ASTR 400B In Class Lab 7

March 15th 2018

The goal of this lab is to make density plots with contours and to re-orient the data to desired viewing angles (i.e. align the disk of a galaxy with the average orbital angular momentum vector **L**).

1. Update your clone of the Class Github Repo (git pull)
2. Under InClassLabs/InClassLab7 you should find a template jupyter notebook and script called 'InClassLab7_Template'. These templates include the same instructions as those provided below.
3. There is also a data file called M31_000.txt. All columns are the same as in previous data files.

1 Part 1: No action required, simply run the cells/code

- We have defined find_confidence_interval() and density_contour() from InClassLab6, which are functions that allow you to plot the 68%, 95% and 99% density contours.
- The M31 data file and all relevant disk particle properties are also loaded already. Note the data has been corrected for the COM.
- The first plot is already set up for you. It should plot the M31 face-on disk (or the z-plane) in a logarithmic scale.

2 Part 2: Velocity-weighted plots

- Edit the code for the second plot to weight the disk particles by the line of sight velocity. Use plt.scatter() where the argument"c" should be the line of sight velocity component. Think about the last two labs to figure out what the line of sight velocity component is for a face-on disk.
- Now write a mask (index) that allows you to choose only the particle that have a *total* velocity between -300 and +300 km/s. These are the particles that most closely match M31's rotation curve.

- Using this mask, refine the previous plot to only show the face-on disk particles weighted by the line of sight velocity for the particles that satisfy $-300\,km/s < v_{tot} < 300\,km/s$.

3 Part 3: Rotate the disk particles to align with the z-axis

- The function RotateFrame() takes in arrays of the 3D positions and velocities and returns 3D positions and velocities that are reoriented such that the disk is aligned the with orbital angular momentum vector **L** of the disk.
- Rotate the positions and velocities of your disk particles while applying the mask from Part 2 (i.e. v_{tot} cut) at the line that starts with *# rn, vn =* (NOTE: If you were interested in looking at warps in the disk, you might want to create a mask that selects only the subset of particles in the innermost 5 kpc or so. As written, the RotateFrame() function might otherwise average out the warp structures, so test out different masks.)
- In the next plotting code section, use plt.scatter() again to plot the face-on disk, this time with the rotated particles. The data should still be weighted by the line of sight velocity, but note that this will have to be the rotated line of sight velocity too. If your rotation worked, the face-on disk should look fairly circular.
- Use the final set of plotting code to plot the edge-on disk weighted by the line of sight velocity. Note that the line of sight velocity will be different than it was for the face-on disk.