Project 3

Anders Eriksen

An examination of the accuracy of the veolcity verlet using object orientation on a model solar system.

I. INTRODUCTION

This is an exercise in numerical integration on a system that lends itself well to object orientation. The main motivations are to study the velocity Verlet method in contrast to a simpler Euler solution to integration as well as implement a system of classes to pass objects to and fro where general solutions are applied to the specific objects on a case by case basis. To this extent, we will generate at least three classes. A "planet builder" to create our planets, a "system" to put our planets into and a "solver" to evolve our system over time.

II. THEORY AND METHODS

The main point of focus for the project is developing methods to solve ordinary differential equations (ODEs) through object orientation with the philosophy "write once and run repeatedly". Our object of study this time around is a model solar system, with the assumpion that there are no other objects in the universe and that the planets are all point particles. Finally, we assume that the only force at work is the conservative gravitational force.

$$F_G = \frac{GM_\odot M_{\text{Earth}}}{r^2},$$

We initially also assume that the sun is far too massive to be affected in any meaningful degree by the planets orbiting it. Observations of solar motion inform us that this is not too far off, as the focal point of the suns motion lies far within the solar radius. We will eventually give position and velocity to the sun, but for our initial purposes and to test the soluions, it is satisfactory. The Newtonian equations of motion (eom) give:

$$\frac{d^2\vec{r}}{dt^2} = \frac{\vec{F_G}}{M_{\text{Earth}}},$$

For systems of this size, meters and kilograms are fare too fine as measurement devices go. We utilize astronomical units (AU) and years (yrs).

$$1AU = 1.5 \times 10^{11} \quad 1M_\odot = 2 \times 10^{30}$$

A list of orbital disatance and masses of relevant planets are provided as reference. [1] These masses will be converted into solar masses in our program.

Planet	Mass, kg	Distance to sun, AU
Earth	$M_{\text{Earth}} = 6 \cdot 10^{24}$	1
Jupiter	$M_{\text{Jupiter}} = 1.9 \cdot 10^{27}$	5.20
Mars	$M_{\text{Mars}} = 6.6 \cdot 10^{23}$	1.52
Venus	$M_{\text{Venus}} = 4.9 \cdot 10^{24}$	0.72
Saturn	$M_{\text{Saturn}} = 5.5 \cdot 10^{26}$	9.54
Mercury	$M_{\text{Mercury}} = 3.3 \cdot 10^{23}$	0.39
Uranus	$M_{\text{Uranus}} = 8.8 \cdot 10^{25}$	19.19
Neptun	$M_{\text{Neptun}} = 1.03 \cdot 10^{26}$	30.06

The first point of order is to examine and contrast the integration methods. At this point, we merely want to build a working system for for the 2-body problem of the Earth and Sun, which we can further generalize and object orient. Before all this, however, we need to discretize our equations.

If we assume ciruclar motion, then we can rewrite the force as

$$F_G = \frac{M_{\text{Earth}}v^2}{r} = \frac{GM_\odot M_{\text{Earth}}}{r^2},$$

using sentripetal acceleration $\frac{v^2}{r}$ and use this to get

$$v^2 r = GM_\odot = 4\pi^2 \text{AU}^3/\text{yr}^2.$$

Giving 4π the dimensions AU^3/yr^2 and use this to eliminate the gravitational constant from our equations.

$$F_G = \frac{4\pi^2 M_{\text{Earth}}}{r^2}$$

With the force sorted, I can look into the Forward Euler and velocty Verlet integration algorithms. Both are based on Taylor expansions around a point.

$$x(t \pm h) = x(t) + h\dot{x} + \frac{h^2}{2!}\ddot{x} \pm O(h^3)$$

For Euler, we use the base equation and find

$$v+ = a \cdot h \tag{1}$$
$$x+ = v \cdot h \tag{2}$$

while the Verlet, with the velocities added, through a taylor expansion of both the velocity and acceleration about the point, as well as the sum $x(t+h) + x(t-h)$

$$x_{i+1}+ = v_i \cdot h + 0.5 \cdot h^2 \cdot a(x_i) \tag{3}$$
$$v_{i+1}+ = 0.5 \cdot h \cdot (a(x_{i+1}) + a(x_i)) \tag{4}$$

implementing these, we can set up algorithms as following.

N = nr. of timesteps
$h = (t_{end} - t_{start})/N$ // steplength
for $i = 0$ **to** $i = N - 1$ **do**

$$\vec{v_{i+1}} += \vec{a_i} \cdot h$$
$$\vec{r_{i+1}} += vi \vec{+} 1 \cdot h$$

end

Algorithm 1: a Forward Euler integration

N = nr. of timesteps
$h = (t_{end} - t_{start})/N$ // steplength
for $i = 0$ **to** $i = N - 1$ **do**

$$\vec{a_i} = \vec{a}(\vec{r_i} i)$$
$$\vec{r_i} \vec{+} 1 += h \cdot \vec{v_i} h^2 a_i/2$$
$$\vec{a_{i+1}} = \vec{a}(\vec{r_{i+1}})$$
$$\vec{v_{i+1}} += h * (a_i + a_{i+1})/2$$

end

Algorithm 2: a velocity Verlet integration

After ensuring that the algortihms are implemented correctly, we generalize the code into an object oriented (OO) implementation. This should constitute a main file to coordinate all the dispersed parts, a "creator" to create the planets, a "system" to hold the planets and to extract interactions between them and finally a "solver" to evolve our system over time. To get a bit og a head start we begin with a main outline from examples found on the course github repository[2].

One way to controll the acuracy of integration over various timesteps, is to let the planets move along paths which we can predict through proper restrictions, such as circular orbits. Next, we can explore the situation of escape velocity, as this is another analytically solvable problem, we can ensure that we get are on the right track, if the planet does escape at the velocity we find and not before.

After exploring our methods, we complicate the system furter through adding another body to our problem. The greatest single contributor besides the sun, Jupiter. The 3-body problem is far more difficult to process than the 2-body problem. with 2 bodies, the earthen orbit is stable and unchanging. Adding in Jupiter, we need to take into account how much of an effect Jupiter has on the earth. Thus, we need the total force as the vector sum of the forces from the sun and Jupiter. Now we have pull from different directions over time, and this should affect the orbit of the planet. The force contribution of Jupiter is on the form

$$F_{\text{Earth-Jupiter}} = \frac{4\pi^2 M_{\text{Jupiter}} M_{\text{Earth}}}{r^2_{\text{Earth-Jupiter}}},$$

Where the masses are given in relation to the sun. To ensure that the solution is stable, we record the momentum and check the drift over time.

The next step in our system model, is to move the sun out of the center and into it's proper place before adding the remainder of the planets. To do this, we use a list of positions and velocities provided by NASA initially, but which I received from Mathias Vege[3].

As we have added most of the desired complexity to our system, we want to end with taking a closer look at the forces we have been working with throughout the project. To do this, we look at a problem which was important to support Einsteins theory of relativity, the perihelion precession of Mercury, which is about $43''$ larger per century than what the Newtonian system predicts. This arises from a small perturbation of the $1/r^2$ dependency of the gravitationsl force which results in the effective precession of the elipse. The correction to the force becomes

$$F_G = \frac{GM_{\text{Sun}} M_{\text{Mercury}}}{r^2} \left[1 + \frac{3l^2}{r^2 c^2}\right]$$

Where $l = |\vec{r} \times \vec{v}|$ is the magnitude of Mercury's orbital angular momentum per unit mass, with c as the speed of light in vacuum. We track the angle of perihelion θ_P with

$$\tan\theta_{\text{p}} = \frac{y_{\text{p}}}{x_{\text{p}}}$$

or the ratio of Mercury's x and y position at perihelion. Using the perihelion speed of Mercury as $12.44\,\text{AU/yr}$ with a distance to the sun of $0.3075\,\text{AU}$. As a controll of the resolution, we can compare with the precession of the perihelion without the correction.

III. RESULTS AND DISCUSION

For the initial incarnation of the solution, just a simple run of either Verlet or Euler with the Earth-Sun system, we can see a change in the distance for the Euler solution, but not for the Verlet solver. This indicates that the energy really is conserved in the integration. For the 2 body problem, the question of a circular orbit is fairly simple, merely following the sentripetal acceleration $a = \frac{v^2}{r}$ which gives us a velocity of $v = \frac{2\pi}{r}$

The Forward Eulre integration has a clear drift over time as seen in figure 1, stemming from a failiure to conserve the energy of the system. The velocity verlet, however maintains the energy, see figure 2, as well as having a numerical error proportional to steplength h^2, as opposed to Euler's h. We can also see that the resulting orbit is circular, as we should expect from the initial velocity of the simple system.

When calculating escape velociy, I used a distance $r = 1AU$ and used the fact that the energy would need to be either 0 or greater if the planet was to escape. Therefore, I sat $E_k = E_p$ and found a velocity of $v = \sqrt{\frac{8\pi^2}{r}}$, the initial distance meant $v = 2\sqrt{2}\pi$. This is illustrated in figure 3, where we see that the motion is less and less affected by the star with distance. As this is the exact escape velocity, it will not disappear until the distance

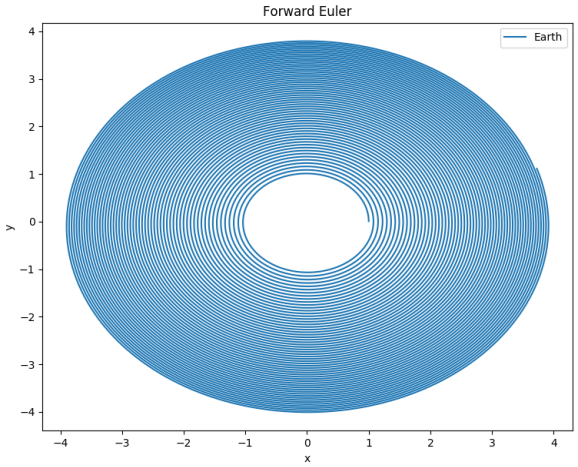

FIG. 1. Forward Euler integration of the 2 body system of a stationary sun and earth. There is a clear drift as time goes by, which empathizes the drif in energy of the system with the Euler, since the integrations are done in a conservative potential field.

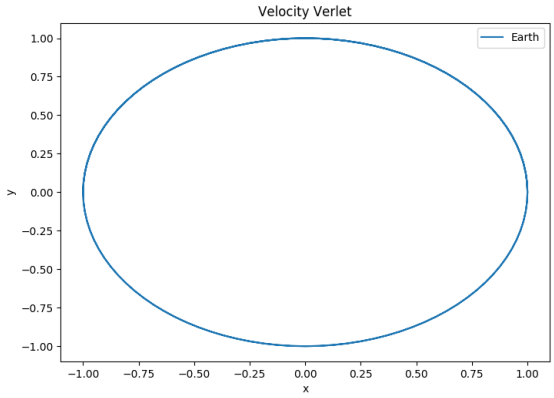

FIG. 2. Velocity Verlet integration of the 2 body system of a stationary sun and earth. This integration is run with as many steps as figure 1, which shows how much more robust the Verlet solution is compared to the Forward Euler. We can also see from the axes that despite the seeming eliptical orbit, this has more to do with improper scaling. The radius is 1 AU in all directions.

is at infinity. Currently, the velocity goes according to $v \propto \frac{1}{\sqrt{r}}$, but if we were to increase the density dropoff of the gravitational force according to the cube of the distance, then the escape velocity would approach $v \propto \frac{1}{r}$.

At this point, we begin to leave the realm of analytical solutions explored by Newton and his contemporaries. Here, we see that the inclusion of Jupiter squishes the circular orbit of the Earth into an elipse. This is also a challenge to pick good values for initial conditions. Therefore,

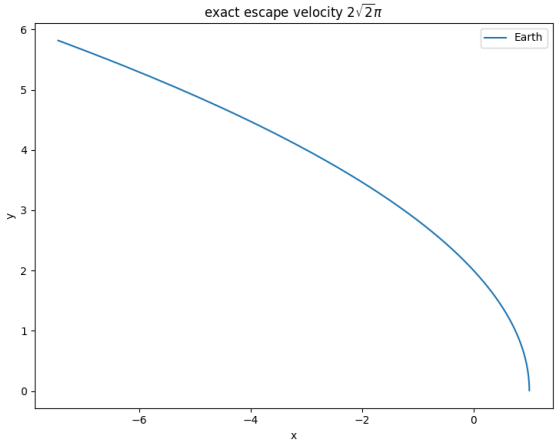

FIG. 3. this is the earths "orbit" around the sun with an escape velocity so $E_k = E_p \rightarrow E = 0$. For a system at a distance of $1AU$, escape velocity is, as stated in the title. Measurements show that just below leads to a stable orbit with a change from $2 \rightarrow 1.9$ in front of the square root.

I use information on planet positions and velocities[3] as mentioned above. Varying Jupiter's mass, perturbs

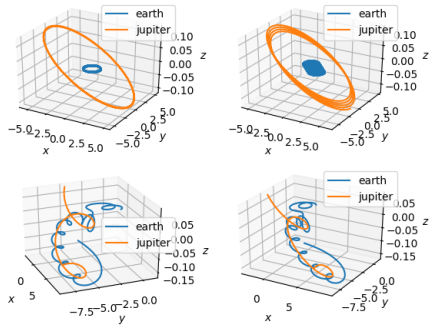

FIG. 4. The orbits of Earth and Jupiter around a stationary sun for varying masses of jupiter. We can see as we increase the mass from Jupoter's regular mass(top left) by 10(top right) and 1000(botom row) respectively, we get increasingly rampant orbits, with the lower spiraling out completely. We can see that as Jupiter's mass approaches that of the sun, it destabilizes the system.

Earth's orbit untill Jupiter is as massive as the sun. At this point, the system breaks down and spirals outwards. What we can notice throughout these, is that the momentum of the system remains the same to a relative error $\epsilon < 1e-5$, which means that the Verlet method still holds even under these strange developements.

As we have now more or less verified our numericals for various systems and number of planets, we can begin to add in both the motion of the sun as well as the remaining planets(excluding Pluto which is currently not within

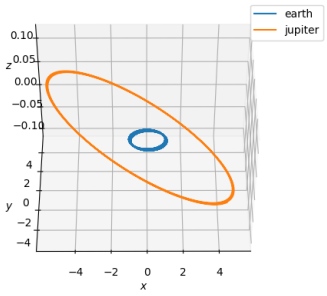

FIG. 5. An enlarged version of the first plot in figure 4. We can see, that the previously circular orbit of the earth is pushed to be eliptical. We also see some slight axial tilt in the axis. the Verlet method retains angular momentum to a great deal, and this should be close to the effect of the planet, rather than iancuracies in the numerical method.

this deifnition) in figures 6 and 7. We can observe a gap between the orbits that seem to almost increase exponentially with the distance form the sun. We can also see that the 5 decades of time developement does not seem to be enough to complete the orbits of the furthest planets. This makes sense, as the gravitational field reduces with $\frac{1}{r}$, and so the resulting velocities that remain bound to the system have to be significantly lower, as the kinetic energy is proportional to the square of the velocities. we can also see, that the initial assumption of coplanar motion is off, but not by terribly much over all, with some tilt in either direction.

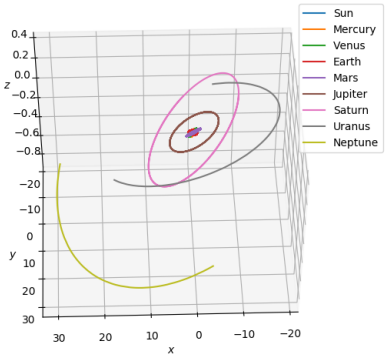

FIG. 6. The full solar system (excluding Pluto). The data is gathered over 5 decades, which is not quite enough time for the outermost planets to complete an orbit, but we see that Saturn and closer planets have seemingly closed orbits. There is a distinct distance between the outer planets Neptune, Uranus, Saturn and, to a degree, Jupiter and the inner planets beginning with mars. This inner part has far tighter orbits in comparison.

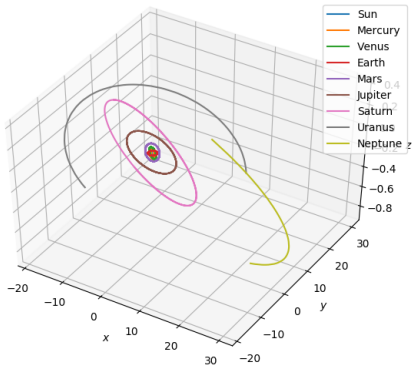

FIG. 7. A shifted angle of the same plot as figure 6, in an attempt to better showcase the inner "nucleus". We can see a progressively closer orbits as we approach the sun, almost seeming to follow an inverse proportionality to the distance from the sun.

A final consideration, is the accuracy of our potential calculations. We assumed the system followed the newtonian gravity, which seems to be fairly accurate on the whole. But there are corrections that can be made with regards to the theories of relativity. In particular, an important test of Einsteins theories was the perihelion precession of Mercury. The perihelion precession was measured to be $43''$ per century. Newtonian gravity predicts several orders of magnitude less in this timeframe. To examine this, I ran a system of pure Sun-Mercury and set Mercury's initial values as those in the perihelion. Letting the simulation run for a century with a sampling of $1e8 - 1e9$ points, I found a precession of $42.8''$ for the relativistically corrected equation presented in the theory above. While the newtonian system predicts a precession of $0.02''$. this shows that the relativistic model is about 1000 more accurate at predicting systems of this magnitude than newton's.

IV. CONCLUSION

As we built up our program and system, we saw that the velocity Verlet method vastly outperformed the Forward Euler, maintaining a good approximation to the planet's path with around a few hundred integration points a day during our revolutions around the star and maintaining a near constant angular momentum through the years in our isolated system.

As we added complexity to our system, we also developed a program with objects that could compartmentalize the required tasks and feed the finished results into eachother, resulting in a program that could near seemlessly add planets and calculate their positions

and velocities over time without more than an added line. This object orientation mainly revolced around 4 objects. A "celestial body" class, which took in the planets and assigned them positions, velocities and other physical observables which could then be accessed down the line. A "system" class which stored all the planets and could work out the forces inbetween each of them and a "Solver" class, which could solve the equations of motion for the planets in the system a step forwards in time. This way of handling things allowed for far less repetition of definitions and helped insulate mistakes.

While the systems we set up worked well, there are a few things that I did not have the time to fully automate, such as the force used, or the calls to various amounts of planets. Ideally, I envision a system in which I can give a filename and the program will implement a planet with name, position, vellocity and other attributes to create the system I wish to study. further, I would like to customize the span of the integration in both length and timestep or number of steps. The main issue I encountered on this front, was in innterpreting the input. I believe a sollution here could be a textfile as input, with relevant info such as data files, integration length and similar, while the system itself has perhaps another object that can interpret input from file and put the results where they ought to go. Finally, I should have specified a folder to write the results into.
Another solution could be a python script to handle inputs and outputs, with plots as necessary. This would be a more involved project than what time allows for this time around.

V. APPENDIX

[1] M. Hjorth-Jensen, http://compphysics.github.io/ComputationalPhysics/doc/Projects/2018/Project3/html/Project3.html (2018).
[2]
[3] M. Vege, Slack (2018).