Motion of Artificial Satellites around Earth

3 Body Problem

Sanjana Gupta

PHY-1110-1

Mathematical Physics 1:
Mathematical and Computational Toolkit

Professor Vikram Vyas

May 1, 2022

Contents

1 Introduction

1.1 Earth and Artificial Satellites

A satellite is an object that orbits another object. In space, satellites may be natural, or artificial. The moon is a natural satellite that orbits the Earth. An artificial satllite is an electronic device that is sent into space and moves around the earth or another planet for a particular purpose. Most artificial satellites orbit Earth. On 4 October 1957, the Soviet Union launched the world's first artificial satellite, Sputnik 1. Since then, about 8,900 artificial satellites from more than 40 countries have been launched. According to a 2018 estimate, about 5,000 remained in orbit. Of those, about 1,900 were operational, while the rest had exceeded their useful lives and become space debris. Artificial satellites are used for many purposes. Among several other applications, they can be used to make star maps and maps of planetary surfaces, and also take pictures of planets they are launched into. Common types include military and civilian Earth observation satellites, communications satellites, navigation satellites, weather satellites, and space telescopes. Space stations and human spacecraft in orbit are also satellites.

1.2 Three-Body Problem

The three-body problem in classical mechanics is the problem of taking three point masses' initial positions and velocities (or momenta) and solving for their future motion using Newton's equations of motion and Newton's law of universal gravitation. A specific case of the n-body problem is the three-body problem. Unlike two-body problems, there is no general closed-form solution since the emerging dynamical system is chaotic for most initial conditions, however programming languages like Python can be used to visualise the motion of the bodies.

2 Aim

To solve for the motion of more than two bodies under the influence of their mutual gravitational by using their initial conditions and other physical parameters in the VPython module of Python.

3 Scenarios simulated

3.1 Problem 1: Simulating the motion of artificial satellites having different mass

I attempted to model the motion of two satellites orbiting earth with vastly differing masses. The mass of one satellite is significantly greater than the mass of the other. The satellites are considered to be at a huge distance from Earth, in the order of million kilometres. The radii of the satellites are not to scale, so that they are large enough to be seen in the window. Because the object's radius here has no influence on the physics involved, using a larger radius is not a problem. The Earth is placed at the centre, and the original values for mass and radius of the Earth are used. The first satellite's velocity is assumed to be 1022 metres per second, while the second satellite's velocity is assumed to be 1029 metres per second. The first satellite's mass is estimated to be 734800 kilograms, while the mass of the second satellite is 2000 kilograms. I simulate the motion of these satellites to see how the bodies' trajectories vary due to differences in mass.

3.2 Problem 2: Simulating the motion of two artificial satellites around Earth having same mass

In this, I attempted to model the motion of two satellites orbiting earth having the same mass. The satellites are considered to be at a huge distance from Earth, in the order of 10^5 kilometres. The radii of the satellites are not to scale, so that they are large enough to be seen in

the window. Because the object's radius here has no influence on the physics involved, using a larger radius is not a problem. The Earth is placed at the centre, and the original values for mass of the Earth is used. The first satellite's velocity is assumed to be 1224 metres per second, while the second satellite's velocity is assumed to be 1029 metres per second. Masses of both the satellites are taken to be 100000 kilograms which is around $\frac{1}{4}$ of the mass of the International Space Station. The simulation mainly focuses on how the difference in initial velocities and positions of the satellites

3.3 Problem 3: Simulating the motion of the same artificial satellites using Leapfrog method

The initial conditions used for the satellites orbiting Earth are the same as the ones used in Problem 2. The only difference is that we are using Leapfrog method to solve the problem in order to reduce the error.

4 Methodology

4.1 Theoretical Background

4.1.1 Equations of Motion

The N body problem can be written as follows:

$$\ddot{\mathbf{r}}_i(t) = \frac{d^2\mathbf{r}_i}{dt^2} = -G \sum_{j=1, j\neq i}^{N} \frac{m_j}{|\mathbf{r}_i - \mathbf{r}_j|^3} (\mathbf{r}_i - \mathbf{r}_j)$$

where $\mathbf{r}_i$ is the position vector of the i^{th} body with respect to suitably chosen origin, m_i is the mass of the body and G is the Gravitational constant. The sum on the left hand side is over all the bodies except the i^{th} body whose acceleration we are evaluating. The label i takes value from 1 to N, where N is the total number of bodies that form our

system. Therefore we have to solve $3N$ coupled differential equations to obtain the future and the past of our solar system.

The three-body problem is a special case of the N body problem and its mathematical statement can be given in terms of the Newtonian equations of motion for vector positions $\mathbf{r_i} = (x_i, y_i, z_i)$ of three gravitationally interacting bodies with masses m_i :

$$\ddot{\mathbf{r}}_1 = -Gm_2\frac{\mathbf{r}_1 - \mathbf{r_2}}{|\mathbf{r}_1 - \mathbf{r}_2|^3} - Gm_3\frac{\mathbf{r}_1 - \mathbf{r}_3}{|\mathbf{r}_1 - \mathbf{r}_3|^3}$$
$$\ddot{\mathbf{r}}_2 = -Gm_3\frac{\mathbf{r}_2 - \mathbf{r}_3}{|\mathbf{r}_2 - \mathbf{r}_3|^3} - Gm_1\frac{\mathbf{r}_2 - \mathbf{r}_1}{|\mathbf{r}_2 - \mathbf{r}_1|^3}$$
$$\ddot{\mathbf{r}}_3 = -Gm_1\frac{\mathbf{r}_3 - \mathbf{r}_1}{|\mathbf{r}_3 - \mathbf{r}_1|^3} - Gm_2\frac{\mathbf{r}_3 - \mathbf{r}_2}{|\mathbf{r}_3 - \mathbf{r}_2|^3}$$

where G is the Universal Gravitational constant.
The above equations describe the second derivative of position, or acceleration and are a set of nine second-order differential equations.

4.1.2 Conserving Angular Momentum

The law of conservation of angular momentum states that when no external torque acts on an object, no change of angular momentum will occur. To simulate a stable system, we ensure that the total angular momentum is conserved. To conserve angular momentum, we write a for loop in python, which adds up the momenta of the bodies one by one, and equates it to zero.
$\vec{\tau} = \frac{\mathrm{d}\vec{\mathrm{L}}}{\mathrm{dt}}$, where τ is the torque.
For the situation in which the net torque is zero, $\frac{\mathrm{dL}}{\mathrm{dt}} = 0$.
If the change in angular momentum ΔL is zero, then the angular momentum is constant
$\therefore \vec{\mathrm{L}}$ = constant (when net T = 0).

4.2 Leapfrog Method

In numerical analysis, leapfrog integration is a method for numerically integrating differential equations of the form or equivalently of the form particularly in the case of a dynamical system of classical mechanics.

$$\mathbf{x}_i(t + \Delta t) = \mathbf{x}_i(t) + \mathbf{v}_i\left(t + \frac{\Delta t}{2}\right)\Delta t$$

$$\mathbf{v}_i\left(t + \frac{\Delta t}{2}\right) = \mathbf{v}_i\left(t - \frac{\Delta t}{2}\right) + \mathbf{a}_i(t)\Delta t$$

To start our iterative process we need the initial conditions, the initial positions and the initial velocities of all the bodies that make up our system: $x_{i0} = x_i(0)$ and $v_{i0} = v_i(0)$, and for implementing the leap-frog method we need one additional calculation.

$$\mathbf{v}_i\left(\frac{\Delta t}{2}\right) = \mathbf{v}_{i0} + \mathbf{a}_i(0)\frac{\Delta t}{2}$$

The Euler-Cromer method, which is another commonly used method of integration, is seen to be the same as the leapfrog method, except that the velocity is computed at the same times as the position rather than intermediate times. Thus, Euler-Cromer becomes leapfrog simply by updating the velocity by an extra half-step at the beginning, and using the resulting value of v as the starting value. By this simple change, a first order method (Euler-Cromer) becomes a second order method (leapfrog). It is therefore recommended to use leapfrog rather than Euler-Cromer.

4.3 Using Vpython

A python package called VPython, which is the Python programming language plus a 3D graphics module called Visual, was used to visualise the solution to the specified three-body problem. VPython allows users to create 3D objects and display them in a window if certain requirements are met.

4.4 Explanation of the Code

4.4.1 Problem 1 and 2

Being the major element of the 3-body problem, the gravitational force is initially defined and its computations are stated. Following that, I establish certain constants for the objects I'm working with, such as mass and radius, as well as initial conditions like position and velocity. The position and momentum of each body in the system under gravitational attraction are updated after the gravitational forces are calculated. Two graphs are plotted from the calculations done in the code. The first one is the momentum time graph of the 3 bodies and the second is the velocity-time graph of the 3 bodies.

4.4.2 Problem 3

Unlike Problem 1 and Problem 2, we use the Leapfrog method in Problem 3 as it gives more accurate results by reducing the error. I first define the acceleration caused due to gravitational force. After defining the constants, I set up the scene for animation with the size and colour of the canvas. Next, I create the bodies and give them initial conditions and use the Leapfrog method of integration to solve the equations of motion. The total momentum of the system is set to zero for creating a stable system. Two graphs are plotted from the calculations done in the code. The first one is the momentum-time graph of the 3 bodies and the second is the velocity-time graph of the 3 bodies.

5 Observations

5.1 Problem 1

Some instants from the animation and the graphs are shown here:

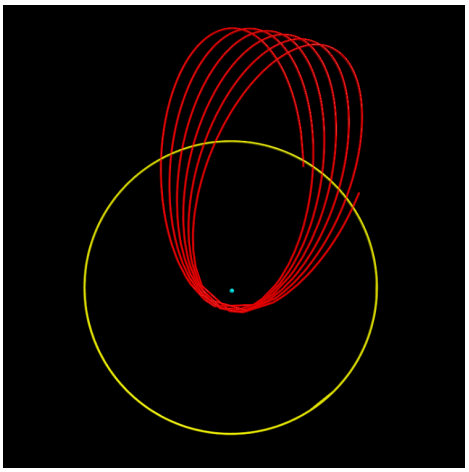

Figure 1: The first few instants: beginning of the pattern

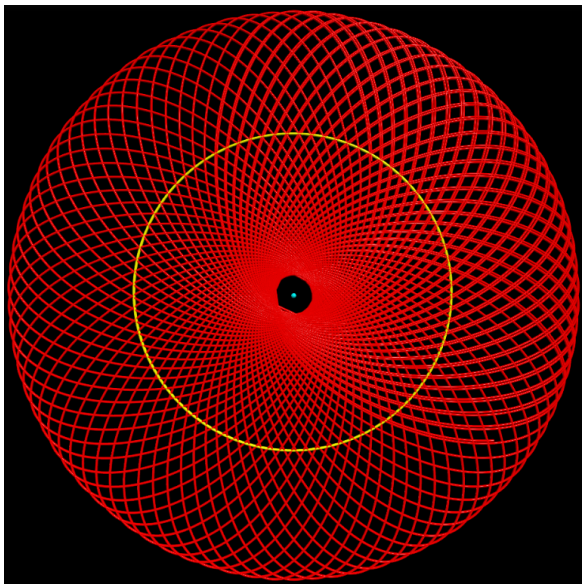

Figure 2: Later stage: completed orbits

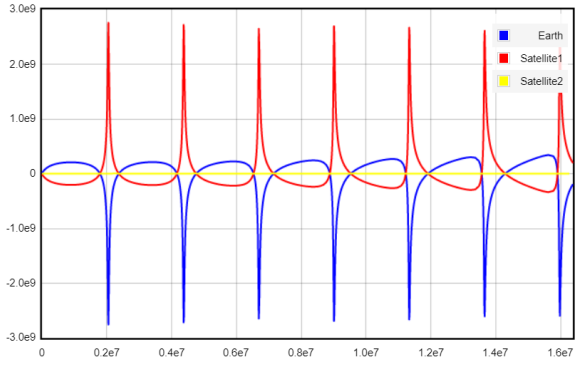

Figure 3: Momentum vs. Time graph of Earth and the satellites

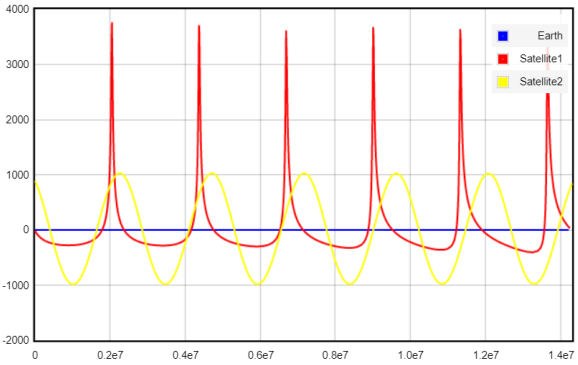

Figure 4: Velocity vs. Time graph of Earth and the satellites

5.2 Problem 2

Some instants from the animation and the plotted graphs are shown here:

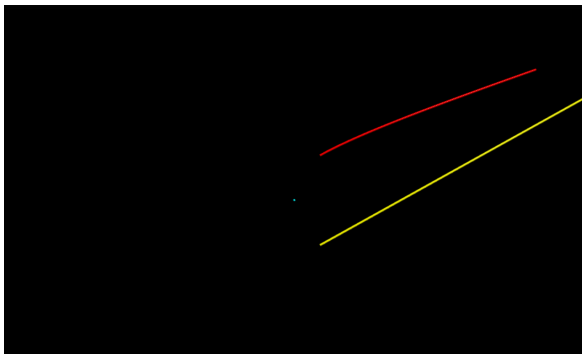

Figure 5: Earth and the satellites

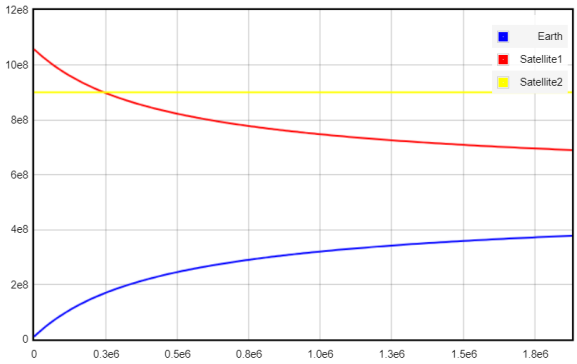

Figure 6: Momentum vs. Time graph of Earth and the satellites

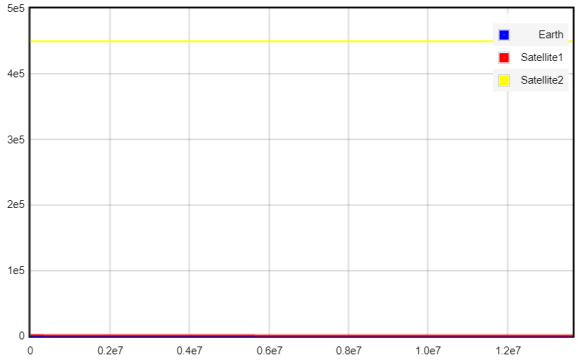

Figure 7: Velocity vs. Time graph of Earth and the satellites

5.3 Problem 3

Some instants from the animation and the plotted graphs are shown here:

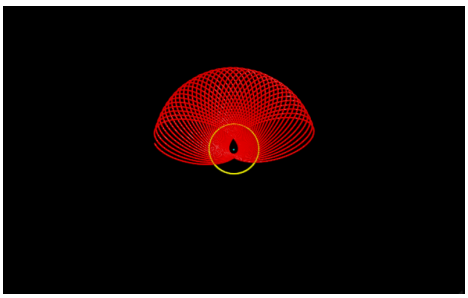

Figure 8: The first few instants: beginning of the pattern

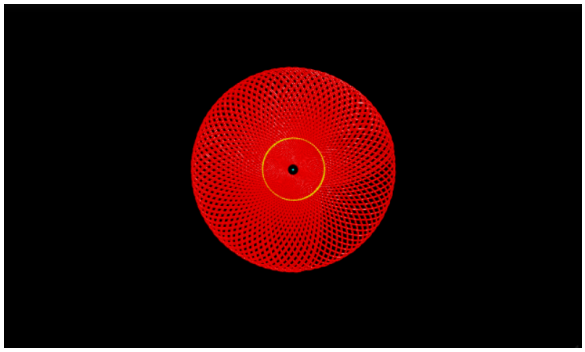

Figure 9: Later stage: completed orbits

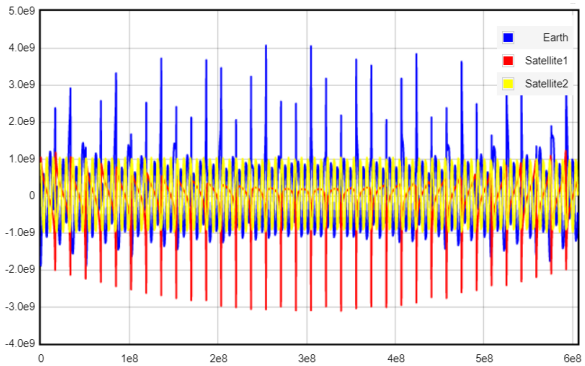

Figure 10: Momentum vs. Time graph of Earth and the satellites: first few instants

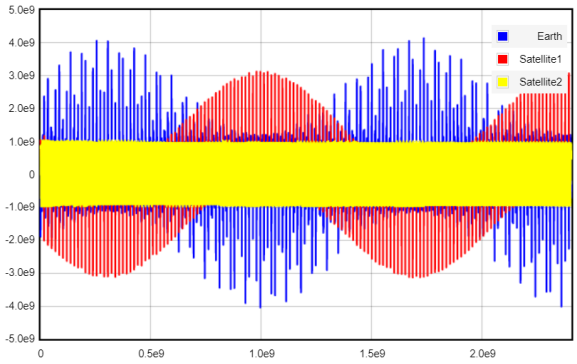

Figure 11: Momentum vs. Time graph of Earth and the satellites: later stage

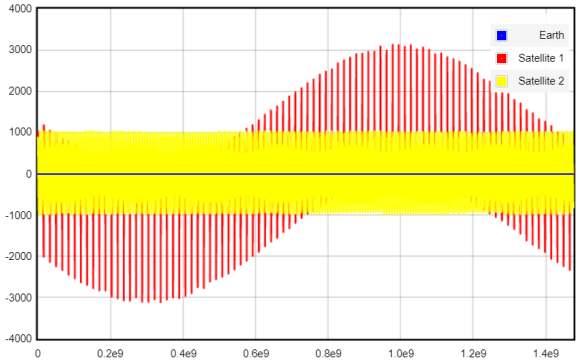

Figure 12: Velocity vs. Time graph of Earth and the satellites

6 Discussion

6.1 On the Code

It has been observed that as the rate is increased, sharp curves in the trajectories of the bodies appear, and the motion is not smooth. This could be due to the fact that as the rate is increased, the number of loops per second increases, and the computer's processing cannot keep up. The animation becomes faster as dt is increased, which could be because there is a greater change in position as dt is increased.

6.2 On Problem 1

Figure 1 shows the beginning of the pattern of the trajectory. The trajectory of the satellite with the heavier mass is a precessing ellipse and it is clear that the Earth lies in one of the two focii of that ellipse. It is to be noted that the satellite with a lower mass, Satellite 2, goes around the Earth in a circle. The graph in Figure 3 shows that the momentum of the first satellite is constant and it leads to the uniform circular motion. Slight changes in the momenta of Satellite 1 and Earth are noticed in Figure 3 as time progresses which might explain why a precessing ellipse is obtained for the trajectory of the first satellite instead of a closed ellipse.
The result obtained from the graph makes us wonder if the huge difference in the shape of the trajectory is actually related to the extremely huge difference in mass of the satellites or there is any other reason contributing to this.

6.3 On Problem 2

Due to extremely high error, this doesn't form a stable system. As seen in Figure 5, both the satellites get ejected from the system. Graph in figure 7 shows how the satellites get ejected with constant velocity.

6.4 On Problem 3

Figure 8 shows the beginning of the pattern of the trajectory. Unlike Problem 1, the masses of the satellites are the same yet the trajectories of the two satellites how a similar pattern to the trajectories of the satellites in Problem 1 which indicates that the difference in the trajectory might be due to the initial velocities of the satellites. The trajectory of the satellite with a higher initial velocity is a precessing ellipse and it is clear that the Earth lies in one of the two focii of that ellipse. The satellite with a lower initial velocity, Satellite 2 goes

around the Earth in a perfect circle which is also clear from the velocity graph Figure 12. The Earth is stationary which is also indicated by the straight line in the velocity graph Figure 12. Figure 11 indicated that the total momentum of the system is almost constant which gives us a stable system.
The motion of three bodies is generally non-repeating but in Figures 1 and 2 and graphs 3 and 4 of Problem 1, and Figures 8 and 9 and graphs 10, 11 and 12 of Problem 3, it can be noticed clearly that the pattern is repeating.

6.5 Probable Sources of Error

A computer we cannot represent a real number like π exactly, as a result real numbers have to be approximately represented, or rounded off. This approximation results in the error in our arithmetic operations. A typical personal computer cannot distinguish between 1 and $1 + \epsilon$ where $\epsilon \approx 10^{16}$. Another source of error is due to the small but finite value of the time interval dt. This error can be reduced by reducing the size of dt, but that would increase the number of intermediate steps required to reach the value $x(t)$ and that correspondingly increases the round off error.

7 Conclusion

From the various situations simulated it is clear that the leapfrog method is a better method for solving equations of motion on a computer as it greatly reduces error. However, more accurate methods like Runge-Kutta 4 and Runge-Kutta 6 exist but they are outside the purview of this project. The project can be extended to include more satellites and also the modelling of space debris but these couldn't be done in this project due to limited computational power.

8 Appendix

The python codes used for the experiment can be found here:
Problem 1
Problem 2
Problem 3

9 References

Going Beyond the Two-body Problem, Vikram Vyas
Feynman Lectures in Physics, Vol-1, Chp. 9
Leapfrog Method
The Visual Python Web site
Wikipedia: N-body Problem
Wikipedia: Satellite (artificial)
Wikipedia: Satellite