CTIO

February 19, 2019

1 Données slitless CTIO

```
In [1]: # %matplotlib notebook
        %matplotlib inline
```

```
In [2]: import numpy as N
        import matplotlib.pyplot as P

        import scipy.interpolate as SI

        import astropy.io.fits as F
        import astropy.nddata as AN
        import astropy.modeling as AM
        import astropy.visualization as AV

        from __future__ import print_function
```

```
In [3]: P.rcParams['figure.figsize'] = (10, 6)
        P.rcParams['image.origin'] = True

        # import mpld3
        # mpld3.enable_notebook()
```

1.1 HD111980 (20170530_130)

This exposure was done w/ a Ronchi dispersor, hence the strong defocus in the red part.

```
In [4]: hdu = F.open("reduc_20170530_130.fits")
        hdu.info()
```

```
Filename: reduc_20170530_130.fits
No.    Name      Ver    Type      Cards   Dimensions   Format
  0  PRIMARY       1 PrimaryHDU     134   (2048, 2048)   float64
```

```
In [5]: hdu[0].header
```

Out[5]:
```
SIMPLE  =                    T / conforms to FITS standard
BITPIX  =                  -64 / array data type
NAXIS   =                    2 / number of array dimensions
NAXIS1  =                 2048
NAXIS2  =                 2048
COMMENT   FITS (Flexible Image Transport System) format is defined in 'Astronomy
COMMENT   and Astrophysics', volume 376, page 359; bibcode: 2001A&A...376..359H
OBJECT  = 'HD11980 '           / Name of object observed
OBSERVER= '        '           / observer
PROPID  = '        '           / Proposal ID
RECID   = 'ct36.20170531.024522' / NOAO Archibe record ID
PROPID  = '        '           / Proposal ID
IMAGETYP= 'object  '           / Type of picture (object, dark, comp, etc)
CCDSUM  = '1 1     '           / On chip summation (X, Y)
XSTART  =                    1 / start of roi in X
YSTART  =                    1 / start of roi in Y
XLENGTH =                 2048 / length of roi in X
YLENGTH =                 2048 / length of roi in Y
UTSHUT  = '2017-05-31T02:45:22.598' / UT of shutter open
UT      = '02:45:22.598'       / UT of TCS coordinates
DATE-OBS= '2017-05-31T02:45:22.598' / date of observations start
DATE    = '2017-05-31T02:46:24' / file creation date (YYYY-MM-DDThh:mm:ss UT)
NAMPSYX = '2 2     '           / Num amps in y and x (eg. '2 2=quad')
AMPLIST = '11 12 21 22'        / Readout order in y,x
GTRON11 =                  12. / (e-) predicted read noise, lower left
GTRON12 =                  12. / (e-) predicted read noise, lower right
GTRON21 =                  12. / (e-) predicted read noise, upper left
GTRON22 =                  12. / (e-) predicted read noise, upper right
GTGAIN11=                   3. / (e-/ADU) predicted gain, lower left
GTGAIN12=                   3. / (e-/ADU) predicted gain, lower right
GTGAIN21=                   3. / (e-/ADU) predicted gain, upper left
GTGAIN22=                   3. / (e-/ADU) predicted gain, upper right
ASEC11  = '[1:1084,1:1024]'    / amplifier section Amp11(LL) detID 1
BSEC11  = '[1045:1084,1:1024]' / bias section Amp11(LL) detID 1
CSEC11  = '[1:1034,1:1024]'    / section in full ccd for DSEC Amp11(LL) detID 1
DSEC11  = '[1:1034,1:1024]'    / image section in raw frame Amp11(LL) detID 1
TSEC11  = '[11:1034,1:1024]'   / trim section Amp11(LL) detID 1
ABSEC11 = '[1045:1084,1:1024]' / overscan inside amp
ADSEC11 = '[1:1024,1:1024]'    / detector section only
ASEC12  = '[1085:2168,1:1024]' / amplifier section Amp12(LR) detID 1
BSEC12  = '[1085:1124,1:1024]' / bias section Amp12(LR) detID 1
CSEC12  = '[1035:2068,1:1024]' / section in full ccd for DSEC Amp12(LR) detID 1
DSEC12  = '[1135:2168,1:1024]' / image section in raw frame Amp12(LR) detID 1
TSEC12  = '[1135:2158,1:1024]' / trim section Amp12(LR) detID 1
ABSEC12 = '[1:40,1:1024]'      / overscan inside amp
ADSEC12 = '[1025:2048,1:1024]' / detector section only
ASEC21  = '[1:1084,1025:2048]' / amplifier section Amp21(UL) detID 1
BSEC21  = '[1045:1084,1025:2048]' / bias section Amp21(UL) detID 1
```

```
CSEC21  = '[1:1034,1025:2048]' / section in full ccd for DSEC Amp21(UL) detID 1
DSEC21  = '[1:1034,1025:2048]' / image section in raw frame Amp21(UL) detID 1
TSEC21  = '[11:1034,1025:2048]' / trim section Amp21(UL) detID 1
ABSEC21 = '[1045:1084,1:1024]' / overscan inside amp
ADSEC21 = '[1:1024,1025:2048]' / detector section only
ASEC22  = '[1085:2168,1025:2048]' / amplifier section Amp22(UR) detID 1
BSEC22  = '[1085:1124,1025:2048]' / bias section Amp22(UR) detID 1
CSEC22  = '[1035:2068,1025:2048]' / section in full ccd for DSEC Amp22(UR) detID
DSEC22  = '[1135:2168,1025:2048]' / image section in raw frame Amp22(UR) detID 1
TSEC22  = '[1135:2158,1025:2048]' / trim section Amp22(UR) detID 1
ABSEC22 = '[1:40,1:1024]'      / overscan inside amp
ADSEC22 = '[1025:2048,1025:2048]' / detector section only
ROISEC00= '[1:2048,1:2048]'    / roi section
DETECTOR= 'Tek2K_3 '           / Detector Identifier
FPA     = 'SITE2K  '           / focal plan array
REXPTIME=                  60. / requested exposure time in secs
EXPTIME =                  60. / Exposure time in secs
DARKTIME=                  60. / Total elapsed time in secs
NIMAGES =                    1 / number of images requested in sequence
PIXELT  = '25000.000000'       / (ns) unbinned pixel read time
DHEINF  = 'MNSN torrent hardware' / controller info
DHEFIRM = '/home/observer/panview/fpas/_biw/config/DETECTOR/site2k_sequencer.uc'
PIXTIME = '25      '           / pixel time (usecs)
POWSTAT = '3.000   '           / power supplies status (3=OK)
CCDSETP = '163.000 '           / ccd temperature setpoint
CCDTEMP = '161.500 '           / CCD temperature
NECKTEMP= '134.000 '           / dewar NECK temperature
HEATERSP= '17.323  '           / Heater power percent.
VDDA    = '24.049  '           / bias output amplifier A
VDDB    = '23.959  '           / bias output amplifier B
VDDC    = '24.408  '           / bias output amplifier C
VDDD    = '24.184  '           / bias output amplifier D
VRDA    = '13.855  '           / Reset Drain amplifier A
VRDB    = '13.825  '           / Reset Drain amplifier B
VRDC    = '14.064  '           / Reset Drain amplifier C
VRDD    = '13.975  '           / Reset Drain amplifier D
LGA     = '-2.022  '           / Reset Drain amplifier A
LGB     = '-2.052  '           / Reset Drain amplifier B
LGC     = '-1.972  '           / Reset Drain amplifier C
LGD     = '-1.952  '           / Reset Drain amplifier D
SLOT00  = 'LCB 0x188538 2.240000' / dhe board: <type> <serial> <firmware>
SLOT01  = 'PSM 0x45834F 2.210000' / dhe board: <type> <serial> <firmware>
SLOT02  = 'CFG 0xNONE 2.240000' / dhe board: <type> <serial> <firmware>
SLOT03  = 'PIX 0xNONE 2.210000' / dhe board: <type> <serial> <firmware>
SLOT04  = 'CCDAFE 0x188167 18805C 2.210000' / dhe board: <type> <serial> <firmwa
SLOT07  = 'CB 0xNONE 2.240000' / dhe board: <type> <serial> <firmware>
SLOT02  = 'TSM 0x3C6784 NONE'  / dhe board: <type> <serial> <firmware>
VANPLUS =             10.54422 / analog voltage plus
```

```
VANMINU =             -10.5137 / analog voltage minus
FPGATEMP=             33.00492 / torrent fpga temperature
ID      = '[ct36.20170531.024522]' / ID
OBSERVAT= 'CTIO    '           / Origin of data
TELESCOP= 'CTIO 0.9 meter telescope' / Specific system
TELID   = 'ct36    '           / CTIO 0.9 meter telescope
TCS-TIME= '2017-05-31T02:45:22.38' / date of observation start
UT      = '02:45:22.38'        / UT of TCS coords
RA      = '12:53:8.11'         / ra
DEC     = '-18:33:27.78'       / dec
EPOCH   =                2000. / epoch
ZD      =                26.15 / zenith distance
HA      = '01:44:11.73'        / hour angle
ST      = '14:37:19.84'        / sidereal time
AIRMASS =                1.114 / airmass
ALT     = '-45.84  '           / altitud
TELFOCUS=               12450. / telescope focus
WEATIME = '2017-05-31 02:45:01' / weather timestamp
OUTTEMP =                  8.5 / outside temp (C)
OUTHUM  =                   25 / outside humidity (%)
OUTPRESS=                 784. / outside pressure (hPa)
WNDSPEED=                  3.8 / wind speed (mph)
WNDDIR  =                  157 / wind dir (degrees)
SEETIME = '2017-05-31 02:46:20' / seeing timestamp
SEEING  =                0.593 / seeing
SAIRMASS=                1.011 / seeing airmass
PANID   = '_biw    '           / PAN identification
COMMENT image
COMMENT Image is trimmed
FILTER1 = 'dia     '           / Filter in wheel 1
FNAME1  = 'DIAFRAGM'           / Full name of filter in wheel 1
FILTER2 = 'Ron400  '           / Filter in wheel 2
FNAME2  = 'Ronchi400'          / Full name of filter in wheel 2
FILTERS = 'dia Ron400'         / Filter positions
INSTRUME= 'cfccd   '           / cassegrain direct imager
XPIXSIZE=                0.401 / Pixel size in X (arcsecs/pix)
YPIXSIZE=                0.401 / Pixel size in Y (arcsecs/pix)
TEST    =                   2. / my keyword
```

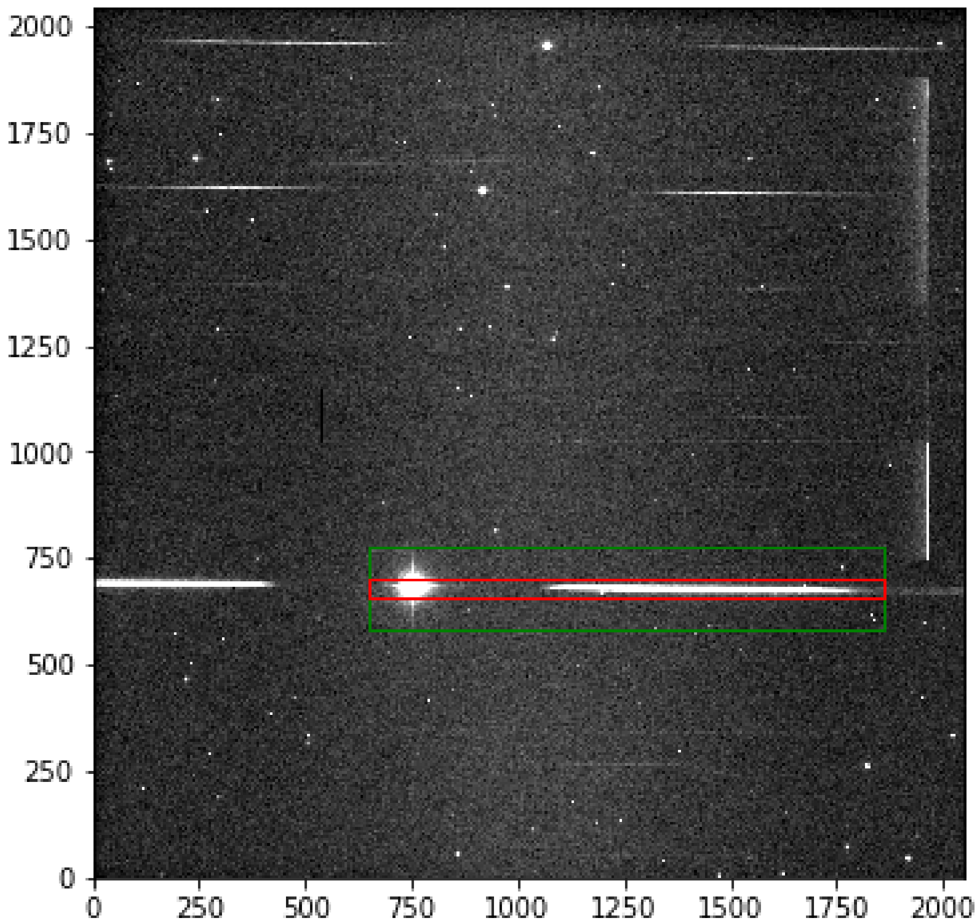
In [6]:
```
# fima = N.ma.masked_greater(hdu[0].data, 65000)  # full image, masked
fima = hdu[0].data  # full image
```

1.1.1 Background subtraction

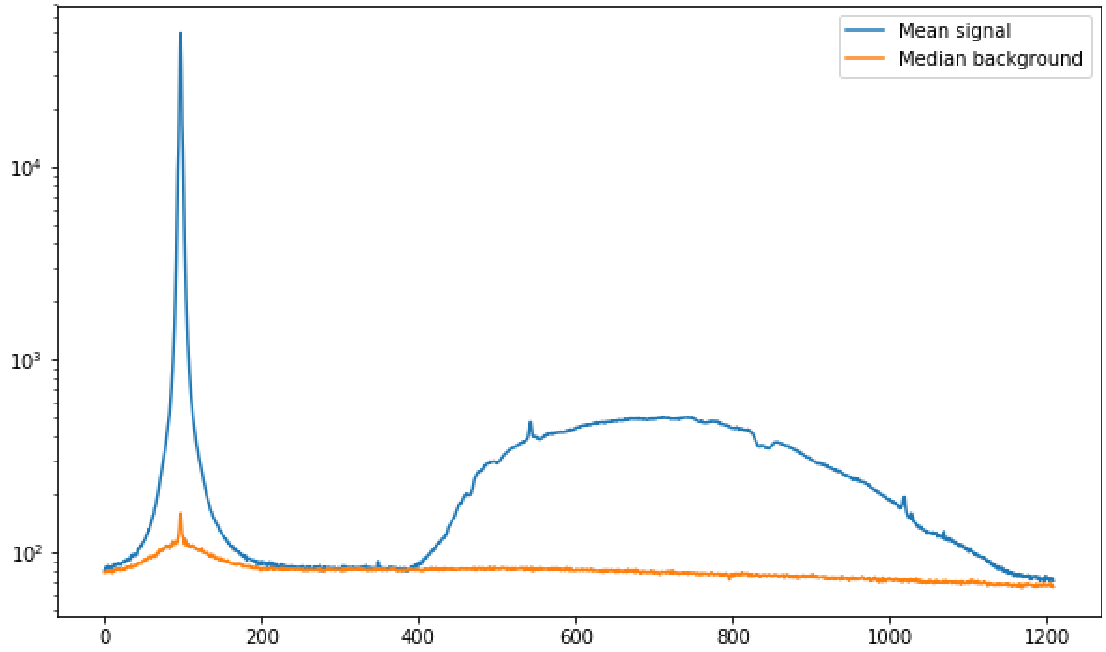
In [7]:
```
center = (1255, 680)  # x, y
size = (49, 1209)     # dy, dx

ima = AN.Cutout2D(fima, center, size, copy=True)         # Cutout image
```

```
bkg = AN.Cutout2D(fima, center, (size[0] * 4, size[1]))  # Larger cutout for background
```

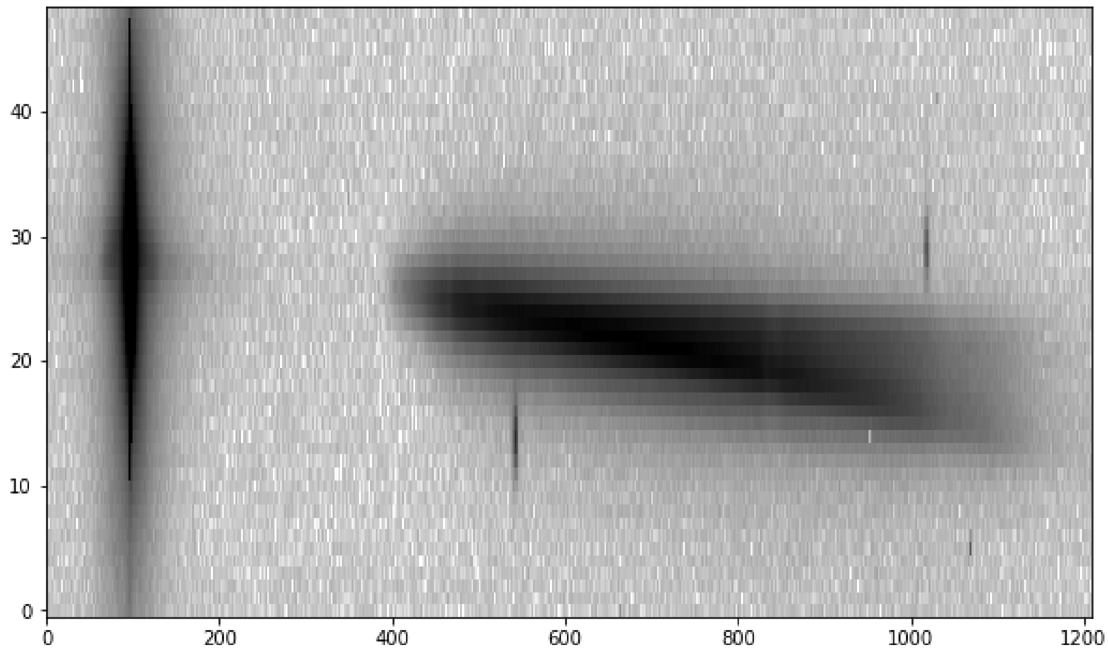
In [8]:
```
fig, ax = P.subplots(1, 1)
ax.imshow(fima, cmap='gray',
          norm=AV.ImageNormalize(fima, interval=AV.PercentileInterval(99)))
bkg.plot_on_original(ax, color='green')
ima.plot_on_original(ax, color='red');
```



In [9]:
```
medbkg = N.ma.median(bkg.data, axis=0)              # Median 1D background

fig, ax = P.subplots(1, 1)
ax.plot(ima.data.mean(axis=0), label="Mean signal")  # X-disp. 1D sum
ax.plot(medbkg, label='Median background')
ax.set_yscale('log')
ax.legend();
```

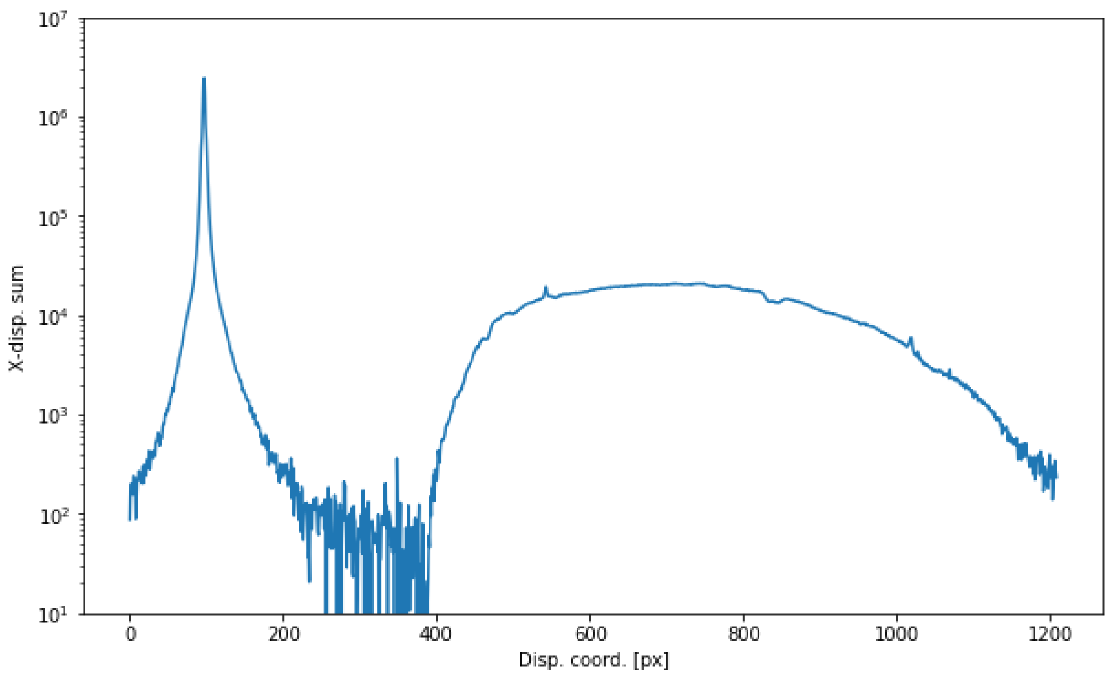



```
In [10]: ima.data -= medbkg  # Median background subtraction

In [11]: fig, ax = P.subplots(1, 1)
         im = ax.imshow(ima.data, cmap='gray_r',
                        norm=AV.ImageNormalize(ima.data, interval=AV.PercentileInterval(99), str
         ax.set_aspect('auto');

/usr/local/lib/python2.7/dist-packages/astropy/visualization/stretch.py:267: RuntimeWarning: inv
  np.log(values, out=values)
/usr/local/lib/python2.7/dist-packages/matplotlib/colors.py:504: RuntimeWarning: invalid value e
  xa[xa < 0] = -1
```




```
In [12]: spec = ima.data.sum(0)  # Cross-dispersion sum
```

```
In [13]: fig, ax = P.subplots(1, 1)
         ax.plot(spec)
         ax.set(yscale='log', ylim=[1e1, 1e7], ylabel='X-disp. sum', xlabel='Disp. coord. [px]')
```



1.1.2 Spectral trace

0th-order

```
In [14]: xf0, yf0 = 748.005, 684.256                   # J. Neveu estimate in full image
         x0, y0 = ima.to_cutout_position((xf0, yf0))  # Position in cutout
         print("0th-order position (cutout):", x0, y0)
```

```
0th-order position (cutout): 97.005 28.256
```

1st-order

```
In [15]: thetadeg = -0.719                          # Rotation [degree] (see below)
         lmin, lmax = 325., 1086.                   # Wavelength coverage [nm] (see below)

         # Curvilinear offset wrt 0th-order along tilted spectral trace [px]
         fds, flbda = N.loadtxt('dispersion_relation.txt', unpack=True)
         sel = (flbda >= lmin) & (flbda <= lmax)
         lbda = flbda[sel]
         print("Spectral coverage: {:.1f}--{:.1f} nm".format(lbda[0], lbda[-1]))

         # Cartesian x- and y-offsets wrt 0th-order [px]
         dx = fds[sel] * N.cos(N.radians(thetadeg))
         dy = fds[sel] * N.sin(N.radians(thetadeg))
         print("Cartesian offset wrt 0th-order: ({:+.1f},{:+.1f})--({:+.1f},{:+.1f}) px".format(
```

```
Spectral coverage: 325.7--1085.9 nm
Cartesian offset wrt 0th-order: (+302.0,-3.8)--(+1107.9,-13.9) px
```

Wavelength solution

```
In [16]: lbdaofx = SI.InterpolatedUnivariateSpline(dx + x0, lbda)  # Wavelength [nm] as a functi
         xoflbda = SI.InterpolatedUnivariateSpline(lbda, dx + x0)  # X-position in cutout as a f
```

X-dispersion profile

```
In [17]: y = N.arange(ima.shape[0])  # X-disp. coordinate [px]
         z = ima.data[:, 600]        # X-disp. profile
         print("Total flux:", z.sum())
```

```
Total flux: 17780.263211400994
```

```
In [18]: fitter = AM.fitting.LevMarLSQFitter()
         gauss = AM.models.Gaussian1D(amplitude=z.max(), mean=z.argmax(), stddev=2)
         gfit = fitter(gauss, y, z)
         print(fitter.fit_info['message'])
         print(gfit)
         print("Total flux: {} ({:.2%} error)".format(gfit(y).sum(), gfit(y).sum()/z.sum() - 1))
```

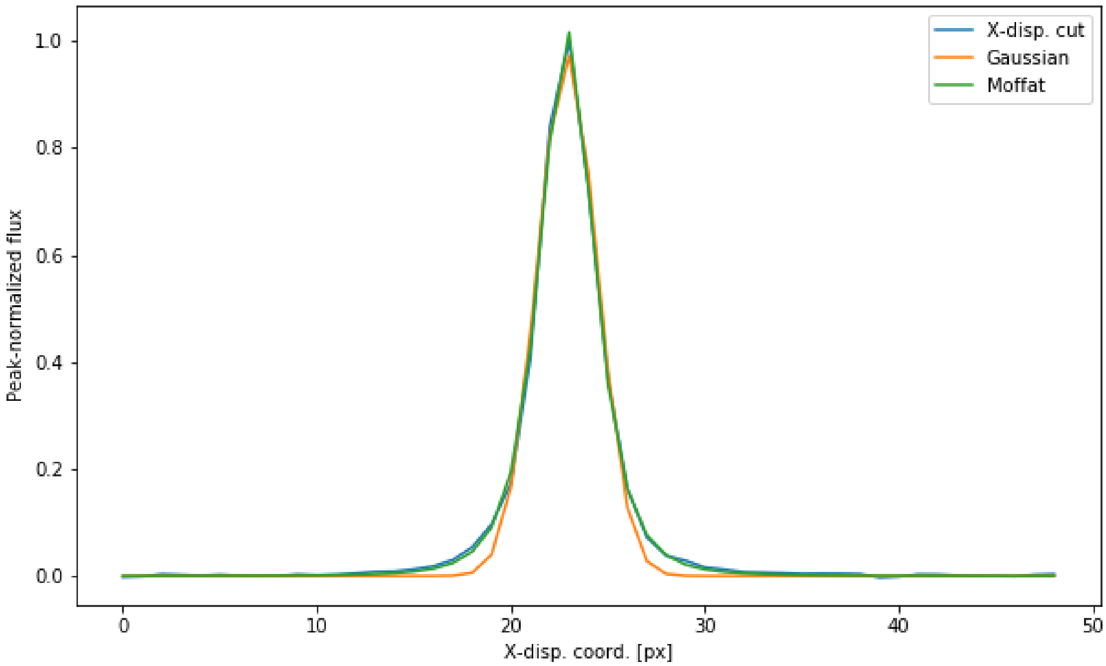
```
Both actual and predicted relative reductions in the sum of squares
  are at most 0.000000
Model: Gaussian1D
Inputs: (u'x',)
Outputs: (u'y',)
Model set size: 1
Parameters:
        amplitude            mean              stddev
    ----------------- ------------------ ------------------
    4190.510142321903 22.897211847060973 1.5392913100225976
Total flux: 16168.7947443 (-9.06% error)
```

WARNING: analytic derivatives of Moffat1D in astropy.modeling (2.0.9 and 3.0.5) are wrong (see issue https://github.com/astropy/astropy/issues/8094).

```
In [19]: moffat = AM.models.Moffat1D(amplitude=z.max(), x_0=z.argmax(), gamma=3, alpha=2)
         mfit = fitter(moffat, y, z, estimate_jacobian=True)  # Workaround to issue #8094
         print(fitter.fit_info['message'])
         print(mfit)
         print("Total flux: {} ({:.2%} error)".format(mfit(y).sum(), mfit(y).sum()/z.sum() - 1))
```

```
Both actual and predicted relative reductions in the sum of squares
  are at most 0.000000 and the relative error between two consecutive iterates is at
  most 0.000000
Model: Moffat1D
Inputs: (u'x',)
Outputs: (u'y',)
Model set size: 1
Parameters:
        amplitude            x_0               gamma              alpha
    ----------------- ------------------ ------------------ -----------------
    4386.936307586843 22.887434087381305 2.6015236725589057 2.066102607639578
Total flux: 17479.3771599 (-1.69% error)
```

```
In [20]: fig, ax = P.subplots(1, 1)
         ax.plot(y, z / z.max(), label="X-disp. cut")
         lg, = ax.plot(y, gfit(y) / z.max(), label="Gaussian")
         lm, = ax.plot(y, mfit(y) / z.max(), label="Moffat")
         ax.set(xlabel="X-disp. coord. [px]", ylabel="Peak-normalized flux")
         ax.legend();
```



In [21]:
```
xs = N.arange(390, 1209, 5)  # Linearly sampled dispersion offset [px]
parameters = N.array([ fitter(gauss, y, ima.data[:, x]).parameters for x in xs ])  # am
# parameters = N.array([ fitter(moffat, y, ima.data[:, x], estimate_jacobian=True).para
#                         for x in xs ])                                                #

amplitudes = parameters[:, 0]
yoffsets = parameters[:, 1] - y0  # X-disp. offset wrt 0th-order [px]
sigmas = parameters[:, 2]         # Sigma [px]

fluxes = amplitudes * sigmas * N.sqrt(2 * N.pi)
```

Effective wavelength coverage

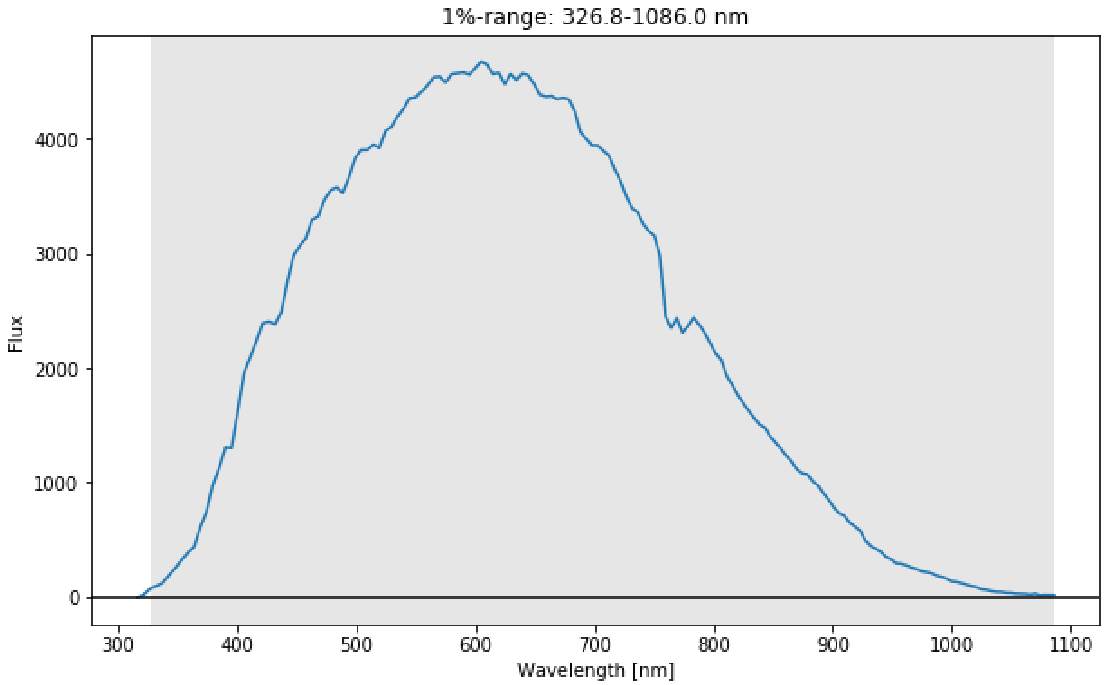
In [22]:
```
fmin = 0.01                                    # Minimal flux fraction
lsel = fluxes >= (fmin * fluxes.max())
xmin, xmax = xs[lsel][0], xs[lsel][-1]
lmin, lmax = lbdaofx([xmin, xmax])
print("{:.0%}-range: {}--{} px = {:.1f}--{:.1f} nm".format(fmin, xmin, xmax, lmin, lmax
```

```
1%-range: 400--1205 px = 326.8--1086.0 nm
```

In [23]:
```
fxofl = SI.InterpolatedUnivariateSpline(lbdaofx(xs), fluxes)
```

In [24]:
```
fig, ax = P.subplots(1, 1)
ax.plot(lbdaofx(xs), amplitudes)
```

```
#ax.plot(lbda, fxofl(lbda))
ax.axhline(fmin, color='k')
ax.axvspan(lbdaofx(xmin), lbdaofx(xmax), fc='0.9')
ax.set(xlabel="Wavelength [nm]", ylabel="Flux",
       title="{:.0%}-range: {:.1f}-{:.1f} nm".format(fmin, lmin, lmax));
```



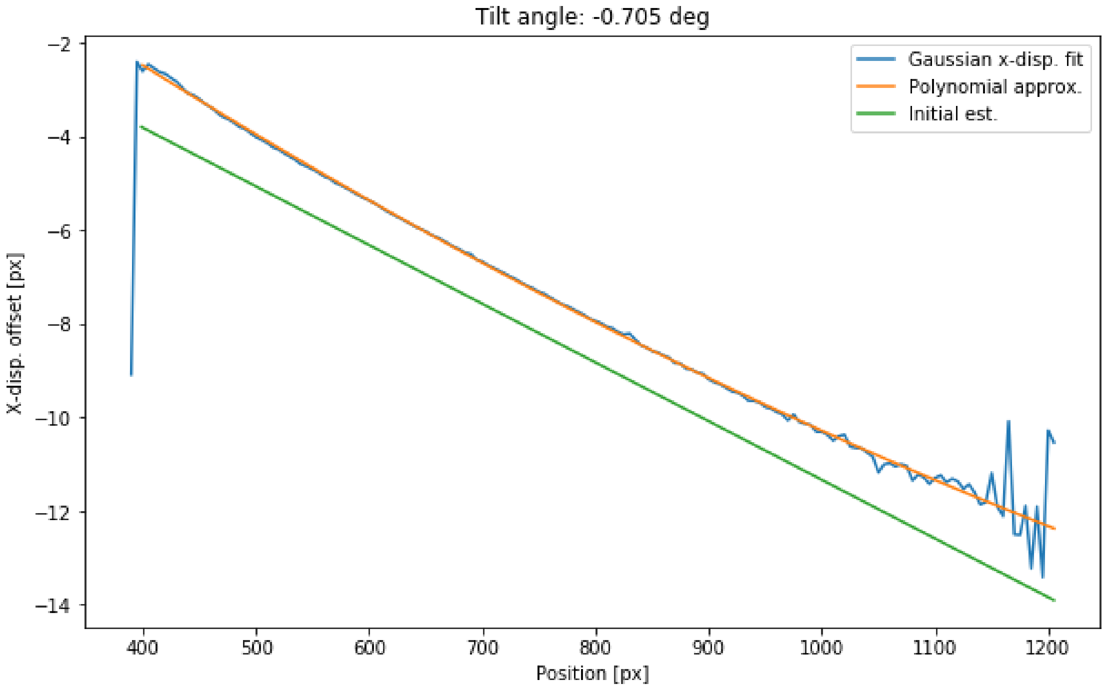
Linear tilt angle

In [25]:
```
from astropy.stats import sigma_clip
robust_fitter = AM.fitting.FittingWithOutlierRemoval(fitter, sigma_clip, niter=3, sigma
```

In [26]:
```
# Robust linear adjustment
# _, dyofx = robust_fitter(AM.models.Linear1D(), xs[lsel], yoffsets[lsel])
# Robust quadratic adjustment
_, dyofx = robust_fitter(AM.models.Legendre1D(2), xs[lsel], yoffsets[lsel])
ys = dyofx(xs)
theta = N.arctan2(N.diff(ys[lsel]), N.diff(xs[lsel])).mean()
print("Tilt angle: {:.3f} deg".format(N.degrees(theta)))

fig, ax = P.subplots(1, 1)
ax.plot(xs, yoffsets, label='Gaussian x-disp. fit')
ax.plot(xs[lsel], dyofx(xs[lsel]), label='Polynomial approx.')
ax.plot(xoflbda(lbda), dy, label="Initial est.")
ax.set(xlabel="Position [px]", ylabel="X-disp. offset [px]",
       title="Tilt angle: {:.3f} deg".format(N.degrees(theta)))
ax.legend();
```

```
WARNING: Model is linear in parameters; consider using linear fitting methods. [astropy.modeling
```

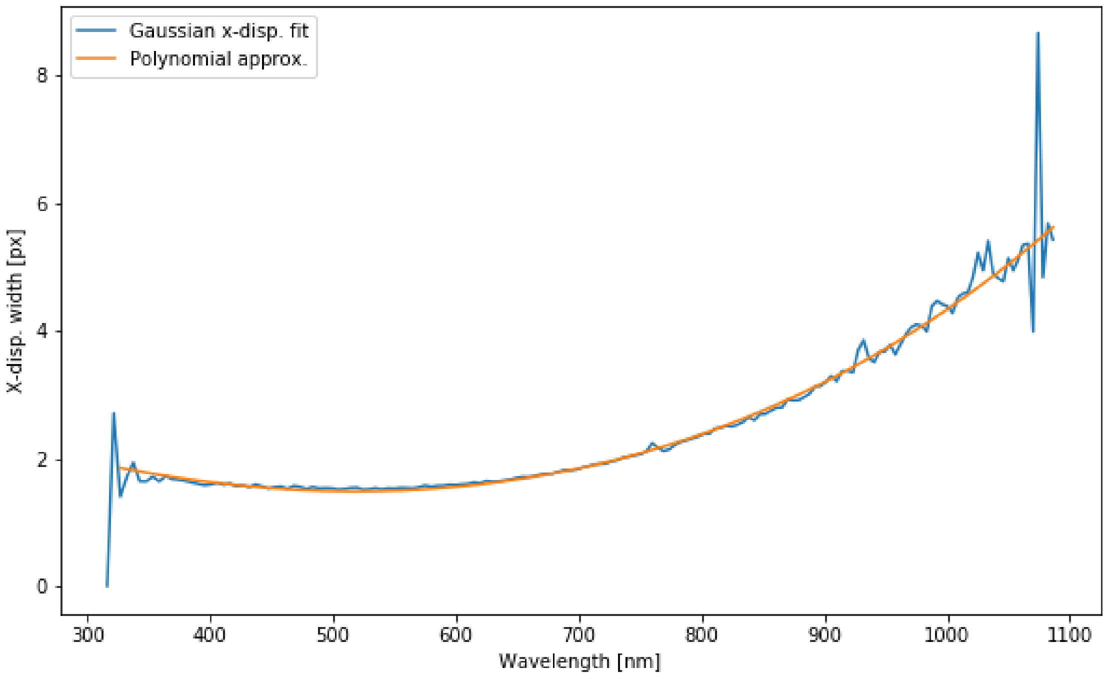
```
Tilt angle: -0.705 deg
```



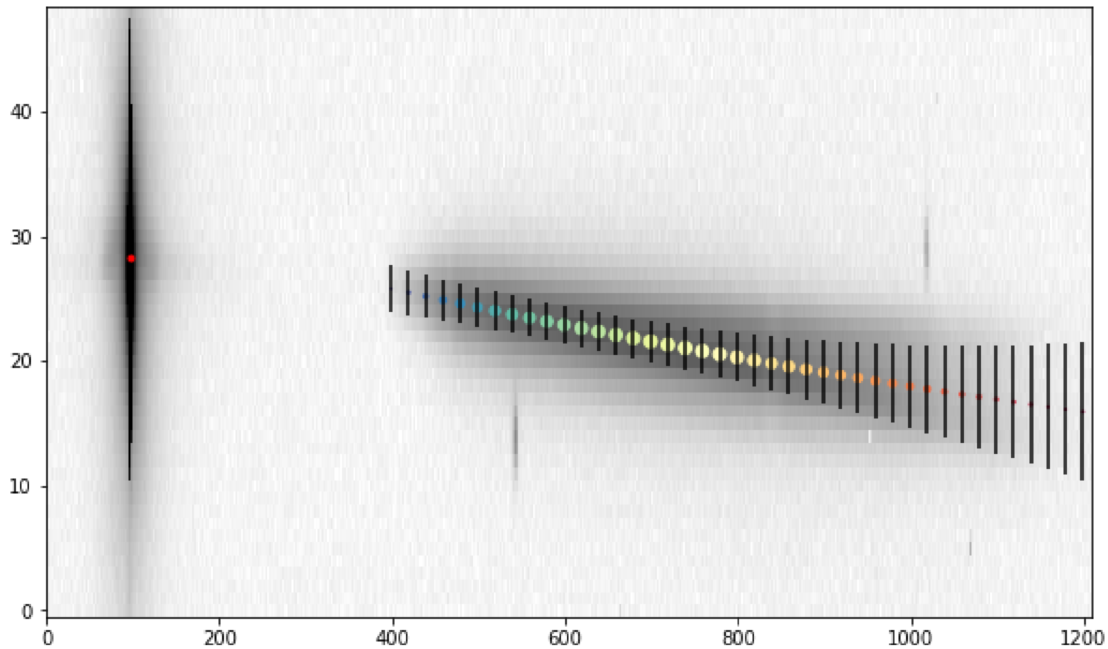
NOTE: there's a one px offset between Jeremy's and present cross-dispersion position.

X-disp. width

```
In [27]: _, sigofx = robust_fitter(AM.models.Legendre1D(2), xs[lsel], sigmas[lsel])

         fig, ax = P.subplots(1, 1)
         ax.plot(lbdaofx(xs), sigmas, label='Gaussian x-disp. fit')
         ax.plot(lbdaofx(xs[lsel]), sigofx(xs[lsel]), label="Polynomial approx.")
         ax.set(xlabel="Wavelength [nm]", ylabel="X-disp. width [px]")
         ax.legend();
```



In [28]:
```
newdy = dyofx(xoflbda(lbda))  # X-disp. offset [px]
sig = sigofx(xoflbda(lbda))   # X-disp. width [px]

fig, ax = P.subplots(1, 1)
ax.imshow(ima.data, cmap='gray_r',
          norm=AV.ImageNormalize(ima.data, interval=AV.PercentileInterval(99.5), stretc
ax.plot([x0], [y0], 'r.')
ax.scatter(x0 + dx[::20], y0 + newdy[::20], c=lbda[::20], s=fxofl(lbda[::20]) / 100, ma
ax.errorbar(x0 + dx[::20], y0 + newdy[::20], yerr=sig[::20], fmt='none', c='k')
ax.set_aspect('auto');
```



1.1.3 Spectral model

PSF model

```
In [29]: import slitless.fourier as S
         print("Submodules: {}".format(', '.join([ x for x in dir(S) if not x.startswith('__') ]
```

```
Submodules: arrays, fourier, misc, models, pkg, plots, wfc3
```

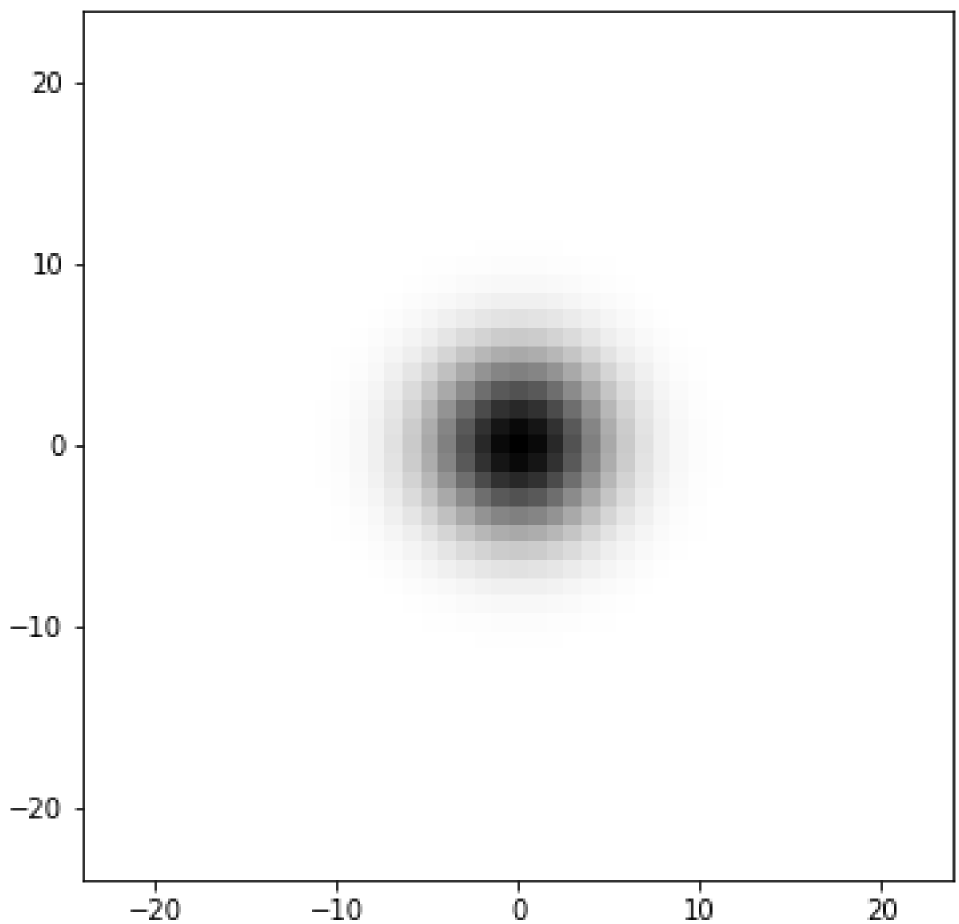
```
In [30]: nima = size[0]
         nlbda = len(lbda)
         print("2D spectrum shape: {} Œ {}".format(nima, nlbda))

         y, x = S.arrays.create_coords((nima, nima), starts='auto')
```

```
2D spectrum shape: 49 Œ 807
```

```
In [31]: _, sigofl = robust_fitter(AM.models.Legendre1D(2), lbdaofx(xs[lsel]), sigmas[lsel])
         sigs = sigofl(lbda)

         psf = S.models.build_cube(x, y, AM.models.Gaussian2D,
                                   x_stddev=sigs, y_stddev=sigs)
         print("PSF cube shape: {}".format(psf.shape))
         print("Normalized:", N.allclose(psf.sum(axis=(-1, -2)), 1))
```

```
PSF cube shape: (807, 49, 49)
Normalized: True
```

```
In [32]: P.imshow(psf[600], extent=(x[0, 0], x[0, -1], y[0, 0], y[-1, 0]), cmap='gray_r');
```



Dispersion law

```
In [33]: disp = dx + 1j * newdy  # Dispersion law
         odisp = int(disp.real.mean()) + 1j * int(disp.imag.mean())
         print("Reference dispersion position: {0.real:+.0f} Œ {0.imag:+.0f} px".format(odisp))
         cdisp = disp - odisp    # Centered dispersion law
```

```
Reference dispersion position: +704 Œ -7 px
```

Simulated spectrum

```
In [34]: dima = S.models.disperse_cube(psf, fxofl(lbda), cdisp)
         print("Simulated dispersed image:", dima.shape)
```

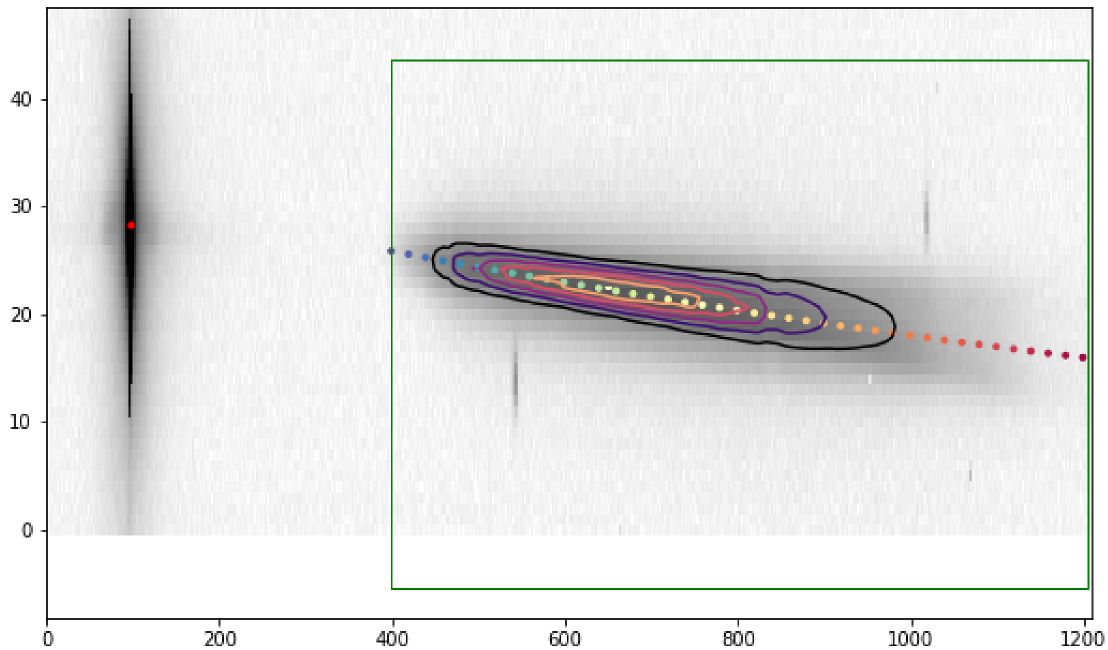
```
Simulated dispersed image: (49, 807)
```

```
In [35]: # Observed and simulated extracted spectral trace
         oima = AN.Cutout2D(ima.data, (x0 + odisp.real, y0 + odisp.imag), (nima, nlbda), mode='p
         sima = AN.Cutout2D(ima.data, (x0 + odisp.real, y0 + odisp.imag - 2), (nima, nlbda), mod
         sima.data = dima
         print("Extracted dispersed image:", sima.shape)
```

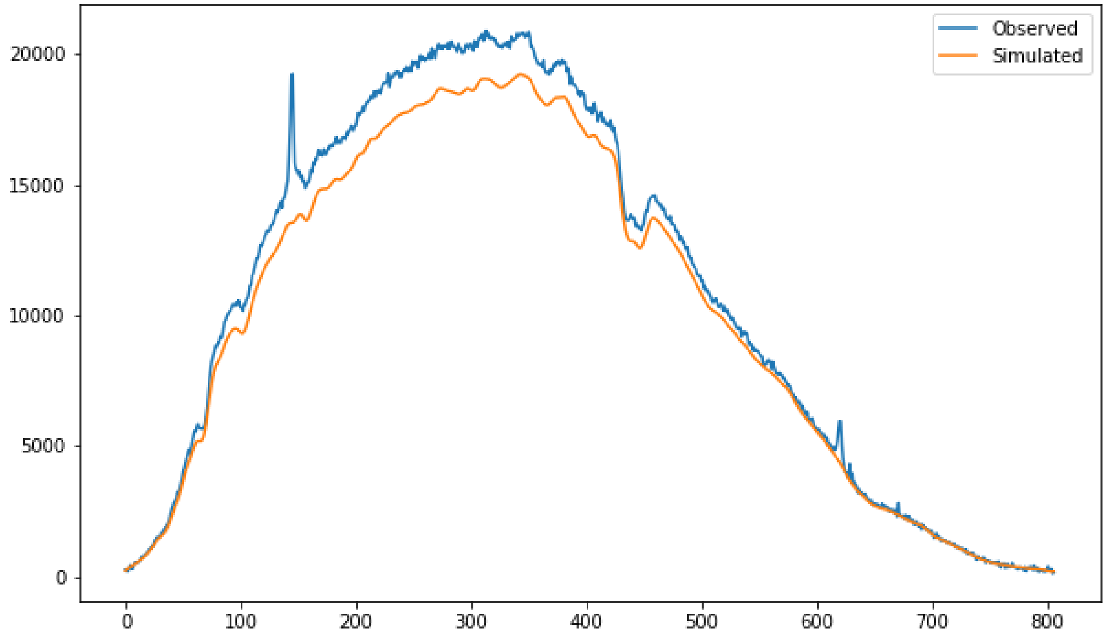
```
Extracted dispersed image: (49, 807)
```

BUG: why a x-disp. offset of ~2 px???

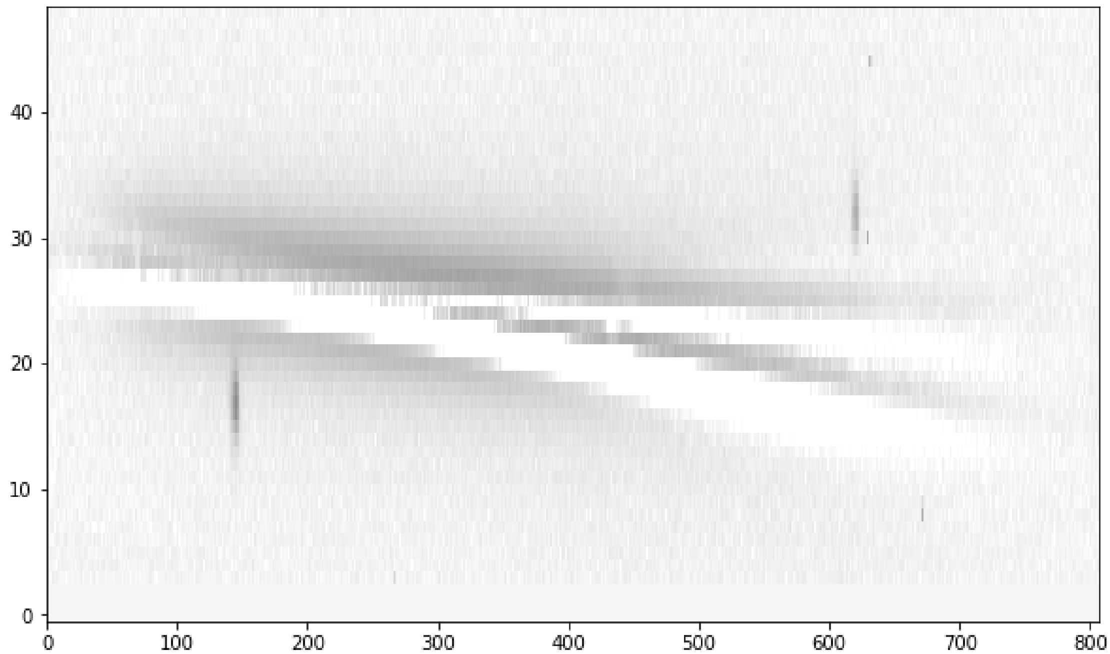
```
In [36]: fig, ax = P.subplots(1, 1)
         ax.imshow(ima.data, cmap='gray_r',
                   norm=AV.ImageNormalize(ima.data, interval=AV.PercentileInterval(99.5), stretc
         ax.plot([x0], [y0], 'r.')
         ax.scatter(x0 + disp.real[::20], y0 + disp.imag[::20], c=lbda[::20], marker='.', cmap='
         sima.plot_on_original(ax, color='green');
         bbox = sima.bbox_original  # ((x, y) lower left, (x, y) upper right)
         ax.contour(sima.data, extent=(bbox[1][0], bbox[1][1], bbox[0][0], bbox[0][1]))
         ax.set_aspect('auto');
```



```
In [37]: fig, ax = P.subplots(1, 1)
         ax.plot(oima.data.sum(axis=0), label='Observed')
         ax.plot(sima.data.sum(axis=0), label='Simulated')
         ax.legend();
```




```
In [38]: fig, ax = P.subplots(1, 1)
         resima = oima.data - sima.data
         ax.imshow(resima, cmap='gray_r',
                   norm=AV.ImageNormalize(ima.data, interval=AV.PercentileInterval(99.5), stretc
         ax.set_aspect('auto');
```



2 Appendix

2.1 Moffat profile

The Moffat (1969A&A.....3..455M) profile is defined as:

$$M(r) = \frac{\beta - 1}{\pi\alpha^2}\left(1 + \frac{r^2}{\alpha^2}\right)^{-\beta}$$

Note this form is a reparameterisation of an uncorrelated bivariate Student distribution. As written, the axisymmetric PSF is flux normalized: $\int_0^\infty M(r)\,2\pi r\,\mathrm{d}r = 1$. The FWHM is $2\alpha\sqrt{2^{1/\beta} - 1}$.

WARNING: this normalization needs to be checked (both in 1D and 2D). See here.

The Fourier transform is (TBC):

$$\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty}(1 + x^2)^{-\beta} e^{i\omega x} dx = \frac{2^{1-\beta}|\omega|^{\beta-1/2} K_{\beta-1/2}(\omega)}{\Gamma(\beta)}$$

where K_n is the modified Bessel function of the second kind.

For a generic 2D-linear transformation C, the normalized 2D-profile can be written

$$M(x, y) = \frac{(\beta - 1)|C|^{1/2}}{\pi\alpha^2}\left(1 + \frac{(xy)C\binom{x}{y}}{\alpha^2}\right)^{-\beta}$$

E.g. elliptical radius $r^2 = x^2 + \epsilon y^2 + 2\xi\, xy$ corresponds to $C = \begin{pmatrix} 1 & \xi \\ \xi & \epsilon \end{pmatrix}$, and $|C| = \epsilon - \xi^2$.