Computational Astrophysics
Assignment - 1

Kiran L
SC17B150

18 September 2020

1 Introduction

The basic laws of Physics that govern the evolution of system in space and/or time are usually expressed in the form of equations. Examples include Newton's law of gravity, Maxwell's equations of electromagnetism, and Schrodinger's equation for describing the behavior of a quantum system. Not in all problems can we obtain closed form solutions to the equations, in which case, one turns to a computer for numerically analysis the solution of the equation under given initial or boundary conditions. For example, the dynamical evolution of massive objects in space and time, as governed by Newton's law of motion for more than three objects does not have an expression, instead results in coupled equations.

The invention and revolution in computer science and electronics has opened avenues to understand various system in the universe, with varying scales of time and space. The invention of powerful GPUs for parallelizing possible operations has reduced the total wall time of computations.

Computation can be used to model or simulate very complex physical situations leading to further insights. For example, in the Solar System, Newton's law of gravity cannot be applied with perfect accuracy because there are eight planets and other small bodies orbiting the Sun and each one exerts gravity on all the others. When astronomers started to simulate the long term behavior of the Solar System, they made some surprising discoveries. They found that small changes in the starting configuration accumulate and lead to large eventual differences in the orbits. This is sometimes called the "butterfly effect". They also found that the system could become chaotic, where some objects stopped being stable and repeatable in their orbits and changed their positions or were even ejected from the system entirely. Now that we know of many planetary systems beyond the Solar System, simulations are extremely important in understanding and predicting their properties.

On the largest scales, where astronomers apply computers to simulate and understand large systems such as galaxies and the universe, most of the massive objects such as stars and galaxies and clusters can be treated as “particles”. A three dimensional grid of space is created virtually in the computer, it is populated with sources of choice, the gravity force of each source on every other is calculated, all the sources respond to the forces by moving slightly in the virtual space, then the gravity forces are recalculated at the new positions, and this process repeats for a long as necessary. In cases of very large spatial scales involved in the problem, the expansion of the universe could be needed to be considered. For a general N body simulation, the number of interactions to be considered ($\frac{N(N-1)}{2}$) increases drastically with increase in N, in turn increasing time and hence, computational complexity.

Computers allow problems to be attacked by brute force but as programmers, it is necessary to try and reduce redundent calculations and improve on algorithms to hasten the simulation. For example, since gravity weakens with the inverse square of distance, in practice it’s not necessary to calculate the force of every particle on every particle. It’s a good approximation to only consider the nearest particles and estimate the net force from all the others.

Thus, computers are ubiquitous in astronomy and simulations are contributing to the health and advancement of the subject. Astronomy stands on a sturdy tripod of observations, theory, and simulations. Observations and data lead the subject since discoveries are being made all the time. Theory is a crucial backdrop for understanding the observations. And simulations play a role in understanding complex situations and suggesting new observations.

In the present report, we analyse the binary systems (planet and star, star and star, star and blackhole and alike). Assumptions made in simulating these systems are enumerated as follows:

1. dynamical stability (center of mass in between the two components and the orbital phase of the two offset by 180^o)
2. perfectly circular orbits
3. equal density of both components (hence, sizes proportional to their masses)

Since, the orbit of the components is known, there is no need for solving the Newton’s equation for force of gravitational interaction between the two objects and solving for the orbit using time stepping methods such as Runge-Kutta (RK4) or Euler’s method. With the masses of the binary components as free parameters (to be input by the user) the dynamical evolution of the system is simulated in the form of an animation. All animation results presented here are 2D simulations. In case of 3D animations is it also possible to demonstrate the conservation of angular momentum by showing that the plane of orbit does not vary with time. Below we present the results obtained along with the observations.

2 Case 1: $M_1 = M_2$

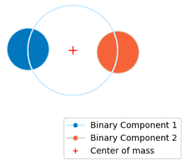

Figure 1: Animation in Cartesian plot with binary component size linearly proportional to their mass

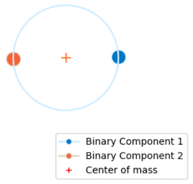

Figure 2: Animation in Cartesian plot with binary component size logarithmically proportional to their mass

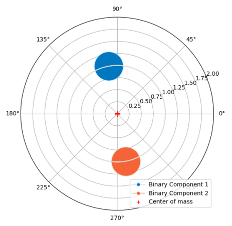

Figure 3: Animation in polar plot with binary component size linearly proportional to their mass

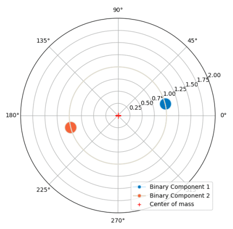

Figure 4: Animation in polar plot with binary component size logarithmically proportional to their mass

From the results displayed above show the case in which the masses of both binary components are equal. We observe 180^o phase difference in the orbit of the same, with the distance from the center of mass (COM) being identically same. Since, the density is assumed to be identical in both components, the two objects in the simulation could possibly represent two stars, galaxies (of similar/identical characteristics), or a binary planet (whose center of mass could be revolving around the center of mass of a larger system).

3 Case 2: $M_1 = 10M_2$

From the results displayed above show the case in which the mass of one of the components is 10 folder greater than the other. Once again, by since we have considered dynamically stable system, we observe 180^o phase difference in the

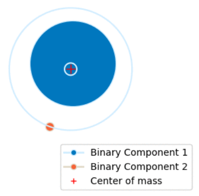

Figure 5: Animation in Cartesian plot with binary component size linearly proportional to their mass

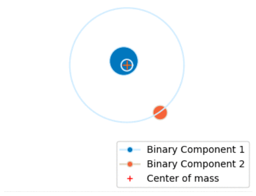

Figure 6: Animation in Cartesian plot with binary component size logarithmically proportional to their mass

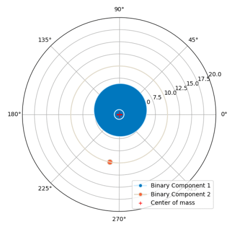

Figure 7: Animation in polar plot with binary component size linearly proportional to their mass

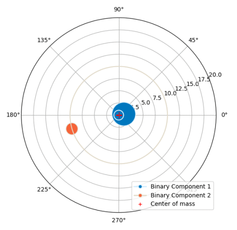

Figure 8: Animation in polar plot with binary component size logarithmically proportional to their mass

orbits. The COM can be seen to lie almost entirely inside the more massive component. Here, the systems are simulated in both Cartesian and polar plots with both cases where the size of the component is linearly and logarithmically proportional to their masses. We observe that a non-linear relation between the mass and the size points to possibly two different type of sources - with entirely different characteristics. Thus, this system could possibly represent a non-contact binary system composed of a **white dwarf** and a massive **giant** or **super giant** star, which are separated enough to prevent any mass flow from one to another (which, if the case, would require more detailed evolution to be considered, since it can results in various possible end results such as a stellar explosion nearly destroying the more massive component and giving rise to newer exotic systems such as Neutron star or Blackhole!). This system, ignoring the fact the the sizes are proportional to their masses (which inherently assumes that the densities are same), can possibly represent a **super-Jupiter** and **Sun** like system (mass of Jupiter is nearly 4 orders of magnitude less than that of Sun). In case of sizes of the components being linearly proportional to their mass, the system can possibly be a binary stars system composed a **giant** or **super-giant** and a **normal** (say, **main sequence**) star.

4 Case 3: $M_1 = 10^6 M_2$

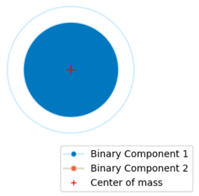

Figure 9: Animation in Cartesian plot with binary component size linearly proportional to their mass

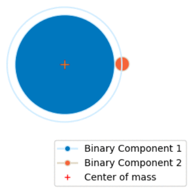

Figure 10: Animation in Cartesian plot with binary component size logarithmically proportional to their mass

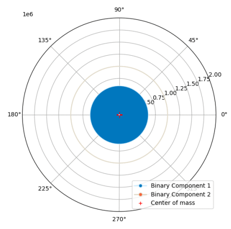

Figure 11: Animation in polar plot with binary component size linearly proportional to their mass

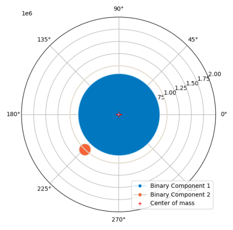

Figure 12: Animation in polar plot with binary component size logarithmically proportional to their mass

From the results displayed above show the case in which the mass of one of the components is 10^6 times greater than the other (6 orders of magnitude difference in masses). As the system is assumed to be dynamically stable, we observe 180^o phase difference in the orbits. The more massive component can be seen to lie at exactly the center of nass with a slight wobbling motion visible in plots where the sizes are considered to be non-linearly proportional to the mass. The systems are simulated in both Cartesian and polar plots with both cases where the size of the component is linearly and logarithmically proportional to their masses. We observe that a non-linear relation between the mass and the size points to possibly two sources of different physical and chemical properties.

Thus, this system could possibly represent a non-contact binary system composed of a **Super massive black hole** and a (relatively) low mass star such as those belonging to luminosity classes III to V (with probability reducing with change of luminosity class from III - normal stars, to V - giant stars). In that case, once could possibly need to consider the Schwarzschild radius of the more massive component, as in case of the same being within the event horizon of the super massive black hole, gets disrupted tidally (spaghettification) leading to dynamically unstable situations. Here, as we have assumed apriori that the system being simulated is dynamically stable, we ignore those cases where the lower mass component is close to or inside the event horizon and assume that the separation between the two is sufficiently large to neglect such interactions. Hence, we assume no mass transfer between the two and that both are compact throughout the simulation. In the simulations with sizes of the components being linearly proportional, we are barely able to see the low mass component. The low mass component is visible only in the case where the size and mass relation is assumed to be non-linearly related. Further, this system can represent an exoplanet system, where one is unable to directly image the planet, and needs to infer its presence and its properties from the spectroscopic studies of the larger mass component. For example, the Mars and Sun like system, where the mass of Mars is 6.39×10^{23} kg which is 7 orders of magnitude lower mass (and different composition) than the Sun. Further, this could even represent (though not commonly observed) a large mass planet (Jupiter like) and its satellites, revolving around it. For example, mass of Jupiter is 1.898×10^{27} kg while the mass of Europa is 4.8×10^{22} kg, which is nearly 5 orders of magnitude lower than that of the planet. Other satellites such as Amalthea has a mass of nearly 2.08×10^{18} kg, which is nearly 9 orders of magnitude less massive than the planet Jupiter! This could even represent a super-giant and a brown dwarf like binary star system where the masses are > 5 orders of magnitude (and also the sizes, and different densities - as in logarithmically related case).

5 Case 4: $M_2 = 10M_1$

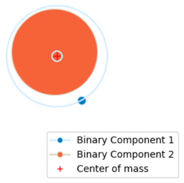

Figure 13: Animation in Cartesian plot with binary component size linearly proportional to their mass

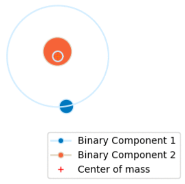

Figure 14: Animation in Cartesian plot with binary component size logarithmically proportional to their mass

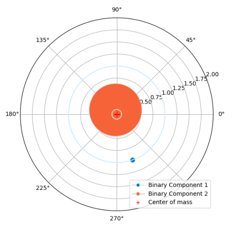

Figure 15: Animation in polar plot with binary component size linearly proportional to their mass

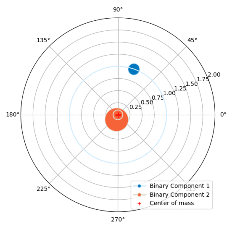

Figure 16: Animation in polar plot with binary component size logarithmically proportional to their mass

The observations and inferences are similar to those as presented in the section where $\frac{M_1}{M_2}$ is 10.

6 Case 5: $M_2 = 10^6 M_1$

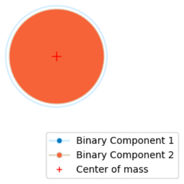

Figure 17: Animation in Cartesian plot with binary component size linearly proportional to their mass

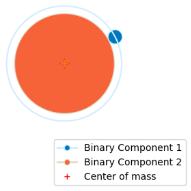

Figure 18: Animation in Cartesian plot with binary component size logarithmically proportional to their mass

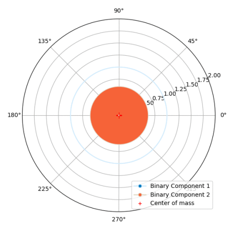

Figure 19: Animation in polar plot with binary component size linearly proportional to their mass

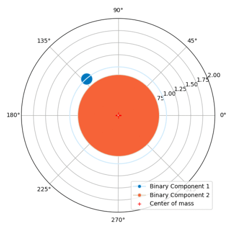

Figure 20: Animation in polar plot with binary component size logarithmically proportional to their mass

The observations and inferences are similar to those as presented in the section where $\frac{M_1}{M_2}$ is 10^6.

7 Implementation

The simulation was performed using Python 3.6.9 with help of libraries - numpy, matplotlib and pylab. The simulation results are displayed in both polar and Cartesian coordinates. Log proportionality was used to compress the difference in magnitude of variation of masses to be visually represented (so that both the components are visible, at least). This was also helpful in simulating systems in which the components have non-identical characteristics (hence, different mass density, physical and chemical properties which influence their sizes.

The code takes following inputs from the user:

Input Parameter	Default value	Description
M_1	1	Mass of first component (in units "x")
M_2	1	Mass of second component (in same units "x")
Key	-1	save as gif(1) or mp4(0), or show(-1)
Proportionality	Linear	Marker size mass proportionality (logarithmic/linear)
Marker Size	15	Base size of the marker
Coordinate system	Cartesian	Type of coordinate system to be used for plotting (polar/cartesian)
Component separation	2.5	Separation between the binary componenets

7.1 Usage

The code can be run from the terminal by providing the input arguments along with the execution command. The below command can be used to get the list of input arguments and a brief description of the same. It also prints the guide to usage on the terminal.

```
$python3 orbit.py -h
```

```
$python3 orbit.py -m1 1 -m2 2 -key 1 -a 3 -ms 10 -plot polar -prop log
```

or example, the above command asks the system to execute the code with mass of the binary components being 1 and 2 units (some common base unit, say $M_\odot$) with marker scale factor of 10, separation between them 3 units (say pc, $\mathring{A}$, etc.,) with the marker size proportional to logarithm of their masses and finally save the simulation (animation) as a ".gif" file.

8 Bibliography

1. https://www.teachastronomy.com/textbook/How-Science-Works/Computer-Simulations/

2. https://github.com/zaman13/Three-Body-Problem-Gravitational-System