HALO21 tutorial: Quasar absorption line fitting

Marie Wingyee Lau (UC Riverside)

- The Voigt profile
  Mark Krumholz's graduate course
  (https://sites.google.com/a/ucsc.edu/krumholz/teaching-and-courses/ast-230-s-12)

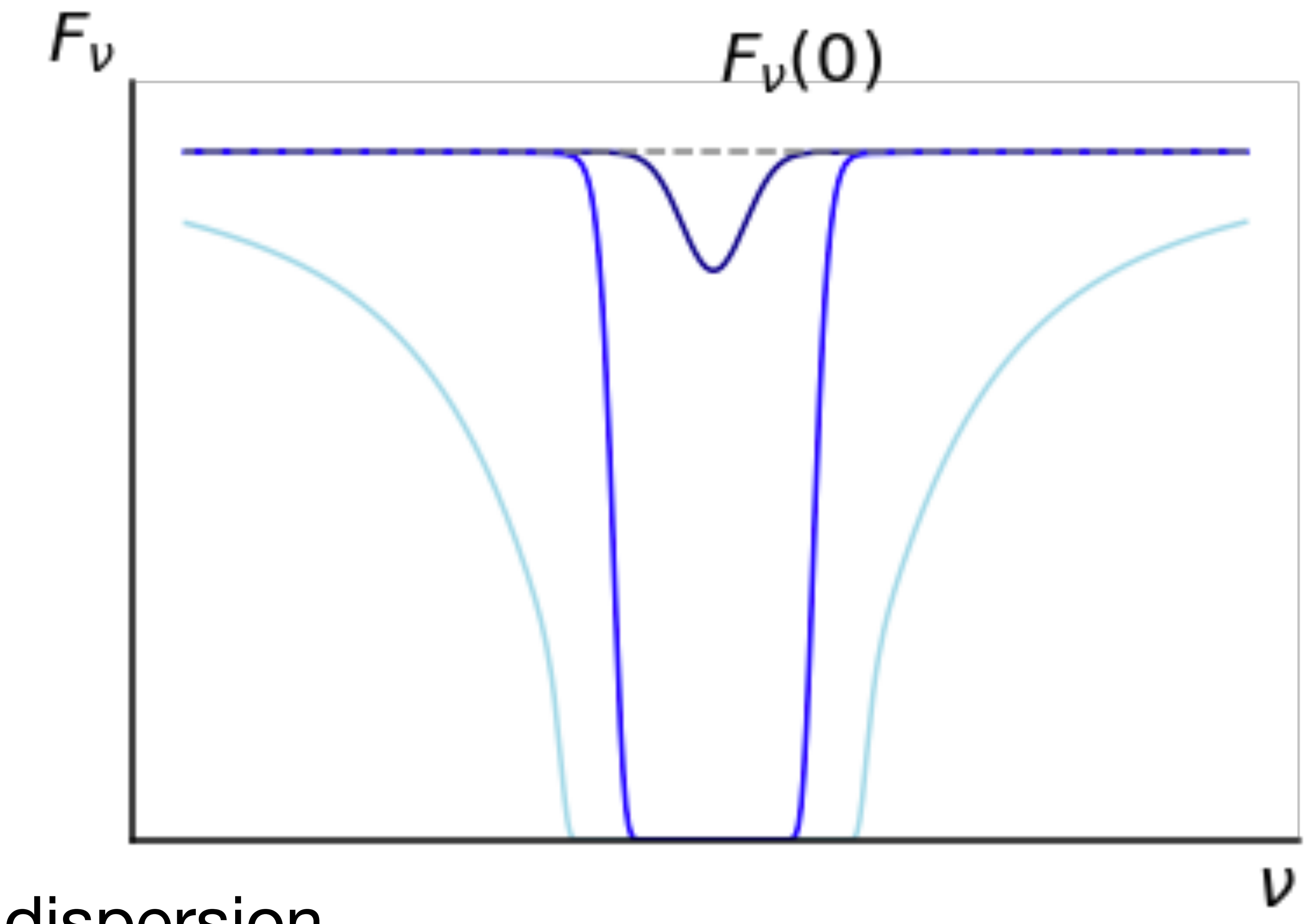
- Fit quasar continuum with linetools (Prochaska, Tejos+)
  https://github.com/linetools/linetools

- Fit absorption lines with ALIS (Ryan Cooke)
  https://github.com/rcooke-ast/ALIS

The Voigt profile

The flux we observe from a source through a foreground gas is $F_\nu = F_\nu(0)\exp(-\tau_\nu)$.

Interpolate in frequency from one side of a line to the other to estimate $F_\nu(0)$.



The optical depth is $\tau_\nu \approx (\pi e^2/m_e c)\, f_{\ell u}\, N_\ell\, \phi_\nu$,
where N_ℓ is the column density through the cloud,
and the line profile function ϕ_ν is a Voigt profile.

A Voigt profile has a Gaussian core and Lorentzian damping wings.

Focus on the core,
$\phi_\nu = 1/(\sqrt{\pi} b \nu_0/c)\exp(-(1-\nu/\nu_0)^2/(b/c)^2)$,
where $b = \sqrt{2}\sigma_v$ is the Doppler parameter, which is $\sqrt{2}$ times the velocity dispersion.

As the optical depth increases, the line center is saturated.
As the optical depth continues to increase, the Lorentzian damping wings dominate.

For HI Lyman α, damping dominates at optical depth ≈ 10^5, corresponding to column density ≈ few×10^{19} cm^{-2} .

Continuum fitting

Fit a quasar continuum by interpolation, and normalize.

The linetools algorithm:

The spectrum is split into an arbitrary number of wavelength intervals.

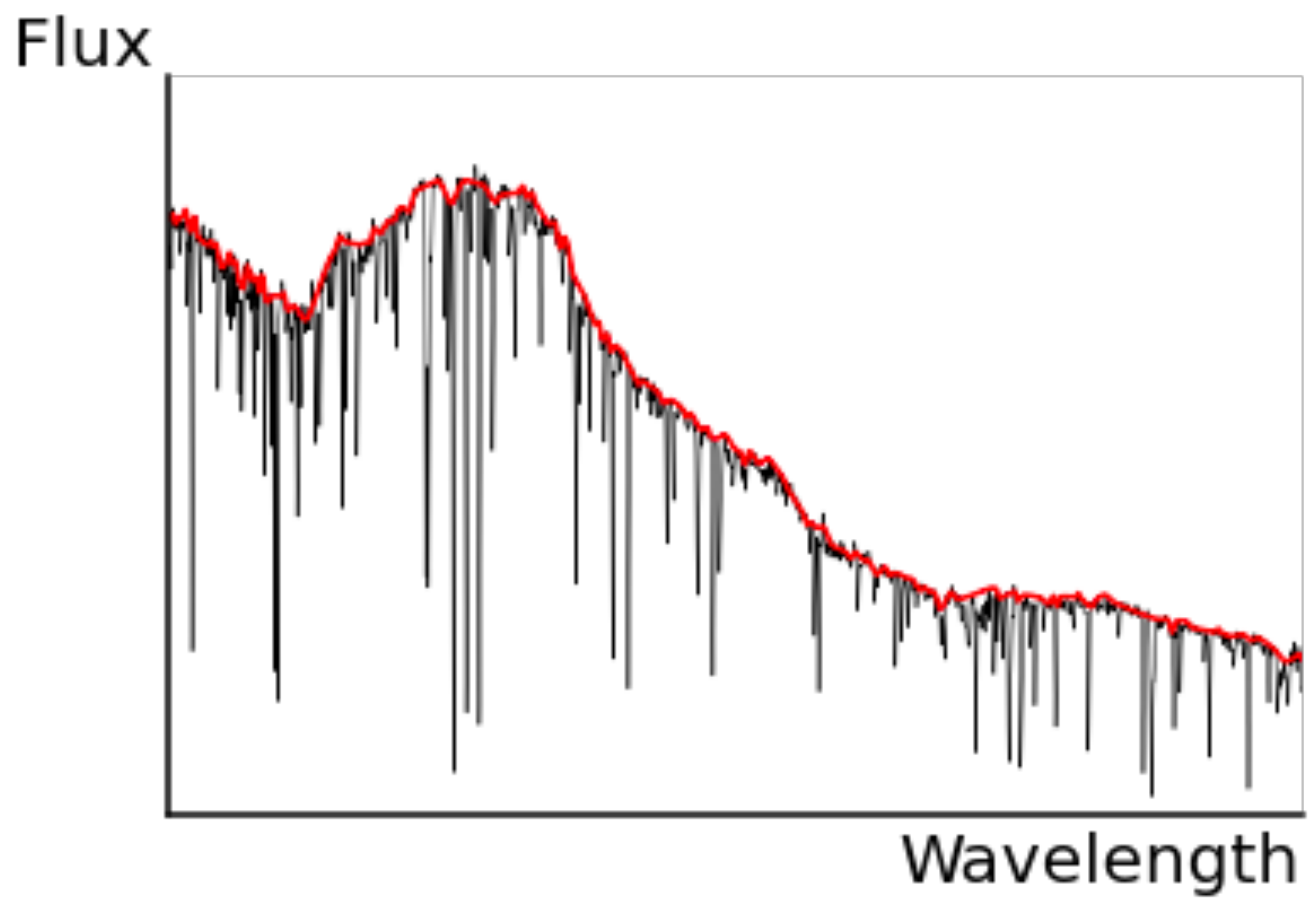
The flux median is calculated in each wavelength interval.

A cubic spline is fitted through points defined by the central wavelength and median flux value in each interval.

Pixels falling more than 1.5 standard deviations below the spline are masked.

Process is iterated over until no further pixels are removed as outliers.

In regions where the fit appears poor, user manually tweaks it.

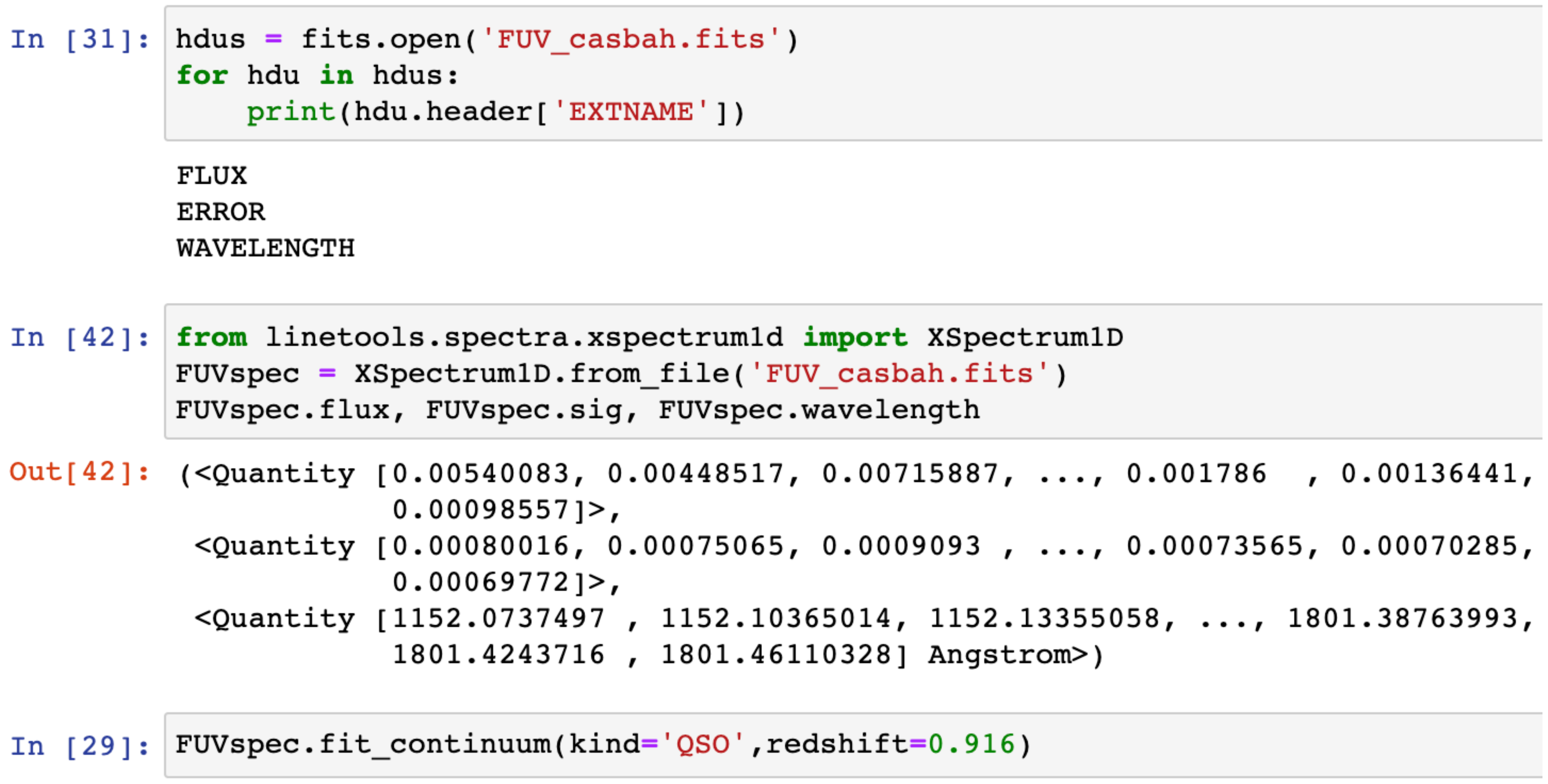


Longer version of the linetools algorithm:

“The unabsorbed quasar continuum is fitted interactively using the Python software package linetools. The spectrum is splitted into an arbitrary number of wavelength intervals that are chosen based on trial and error. They are about 12.5 rest frame Å wide shortward of the Lyα emission, smaller across emission lines, and larger in regions free of emission lines and longward of Lyα. The flux median is calculated in each wavelength interval. An Akima spline is fitted through a set of points defined by the central wavelength and median flux value in each interval. (Akima's interpolation method uses a continuously differentiable sub-spline built from piecewise cubic polynomials. The resultant curve passes through the given data points and will appear smooth and natural.) A linear spline is also fitted for the purpose of removing outliners from the fitting routine, and pixels falling more than 1.5 standard deviations below the linear spline are masked. This process is iterated over until no further pixels are removed as outliers. The resutant Akima spline from this automated process may still have regions where the continuum fit appears poor, due to strong absorption lines, strong emission lines, or artifacts in the data. In regions where the fit appears poor, new spline knots are manually added or existing knots' wavelength locations are manually moved. Knots are added or moved until the fit is satisfactory.”

Example using linetools

A COS FUV spectrum of FBQS0751+2919 from CASBAH (Tripp, Burchett, Prochaska+), at redshift 0.9158.
Load the data into an XSpectrum1D object.

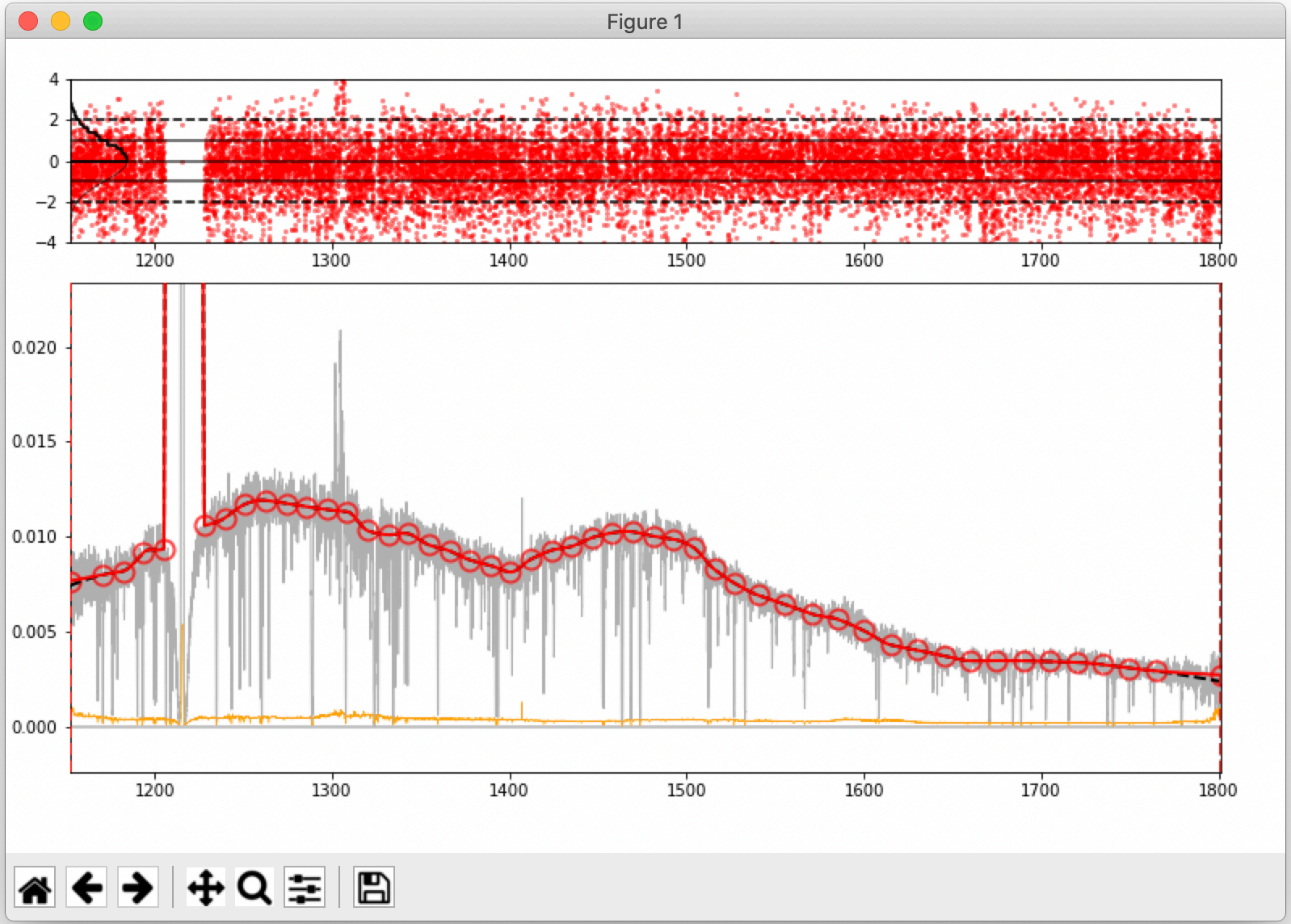


```
In [31]: hdus = fits.open('FUV_casbah.fits')
         for hdu in hdus:
             print(hdu.header['EXTNAME'])

         FLUX
         ERROR
         WAVELENGTH

In [42]: from linetools.spectra.xspectrum1d import XSpectrum1D
         FUVspec = XSpectrum1D.from_file('FUV_casbah.fits')
         FUVspec.flux, FUVspec.sig, FUVspec.wavelength

Out[42]: (<Quantity [0.00540083, 0.00448517, 0.00715887, ..., 0.001786  , 0.00136441,
                     0.00098557]>,
          <Quantity [0.00080016, 0.00075065, 0.0009093 , ..., 0.00073565, 0.00070285,
                     0.00069772]>,
          <Quantity [1152.0737497 , 1152.10365014, 1152.13355058, ..., 1801.38763993,
                     1801.4243716 , 1801.46110328] Angstrom>)

In [29]: FUVspec.fit_continuum(kind='QSO',redshift=0.916)
```

Example using linetools

Interactively zoom in/out,
set plot limits,
pan left/right,
add a new spline knot,
delete the nearest knot,
or move the nearest knot.

Type q to quit and save the continuum in the XSpectrum1D object.

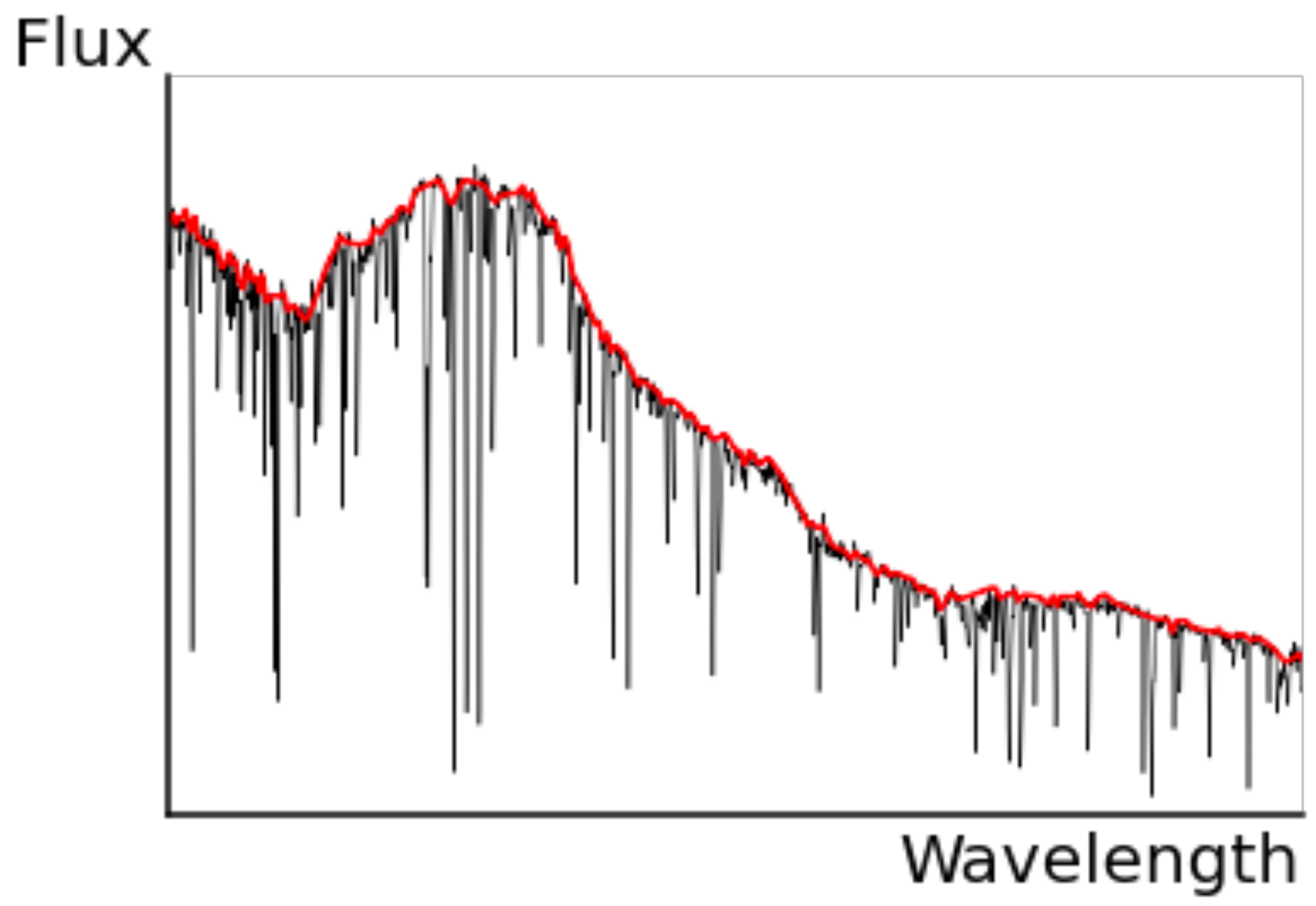


Example using linetools

Save to a file.

```
In [8]: FUVspec.write('FUV_casbah.fits')

        Wrote spectrum to FUV_casbah.fits
```

```
In [9]: hdus = fits.open('FUV_casbah.fits')
        for hdu in hdus:
            print(hdu.header['EXTNAME'])

        FLUX
        ERROR
        WAVELENGTH
        CONTINUUM
```

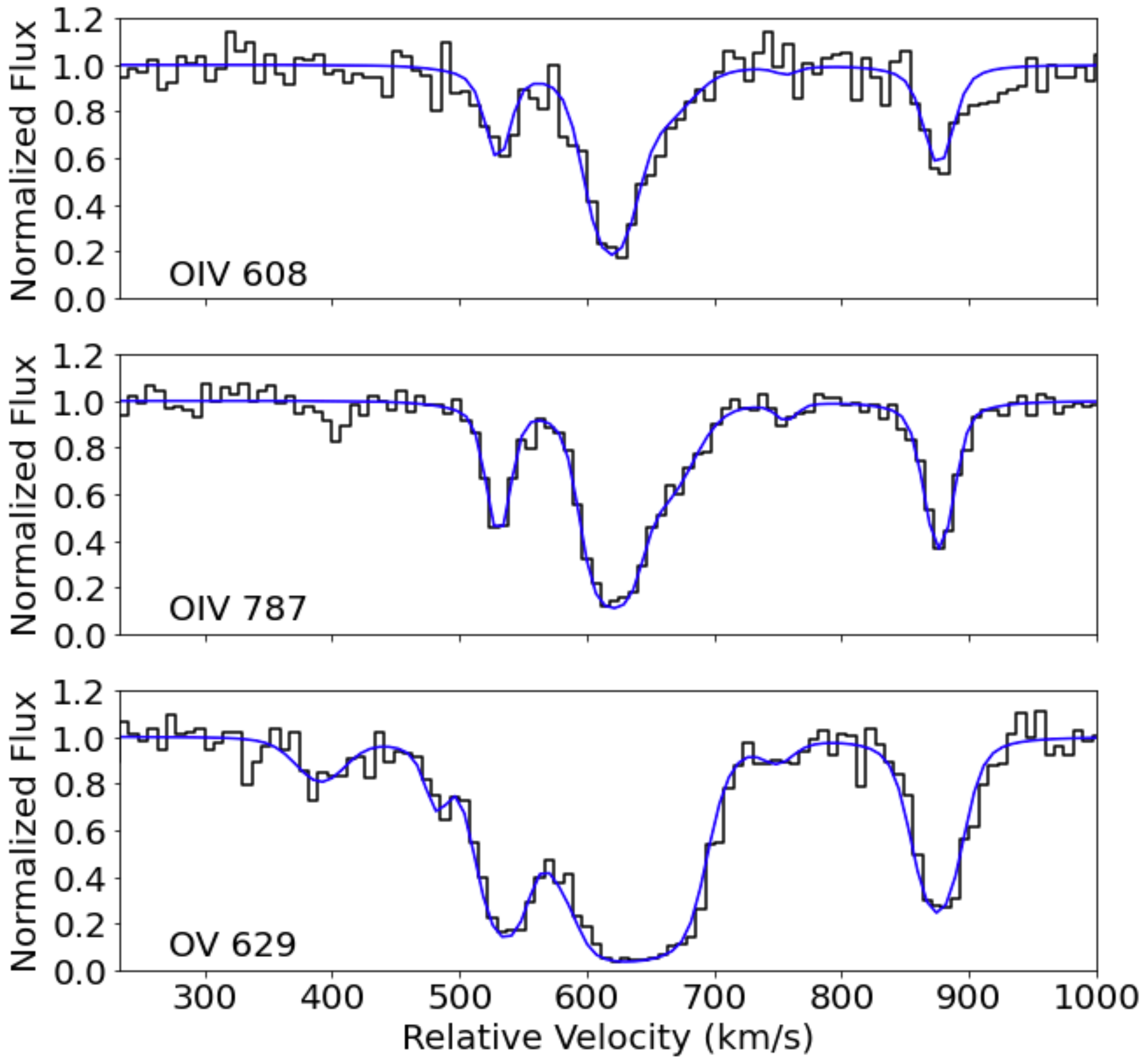


Tools for fitting absorption lines

From Slack #halo21-analysis-codes channel:
ALIS (Ryan Cooke),
Veeper (Burchett),
rbvfit (Bordoloi).
All written in Python.

Example here is a few associated absorption lines in FBQS0751+2019.
Fitting results by Todd Tripp using the Veeper are used as the truth set.

ALIS depends on linetools.

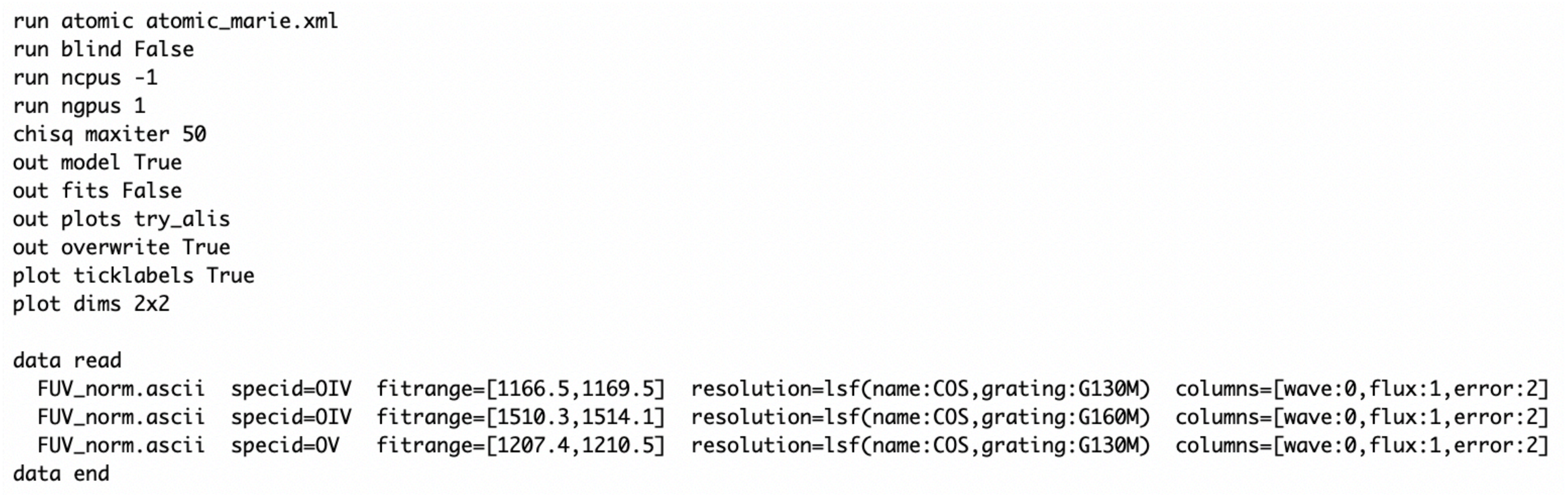


Example using ALIS: input

Input file is named XX.mod.

Give the paths to data, the wavelength ranges for fitting, and the spectral resolutions.

Data in ascii format and flux should be normalized.

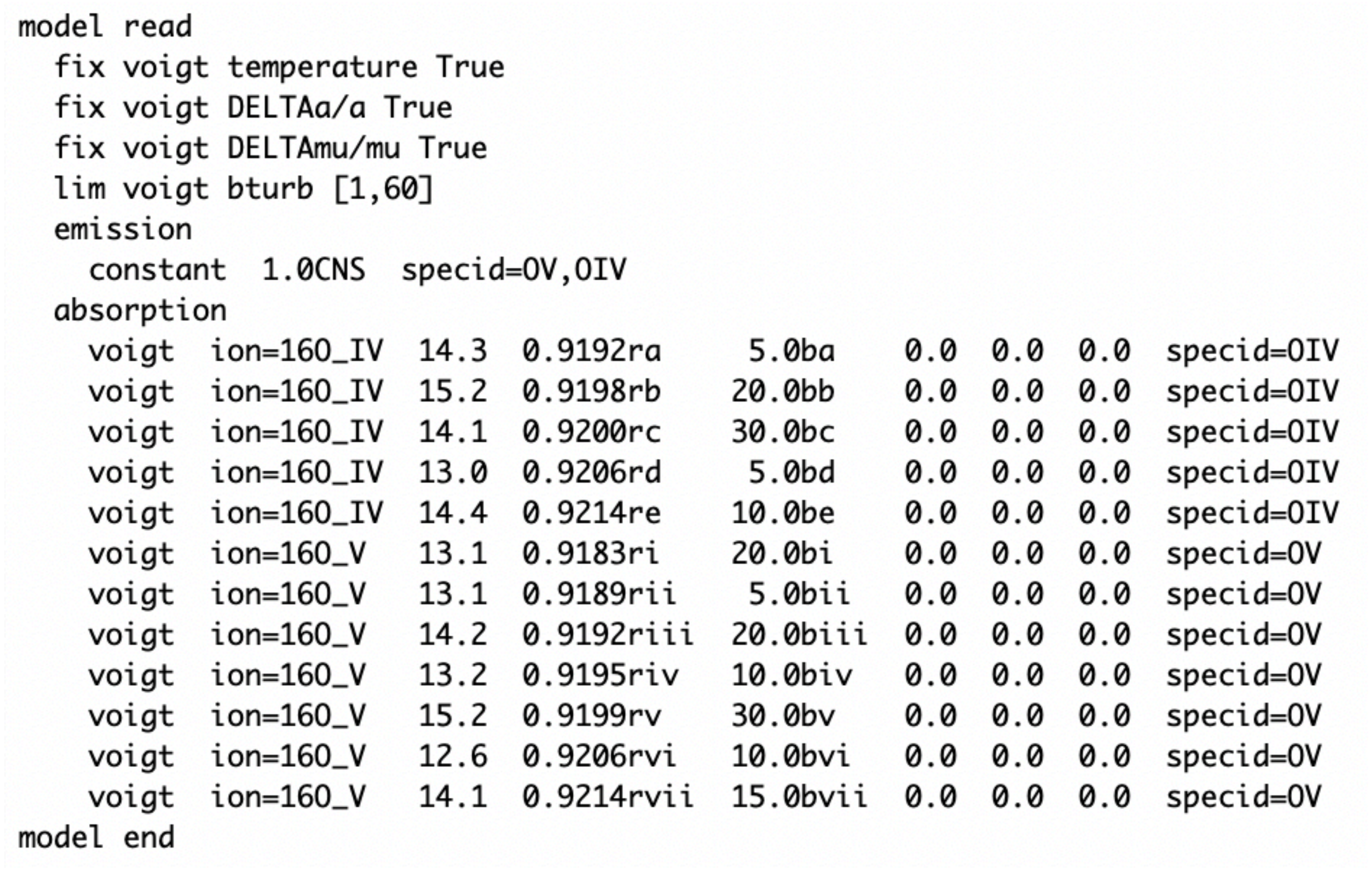


```
run atomic atomic_marie.xml
run blind False
run ncpus -1
run ngpus 1
chisq maxiter 50
out model True
out fits False
out plots try_alis
out overwrite True
plot ticklabels True
plot dims 2x2

data read
  FUV_norm.ascii  specid=OIV  fitrange=[1166.5,1169.5]  resolution=lsf(name:COS,grating:G130M)  columns=[wave:0,flux:1,error:2]
  FUV_norm.ascii  specid=OIV  fitrange=[1510.3,1514.1]  resolution=lsf(name:COS,grating:G160M)  columns=[wave:0,flux:1,error:2]
  FUV_norm.ascii  specid=OV   fitrange=[1207.4,1210.5]  resolution=lsf(name:COS,grating:G130M)  columns=[wave:0,flux:1,error:2]
data end
```

Example using ALIS: input

For each ionic absorption component, give a guess to column density, redshift, and Doppler *b*.

Use uppercase letters to specify fixed parameters, and lowercase letters to specify tied parameters.



```
model read
  fix voigt temperature True
  fix voigt DELTAa/a True
  fix voigt DELTAmu/mu True
  lim voigt bturb [1,60]
  emission
    constant  1.0CNS  specid=OV,OIV
  absorption
    voigt  ion=16O_IV  14.3  0.9192ra     5.0ba    0.0  0.0  0.0  specid=OIV
    voigt  ion=16O_IV  15.2  0.9198rb    20.0bb    0.0  0.0  0.0  specid=OIV
    voigt  ion=16O_IV  14.1  0.9200rc    30.0bc    0.0  0.0  0.0  specid=OIV
    voigt  ion=16O_IV  13.0  0.9206rd     5.0bd    0.0  0.0  0.0  specid=OIV
    voigt  ion=16O_IV  14.4  0.9214re    10.0be    0.0  0.0  0.0  specid=OIV
    voigt  ion=16O_V   13.1  0.9183ri    20.0bi    0.0  0.0  0.0  specid=OV
    voigt  ion=16O_V   13.1  0.9189rii    5.0bii   0.0  0.0  0.0  specid=OV
    voigt  ion=16O_V   14.2  0.9192riii  20.0biii  0.0  0.0  0.0  specid=OV
    voigt  ion=16O_V   13.2  0.9195riv   10.0biv   0.0  0.0  0.0  specid=OV
    voigt  ion=16O_V   15.2  0.9199rv    30.0bv    0.0  0.0  0.0  specid=OV
    voigt  ion=16O_V   12.6  0.9206rvi   10.0bvi   0.0  0.0  0.0  specid=OV
    voigt  ion=16O_V   14.1  0.9214rvii  15.0bvii  0.0  0.0  0.0  specid=OV
model end
```

In terminal prompt, run this: run_alis XX.mod

Example using ALIS: outputs

Fitting results are printed to a file named XX.mod.out.

The fitted model is printed in the same format as the initial guess model.

```
model read
  fix voigt temperature True
  fix voigt DELTAa/a True
  fix voigt DELTAmu/mu True
  lim voigt bturb [1,60]
 emission
  constant   1.00000000CNS       specid=OV,OIV
 absorption
  voigt   ion=16O_IV   14.2720348      0.919192125485ra      7.069894ba      0.0000000      0.0000000      0.0000000      specid=OIV
  voigt   ion=16O_IV   15.0745141      0.919770318579rb      16.505493bb      0.0000000      0.0000000      0.0000000      specid=OIV
  voigt   ion=16O_IV   13.9762821      0.920064267764rc      23.013518bc      0.0000000      0.0000000      0.0000000      specid=OIV
  voigt   ion=16O_IV   12.9291408      0.920651516716rd      4.414600bd      0.0000000      0.0000000      0.0000000      specid=OIV
  voigt   ion=16O_IV   14.3989017      0.921414156938re      8.547070be      0.0000000      0.0000000      0.0000000      specid=OIV
  voigt   ion=16O_V    13.1281528      0.918322947114ri      22.696606bi      0.0000000      0.0000000      0.0000000      specid=OV
  voigt   ion=16O_V    13.2146548      0.918903349489rii      3.440261bii      0.0000000      0.0000000      0.0000000      specid=OV
  voigt   ion=16O_V    14.2043471      0.919235813246riii      17.415544biii      0.0000000      0.0000000      0.0000000      specid=OV
  voigt   ion=16O_V    13.7214609      0.919484542242riv      1.578201biv      0.0000000      0.0000000      0.0000000      specid=OV
  voigt   ion=16O_V    15.1450418      0.919907603825rv      29.763870bv      0.0000000      0.0000000      0.0000000      specid=OV
  voigt   ion=16O_V    12.5945797      0.920617169235rvi      10.967802bvi      0.0000000      0.0000000      0.0000000      specid=OV
  voigt   ion=16O_V    14.0554719      0.921416653598rvii      16.392565bvii      0.0000000      0.0000000      0.0000000      specid=OV
model end
```

Errors of the fitted model are printed after the fitted model.

Also output figures of the fits and residuals.

Example using ALIS: verify the fitting results

	This trial	Tripp/Veeper
OIV component a	N=$10^{14.27}$ cm^{-2}, b=7 km s^{-1}, v=+530 km s^{-1}	N=$10^{14.33}$ cm^{-2}, b=6 km s^{-1}, v=+530 km s^{-1}
OIV component b	N=$10^{15.07}$ cm^{-2}, b=17 km s^{-1}, v=+621 km s^{-1}	N=$10^{15.15}$ cm^{-2}, b=15 km s^{-1}, v=+619 km s^{-1}
OIV component c	N=$10^{13.98}$ cm^{-2}, b=23 km s^{-1}, v=+666 km s^{-1}	N=$10^{14.09}$ cm^{-2}, b=27 km s^{-1}, v=+660 km s^{-1}
OIV component d	N=$10^{12.93}$ cm^{-2}, b=4 km s^{-1}, v=+758 km s^{-1}	N=$10^{12.96}$ cm^{-2}, b=4 km s^{-1}, v=+756 km s^{-1}
OIV component e	N=$10^{14.40}$ cm^{-2}, b=9 km s^{-1}, v=+877 km s^{-1}	N=$10^{14.36}$ cm^{-2}, b=9 km s^{-1}, v=+876 km s^{-1}
OV component i	N=$10^{13.13}$ cm^{-2}, b=23 km s^{-1}, v=+394 km s^{-1}	N=$10^{13.12}$ cm^{-2}, b=22 km s^{-1}, v=+391 km s^{-1}
OV component ii	N=$10^{13.21}$ cm^{-2}, b=3 km s^{-1}, v=+485 km s^{-1}	N=$10^{13.08}$ cm^{-2}, b=6 km s^{-1}, v=+482 km s^{-1}
OV component iii	N=$10^{14.20}$ cm^{-2}, b=17 km s^{-1}, v=+538 km s^{-1}	N=$10^{14.20}$ cm^{-2}, b=17 km s^{-1}, v=+535 km s^{-1}
OV component iv	N=$10^{13.72}$ cm^{-2}, b=2 km s^{-1}, v=+575 km s^{-1}	N=$10^{13.22}$ cm^{-2}, b=12 km s^{-1}, v=+577 km s^{-1}
OV component v	N=$10^{15.15}$ cm^{-2}, b=30 km s^{-1}, v=+642 km s^{-1}	N=$10^{15.16}$ cm^{-2}, b=29 km s^{-1}, v= +640 km s^{-1}
OV component vi	N=$10^{12.59}$ cm^{-2}, b=11 km s^{-1}, v=+753 km s^{-1}	N=$10^{12.61}$ cm^{-2}, b=12 km s^{-1}, v=+750 km s^{-1}
OV component vii	N=$10^{14.06}$ cm^{-2}, b=16 km s^{-1}, v=+878 km s^{-1}	N=$10^{14.05}$ cm^{-2}, b=16 km s^{-1}, v=+875 km s^{-1}

Tips on using ALIS

Augment the atomic data table with e.g. OIV608, OV629, OIV*609, OIV*790.
Edit atomic.xml.

Handling partial coverage of the emitting source is not straightforward (Hamann+2011).
To customize it, clone the ‘coverfactor’ branch on github and edit alfunc_voigt.py.