Spectral Analysis

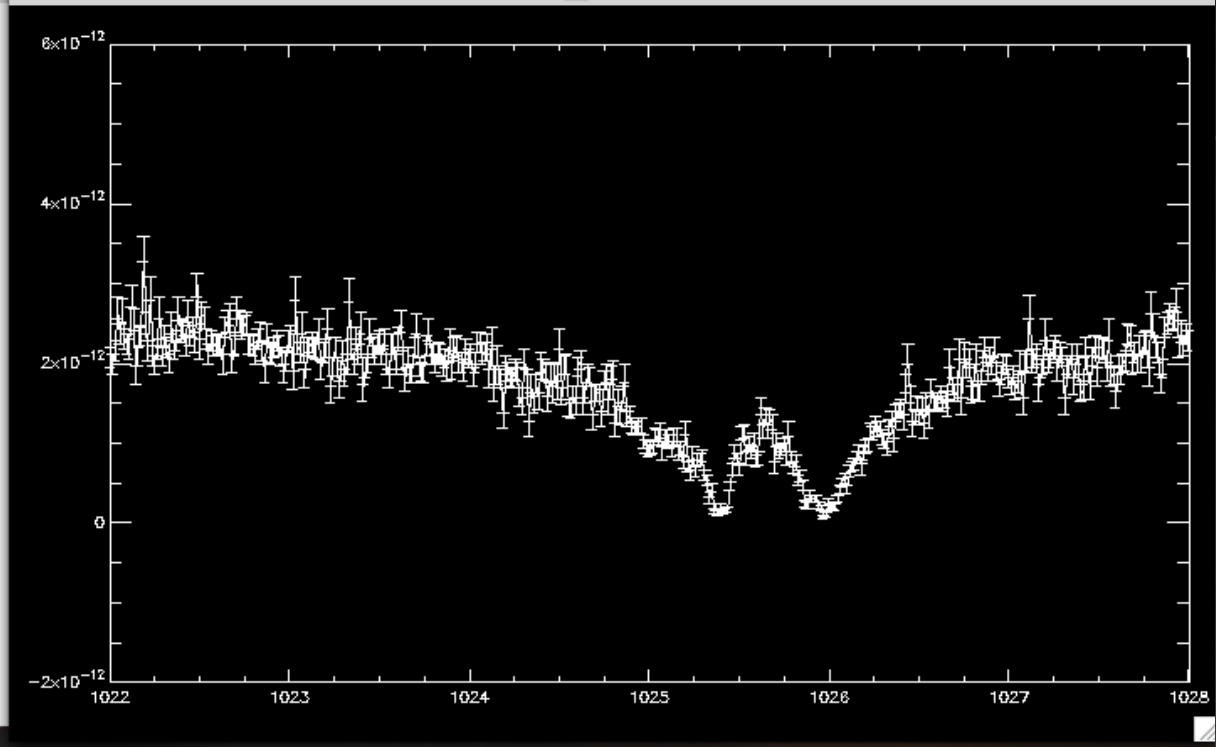
Jayant Murthy
The Indian Institute of Astrophysics
www.iiap.res.in
jmurthy@yahoo.com

Spectrum

2. vim

```
im=mrdfits("data_files/Q10901010031alif4ttagfcal.fit.gz",1,hdr)
hprint,hdr
plot,im.wave,im.flux
ans=""
read,ans
plot,im.wave,im.flux,xrange=[1022,1028]
min_wave=min(where(im.wave gt 1022))
max_wave=min(where(im.wave gt 1028))
wave = im(min_wave:max_wave).wave
flux = im(min_wave:max_wave).flux
err  = im(min_wave:max_wave).error
errplot,wave,flux-err/2,flux+err/2
plot,im.wave,im.flux,xrange=[1020,1023]
errplot,im.wave,im.flux-im.error,im.flux+im.error
s1=min(where(im.wave gt 1021))
s2=min(where(im.wave gt 1022.5))
mn  = mean(im(s1:s2).flux)
std = stdev(im(s1:s2).flux)
oplot,[1020,1023],[mn, mn]
oplot,[1020,1023],[mn - std, mn - std],col=255
oplot,[1020,1023],[mn + std, mn + std],col=255
~
~
~
"spec_fit.com" 21L, 694C written
```

GDL 0

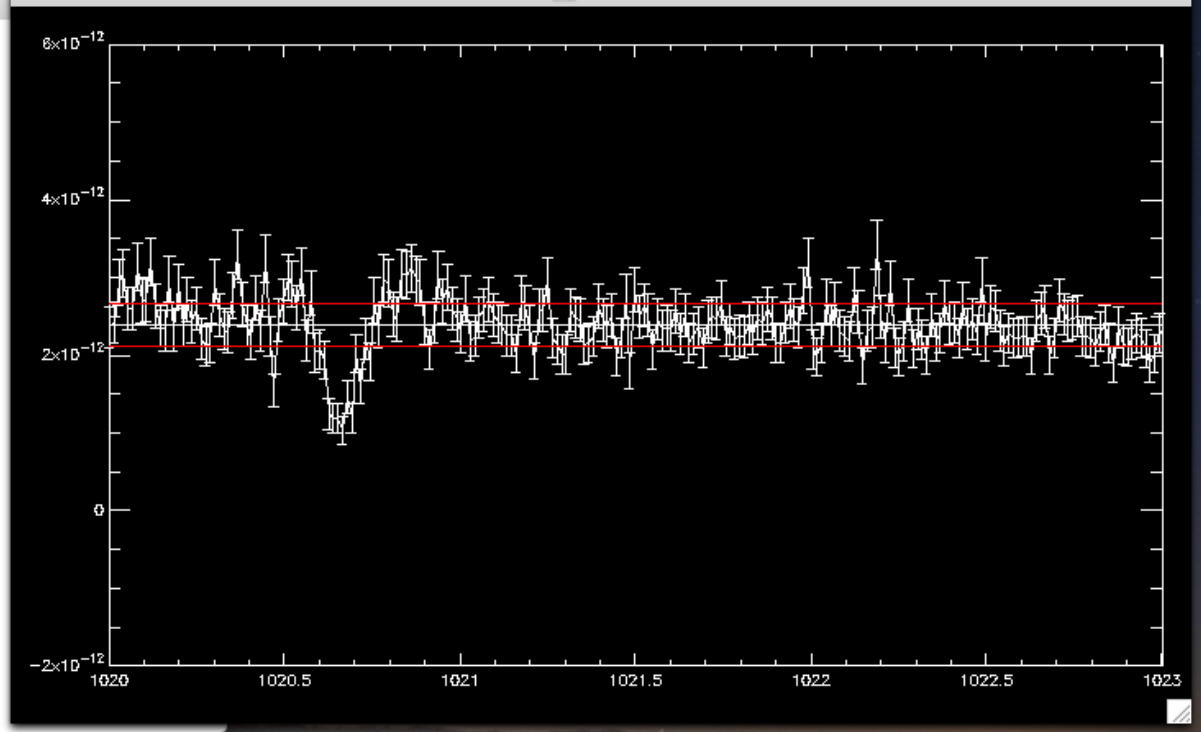


Error Estimation

2. vim

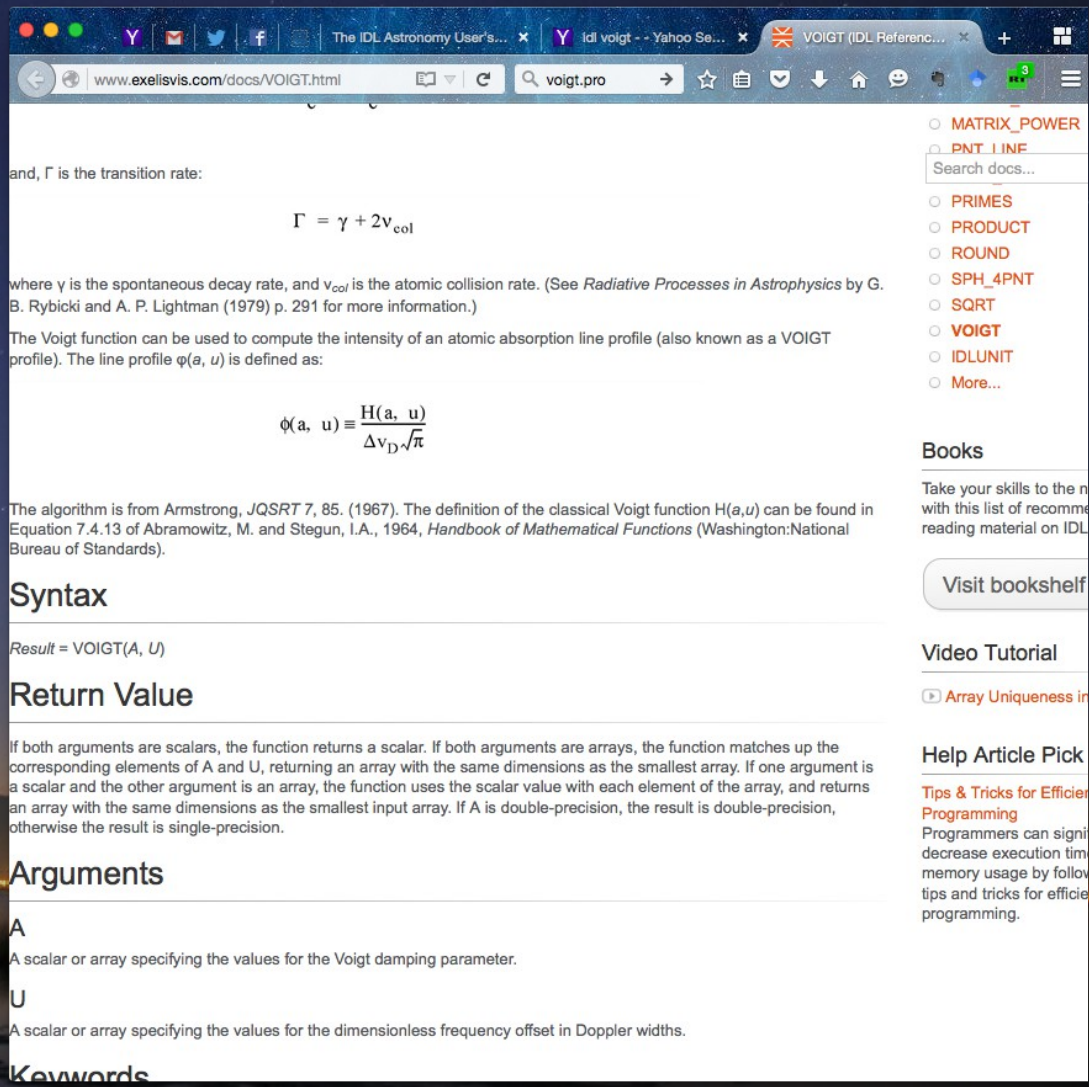
```
im=mrdfits("data_files/Q10901010031alif4ttagfcal.fit.gz",1,hdr)
hprint,hdr
plot,im.wave,im.flux
ans=""
read,ans
plot,im.wave,im.flux,xrange=[1022,1028]
min_wave=min(where(im.wave gt 1022))
max_wave=min(where(im.wave gt 1028))
wave = im(min_wave:max_wave).wave
flux = im(min_wave:max_wave).flux
err  = im(min_wave:max_wave).error
errplot,wave,flux-err/2,flux+err/2
plot,im.wave,im.flux,xrange=[1020,1023]
errplot,im.wave,im.flux-im.error,im.flux+im.error
s1=min(where(im.wave gt 1021))
s2=min(where(im.wave gt 1022.5))
mn  = mean(im(s1:s2).flux)
std = stdev(im(s1:s2).flux)
oplot,[1020,1023],[mn, mn]
oplot,[1020,1023],[mn - std, mn - std],col=255
oplot,[1020,1023],[mn + std, mn + std],col=255
~
~
~
"spec_fit.com" 21L, 694C written
```

GDL 0



Fitting Lines

- Absorption/emission lines are convolution of Lorentz + Gaussian
  - Voigt profile.
  - Function of a,u.



and, Γ is the transition rate:

$$\Gamma = \gamma + 2\nu_{col}$$

where γ is the spontaneous decay rate, and ν_{col} is the atomic collision rate. (See *Radiative Processes in Astrophysics* by G. B. Rybicki and A. P. Lightman (1979) p. 291 for more information.)

The Voigt function can be used to compute the intensity of an atomic absorption line profile (also known as a VOIGT profile). The line profile φ(*a*, *u*) is defined as:

$$\phi(a, u) \equiv \frac{H(a, u)}{\Delta\nu_D\sqrt{\pi}}$$

The algorithm is from Armstrong, *JQSRT* 7, 85. (1967). The definition of the classical Voigt function H(*a*,*u*) can be found in Equation 7.4.13 of Abramowitz, M. and Stegun, I.A., 1964, *Handbook of Mathematical Functions* (Washington:National Bureau of Standards).

Syntax

Result = VOIGT(*A*, *U*)

Return Value

If both arguments are scalars, the function returns a scalar. If both arguments are arrays, the function matches up the corresponding elements of A and U, returning an array with the same dimensions as the smallest array. If one argument is a scalar and the other argument is an array, the function uses the scalar value with each element of the array, and returns an array with the same dimensions as the smallest input array. If A is double-precision, the result is double-precision, otherwise the result is single-precision.

Arguments

A

A scalar or array specifying the values for the Voigt damping parameter.

U

A scalar or array specifying the values for the dimensionless frequency offset in Doppler widths.

Keywords

Fitting Methods

- Use chi square minimization.
  - Model is a function of N parameters.
  - Find combination of parameters which minimize chi square.
- Brute force.
- Find chi square for every combination of parameters and pick minimum.
- Time consuming.

Fitting Methods

- Use chi square minimization.
  - Model is a function of N parameters.
  - Find combination of parameters which minimize chi square.
- Levenberg-Marquardt.
- Find rate of change of chi square.
- Use slope to find minimum.
  - Can get stuck and find local minimum.

mpfit: http://cow.physics.wisc.edu/~craigm/idl/idl.html

MPFIT

- The user must supply the following items:
  - An array of independent variable values ("X").
  - An array of "measured" *dependent* variable values ("Y").
  - An array of "measured" 1-sigma uncertainty values ("ERR").
  - The name of an IDL function which computes Y given X ("MYFUNCT").
  - Starting guesses for all of the parameters ("START_PARAMS").

- function mpfitfun, fcn, x, y, err, p, WEIGHTS=wts, FUNCTARGS=fa, $
  - BESTNORM=bestnorm, nfev=nfev, STATUS=status, $
  - best_resid=best_resid, pfree_index=ifree, $
  - calc_fjac=calc_fjac, best_fjac=best_fjac, $
  - parinfo=parinfo, query=query, CASH=cash, $
  - covar=covar, perror=perror, yfit=yfit, $
  - niter=niter, nfree=nfree, npegged=npegged, dof=dof, $
  - quiet=quiet, ERRMSG=errmsg, NAN=NAN, _EXTRA=extra

Final Fit

1. bash

```
$cat linfit/voigtfit.pro
FUNCTION voigtfit,wave,par,gamma
gamma=1e8
;wave: wavelength in Angstroms
;a = GAMMA/(4*PI*DELTA_VD)
;DELTA_VD = V0/C * B
;u = (NU - NU0)/DELTA_VD
;NU = C/LAMBDA
;phi(a, u) = H(a, u)/DELTA_VD/SQRT(PI)
;
;par(0) = LAMBDA0 in A
;par(1) = B in km/s
;par(2) = N in cm-2
        c_km = 3.e5; Wavelength of light in km/s
        c_ang = 3.e18; speed of light in A/s
        nu      = c_ang/wave
        nu0 = c_ang/par(0)
        delta_vd = nu0*par(1)/c_km
        a = gamma/(4*!pi*delta_vd)
        u = (nu - nu0)/delta_vd
        phi = voigt(a, u)/delta_vd/sqrt(!pi)

;2nd component
        c_km = 3.e5; Wavelength of light in km/s
        c_ang = 3.e18; speed of light in A/s
        nu      = c_ang/wave
        nu0 = c_ang/par(3)
        delta_vd = nu0*par(4)/c_km
        a = gamma/(4*!pi*delta_vd)
        u = (nu - nu0)/delta_vd
        phi1 = voigt(a, u)/delta_vd/sqrt(!pi)

        prof = exp(-(par(2)*phi + par(5)*phi1));Output
        return,prof
END


$
```

1. bash

```
        u = (nu - nu0)/delta_vd
        phi1 = voigt(a, u)/delta_vd/sqrt(!pi)

        prof = exp(-(par(2)*phi + par(5)*phi1));Output
        return,prof
END


$cat linfit/analysis.com
.run linfit/mpfitfun
.run linfit/voigtfit

im=mrdfits("data_files/Q10901010031alif4ttagfcal.fit.gz",1,hdr)
min_wave=min(where(im.wave gt 1022))
max_wave=min(where(im.wave gt 1028))
wave = im(min_wave:max_wave).wave
flux = im(min_wave:max_wave).flux
err  = im(min_wave:max_wave).error

par=replicate({value:0.0,fixed:0,limited:[0,0],limits:[0.D,0.D]},6)
par[0].value = 1025.4
par[3].value = 1026.0
par[1].value = 100.
par[4].value = 100.
par[2].value = 3.e12
par[5].value = 3.e12
gamma = 1e8

mf = mean(flux[0:20])
flux = flux/mf
err  = err/mf
result = mpfitfun('voigtfit',wave, flux, err,parinfo=par)

model = voigtfit(wave,result)
plot,wave,flux & oplot,wave,model,col=255
stop


$
```

Final Fit