Building a numerical model for the solar system using ordinary differential equations

Anna Eliassen, Aron Jansson Nordberg and Kristina Othelia Lunde Olsen
Department of Physics, University of Oslo, Norway
{anna.eliassen, aronjn, koolsen}@fys.uio.no

October 28, 2020

Abstract

We have investigated two numerical algorithms, the Forward Euler algorithm and the Velocity Verlet algorithm, for solving ordinary differential equations. These methods have been investigated by simulating our solar system, from a simple Earth-Sun system to a n-body problem. Already for the Earth-Sun system, we could conclude that the Velocity Verlet algorithm outperforms the Forward Euler algorithm, as only the Velocity Verlet method conserves energy and angular momentum. The Forward Euler method has shorter CPU time, but needs too many integration points, $n = 10^6$, before producing good results. The Velocity Verlet method produces good results with $n = 10^4$. We also show that our Velocity Verlet algorithm manages to produce good results when looking at a 3-body problem, as well as for the solar system with 9 planets and the Sun orbiting a center of mass. In addition, we investigated the perihelion precision of Mercury by including relativistic correction. By increasing the number of integration points to $n = 4.5 \cdot 10^5$, we managed to obtain a more stable orbit. We calculated the perihelion precision to be 42.87″ (42.87 arc seconds), which corresponds well to the 42.98″ predicted by general relativity [8].

1 Introduction

The solar system consists of many celestial bodies bound together by gravity. It consists of a central star, the Sun, and in orbit around it are nine planets, when we include Pluto for historical reasons. There are various other bodies, such as dwarf planets, asteroids and moons orbiting the various planets. The bodies move according to quite simple laws of physics that can be expressed as differential equations. The number of such equations that can be solved analytically is quite small, but if we make use to numerical methods we can solve virtually any set of such equations for nearly any initial conditions. When considering n bodies moving under mutual attractive gravitational force, it is called an n-body problem. The 2-body problem can be solved analytically, but already at 3 bodies there are only a very limited amount of stable solutions. Anything beyond this requires the use of numerical methods to model.

We have undertaken the challenge with the use of two popular algorithms for solving differential equations, the Forward Euler algorithm and the Velocity Verlet algorithm. We will first look at a simple model, only including the Earth and the Sun. This model will gradually be expanded until we have a solar system with the Sun and all the planets. We will also study the perihelion precision of Mercury, since the Newtonian law of gravity cannot account for the observed movement of Mercury. We will implement an adjusted model for gravity using the results from General Relativity.

For this project we used the programming language `Python`. The equations to be solved are quite similar, only minor changes in the input parameters, we have focused on developing an object-oriented code. Where we made the use of classes when modelling the solar system and solving the differential equations. When simulating the solar system, we will use real HORIZONS data from NASA for all celestial bodies, in the form of mass, initial position and initial velocity for a given time. This will make sure that our simulations start with conditions that are realistic and found by observations.

In this report we will first present the necessary theory needed to solve the problems, before an quick section about the methods used. Here we will explain the two algorithms, as well as the different models we are simulating. At the end we present and discuss our results, before concluding our findings. All figures and Python code used for this project can be found on our GitHub page [2].

2 Theory

2.1 Newtonian Mechanics

Newton's law of universal gravitation tells us that the gravitational interaction between two objects, with masses m_1 and m_2, can be described as a force acting on m_1:

$$\mathbf{F}_G = -\frac{Gm_1 m_2}{r^2}\hat{\mathbf{r}}, \tag{1}$$

where G is the gravitational constant and r is the distance between the two objects. $\hat{\mathbf{r}}$ is a unit vector pointing from m_1 to m_2 [4].

2.1.1 Earth-Sun system

For the Earth-Sun system the force of gravity on the Earth from the Sun, based on equation (1), becomes

$$\mathbf{F}_G = -\frac{GM_\odot M_{\text{Earth}}}{r^2}\hat{\mathbf{r}}, \tag{2}$$

where $r \approx 1$ AU. The Sun is motionless in the origin, and we only consider the motion of the Earth under the influence from the Sun's gravitational pull. Newton's second law of motion is

$$\mathbf{F}_{ext} = m\mathbf{a}, \tag{3}$$

where $\mathbf{F}_{ext}$ is the sum of external forces acting on an object, m is its mass and $\mathbf{a}$ its acceleration. Since the only external force is gravity, we can combine (2) with (3) and get a second-order differential equations that govern the Earth's motion in 2D space under the influence of the Sun's gravity [5].

$$\begin{aligned}
\mathbf{F}_{ext} &= \mathbf{F}_G \\
M_{\text{Earth}}\mathbf{a} &= -\frac{GM_\odot M_{\text{Earth}}}{r^2}\hat{\mathbf{r}} \\
\mathbf{a} &= -\frac{GM_\odot}{r^2}\hat{\mathbf{r}} \\
\frac{d^2\mathbf{r}}{dt^2} &= -\frac{GM_\odot}{r^2}\hat{\mathbf{r}}
\end{aligned}$$

Since we only consider 2D motion, we can decompose this vector equation into two scalar equation, one for each coordinate x and y.

$$\begin{aligned}
\frac{d^2x}{dt^2} &= \frac{F_{G,x}}{M_{\text{Earth}}} \\
\frac{d^2y}{dt^2} &= \frac{F_{G,y}}{M_{\text{Earth}}}
\end{aligned}$$

$F_{G,x}$ and $F_{G,y}$ are the gravitational forces in the x and y directions respectively [4].

A second-order differential equation can be reduced to two first-order coupled differential equations. We introduce the velocity v, such that $a = dv/dt$. The equations above then reduce to

$$
\begin{aligned}
\frac{dx}{dt} = v_x \longrightarrow \frac{dv_x}{dt} = \frac{F_{G,x}}{M_{\text{Earth}}} = a_x(x,y) \\
\frac{dy}{dt} = v_y \longrightarrow \frac{dv_y}{dt} = \frac{F_{G,y}}{M_{\text{Earth}}} = a_y(x,y)
\end{aligned}
$$

We can simplify our calculations by using properties that are characteristics of the Sun-Earth system. We will also assume a near-circular orbit. First, if we use Astronomical Units (AU) for length and AU/year for velocity, we can use Kepler's third law of planetary motion to get a compact value for the gravitational constant. Kepler's third law is

$$
P^2 = \frac{4\pi^2}{G(m_1 + m_2)} a^3, \tag{4}
$$

where P is the period of the orbit, which for the Earth is 1 year. a is the semi-major axis of an elliptical orbit, which in the case of a circular orbit is just the radial distance from the Sun $a = r =$1 AU. m_1 and m_2 are the masses of the Sun and Earth, respectively. We can neglect Earth's mass since it is very small compared to the Sun's. Equation (4) can the be used to express the following

$$
\begin{aligned}
P^2 &= \frac{4\pi^2}{GM_\odot} a^3, \\
GM_\odot &= 4\pi^2 \frac{a^3}{P^2} = 4\pi^2 \frac{\text{AU}^3}{\text{yr}^2}.
\end{aligned}
$$

Second, we can use the expression for centripetal acceleration to derive an analytical expression for the velocity, which can be used for input in our simulations. We will only consider the scalar magnitude of the accelerations and forces here, since they all happen along the radial axis. The expression of the centripetal force is [3] (pp. 41).

$$
F_c = \frac{mv^2}{r}. \tag{5}
$$

We thus get the following relationship with the Earth-Sun system.

$$
\frac{GM_\odot}{r^2} = \frac{v^2}{r} \longrightarrow v = \sqrt{\frac{GM_\odot}{r}}
$$

As we have shown, $GM_\odot = 4\pi^2\text{AU}^3/\text{yr}^2$, and $r = 1\text{AU}$. so the speed of the Earth in the orbit at any time is

$$
v_0 = 2\pi \text{ AU/yr} \tag{6}
$$

2.1.2 Earth-Jupiter-Sun system. Stationary Sun

If we consider the same system as before, but add the planet of Jupiter with mass $M_{Jupiter}$, we need to modify the equation that govern Earth's motion. The acceleration due to the gravitational pull of Jupiter needs to be added.

$$\mathbf{a}_{\text{Earth}} = -\frac{GM_{\odot}}{r^2_{\text{Earth}-\text{Sun}}}\hat{\mathbf{r}}^2_{\text{Earth}-\text{Sun}} - \frac{GM_{\text{Jupiter}}}{r^2_{\text{Earth}-\text{Jupiter}}}\hat{\mathbf{r}}^2_{\text{Earth}-\text{Jupiter}}$$

Similarly, the equation that governs Jupiter's motion will have the following acceleration [6]

$$\mathbf{a}_{\text{Jupiter}} = -\frac{GM_{\odot}}{r^2_{\text{Jupiter}-\text{Sun}}}\hat{\mathbf{r}}^2_{\text{Jupiter}-\text{Jupiter}} - \frac{GM_{\text{Earth}}}{r^2_{\text{Jupiter}-\text{Earth}}}\hat{\mathbf{r}}^2_{\text{Jupiter}-\text{Earth}}$$

2.1.3 n-body problem. Non-stationary Sun

When considering the motion of the entire solar system (the Sun and all 8 planets + Pluto), we must take into consideration the acceleration from gravity from all the planets on all the planets. The acceleration of planet i becomes

$$\mathbf{a}_i = -G\sum_{i\neq j}^{n}\frac{M_j}{r^2_{ij}}\hat{\mathbf{r}}_{ij} \tag{7}$$

Note that the n bodies include the Sun. When considering an n body system, it is useful to consider motions relative to the centre of mass (CM). If there are no external forces, then the CM will not accelerate. That way, it is easier to interpret results of the simulations of the motions. The CM is the mean of the position of all the bodies, weighted by their masses.

$$\mathbf{R} = \frac{1}{M}\sum_{i}^{n} m_i\mathbf{r}_i \tag{8}$$

where $\mathbf{r}_i$ is the position of planet i relative to CM, and M is the sum of all the masses. Similar for the velocity of the CM

$$\mathbf{V} = \frac{1}{M}\sum_{i}^{n} m_i\mathbf{v}_i \tag{9}$$

2.2 Conserved quantities in the system

2.2.1 Conservation of angular momentum

Kepler's second law of planetary motions states that "A line joining a planet and the Sun sweeps out equal areas during equal intervals of time". Another way to state this is by

$$\frac{dA}{dt} = constant,$$

where dA is the infinitesimal area swept out after time dt. We will in the following use the approach from the book "An Introduction to Modern Astrophysics", chapter 2.3 (pp. 53-56) [3].

After sweeping out an angle $d\theta$, we have a triangle with lengths r and ds. We have that $dS = d\theta r$, thus the area is $dA = \frac{1}{2}rdS = r^2 d\theta$. Dividing both sides with dt we get

$$\frac{dA}{dt} = \frac{1}{2}r^2\frac{d\theta}{dt} \tag{10}$$

The magnitude of angular momentum per mass in polar coordinates is

$$L = |\mathbf{L}| = |\mathbf{r} \times \mathbf{v}| = r^2\frac{d\theta}{dt}$$

We now insert (10) and get the final expression for the magnitude of angular momentum per mass

$$L = 2\frac{dA}{dt}$$

Given Kepler's second law, this expression is constant in time.

2.2.2 Conservation of energy

The total mechanical energy, E_{tot}, in the system is conserved since there are no external forces acting on the system.

$$E_{tot} = E_k + E_p = \frac{1}{2}m_1 v^2 - \frac{Gm_1 m_2}{r} \tag{11}$$

where E_k is the kinetic energy and E_p is the potential energy [1].

2.3 Escape velocity

The escape velocity is the needed/initial velocity a planet must have to escape from the gravitational pull from the Sun. This occurs when the kinetic energy E_k becomes larger than the potential energy E_p. By using energy conservation, we can find an equation for the escape velocity.

$$\begin{aligned} E_k &> E_p \\ \frac{1}{2}M_{Earth}v_{esc}^2 &> \frac{GM_\odot M_{Earth}}{r} \\ v_{esc} &> \sqrt{\frac{2GM_\odot}{r}} \end{aligned} \tag{12}$$

For the Earth-Sun system, we have $GM_\odot = 4\pi^2\text{AU}^3/\text{yr}^2$ and $r = 1$ AU, giving us $v_{esc} = 2\sqrt{2}\pi = \sqrt{2}v_0$ AU/yr.

2.4 The perihelion precession of Mercury

Mercury is the planet closest to the Sun, and has a relatively high eccentricity, making the orbit quite elliptical compared to the other planets. The point in its orbit that is closest to the Sun is 0.3075 AU, and is called the perihelion.

It was discovered in the late 19th century that the perihelion point changes over time. Even when subtracting other sources that could influence the position of the perihelion, like the presence of the other planets, it was found that the precision of Mercury's perihelion was about 42.98″ ($\sim$ 43 arc seconds) per century [8].

Einstein's theory of general relativity (GR) is the fundamental theory of spacetime and is a more general description of gravity than Newton's laws. One can derive Newton's law of gravity (1) as a special case of the more general solution. Newtonian gravity is $F_G \propto 1/r^2$, but if we add another term from the general solution we get a term that is $F_G \propto 1/r^4$. This term is negligible for most planets around the Sun, but since Mercury is so close, this will have observable effects. Adding this relativistic correction, we get the following modified law of gravity

$$F_G = \frac{GM_\odot M_{\text{Mercury}}}{r^2}\left[1 + \frac{3l^2}{r^2c^2}\right], \tag{13}$$

where M_{Mercury} is the mass of Mercury, r is the distance between Mercury and the Sun, c is the speed of light in vacuum and $l = |\mathbf{r} \times \mathbf{v}|$ is the magnitude of Mercury's orbital angular momentum per unit mass. This correction means that a closed elliptical orbit is no longer possible, as that was a result of the $1/r^2$ dependence of Newton's law. This means that the planet will not end up in the exact same place after each motion around the Sun, so the perihelion will move over time.

The perihelion angle θ_{p} is defined as

$$\tan(\theta_{\text{p}}) = \frac{y_{\text{p}}}{x_{\text{p}}} \tag{14}$$

where x_{p} and y_{p} are the x and y position of Mercury at perihelion.

3 Method

3.1 Algorithms

We will implement two algorithms in our solver to solve the set of coupled differential equations numerically. In general, the coupled equations we want to solve for the motion of the solar system is on the form

$$\begin{aligned}\frac{dv}{dt} &= a(t,r)\\ \frac{dr}{dt} &= v(t,r)\end{aligned}$$

In order to solve them numerically, we must discretize the continuous equations, and solve them for a finite number of grid points. Let $t_0 = 0$ be the initial time point for our simulation, and T will be the end point. We then choose N grid points $t_i = t_0 + hi$, were $h = \frac{T-t_0}{N}$ is the step length in time. Given inital conditions $r_0 = r(t_0)$ and $v_0 = v(t_0)$ we can use approximation schemes to compute values for $r(t)$ and $v(t)$ for every time step.

3.1.1 Forward Euler algorithm

The Forward Euler method is based on the straight-forward definition of the derivative.

$$f'(x) = \lim_{x\to\infty} \frac{f(x+h) - f(x)}{h}$$

A finite version of this is

$$\begin{aligned} f'(x) &\approx \frac{f(x+h) - f(x)}{h} \\ f(x+h) &= f(x) + hf'(x) \end{aligned}$$

Thus, given an initial value $f_0 = f(x_0)$ we can compute the next step $f_1 = f(x_0 + h)$ and so on. The accuracy is dependent on the stepsize h, and the truncation error goes like $O(h^2)$. The Forward Euler method implemented for our case of the solar system, the coupled equations are given by

$$\begin{aligned} v_{i+1} &= v_i + a_i h \\ r_{i+1} &= r_i + v_i h \end{aligned}$$

for $i = 0, 1, \ldots, N-1$.

3.1.2 Velocity Verlet algorithm

The method is derived from using two Taylor expansions and subtracting one from the other. The error of the remainder goes as $O(h^3)$. What makes this method efficient is all the odd terms cancel out.

$$\begin{aligned} r_{i+1} &= r_i + v_i h + \frac{h^2}{2} a_i \\ v_{i+1} &= v_i + \frac{h}{2}(a_i + a_{i+1}) \end{aligned}$$

Notice that the a_{i+1} depends on r_{i+1}, so it must be computed first, in reverse order of the Euler method.

3.2 Models

As the solar system consist of many celestial bodies, we implemented four different models for testing our simulation under increasingly more complex systems.

3.2.1 Model 1: The Earth-Sun system

The first model is a simple system only containing the Earth and the Sun, where the Sun is stationary in the origin, with an orbiting the Earth. This can be thought of as the most simplified version of the solar system. This system is simulated using the Euler Forward algorithm and the Velocity Verlet algorithm as described in section 3.1. By starting with a well known system, we could easily spot problems with our solver, as well as determine which algorithm were the most suitable for further use. This was determined by comparing CPU times, energy and angular momentum conservation, as well as the stability of Earth's orbit around the Sun. We also did tests to check the escape velocity of our system, and compared it with the analytical solution (6).

The Python code is object oriented, but to make sure all functions were correctly constructed, we first made a program (`3b.py` [2]) without object orientation.

3.2.2 Model 2: The 3-body system

In the second model we introduced the 3-body problem, by adding Jupiter to our system. The simulation were first done with a stationary Sun. For this model we used the Velocity Verlet solver only, and we implemented initial positions and velocities for the planets by adding NASA data as described in Appendix A. Jupiter is the most massive planet, and will therefore have a measurable effect on Earth's orbit. We made this effect more extreme by increasing the mass of Jupiter by a factor 10 and then a factor 1000. For all three cases, we simulated the system for 100 years.

We also ran a simulation with a non-stationary Sun. Because the momentum of the solar system is generally not zero, we recorded all motion relative to the center of mass (CM). There are no external forces on the solar system, so the CM is not accelerated. This made the results more easy to interpret. By using the expressions for CM (8) and CM-velocity 9, we subtracted them from the initial position and velocities read from the NASA data set. See Appendix A.

3.2.3 Model 3: The n-body system or The solar system (n-body problem)

In the third model we added all planets in the solar system, including Pluto. We also made the simulation more realistic by using a non-stationary Sun, making all celestial bodies orbit around a center of mass. The system were simulated over a 250 year period, as the orbital time for Pluto is 248 years [9]. For this model we also used the Velocity Verlet solver, but we had to increase the number of iteration points to $n = 5 * 10^5$ to get a more stable Mercury orbit. Because we had 9 planets and the Sun, each with a mass that gave rise to a gravitational pull on all other planets with their masses, we had to perform a triple for-loop to simulate the motions. First we loop over all the time steps, for each time step we loop over each planet, for each planet we loop over all the other planets and compute the accumulated acceleration that planer will experience. To do this we used equation (7) for each time step. We also used CM, as in section 3.2.2.

3.2.4 Model 4: The Mercury-Sun system

In the last model we are taking a closer look at the relationship between the Sun and the nearest planet, Mercury. The aim with this model was to investigate the perihelion precision of Mercury, as described in section 2.4. In this model we also used the Velocity Verlet solver to simulate the system, starting with $n = 10^4$. Then we increased the number of integration points to observe the effects of a smaller time step.

In order to control for all the other influences of the perihelion's position, we only simulate Mercury's orbit around the Sun, neglecting all other planets. The expression for the acceleration now included the correction from general relativity, as shown in equation (13). The angular momentum, needed to be calculated for every time step. After running the simulation for 100 years, we found the last index of which the distance to the Sun was at a minimum, and used this index to find the corresponding x and y position of Mercury.

We then used equation (14) to calculate the perihelion angle at the beginning, $t = 0$ and at the end $t = 100$ years. We then subtracted one from the other to see how much it has changed over a century, and then compared with the known value from observations. We started our simulation at the perihelion, with $x = 0.3075$AU, $y = 0$ and $vx = 0$, $vy = 12.44$AU/yr.

4 Results and Discussion

4.1 Model 1: The Earth-Sun system

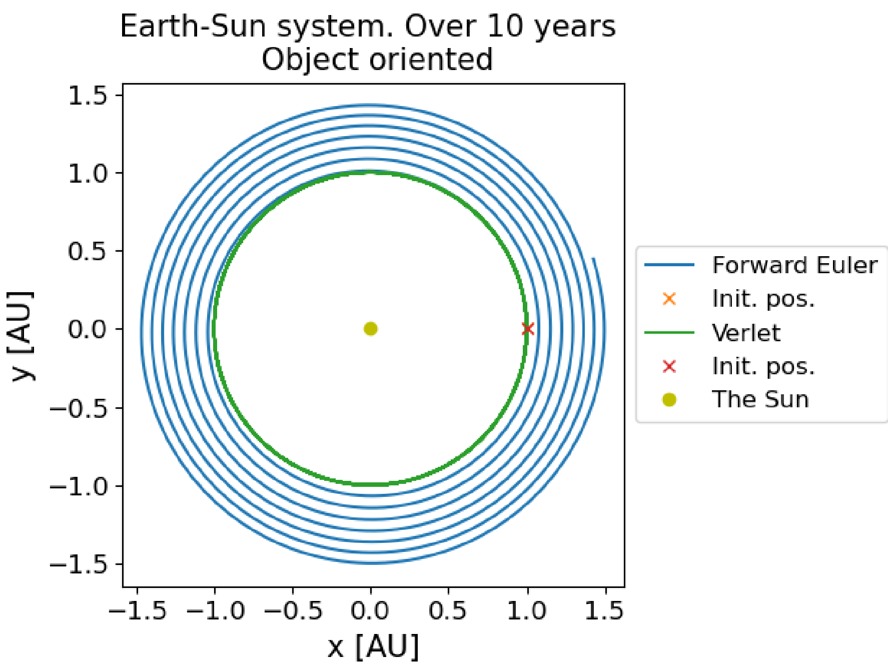

Figure 1: The Earth-Sun system using two methods, Forward Euler and Velocity Verlet. The Sun in the center of the system is not to scale. Initial velocity is $v_{0,y} = 2\pi$

In figure 1 we simulated the Earth-Sun system using Forward Euler and Velocity Verlet algorithm, where we used eq. (2) in our solvers, and $v_0 = 2\pi$, as explained in eq. (6).

We can clearly see that Velocity Verlet performs better than Forward Euler, as Forward Euler does not keep an stable orbit. Instead Earth's orbit spirals further and further away from the Sun. This is because of numerical errors where the calculated position is not accurate enough in the Forward Euler algorithm. It cannot account for the curvature accurately enough, so it spirals outwards. The Velocity Verlet method preserves energy in the system, unlike Forward Euler. In order for orbits to stay stable for a long period of time, the energy has to be conserved. In systems like this, with continuously changing velocities, more advanced methods than the Forward Euler method should be considered.

We tested our solver as a non-object oriented code in **3b.py**, giving us the exact same result as figure 1. The plot (**3b_Earth_Sun_system.png**) can be found on the GitHub page under *Results* [2].

4.1.1 Integration points & Conservation of energy and angular momentum

In Appendix B, we have illustrated the importance of a right time step $dt = \text{dist}/n$ (dist is the distance between two points) in the simulation, by varying the number of integration points n. We can see from figure 20 where $n = 10^6$, that both algorithms produces acceptable results, but the computational time becomes too long. See table 1. In figure 15 where $n = 10^1$, both algorithms fails to compute an orbit. The Velocity Verlet method gives an stable orbit in figure 16-19 ($n \in [10^2, 10^5]$) while Forward Euler fails.

Table 1: Computational time for the Earth-Sun system, using the Forward Euler and Velocity Verlet algorithm.

Method		n=10^1	n=10^2	n=10^3	n=10^4	n=10^5	n=10^6
Forward Euler	[s]	0.00	0.02	0.09	0.71	6.34	66.7
Velocity Verlet	[s]	0.00	0.05	0.14	1.16	11.7	124

Another important factor in choosing the best algorithm is to check if the energy and angular momentum is conserved in the system. In the two plots to the right in figure 2, we observe that both the energy and the angular momentum is conserved using the Velocity Verlet algorithm. For the Earth to be gravitationally bound by the Sun, the potential energy has to be larger than the kinetic energy, and therefor the total mechanical energy is negative, half of the potential energy [10]. To the left, we observe that for the Forward Euler algorithm, the total energy decreases with time, while the angular momentum increases. As these quantities should be conserved for circular orbits around a stationary body, the Forward Euler algorithm fails. In this case, Earth's orbit gets wider as the energy falls, and the angular momentum increases.

Based on these results, as well as the computational times in table 1, the Velocity Verlet method, with $n \in [10^4, 10^5]$, is the best algorithm to use further in our solar system simulation.

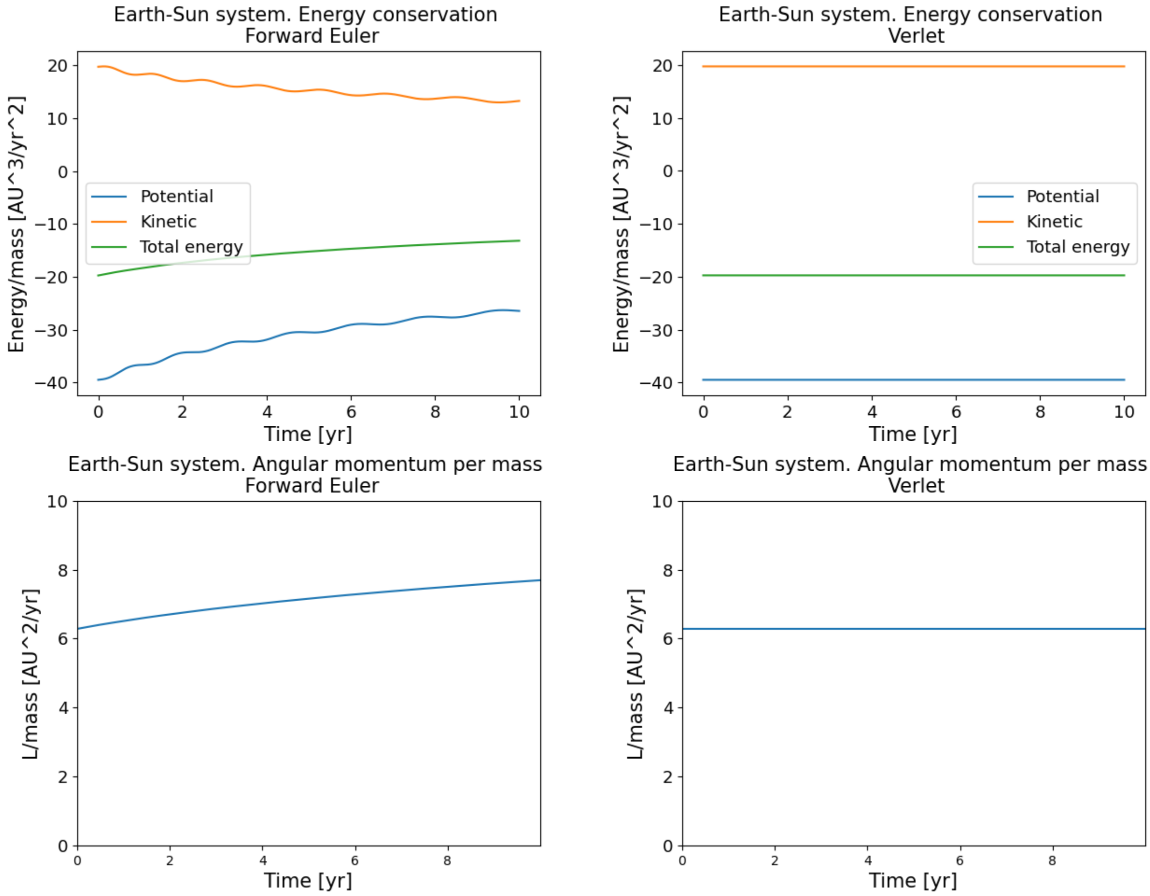

Figure 2: To the left we have plotted the energy and angular momentum for a 10 year period of the Earth-Sun system with Forward Euler, and with Velocity Verlet to the right. n=10^4.

4.1.2 Testing forms of the gravitational force

Until now we have assumed an inverse-square force, equation (2), but we can also investigate the case where we replace it with the following force

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

where we inserted $\beta = [2.0, 2.9, 3.0]$. We can see from figure 3 that as expected, $\beta = 2$ produces an bound circular orbit around the Sun. This is the case for all $\beta < 3$, but when $\beta = 3$, the system is no longer stable as the orbital period changes. The orbit of Earth spins outward, and will eventually escape the gravitational pull of the Sun.

We also created a system where the planet had an initial position at 1 AU from the Sun, like Earth, but an initial velocity of $v_{0,y} = 5$ AU/yr. See figure 4. We can see that we get an elliptical orbit, but as long as there is no external force acting on the system, the orbit will remain stable and the total energy is constant, as seen in figure 5. Because of the elliptical orbit, the scale of the gravitational interaction with the Sun will vary with the position on the orbit, making the potential and kinetic energy fluctuate up and down. The potential energy and the kinetic energy will be opposite of each other in the perihelion (closest) and aphelion (farthest) points, where we only have potential energy [10]. As mentioned in section 2.2.1, the angular momentum is constant in time for the stable system, as seen in the right plot. This is due to the gravitational force only acting in the radial direction [10].

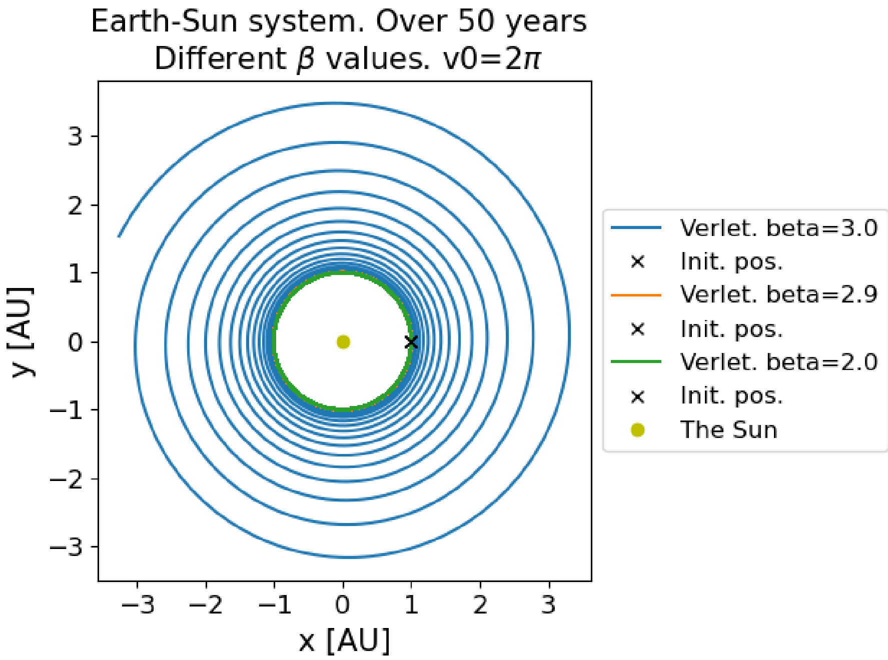

Figure 3: Testing β values for the Earth-Sun system with init.pos.=[1, 0] and $v_{0,y} = 2\pi$. n=10^4

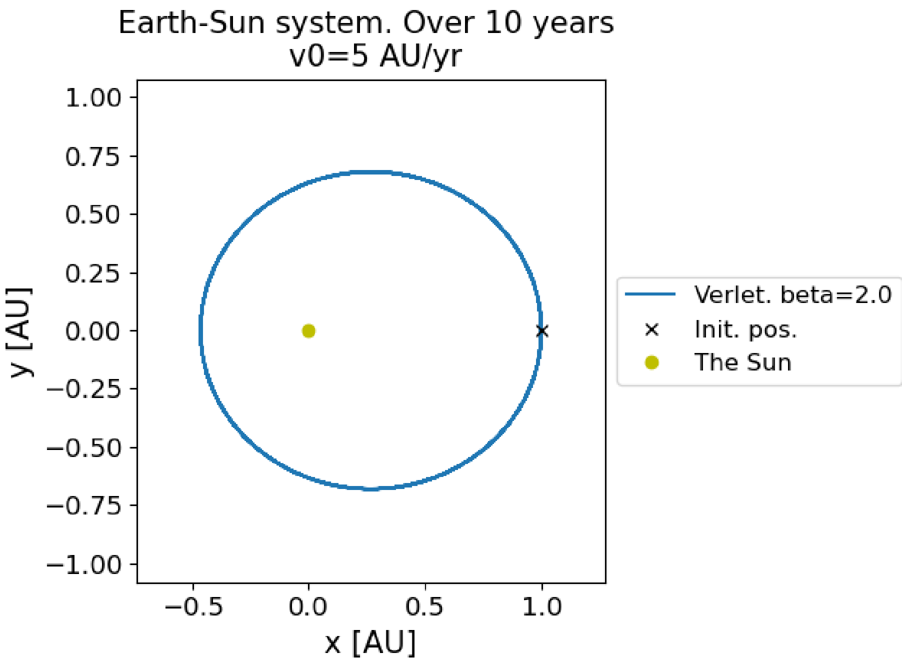

Figure 4: Earth-Sun system with init.pos.=[1, 0] and $v_{0,y} = 5$ AU/yr. n=10^4

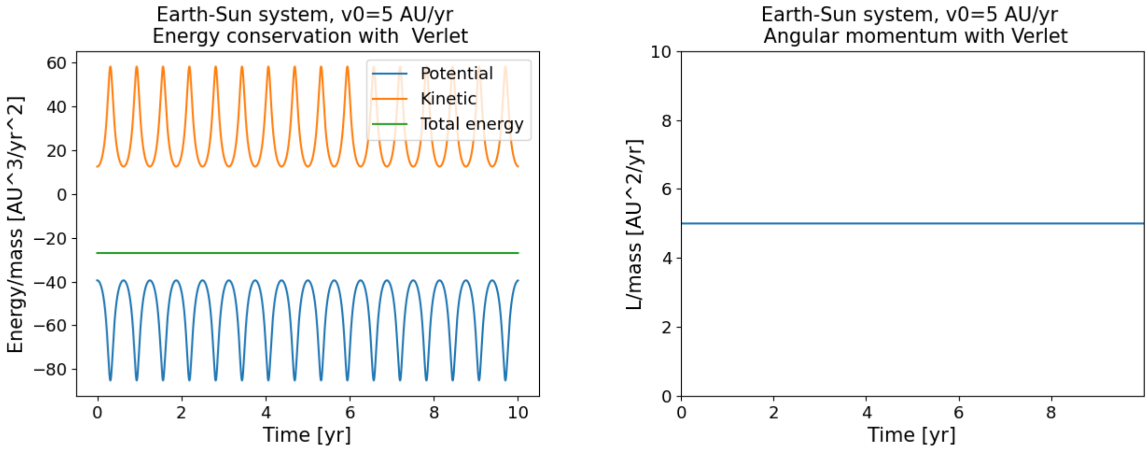

Figure 5: To the left we have plotted the system energy for a 10 year period of the Earth-Sun system, and the angular momentum to the right, with Velocity Verlet. n=10^4.

4.1.3 Escape velocity

In figure 6 we tested for different initial velocities $v_{0,y} = 2\pi * [0.9, 1.1, 1.3, 1.4, 1.415]$, to find the escape velocity for Earth. By implementing the formula for escape velocity (12), we observe that Earth escapes from the gravitational pull from the Sun when $v_{esc} = 2\sqrt{2}\pi \approx 8.89$. This was also achieved when $v_{0,y} = 2\pi * 1.415 \approx 8.89$ AU/yr. Velocities lower than this will not be escape velocities, but will make elliptical orbits. Besides from the initial velocity $v_0 = 2\pi$ that gives an circular orbit.

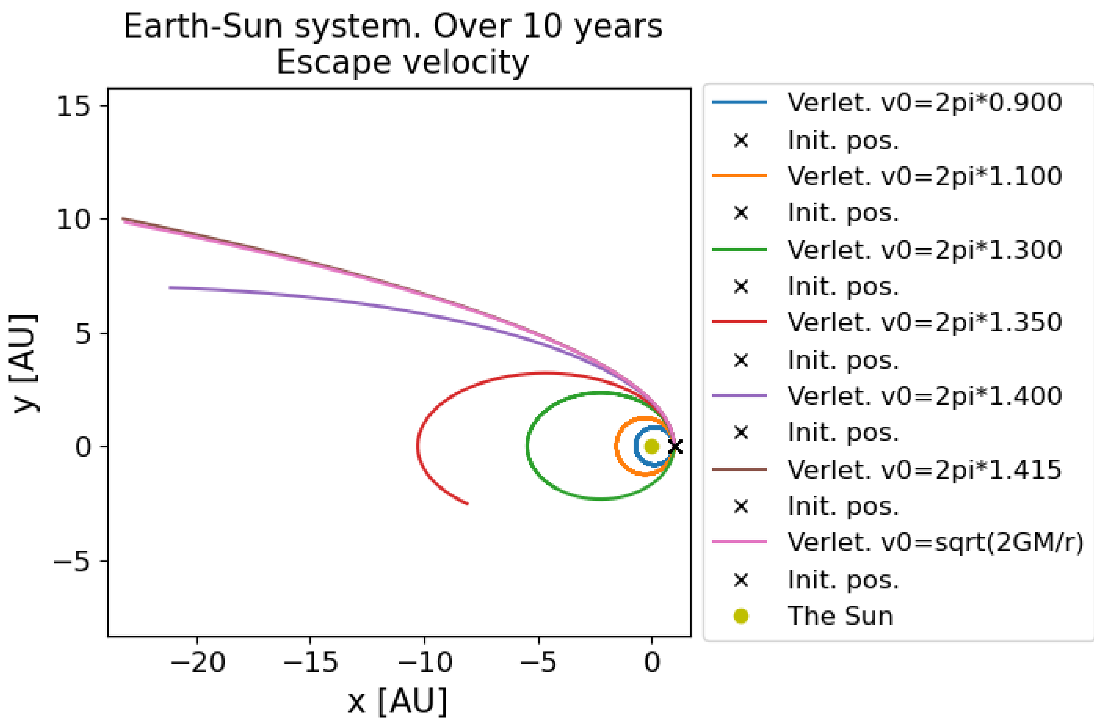

Figure 6: Testing various initial velocities to achieve escape velocity for the planet/Earth over 10 years. The Sun in the center of the system is not to scale.

4.2 The n-body problem

When expanding our model to represent the solar system, we suddenly have an n-body problem. This is not possible to express analytically, but our Velocity Verlet solver can. As the solar system is complex, we first looked at a 3-body problem, between the Sun, Earth and Jupiter. Now we also use initial positions and velocities as described in Appendix A, except for the Sun while stationary.

4.2.1 Model 2: 3-body problem. Earth-Jupiter-Sun

In the 3-body problem, we kept the Sun stationary, while the gravitational force between the three masses interacts with each other. In figure 7, we have simulated the system over 100 years, showing stable orbits. To make sure the gravitational force from Jupiter interacts with Earth, we increased the mass of Jupiter. First, we increased the mass by a factor of 10, where we could see Earths orbit got a slight wobble. This plot can be found on the GitHub page under *Results* [2], **3g_E_J_Sun_system_m10.png**.

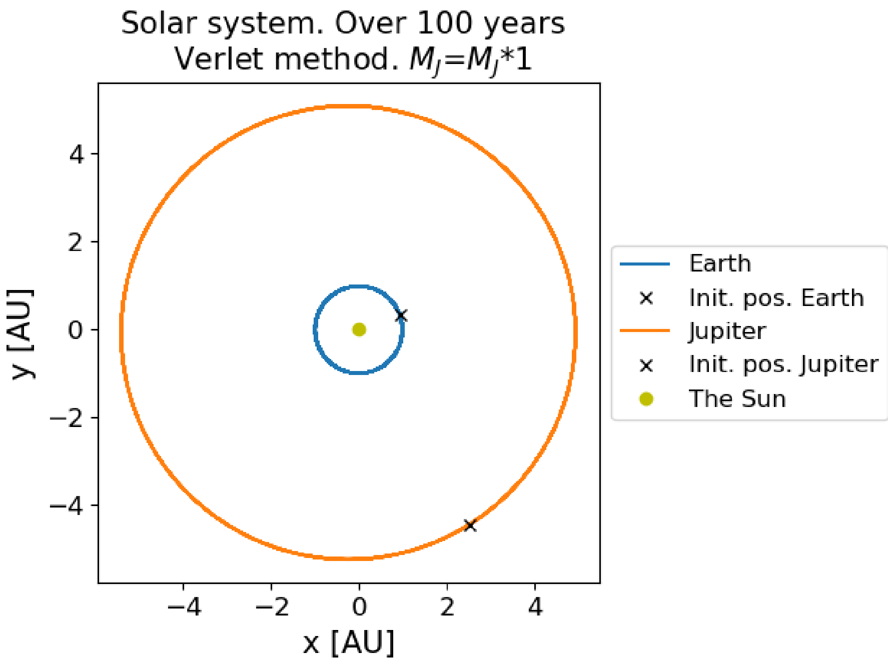

Figure 7: Earth-Jupiter-Sun system. n=10^4

Then, we increased the mass of Jupiter by a factor 1000, as shown in figure 8. The new mass of Jupiter is almost the same as the Sun, making the system chaotic. As the Sun is fixed in the center, the orbit of Jupiter is still stable, but the orbit of Earth is now disrupted and out of control. We can see in the left plot in figure 8 that the new orbit is chaotic around the Sun. This illustrates how unstable a n-body problem can be, and the more bodies we add, the more complex the system gets.

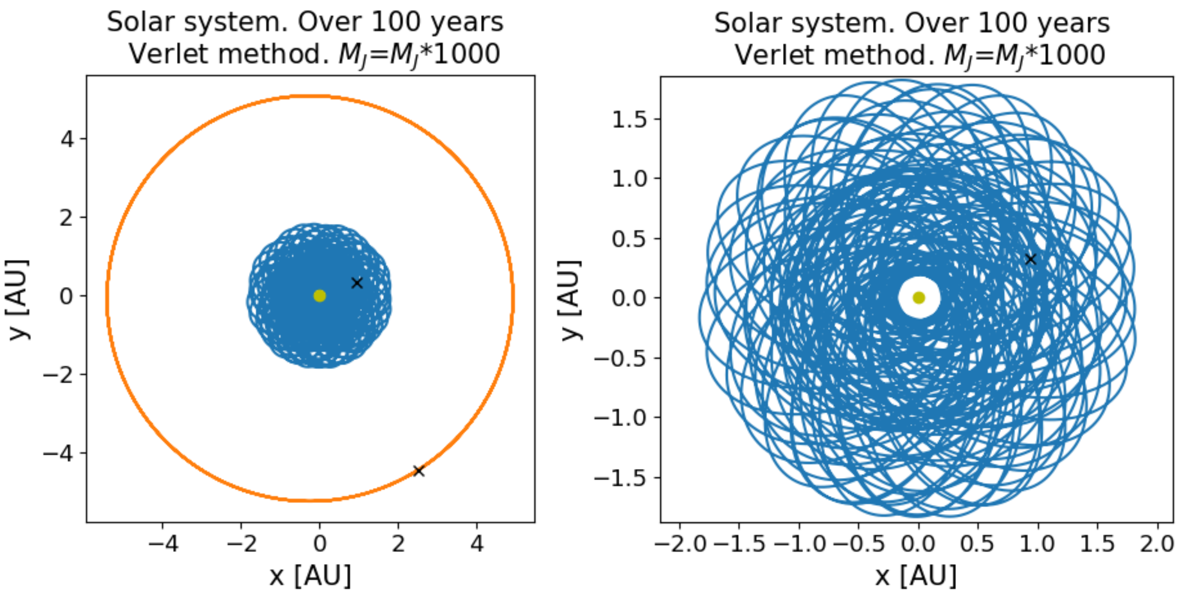

Figure 8: Earth-Jupiter-Sun system with $M_J = M_J * 1000$. Same label as figure 7. To the right is a zoomed in plot of Earth's disrupted orbit, with n=10^4

4.2.2 Model 2: 3-body problem. Earth-Jupiter-Sun. Non-stationary Sun

To make the previous 3-body problem more realistic, also the Sun is given an initial position and velocity as described in Appendix A. That means the Sun and the planets are orbiting around a center of mass as described in section 3.2.3, as seen in figure 9. This were mainly meant as a test for implementing the center of mass, before adding additional planets to the system.

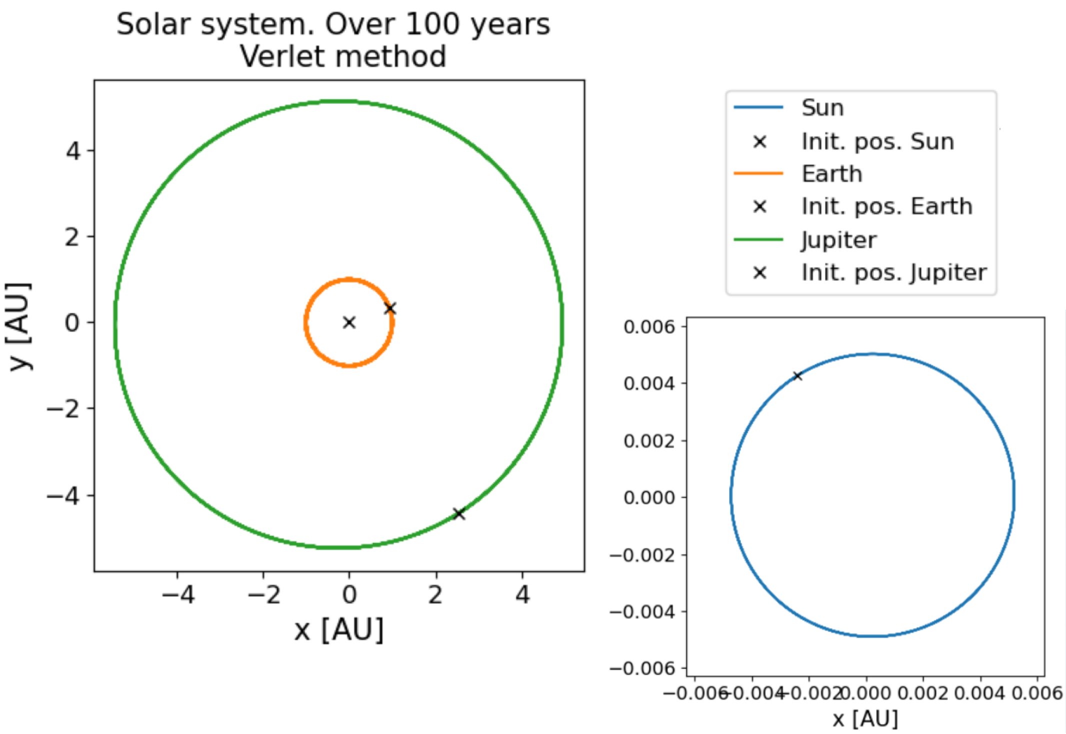

Figure 9: Earth, Jupiter and the Sun orbiting a center of mass

4.2.3 Model 3: Solar system model. Non-stationary Sun

In figure 10 we see the orbits of all the planets in the solar system (including Pluto), for a time period of 250 years. The orbits are as expected, and we see that Pluto "crosses" the orbit of Neptune (Pluto has a very different inclination [7]). In figure 11, we take a closer look at the inner planets. As we observe, the orbit of Mercury has a thicker line, meaning the orbit is unstable. The observed "drift" is likely caused by relativistic effects due to the closeness to the Sun, which is not taken into account.

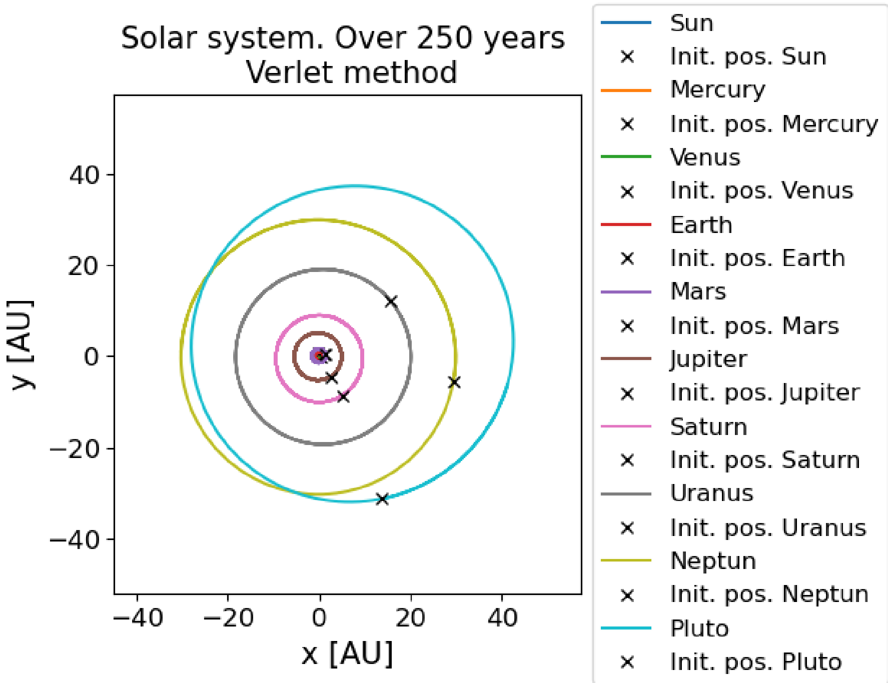

Figure 10: All the planets in the solar system with a non-stationary Sun

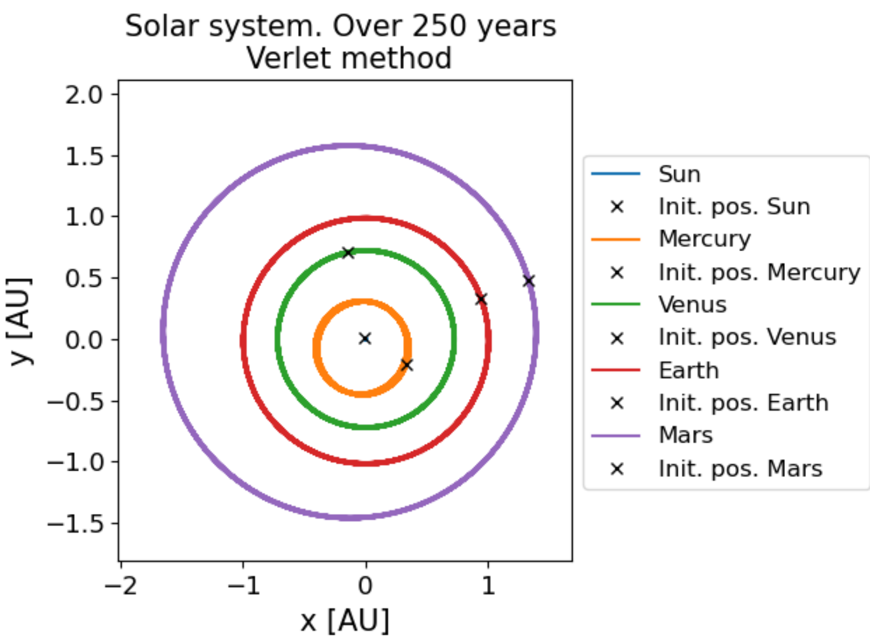

Figure 11: The inner planets in the solar system with a non-stationary Sun

4.3 Model 4: The perihelion precision of Mercury

In order to check the relativistic correction proposed in section 2.4, we implemented equation (13) in our solver. In figure 12 we can see that for $n = 10^4$, the relativistic correction was not able to stop the planet from drifting. When increasing the number of integration points, we could see that the orbit became more stable with the decreasing time step. With an increase of integration points to $n = 4.5 \cdot 10^5$ the orbit became much more stable, as presented in figure 13. In the figure we also see the position of Mercury for our last time step (black cross). In figure 14 we have zoomed in on the previous last minimum (purple cross), we see that the perihelion slightly differs from the initial perihelion (red cross).

With the $n = 4.5 \cdot 10^5$, we calculated the perihelion using equation (14)

$$\theta_p = arctan\left(\frac{-0.006392}{0.307434}\right) = -1.550009.$$

The initial position of (0.3075,0) yields an initial perihelion of 0.015708, so the deviation then becomes

$$\delta\theta_p = (0.015708 - 1.550009) = 0.000208 = 42.876311'' \approx 42.87'',$$

which is about what we expected from the model. As mentioned in section 2.4, the precision of $\sim 43''$ per century due to the relativistic force, was the missing contribution of the observed precision of Mercury's orbit [8]. Although the result was expected, it did take some time to find a value of n which resulted in such a close correspondence with the theory.

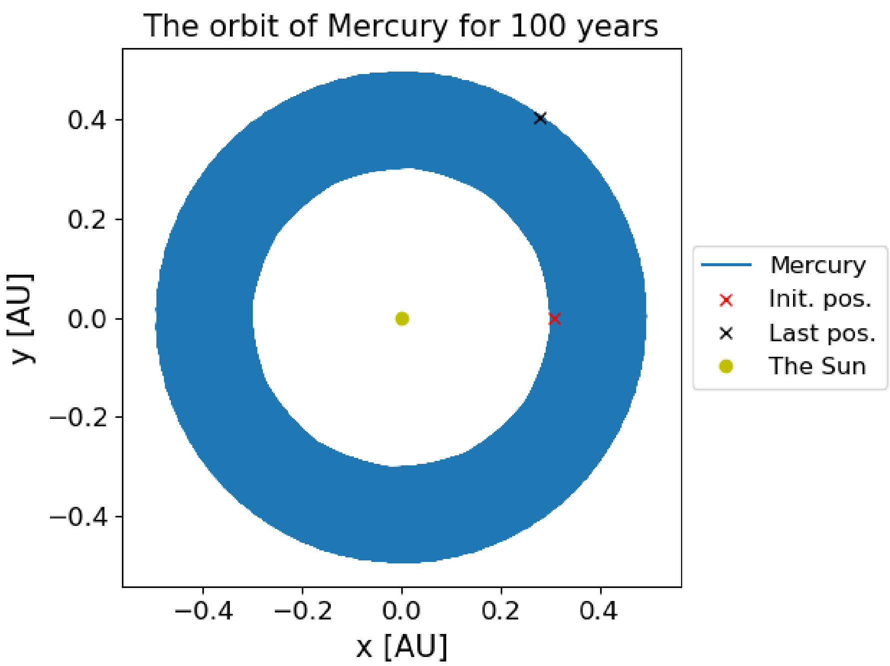

Figure 12: The orbit of Mercury, with relativistic correction and $n = 10^4$, is still drifting.

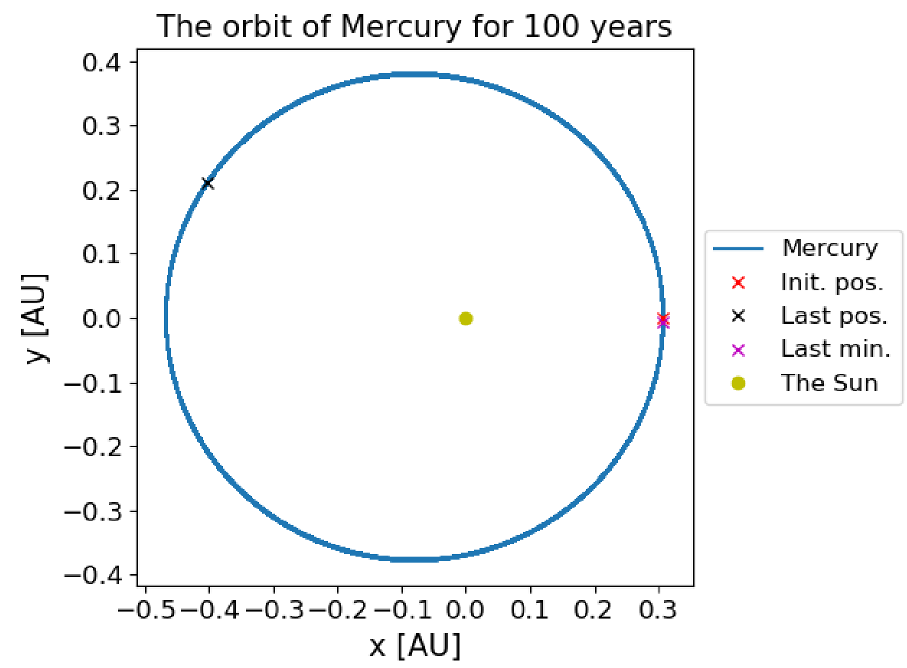

Figure 13: The orbit of Mercury becomes more stable with a relativistic correction and $n = 4.5 \cdot 10^5$.

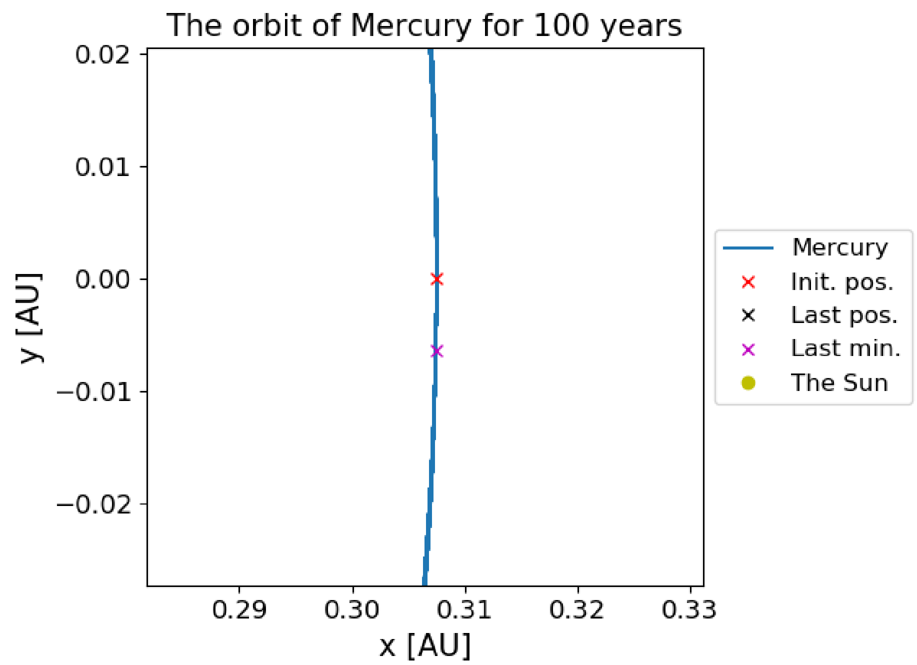

Figure 14: The initial perihelion (red cross) and the perihelion shift (purple cross) after one century, $n = 4.5 \cdot 10^5$.

Conclusion

By making various numerical models of the solar system, we have shown that the Velocity Verlet ODE solver works well for a system with preserved energy and torque. We also investigated the performance of the Forward Euler solver, and found that it did not perform as well. It takes less computation time, but needs a significantly higher number of integration points, which is why we mainly used the Velocity Verlet solver in this project.

We first looked into a simple 2-body problem, with the Earth and the Sun. We tested our algorithms' performance, by calculating conservation of energy and angular momentum, and testing if Earth escapes its orbit around the Sun as expected when given a certain escape velocity. After some testing of our algorithm and achieving satisfying results, we continued to more complex n-body problems. First in the special case of a 3-body problem, featuring the Earth-Jupiter-Sun system. This were done with a stationary Sun and a non-stationary Sun. In both cases the planets orbited the Sun with pretty similar orbits. When we increased the mass of Jupiter, this drastically changed the orbit of the Earth, creating a chaotic orbit. When looking at the n-body problem featuring all 9 planets and a non-stationary Sun, we observed more or less stable orbits. This is due to the center of mass, and an increase of integration points.

When taking a closer look at the inner planets, the orbit of Mercury were not as stable as the rest of the orbits of the planets. This behaviour was expected because of Mercury's close position to the Sun. When applying a relativistic correction to the gravitational force, we were able to get a more stable orbit for Mercury. Then we calculated the precision of the perihelion caused by the relativistic effects, and found a precision of 42.87" per century, which corresponds quite well to the prediction of general relativity. However, our predictions was quite sensitive for the choice of integration points around this order of magnitude. In the future, it will be interesting to see if the prediction of the perihelion precision is more stable for a higher number of integration points.

References

[1] Khan Academy. *What is conservation of energy?* Visited: 24.10.2020 https://www.khanacademy.org/science/physics/work-and-energy/work-and-energy-tutorial/a/what-is-conservation-of-energy.

[2] Aron J. Nordberg Anna Eliassen and Kristina Othelia L. Olsen. *FYS4150: Project 3 - GitHub*. https://github.com/kristinaothelia/FYS4150/tree/master/Project3.

[3] Dale A. Ostlie Bradley W. Carroll. “An Introduction to Modern Astrophysics”. In: 2017. ISBN: 9781108422161.

[4] Morten Hjorth-Jensen. “Computational Physics Lectures: Ordinary differential equations”. In: (Oct. 2017). http://compphysics.github.io/ComputationalPhysics/doc/pub/ode/pdf/ode-print.pdf, p. 18.

[5] Morten Hjorth-Jensen. “Computational Physics Lectures: Ordinary differential equations”. In: (Oct. 2017). http://compphysics.github.io/ComputationalPhysics/doc/pub/ode/pdf/ode-print.pdf, p. 19.

[6] Morten Hjorth-Jensen. “Computational Physics Lectures: Ordinary differential equations”. In: (Oct. 2017). http://compphysics.github.io/ComputationalPhysics/doc/pub/ode/pdf/ode-print.pdf, p. 20.

[7] The Planetary Science Communications team at NASA’s Jet Propulsion Laboratory. *In Depth: Pluto*. Page Updated: December 19, 2019 https://solarsystem.nasa.gov/planets/dwarf-planets/pluto/in-depth/.

[8] M. G. Stewart. “Precession of the perihelion of Mercury’s orbit”. In: *American Journal of Physics* 73.730 (2005). DOI: 10.1119/1.1949625.

[9] Wikipedia. *Pluto: Orbit*. Last edited: 26.10.2020 https://en.wikipedia.org/wiki/Pluto#Orbit.

[10] Young and Freedman. “University Physics with modern physics”. In: 13th ed. Vol. 1. Pearson, 2011. Chap. 13, pp. 414–416. ISBN: 9780321696861.

Appendix A Initial conditions & Planetary data

In this project we will use astronomical units (AU), where a unit of length is known as 1 AU. This is the average distance between the Sun and Earth, 1 AU = 1.5×10^{11} m. Further, we will use years instead of seconds, since it's a more appropriate time scale when simulating planet orbits. On the NASA site (http://ssd.jpl.nasa.gov/horizons.cgi#top) we collected initial conditions in order to start our differential equation solver. To download the data for each planet, we changed **Ephemeris Type** from **OBSERVER** to **VECTOR**. The .txt file for each planet contains the x, y and z values as well as their corresponding velocities [AU/day]. All NASA .txt files are located on our GitHub page under Data, and the used data are for 12.10.2020. A summary of the data used for our simulations are listed in table 2.

When looking at the Earth-Sun system alone, we will initialize Earth's position with $x = 1$ AU and $y = 0$ AU.

Table 2: Mass and distance from the Sun for all planets (and Pluto) in the solar system.

Object	**Mass [kg]**	**Initial pos. (x,y) [AU]**	**Inital vel. (x,y) [AU/day]**
Planet			
Mercury	3.3×10^{23}	$(3.29, -2.04)\times10^{-1}$	$(9.42, 25.1)\times10^{-3}$
Venus	4.9×10^{24}	$(-1.42, 7.12)\times10^{-1}$	$(-19.9, -3.95)\times10^{-3}$
Earth	6×10^{24}	$(9.38, 3.29)\times10^{-1}$	$(-5.84, 1.62)\times10^{-2}$
Mars	6.6×10^{23}	$(1.33, 0.48)\times10^{0}$	$(-4.17, 14.7)\times10^{-3}$
Jupiter	1.9×10^{27}	$(2.53, -4.45)\times10^{0}$	$(6.46, 4.09)\times10^{-3}$
Saturn	5.5×10^{26}	$(5.13, -8.58)\times10^{0}$	$(4.48, 2.85)\times10^{-3}$
Uranus	8.8×10^{25}	$(1.55, 1.22)\times10^{1}$	$(-2.46, 2.91)\times10^{-3}$
Neptun	1.03×10^{26}	$(29.4, -5.48)\times10^{0}$	$(0.56, 3.11)\times10^{-3}$
Pluto	1.31×10^{22}	$(1.38, -3.12)\times10^{1}$	$(2.93, 0.59)\times10^{-3}$
Star	$M_\odot$		
The Sun	2×10^{30}	$(-6.08, 6.44)\times10^{-3}$	$(-7.31, -5.06)\times10^{-6}$

Note 1: *The initial velocity for all objects are scaled to AU/yr in the simulation*
Note 2: *Although Pluto no longer is classified as a planet in the Solar System, we choose to include it for historical reasons*

Appendix B Integration points n, orbit stability

In this Appendix we have plotted the Earth-Sun system over a 10 year period for $n \in [10^1, 10^6]$. The legend for all orbit stability plots are as follows

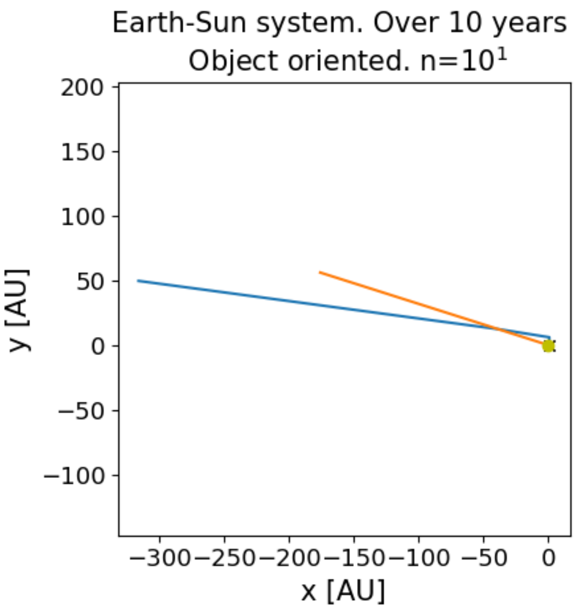

Figure 15: Earth-Sun system with $n = 10^1$

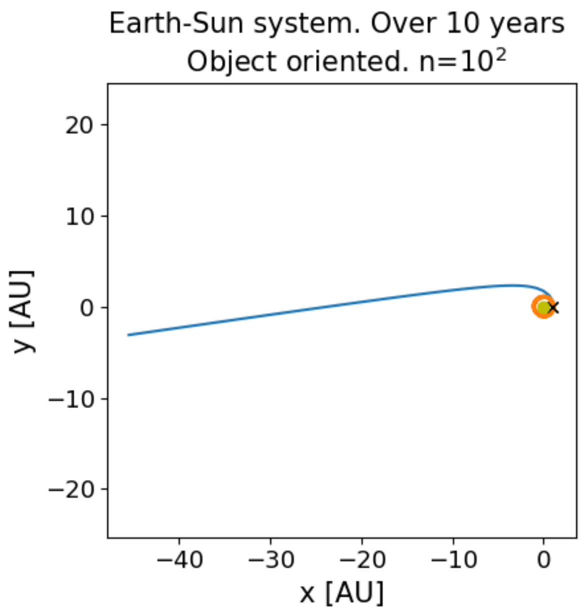

Figure 16: Earth-Sun system with $n = 10^2$

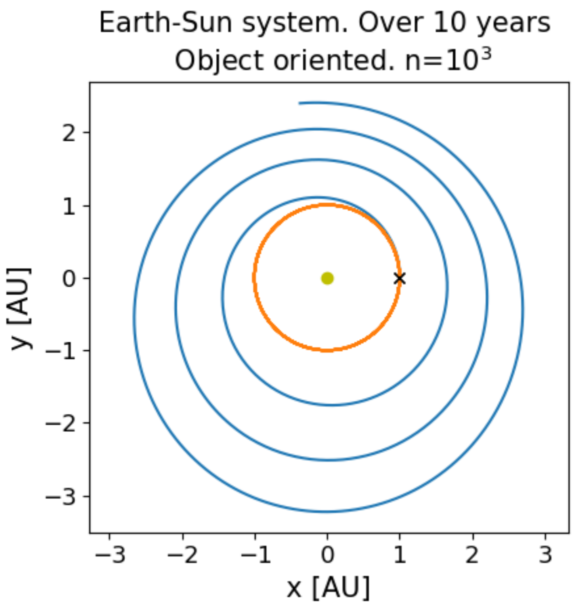

Figure 17: Earth-Sun system with $n = 10^3$

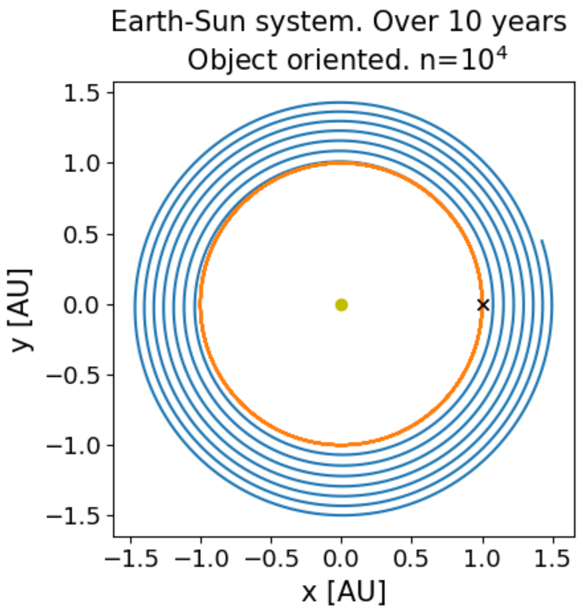

Figure 18: Earth-Sun system with $n = 10^4$

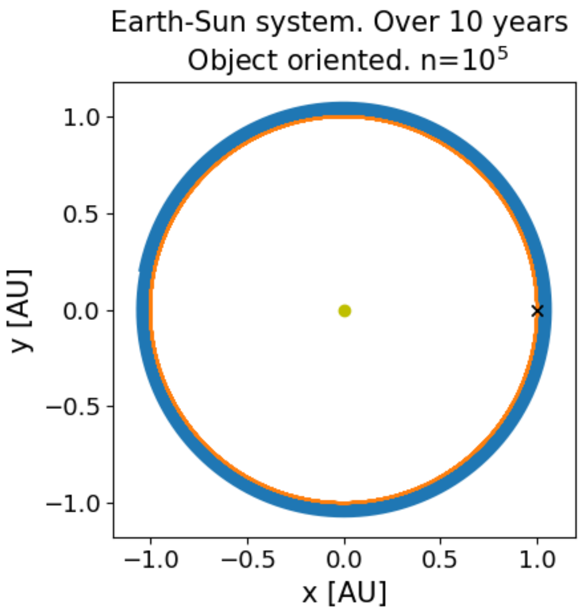

Figure 19: Earth-Sun system with $n = 10^5$

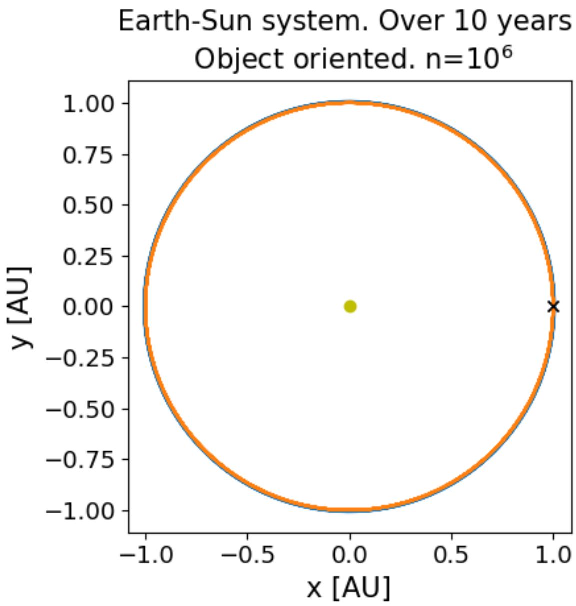

Figure 20: Earth-Sun system with $n = 10^6$