ASTR 400B Homework 2

Due: Jan 30th 2025 5 PM

In this assignment you will write a code to read in the data file you symbolically linked to your home directory in the first homework, MW_000.txt using Python. We are also going to get used to keeping track of units using AstroPy.

1 Storing Data Locally

If you wish to download a file from nimoy to your laptop you can do this by:

1. start up a VPN connection
2. navigate from the command line to the desired place on your computer that you want to store the file.
3. in the command line, type: sftp username@nimoy.as.arizona.edu
4. this will prompt you for your password, then you will be connected to your nimoy home directory (like with ssh)
5. navigate to wherever the file is on nimoy
6. type: get filename (where filename is the file you want. like MW_000.txt)

The file will show up on your computer in the directory you listed the sftp command.

2 Organization of Data File

Each files is organized as follows:

- First Row is the time in units of Myr (equivalent to SnapNumber x 10/0.7)
- Second Row is the total number of particles
- Third Row describes the units of the columns that follow after the fourth row
- Fourth Row describes the header name for each column that follows

The remaining rows contain the particle data:

1. First Column *type*: Particle Type. Type 1 = Dark Matter, Type 2 = Disk Stars, Type 3 = Bulge Stars

2. Second Column m : Mass of the particle in units of $10^{10} M_{\odot}$

3. Third-Fifth Columns *x, y, z* : Position (x,y,z) in kpc measured from the center of mass position of the Milky Way

4. Sixth-Eighth Columns *vx, vy, vz*: Velocity (vx,vy,vz) in km/s measured in a Cartesian coordinate system centered on the location of the Milky Way

Each column is delimited by tabs (spaces)

3 Create a ReadFile Program

Create a new python script, or Jupyter notebook, entitled *ReadFile*. If you are creating a python script, make sure to give the .py extension. We will use this code throughout the course, so if you start with a Jupyter notebook, you will need to download the final product as a python script (open the File tab, select Download as Python) so that you can import it in later programs (like in Part 4).

In *ReadFile*, create a function called *Read* that will :

1. open and read the MW_000.txt data file.

2. returns: the time, and total number of particles as variables (first two lines of the file).

3. returns: particle type, mass, x,y,z, vx,vy,vz columns as a data array

To do this, follow these steps in Python (watch your indentation!):

- the first lines of the code need to import relevant modules: NumPy and AstroPy
  import numpy as np
  import astropy.units as u

- Define the function *Read* that takes the name of the file as input.
  def Read(filename):

- open the file
  file = open(filename,'r')

- Read the first line and store the time in units of Myr. For example:
  line1 = file.readline()
  label, value = line1.split()
  *time = float(value)*u.Myr*
  Do the same for the 2nd line for the *total* number of particles. Next time you run readline(), it will read the second line.

- close the file
  file.close()
- store the remainder of the file using the NumPy function np.genfromtxt. This allows you to use the column header information (4th line of the file, starting with #)

  data = np.genfromtxt(filename,dtype=None,names=True,skip_header=3)

  parameters:

  - "dtype=None" tells python to interpret the column's data types as if you were assigning variables, whereas "delimiter=None" tells it that the columns are separated as white space, which is the default setting.
  - "skip_header=3" means skipping the first 3 lines .
  - the flag "names=True" creates arrays to store the data with the right labels
  - the labels are : "type, m, x, y,z, vx, vy, vz"
- return the time, total number of particles and the full data array as separate quantities (not a list)
- test your code!! make sure it returns the right properties. For example, if you wanted to know the particle type of the 2nd particle in the file you would write *print(data['type'][1])* and then check this against the real value in MW_000.txt. Make sure to test the other data labels too. Document your tests by taking a screenshot or by compiling them in a Jupyter notebook.
- If you are in a Jupyter Notebook, download the file as a .py file (called ReadFile.py). This way you can import it into the next code you need to write. Note, don't forget to delete the tests from the actual .py script so that the function can be imported in Part 3.

4 Create a new program, ParticleProperties, that uses ReadFile

Create a new python script or Jupyter notebook, entitled *ParticleProperties*. In that code, create a new function called *ParticleInfo*, which takes as inputs: filename, particle type and particle number. The function should return the following properties for **any given particle of any given type** (Disk, Halo, etc):

1) Magnitude of the distance in kpc;
2) Magnitude of the velocity in km/s;
3) Mass in units of $M_\odot$.

Round the distance and velocity values to 3 decimal places using *np.around(value,3)*.

Since you need to read in the file again, call *Read* from *ReadFile* by adding the following to the list of modules: *from ReadFile import Read*

Hints:

- To read in, say, the x position of *all* particles: *x = data['x']* . Don't forget to assign kpc as the unit using AstroPy (or km/s for velocity).
- To create an index for all particles with a given property, use the NumPy function *np.where*. For example, if I wanted to store all x components of particles that are more than 2 kpc away in the x component, I would write:

  index = np.where(data['x'] > 2)
  xnew = data['x'][index]

5 Prove Your Codes Work!

Use *ParticleProperties* to determine the following properties of 100th disk particle of the Milky Way at SnapNumber 0 (be careful of indexing to get the right particle!):

1. 3D Distance,
2. 3D Velocity
3. mass
4. Convert the 3D Distance of the particle to LIGHTYEARS (3 decimal places) using AstroPy (hint: use the "to" function - google it if you're not sure !).

6 Homework Submission

- You must DOCUMENT your code . Explain EACH step. I'm not joking. Format that I'm expecting: Three lines at the top of the code that describes the function, the inputs and what the function returns. Then a comment next to every variable that explain what it is.

  # This function adds up three distances

  # Inputs: a (x coord in kpc); b (y coord in kpc); c (z coord in kpc)

  # Returns: Sum of a, b, c in kpc

  def AddItUp(a,b,c):

  d= a+b+c *# Adding up the inputs*

  return d

- Clone your class repository (or go to your cloned version).

- Create a directory called Homeworks/Homework2. Save your code and answers to question 5 in that directory. Your answer to question 5 should be saved EITHER:
    1. As part of your Jupyter notebook solution: in this case please document your jupyter notebook.
    2. Take a screen shot of your python output (with the relevant print statements for each quantity) from the command line.
- Push your Homework2 directory to your 400B repository on GitHub