Introduction :

Instructions

Working with FITS files

Notes

One of the most widely used formats for astronomical images is the *Flexible Image Transport System*. In a FITS file, the image is stored in a numerical array, which we can load into a NumPy array.

FITS files also have headers which store metadata about the image.

FITS files are a standard format and astronomers have developed many libraries (in many programming languages) that can read and write FITS files. We're going to use the Astropy module.

The following code shows how we can open a FITS file and print out its header information:

```
from astropy.io import fits
hdulist = fits.open('image0.fits')
hdulist.info()
```

```
Filename: test0.fits
No.    Name         Type      Cards   Dimensions   Format
0    PRIMARY     PrimaryHDU       7   (200, 200)   float64
```

Instructions

Reading in FITS files

Notes

Opening a FITS file in Astropy returns a HDU (*Header/Data Unit*) list. Each HDU stores headers and (optionally) image data.

The header contains metadata about the HDU object, e.g. its dimensions and data type. Every HDU can contain image data. The first HDU is called the *primary HDU*.

If we want to access individual HDUs, we can index the HDU list object returned by `fits.open`. The image data can be accessed using the `data` attribute:

```
from astropy.io import fits

hdulist = fits.open('image0.fits')
data = hdulist[0].data

print(data.shape)
```

The image data is conveniently stored in a NumPy array, so we can operate on it directly. This example prints the dimensions of the image in the primary HDU.

Instructions

Visualising FITS images

Notes

You often want to visualise the image data stored in FITS files. We can do this using the plotting library matplotlib.

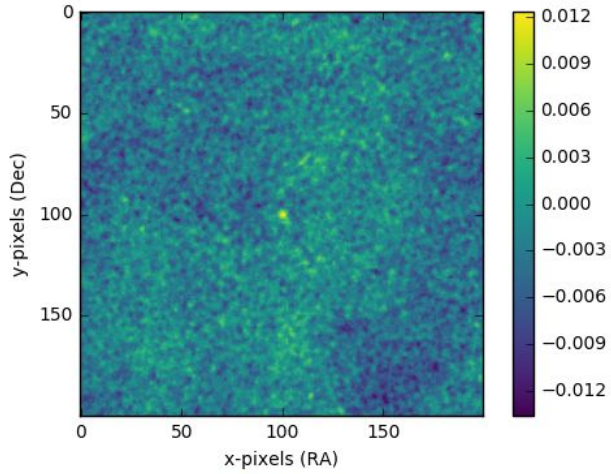
This example creates a 2D plot from the previous FITS image:

```
from astropy.io import fits
import matplotlib.pyplot as plt

hdulist = fits.open('image0.fits')
data = hdulist[0].data

# Plot the 2D array
plt.imshow(data, cmap=plt.cm.viridis)
plt.xlabel('x-pixels (RA)')
plt.ylabel('y-pixels (Dec)')
plt.colorbar()
plt.show()
```

The code above produces the following image:



2D plot of a FITS image

Assignment:

Instructions

Read a FITS file

Problem | Solutions

Write a **load_fits** function that loads in a FITS file and finds the position of the brightest pixel (i.e. the maximum value) in its image data. To make this function work for arbitrary files, pass the name of the FITS file as an argument to the function.

Using the file **image0.fits** from the previous examples, your program should work like this:

```
>>> load_fits('image0.fits')
(100, 100)
```

The brightest pixel in this image is exactly in the centre of the array, as you can check visually by plotting it: