Experiment 301: Importing Data into Python

Aim

To import and manipulate data files in Python and Spyder.

References

1. `http://docs.scipy.org/doc/numpy/reference/generated/numpy.loadtxt.html`

Introduction

This is intended as a "cheat sheet" to help Stage 3 students import data files into Spyder.

Spyder – Drag-and-drop

Spyder is an interactive graphical user interface (GUI) for Python that allows us to easily import the contents of a data file into the workspace as a matrix. If the data set is contained in one of the following:

- GIF images (*.gif),
- NumPy zip arrays (*.npz),
- Matlab files (*.mat),
- Pickle files (*.pkl, *.pickle),
- TIFF images (*.tif),
- PNG images (*.png),
- Text files (*.txt),
- JPEG images (*.jpg),
- NumPy arrays (*.npy)
- HDF5 (*.h5),
- CSV text files (*.csv),
- JSON files (*.json),
- Spyder data files (*.spydata)

the file itself can be dragged on to Spyder's "Variable explorer" (Fig. 1). Here we use `data.txt` as an example which was taken in Experiment 341: Spectrum of Hydrogen. The data is imported as a three-column matrix which we name `mydata`. Each column is separated by a comma `,` which is identified in `Column separator`. Since there is nothing at the end of each line, we can just leave `Row separator` as `EOL`, or end-of-line. We have also chosen not to import the header on the first seven rows of the file in `Skip rows`.

Fig. 2 is a preview of the matrix to be stored in the workspace. The data should be imported as `array`, since arrays are more easily manipulated during data analysis.

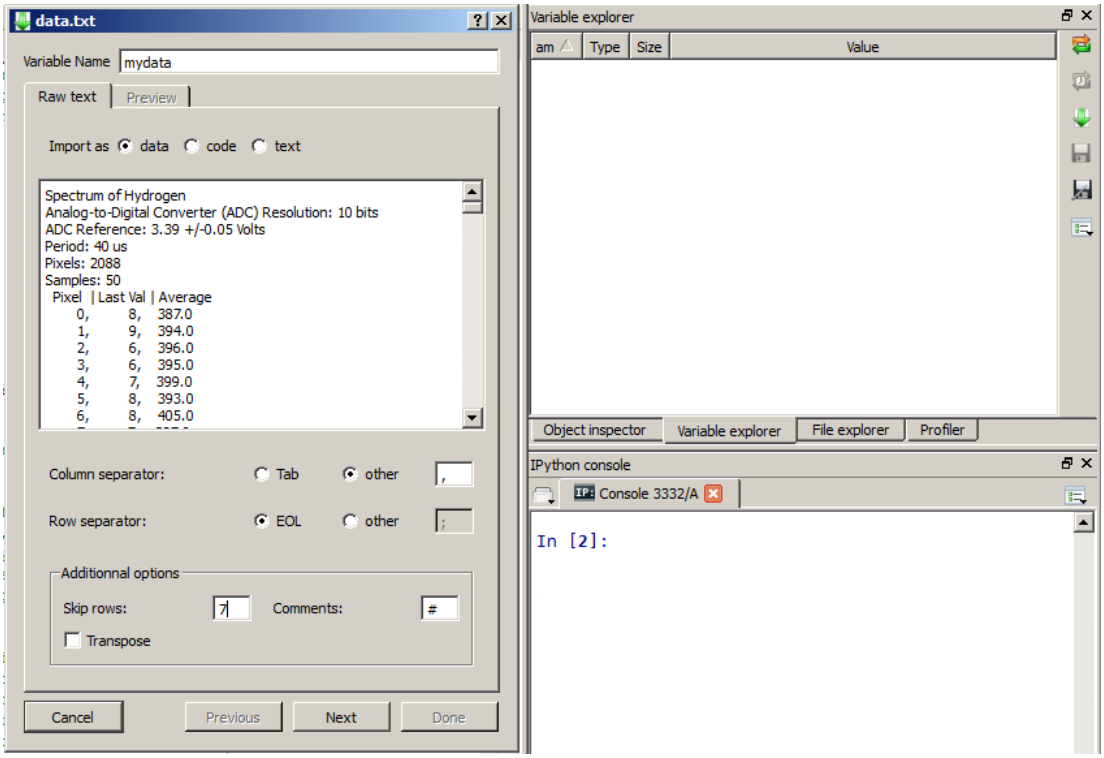

Figure 1: After `data.txt` is dragged on to "Variable Explorer", we get the pop-up window on the left.

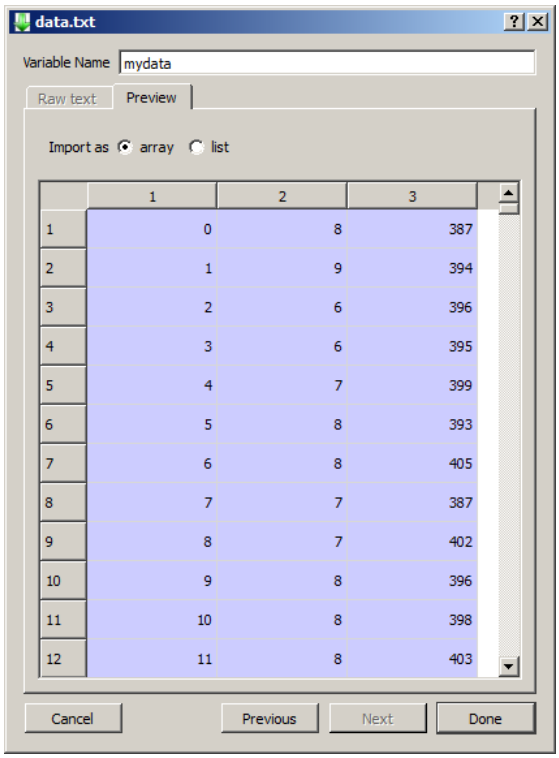

Figure 2: Previewing the data.

Now to separate the columns into individual Python arrays, use:

```
x = mydata[:,0]    # first column
y = mydata[:,1]    # second column
z = mydata[:,2]    # third column
```

Python – Code-based method

The above section is useful when we want to quickly check one data set. However if we are not in the GUI environment or if we want to return to the data analysis at a later time, this method may not be convenient. Instead, the following piece of code shows how a `.tka` file saved from the Gamma Acquisition software (Experiment 352: Interaction of Gamma Rays, say) can be imported into Spyder:

```
import numpy as np
w = np.loadtxt('data4.tka')
```

If the data file is not in the same folder as your Python programme, ensure that the correct path is entered. For example

```
w = np.loadtxt('H:/e352/data4.tka')
```

if it is saved somewhere on your home drive.

Note that the values in `.tka` files are just separated by white space. Any symbols which separate values and rows which are not imported into Spyder must be specified in `delimiter` and `skiprows` respectively. Returning to the Spectrum of Hydrogen example, we use

```
mydata = np.loadtxt('data.txt', delimiter=',', skiprows=7)
pixel = mydata[:,0]         # first column is pixel number
count = mydata[:,2]         # third column is amplitude
```

and all this can be achieved in one line:

```
pixel, _, count = np.loadtxt('mydata.txt', delimiter=',', skiprows=7, unpack=1)
```

The `unpack` command loads the first and third columns into arrays `pixel` and `count` respectively. We do not need to analyse the second column, so we use an underscore which tells Spyder not to import this.

NI ELVIS II plots

When taking data with the oscilloscope and Bode analyser functions of the NI Elvis II, it is useful to be able to export the plots to Python. To save the results, first Stop the scope then click the Log button on either the oscilloscope or the Bode analyser. For this example we shall use files named `SCOPE.txt` and `BODE.txt` to analyse the oscilloscope log and Bode data respectively.

For the Bode data, simply load `BODE.txt` using `loadtxt` to end up with arrays containing the frequency, gain and phase.

The oscilloscope logs require more work, as the data contain both spaces and colons (`:`). The following shows a possible way of importing the time and voltage data, the second and third columns respectively, taken from one channel of the scope:

```
import numpy as np

elvis = np.loadtxt('SCOPE.txt', dtype='string', skiprows=5)

# get time data
time = elvis[:,1]
time = np.array([s.replace(':','') for s in time])   # removes :
time = time.astype(np.float)                         # turns string into floats
time = time-time[0]                                  # start time at t=0, subtract
                                                     # initial time from each data point

# get voltage data
y0 = np.array(elvis[:,2])
y0 = y0.astype(np.float)
```

If we take data from both oscilloscope channels, we will also need to import the fifth and sixth columns to get the time and voltage readings from the second channel.

R. Au-Yeung: 17 November 2014