Fuck you Nat! I title this whatever I want: A kinda ok model of the solar system and pluto: A Novel and Other Short Stories

Nat Hawkins, Victor Ramirez, Mike Roosa, Pranjal Tiwari

March 30, 2017

Abstract

In this project, we made a model of the orbits of the planets in our solar system and Pluto. The model was built using the Velocity Verlet algorithms, which uses numerical methods to create a system of four coupled differential equations. In this project we implemented methods of object oriented programming. This allowed us to reduce the code for the Verlet algorithm from around 500 lines of code to roughly 75 lines of code. This significant decrease demonstrated the power of object orientation in reducing our computation load as well as creating a more user-friendly program. The final product was a visualization of our model for the solar system shown in Figure 1 below. Some additional investigation into the three body problem of Earth-Sun-Jupiter and how scaling of masses affects orbits. We also were able to demonstrate the concept of escape velocity with an arbitrary "asteroid".

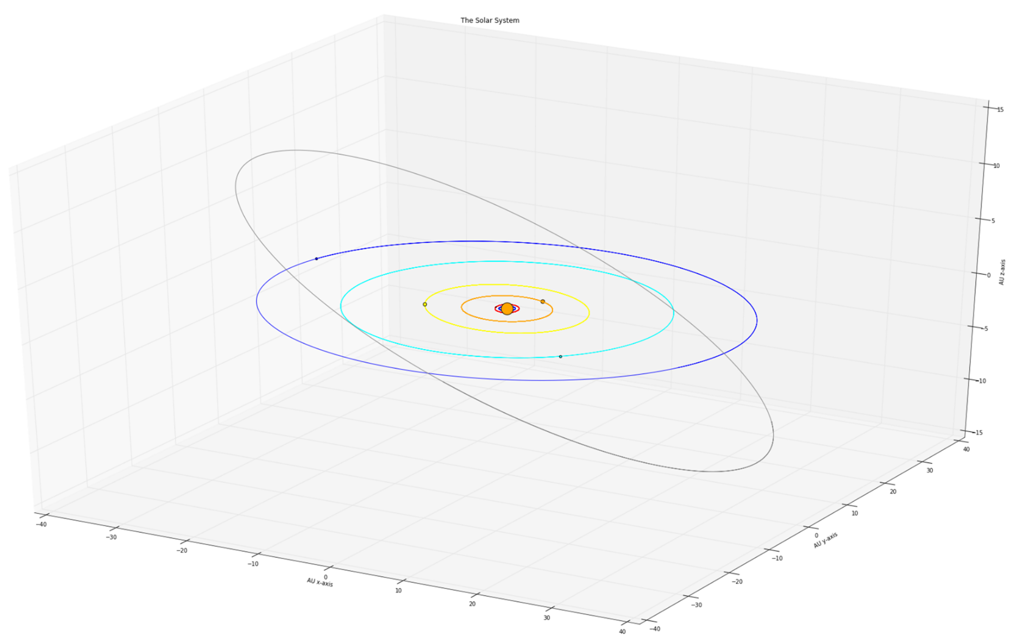

Figure 1: The final output of our solar system model with some fancy coloring and visual effects added. The plot shows the orbital paths of the planets of the solar system over the course of a full solar system orbital period (time required for all plates to complete a full revolution around the sun), which is ≈ 250 years. Visual created by our own Victor "Venv" Ramirez.

1 Introduction

1.1 Motivation

Throughout history, mankind has wondered about the motion of the sun, the planets, and whatever else is out there. Today, we can roughly simulate the Solar System with a computational power of a typical laptop. In this project, we will be modeling our Solar System by calculating the motion of the 9 historical planets and the Sun. We can calculate the motion by solving Newton's Laws of Gravitation using the popular Verlet Algorithm. Using this algorithm, and applying object oriented programming techniques, we created a fairly accurate model of the Solar System that is consistent with what we know about it.

1.2 Object Orientation

Object orientation in this case is very useful, since we have to show the effects of all planets on one another for every time step. This will result in a multitude of equations that all have the same setup and only differ by the variables used. Object orientation can be very useful to make our code easier to read and likely faster. Alright, so here's all the code without object orientation HERE . As you can see, this is a little ridiculous, upon closer inspection you may be able to see that the blocks of code are rather repetitive with small variations. Since this was all in a single function without any loops of any kind, it would be easy to think of a way to optimize this code. Doing so, would make the code much more readable to another user.

2 Solution

2.1 Setup

To begin this problem, we have to define the initial positions and velocities of the planets, which we got from NASA's JPL web-page. This data was then used to define where the planets are with respect to the sun. We used a class to actually define this initial position and simply updated the class with the new position after every time step.

Once the setup of the initial positions were set, we moved on to how the planets would react when in the presence of the sun, which is the main contributor to the force acting on each planet to see if our code worked correctly and can be seen in the Two Planet case in the Results section. With our new found confidence in our ability to make circles, we expanded this to include all the planets and Pluto into our model.

2.2 Method

From a mathematical perspective, this project boils down to solving one equation:

$$\frac{d^2x}{dt^2} = \frac{F_{G,x}}{M_{Planet}} \tag{1}$$

This is Newton's second law solved for the gravitational field of our solar system. The force in this case will need to account for the interaction between an object and every other object we are considering within the system. To evaluate this ODE we need to first rewrite it as two coupled differential equation's.

$$\frac{dx}{dt} = v(x,t) \qquad \text{and} \qquad \frac{dv}{dt} = F(x,t)/m = a(x,t) \tag{2}$$

This allows us to easily evaluate the solution using a simple numerical method such as the velocity Verlet or RK4 method. Both will be discussed but we decided only to build simulations using the Verlet method for reasons which will be discussed later.

2.2.1 Many-Body gravitational force

In order to accurately model the solar system we need to consider the interactions between the planets in addition to the force from the sun. Mathematically this entails adding interaction terms from each planet to the effective force equation. Interactions between Planet 1 and Planet 2 take the one dimensional form:

$$F_x^{1,2} = -\frac{GM_1M_2}{r_{1,2}^3}(x_1 - x_2) \tag{3}$$

When implemented, we rescaled the masses relative to the sun's mass which further simplifies our acceleration equations.

2.2.2 Verlet Solver

Now that we have constructed an acceleration function we solve our equation. As stated earlier we relied on the Verlet method to produce our simulations. In parallel with our discussion in class[1], the Verlet method can be derived by considering th Taylor expansions of both the position and velocity functions:

$$x_{i+1} = x_i + hx_i^{(1)} + \frac{h^2}{2}x_i^{(2)} + O(h^3) \tag{4}$$

$$v_{i+1} = v_i + hv_i^{(1)} + \frac{h^2}{2}v_i^{(2)} + O(h^3) \tag{5}$$

Recall, we have built an expression for the velocity derivative above. Now we can rearrange the velocity function to find an expression of the jerk in terms of the acceleration:

$$hv_i^{(2)} \approx v_{i+1}^{(1)} - v_i^{(1)} \tag{6}$$

2.2.3 Why Rk4 sucks!

We decided not to construct a simulation using the Rk4 method. We can, however, with confidence say that it would not be able to produce stable orbits. This is because it accumulates a global error. With this knowledge, we elected to use the Verlet Solver instead, as it is much more accurate.

3 Results

3.1 Earth-Sun Case

Our first model was the two-body Earth-Sun system. In this system, the acceleration is only affected by the gravity of both bodies. In the figure below, the Earth's orbit follows a nearly perfect elliptical shape around the Sun, while the Sun remains relatively still.

3.2 Earth-Jupiter Case

When looking at Jupiter with an increased mass, the motion of Earth will be perturbed by a very large amount. Having a Jupiter 10 times more massive as its original size would cause Earth's orbit to bend in a cyclical pattern depending on how close the two planets are to one another during the course of their orbits. With a Jupiter mass 1000 times its current mass, The Earth would be ejected out of the Solar System at a velocity above solar escape velocity

3.3 All Planet Interaction Case

The case where we included all planets into the program is very similar to the case with Earth and Jupiter, just with more planets. This is exemplified by figure 1, which has all the planets and includes the interactions they have with one another. We ran this program for 50 years and until that point, the program is observationally stable. Since most planets aren't moving at relativistic velocities, it would be safe to assume that the system would be stable for a period of time much longer than the tested 50 years.

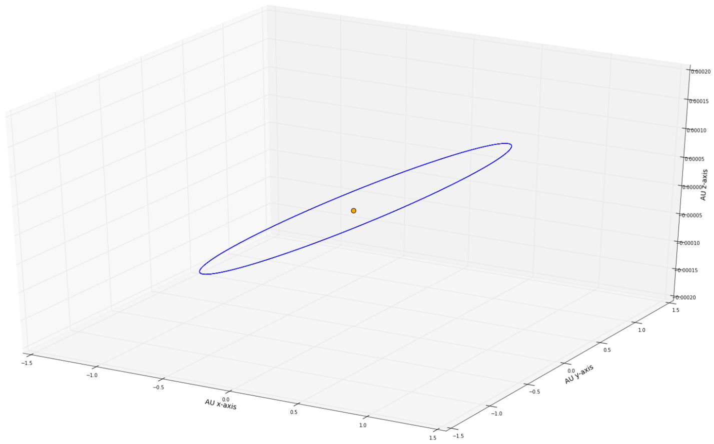

Figure 2: The earth-sun system for a period of 5 years. Fortunately for us, this orbit is stable.

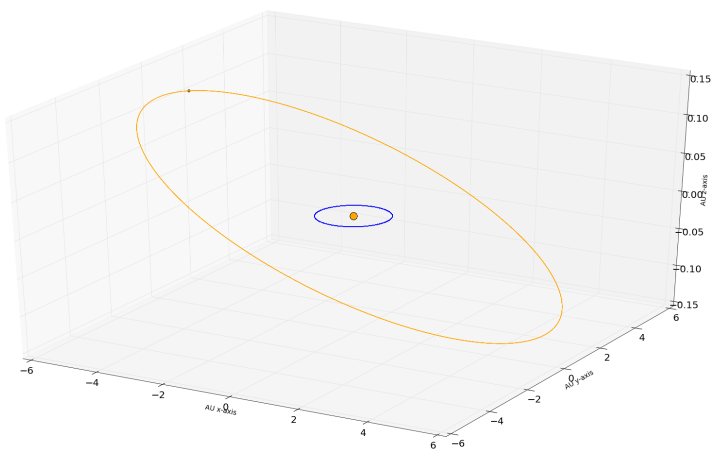

Figure 3: The earth-sun-jupiter system for 50 years.

3.4 Conservation of Energy

The Earth revolves around the Sun in an elliptical orbit, this would mean that the potential and kinetic energies relative to the sun would fluctuate in a cyclical pattern as shown in figure 5. shows that the kinetic and potential energies of the Earth do change over time. But something to note is that the sum of these two remains constant, which shows that energy is conserved for the Earth including all interactions from the other planets and Pluto.

From what we can see, Mercury's orbit is much more elliptical than Earth's from the amplitudes in the oscillations of Mercury's Kinetic Energy as you can see from Figure 6. The total energy seems to be oscillatory as well, if we zoom in on it, as you can see from Figure 7, the total energy of Mercury does oscillate in a cyclic manner, but not in such a way that would correspond to a sine wave, which implies that there is an additional term to the energies that we have not included. We will have to make additional corrections to account for this, but as you can see from the y-axis of this graph, the change in total energy is not very large and doesn't seem to change over a period of time, so unless we let the program run for a very long time, like a million years, the solar system as a whole will stay stable.

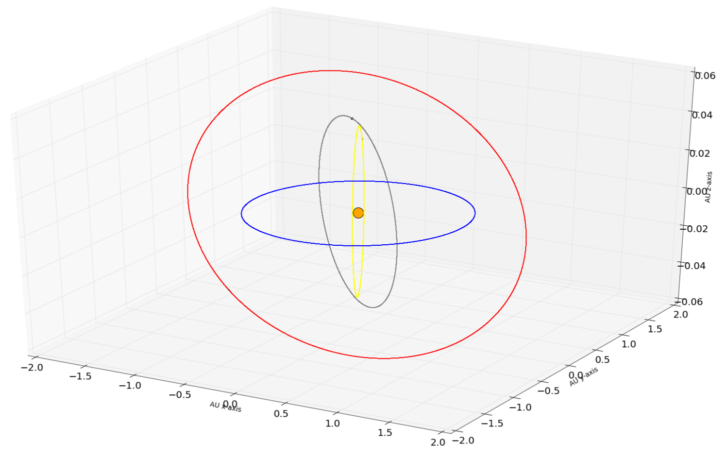

Figure 4: A zoomed in image of the first few planets.

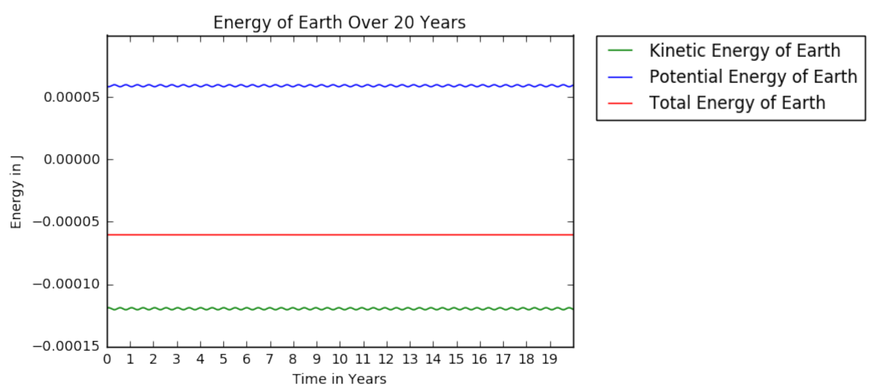

Figure 5: Total Energy Diagram for Earth

3.5 Escape Velocity

We began by programming in an object (we will refer to it as an asteroid for the purpose of this discusssion). The asteroid begins at approximately 1 AU from the Sun. We chose to put it at 1 AU in the x-direction, and 0 AU in both the y and z directions. This was just an arbitrary choice for the sake of the program, but we could have chosen to make any coordinate choice. According to [5], we calculate escape velocity through the following equation:

$$v_{escape} = \sqrt{\frac{2GM}{R}} \tag{7}$$

For the sake of finding the escape velocity, we used a similar approach and scaled $GM_\odot \approx 4\pi^2$. This simplified our calculation of the escape velocity to a numeric result of approximately $v = \sqrt{8\pi^2}$ AU/year. We chose to let this object travel in the x-direction with this velocity to see if it actually will escape with this numerical result.

Figure 8 shows the asteroid escaping the solar system. The scales of the axes got scaled in a non-desirable way when the output was made, but the concept still carries. We can see the asteroid flies past the outer orbit of the solar system. In regards to our numerical result calculated based on what we know about escape velocity, this demonstrates the concept effectively. We can see an object starting at approximately 1 AU from the sun can escape the inner solar system for sufficient velocity. Notice the sheer distance that the object covers over the 250 year period that the program compiled over. This was also demonstrated when we

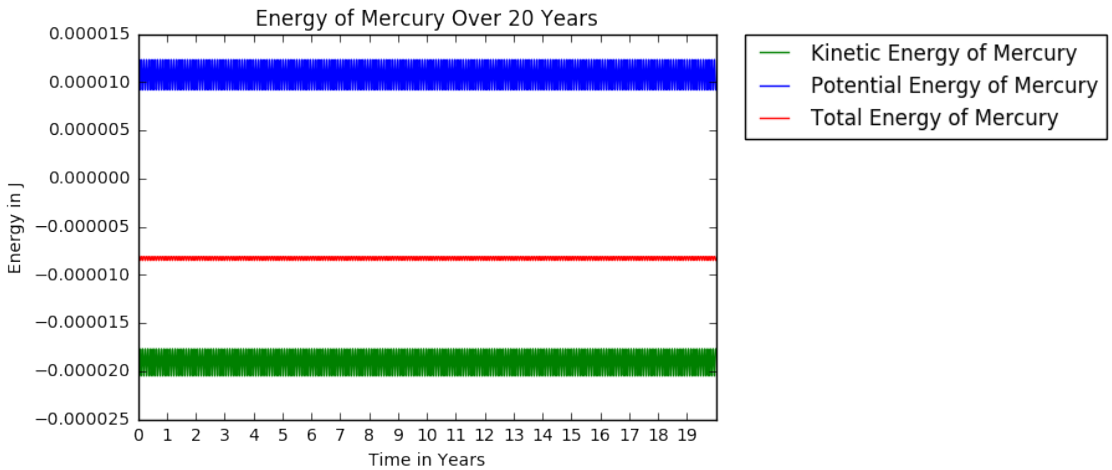

Figure 6: Total Energy Diagram for Mercury

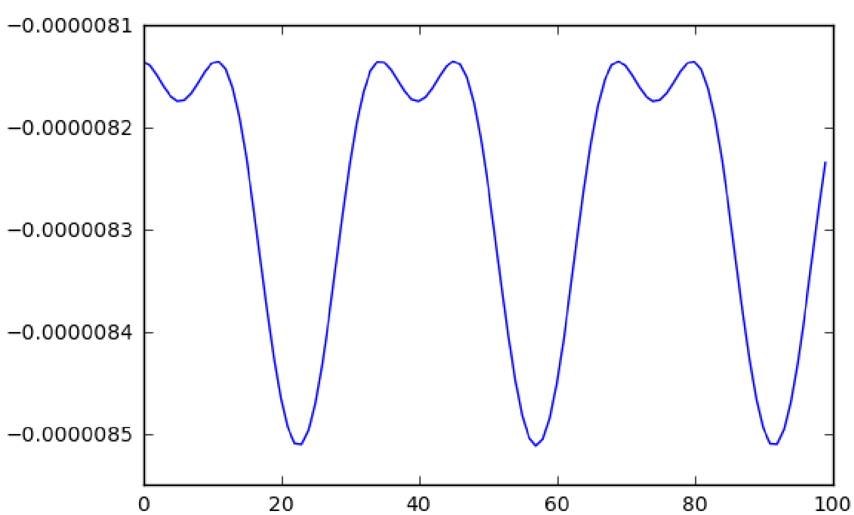

Figure 7: Total Energy of Mercury

scaled the mass of Jupiter in the Earth-Jupiter to system to be 1000 times its original mass. Figure 9 shows that after 100 years with Jupiter's mass scaled closer to being that of the mass of Sun, then we can actually accelerate Earth to escape velocity.

This demonstrates the concept of escape velocity in a very interesting way. Looking at planetary motion in upper limits where planets literally fly off into deep space is an interesting way to look at a concept that we usually associate with when thinking about objects leaving Earth. With this concept in mind, and knowing we can show it in our simulation, we can discuss some commercial aspects of launching objects into space in the future. See "Future Additions" section for discussion on this.

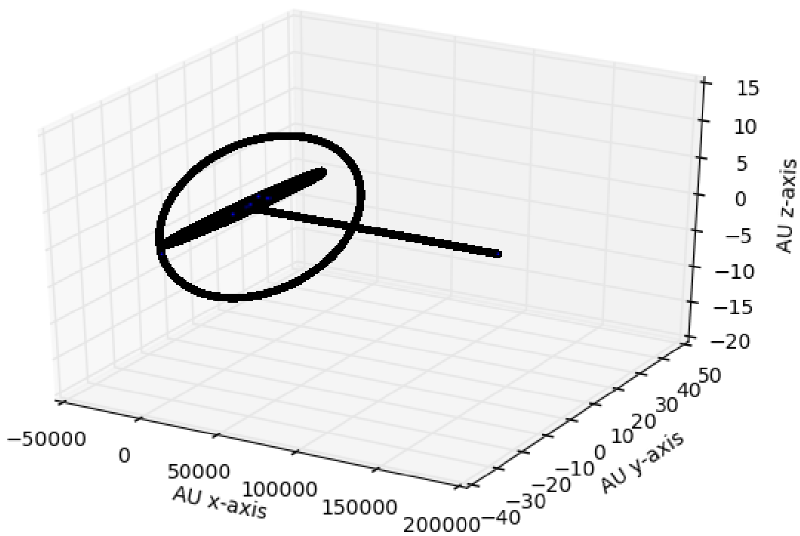

Figure 8: Object achieving escape velocity for the Sun. Beginning 1 AU from the Sun.

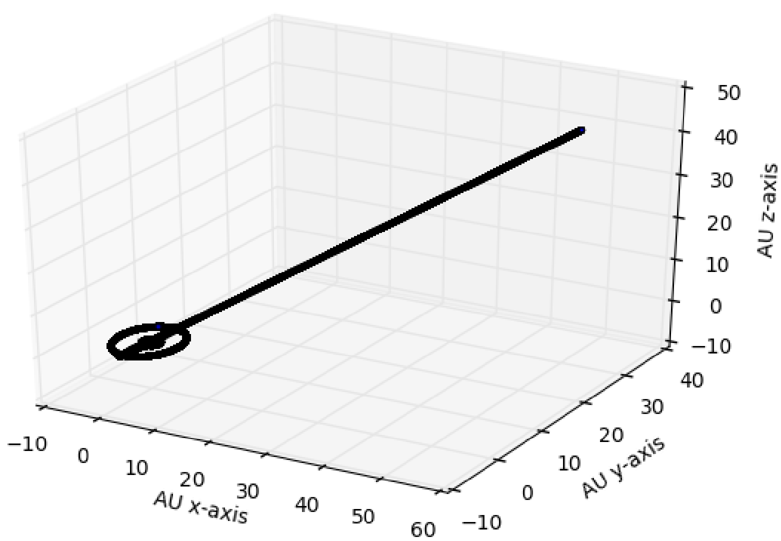

Figure 9: Earth being accelerated to escape velocity after we rescale the mass of Jupiter 1000 times its original mass. Simulation time: 100 years.

4 Future Additions

This model of the solar system is decent, but there are some more inclusions to the program that could be made to give a more accurate representation of the solar system.

For starters, we could include more bodies into the code. The code is set up to be simple to add additional objects to the code. The only changes that would need to be made are gathering the data for whatever body we are trying to include in the model (through JPL and NASA as we did with the other bodies in our code), and run it through our "makeplanet" class. Then, we append it to the array of other planets and have the Verlet solver include that in the algorithmic iterations. The inclusion is simple, but as we saw with the Earth-Jupiter-Sun system, including different masses can have an effect on the orbit of a planet as Jupiter's mass in certain circumstances was able to alter Earth's orbit by a non-zero amount. The inclusions could range from very simple to very computationally dense. We could include Earth's moons as well as the numerous moons of Jupiter and Saturn into our code. More complex additions could include asteroid belts, the rings of Saturn, and even additional solar systems in the neighboring Milky Way galaxy. The possibilities are endless when it comes to what more we could add to this code. As we make it more aligned with the "real world" solar system, we would still hope to see the same level of stability. At least for the sake of our lives that is.

We could continue our analysis by including relativistic effect into our code. We know the Mercury processes due to some relativistic corrections, and including more bodies into this code may include the addition of high velocity bodies or systems, also under the influence of relativistic effects. Relativity may not be the most important aspect to this model, but it would be more aligned with the actual behavior of the solar system.

Some practical things to introduce to the code would be human exploratory efforts. Since we have a working model of the solar system, we could use this to some rough extent to plot the trajectory of an object being launched into deep space, or being put into orbit around a planet, or surveying a nearby galaxy. This model allows us to map motion outside of the realm of human interactions, and from a commercial standpoint, that is a valuable tool when considering launching multi-million dollar satellites or probes into space. This is a computational tool that, given some initial parameters or small adaptations, can be included in our model very easily.

Integration to more complex software would be the next step from a computational standpoint. This was all coded on personal laptop computers with commercially available processor power and memory. Integrating this into some more elaborate software with higher processing power and CPU space could yield faster calculations for larger scale systems. If there were some way to integrate this into a visual output software, we could make a model that updates in real time and visually updates. Our current model outputs one 3 dimensional plot at the end of whatever time frame we tell the program to run over. Being able to actually see the motion unfolding adds another dimension to the interaction between programmer and program, and could be an interesting software development project.

5 Conclusion

The Verlet Algorithm provides a fairly method to simulate the motion of the historical planets. The orbits are stable within the timescale of about 100 years perhaps more if we're able to simulate for much longer. The energies for most of the bodies remain conserved, however Mercury's energy showed some fluctuations due to relativistic effects that we did not take into account. Using object oriented programming techniques significantly streamlined the process of tracking the motion of the planets. Although our model only consists of the 9 historical planets, we have the foundation to that allows us to expand this model by adding more celestial bodies, such as the moons.

References

[1] Discussions and Lecture Slides on Ordinary Differential Equations. Morten Hjorth-Jensen. https://compphysics.github.io/ComputationalPhysicsMSU/doc/pub/ode/html/ode-reveal.html. GitHub.

[2] Hans Petter Langtangen. *Scaling of Differential Equations*. Source: http://hplgit.github.io/scaling-book/doc/pub/book/html/._scaling-book000.html. GitHub. 2016.

[3] Solar System Dynamics. HORIZONS Web-Interface. https://ssd.jpl.nasa.gov/horizons.cgi#top. Jet Propulsion Labratory. NASA.

[4] Wikipedia. *Newton's Law of Universal Gravitation.* https://ssd.jpl.nasa.gov/horizons.cgi#top.

[5] Discussion on Escape Velocity. HyperPhysics. http://hyperphysics.phy-astr.gsu.edu/hbase/vesc.html.

[6] Discussion of Conservation of Energy. https://www.khanacademy.org/science/physics/work-and-energy/work-and-energy-tutorial/a/what-is-conservation-of-energy. Khan Academy.