What if?

What if the Earth’s solar system was a binary star system? As an undergraduate student, we discussed in astrophysics class the idea that had Jupiter accumulated more mass, it may have become massive enough such that the internal pressure and temperature would reach the threshold needed to start thermonuclear reactions and ignite into star. Of course, had this occurred early on in our solar system’s development, there would be a much different structure and planetary composition than we have today, including the likelihood that the Earth would not exist as we know it, if at all.

An assumption in this project will be that the solar system developed as-is into its current configuration. The first goal will be to develop an eight-planet solar system model with the known planets (Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, and Neptune) and their masses and orbital mechanics, along with the Sun. The Sun will be fixed at the center position of the model. Animating the current configuration will be a part of this and will initially be a 2-D model. This model will use the values defining the planetary attributes and orbital mechanics of the solar system, as defined at: <https://nssdc.gsfc.nasa.gov/planetary/factsheet/>.

Once this is complete, Jupiter will be modified to be a small star by appropriately increasing its mass. From Dr Alastair Gunn at BBC Science Focus (<https://www.sciencefocus.com/space/could-jupiter-become-a-star>):

*“Jupiter is often called a ‘failed star’ because, although it is mostly hydrogen like most normal stars, it is not massive enough to commence thermonuclear reactions in its core and thus become a ‘real star’. But the term ‘failed star’ is a bit of a misnomer. Theoretically, any object at all could be made into a star, simply by adding enough matter to it. With enough mass, the internal pressure and temperature of the object will reach the threshold needed to start thermonuclear reactions. That threshold is the least for the simplest element, hydrogen. In order to turn Jupiter into a star like the Sun, for example, you would have to add about 1,000 times the mass of Jupiter. But, to make a cooler ‘red dwarf’ you would only need to add about 80 Jupiter masses. Although the exact numbers are still a bit uncertain, it is possible that a ‘brown dwarf’ could still form (in which deuterium, rather than hydrogen, fuses in the star’s core) with only about 13 Jupiter masses. So, Jupiter cannot and will not spontaneously become a star, but if a minimum of 13 extra Jupiter-mass objects happen to collide with it, there is a chance it will.”*

Jupiter’s mass will be increased by 13, 80, and 1,000 times its current mass to simulate Brown Dwarf, Red Dwarf, and Yellow Sun-like stars in the place of Jupiter’s current position in the solar system, as discussed above. This will be a drop-in replacement to the current configuration of the solar system. The model will be allowed to run and calculate the new star’s influence, allowing for any calculated changes to the orbits of the planets, as well as the influence on the Sun, and determine what changes may occur. At this time, it has not been decided if the Sun will always remain at the center of the model or if it will be allowed to move along the grid as well. The model will again be updated to remove the mass added to Jupiter from the other three the Jovian planets (Saturn, Uranus, and Neptune). This loss of mass will be varied amongst the three outer gas giants to model potential changes to the configuration of the solar system.

Once a stable model is working for our current solar system, the code will be modified to allow the user to change the configuration via a user interface, including the number of planets and their masses and orbital parameters, as well as the mass of the central star and possibly the number of stars (within reason) for the model that the user desires. This may just be for the initial 2-D model, but a goal is to get a 3-D model working and include this as an add-on. The code, of course, will be modifiable for anyone wanting to create their own solar system.

Another proposed concept that may be added if time permits will be to calculate orbital capture of an interstellar object like 'Oumuamua from 2017-19 (see <https://science.nasa.gov/solar-system/comets/oumuamua/>).

*“Astronomers estimate that an interstellar object similar to 'Oumuamua passes through the inner solar system about once per year, but they are faint and hard to spot and have been missed until now. It is only recently that survey telescopes, such as Pan-STARRS1, are powerful enough to have a chance to discover them.”*

'Oumuamua was about 20 degrees above the plane of the planets orbiting the Sun and was estimated to be moving at about 38.3 kilometers per second. This idea seems far-fetched given that something entering our solar system will likely be moving too fast to be captured by the Sun, but as a start I will try to simulate capture of such an object and try to determine velocity (speed and direction) and mass configurations that may be captured by passing through the Sun’s sphere of influence (Hill sphere). If not already achieved in the model, this will require going to a 3-D model. Initial tests of this model may include adding a known comet to the solar system to show a stable eccentric orbit.

The intent is to primarily utilize Python code and scripts, with an accompanying Jupyter Notebook. The code will be structured to also run independently from a Unix/Linux Kernel, but the Jupyter Notebook will establish structure to go down through the model as new items are implemented and tested. Outputs will include 2-D (and 3-D) animations of a working solar system model, plus graphs and plots to show calculations from planned upgrades to the initial model. Time permitting, some code may be written in C. Further time permitting, Java code may be used to create windows for a user interface. I have taken some Java courses and have always wanted to add in a user interface to my existing C code, so may give it a try here. I do, however, have limited experience with user interfaces, so anything designed may be crude at best.

Classes will likely be used to define the various objects (i.e., Star, Planet, Other, etc.), which will be contained in their own py file that will include importing appropriate libraries. Methods (or functions) will need to be defined to calculate the gravitational influence on each object from the rest of the objects in the model, including movement within given timesteps, and needed methods to update the animation. These methods will be contained in a second py file (possibly third for animation) and will import appropriate libraries and the accompanying Classes file. Methods will include calculating the gravitational forces from Newton’s law of gravitation:

F = GMm/r2

where the Gravitational Constant G = 6.673 × 10−11 Nm2kg−2(kg−1m3s−2). M is the mass of the larger object, m is the mass of the smaller object, and r is the distance between the two objects.

Calculations for gravitational acceleration and trajectories of the objects will be defined in methods. Some methods may be included within the Star, Planet, and Other Classes. Numerical integration methods, like Euler's method or more sophisticated methods like Runge-Kutta and Leap-Frog are likely to be used. Outputs will include the model and graphs associated with some of the planned modifications.

The capture problem will include determining whether an object's trajectory will intersect with the sphere of influence, also known as the Hill sphere, of the larger body, in this case the Sun. The Hill sphere is the region around a celestial body where its gravity dominates over that of other bodies.

rsoi=>a(m/M)

Here M is again the mass of the larger object, m is the mass of the smaller object, and a is the semimajor axis of the smaller object's orbit. If the object enters this region, it may be subject to capture, subject to the object's velocity relative to the larger body being within the range required for capture into a stable orbit. This velocity depends on factors such as the object's distance from the larger body and its direction of motion. Evaluating whether the captured orbit is stable over the long term will need to be determined by analyzing the object's orbital elements, such as eccentricity and semi-major axis, to ensure that it does not intersect with other bodies or suffer from gravitational perturbations. If time allows for the orbital capture problem, graph(s) comparing masses and velocities that allow for capture will be generated.

Caveats:

There are concerns that running graphics simulators in python via matplotlib, turtle, and pygames are currently not working on my laptop. I was hoping to have one of the many solar system models out there up and running by the submission of this proposal but have not been successful yet.