# How to read and display the 1D spectral data format

Laura Mascetti - 2023-09-28 - Comments (0) - Phase 3 FAQs

The one-dimensional (1D) spectrum format of the ESO Science Data Products Standard (SDP) is a binary table that contains information for an individual object. It is made of a primary header and one single extension. The primary header contains the main SDP keywords whose values are used for the identification/selection of the product in the ESO SAF query forms (see for example: http://archive.eso.org/scienceportal or http://archive.eso.org/wdb/wdb/adp/phase3_spectral/form). The extension contains a FITS binary table, with its own header unit and with the data arrays (wavelength, flux, error, etc.) stored as 1D vectors in single cells (NAXIS2 must be set to 1). Each FITS file contains only one science spectrum, plus associated information; for instance: sky background subtracted spectrum, error spectrum, best fitted model for the continuum, etc. The spectrum binary table format complies with the basic requirements of the Virtual Observatory (VO Spectrum Data Model v1.1) standard.

**How to display the ESO SAF spectra stored in binary table format**
According to our experience the following tools support the science 1d spectra downloaded from the ESO science archive: Python, IRAF onedspec (splot) and rv packages, fv, vospec, splatvo. A list of instructions for displaying the spectra with the mentioned tools is provided here below.

## Python

For python users, the lines below illustrate a simple script to read and display a binary table format file.

A more detailed script, able to recognise the main scientific arrays in a 1d spectrum (using UTYPEs), and also able to deal with spectra containing multiple flux and flux error arrays, is available: 1dspectrum.py (version: 2017-08-07).

```
#!/usr/bin/python
from astropy.io import fits

hdulist = fits.open( "your_1d_spectrum_here.fits" )

# print column information
hdulist[1].columns

# get to the data part (in extension 1)
scidata = hdulist[1].data

wave = scidata[0][0]
arr1 = scidata[0][1]
arr2 = scidata[0][2]
# etc.
# where arr1 will contain the data corresponding to the column named: hdulist[1].columns[1]
# where arr2 will contain the data corresponding to the column named: hdulist[1].columns[2]
# etc.

# To plot using maptplotlib:
import matplotlib.pyplot as plt
plt.plot(wave, arr1)

plt.show()
```

**IDL**

For IDL/astrolib users, the lines below illustrate the script to read and display a binary table format file.

```
IDL> a=mrdfits('ADP.2013-09-24T15:43:19.447.fits',1)

IDL> help,a

** Structure <300fa08>, 6 tags, length=1983944, data length=1983940, refs=1:

   WAVE            DOUBLE    Array[70855

   FLUX_REDUCED    FLOAT     Array[70855]

   ERR_REDUCED     FLOAT     Array[70855]

   BGFLUX_REDUCED  FLOAT     Array[70855]

   FLUX            FLOAT     Array[70855]

   ERR             FLOAT     Array[70855]

IDL> plot, a.WAVE, a.FLUX, xrange=[3780, 3820]
```

**fv**

On a Linux system with Scisoft installed, fv can be launched using the unix command: fv. To download a stand alone version of fv, please visit: http://heasarc.gsfc.nasa.gov/lheasoft/ftools/fv/

Once fv is opened, make sure that POW (and not ds9) is selected as Display Device. To do this click Display Device on the fv main panel and select POW.

To load a 1D spectrum in the fv panel:

- choose “Open File” and navigate to the folder where the 1D spectra are stored,
- select the spectrum of interest by clicking on it,
- click "Open".

The “Summary” panel opens and shows the structure of the selected FITS file with a record for the primary header unit and a record for the FITS extension that hosts the binary table with the spectral arrays.

To visualize the spectrum:

- click "Plot" on the second record (the "Select Plot Columns" panel shows up) ,
- click WAVE (on the left hand side) and then the "X" button (in the center) to assign the wavelength array to the X axis,
- click FLUX (on the left hand side) and then the "Y" button (in the center) to assign the flux calibrated array to the Y axis. If the flux calibrated array is not available, then click on the FLUX_REDUCED array. As result of these actions, the words WAVE and FLUX should now also appear close (on the right) to the X and Y buttons. Click “Plot” and the selected spectrum is displayed in the POW window.

**SPLAT VO**

On a Linux system with Scisoft installed, splat can be launched using the unix command: splat. On all operating systems you can start up (webstart) or install SPLAT VO; please visit: http://star-www.dur.ac.uk/~pdraper/splat/splat-vo/splat-vo.html Once started, the user can open a 1D spectrum by choosing "File -> Open" option, or simply pressing CTRL-O, or by clicking on the opened folder (first icon on the upper left corner), and then navigate to the folder containing the spectrum, and select it by clicking the "Open" button. Two new windows appear:

- The "Starlink SPLAT-VO: <plot0>" window displays the reduced spectrum by default (not the flux calibrated one);
- the "Starlink SPLAT-VO: A Spectral Analysis Tool" window, where the user can change what is displayed by choosing to plot the calibrated FLUX array instead of the FLUX_REDUCED, in the “Data” pull-down menu.

Update (25/Nov/2013): Please note that the Data pull-down menu problem previously reported has been fixed. Please download the latest version of SPLAT VO from the GAVO SPLAT-VO page. GAVO stands for German Astrophysical Virtual Observatory.

**If you are still using IRAF**

**IRAF/SPTABLE external package, including xonedspec and xrv**

The SPTABLE IRAF external package is able to read, display, and analyse (via the onedspec and rv packages) the ESO one-dimensional science spectra. Please find the IRAF announcement with instructions. Please, note that the SPTABLE package is already configured to recognise all the ESO 1d science spectral data products, so no other action is needed to start working with the data. The IRAF/SPTABLE Spectrum Table Database is already fully setup for all current ESO spectral types, and therefore e.g. the xonedspec splot task can be used right away.

The other mentioned tools (excluding the IRAF ones) are available directly within the Linux distribution of Scisoft; if you have Scisoft install there is no further installation required. Otherwise, instructions on how to install them, or to launch them directly from the web, are provided below. None of the mentioned tools is available in the Scisoft Mac distribution. To download Scisoft, please visit: http://www.eso.org/sci/software/scisoft/

**IRAF generation of an ASCII file**

Within IRAF, using the tables external package (by STScI), tprint can be used to output to an ascii file the data in a tabular format (not as a list of arrays):

```
cl> tables
tables> ttools
ttools> tprint a.fits[1] pwidth=200 > a.txt
tools> !more a.txt

#  Table b.fits[1]  Tue 17:22:44 22-Oct-2013

# row          WAVE                      FLUX                       ERR                    SKYBACK
#          angstrom erg cm**(-2) s**(-1) angstrom**(-1) erg cm**(-2) s**(-1) angstrom**(-1) erg cm**(-2) s**(-1)
angstrom**(-1)

   1        3648.393                2.265424E-17               8.562715E-18               6.328494E-16
            3653.916                3.985785E-17               8.668060E-18               6.341451E-16
            3659.438                2.100061E-17               8.322070E-18               6.107177E-16
            3664.961                3.735991E-17               8.230550E-18               5.808179E-16
```

**IRAF generation of a scalar FITS table**

It is also possible to use a combination of tdump, tprint, and tcreate to generate a normal table via an ascii file.

- Input file: a.fits (in the ESO Science Data Product standard format (each data arary in one cell))
- Output file: table.fits (a FITS binary table where each cell contains a scalar)

tcreate is used to generate the desired output file.

tcreate requires two input files that must be generated: the column definition file, and an ASCII file containing the data.

The column definition file can be created using tdump, while the ASCII file containing the data can be created using tprint.

```
cl> tables
tables> ttools
ttools> tdump a.fits[1] cdfile="a.cd" > dev$null   # to create the column definition file: a.cd
```

The a.cd (column definition file) shows something similar to:

```
WAVE            R[1015]     % 15.7g angstrom

FLUX            R[1015]     % 15.7g "erg cm**(-2) s**(-1) angstrom**(-1)"

ERR             R[1015]     % 15.7g "erg cm**(-2) s**(-1) angstrom**(-1)"

SKYBACK         R[1015]     % 15.7g "erg cm**(-2) s**(-1) angstrom**(-1)"
```

The square brackets in the second column indicate the array nature of the cells. The a.cd file must be edited and the brackets removed, as in:

```
WAVE            R     % 15.7g angstrom

FLUX            R     % 15.7g "erg cm**(-2) s**(-1) angstrom**(-1)"

ERR             R     % 15.7g "erg cm**(-2) s**(-1) angstrom**(-1)"
```

SKYBACK R % 15.7g "erg cm**(-2) s**(-1) angstrom**(-1)"

The ASCII tabular data can be created using the tprint command, as already seen earlier, but with some extra parameters:

ttools> tprint a.fits[1] pwidth=200 orig_row=no showrow=no showhdr=no showunits=no > table.dat

And finally the command to generate the output FITS "scalar" binary table:

ttools> tcreate table.fits a.cd table.dat

- Tags
- FAQ
- Phase 3