

File-io with Python:

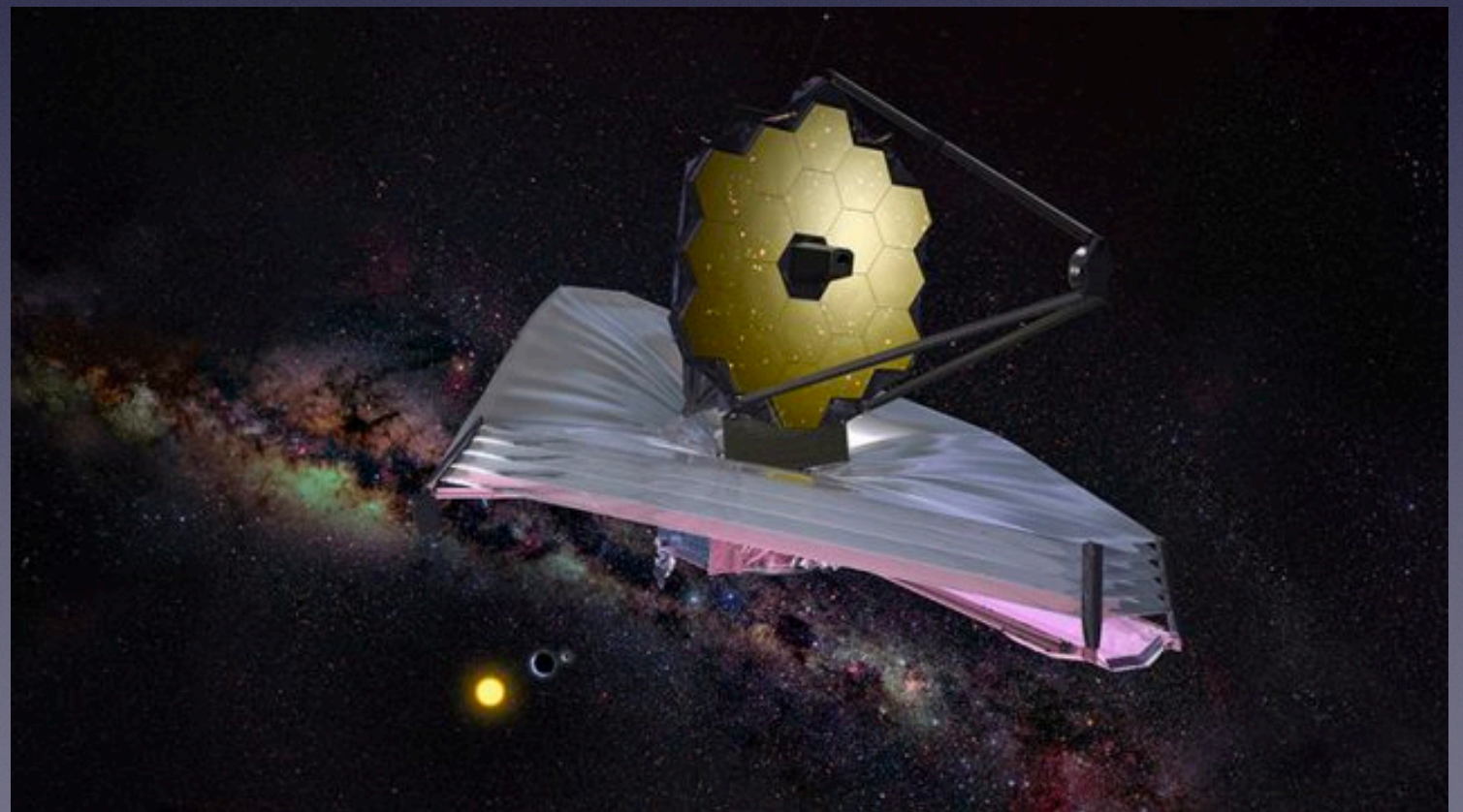
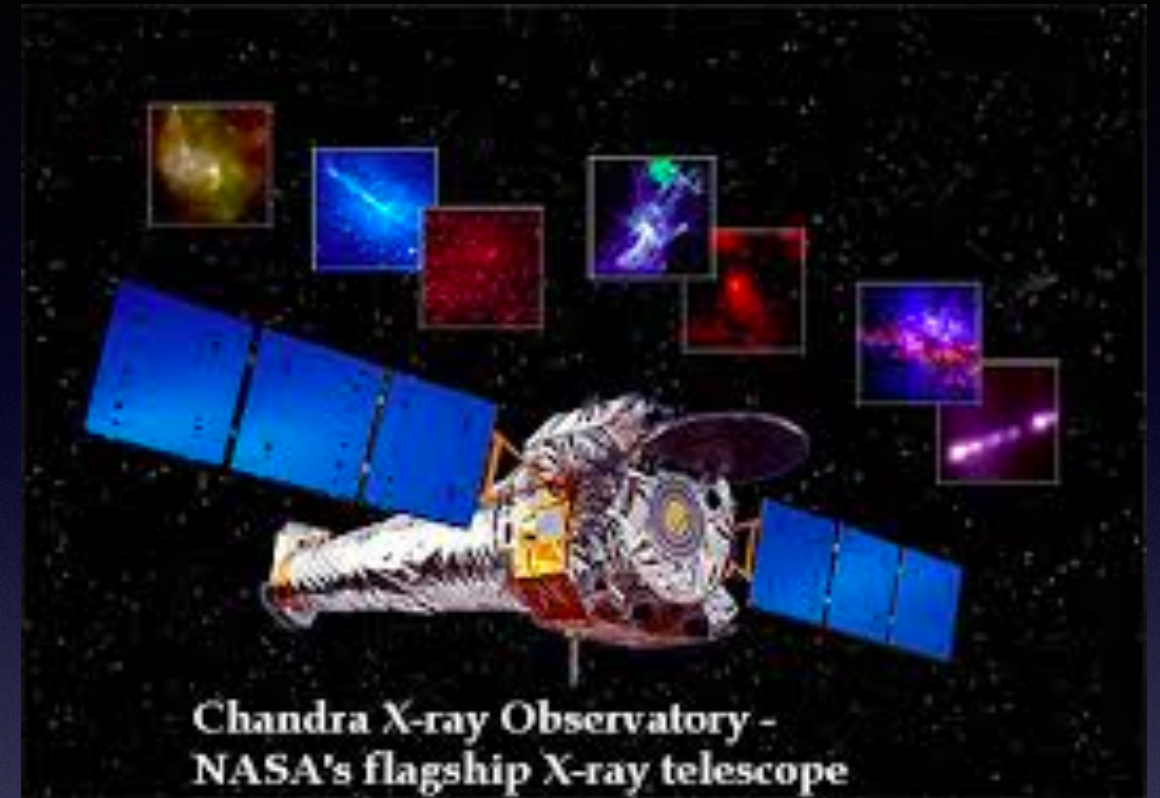
Reading and writing files

Instructor:
Oscar A. Chavez Ortiz

Objectives

- Learn how to read in .txt and .fits files
- Extract necessary data from .txt and .fits files
- Learn astropy's file-io package
- How to maneuver around a FITS file

Background



Introduction to File Handling

The concept of reading in data from a file is similar to reading a letter.

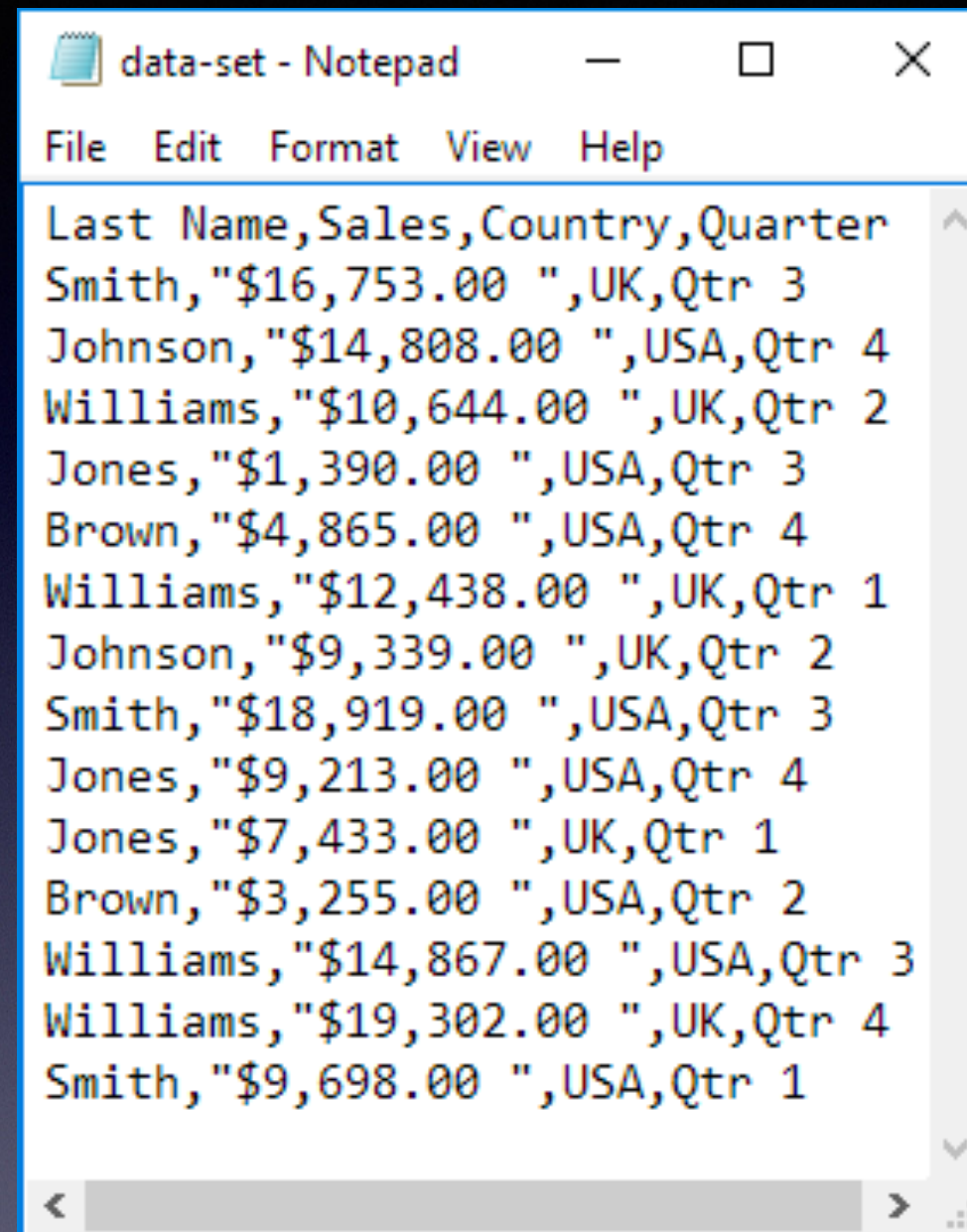
What do you do as you open a letter?

- First we check that the letter is for us
- Then we open the letter
- Read the content of the letter
- Then place it back and close the letter.

Python Version

- Check to make sure that the file you want to open is the right one, write into the opening function the correct filename
- Open the file using some io packages depending on the extension of the file. Each extension has its own functions to open it
- Reading the file by extracting the necessary data from the file
- Close the file once you have extracted everything of use to free up space and memory

Text files



```
data-set - Notepad
File Edit Format View Help
Last Name,Sales,Country,Quarter
Smith,"$16,753.00 ",UK,Qtr 3
Johnson,"$14,808.00 ",USA,Qtr 4
Williams,"$10,644.00 ",UK,Qtr 2
Jones,"$1,390.00 ",USA,Qtr 3
Brown,"$4,865.00 ",USA,Qtr 4
Williams,"$12,438.00 ",UK,Qtr 1
Johnson,"$9,339.00 ",UK,Qtr 2
Smith,"$18,919.00 ",USA,Qtr 3
Jones,"$9,213.00 ",USA,Qtr 4
Jones,"$7,433.00 ",UK,Qtr 1
Brown,"$3,255.00 ",USA,Qtr 2
Williams,"$14,867.00 ",USA,Qtr 3
Williams,"$19,302.00 ",UK,Qtr 4
Smith,"$9,698.00 ",USA,Qtr 1
```


Opening .txt files with Numpy

The way we open a .txt files is by using numpy's file io functions loadtxt and genfromtxt

Syntax for this would be:

```
f = np.loadtxt('filename.txt')
```

Or

```
f = np.genfromtxt('filename.txt')
```


Saving/Writing file

To save a file using numpy use the savetxt function and this will save your data into a text file.

```
np.savetxt('filename', data)
```

You can also use np.save and np.savez. However these are saved to .npy and .npz files respectively and to open these requires you to use np.load.

Examples!!

FITS Files

FITS stands for Flexible Image Transport System and is the main storage of astronomical data. This can store information regarding object location, flux, can store images, Astropy Tables.

To deal with these types of files we need to utilize a package from astropy that deals specifically with handling fits files. This comes from the [astropy.io](https://docs.astropy.org/en/stable/astropy.io/) library and we need to import the fits package.

This package comes with some functions that allow us to open these types of files.

Opening FITS files

First import the fits package into your program using:
`from astropy.io import fits`

Then to open a file you will use the open function from fits.

Its customary to assign this opened file to a variable called the hdu. The hdu stands for Header Data Unit.

```
hdu = fits.open('filename.fits')
```


Maneuvering around FITS files

Keyword commands to help you navigate FITS files:

hdu.info() - This command gives you the information of what resides within the fits file.

```
In [14]: hdulist
Out[14]:
[<pyfits.core.PrimaryHDU object at 0x90d658c>,
 <pyfits.core.BinTableHDU object at 0x90e008c>]

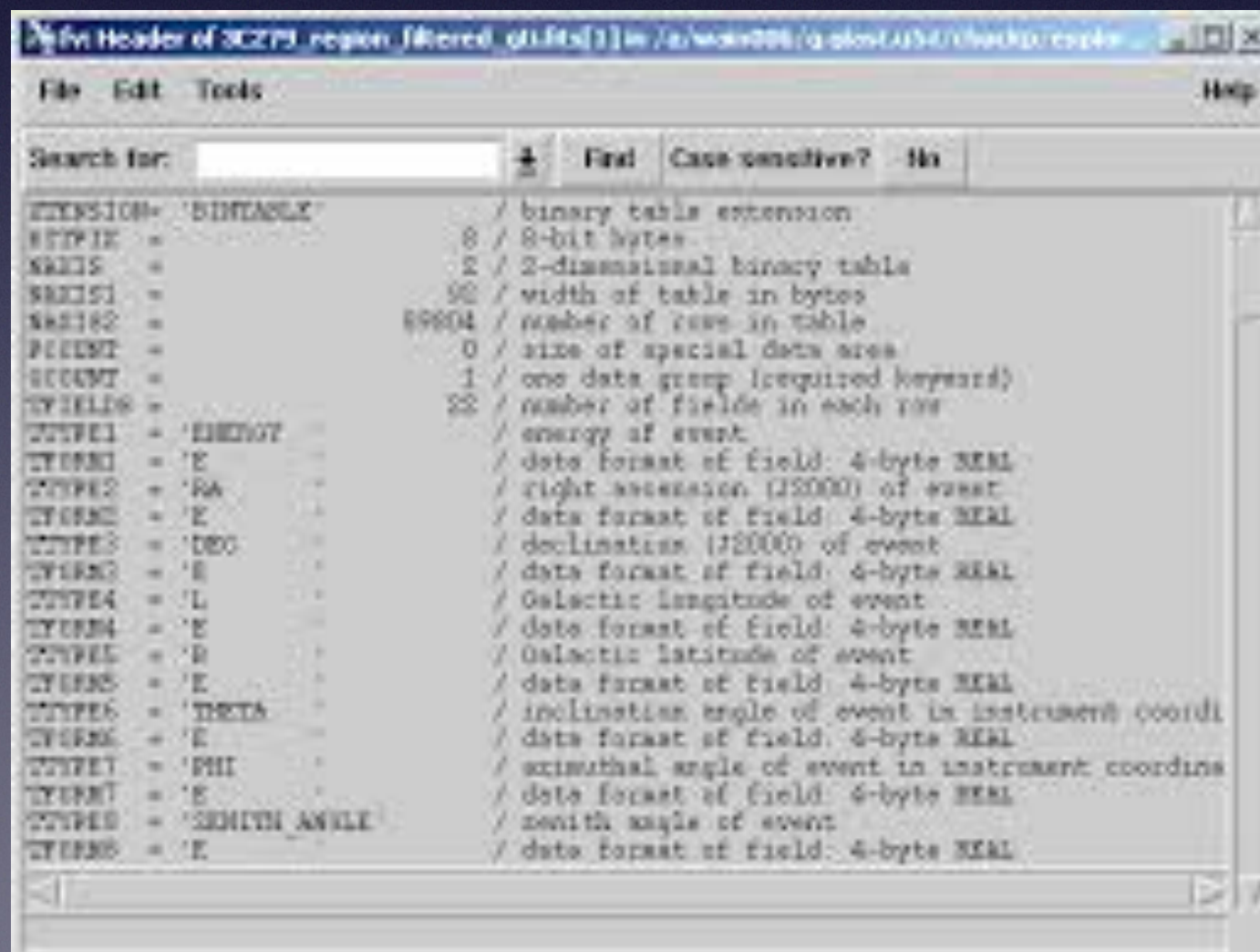
In [15]: hdulist.info()
Filename: /home/quijote/Desktop/Summer/img/kaCal0bjBest.fit
No.    Name      Type      Cards  Dimensions  Format
0      PRIMARY   PrimaryHDU  4      ()          uint8
1                      BinTableHDU  26     336R x 9C   [1J, 1D, 1E, 1D, 1E, 5E, 5E, 5J, 5

In [16]: for hdu in hdulist:
.....:     print hdu
.....:
.....:
<pyfits.core.PrimaryHDU object at 0x90d658c>
<pyfits.core.BinTableHDU object at 0x90e008c>
```


Maneuvering around FITS files

Keyword commands to help you navigate FITS files:

hdu[0].header - This command gives you the information of the Primary extension. This typically holds information about the telescope, instrumentation and object.



Extensions

Often times when you open a FITS file the data will not be stored in the PrimaryHDU but rather be in one of the extensions. Extension can be seen when you run the hdu.info() Command.

Everything that is not the PrimaryHDU is considered an extension and to see what is inside these extension you need to see the header of each extension.

```
hdr_ext1 = hdu[1].header
```

```
hdr_ext2 = hdu[2].header
```

Etc. for subsequent extensions.

Getting Data

Data are stored in FITS files in two main ways either as an Image or as BinTable. To see what kind of data is stored in your fits file you use the `.info()` command. And under Type you will see the data stored. If data is stored in PrimaryHDU it won't show up but you can check the dimensions to see if data is stored.

BinTableHDU

BinTableHDU store data in a table and you access the data by accessing the column name.

First make a table object this can be done using:

```
table = hdu[1].data
```

Then to get the columns associated with this table use:

```
col = table.columns
```

Then to access data you use:

```
data = table['Column Name']
```


ImageHDU

ImageHDU typically hold an N-D array but the most common ones you will see are 1D, 2D and/or 3D arrays. These do not necessarily hold image as can be seen by the 1D arrays stored.

If you know which extension the ImageHDU is located you can get the data using:

```
data = hdu[extension].data
```

This will return the N-D array that is stored there and you can find the size of the array from the `.info()` command.

Examples!!