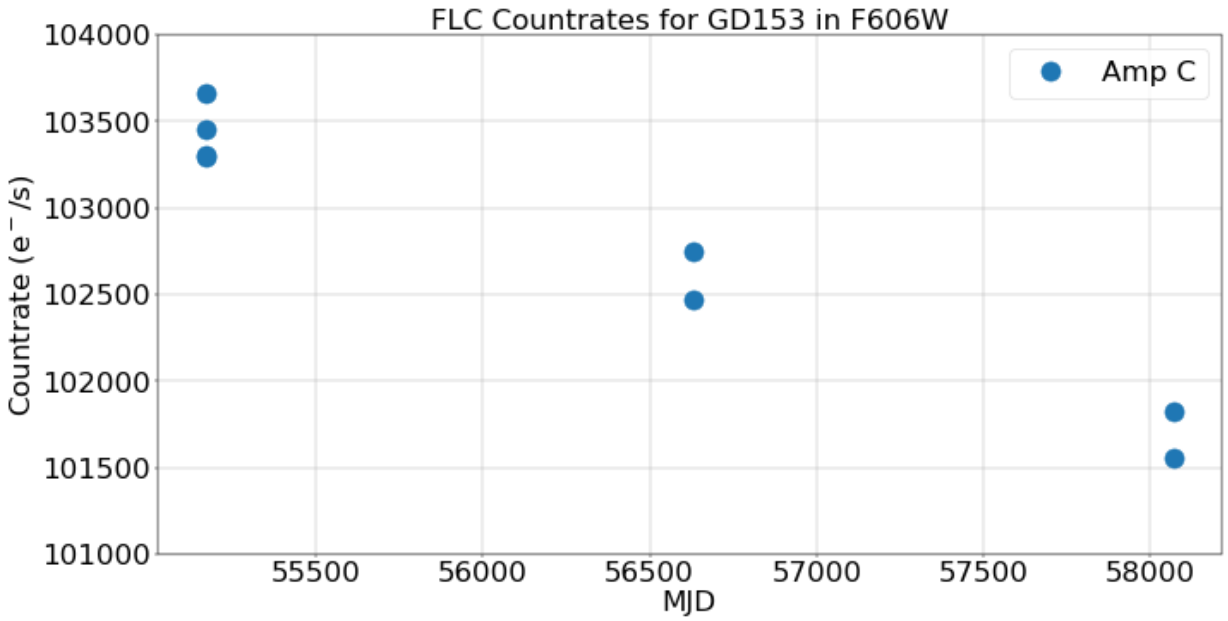
In [9]:

```
fig = plt.figure(figsize=(20, 10), dpi=40)

plt.plot(df['MJD'], df['Countrate'], 'o', markersize=20, label='Amp C')
plt.grid()
plt.xlabel('MJD', fontsize=30)
plt.xticks(fontsize=30)
plt.yticks(fontsize=30)
plt.ylabel(r'Countrate (e$^-$/s)', fontsize=30)
plt.title('FLC Countrates for GD153 in F606W', fontsize=30)
plt.ylim(101000, 104000)
plt.legend(loc=0, fontsize=30)
```

Out[9]:

```
<matplotlib.legend.Legend at 0x7fe6210579d0>
```



4.4 Plot magnitude vs date

Now, we plot the ST magnitude versus time in MJD. This is computed using the 'corrected' PHOTFLAM keyword, so the magnitude values are stable over time.

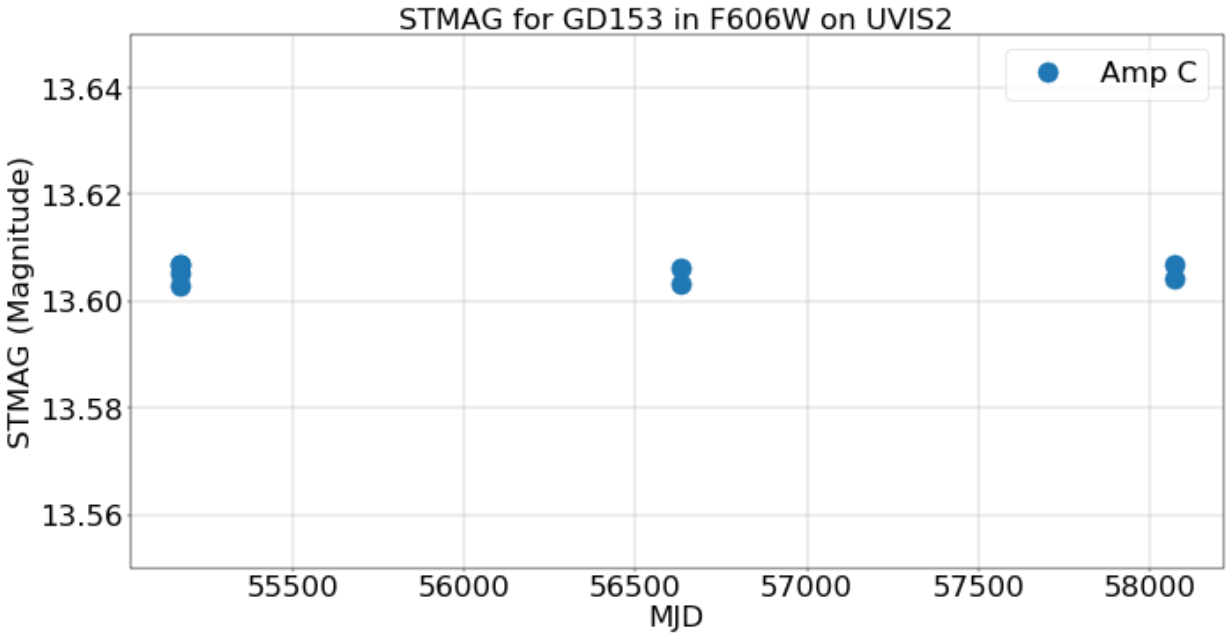
In [10]:

```
fig = plt.figure(figsize=(20, 10), dpi=40)

plt.plot(df['MJD'], df['STMags'], 'o', markersize=20, label='Amp C')
plt.grid()
plt.ticklabel_format(useOffset=False)
plt.xlabel('MJD', fontsize=30)
plt.xticks(fontsize=30)
plt.yticks(fontsize=30)
plt.ylabel('STMAG (Magnitude)', fontsize=30)
plt.ylim(13.55, 13.65)
plt.title('STMAG for GD153 in F606W on UVIS2', fontsize=30)
plt.legend(loc=0, fontsize=30)
```

Out[10]:

```
<matplotlib.legend.Legend at 0x7fe650ee9350>
```



5. Correct the FLC frames using 'photometric equalization'

This single-line step will equalize the countrates in the science array of the FLC frames to match any specified 'reference' image. For more details, see the drizzlepac documentation for the phot_eq software (https://drizzlepac.readthedocs.io/en/latest/photeq.html). Note that at this step, we overwrite the science pixels in the original FLC files. In this case the data are sorted, and the software automatically uses the PHOTFLAM value of the first 2009 image as a reference for matching the with the other images. You can supply a given reference PHOTFLAM value to the photeq call and/or ensure that your data are time-sorted.

Note: Running this cell will edit the FLCs in the local directory, and you will need to download the files again if you require the original data.

In [11]:

```
1 photeq.photeq(','.join(df['FLC'].values), readonly=False)
```

```
***** drizzlepac.photeq started on 2021-11-08 17:14:05.094036
      Version 0.2 (06-Nov-2015)

PRIMARY PHOTOMETRIC KEYWORD: PHOTFLAM
SECONDARY PHOTOMETRIC KEYWORD(S): PHOTFNU
REFERENCE VALUE FROM FILE: 'ibcda4d0q_flc.fits[0]'
REFERENCE 'PHOTFLAM' VALUE IS: 1.1539911e-19

Processing file 'ibcda4d0q_flc.fits'
   * Primary header:
     - 'PHOTFLAM' = 1.1539911e-19 found in the primary header.
     - Data conversion factor based on primary header: 1.0
   * EXT: ('SCI', 1)
     - Setting PHOTFLAM to 1.1539911e-19 (old value was 1.1539911e-19)
     - Computed conversion factor for data: 1.0
     - Setting PHOTFNU to 1.3456969e-07 (old value was 1.3456969e-07)
     - Data have been multiplied by 1.0
     - Error array (ext=('ERR', 1)) has been multiplied by 1.0
```

5.1 Recompute countrates

We repeat the same aperture photometry as in Section 4.1 but using the photometrically equalized FLC data.

In [12]:

```
phots = []
for i, flc in enumerate(df['FLC'].values):
    with fits.open(flc) as f:
        data = f[1].data
        exptime = f[0].header['EXPTIME']
    data = data / exptime
    data = data * pams[df.at[i, 'Amp']]

    positions = (df.at[i, 'Centx'], df.at[i, 'Centy'])
    aperture = CircularAperture(positions, ap)
    annulus_aperture = CircularAnnulus(positions, r_in=skyrad[0], r_out=skyrad[1
    annulus_masks = annulus_aperture.to_mask(method='center')
    annulus_data = annulus_masks.multiply(data)
    mask = annulus_masks.data
    annulus_data_1d = annulus_data[mask > 0]
    mean_sigclip, _, _ = sigma_clipped_stats(annulus_data_1d)
    background = mean_sigclip * aperture.area

    apers = [aperture, annulus_aperture]
    phot_table = aperture_photometry(data,apers)

    final_sum = phot_table['aperture_sum_0'] - background
    phots.append(final_sum[0])
    mjds.append(mjd)
```

We'll append the photometrically equalized countrates to our dataframe.

In [13]:

```
df['Phot-eq'] = phots
```

In [14]:

```
1 df
```

Out[14]:

	FLC	Centx	Centy	Filter	Amp	Chip	Epoch	Countrate	MJD
0	ibcda4d0q_flc.fits	289.940	285.934	F606W	C	2	1	103300.375759	55170.563332
1	ibcda4d6q_flc.fits	325.855	299.742	F606W	C	2	1	103292.263099	55170.569026
2	ibcda4dcq_flc.fits	312.830	321.911	F606W	C	2	1	103661.304094	55170.574721
3	ibcda4diq_flc.fits	276.756	308.288	F606W	C	2	1	103448.275067	55170.620196
4	ich304prq_flc.fits	246.181	331.752	F606W	C	2	2	102472.013533	56629.866744
5	ich304q1q_flc.fits	248.901	334.022	F606W	C	2	2	102744.141944	56629.877033
6	idbha6ntq_flc.fits	262.719	254.764	F606W	C	2	3	101822.670079	58068.856082
7	idbha6nyq_flc.fits	265.020	257.049	F606W	C	2	3	101552.953214	58068.861359

5.2 Plot corrected countrate vs date

We plot the photometric countrate (electrons per second) versus time in MJD. The corrected data in blue shows that the countrate is now ~flat over time to within the measurment errors. The original countrates are shown in grey and show a decline of nearly 2% over the three epochs.

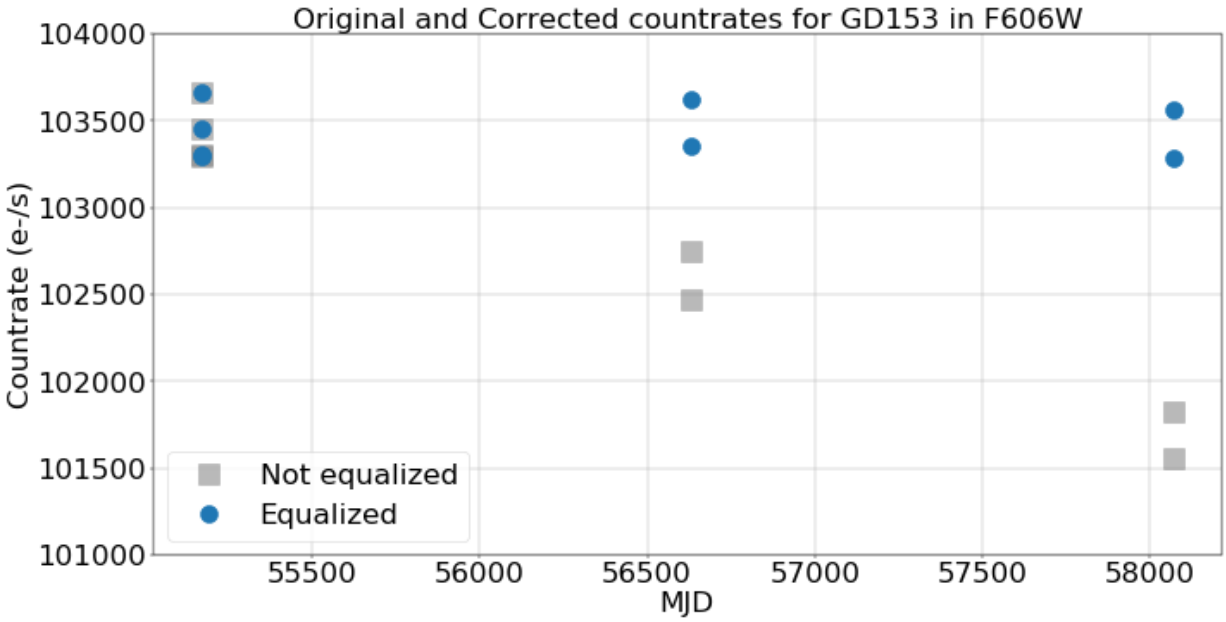
In [15]:

```
fig = plt.figure(figsize=(20, 10), dpi=40)

plt.plot(df['MJD'], df['Countrate'], 's', markersize=20, label='Not equalized',
plt.plot(df['MJD'], df['Phot-eq'], 'o', markersize=18, label='Equalized')
plt.grid()
plt.xlabel('MJD', fontsize=30)
plt.xticks(fontsize=30)
plt.yticks(fontsize=30)
plt.ylabel('Countrate (e-/s)', fontsize=30)
plt.title('Original and Corrected countrates for GD153 in F606W', fontsize=30)
plt.ylim(101000, 104000)
plt.legend(loc=0, fontsize=30)
```

Out[15]:

```
<matplotlib.legend.Legend at 0x7fe620ee7810>
```



6. Redrizzle the corrected FLC frames for each epoch

The corrected FLC data from each epoch can now be redrizzled to correct for distortion, to remove cosmic rays and bad pixels, and to improve the signal-to-noise in the combined DRC product at each date. (Alternately, the entire set of FLC images may be combined to produce a single DRC image to use for photometry.)

Warning: This cell may take a few minutes to complete.

In [16]:

```
for asn in glob.glob('*asn.fits'):
    astrodrizzle.AstroDrizzle(asn,
                skymethod= 'match',
                skystat='mean',
                driz_sep_bits='80',
                combine_type='median',
                combine_nhigh=1,
                driz_cr_snr= '3.5 3.0',
                driz_cr_scale= '2.0 1.5',
                final_bits= '80',
                build=True,
                clean=True,
                preserve=False,
                num_cores=1)
```

```
Setting up logfile :  astrodrizzle.log
AstroDrizzle Version 3.1.3 (2019-11-06 14:37:50 -0500) started at: 17:
14:35.660 (08/11/2021)

==== Processing Step  Initialization  started at  17:14:35.662 (08/11/
2021)
WCS Keywords

Number of WCS axes: 2
CTYPE : 'RA---TAN'  'DEC--TAN'
CRVAL : 194.25951624762993  22.03131176266394
CRPIX : 259.0  273.0
CD1_1 CD1_2  : 1.0852229209200958e-05  1.8306773584773e-06
CD2_1 CD2_2  : 1.8306773584773e-06  -1.0852229209200958e-05
NAXIS : 518  546
******************************************************************
**********
*
*  Estimated memory usage:  up to 5 Mb.
```

In [18]:

```
drcs = ['ibcda4010_drc.fits', 'ich3040f0_drc.fits', 'idbha6040_drc.fits']
drcents = [(327.9, 341.4), (251.1, 350.0), (266.6, 273.6)]
```

6.1 Recalculate photometry on the new DRC frames

We now perform the photometry on the drizzled (DRC) products, using similar techniques as in Section 4.1. Drizzled images are in units of electrons per second are already corrected for distortion, so we no longer need to divide by the exposure time or apply the PAM prior to computing photometry.

In [19]:

```
phots = []
mjds = []
pfls = []
for i, drc in enumerate(drcs):
    data = fits.getdata(drc, ext=1)

    mjds.append(np.mean(df.query('Epoch == {}'.format(i+1))['MJD']))
    pfls.append(df.query('Epoch == {}'.format(i+1))['PHOTFLAM'].values[0])

    positions = drcents[i]
    aperture = CircularAperture(positions, ap)
    annulus_aperture = CircularAnnulus(positions, r_in=skyrad[0], r_out=skyrad[1
    annulus_masks = annulus_aperture.to_mask(method='center')
    annulus_data = annulus_masks.multiply(data)
    mask = annulus_masks.data
    annulus_data_1d = annulus_data[mask > 0]
    mean_sigclip, _, _ = sigma_clipped_stats(annulus_data_1d)
    background = mean_sigclip * aperture.area

    apers = [aperture, annulus_aperture]
    phot_table = aperture_photometry(data,apers)

    final_sum = phot_table['aperture_sum_0'] - background
    phots.append(final_sum[0])
```

In [20]:

```
mags = -21.1 - 2.5*np.log10(pfls[0]) - 2.5*np.log10(phots) - 2.5*np.log10(1./EE_
print(mags)
```

```
[13.60509475 13.60322167 13.60502334]
```

6.2 Plot DRC countrate vs date

The corrected photometric countrate from the DRC and the recomputed magnitudes are plotted below. Note how they agree with the results from the previous step using the FLC and the PAM.

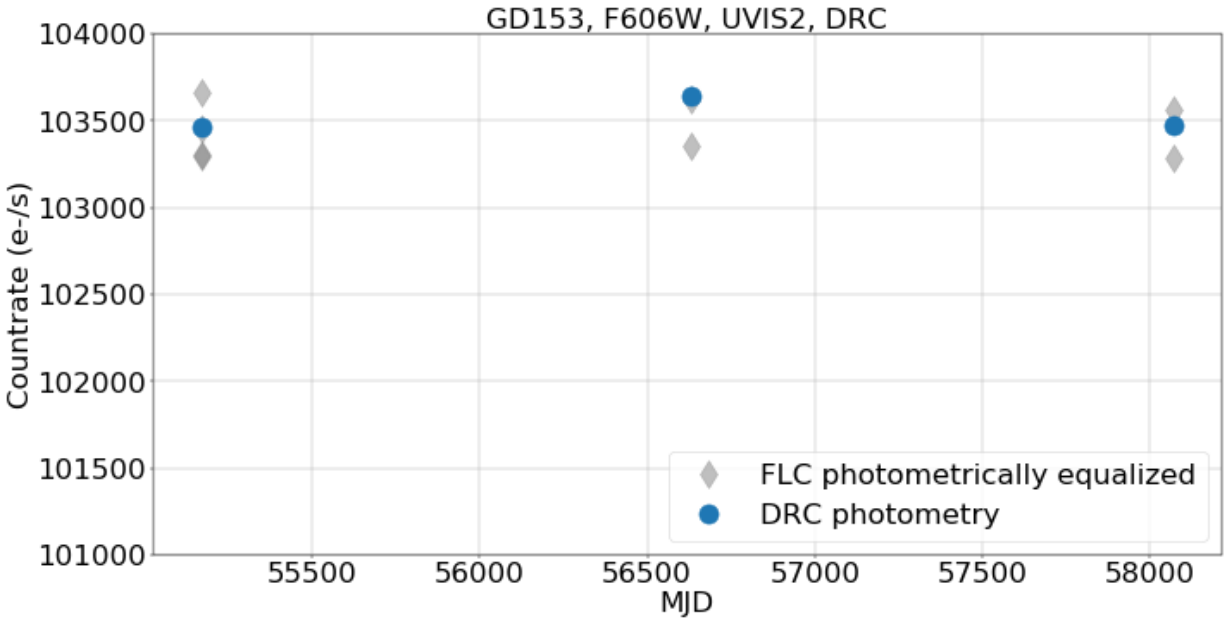
In [21]:

```
fig = plt.figure(figsize=(20, 10), dpi=40)

plt.plot(df['MJD'], df['Phot-eq'], 'd', markersize=20, color='grey', label='FLC
plt.plot(mjds, phots, 'o', markersize=20, label='DRC photometry')
plt.grid()
plt.xlabel('MJD', fontsize=30)
plt.xticks(fontsize=30)
plt.yticks(fontsize=30)
plt.ylabel('Countrate (e-/s)', fontsize=30)
plt.title('GD153, F606W, UVIS2, DRC', fontsize=30)
plt.ylim(101000, 104000)
plt.legend(loc=4, fontsize=30)
```

Out[21]:

<matplotlib.legend.Legend at 0x7fe651d479d0>



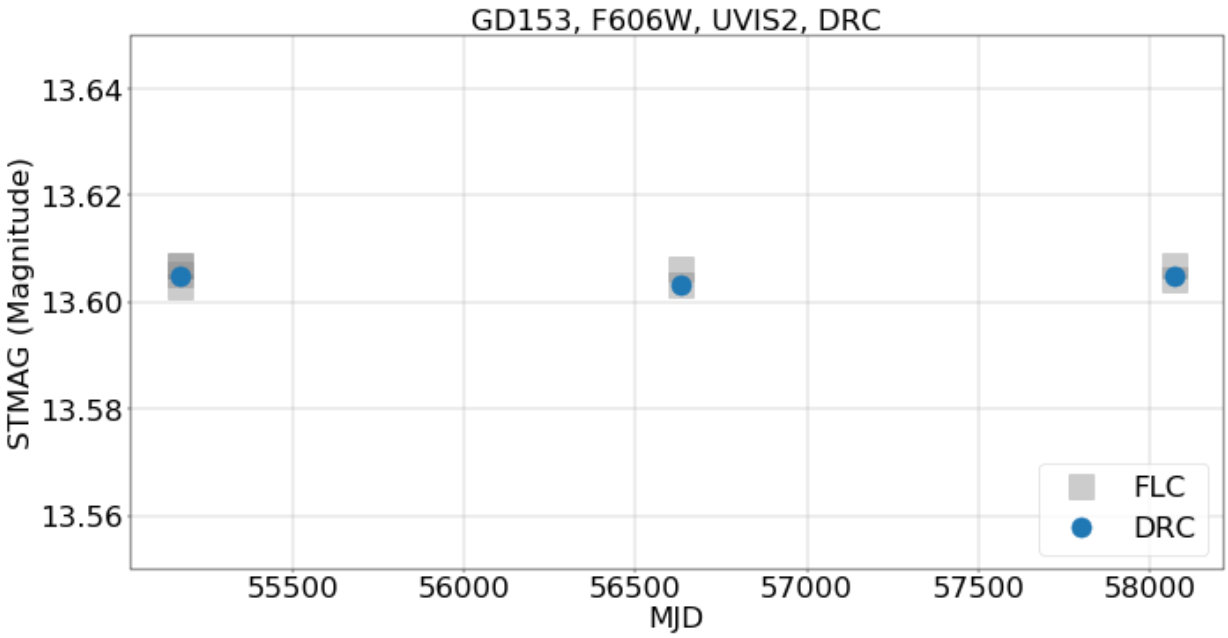
In [22]:

```
fig = plt.figure(figsize=(20, 10), dpi=40)

plt.plot(df['MJD'], df['STMags'], 's', markersize=25, label='FLC', alpha=0.4, co
plt.plot(mjds, mags, 'o', markersize=20, label='DRC')
plt.grid()
plt.ticklabel_format(useOffset=False)
plt.xlabel('MJD', fontsize=30)
plt.xticks(fontsize=30)
plt.yticks(fontsize=30)
plt.ylabel('STMAG (Magnitude)', fontsize=30)
plt.ylim(13.55, 13.65)
plt.title('GD153, F606W, UVIS2, DRC', fontsize=30)
plt.legend(loc=4, fontsize=30)
```

Out[22]:

```
<matplotlib.legend.Legend at 0x7fe6009a2350>
```



7. Conclusions

Thank you for walking through this notebook. Now using WFC3 data, you should be more familiar with:

- Computing aperture photometry and magnitudes on:

- FLC frames using new time-dependent photometry keywords.
- FLC frames with equalized countrate values.
- DRC frames produced from corrected FLCs.

Congratulations, you have completed the notebook!

Additional Resources

Below are some additional resources that may be helpful. Please send any questions through the HST Helpdesk (https://stsci.service-now.com/hst).

- WFC3 Website (https://www.stsci.edu/hst/instrumentation/wfc3)
- WFC3 Instrument Handbook (https://hst-docs.stsci.edu/wfc3ihb)
- WFC3 Data Handbook (https://hst-docs.stsci.edu/wfc3dhb)
  - see section 9.5.2 for reference to this notebook

About this Notebook

Authors: Harish Khandrika, Jennifer Mack; WFC3 Instrument Team

Updated on: 2021-09-10

Citations

If you use `numpy`, `astropy`, `drizzlepac`, and `photutils` for published research, please cite the authors. Follow these links for more information about citing the libraries below:

- Citing `numpy` (https://www.scipy.org/citing.html#numpy)
- Citing `astropy` (https://www.astropy.org/acknowledging.html)
- Citing `drizzlepac` (https://drizzlepac.readthedocs.io/en/latest/LICENSE.html)
- Citing `photutils` (https://photutils.readthedocs.io/en/stable/license.html)

Top of Page