SPIRE Line Fitting using a script

Ed Polehampton

On behalf of the SPIRE ICC

SPIRE Spectral Line Fitting..

- See SPIRE Data Reduction Guide

  6.2.6. Spectral Analysis

- Advantage of SPIRE – fitting many lines together
- Considerations in SPIRE line fitting:
  - **Local Standard of Rest** correction is not applied by the pipeline

HIPE product metadata name	FITS header variable	Example value	Definition
velocityDefinition	VELDEF	'RADI-LSR'	The velocity definition and frame
radialVelocity	VFRAME	21.08429538498	Spacecraft velocity in units of [km s^{-1}] along the line of sight of the telescope wrt the local standard of rest

SPIRE Spectral Line Fitting..

- **Unapodized** data contains the most spectral information – should be fitted with a Sinc function:

$$f(x:p) = p_0 \sin([x - p_1]/p_2) / ([x - p_1]/p_2)$$

- **Apodized** data can be fitted with a Gaussian

(but can overestimate true flux by up to 5%)

- Conversion from **wavenumber** to **frequency** can be done inside Hipe

*using a rounded value of c **does** make a difference!*

- In general the **line width** should be **fixed** (except for galaxies with partially resolved lines)

Multiple Line Fitting Script in Hipe

SpireSpectrumFitterDemo.py

```
# SPIRE Line Fitting script example based on using the SpectrumExplorer
# SpectrumFitterGUI
#
# This script was created by running a single line fit by hand in the GUI,
# then using the option in the GUI to save the procedure as a script. It
# was then extended to include other lines read from a catalogue.
#
# OT1 DP Workshop at ESAC 16-18 March 2011
#

# Read in the point source spectrum product
# Either from a FITS file (the end result of the User Reprocessing Script)
# pointSourceSda = fitsReader('1342189124_spectrum_point_unapod.fits')
# Or read from the observation context pool
oba = getObservation(1342189124, poolName="spireSpec1342189124_NGC7027")
# Using the syntax from the Product Viewer:
pointSourceSda = oba.refs["level2"].product.refs["HR_unapodised_spectrum"].product
# (or the simplified syntax!)
# pointSourceSda = oba.level2.getProduct("HR_unapodised_spectrum")

# Define some constants
from herschel.share.unit import Constant
k = Constant.K_BOLTZMANN.value
c = Constant.SPEED_OF_LIGHT.value
sourceVelocity = 20.0*1e3

# Get the satellite radial velocity from the metadata
radialVelocity = pointSourceSda.meta['radialVelocity'].value

# Open a file for the line fit results
fitResults = open('C:\\Users\\stp87\\Desktop\\DPWorkshop\\fit_results.dat','w')

# Read the line catalogue
name     = StringId([])
lineFreqs = DoubleId([])
file  = open('C:\\Users\\stp87\\Desktop\\DPWorkshop\\spirelines.dat','r')
for line in file.readlines():
    if line[0] != "#":
        name.append(line.split()[0])
        lineFreqs.append(float(line.split()[1]))
file.close()

pl=PlotXY()
# Loop over the two centre detectors
for detector in ["SLWC3", "SSWD4"]:
    spireSpectrumId = pointSourceSda['0000'][detector].copy()
    # convert the wavescale to GHz
    gHzSpectrum = convertWavescale(ds=spireSpectrumId, to="GHz")
    # correct to LSR using satellite radial velocity
    gHzSpectrum['frequency'].data = gHzSpectrum['frequency'].data*(1.-radialVelocity/c)
    # Plot the spectrum
    pl.addLayer(LayerXY(gHzSpectrum['frequency'].data, gHzSpectrum['flux'].data, color=java.awt.Color.BLUE))

    # Initialise the Fitter
    sf = SpectrumFitter(gHzSpectrum, 0, 1, False)
    # Specify fit engine (1 - LevenbergMarquardt,2 - Amoeba, 3 - Linear, 4 - MP, 5 - Conjugated Gradient).
    sf.useFitter(1)
    # Add the models. First, polynomial for continuum (=initial guess)
    Mol = sf.addModel('poly', [3.0], [514.0577074148017,-1.3697323948473303,0.0012781730567420267,-2.8923270579794556E-7])

    # Correct the catalogue frequencies for the source velocity (for initial guess)
    lineFreqsCorr = lineFreqs*(1.-sourceVelocity/c)
    # Select only lines within the band and add a model for each
    sel = lineFreqsCorr.where(lineFreqsCorr.gt(MIN(gHzSpectrum['frequency'].data)).and(lineFreqsCorr.lt(MAX(gHzSpectrum['frequency'].data))))
    for idx in sel.toIntId():
       line = lineFreqsCorr[idx]
       exec("M%i = sf.addModel('sinc', [10.0, line, (100*c*0.04)/(Math.PI*1e9)])"%idx)
       # fix the width of the line to the resolution
       exec("M%i.setFixed(IntId([2]))"%idx)
       # Plot the rest frequencies (not shifted by source velocity)
       pl.addLayer(LayerXY(DoubleId([lineFreqs[idx],lineFreqs[idx]]),DoubleId([0,200])))
       pl.addAnnotation(Annotation(lineFreqs[idx], -50, name[idx], angle=90))

    # Do the fit, calculate the residual
    sf.doGlobalFit()
    sf.residual()
    tm = sf.getTotalModel()

    # Plot the total model, and residual
    pl.addLayer(LayerXY(gHzSpectrum['frequency'].data, tm, color=java.awt.Color.RED))
    pl.addLayer(LayerXY(gHzSpectrum['frequency'].data, gHzSpectrum['flux'].data-tm, color=java.awt.Color.GREEN))

    # calculate line fluxes from the fit parameters
    for idx in sel.toIntId():
       # line area = Jy.Hz
       exec("area =  1e-26*M%i.getFittedParameters()[0]*Math.PI*1e9*M%i.getFittedParameters()[2]"%(idx,idx))
       exec("print name[idx], M%i.getFittedParameters(), area"%idx)
       exec("fitResults.write('%%s %%s %%s %%s \\n'%%(name[idx], lineFreqs[idx], M%i.getFittedParameters()[1], area))"%idx)

pl.xtitle="Frequency (GHz)"
pl.ytitle="Flux Density (Jy)"

fitResults.close()
```

Read in the level-2 point source spectrum

Constants are already defined inside the system
(but require importing an extra package)

Read in a file containing line frequencies
(line by line – could also have used asciiTableReader task!)

Loop over the two central detectors

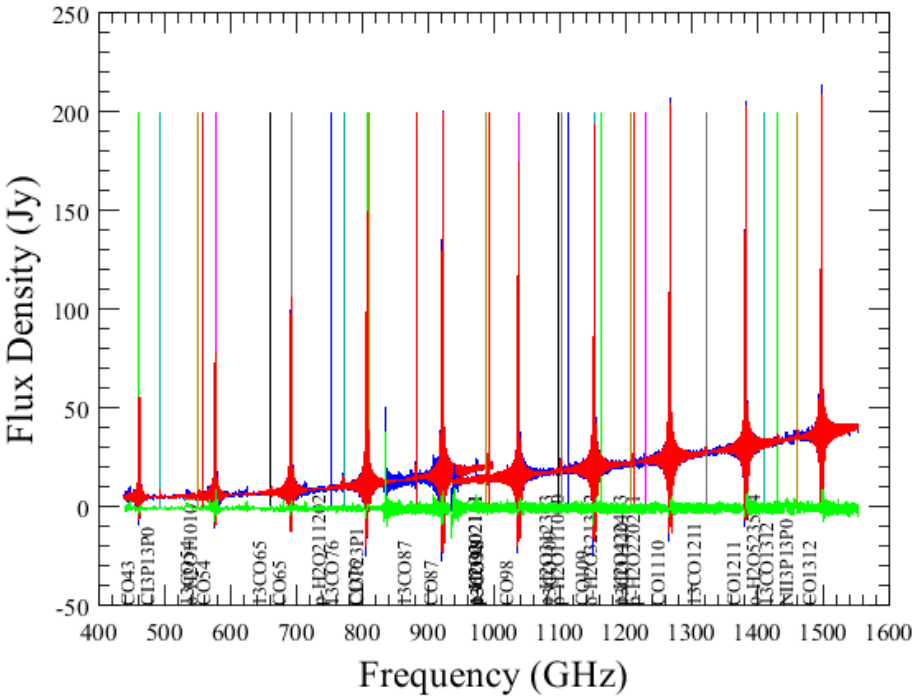
- Convert **wavenumber** to **frequency**
- Convert to **LSR** using satellite velocity
- Set up the "**models**" to be fitted (from the line catalogue)
- Width is **fixed** to instrument resolution
- Position & height are **free** (with catalogue position as initial guess)

Calculate the integrated line flux from the fitted parameters:

$$I = \int f(x{:}p) = 10^{-26}\, p_0\, \pi\, p_2$$

Result of running the script for NGC7027:

Plot of data, fit & residual with lines marked



Text file containing line name, rest freq, fitted freq & integrated flux

emacs@SSTDLETP

File Edit Options Buffers Tools Help

```
CI3P13P0 492.161 491.99491209897724 4.3257603938655746E-17
CO43 461.041 460.99752299769864 6.1023948899658O1E-16
13CO54 550.926 550.7495240681887 4.3677664459551 98E-17
o-H2O110101 556.936 556.868648654039 1.7443826809886026E-17
CO54 576.268 576.2846275266446 8.799981932854693E-16
13CO65 661.067 661.0954174663423 4.7773813612798774E-17
CO65 691.473 691.4112915588944 1.1851471814264613E-15
p-H2O211202 752.033 751.1564599745309 8.382138537692064E-18
13CO76 771.184 771.0642449917533 7.2504393154104l7E-17
CO76 806.652 806.6044751438731 1.6917265917148065E-15
CI3P23P1 809.342 809.1308114264003 1.3171178967766926E-16
13CO87 881.273 881.2387469061148 8.4104695458921E-17
CO87 921.8 921.7249328740388 2.225383818517116E-15
p-H2O202111 987.927 987.0643903719049 1.2370711309005 98E-17
13CO98 991.329 991.314366661258 9.0509879316252 33E-17
p-H2O202111 987.927 987.1175181212849 2.2305029737315664E-17
13CO98 991.329 991.2440835785l3 8.53440988572154E-17
CO98 1036.912 1036.8415895973876 1.9593248269541595E-15
o-H2O312303 1097.365 1096.7641969341844 1.037783087898636E-17
13CO109 1101.35 1101.3113971979838 6.67174814523376E-17
p-H2O111000 1113.343 1113.294334000571 3.1432235226057 92E-17
CO109 1151.985 1151.8865645897765 2.107050927304347E-15
o-H2O321312 1162.912 1163.5857515808027 8.31712116984742E-18
p-H2O422413 1207.639 1210.62653656742 28 -3.9346767266384375E-17
13CO1110 1211.33 1211.0867392501557 8.44769834180141 8E-17
p-H2O220211 1228.789 1228.917841304385 3.3711087368569504E-17
CO1110 1267.014 1266.9249558988975 2.170848988287318E-15
13CO1211 1321.265 1321.2899330252326 3.463134661384508E-17
CO1211 1381.995 1381.8995504349425 2.1245987976211 45E-15
o-H2O523514 1410.618 1409.7313033253376 -4.522704928162886E-18
13CO1312 1431.153 1430.9431185667725 3.460430004269E-17
NII3P13P0 1461.13 1461.0534004240335 5.8665695546760 76E-17
CO1312 1496.923 1496.8474775132534 2.0697764388681177E-15
```

--\-- fit_results.dat (Fundamental)--L1--All

Other SPIRE line fitting methods

- Line fitting code being developed in **IDL** at **University of Lethbridge** (for SPIRE and JCMT FTS2)

- Iterative line fitter in IDL developed at **Cardiff University**