N-body program using a Verlet integrator

Program description

The program takes the masses, initial positions and velocities of N bodies and evolves the position of the bodies over time under Newtonian gravity using a Verlet integrator **[1]**. The initial parameters are defined as global variables which can be tweaked by the user, including the integrator's time step and total time. Upon running the program's Verlet function, it returns an array of all the trajectories and additionally, the total energy and angular momentum over time. Trajectories, energy and angular momentum are then directly plotted for analysis.

The program is tested using some well known circular and elliptical 2-body orbital setups, e.g. Earth and Moon. After running, I check how well energy and angular momentum are conserved and also if Kepler's 2nd law is obeyed. The result is also compared to the analytical plots of the given circular or elliptical orbit.

After testing, I use it to reproduce some well studied 3-body periodic solutions. In a separate program, I made use of an adaptive time step to reproduce one of the more complex 3-body solutions.

2-body validation

After developing the program, I initially tested it with some well known 2 body near circular and more elliptical orbits. To test the results, we first expect the total energy, E and total angular momentum, L to be constant at every instance of time. The total energy is just ΣKE + ΣGPE over all bodies. For kinetic energy we have:

$$K_{\text{total}} = \frac{1}{2} \sum_{i=1}^{N} m_i \mathbf{v}_i \cdot \mathbf{v}_i$$

Where m_i is the mass of the ith body and v_i is the velocity of the ith body. For the total GPE we have to sum over all pairs, also making sure we only count each pair once, thus it's given by the double summation:

$$U_{\text{total}} = -G \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \frac{m_i m_j}{r_{ij}}$$

Where r_{ij} is the distance between the ith and jth body. For the total angular momentum, I chose the origin as the reference frame. Similar to the KE, we can summate **L** over all bodies:

$$\mathbf{L}_{\text{total}} = \sum_{i=1}^{N} m_i (\mathbf{r}_i \times \mathbf{v}_i)$$

I implement these formulae in my program within a function that takes the current position and velocities of all bodies, then returns E = K + U and |**L**|.

Another test for 2 body orbits is Kepler's 2nd law, which states, "For 2 bodies in orbit about a common center of mass, an imaginary line connecting one of the bodies to the center of mass will sweep out equal areas within equal time intervals.". In our program we can naturally test this by considering the area swept between the integrator's time steps. One can approximate this area using Heron's formula demonstrated in **Figure 1**.

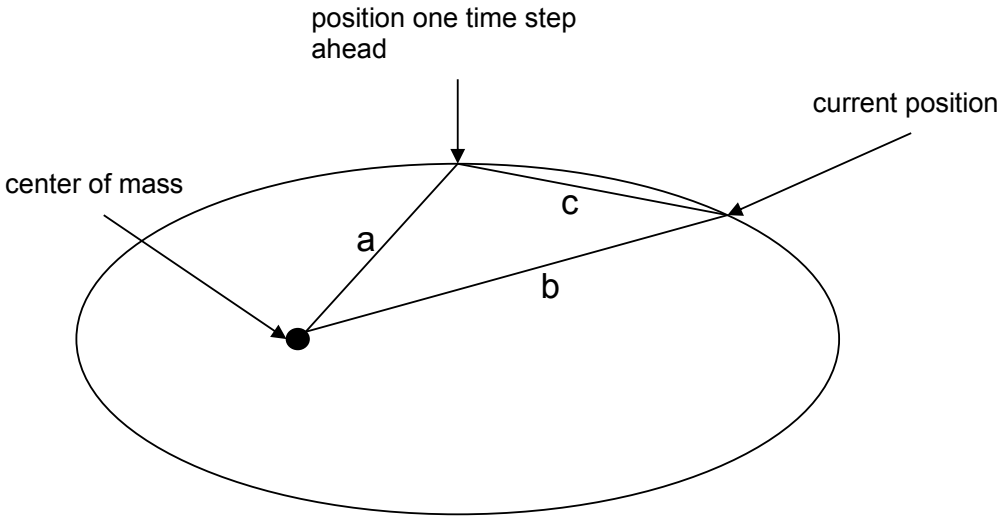

Figure 1: Shows the triangle abc we want to compute the area of, using Heron's formula for some arbitrary 2 body orbit. This works out as a good approximation when c/p << 1, where p is the perimeter of the ellipse.

Heron's formula **[2]**:

$$A = \sqrt{s(s-a)(s-b)(s-c)}$$

Where A is the triangle area, s is the semi perimeter of triangle abc and a, b, c are the side lengths. This area is computed within our program between every time step and we expect a constant area plot with a constant time step, if Kepler's 2nd law holds.

A third test which is more qualitative, is done by plotting a known circular or elliptical orbit and comparing it to our model. For example, we can take the circular radius of the international space station (ISS) orbit and plot that about the origin, we then enter some initial conditions into the program (initial orbital velocity and orbital radius). An orbit along the same trajectory as the known circular orbit of the ISS, indicates our program is running as intended. For a more quantitative comparison we output the error in the mean difference of the numerical and analytical trajectories.

An analytical circular orbit trajectory is plotted as:

$$\mathbf{r} = [R\cos(\omega t), R\sin(\omega t)]$$
$$\omega = \sqrt{\frac{GM}{R^3}}$$

Where R is a constant orbital radius, t is time and M is the central body's mass. For an elliptical trajectory, there isn't an analytical solution w.r.t time. Instead, I chose to produce a domain of the polar coordinate θ using the trajectories computed by the program, as such:

$$\theta = \arccos\left(\frac{\mathbf{r}\cdot\mathbf{r}_0}{|\mathbf{r}||\mathbf{r}_0|}\right)$$

Where we set the initial position, $\mathbf{r}_0$ as [A, 0], where A is the expected aphelion of the ellipse . An important note is if $\mathbf{r} \times \mathbf{r}_0 < 0$, we have to set θ to $2\pi - \theta$, enforcing a full revolution. Using the domain of θ, we can calculate the given elliptical trajectory as:

$$R(\theta) = \frac{a(1-e^2)}{1-e\cos\theta}$$

$$\mathbf{r} = [R\cos(\theta), R\sin(\theta)]$$

Where a is the ellipse semi major axis and e is the eccentricity of our ellipse, where we can use known values, i.e. the eccentricity of Mercury's orbit. The justification for this method is that my program has no knowledge of the semi major axis and eccentricity of the orbit it's modelling, and should produce the ellipse from Newton's laws only.

2-body results

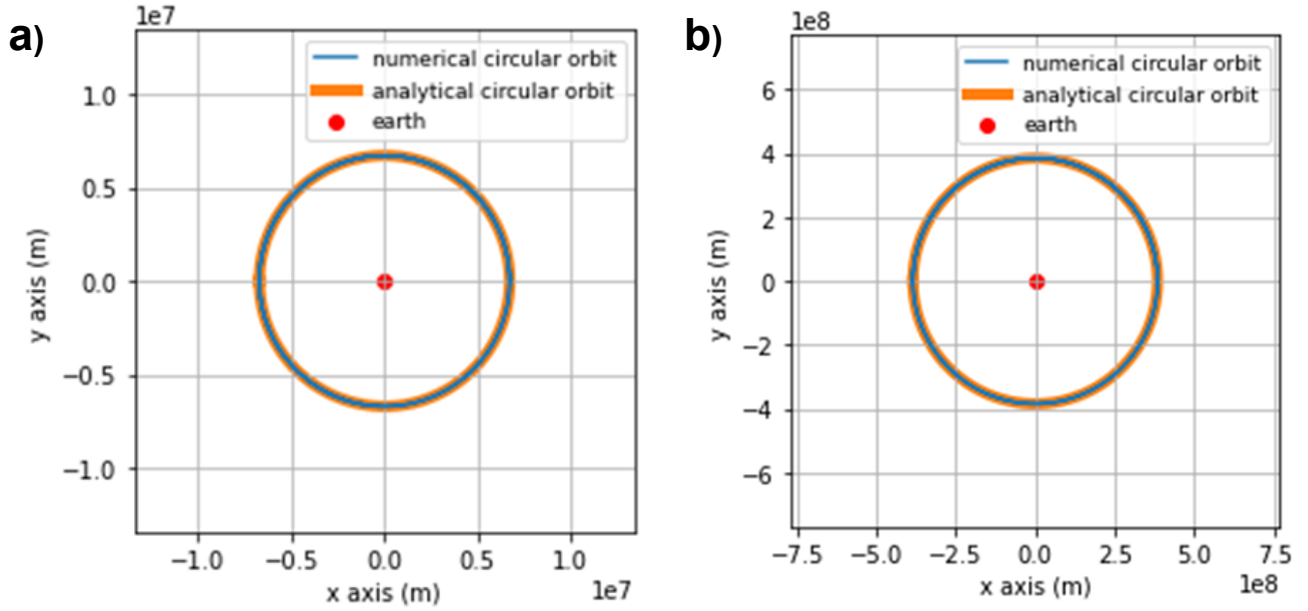

Figure 2: Shows 2 body circular orbits produced using the N-body program. For **a),** the initial parameters are an Earth-ISS setup, with a timestep, h = 1s, integrated over 15000 timesteps. For **b)**, the initial parameters are an Earth-Moon setup, h = 60s, integrated over 100000 timesteps. Also in **b)**, the initial velocity of the Earth had to be set so that total linear momentum = 0, to avoid the system drifting, as the Moon's gravitational pull on the Earth is not negligible.

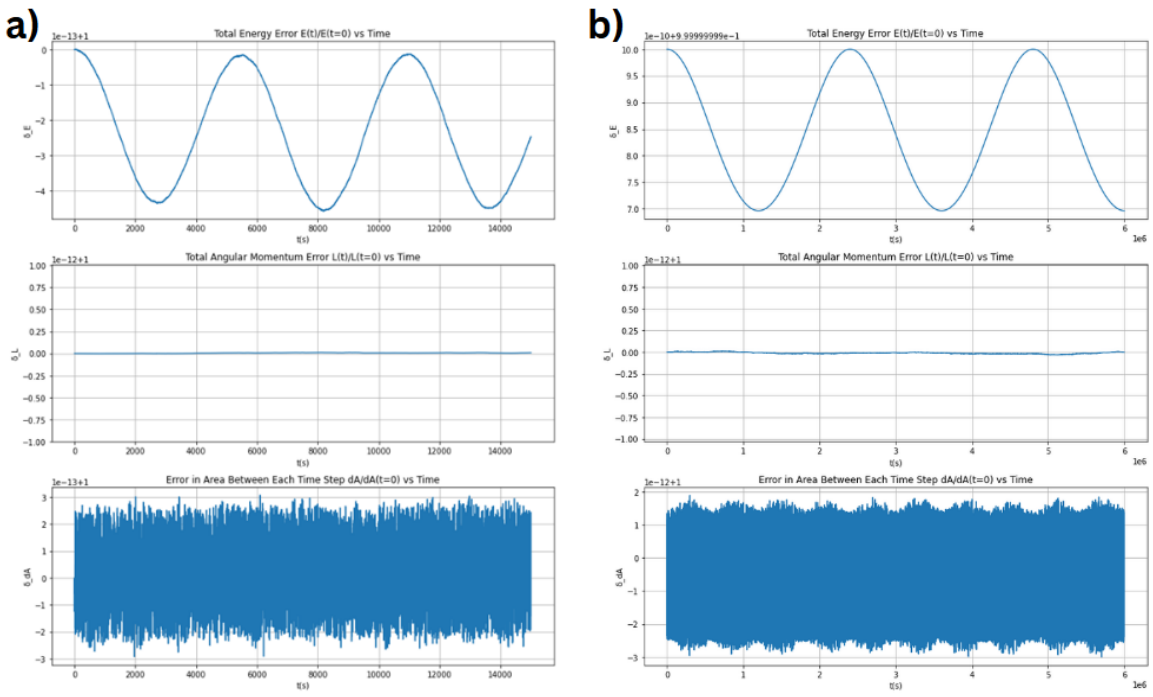

Figure 3: *Click image to view in detail.* **a)**, shows the energy, angular momentum and area between timestep errors corresponding to **Figure 2 a)**. Likewise, **b)** displays the same data corresponding to **Figure 2 b)**.

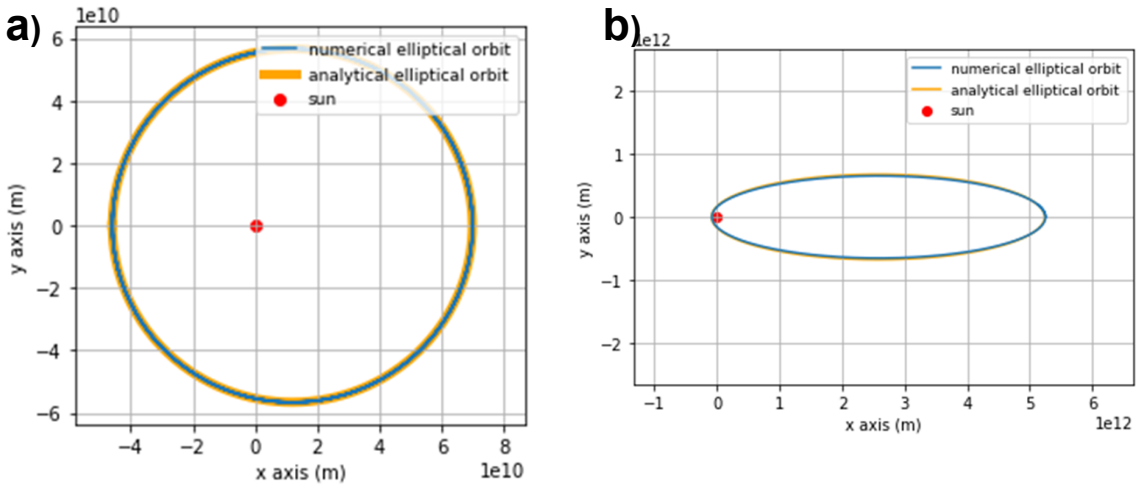

Figure 4: Shows 2 body elliptical orbits produced using the N-body program. **a)** shows a Sun-Mercury setup with h = 500s integrated over 60000 timesteps and **b)** shows a Sun-1P/Halley setup, with h = 43200s integrated over 60000 timesteps.

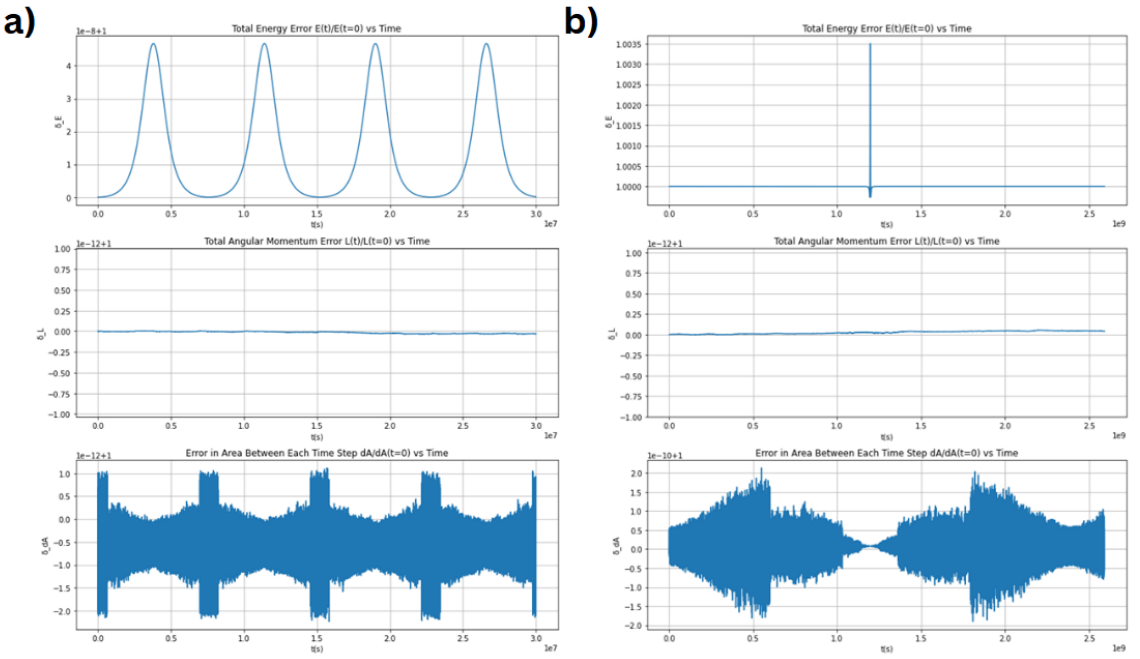

Figure 5: *Click image to view in detail.* **a)**, shows the energy, angular momentum and area between timestep errors corresponding to **Figure 4 a)**. Likewise, **b)** displays the same data corresponding to **Figure 4 b)**.

Orbital setup	Mean difference error (mde)
Earth-ISS **Figure 2 a)**	3.81×10^{-6}
Earth-Moon **Figure 2 b)**	0.00013
Sun-Mercury **Figure 4 a)**	0.00014
Sun-1P/Halley **Figure 4 b)**	0.0063

Table 1: Shows the mean difference error (mde) between the numerical and analytical trajectories for the 2 body orbits above. For the circular orbits this is calculated as avg($|\mathbf{r}_{analytical} - \mathbf{r}_{numerical}|$)/R and for the elliptical orbits it's calculated as avg($|\mathbf{r}_{analytical} - \mathbf{r}_{numerical}|$)/a.

Our N-body program seems to model 2 body circular orbits with little error. Fluctuations in energy and angular momentum error are typically << 1, meaning energy and angular momentum are well conserved; this is similar for the area plots, showing Kepler's 2nd law is obeyed well. An interesting result is an observed spike in the energy error in **Figure 4 b)**. This is where 1P/Halley approaches the sun and is at its maximum orbital velocity, the highly elliptical nature causes the velocity scale to increase by 2 orders of

magnitudes around the perihelion. As a result, the larger timestep becomes less appropriate for the dynamics and causes high amounts of energy error to accumulate.

The Earth-Moon mde is small but its presence is due to the oscillating Earth-Moon common center of mass, because the Moon exerts a significant pull on the Earth.The larger mde of 1P/Halley is explained by the fact that it's initial velocity isn't well defined and had to be estimated using the vis-viva equation **[3]**. Overall mde presence is due to a combination of small energy errors and a lack of precision in the initial conditions.

3-body orbits

3+body orbits where masses are similar can easily become chaotic, with most setups having no clear analytical solutions. The program is used to reproduce 6, 3-body well studied stable periodic orbits, presented in **Figure 6 [4]**, with their total energies listed in **Table 2**.

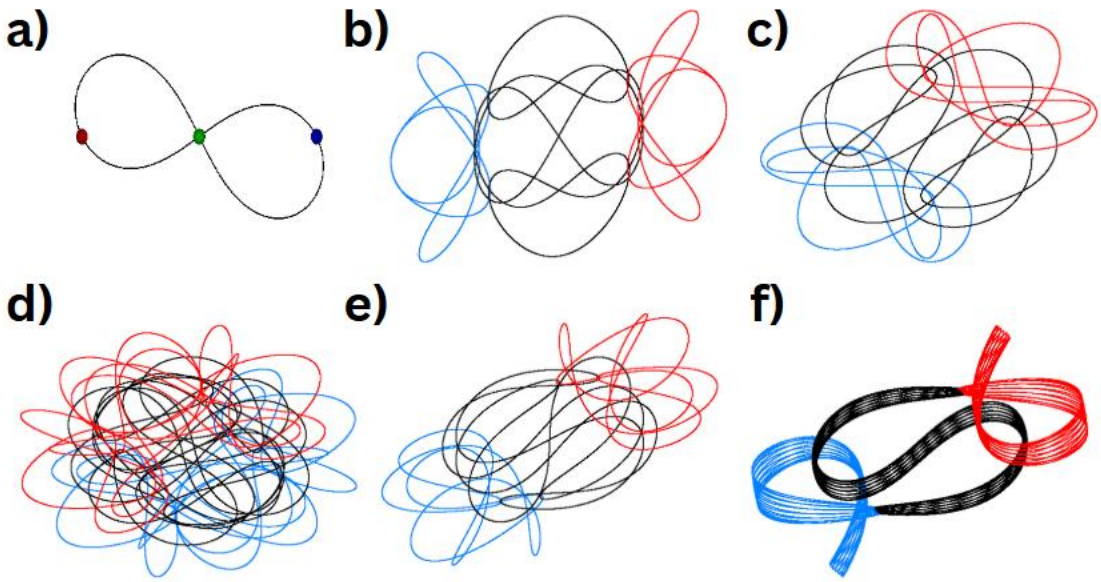

Figure 6: Shows the periodic 3 body orbit patterns that we want to reproduce. They're labeled as: **a)** "FIGURE 8", **b)** "BUTTERFLY I", **c)** "MOTH I", **d)** "YARN", **e)** "YIN-YANG 1a" and **f)** "YIN-YANG 2a". Their initial positions are always [1,0], [-1,0] and [0,0], all have a total angular momentum of 0 and the gravitational coupling constant, G, is set to 1. See **[4]** for their initial velocities and periods.

3-body orbit solution	Total energy (J)
FIGURE 8	-1.287146
BUTTERFLY I	-2.170195
MOTH I	-1.382281
YARN	-1.196537
YIN YANG 1a	-1.429011
YIN YANG 2a	-1.651418

Table 2: Lists the total energy for every orbit from **Figure 6**.

The program should produce the orbital patterns in **Figure 6** and with energy graphs flatlined at the total energies listed in **Table 2** with little error fluctuations. It's also expected that the final position of the bodies end up at the initial positions described in **Figure 6** after the program integrates over the orbit's period.

3-body results

3-body orbit solution	Time step used (ms)	Total time = period (s)
FIGURE 8	1	6.3240
BUTTERFLY I	0.1	6.2360
MOTH I	1	14.890
YARN	0.01	55.50180
YIN YANG 1a	0.1	17.3280
YIN YANG 2a	~	55.789829

Table 3: Lists the time properties entered before running the program.

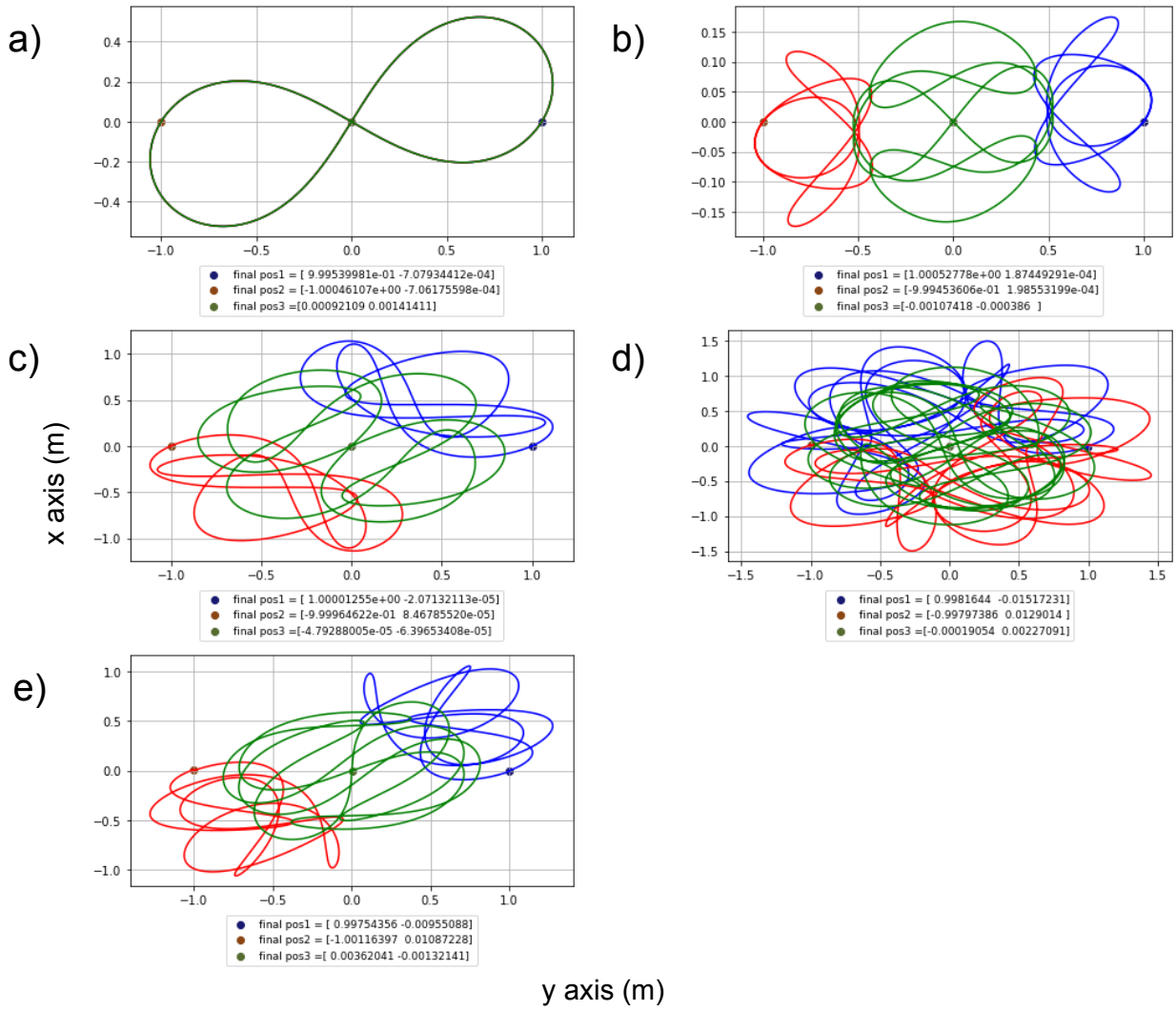

Figure 7: *Click image to view in detail.* Displays the results after running the program with the initial conditions of the orbits in **Figure 6**. Also lists the final positions of the 3 bodies. Click here, to view the corresponding energy and angular momentum plots.

Figure 7 shows the program can successfully produce the orbits in **Figure 6 a) - e).** Angular momentum is well conserved with the highest drift for the YARN orbit only an OM of 10^{-14}. The energy is well conserved for the FIGURE 8 orbit with oscillations of only ~ 10^{-7}J about the expected value listed in **Table 2**. As for the energy in the other orbits, we observe a flatline at the expected value followed by occasional spikes, similar to the energy graph in **Figure 5 b)** describing 1P/Halley. The spike's presence is due to the same reason described earlier; typically when the velocity scale for any of the

bodies suddenly increases, the time step is too large to accurately describe the dynamics. The spikes are only a minor change, not enough for the bodies to escape the periodic orbit, but still slightly reduces the accuracy of the trajectory. The trajectory error is shown by the final positions of the orbits in **Figure 7** slightly differ from their initial position.

Adaptive time step implementation

It's important to mention that the production of the YARN's orbit in **Figure 7 d),** required a much smaller timestep, corresponding to the trajectory having ~5.5 million data points, taking ~30 mins to run. This problem becomes worse when trying to reproduce the YIN-YANG 2a orbit in **Figure 6 f)**. Using the same time step as the YARN's, it failed to produce YIN-YANG 2a, reducing time step by factor of 10, corresponding to a run time ~5 hours, still failed to produce YIN-YANG 2a. Reducing by another factor of 10 estimates a run time of at least 2 days on my machine.

Clearly having a constant time step for certain orbits is inefficient. There are long periods of the orbit where the dynamics can be modeled using a greater time step, but short moments where said time step is insufficient. Using a smaller timestep all round just to account for these short moments drastically reduces efficiency.

To work out when and how to adjust the time step, we observe how the magnitude of acceleration is also changing with time. This is shown in **Figure 8**.

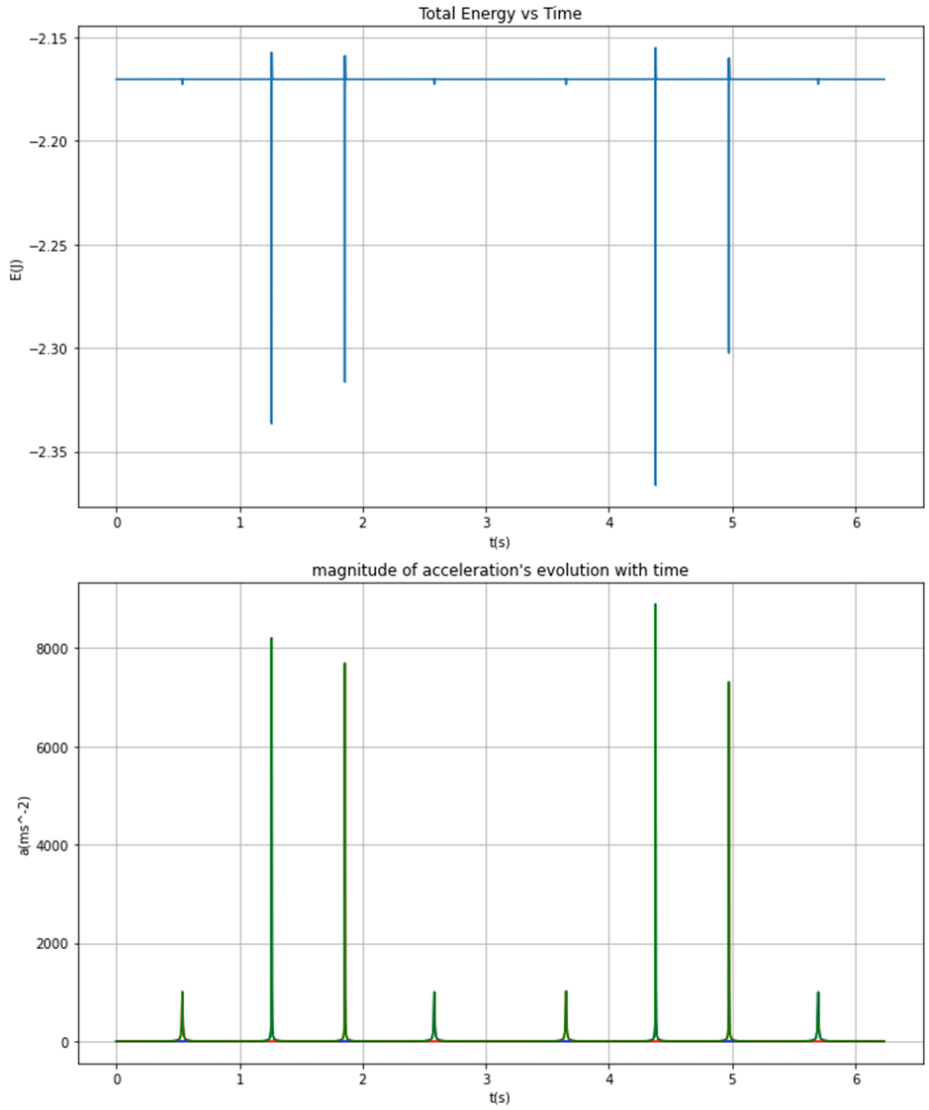

Figure 8: Shows the energy graph of the BUTTERFLY 1 orbit along with a graph of the acceleration magnitude for all bodies.

It's observed in **Figure 8** that unwanted spikes in energy occur when the acceleration magnitude undergoes a sharp increase, which typically happens when the bodies closely approach one another. From this, I chose a time step that depends on the max acceleration magnitude out of all bodies:

```
h=max(h_0 * a_i/max(a), h_min)
```

Where, `h_0` is the initial timestep, `a_i` is the initial acceleration magnitude (= 1.25 ms^{-2}) and `h_min` is a chosen minimum time step. `h` reduces when the maximum acceleration of all the bodies is greater than the initial acceleration and vice versa. The line of code above also imposes that `h > h_min` to avoid it being overly reduced.

Creating a separate program with this adaptive time step implemented, I first tried to produce the YARN orbit again, setting h_0 = 1ms, and h_min = h_0/100. The program was successful, only taking ~3 mins compared to ~30 mins previously. Additionally it was found that use of an adaptive time step better conserves energy, shown in **Figure 9**.

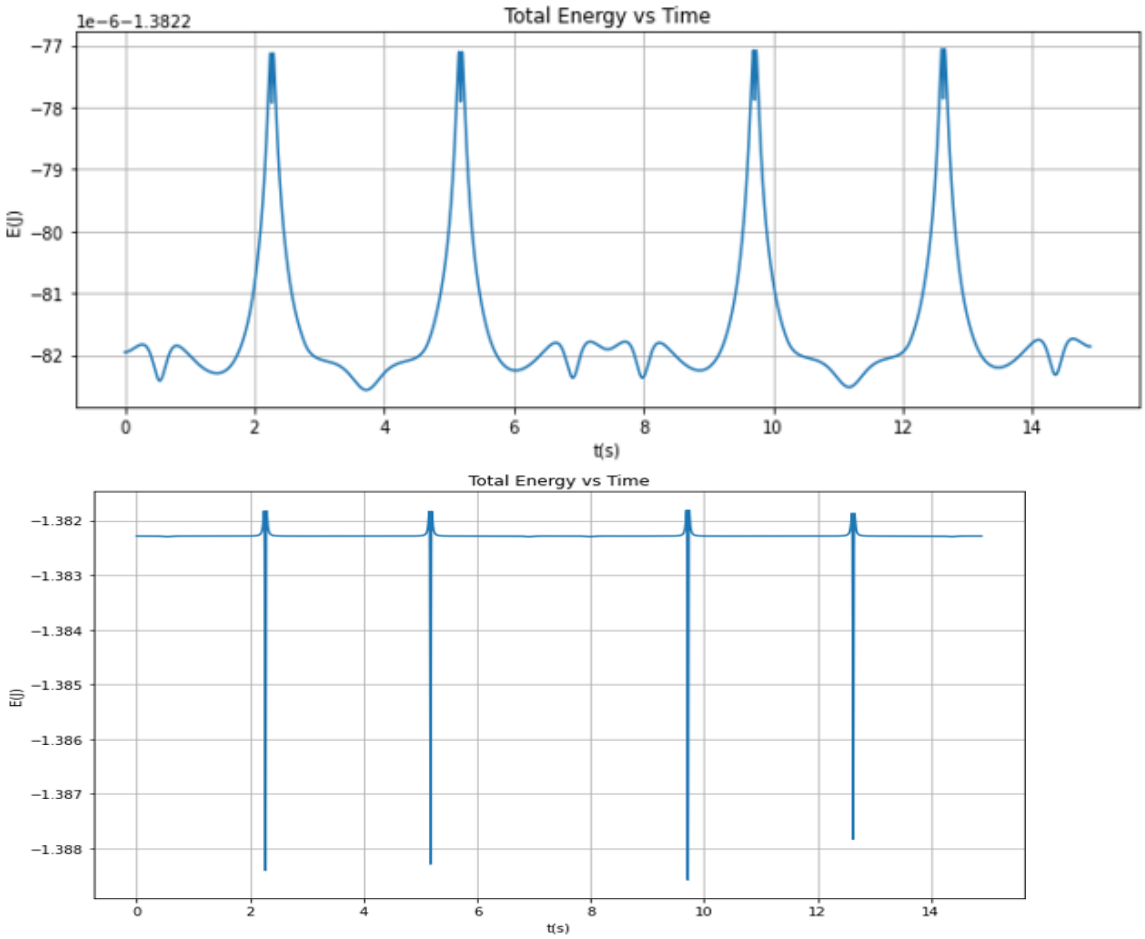

Figure 9: Energy graph for the MOTH I orbit using the adaptive time (top graph) vs the constant time step (bottom graph). Here h_0 = 1ms and h_min = h_0/100.

Before, the MOTH I orbit had energy spikes of ~10^{-3} J, now observed in **Figure 9**, they're reduced to ~10^{-5} J.

After verifying the improvements in efficiency of the adaptive time step program, a successful attempt was made to produce the YIN YANG 2a orbit with a run time of ~1hr, presented in **Figure 10**.

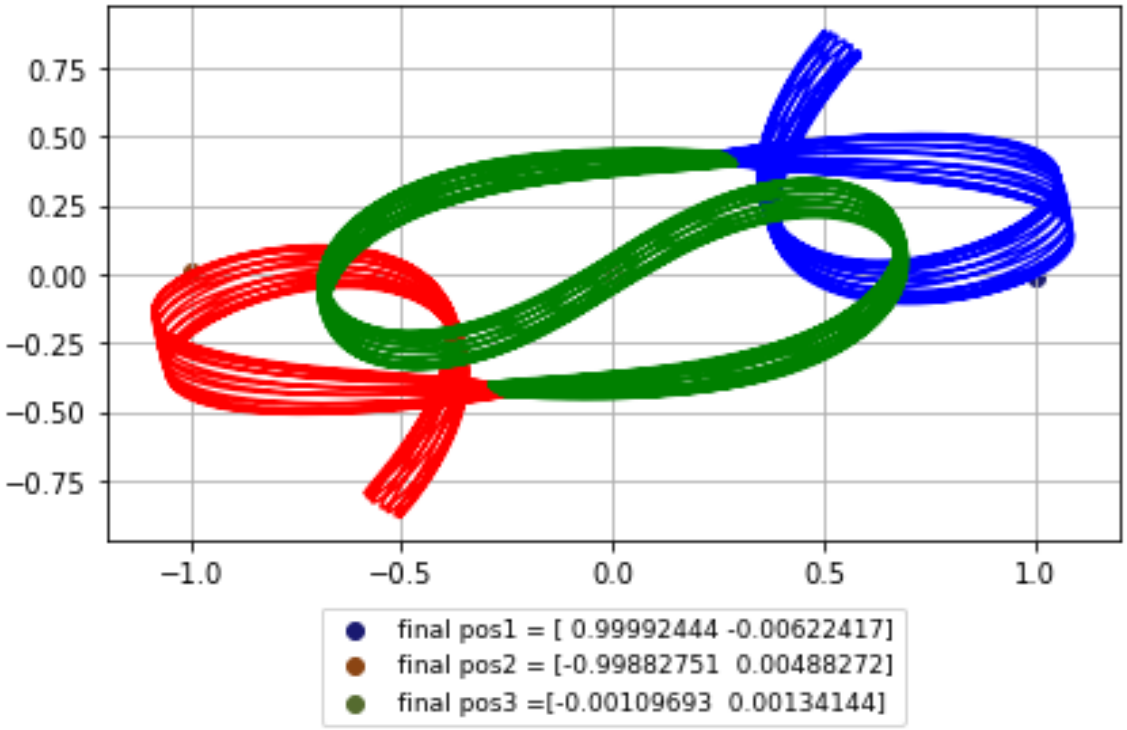

Figure 10: YIN YANG 2a orbit produced using h_0 = 1ms and h_min = 10^{-7}s.

Conclusions

The use of a Verlet integrator is an effective n-body gravitational solver for a relatively small number of bodies, showing success at conserving total energy and angular momentum, when using an appropriate time step. The 2 body systems that were run, showed strong correlation with their analytical counterparts as well as sufficient Kepler's 2nd law obedience. It would become inefficient for a larger number of bodies, because evaluating the accelerations requires a double loop where the number of operations is ~ $O(N^2)$, N being the number of bodies.

The integrator could successfully produce 5 of the 6, 3 body solutions in **Figure 6** with reasonable accuracy but efficiency issues were discovered when using a constant time step, which is why we initially failed to produce the 6th YIN-YANG 2a orbit.

By analysing the trends in the energy graphs, an adaptive time step program was implemented, which turned out to have a significant increase in efficiency, enabling us to produce the YIN-YANG 2a orbit within reasonable run time.

References

[1]: Verlet integration, accessed November 2024
[2]: Heron's formula, accessed November 2024
[3]: Tom Logsdon (1998). Orbital Mechanics: Theory and Applications. John Wiley & Sons. ISBN 978-0-471-14636-0.
[4]: Three body gallery, accessed November 2024