The N-Body Problem

Michael Kane

April 3, 2020

1 Introduction

The N-Body problem offers an insight into the movement of N bodies in a typically celestial system. It can be applied to any system containing massive bodies with the presences of a gravitational field, such as our solar system.

2 Overview

When starting this problem, I decided to use the Forward Euler approach to plot the position of each mass with respect to time; the goal was to see how changes in the initial orientation and initial velocity of different planets could affect its orbit as well as the orbit of other planets in the solar system. We know that the EOMs that control the motion of each particle can be written as

$$m_i \frac{d^2\vec{r}_i}{dt^2} = \sum_{j=1}^{N} \frac{G m_i m_j (\vec{r}_j - \vec{r}_i)}{\left\|\vec{r}_j - \vec{r}_i\right\|^3} \qquad j \neq i \tag{1}$$

we know that $\vec{a}_i = \frac{d^2\vec{r}_i}{dt^2}$. From this, w can simplify (1) to get

$$\vec{a}_i = \sum_{j=1}^{N} \frac{G m_j (\vec{r}_j - \vec{r}_i)}{\left\|\vec{r}_j - \vec{r}_i\right\|^3} = f(\vec{x}(t)) \tag{2}$$

I broke this into two first order ODEs using the velocity of each particle giving

$$\frac{d\vec{v}}{dt} = f(\vec{x}(t)) \tag{3}$$

$$\frac{d\vec{x}}{dt} = \vec{v} \tag{4}$$

I wanted to use the Euler approach because although it is simple in design, it is quite powerful in what it can accomplish.

3 Derivation and Implementation

3.1 Deriving the Forward Euler Method

I decided to use the forward Euler approach to solve these sets of equations because it is a robust method that is applicable to many scenarios and is easy to implement. From class we know that forward Euler is a one-step method as seen in lecture 11. This method takes the general form of

$$\vec{u}_{k+1} = \vec{u}_k + \Delta t \boldsymbol{f}(\vec{u}_k, t_k) \tag{5}$$

This clearly shows that the forward Euler is an explicit method where an initial condition, $\vec{u}_k$, explicitly gives an estimate value, $\vec{u}_{k+1}$; we only need to evaluate the right hand side of the equation. This equation comes from (6) where we can define the right hand side as a constant, leading to (7).

$$\int_{t_k}^{t_{k+1}} \dot{\vec{u}} dt = \int_{t_k}^{t_{k+1}} \boldsymbol{f}(\vec{u}, t) dt \tag{6}$$

$$\int_{t_k}^{t_{k+1}} \dot{\vec{u}} dt = \Delta t \boldsymbol{f}(\vec{u}(t_k), t_k) \tag{7}$$

The left hand side of (7) yields

$$\vec{u}_{k+1} - \vec{u}_k = \Delta t \boldsymbol{f}(\vec{u}(t_k), t_k) \tag{8}$$

as discussed in class; we can see that (8) is the same as (5). We represent $\boldsymbol{f}$ as a constant over our interval, $t \in [t_k, t_{k+1}]$, because at each t_k we are interpolating locally and finding a single value at that time. We can find our new approximate value by starting at our initial condition, u_k, and advancing it at every time increment to find our new value. The Forward Euler method is considered one-step because we are working explicitly, meaning we can only use the information that is contained within our time interval. We are starting at initial values and approximating the new values in $[t_k, t_{k+1}]$ at every time increment using equations (2),(3), and (4). The truncation error of the forward Ruler approach is defined as

$$\tau_k = \frac{\vec{u}(t_{k+1}) - \vec{u}(t_k)}{\Delta t} - \boldsymbol{f}(\vec{u}(t_k), t_k) \tag{9}$$

As we discussed in lecture 12, we can rewrite the truncation error to be

$$\tau_k = \frac{\vec{u}(t_{k+1}) - \vec{u}(t_k)}{\Delta t} - \dot{\vec{u}}(t_k) \tag{10}$$

$$\tau_k = \frac{\vec{u}(t_{k+1}) - \vec{u}(t_k)}{\Delta t} - \frac{\vec{u}(t_{k+1}) - \vec{u}(t_k)}{\Delta t} + O(\Delta t) \tag{11}$$

$$\tau_k = O(\Delta t) \tag{12}$$

The truncation error is the error that arises when we take the exact solution and advance it by the time step Δt. There is also cumulative error that comes from advancing the exact solution multiple times; this along with the truncation will lead to a global error.

3.2 Implementation

When implementing this method, I used the canned MatLab function **planetEphemeris()** to find the initial conditions for each planet in the solar system at a given time. Acceleration for each mass was found using the positions of each planet and (2). In order to find the new velocity, $v(t_{k+1})$, the initial acceleration was multiplied by the time step and added to the old velocity, $v(t_k)$. The same process was used using the initial positions and the calculated new velocity to find the new position. This position was tracked over the course of the simulation and then used to find the new acceleration as the process was repeated. The simulation was ran for the length of Pluto's orbit so as to get an accurate result of each planet's orbit. A time step of $1x10^4$ was used; this is because if a small time step was used, this method would have broken since the run time was so large ($7.9243800x10^9$ seconds).

4 Testing

Once I successfully simulated the solar system's orbit, I wanted to see how changing various conditions of the solar system. Primarily, I wanted to see how sensitive the system was to changes in mass, whether it be the mass of the Sun or individual planets. I began by increasing and decreasing the mass of the Sun and compared the orbits to that of Earth and Mars. I then expanded this notion to other parts of the solar system.

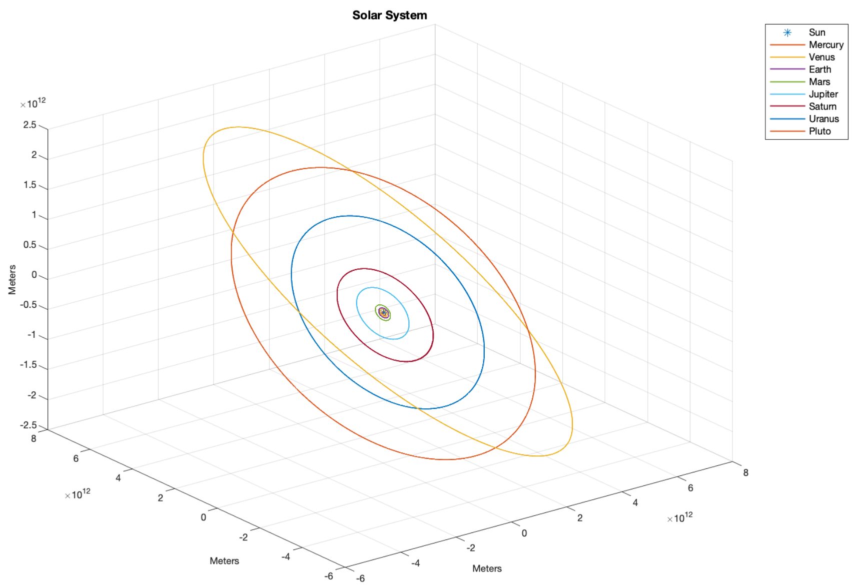

Figure 1: Plots of the Solar Systems Orbit (N = 11)

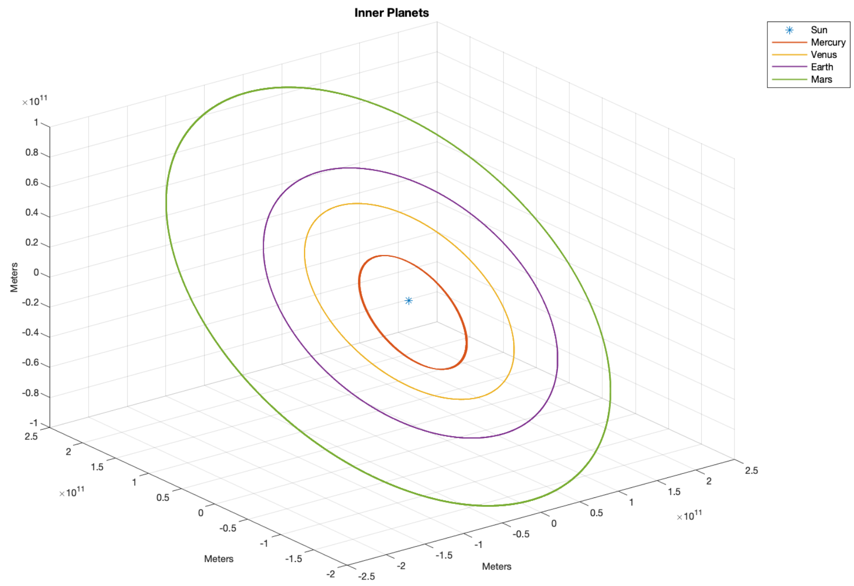

Figure 2: Plots of the Inner Planets

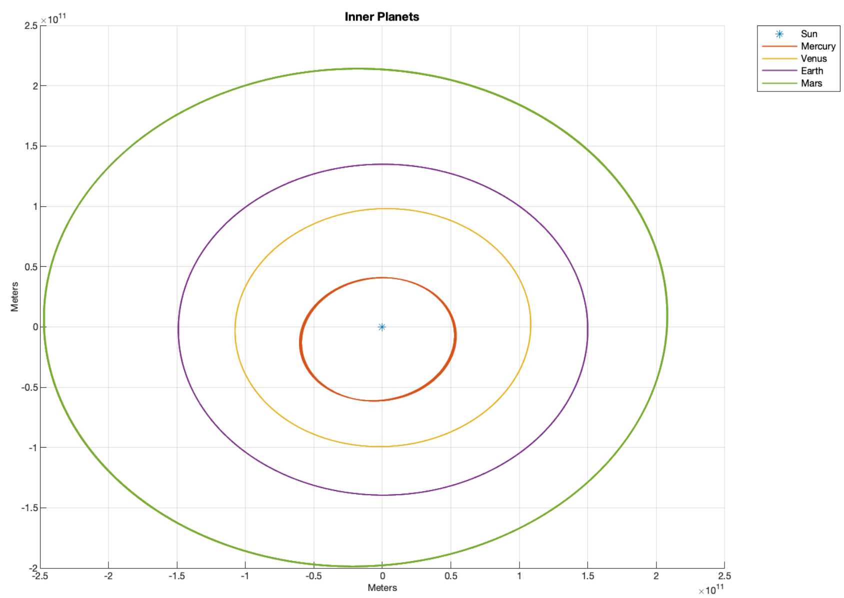

Figure 3: Plots of the Inner Planets in the X-Y Plane

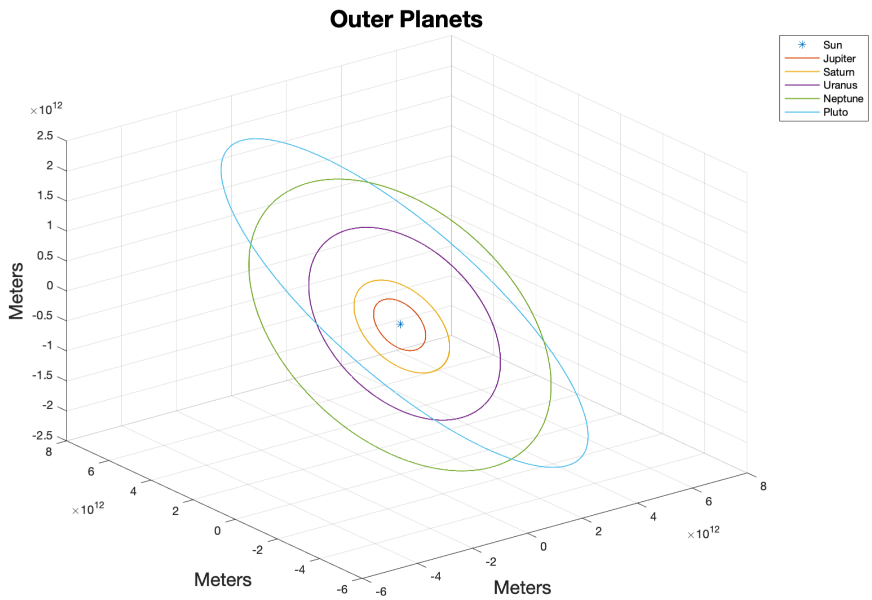

Figure 4: Plots of the Outer Planets

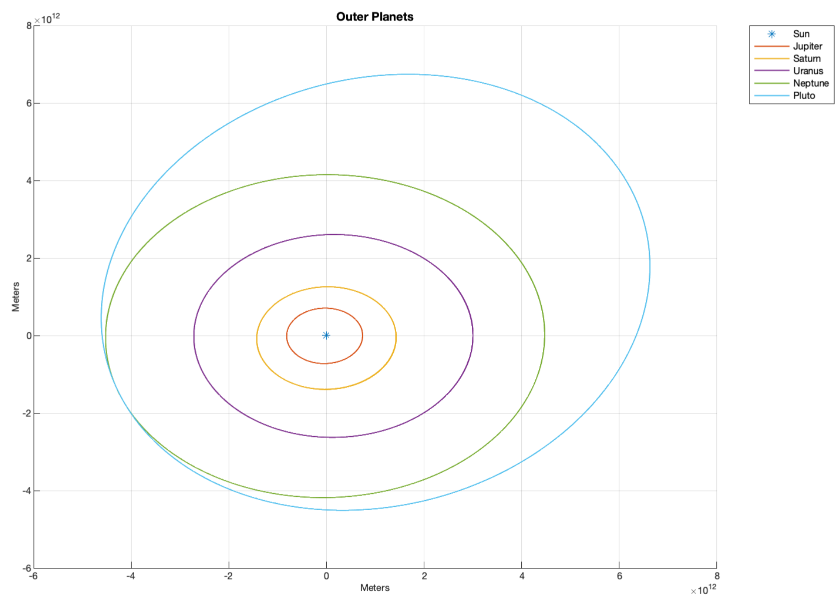

Figure 5: Plots of the Outer Planets

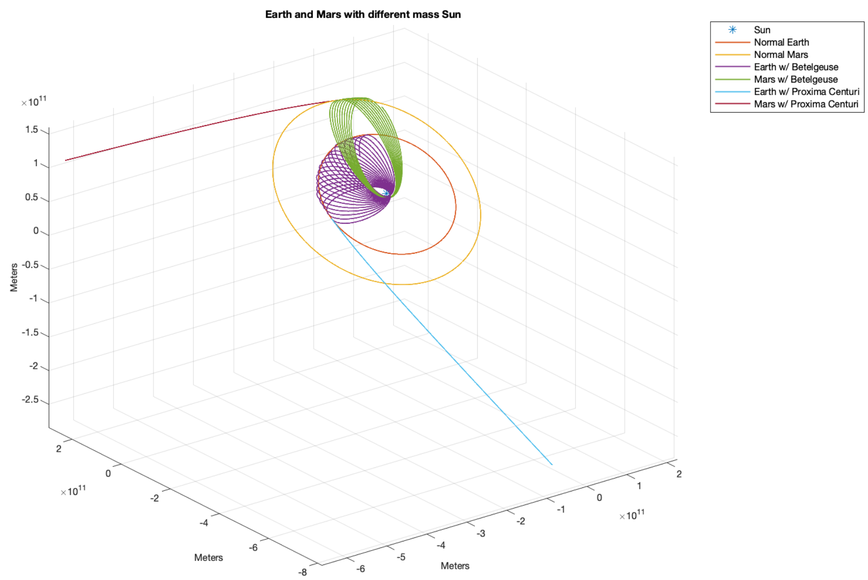

Figure 6: Plots of Earth and Sun with different masses for the Sun

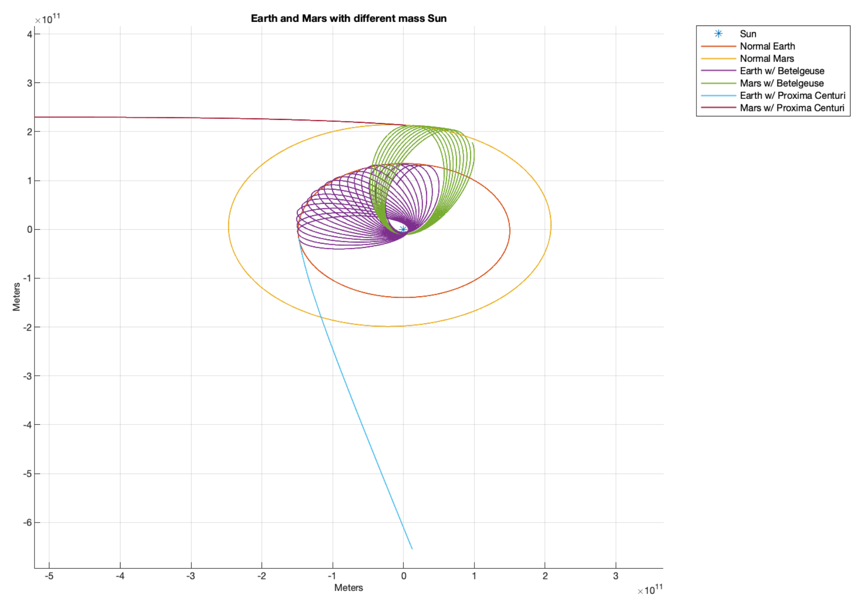

Figure 7: Plots of Earth and Sun with different masses for the Sun (Top View)

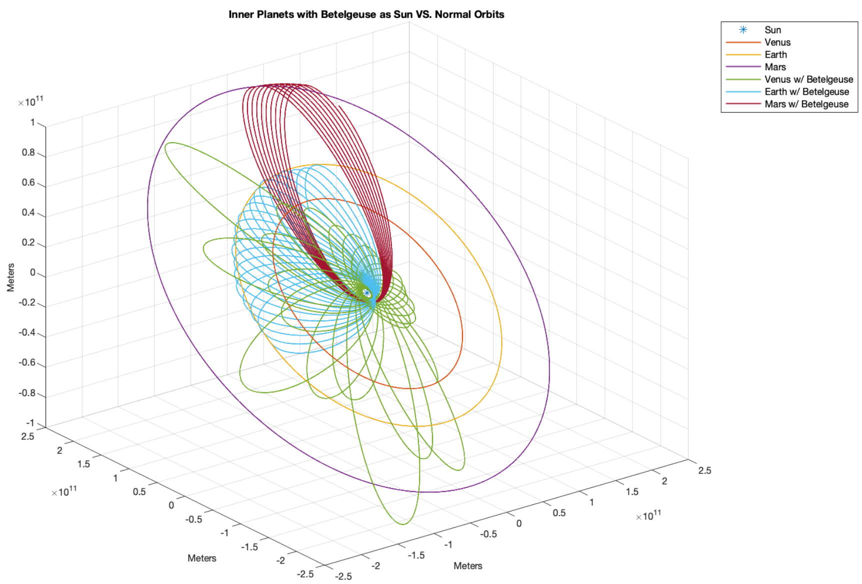

Figure 8: Plots of the Inner Planets with a more massive Sun

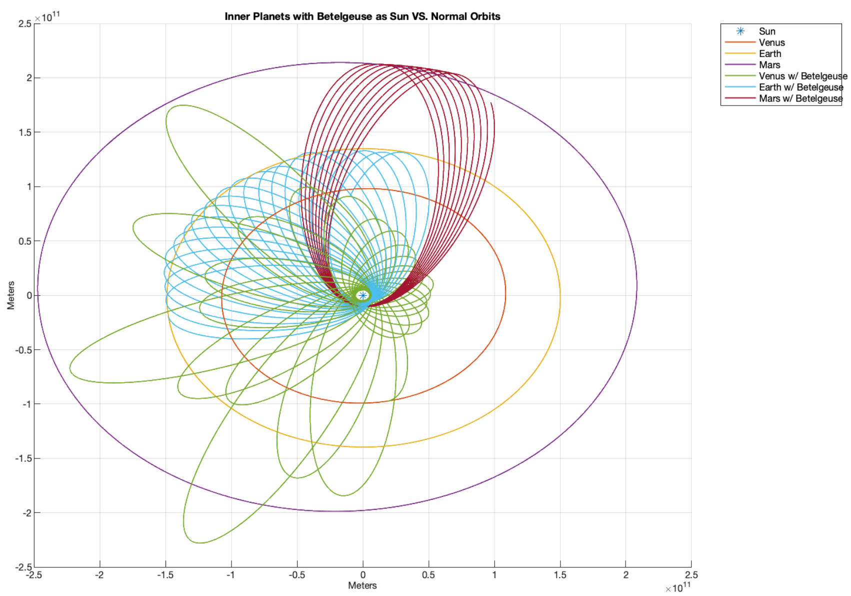

Figure 9: Plots of the Inner Planets with a more massive Sun (Top View)

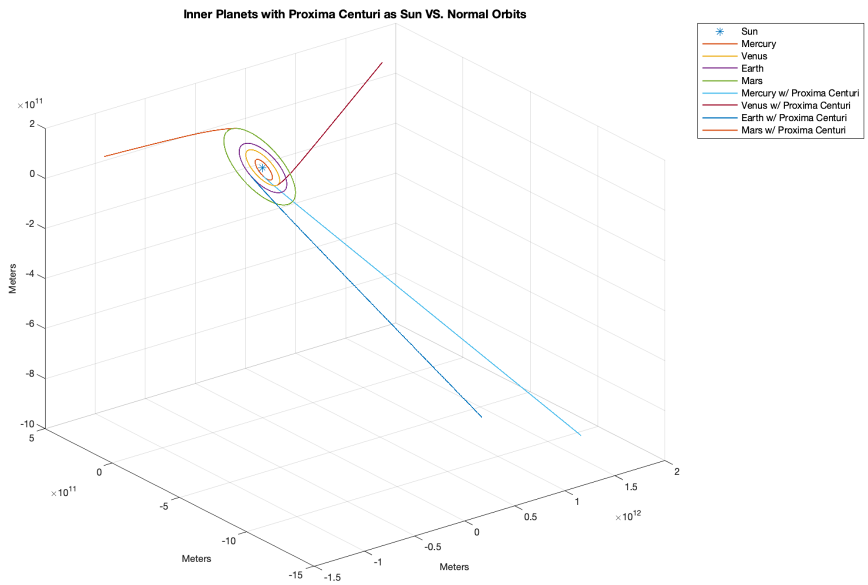

Figure 10: Plots of the Inner Planets with a less massive Sun

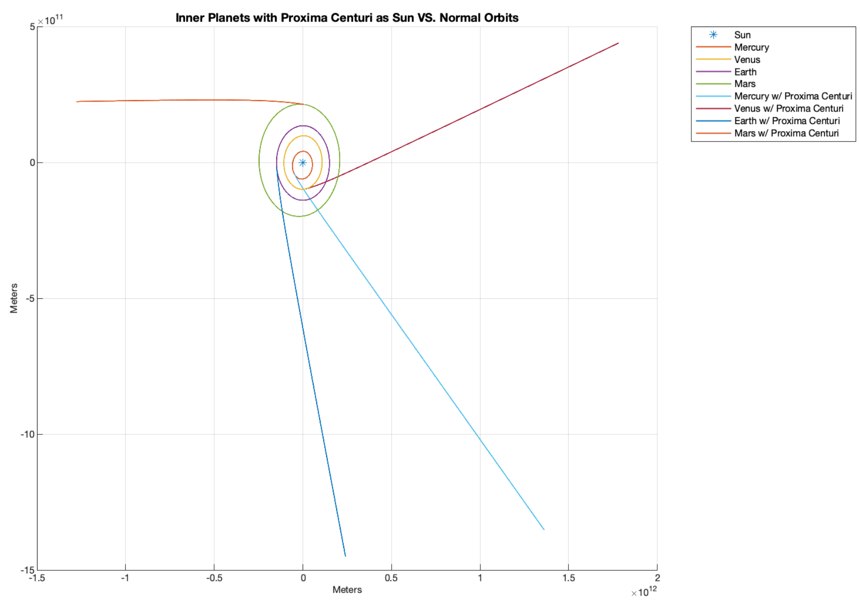

Figure 11: Plots of the Inner Planets with a less massive Sun (Top View)

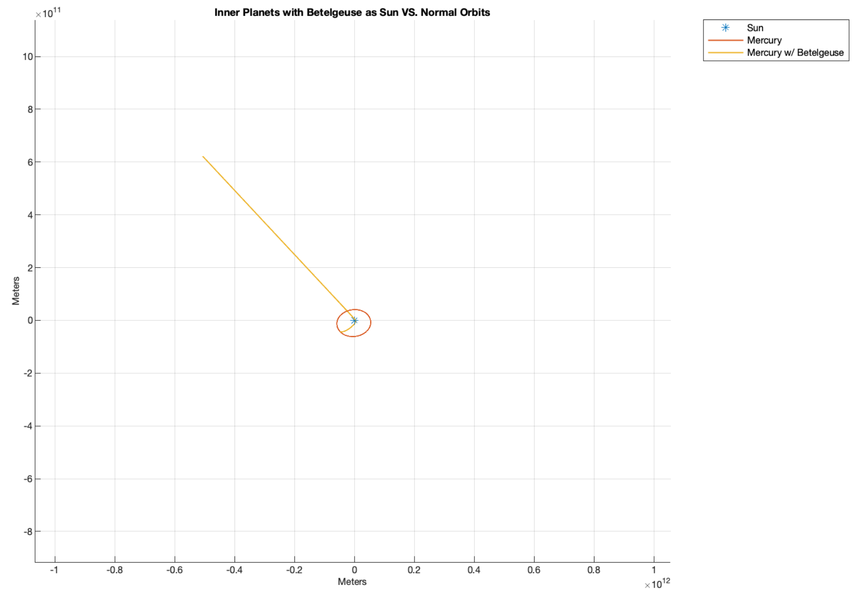

Figure 12: Plots of Mercury with massive Sun (Top View)

4.1 Results

Figure 1 displays the positions of each of the planets in our solar system (including Pluto), where the Sun is the center. This represents a system with $N = 11$ bodies; figure 2 and figure 4 break this system into two graphs as it is difficult to see the inner planets since the outer planets orbits are much larger. Figures 3 and 5 give a view of the inner and outer planets on the $XY - plane$. After generating the baseline results, I began changing the mass of the Sun to see how it would affect the orbits of each planet. I knew that decreasing the mass of the Sun would increase the size of the orbits as well as the orbital periods and vise versa increasing the mass of the sun. Figures 6 and 7 compare the orbits of Mars and Earth with their respective orbits if the Sun had a smaller mass and a larger mass. Figures 8, 9, 10, and 11 show the inner planets when the mass of the Sun is changed. In the case of a less massive Sun, I used the mass of Alpha Centauri ($2.446x10^{29}kg$). In the case of a more massive Sun, I used the mass of Betelgeuse ($2.188x10^{31}$) to represent the mass of the Sun.

4.2 Analysis of Results

As seen in figure 6 and 7, changing the mass of the Sun caused the orbit of Earth and Mars to change drastically. I used the mass of Alpha Centauri and Betelgeuse as the Sun because I wanted to see what would happen if our Sun was replaced with a larger and smaller star; Alpha Centauri's mass is much smaller than that of the Sun, so the motion of the planets is expected as they shot out of orbit. The force due to gravity from the Sun was not strong enough to keep the planets in orbit. As for the path of the planets with a more massive Sun, the orbital period was drastically decreased as well as the perihelion of each planet. It appeared that that the new aphelion of each planet initially occurred at the same time as the planets were shot out of orbit for the less massive Sun.

I then wanted to see what would happen if I expanded this to the inner planets. When looking at the system with the more massive Sun, I could not include Mercury when viewing all of the inner planets; the force of the star pulled Mercury out of its orbit and sent it into space as seen in 12. The other planets are seen in 8 and 9. Similar results were found as decreasing the mass of the Sun caused all orbits to stray from the normal orbits. Increasing the mass of the Sun provided some very interesting orbits for Venus.

The greater mass pulled the planets closer to the center of the system and caused the path of the planets to change with each pass around the Sun.

In order to keep the results consistent and see what would happen over a larger time, the total length of the simulation corresponded with 1 orbit of Mars; this is because it had the largest period.

5 Conclusions

When diving into the N-body problem, I focused my project and research around the forward Euler method on our solar system. I plotted the orbital trajectories of the planets in the system with the Sun being the center ($N = 11$). I dove into the trajectories of the inner planets if we were to have a different Sun or star by changing the initial mass. I found that changes in mass lead to massive changes in the orbits of the planets; a decrease in the mass of the Sun caused all of the planets to shoot out of orbit where as an increase in the mass caused the orbital periods to decrease and the planets to seem to oscillate around the center of the system. When looking at the results of the simulation, it is important to account for the error that comes with using the forward Euler method as it approximates a new value at a set number of time

6 Appendix

References

[1] Goza, Andres. *Initial value problems: introduction and one-step methods*. Lecture 11-13

[2] G. Strang. *Computational Sciene and Engineering*. Wellesley-Cambridge Press, 2007.

[3] Williams, David R. *Planetary Fact Sheet*. `nssdc.gsfc.nasa.gov/planetary/factsheet/`, 2019.

[4] Zeltkevic, Michael. *"Forward and Backward Euler Methods."*. $web.mit.edu/10.001/Web/Course_{N}otes/Differential_{E}quations_{N}otes/node3.html$ 1998


```
clear all
close all
clc

tic
%Parameters and Initial Conditions
G = 6.6743e-11;
m_1 = 1.9891e30; %mass of 1
m_2 = .330e24; %mass of Mercury
m_3 = 4.87e24; %mass of Venus
m_4 = 5.97e24;%Earth
m_5 = .642e24;%Mars
m_6 = 1898e24;%Jupiter
m_7 = 568e24;%Saturn
m_8 = 86.8e24;%Uranus
m_9 = 102e24;%Neptune
m_10 = .0146e24;%pluto

time = 7.92438e09; %how long to run simulation
%time = 5.93e7;
h = 10000; %time step

[position_mercury,velocity_mercury] = planetEphemeris(juliandate(2019,3,29),'Sun','Mercury');
[position_venus,velocity_venus] = planetEphemeris(juliandate(2019,3,29),'Sun','Venus');
[position_earth,velocity_earth] = planetEphemeris(juliandate(2019,3,29),'Sun','Earth');
[position_mars,velocity_mars] = planetEphemeris(juliandate(2019,3,29),'Sun','Mars');
[position_jupiter,velocity_jupiter] = planetEphemeris(juliandate(2019,3,29),'Sun','Jupiter');
[position_saturn,velocity_saturn] = planetEphemeris(juliandate(2019,3,29),'Sun','Saturn');
[position_uranus,velocity_uranus] = planetEphemeris(juliandate(2019,3,29),'Sun','Uranus');
[position_neptune,velocity_neptune] = planetEphemeris(juliandate(2019,3,29),'Sun','Neptune');
[position_pluto,velocity_pluto] = planetEphemeris(juliandate(2019,3,29),'Sun','Pluto');

%initial conditions
```

```
r_1 = [0;0;0]; %position Sun
r_2 =  position_mercury*10^3;%position Mercury
r_3 =  position_venus*10^3;%Venus
r_4 =  position_earth*10^3;%Earth
r_5 = position_mars*10^3;%Mars
r_6 = position_jupiter*10^3;%Jupiter
r_7 = position_saturn*10^3;%Saturn
r_8 = position_uranus*10^3;%Uranus
r_9 = position_neptune*10^3;%Neptune
r_10 = position_pluto*10^3;%Pluto


x_2 = r_2(1);
y_2 = r_2(2);
z_2 = r_2(3);

x_3 = r_3(1);
y_3 = r_3(2);
z_3 = r_3(3);

x_4 = r_4(1);
y_4 = r_4(2);
z_4 = r_4(3);

x_5 = r_5(1);
y_5 = r_5(2);
z_5 = r_5(3);

x_6 = r_6(1);
y_6 = r_6(2);
z_6 = r_6(3);

x_7 = r_7(1);
y_7 = r_7(2);
z_7 = r_7(3);

x_8 = r_8(1);
y_8 = r_8(2);
z_8 = r_8(3);

x_9 = r_9(1);
y_9 = r_9(2);
z_9 = r_9(3);
x_10 = r_10(1);
y_10 = r_10(2);
z_10 = r_10(3);

v_1 = [0;0;0];     %center body
v_2 = velocity_mercury*10^3;    %second body
v_3 = velocity_venus*10^3;
v_4 = velocity_earth*10^3;
v_5 = velocity_mars*10^3;
v_6 = velocity_jupiter*10^3;
v_7 = velocity_saturn*10^3;
```

```
v_8 = velocity_uranus*10^3;
v_9 = velocity_neptune*10^3;
v_10 = velocity_pluto*10^3;

r_2keep = zeros(3,time/h);
r_3keep = zeros(3,time/h);
r_4keep = zeros(3,time/h);
r_5keep = zeros(3,time/h);
r_6keep = zeros(3,time/h);
r_7keep = zeros(3,time/h);
r_8keep = zeros(3,time/h);
r_9keep = zeros(3,time/h);
r_10keep = zeros(3,time/h);

%Acceleration Body Mercury(2)
f2x = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(1) - r_2(1)) /
    norm(r_1-r_2) ^3 + ...
     m_3 * (r_3(1) - r_2(1)) / norm(r_3 - r_2) ^3 + ...
     m_4 * (r_4(1) - r_2(1)) / norm(r_4 - r_2) ^3 + ...
     m_5 * (r_5(1) - r_2(1)) / norm(r_5 - r_2) ^3 + ...
     m_6 * (r_6(1) - r_2(1)) / norm(r_6 - r_2) ^3 + ...
     m_7 * (r_7(1) - r_2(1)) / norm(r_7 - r_2) ^3 + ...
     m_8 * (r_8(1) - r_2(1)) / norm(r_8 - r_2) ^3 + ...
     m_9 * (r_9(1) - r_2(1)) / norm(r_9 - r_2) ^3 + ...
     m_10 * (r_10(1) - r_2(1)) / norm(r_10 - r_2) ^3);
f2y = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(2) - r_2(2)) /
    norm(r_1-r_2) ^3 + ...
     m_3 * (r_3(2) - r_2(2)) / norm(r_3 - r_2) ^3 + ...
     m_4 * (r_4(2) - r_2(2)) / norm(r_4 - r_2) ^3 + ...
     m_5 * (r_5(2) - r_2(2)) / norm(r_5 - r_2) ^3 + ...
     m_6 * (r_6(2) - r_2(2)) / norm(r_6 - r_2) ^3 + ...
     m_7 * (r_7(2) - r_2(2)) / norm(r_7 - r_2) ^3 + ...
     m_8 * (r_8(2) - r_2(2)) / norm(r_8 - r_2) ^3 + ...
     m_9 * (r_9(2) - r_2(2)) / norm(r_9 - r_2) ^3 + ...
     m_10 * (r_10(2) - r_2(2)) / norm(r_10 - r_2) ^3);
f2z = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(3) - r_2(3)) /
    norm(r_1-r_2) ^3 + ...
     m_3 * (r_3(3) - r_2(3)) / norm(r_3 - r_2) ^3 + ...
     m_4 * (r_4(3) - r_2(3)) / norm(r_4 - r_2) ^3 + ...
     m_5 * (r_5(3) - r_2(3)) / norm(r_5 - r_2) ^3 + ...
     m_6 * (r_6(3) - r_2(3)) / norm(r_6 - r_2) ^3 + ...
     m_7 * (r_7(3) - r_2(3)) / norm(r_7 - r_2) ^3 + ...
     m_8 * (r_8(3) - r_2(3)) / norm(r_8 - r_2) ^3 + ...
     m_9 * (r_9(3) - r_2(3)) / norm(r_9 - r_2) ^3 + ...
     m_10 * (r_10(3) - r_2(3)) / norm(r_10 - r_2) ^3);

%Acceleration Body Venus(3)
f3x = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_2 * (r_2(1) - r_3(1)) /
    norm(r_2-r_3) ^3 + ...
     m_1 * (r_1(1) - r_3(1)) / norm(r_1 - r_3) ^3 + ...
     m_4 * (r_4(1) - r_3(1)) / norm(r_4 - r_3) ^3 + ...
     m_5 * (r_5(1) - r_3(1)) / norm(r_5 - r_3) ^3 + ...
     m_6 * (r_6(1) - r_3(1)) / norm(r_6 - r_3) ^3 + ...
     m_7 * (r_7(1) - r_3(1)) / norm(r_7 - r_3) ^3 + ...
```

```
    m_8 * (r_8(1) - r_3(1)) / norm(r_8 - r_3) ^3 + ...
    m_9 * (r_9(1) - r_3(1)) / norm(r_9 - r_3) ^3 + ...
    m_10 * (r_10(1) - r_3(1)) / norm(r_10 - r_3) ^3);
f3y = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_2 * (r_2(2) - r_3(2)) / norm(r_2-r_3) ^3 + ...
    m_1 * (r_1(2) - r_3(2)) / norm(r_1 - r_3) ^3 + ...
    m_4 * (r_4(2) - r_3(2)) / norm(r_4 - r_3) ^3 + ...
    m_5 * (r_5(2) - r_3(2)) / norm(r_5 - r_3) ^3 + ...
    m_6 * (r_6(2) - r_3(2)) / norm(r_6 - r_3) ^3 + ...
    m_7 * (r_7(2) - r_3(2)) / norm(r_7 - r_3) ^3 + ...
    m_8 * (r_8(2) - r_3(2)) / norm(r_8 - r_3) ^3 + ...
    m_9 * (r_9(2) - r_3(2)) / norm(r_9 - r_3) ^3 + ...
    m_10 * (r_10(2) - r_3(2)) / norm(r_10 - r_3) ^3);
f3z = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_2 * (r_2(3) - r_3(3)) / norm(r_2-r_3) ^3 + ...
    m_1 * (r_1(3) - r_3(3)) / norm(r_1 - r_3) ^3 + ...
    m_4 * (r_4(3) - r_3(3)) / norm(r_4 - r_3) ^3 + ...
    m_5 * (r_5(3) - r_3(3)) / norm(r_5 - r_3) ^3 + ...
    m_6 * (r_6(3) - r_3(3)) / norm(r_6 - r_3) ^3 + ...
    m_7 * (r_7(3) - r_3(3)) / norm(r_7 - r_3) ^3 + ...
    m_8 * (r_8(3) - r_3(3)) / norm(r_8 - r_3) ^3 + ...
    m_9 * (r_9(3) - r_3(3)) / norm(r_9 - r_3) ^3 + ...
    m_10 * (r_10(3) - r_3(3)) / norm(r_10 - r_3) ^3);

%Acceleration Body Earth(4)
f4x = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(1) - r_4(1)) / norm(r_1 - r_4) ^3 + ...
    m_2 * (r_2(1) - r_4(1)) / norm(r_2 - r_4) ^3 + ...
    m_3 * (r_3(1) - r_4(1)) / norm(r_3 - r_4) ^3 + ...
    m_5 * (r_5(1) - r_4(1)) / norm(r_5 - r_4) ^3 + ...
    m_6 * (r_6(1) - r_4(1)) / norm(r_6 - r_4) ^3 + ...
    m_7 * (r_7(1) - r_4(1)) / norm(r_7 - r_4) ^3 + ...
    m_8 * (r_8(1) - r_4(1)) / norm(r_8 - r_4) ^3 + ...
    m_9 * (r_9(1) - r_4(1)) / norm(r_9 - r_4) ^3 + ...
    m_10 * (r_10(1) - r_4(1)) / norm(r_10 - r_4) ^3);
f4y = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(2) - r_4(2)) / norm(r_1 - r_4) ^3 + ...
    m_2 * (r_2(2) - r_4(2)) / norm(r_2 - r_4) ^3 + ...
    m_3 * (r_3(2) - r_4(2)) / norm(r_3 - r_4) ^3 + ...
    m_5 * (r_5(2) - r_4(2)) / norm(r_5 - r_4) ^3 + ...
    m_6 * (r_6(2) - r_4(2)) / norm(r_6 - r_4) ^3 + ...
    m_7 * (r_7(2) - r_4(2)) / norm(r_7 - r_4) ^3 + ...
    m_8 * (r_8(2) - r_4(2)) / norm(r_8 - r_4) ^3 + ...
    m_9 * (r_9(2) - r_4(2)) / norm(r_9 - r_4) ^3 + ...
    m_10 * (r_10(2) - r_4(2)) / norm(r_10 - r_4) ^3);
f4z = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(3) - r_4(3)) / norm(r_1 - r_4) ^3 + ...
    m_2 * (r_2(3) - r_4(3)) / norm(r_2 - r_4) ^3 + ...
    m_3 * (r_3(3) - r_4(3)) / norm(r_3 - r_4) ^3 + ...
    m_5 * (r_5(3) - r_4(3)) / norm(r_5 - r_4) ^3 + ...
    m_6 * (r_6(3) - r_4(3)) / norm(r_6 - r_4) ^3 + ...
    m_7 * (r_7(3) - r_4(3)) / norm(r_7 - r_4) ^3 + ...
    m_8 * (r_8(3) - r_4(3)) / norm(r_8 - r_4) ^3 + ...
    m_9 * (r_9(3) - r_4(3)) / norm(r_9 - r_4) ^3 + ...
```

```
    m_10 * (r_10(3) - r_4(3)) / norm(r_10 - r_4) ^3);

%Acceleration Body Mars(5)
f5x = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(1) - r_5(1)) / norm(r_1-r_5) ^3 + ...
    m_2 * (r_2(1) - r_5(1)) / norm(r_2 - r_5) ^3 + ...
    m_3 * (r_3(1) - r_5(1)) / norm(r_3 - r_5) ^3 + ...
    m_4 * (r_4(1) - r_5(1)) / norm(r_4 - r_5) ^3 + ...
    m_6 * (r_6(1) - r_5(1)) / norm(r_6 - r_5) ^3 + ...
    m_7 * (r_7(1) - r_5(1)) / norm(r_7 - r_5) ^3 + ...
    m_8 * (r_8(1) - r_5(1)) / norm(r_8 - r_5) ^3 + ...
    m_9 * (r_9(1) - r_5(1)) / norm(r_9 - r_5) ^3 + ...
    m_10 * (r_10(1) - r_5(1)) / norm(r_10 - r_5) ^3);
f5y = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(2) - r_5(2)) / norm(r_1-r_5) ^3 + ...
    m_2 * (r_2(2) - r_5(2)) / norm(r_2 - r_5) ^3 + ...
    m_3 * (r_3(2) - r_5(2)) / norm(r_3 - r_5) ^3 + ...
    m_4 * (r_4(2) - r_5(2)) / norm(r_4 - r_5) ^3 + ...
    m_6 * (r_6(2) - r_5(2)) / norm(r_6 - r_5) ^3 + ...
    m_7 * (r_7(2) - r_5(2)) / norm(r_7 - r_5) ^3 + ...
    m_8 * (r_8(2) - r_5(2)) / norm(r_8 - r_5) ^3 + ...
    m_9 * (r_9(2) - r_5(2)) / norm(r_9 - r_5) ^3 + ...
    m_10 * (r_10(2) - r_5(2)) / norm(r_10 - r_5) ^3);
f5z = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(3) - r_5(3)) / norm(r_1-r_5) ^3 + ...
    m_2 * (r_2(3) - r_5(3)) / norm(r_2 - r_5) ^3 + ...
    m_3 * (r_3(3) - r_5(3)) / norm(r_3 - r_5) ^3 + ...
    m_4 * (r_4(3) - r_5(3)) / norm(r_4 - r_5) ^3 + ...
    m_6 * (r_6(3) - r_5(3)) / norm(r_6 - r_5) ^3 + ...
    m_7 * (r_7(3) - r_5(3)) / norm(r_7 - r_5) ^3 + ...
    m_8 * (r_8(3) - r_5(3)) / norm(r_8 - r_5) ^3 + ...
    m_9 * (r_9(3) - r_5(3)) / norm(r_9 - r_5) ^3 + ...
    m_10 * (r_10(3) - r_5(3)) / norm(r_10 - r_5) ^3);

%Acceleration Body Jupiter(6)
f6x = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(1) - r_6(1)) / norm(r_1-r_6) ^3 + ...
    m_2 * (r_2(1) - r_6(1)) / norm(r_2 - r_6) ^3 + ...
    m_3 * (r_3(1) - r_6(1)) / norm(r_3 - r_6) ^3 + ...
    m_4 * (r_4(1) - r_6(1)) / norm(r_4 - r_6) ^3 + ...
    m_5 * (r_5(1) - r_6(1)) / norm(r_5 - r_6) ^3 + ...
    m_7 * (r_7(1) - r_6(1)) / norm(r_7 - r_6) ^3 + ...
    m_8 * (r_8(1) - r_6(1)) / norm(r_8 - r_6) ^3 + ...
    m_9 * (r_9(1) - r_6(1)) / norm(r_9 - r_6) ^3 + ...
    m_10 * (r_10(1) - r_6(1)) / norm(r_10 - r_6) ^3);
f6y = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(2) - r_6(2)) / norm(r_1-r_6) ^3 + ...
    m_2 * (r_2(2) - r_6(2)) / norm(r_2 - r_6) ^3 + ...
    m_3 * (r_3(2) - r_6(2)) / norm(r_3 - r_6) ^3 + ...
    m_4 * (r_4(2) - r_6(2)) / norm(r_4 - r_6) ^3 + ...
    m_5 * (r_5(2) - r_6(2)) / norm(r_5 - r_6) ^3 + ...
    m_7 * (r_7(2) - r_6(2)) / norm(r_7 - r_6) ^3 + ...
    m_8 * (r_8(2) - r_6(2)) / norm(r_8 - r_6) ^3 + ...
    m_9 * (r_9(2) - r_6(2)) / norm(r_9 - r_6) ^3 + ...
```

```
    m_10 * (r_10(2) - r_6(2)) / norm(r_10 - r_6)^3);
f6z = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(3) - r_6(3)) / norm(r_1-r_6)^3 + ...
    m_2 * (r_2(3) - r_6(3)) / norm(r_2 - r_6)^3 + ...
    m_3 * (r_3(3) - r_6(3)) / norm(r_3 - r_6)^3 + ...
    m_4 * (r_4(3) - r_6(3)) / norm(r_4 - r_6)^3 + ...
    m_5 * (r_5(3) - r_6(3)) / norm(r_5 - r_6)^3 + ...
    m_7 * (r_7(3) - r_6(3)) / norm(r_7 - r_6)^3 + ...
    m_8 * (r_8(3) - r_6(3)) / norm(r_8 - r_6)^3 + ...
    m_9 * (r_9(3) - r_6(3)) / norm(r_9 - r_6)^3 + ...
    m_10 * (r_10(3) - r_6(3)) / norm(r_10 - r_6)^3);

%Acceleration Body Saturn(7)
f7x = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(1) - r_7(1)) / norm(r_1-r_7)^3 + ...
    m_2 * (r_2(1) - r_7(1)) / norm(r_2 - r_7)^3 + ...
    m_3 * (r_3(1) - r_7(1)) / norm(r_3 - r_7)^3 + ...
    m_4 * (r_4(1) - r_7(1)) / norm(r_4 - r_7)^3 + ...
    m_5 * (r_5(1) - r_7(1)) / norm(r_5 - r_7)^3 + ...
    m_6 * (r_6(1) - r_7(1)) / norm(r_6 - r_7)^3 + ...
    m_8 * (r_8(1) - r_7(1)) / norm(r_8 - r_7)^3 + ...
    m_9 * (r_9(1) - r_7(1)) / norm(r_9 - r_7)^3 + ...
    m_10 * (r_10(1) - r_7(1)) / norm(r_10 - r_7)^3);
f7y = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(2) - r_7(2)) / norm(r_1-r_7)^3 + ...
    m_2 * (r_2(2) - r_7(2)) / norm(r_2 - r_7)^3 + ...
    m_3 * (r_3(2) - r_7(2)) / norm(r_3 - r_7)^3 + ...
    m_4 * (r_4(2) - r_7(2)) / norm(r_4 - r_7)^3 + ...
    m_5 * (r_5(2) - r_7(2)) / norm(r_5 - r_7)^3 + ...
    m_6 * (r_6(2) - r_7(2)) / norm(r_6 - r_7)^3 + ...
    m_8 * (r_8(2) - r_7(2)) / norm(r_8 - r_7)^3 + ...
    m_9 * (r_9(2) - r_7(2)) / norm(r_9 - r_7)^3 + ...
    m_10 * (r_10(2) - r_7(2)) / norm(r_10 - r_7)^3);
f7z = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(3) - r_7(3)) / norm(r_1-r_7)^3 + ...
    m_2 * (r_2(3) - r_7(3)) / norm(r_2 - r_7)^3 + ...
    m_3 * (r_3(3) - r_7(3)) / norm(r_3 - r_7)^3 + ...
    m_4 * (r_4(3) - r_7(3)) / norm(r_4 - r_7)^3 + ...
    m_5 * (r_5(3) - r_7(3)) / norm(r_5 - r_7)^3 + ...
    m_6 * (r_6(3) - r_7(3)) / norm(r_6 - r_7)^3 + ...
    m_8 * (r_8(3) - r_7(3)) / norm(r_8 - r_7)^3 + ...
    m_9 * (r_9(3) - r_7(3)) / norm(r_9 - r_7)^3 + ...
    m_10 * (r_10(3) - r_7(3)) / norm(r_10 - r_7)^3);

%Acceleration Body Uranus(8)
f8x = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(1) - r_8(1)) / norm(r_1-r_8)^3 + ...
    m_2 * (r_2(1) - r_8(1)) / norm(r_2 - r_8)^3 + ...
    m_3 * (r_3(1) - r_8(1)) / norm(r_3 - r_8)^3 + ...
    m_4 * (r_4(1) - r_8(1)) / norm(r_4 - r_8)^3 + ...
    m_5 * (r_5(1) - r_8(1)) / norm(r_5 - r_8)^3 + ...
    m_6 * (r_6(1) - r_8(1)) / norm(r_6 - r_8)^3 + ...
    m_7 * (r_7(1) - r_8(1)) / norm(r_7 - r_8)^3 + ...
    m_9 * (r_9(1) - r_8(1)) / norm(r_9 - r_8)^3 + ...
```

```
    m_10 * (r_10(1) - r_8(1)) / norm(r_10 - r_8)^3);
f8y = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(2) - r_8(2)) / norm(r_1-r_8)^3 + ...
    m_2 * (r_2(2) - r_8(2)) / norm(r_2 - r_8)^3 + ...
    m_3 * (r_3(2) - r_8(2)) / norm(r_3 - r_8)^3 + ...
    m_4 * (r_4(2) - r_8(2)) / norm(r_4 - r_8)^3 + ...
    m_5 * (r_5(2) - r_8(2)) / norm(r_5 - r_8)^3 + ...
    m_6 * (r_6(2) - r_8(2)) / norm(r_6 - r_8)^3 + ...
    m_7 * (r_7(2) - r_8(2)) / norm(r_7 - r_8)^3 + ...
    m_9 * (r_9(2) - r_8(2)) / norm(r_9 - r_8)^3 + ...
    m_10 * (r_10(2) - r_8(2)) / norm(r_10 - r_8)^3);
f8z = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(3) - r_8(3)) / norm(r_1-r_8)^3 + ...
    m_2 * (r_2(3) - r_8(3)) / norm(r_2 - r_8)^3 + ...
    m_3 * (r_3(3) - r_8(3)) / norm(r_3 - r_8)^3 + ...
    m_4 * (r_4(3) - r_8(3)) / norm(r_4 - r_8)^3 + ...
    m_5 * (r_5(3) - r_8(3)) / norm(r_5 - r_8)^3 + ...
    m_6 * (r_6(3) - r_8(3)) / norm(r_6 - r_8)^3 + ...
    m_7 * (r_7(3) - r_8(3)) / norm(r_7 - r_8)^3 + ...
    m_9 * (r_9(3) - r_8(3)) / norm(r_9 - r_8)^3 + ...
    m_10 * (r_10(3) - r_8(3)) / norm(r_10 - r_8)^3);

%Acceleration Body Neptune(9)
f9x = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(1) - r_9(1)) / norm(r_1-r_9)^3 + ...
    m_2 * (r_2(1) - r_9(1)) / norm(r_2 - r_9)^3 + ...
    m_3 * (r_3(1) - r_9(1)) / norm(r_3 - r_9)^3 + ...
    m_4 * (r_4(1) - r_9(1)) / norm(r_4 - r_9)^3 + ...
    m_5 * (r_5(1) - r_9(1)) / norm(r_5 - r_9)^3 + ...
    m_6 * (r_6(1) - r_9(1)) / norm(r_6 - r_9)^3 + ...
    m_7 * (r_7(1) - r_9(1)) / norm(r_7 - r_9)^3 + ...
    m_8 * (r_8(1) - r_9(1)) / norm(r_8 - r_9)^3 + ...
    m_10 * (r_10(1) - r_9(1)) / norm(r_10 - r_9)^3);
f9y = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(2) - r_9(2)) / norm(r_1-r_9)^3 + ...
    m_2 * (r_2(2) - r_9(2)) / norm(r_2 - r_9)^3 + ...
    m_3 * (r_3(2) - r_9(2)) / norm(r_3 - r_9)^3 + ...
    m_4 * (r_4(2) - r_9(2)) / norm(r_4 - r_9)^3 + ...
    m_5 * (r_5(2) - r_9(2)) / norm(r_5 - r_9)^3 + ...
    m_6 * (r_6(2) - r_9(2)) / norm(r_6 - r_9)^3 + ...
    m_7 * (r_7(2) - r_9(2)) / norm(r_7 - r_9)^3 + ...
    m_8 * (r_8(2) - r_9(2)) / norm(r_8 - r_9)^3 + ...
    m_10 * (r_10(2) - r_9(2)) / norm(r_10 - r_9)^3);
f9z = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(3) - r_9(3)) / norm(r_1-r_9)^3 + ...
    m_2 * (r_2(3) - r_9(3)) / norm(r_2 - r_9)^3 + ...
    m_3 * (r_3(3) - r_9(3)) / norm(r_3 - r_9)^3 + ...
    m_4 * (r_4(3) - r_9(3)) / norm(r_4 - r_9)^3 + ...
    m_5 * (r_5(3) - r_9(3)) / norm(r_5 - r_9)^3 + ...
    m_6 * (r_6(3) - r_9(3)) / norm(r_6 - r_9)^3 + ...
    m_7 * (r_7(3) - r_9(3)) / norm(r_7 - r_9)^3 + ...
    m_8 * (r_8(3) - r_9(3)) / norm(r_8 - r_9)^3 + ...
    m_10 * (r_10(3) - r_9(3)) / norm(r_10 - r_9)^3);
```

```
%Acceleration Body Pluto(10)
f10x = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(1) - r_10(1)) /
    norm(r_1-r_10) ^3 + ...
    m_2 * (r_2(1) - r_10(1)) / norm(r_2 - r_10) ^3 + ...
    m_3 * (r_3(1) - r_10(1)) / norm(r_3 - r_10) ^3 + ...
    m_4 * (r_4(1) - r_10(1)) / norm(r_4 - r_10) ^3 + ...
    m_5 * (r_5(1) - r_10(1)) / norm(r_5 - r_10) ^3 + ...
    m_6 * (r_6(1) - r_10(1)) / norm(r_6 - r_10) ^3 + ...
    m_7 * (r_7(1) - r_10(1)) / norm(r_7 - r_10) ^3 + ...
    m_8 * (r_8(1) - r_10(1)) / norm(r_8 - r_10) ^3 + ...
    m_9 * (r_9(1) - r_10(1)) / norm(r_9 - r_10) ^3);
f10y = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(2) - r_10(2)) /
    norm(r_1-r_10) ^3 + ...
    m_2 * (r_2(2) - r_10(2)) / norm(r_2 - r_10) ^3 + ...
    m_3 * (r_3(2) - r_10(2)) / norm(r_3 - r_10) ^3 + ...
    m_4 * (r_4(2) - r_10(2)) / norm(r_4 - r_10) ^3 + ...
    m_5 * (r_5(2) - r_10(2)) / norm(r_5 - r_10) ^3 + ...
    m_6 * (r_6(2) - r_10(2)) / norm(r_6 - r_10) ^3 + ...
    m_7 * (r_7(2) - r_10(2)) / norm(r_7 - r_10) ^3 + ...
    m_8 * (r_8(2) - r_10(2)) / norm(r_8 - r_10) ^3 + ...
    m_9 * (r_9(2) - r_10(2)) / norm(r_9 - r_10) ^3);
f10z = @(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2) G * (m_1 * (r_1(3) - r_10(3)) /
    norm(r_1-r_10) ^3 + ...
    m_2 * (r_2(3) - r_10(3)) / norm(r_2 - r_10) ^3 + ...
    m_3 * (r_3(3) - r_10(3)) / norm(r_3 - r_10) ^3 + ...
    m_4 * (r_4(3) - r_10(3)) / norm(r_4 - r_10) ^3 + ...
    m_5 * (r_5(3) - r_10(3)) / norm(r_5 - r_10) ^3 + ...
    m_6 * (r_6(3) - r_10(3)) / norm(r_6 - r_10) ^3 + ...
    m_7 * (r_7(3) - r_10(3)) / norm(r_7 - r_10) ^3 + ...
    m_8 * (r_8(3) - r_10(3)) / norm(r_8 - r_10) ^3 + ...
    m_9 * (r_9(3) - r_10(3)) / norm(r_9 - r_10) ^3);


v2x = v_2(1);
v2y = v_2(2);
v2z = v_2(3);
a2x = f2x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a2y = f2y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a2z = f2z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

v3x = v_3(1);
v3y = v_3(2);
v3z = v_3(3);
a3x = f3x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a3y = f3y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a3z = f3z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

v4x = v_4(1);
v4y = v_4(2);
v4z = v_4(3);
a4x = f4x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a4y = f4y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a4z = f4z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
```

```
v5x = v_5(1);
v5y = v_5(2);
v5z = v_5(3);
a5x = f5x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a5y = f5y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a5z = f5z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

v6x = v_6(1);
v6y = v_6(2);
v6z = v_6(3);
a6x = f6x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a6y = f6y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a6z = f6z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

v7x = v_7(1);
v7y = v_7(2);
v7z = v_7(3);
a7x = f7x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a7y = f7y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a7z = f7z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

v8x = v_8(1);
v8y = v_8(2);
v8z = v_8(3);
a8x = f8x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a8y = f8y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a8z = f8z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

v9x = v_9(1);
v9y = v_9(2);
v9z = v_9(3);
a9x = f9x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a9y = f9y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a9z = f9z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

v10x = v_10(1);
v10y = v_10(2);
v10z = v_10(3);
a10x = f10x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a10y = f10y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
a10z = f10z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

for j = 1:time/h
    %Mercury
    v2x = v2x + h*a2x;
    v2y = v2y + h*a2y;
    v2z = v2z + h*a2z;

    r2x = x_2 + h*v2x;
    r2y = y_2 + h*v2y;
    r2z = z_2 + h*v2z;

    r_2 = [r2x;r2y;r2z];
    x_2 = r_2(1);
```

```
    y_2 = r_2(2);
    z_2 = r_2(3);


    %Venus
    v3x = v3x + h*a3x;
    v3y = v3y + h*a3y;
    v3z = v3z + h*a3z;

    r3x = x_3 + h*v3x;
    r3y = y_3 + h*v3y;
    r3z = z_3 + h*v3z;

    r_3 = [r3x;r3y;r3z];
    x_3 = r_3(1);
    y_3 = r_3(2);
    z_3 = r_3(3);

    %Earth
    v4x = v4x + h*a4x;
    v4y = v4y + h*a4y;
    v4z = v4z + h*a4z;

    r4x = x_4 + h*v4x;
    r4y = y_4 + h*v4y;
    r4z = z_4 + h*v4z;

    r_4 = [r4x;r4y;r4z];
    x_4 = r_4(1);
    y_4 = r_4(2);
    z_4 = r_4(3);

    %Mars
    v5x = v5x + h*a5x;
    v5y = v5y + h*a5y;
    v5z = v5z + h*a5z;

    r5x = x_5 + h*v5x;
    r5y = y_5 + h*v5y;
    r5z = z_5 + h*v5z;

    r_5 = [r5x;r5y;r5z];
    x_5 = r_5(1);
    y_5 = r_5(2);
    z_5 = r_5(3);

    %Jupiter
    v6x = v6x + h*a6x;
    v6y = v6y + h*a6y;
    v6z = v6z + h*a6z;

    r6x = x_6 + h*v6x;
    r6y = y_6 + h*v6y;
    r6z = z_6 + h*v6z;
```

```
r_6 = [r6x;r6y;r6z];
x_6 = r_6(1);
y_6 = r_6(2);
z_6 = r_6(3);

%Saturn
v7x = v7x + h*a7x;
v7y = v7y + h*a7y;
v7z = v7z + h*a7z;

r7x = x_7 + h*v7x;
r7y = y_7 + h*v7y;
r7z = z_7 + h*v7z;

r_7 = [r7x;r7y;r7z];
x_7 = r_7(1);
y_7 = r_7(2);
z_7 = r_7(3);

%Uranus
v8x = v8x + h*a8x;
v8y = v8y + h*a8y;
v8z = v8z + h*a8z;

r8x = x_8 + h*v8x;
r8y = y_8 + h*v8y;
r8z = z_8 + h*v8z;

r_8 = [r8x;r8y;r8z];
x_8 = r_8(1);
y_8 = r_8(2);
z_8 = r_8(3);

%Neptune
v9x = v9x + h*a9x;
v9y = v9y + h*a9y;
v9z = v9z + h*a9z;

r9x = x_9 + h*v9x;
r9y = y_9 + h*v9y;
r9z = z_9 + h*v9z;

r_9 = [r9x;r9y;r9z];
x_9 = r_9(1);
y_9 = r_9(2);
z_9 = r_9(3);

%Pluto
v10x = v10x + h*a10x;
v10y = v10y + h*a10y;
v10z = v10z + h*a10z;

r10x = x_10 + h*v10x;
```

```
    r10y = y_10 + h*v10y;
    r10z = z_10 + h*v10z;

    r_10 = [r10x;r10y;r10z];
    x_10 = r_10(1);
    y_10 = r_10(2);
    z_10 = r_10(3);


    %Update Accelerations
    a2x = f2x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a2y = f2y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a2z = f2z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

    a3x = f3x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a3y = f3y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a3z = f3z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

    a4x = f4x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a4y = f4y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a4z = f4z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

    a5x = f5x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a5y = f5y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a5z = f5z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

    a6x = f6x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a6y = f6y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a6z = f6z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

    a7x = f7x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a7y = f7y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a7z = f7z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

    a8x = f8x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a8y = f8y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a8z = f8z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

    a9x = f9x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a9y = f9y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a9z = f9z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

    a10x = f10x(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a10y = f10y(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);
    a10z = f10z(r_10,r_9,r_8,r_7,r_6,r_5,r_4,r_3,r_2);

    r_2keep(:,j) = r_2;
    r_3keep(:,j) = r_3;
    r_4keep(:,j) = r_4;
    r_5keep(:,j) = r_5;
    r_6keep(:,j) = r_6;
    r_7keep(:,j) = r_7;
    r_8keep(:,j) = r_8;
    r_9keep(:,j) = r_9;
```

```
    r_10keep(:,j) = r_10;
end

toc

figure(1)
plot3(0,0,0,'*')
hold on
plot3(r_2keep(1,:),r_2keep(2,:),r_2keep(3,:),'LineWidth',1)
hold on
plot3(r_3keep(1,:),r_3keep(2,:),r_3keep(3,:),'LineWidth',1)
hold on
plot3(r_4keep(1,:),r_4keep(2,:),r_4keep(3,:),'LineWidth',1)
hold on
plot3(r_5keep(1,:),r_5keep(2,:),r_5keep(3,:),'LineWidth',1)
hold on
plot3(r_6keep(1,:),r_6keep(2,:),r_6keep(3,:),'LineWidth',1)
hold on
plot3(r_7keep(1,:),r_7keep(2,:),r_7keep(3,:),'LineWidth',1)
hold on
plot3(r_8keep(1,:),r_8keep(2,:),r_8keep(3,:),'LineWidth',1)
hold on
plot3(r_9keep(1,:),r_9keep(2,:),r_9keep(3,:),'LineWidth',1)
hold on
plot3(r_10keep(1,:),r_10keep(2,:),r_10keep(3,:),'LineWidth',1)
grid on
legend('Sun','Mercury','Venus','Earth','Mars','Jupiter','Saturn','Uranus','
    Pluto')
title('Solar System','Fontsize',20)
xlabel('Meters','Fontsize',16)
ylabel('Meters','Fontsize',16)
zlabel('Meters','Fontsize',16)
set(gcf, 'PaperPositionMode', 'manual')
set(gcf, 'Color', [1 1 1])
set(gca, 'Color', [1 1 1])
set(gcf, 'PaperUnits', 'centimeters')
set(gcf, 'PaperSize', [30 20])
set(gcf, 'Units', 'centimeters' )
set(gcf, 'Position', [0 0 30 20])
set(gcf, 'PaperPosition', [0 0 30 20])
saveas(figure(1),'SolarSystemnormal.png')

figure(2)
plot3(0,0,0,'*')
hold on
plot3(r_2keep(1,:),r_2keep(2,:),r_2keep(3,:),'LineWidth',1)
hold on
plot3(r_3keep(1,:),r_3keep(2,:),r_3keep(3,:),'LineWidth',1)
hold on
plot3(r_4keep(1,:),r_4keep(2,:),r_4keep(3,:),'LineWidth',1)
hold on
plot3(r_5keep(1,:),r_5keep(2,:),r_5keep(3,:),'LineWidth',1)
grid on
legend('Sun','Mercury','Venus','Earth','Mars','Fontsize',10)
```

```
title('Inner Planets','Fontsize',20)
xlabel('Meters','Fontsize',16)
ylabel('Meters','Fontsize',16)
zlabel('Meters','Fontsize',16)
set(gcf, 'PaperPositionMode', 'manual')
set(gcf, 'Color', [1 1 1])
set(gca, 'Color', [1 1 1])
set(gcf, 'PaperUnits', 'centimeters')
set(gcf, 'PaperSize', [30 20])
set(gcf, 'Units', 'centimeters' )
set(gcf, 'Position', [0 0 30 20])
set(gcf, 'PaperPosition', [0 0 30 20])
saveas(figure(2),'innerPlanets.png')

figure(3)
plot3(0,0,0,'*')
hold on
plot3(r_6keep(1,:),r_6keep(2,:),r_6keep(3,:),'LineWidth',1)
hold on
plot3(r_7keep(1,:),r_7keep(2,:),r_7keep(3,:),'LineWidth',1)
hold on
plot3(r_8keep(1,:),r_8keep(2,:),r_8keep(3,:),'LineWidth',1)
hold on
plot3(r_9keep(1,:),r_9keep(2,:),r_9keep(3,:),'LineWidth',1)
hold on
plot3(r_10keep(1,:),r_10keep(2,:),r_10keep(3,:),'LineWidth',1)
grid on
legend('Sun','Jupiter','Saturn','Uranus','Neptune','Pluto')
title('Outer Planets','Fontsize',20)
xlabel('Meters','Fontsize',16)
ylabel('Meters','Fontsize',16)
zlabel('Meters','Fontsize',16)
set(gcf, 'PaperPositionMode', 'manual')
set(gcf, 'Color', [1 1 1])
set(gca, 'Color', [1 1 1])
set(gcf, 'PaperUnits', 'centimeters')
set(gcf, 'PaperSize', [30 20])
set(gcf, 'Units', 'centimeters' )
set(gcf, 'Position', [0 0 30 20])
set(gcf, 'PaperPosition', [0 0 30 20])
saveas(figure(3),'outerPlanets.png')
```