Calculating WFC3 Zeropoints with STSynphot

Learning Goals

By the end of this tutorial, you will:

- Calculate zeropoints and other photometric properties using `stsynphot`.
- Create, plot, and save 'total system throughput' tables.

Table of Contents

Introduction

This notebook shows how to calculate photometric zeropoints using the Python package `stsynphot` for any WFC3 detector, filter, date, or aperture. This tutorial is especially useful for calculating Vegamag zeropoints, which require an input spectrum. The notebook is also useful for computing time-dependent WFC3/UVIS zeropoints for any observation date, as the values listed in 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) are defined for the reference epoch. As of mid-2021, the WFC3/IR zeropoints are not time-dependent.

More documentation on `stsynphot` is available here (https://stsynphot.readthedocs.io/en/latest/index.html). Using `stsynphot` requires downloading the throughput curves for the HST instruments and optical path. One method of doing this is shown in Section 2. More information on the throughput tables can be found here (https://www.stsci.edu/hst/instrumentation/reference-data-for-calibration-and-tools/synphot-throughput-tables).

1. Imports

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

We import:

- *os* for setting environment variables
- *numpy* for handling array functions
- *matplotlib.pyplot* for plotting data
- *astropy* for astronomy related functions
- *synphot* and *stsynphot* for evaluating synthetic photometry

In [1]:

```
import os

import numpy as np
import matplotlib.pyplot as plt

from astropy.table import Table
from astropy.time import Time

from synphot import Observation
import stsynphot as stsyn
```

2. Download throughput tables and define variables

This section obtains the WFC3 throughput component tables for use with `synphot`. If reference files need to be downloaded, please uncomment and execute the code block below.

In [ ]:

```
# cmd_input = 'curl -O ftp://archive.stsci.edu/pub/hst/pysynphot/synphot1.tar.gz
# os.system(cmd_input)
```

Once the files are downloaded, unpack the files and set the environment variable `PYSYN_CDBS` to the path of the unpacked files.

In [ ]:

```
# os.environ['PYSYN_CDBS'] = '/path/to/my/reference/files/'
```

Rather than downloading the entire calspec database (synphot6.tar.gz), we can point directly to the latest Vega spectrum which is required for computing VEGAMAG.

In [2]:

```
vega_url = 'https://ssb.stsci.edu/trds/calspec/alpha_lyr_stis_010.fits'
stsyn.Vega = stsyn.spectrum.SourceSpectrum.from_file(vega_url)
```

3. Set up the 'obsmode' string

Parameters to set in the `obsmode` string include:

1. detector,
2. filter,
3. observation date (WFC3/UVIS only), and
4. aperture size (in arcsec).

Note that a 6.0" aperture is considered to be "infinite", thus containing all of the flux. The zeropoints posted on the WFC3 website are calculated for an infinite aperture, so when computing photometry for smaller radii, aperture corrections must be applied.

The inputs below can be changed to any desired `obsmode` , with examples of alternate parameters shown as commented lines.

First, here are some detector examples with WFC3/UVIS1 as the default, and other options including both WFC3/UVIS chips or the WFC3/IR detector.

Note: if the IR detector is chosen, the filtnames below must be updated.

In [3]:

```
detectors  = ['uvis1']
#detectors = ['uvis1', 'uvis2']
#detectors = ['ir']
```

Next, here are some filter examples with all WFC3/UVIS filters as the default, and other options including just F606W and the WFC3/IR filters.

Note: if WFC3/IR filters is chosen, the detectors above must be set to ['ir'].

In [4]:

```
filtnames = ['f200lp','f218w','f225w','f275w','f280n','f300x', 'f336w','f343n','
             'f373n', 'f390m','f390w','f395n','f410m','f438w', 'f467m','f469n','
             'f475x', 'f487n','f502n','f547m','f555w','f600lp','f606w','f621m','
             'f631n', 'f645n','f656n','f657n','f658n','f665n', 'f673n','f680n','
             'f763m', 'f775w','f814w','f845m','f850lp','f953n']
#filtnames = ['f606w']
#filtnames = ['f098m','f105w','f110w','f125w','f126n','f127m','f128n','f130n','i
```

Now, here are some date examples with the WFC3/UVIS reference epoch (55008 in MJD; 2009-06-26) as the default, and the other option being the time right now.

In [5]:

```
mjd = '55008'
# mjd = str(Time.now().mjd)
```

Finally, here are some aperture radius examples with 6.0" (151 pixels; "infinity") as the default, and the other options including 0.396" (10 pixels for WFC3/UVIS) and 0.385" (3 pixels for WFC3/IR).

In [6]:

```
aper = '6.0'
#aper = '0.396'
#aper = '0.385'
```

4. Basic usage for a single 'obsmode'

The calculation of the zeropoints starts with creating a specific bandpass object. Bandpasses generally consist of at least an instrument name, detector name, and filter name, though other parameters (such as the MJD and aperture radius shown above) are optional.

The cell below defines `obsmode` and creates a bandpass object.

In [7]:

```
obsmode = 'wfc3,uvis1,f200lp'
bp = stsyn.band(obsmode)
```

Optional parameters are supplied on the end of the basic bandpass:

In [8]:

```
obsmode = 'wfc3,uvis1,f200lp,mjd#55008,aper#6.0'
bp = stsyn.band(obsmode)
```

In addition, we can use the parameters defined in Section 3.

In [9]:

```
obsmode = 'wfc3,{},{},mjd#{},aper#{}'.format(detectors[0],filtnames[0],mjd,aper)
bp = stsyn.band(obsmode)
```

5. Compute zeropoints and other photometric properties

With the bandpass objects, we can now calculate zeropoints, pivot wavelengths, and photometric bandwidths.

With the bandpass objects, we can now calculate zeropoints, pivot wavelengths, and photometric bandwidths.
To calculate Vegamag zeropoints, we use the Vega spectrum to calculate the flux in a given bandpass.

In [10]:

```
def calculate_values(detector, filt, mjd, aper):
    # parameters can be removed from obsmode as needed
    obsmode = 'wfc3,{},{},mjd#{},aper#{}'.format(detector, filt, mjd, aper)
    bp = stsyn.band(obsmode)

    # STMag
    photflam = bp.unit_response(stsyn.conf.area)  # inverse sensitivity in flam
    stmag = -21.1 -2.5 * np.log10(photflam.value)

    # Pivot Wavelength and bandwidth
    photplam  = bp.pivot() # pivot wavelength in angstroms
    bandwidth = bp.photbw() # bandwidth in angstroms

    # ABMag
    abmag = stmag - 5 * np.log10(photplam.value) + 18.6921

    # Vegamag
    obs = Observation(stsyn.Vega, bp, binset=bp.binset)  # synthetic observatio
    vegamag = -obs.effstim(flux_unit='obmag', area=stsyn.conf.area)

    return obsmode, photplam.value, bandwidth.value, photflam.value, stmag, abma
```

In [11]:

```
obsmode, photplam, bandwidth, photflam, stmag, abmag, vegamag = calculate_values

# print values
print('Obsmode                                  PivotWave Photflam   STMAG   ABMAG
print(f'{obsmode}, {photplam:.1f}, {photflam:.4e}, {stmag:.3f}, {abmag:.3f}, {ve

```

```
Obsmode                                     PivotWave Photflam   STMAG   ABMA
G   VEGAMAG
wfc3,uvis1,f200lp,mjd#55008,aper#6.0, 4971.9, 4.9157e-20, 27.171, 27.3
81, 26.931
```

6. Iterate over multiple 'obsmodes'

To calculate zeropoints for multiple detectors and/or filters, we can use the function defined above and loop through detectors and filters defined in Section 3.

In [12]:

```
oms, pivots, bws, pfs, st, ab, vm = [], [], [], [], [], [], []

print('Obsmode                              PivotWave Photflam   STMAG   ABMAG
for detector in detectors:
    for filt in filtnames:
        res = calculate_values(detector, filt, mjd, aper)
        obsmode, photplam, bandwidth, photflam, stmag, abmag, vegamag = res # sc

        # print values
        print(f'{obsmode}, {photplam:.1f}, {photflam:.4e}, {stmag:.3f}, {abmag:.

        oms.append(obsmode)
        pivots.append(photplam)
        bws.append(bandwidth)
        pfs.append(photflam)
        st.append(stmag)
        ab.append(abmag)
        vm.append(vegamag)

```

```
wfc3,uvis1,f280n,mjd#55008,aper#6.0, 2832.9, 5.7472e-17, 19.501, 20.93
2, 19.516
wfc3,uvis1,f300x,mjd#55008,aper#6.0, 2820.5, 1.4093e-18, 23.527, 24.96
8, 23.565
wfc3,uvis1,f336w,mjd#55008,aper#6.0, 3354.5, 1.2848e-18, 23.628, 24.69
2, 23.527
wfc3,uvis1,f343n,mjd#55008,aper#6.0, 3435.2, 2.5672e-18, 22.876, 23.88
9, 22.754
wfc3,uvis1,f350lp,mjd#55008,aper#6.0, 5873.9, 5.1638e-20, 27.118, 26.9
65, 26.810
wfc3,uvis1,f373n,mjd#55008,aper#6.0, 3730.2, 1.3488e-17, 21.075, 21.90
9, 21.036
wfc3,uvis1,f390m,mjd#55008,aper#6.0, 3897.2, 2.5524e-18, 22.883, 23.62
1, 23.545
wfc3,uvis1,f390w,mjd#55008,aper#6.0, 3923.7, 5.0142e-19, 24.649, 25.37
3, 25.174
wfc3,uvis1,f395n,mjd#55008,aper#6.0, 3955.2, 5.9589e-18, 21.962, 22.66
8, 22.712
wfc3,uvis1,f410m,mjd#55008,aper#6.0, 4109.0, 2.3481e-18, 22.973, 23.59
7, 23.771
```

Values can also be written into an astropy table.

In [13]:

```
tbl = Table([oms, pivots, bws, pfs, st, ab, vm],
            names=['Obsmode', 'Pivot Wave', 'Bandwidth', 'Photflam', 'STMag', 'A
```

We'll also round columns to a smaller number of decimals.

In [14]:

```
for col in tbl.itercols():
    if col.name == 'Photflam':
        col.info.format = '.4e'
    elif col.info.dtype.kind == 'f':
        col.info.format = '.3f'
```

Let's view our astropy table:

In [15]:

```
1 tbl
```

Out[15]:

Table length=42

Obsmode	Pivot Wave	Bandwidth	Photflam	STMag	ABMag	VegaMag
str36	float64	float64	float64	float64	float64	float64
wfc3,uvis1,f200lp,mjd#55008,aper#6.0	4971.860	1742.198	4.9157e-20	27.171	27.381	26.931
wfc3,uvis1,f218w,mjd#55008,aper#6.0	2228.039	128.941	1.4594e-17	20.990	22.942	21.278
wfc3,uvis1,f225w,mjd#55008,aper#6.0	2372.053	177.430	4.5688e-18	22.251	24.067	22.430
wfc3,uvis1,f275w,mjd#55008,aper#6.0	2709.689	164.435	3.2206e-18	22.630	24.158	22.677
wfc3,uvis1,f280n,mjd#55008,aper#6.0	2832.862	200.689	5.7472e-17	19.501	20.932	19.516
wfc3,uvis1,f300x,mjd#55008,aper#6.0	2820.469	316.561	1.4093e-18	23.527	24.968	23.565
wfc3,uvis1,f336w,mjd#55008,aper#6.0	3354.492	158.422	1.2848e-18	23.628	24.692	23.527
wfc3,uvis1,f343n,mjd#55008,aper#6.0	3435.151	86.713	2.5672e-18	22.876	23.889	22.754
wfc3,uvis1,f350lp,mjd#55008,aper#6.0	5873.870	1490.060	5.1638e-20	27.118	26.965	26.810
wfc3,uvis1,f373n,mjd#55008,aper#6.0	3730.170	18.343	1.3488e-17	21.075	21.909	21.036
...	...	...	...	...	...	...
wfc3,uvis1,f665n,mjd#55008,aper#6.0	6655.876	42.191	1.9774e-18	23.160	22.736	22.492
wfc3,uvis1,f673n,mjd#55008,aper#6.0	6765.939	41.943	2.1926e-18	23.048	22.588	22.343
wfc3,uvis1,f680n,mjd#55008,aper#6.0	6877.596	112.013	6.8241e-19	24.315	23.820	23.556
wfc3,uvis1,f689m,mjd#55008,aper#6.0	6876.755	207.613	3.7208e-19	24.973	24.479	24.196
wfc3,uvis1,f763m,mjd#55008,aper#6.0	7614.371	229.425	3.8291e-19	24.942	24.226	23.837
wfc3,uvis1,f775w,mjd#55008,aper#6.0	7651.363	419.719	2.0922e-19	25.599	24.872	24.480
wfc3,uvis1,f814w,mjd#55008,aper#6.0	8039.056	666.760	1.4994e-19	25.960	25.126	24.698
wfc3,uvis1,f845m,mjd#55008,aper#6.0	8439.057	260.304	4.5207e-19	24.762	23.823	23.316

Obsmode	Pivot Wave	Bandwidth	Photflam	STMag	ABMag	VegaMag
wfc3,uvis1,f850lp,mjd#55008,aper#6.0	9176.126	470.529	3.7052e-19	24.978	23.857	23.326
wfc3,uvis1,f953n,mjd#55008,aper#6.0	9530.579	71.190	8.0946e-18	21.630	20.426	19.803

We can finally save the table as a .txt file.

In [16]:

```
tbl.write('uvis_zp_tbl.txt', format='ascii.commented_header')
```

7. Create and plot 'total system throughput' tables

The function below returns a tuple containing two objects, the first being an array of wavelengths, and the second being the throughput at each of those wavelengths.

In [17]:

```
def calculate_bands(bp, save=False):
    # Pass in bandpass object as bp
    waves = bp.waveset
    throughput = bp(waves)

    if save:
        tmp = Table([waves, throughput], names=['WAVELENGTH', 'THROUGHPUT'])
        tmp.write(','.join(bp.obsmode.modes)+'.txt', format='ascii.commented_hea

    return (waves, throughput)
```

We'll calculate the throughput table for WFC3/UVIS1 in F200LP.

In [18]:

```
obsmode = 'wfc3,uvis1,f200lp'
bp = stsyn.band(obsmode)
wl, tp = calculate_bands(bp)
```

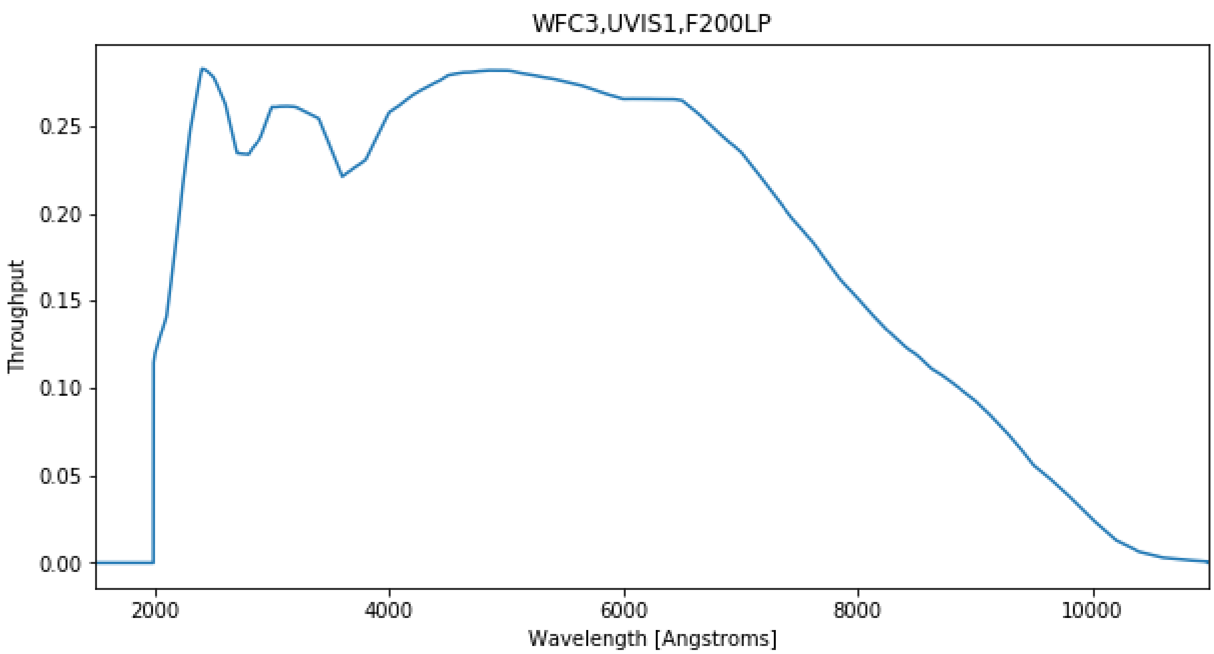
Now, let's plot our results.

In [19]:

```
fig = plt.figure(figsize=(10,5))
plt.plot(wl, tp)
plt.xlim(1500, 11000)
plt.xlabel('Wavelength [Angstroms]')
plt.ylabel('Throughput')
plt.title('WFC3,UVIS1,F200LP')
```

Out[19]:

```
Text(0.5, 1.0, 'WFC3,UVIS1,F200LP')
```



To save the curve in an ascii table, simply pass the argument `save=True` :

In [20]:

```
calculate_bands(bp, save=True)
```

Out[20]:

```
(<Quantity [  500.,  1000.,  1010., ..., 20002., 30000., 30010.] Angst
rom>,
 <Quantity [0., 0., 0., ..., 0., 0., 0.]>)
```

To save curves for all obsmodes defined in [Section 3](#) in the input list, we can loop through detectors and filters.

In [21]:

```
for det in detectors:
    for filt in filtnames:
        obsmode = 'wfc3,{},{}'.format(det, filt)
        bp = stsyn.band(obsmode)
        calculate_bands(bp, save=True)
```

```
WARNING: AstropyDeprecationWarning: wfc3,uvis1,f200lp.txt already exis
ts. Automatically overwriting ASCII files is deprecated. Use the argum
ent 'overwrite=True' in the future. [astropy.io.ascii.ui]
WARNING:astropy:AstropyDeprecationWarning: wfc3,uvis1,f200lp.txt alrea
dy exists. Automatically overwriting ASCII files is deprecated. Use th
e argument 'overwrite=True' in the future.
```

In addition, we'll create a directory called `obsmodes_curves` and move all the saved files to that directory.

In [22]:

```
! mkdir obsmodes_curves
! mv wfc3*txt obsmodes_curves
! ls obsmodes_curves
```

```
wfc3,uvis1,f200lp.txt wfc3,uvis1,f438w.txt  wfc3,uvis1,f645n.txt
wfc3,uvis1,f218w.txt  wfc3,uvis1,f467m.txt  wfc3,uvis1,f656n.txt
wfc3,uvis1,f225w.txt  wfc3,uvis1,f469n.txt  wfc3,uvis1,f657n.txt
wfc3,uvis1,f275w.txt  wfc3,uvis1,f475w.txt  wfc3,uvis1,f658n.txt
wfc3,uvis1,f280n.txt  wfc3,uvis1,f475x.txt  wfc3,uvis1,f665n.txt
wfc3,uvis1,f300x.txt  wfc3,uvis1,f487n.txt  wfc3,uvis1,f673n.txt
wfc3,uvis1,f336w.txt  wfc3,uvis1,f502n.txt  wfc3,uvis1,f680n.txt
wfc3,uvis1,f343n.txt  wfc3,uvis1,f547m.txt  wfc3,uvis1,f689m.txt
wfc3,uvis1,f350lp.txt wfc3,uvis1,f555w.txt  wfc3,uvis1,f763m.txt
wfc3,uvis1,f373n.txt  wfc3,uvis1,f600lp.txt wfc3,uvis1,f775w.txt
wfc3,uvis1,f390m.txt  wfc3,uvis1,f606w.txt  wfc3,uvis1,f814w.txt
wfc3,uvis1,f390w.txt  wfc3,uvis1,f621m.txt  wfc3,uvis1,f845m.txt
wfc3,uvis1,f395n.txt  wfc3,uvis1,f625w.txt  wfc3,uvis1,f850lp.txt
wfc3,uvis1,f410m.txt  wfc3,uvis1,f631n.txt  wfc3,uvis1,f953n.txt
```

8. Conclusions

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

- Calculating zeropoints and other photometric properties using `stsynphot` .
- Creating, plotting, and saving 'total system throughput' tables.

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)](https://stsci.service-now.com/hst).

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

About this Notebook

Authors: Varun Bajaj, Jennifer Mack; WFC3 Instrument Team

Updated on: 2021-09-08

Citations

If you use `numpy`, `astropy`, `synphot`, or `stsynphot` 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)](https://www.scipy.org/citing.html#numpy)
- [Citing `astropy` (https://www.astropy.org/acknowledging.html)](https://www.astropy.org/acknowledging.html)
- [Citing `synphot` (https://synphot.readthedocs.io/en/latest/)](https://synphot.readthedocs.io/en/latest/)
- [Citing `stsynphot` (https://stsynphot.readthedocs.io/en/latest/index.html)](https://stsynphot.readthedocs.io/en/latest/index.html)

[Top of Page](#)

STScI | SPACE TELESCOPE SCIENCE INSTITUTE

In [ ]:

```
1
```