Satellite problem and N-body problem, solved in *C*

Authors: Aurora Abbondanza, Alessandro Agapito, Andrea Belli Contarini, Riccardo Caleno

Theoretical Astrophysics Course - MSc Physics

Abstract

As a first project's objective, we wanted to study the fall of a satellite on Earth, at a certain distance from it. Therefore, we wrote a program in *C* language which solves the differential equation that governs the motion of the satellite and we studied (numerically) what happens if some motion parameters would change.
Additionally, we worked on the *N*-body problem. Thus, we wrote a *C* program which predicts the motion of *N* celestial bodies under the influence of their mutual gravitational forces.

All the results (plots, data, *C* and *Python* codes) are stored in the *Google Drive folder* linked.

1 Introduction: problem description

1.1 Satellite

We are interested in studying the fall of a satellite on Earth, at a certain distance (which we set equal to the geostationary radius[1] $r_{geo} = 4.216 \times 10^7\ m$) from the center of our planet (origin of our coordinates system). We want to study how the satellite initial conditions (position and velocity) and mass loss over time would affect its trajectory, radial distance from Earth, energy and time flight. To accomplish this, we used the mass and mass loss data of the Hubble Space Telescope, which has been in orbit around Earth since 1990.
In general, the equation of motion of the satellite, neglecting the Earth motion ($m_s \ll M_E$), is the following:

$$\ddot{\boldsymbol{r}} = \boldsymbol{\nabla} U - \left(\frac{\lambda^2 + \dot{m}_s}{m_s}\right)\dot{\boldsymbol{r}} = -\left(\frac{GM_E}{r^3}\right)\boldsymbol{r} - \left(\frac{\lambda^2 + \dot{m}_s}{m_s}\right)\dot{\boldsymbol{r}} \tag{1}$$

where λ^2 is the drag coefficient, m_s is the mass of the satellite, which, for Hubble, is equal to 11 110 kg; its mass loss (per unit time) due to fuel consumption is $\dot{m}_s = \frac{dm_s}{dt}$. The Earth mass is $M_E = 5.972 \times 10^{24}\ kg$, $G = 6.674 \times 10^{-11}\ Nm^2kg^{-2}$ is the gravitational constant and $r = \sqrt{x^2 + y^2 + z^2}$ is the module distance from the center of Earth.

To solve the problem, we developed a computer program (`satellite.c`) in the programming language *C*. The program used a specific numerical method (*Runge-Kutta 4*) to solve the equation of motion (1), taking into account the gravitational attraction between the satellite and Earth, as well as the effect of atmospheric drag, the satellite changing mass due to fuel consumption and other factors.
By running the program, we were able to obtain the trajectory of the satellite over time, as well as its radial distance from Earth and its energy at each point along the trajectory. This allowed us to analyze many aspects of the motion, and plot the results (with *Python*). Moreover, we made additional several plots, in order to study the behavior of the satellite when some parameters (such as the atmospheric drag) were changed.

1.2 N-body

In the second part of our project, we focused on the *N*-body problem. This is a fundamental issue in astrophysics that deals with the motion of a group of celestial bodies, which interact with each other through gravitational field. In essence, it is the problem of predicting the motion of N point like celestial bodies under the influence of their mutual gravitational interaction[2].
The *N*-Body problem is governed by a set of $3N$ coupled second-order differential equations, which describe how the positions and velocities of the bodies evolve over time in response to the gravitational field of $N-1$ bodies for each body:

$$\ddot{\boldsymbol{r}}_{\boldsymbol{i}} = \boldsymbol{\nabla}_{\boldsymbol{i}} U = \frac{\boldsymbol{F_i}}{m_i} = \sum_{\substack{j=1 \\ i \neq j}}^{N} \frac{G m_j (\boldsymbol{r_j} - \boldsymbol{r_i})}{r_{ij}^3} \tag{2}$$

[1] A geostationary orbit is a type of orbit around Earth, in which a satellite orbital period matches the Earth rotational period $T_{geo} \simeq 24\ hours$, so that the satellite remains in a fixed position relative to the Earth surface.
[2] We assume the motion as "collisionless" in order to simplify the interactions.

where $r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 + \epsilon^2}$ is the distance between the couple, m_j is the mass of the j-body, G is the gravitational constant and ϵ is the softening parameter lenght.
To avoid the *UV problem* in the potential U_i and the collisions where $r_{ij} = 0$, we defined the so called softening parameter ϵ. The choice of this constant's value will be discussed later on in the box "softening parameter" and in sec.3.4.1.

Initially, in order to test the validity of our program, we solved the 2-body problem, since its analytic solution is known (C program: `2body.c`); we would input different initial velocities for each sample and see how the trajectories would change.

Afterwards, we solved a particular 3-body problem (`3body.c`), setting the initial positions of the three bodies on the vertices of the Pitagora's triangle, and null initial velocities.

Finally, we developed the general program `Nbody.c`, where the N bodies' initial positions would be randomly chosen by the program itself, following an homogeneous distribution in a sphere, while the initial velocities in a way that will be discussed in detail in the subsection 3.5.

All these programs used the *leapfrog* numerical method to solve the equation of motion (2). Although it converges to the exact solution at a slower rate, the *leapfrog* algorithm has been chosen over *RK4*, since the computational effort turns out to be lower. Computing *RK4* for $N = 500$ or 1000 bodies would take a much larger amount of time.

2 Satellite numerical solutions

2.1 Dimensionless re-scaling

In order to deal with an equation that has to be integrated numerically (with *RK4*) we need to re-scale the equation of motion using dimensionless variables, which we define as:

$$\begin{cases} m' \coloneqq \dfrac{m}{M_E} \\ \boldsymbol{r'} \coloneqq \dfrac{\boldsymbol{r}}{r_{geo}} \\ t' \coloneqq \sqrt{\dfrac{GM_E}{r_{geo}^3}}\, t \simeq (7.3 \times 10^{-5} Hz) \cdot t \end{cases}$$

where $r_{geo} \simeq 4.216 \times 10^7\ m$ represents the radius of the geostationary orbit.
We can also obtain the dimensionless velocity and energy per unit mass:

$$\begin{cases} \dot{\boldsymbol{r'}} \coloneqq \sqrt{\dfrac{r_{geo}}{GM_E}}\, \dot{\boldsymbol{r}} \\ \dfrac{E'}{m'} \coloneqq \dfrac{1}{2}\, \dot{r'}^2 - \dfrac{1}{r'} \end{cases}$$

Therefore, eq.1 can be rewritten as:

$$\boxed{\ddot{\boldsymbol{r'}} = -\left(\frac{1}{r'^3}\right)\boldsymbol{r'} - \beta(t)\dot{\boldsymbol{r'}}} \tag{3}$$

where the dimensionless coefficient $\beta(t) = \left(\dfrac{\lambda^2 + \dot{m}}{m(t)}\right)\sqrt{\dfrac{r_{geo}^3}{GM_E}} = \left(\dfrac{\lambda^2 + \dot{m}}{m(t)}\right) \cdot (1.37 \times 10^4\ s)$ and the time-dependent mass is $m(t) = m_0 + \dot{m}t$, assuming $\ddot{m} = 0$.

2.2 Satellite circular solution

In this case, with the assumption of $\dot{m} = 0$, the equation of motion has the same vectorial form of eq.1:

$$\boxed{\ddot{\boldsymbol{r}} = -\left(\frac{1}{r^3}\right)\boldsymbol{r} - \beta\dot{\boldsymbol{r}}} \tag{4}$$

where $\beta = \frac{\lambda^2}{m}\sqrt{\frac{r_{geo}^3}{GM_E}}$.

Firstly, we studied the planar motion on the xz plane, considering the following initial conditions of the problem:

$$\boldsymbol{r_0} = (1,\ 0,\ 0) \text{ and } \boldsymbol{v_0} = (0,\ 0,\ 1)$$

The satellite lands after $t_{fall} = 24.62$ *hours* and the trajectory is shown (red) in fig.1:

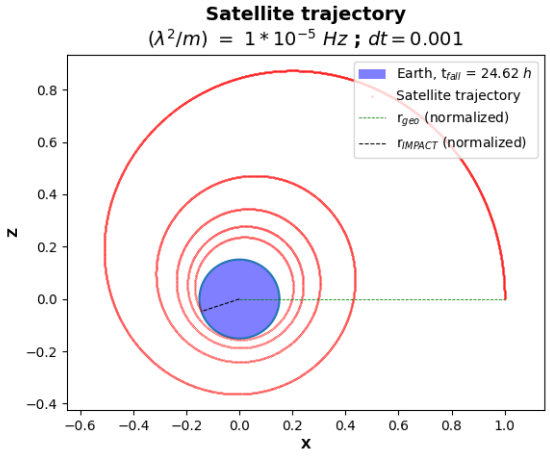


Figure 1: 2D plot, with a drag coefficient per unit mass of $\frac{\lambda^2}{m} = 1 \times 10^{-5}\ Hz$ and a dimensionless numerical time step $dt = 0.001 \simeq 13.7\ s$. The notation of $r_{impact} \simeq 6.4 \times 10^6\ m$ represents the radius of the Earth (blue circle).

We observe circular orbits, whose radius decreases as the satellite speeds up approaching Earth. This solution can be approximated with an analytical function of the type:

$$r(t) = e^{-2\beta t} \tag{5}$$

assuming a circular velocity for the satellite, $v^2 = v_{circ}^2 = \frac{1}{r}$.
From this expression we can also evaluate the first order approximation $r(t) = 1 - 2\beta t + o(\beta^2)$, from which we notice that $r(t) = 1$ if $|2\beta t| \ll 1$ and so we would recover a circular orbit with constant radius. In fig.3 we will make a comparison between this theoretical approximation and our numerical solution.

Drag coefficient

The choice of the drag coefficient value is crucial for the validity of our solution, in fact it determines whether the motion tends to remain circular (even with decreasing r_{orbit}) or become radial.
We decided to use the following qualitative method: when $\frac{\lambda^2}{m} \gg \left(\frac{\lambda^2}{m}\right)^* = \frac{1}{T_{geo}} \simeq 1.2 \times 10^{-5}\ Hz$, the motion tends to be radial, otherwise, when $\frac{\lambda^2}{m} \lesssim \left(\frac{\lambda^2}{m}\right)^*$, it remains circular with decreasing radius $r(t)$ (as shown in fig.1). The typical drag rate has to be less than the circular orbit rate; in this way the satellite energy loss is spread over more time and the orbit is less deformed (zero order approximation).
From eq.5, we can compute the theoretical landing time as:

$$t_{fall}^{exp} = \frac{a}{\beta}, \text{ where } a = \frac{1}{2}\ln\left(\frac{r_0}{r_{impact}}\right) \simeq 0.95 \tag{6}$$

in fact if the drag is zero, we recover a circular orbit with $r(t) = const = r_0$ over time and $t_f^{exp} \longrightarrow \infty$, otherwise if $\frac{\lambda^2}{m} \longrightarrow \infty$ the satellite remains still at $\boldsymbol{r} = \boldsymbol{r_0}$.
For numerical purposes, we want to compute an upper limit $(\frac{\lambda^2}{m})_{lim}$ for the drag coefficient, in order to avoid a situation where the motion becomes totally radial even with an initial circular velocity: the satellite landing time has to be greater in the case of circular decreasing motion than in the case of in going radial motion with no drag $\left(t_{fall}^{exp} > t_{fall}^{rad}\right)$, assuming the same value of the initial initial energy $\frac{E}{m} = \frac{1}{2}\dot{r}^2 - \frac{1}{r} = -\frac{1}{2r_0}$.
We can recover an equation for in going $\dot{r}$, that is:

$$\dot{r} = \frac{dr}{dt} = -\sqrt{\frac{2}{r} - \frac{1}{r_0}}$$

and so an equation for $\frac{dt}{dr}$ that can be integrated from r_0 to r_{impact} to obtain t_{fall}^{rad}:

$$t_{fall}^{rad} = t(r_{impact}) = 0.49$$

The upper value is $\left(\frac{\lambda^2}{m}\right)_{lim} = \beta_{lim}\sqrt{\frac{GM_E}{r_{geo}^3}} \simeq 1.4 \times 10^{-4} Hz$, that satisfies the limit situation, in which $t_{fall}^{exp} = t_{fall}^{rad}$. The trajectory related to this limit situation is the first shown in fig.4.

In the case of the same drag coefficient $\frac{\lambda^2}{m}$, numerical time step dt, but different initial conditions:

$$\boldsymbol{r_0} = (1,\ 0,\ 0) \text{ and } \boldsymbol{v_0} = (0,\ \tfrac{1}{\sqrt{2}},\ \tfrac{1}{\sqrt{2}})$$

the trajectory is represented by the following 3-dimensional plot:

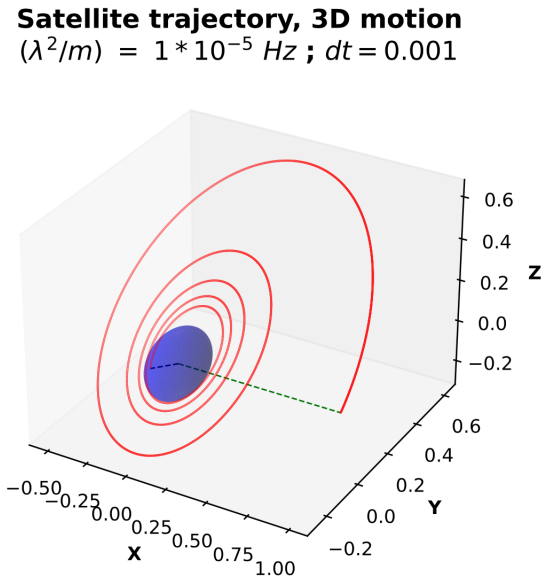
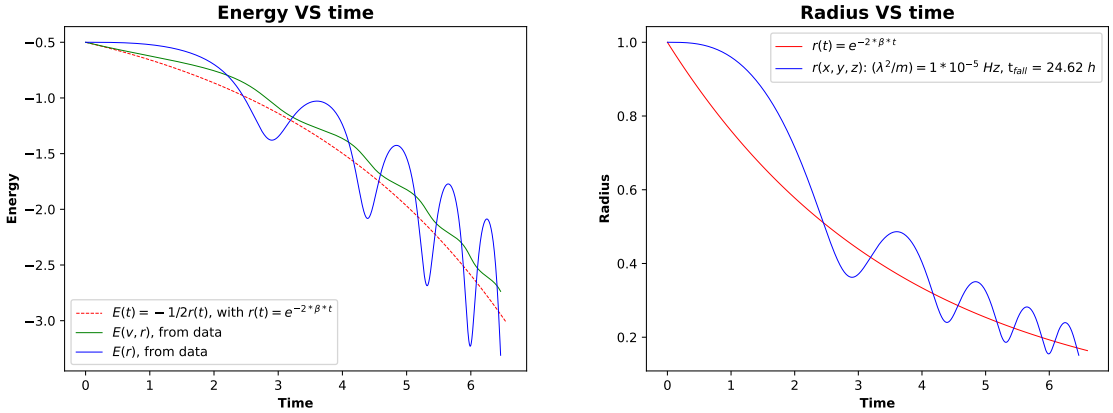


Figure 2: 3D plot, with a drag of $\frac{\lambda^2}{m} = 1 \times 10^{-5}\ Hz$ and time step $dt = 0.001$; we still have $t_{fall} = 24.62\ hours$.

Additionally, we printed to file, and therefore plotted (fig.3), the behavior of the satellite's energy per unit mass $\frac{E}{m}$ and distance r from Earth, as functions of time:



(a) Theoretical approximated behaviour (red line) of $\frac{E}{m}(t)$ and numerical results (green/blue lines).

(b) Theoretical approximated behaviour (red line) of $r(t)$ and numerical result (blue line).

Figure 3: In 3a, the "green" energy is defined as $E(v, r) = \frac{v^2}{2} - \frac{1}{r}$ while the "blue" one as $E(r) = -\frac{1}{2r}$, assuming a circular velocity $v^2 = v_{circ}^2 = \frac{1}{r}$. In fig.3b it is shown the evolution of the satellite distance from Earth over time.

As far as the energy is concerned, it can be easily observed that the "blue" energy swings above and under the theoretical trend, and its oscillations' amplitude around it grows as time passes (in fact the oscillation's amplitude of the radius decreases).
On the other hand, the "green" energy always stays above the "red" one and its oscillations' amplitude is significantly shorter than the blue one.

In order to justify this behavioral difference, we can claim that this is due to the fact that, in the first case (green energy), where we do not assume a circular motion, the integration error (on the velocity computation) is quadratic, rather than linear, and therefore the slope is always above the theoretical trend.

Since the satellite motion is planar, we show below some possible trajectories, with different drag coefficient values, in 2 dimensions. The initial conditions have always been set at:

$$\boldsymbol{r_0} = (1,\ 0,\ 0) \text{ and } \boldsymbol{v_0} = (0,\ 0,\ 1)$$

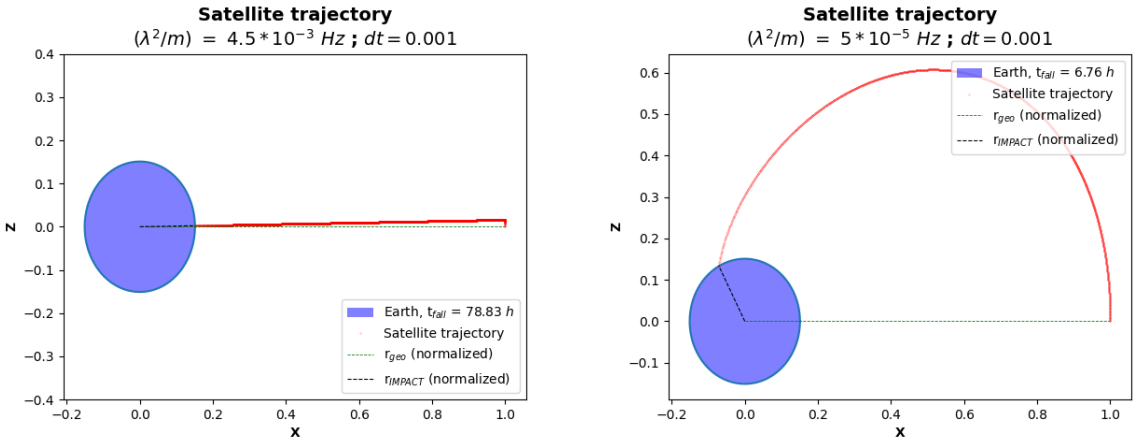


Figure 4: Trajectories with $\frac{\lambda^2}{m} = 4.5 \times 10^{-3}\ Hz$ (left) and $\frac{\lambda^2}{m} = 5 \times 10^{-5}\ Hz$ (right).

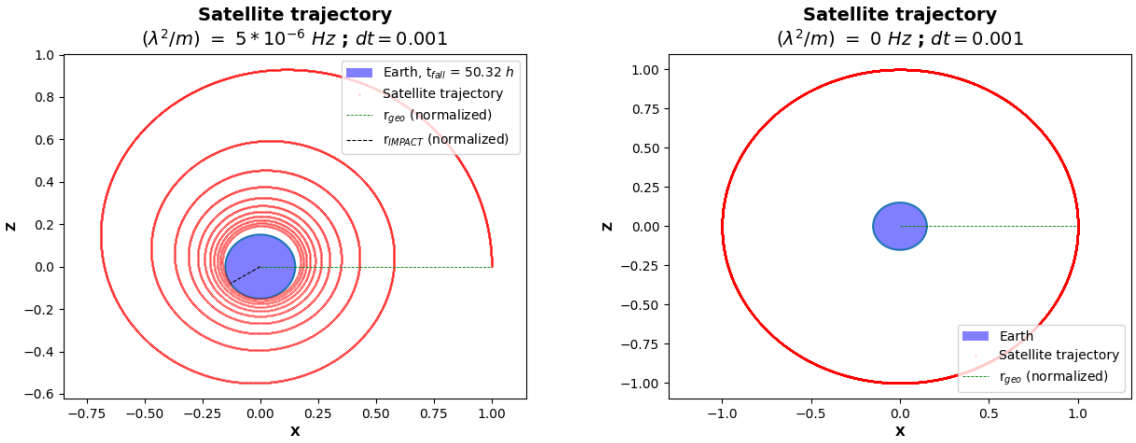


Figure 5: Satellite trajectory, with low drag (left) and no drag (right).

In fig.6, we show the trend of t_{fall} in function of $\frac{\lambda^2}{m}$, in order to study how the landing time is affected by the value of the drag coefficient, when this would be changed. Below, in tab.1, we compare the theoretical t_{fall}^{exp}, defined in eq.6, and the numerical (the one calculated by the program) t_{fall}:

$\frac{\lambda^2}{m}$ $[Hz]$	t_{fall} numerical $[hours]$	t_{fall}^{exp} theoretical $[hours]$
1×10^{-6}	259.25	262.61
5×10^{-6}	50.32	52.52
1×10^{-5}	24.62	26.26
5×10^{-5}	6.76	5.25
1×10^{-4}	6.27	2.63
1×10^{-3}	18.51	0.26
4.5×10^{-3}	78.39	0.06

Table 1: Drag coefficient value's influence on numerical and theoretical $\left(t_{fall}^{exp}\right)$ landing time.

What appears to be clear (especially by observing fig.6) is that a turning point exist (approximately around $\left(\frac{\lambda^2}{m}\right)_{lim} \lesssim 1 \times 10^{-4} Hz$) from which the theoretical and numerical time no longer precisely coincide. Indeed, before $\left(\frac{\lambda^2}{m}\right)_{lim}$ the satellite's orbits are well approximated by decreasing circular ones and the landing time is well predicted by eq.6. From this point, the motion of the satellite tends to be increasingly radial (and slow) as the drag coefficient increases. We notice that our numerical result of $\left(\frac{\lambda^2}{m}\right)_{lim}$ is near to $4.5 \times 10^{-3}\ Hz$, which is our qualitative estimation made in the *drag coefficient box*.

We show the results of the comparison in figure 6:

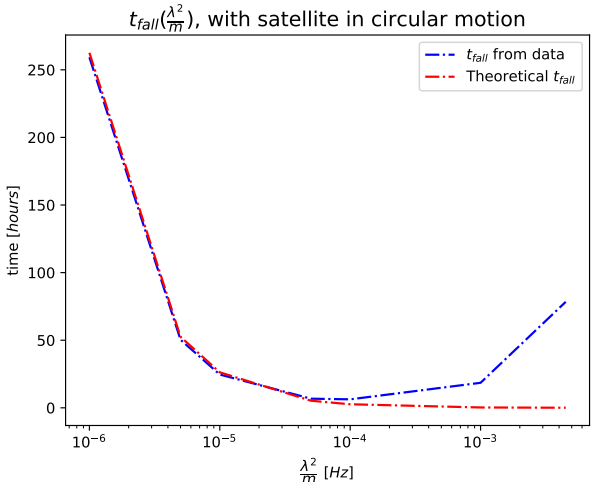


Figure 6: Comparison between theoretical (red dashed) and numerical (blue dashed) landing time trend.

In order to understand the values in the region where $\frac{\lambda^2}{m} > \left(\frac{\lambda^2}{m}\right)_{lim}$ we need to study the radial problem.

2.3 Satellite radial solution

In this case with $\dot{m} = 0$, the equation of motion has a scalar form:

$$\boxed{\ddot{r} = -\frac{1}{r^2} - \beta\dot{r}} \tag{7}$$

In fig.7, we show the trend of $r(t)$ with initial outgoing radial velocity $v_r = 1$ and how the time flight is affected by the value of the drag coefficient. Below, in tab.2, we display the amounts of time in which the satellite keeps moving away from Earth (Δt_+, with $v_r > 0$), towards Earth (Δt_-, with $v_r < 0$) and the total falling time ($t_{fall} = \Delta t_+ + \Delta t_-$).

$\frac{\lambda^2}{m}$ $[Hz]$	Δt_+ $(v_r > 0)$ $[hours]$	Δt_- $(v_r < 0)$ $[hours]$	t_{fall} $[hours]$
1×10^{-8}	9.79	11.86	21.65
1×10^{-7}	9.76	11.83	21.59
1×10^{-6}	9.47	11.65	21.12
1×10^{-5}	7.48	10.38	17.86
1×10^{-4}	3.16	8.47	11.63
1×10^{-3}	0.77	21.67	22.46
5×10^{-3}	0.24	90.63	90.87
1×10^{-2}	0.14	177.32	177.46
1×10^{-1}	0.02	1738.71	1738.73

Table 2: Drag coefficient value's influence on time flight.

Initially, we observe that as the value of $\frac{\lambda^2}{m}$ increases (until 1×10^{-4} Hz), the falling time (t_{fall}) of the satellite tends to decrease. This is due to the fact that a higher value of $\frac{\lambda^2}{m}$ corresponds to a greater drag force acting on the satellite during its outgoing flight. As a result, the satellite encounters greater atmospheric resistance and loses outgoing velocity more rapidly, leading to a reduction in the outgoing flight time.

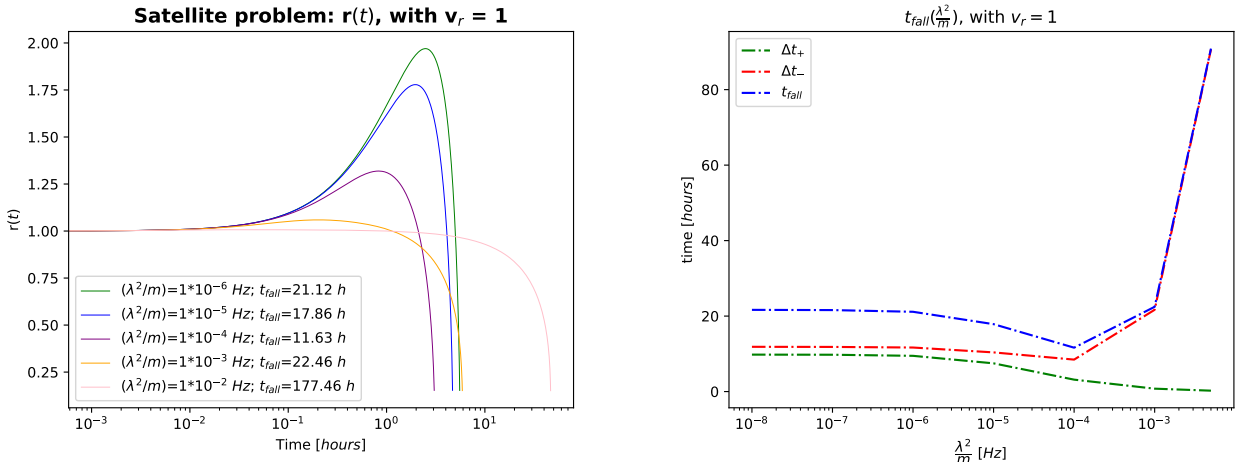


Figure 7: $r(t)$ slope on the left and $t_{fall}\left(\frac{\lambda^2}{m}\right)$ on the right, for initial outgoing $v_r = 1$.

There exists a turning point $\left(\frac{\lambda^2}{m}\right)_{min}$ (in fig.7, on the right, approximately between 10^{-4} Hz and 10^{-3} Hz), where t_{fall} reaches a minimum value and starts increasing as $\frac{\lambda^2}{m}$ continues to grow. This behavior can be explained by the fact that a very high value of the drag coefficient results in a significant drag force that considerably slows down the satellite, even in the phase of motion where $v_r < 0$. Consequently, the satellite spends more time at lower velocities and this prolongs the time Δt_- required to reach the Earth surface. Strictly speaking, $\Delta t_+\left(\frac{\lambda^2}{m}\right)$ is a monotonic decreasing function towards zero, whereas $\Delta t_-\left(\frac{\lambda^2}{m}\right)$ has a minimum.

Below, in fig.8, we show the trend of $r(t)$ with null and in going initial velocity:

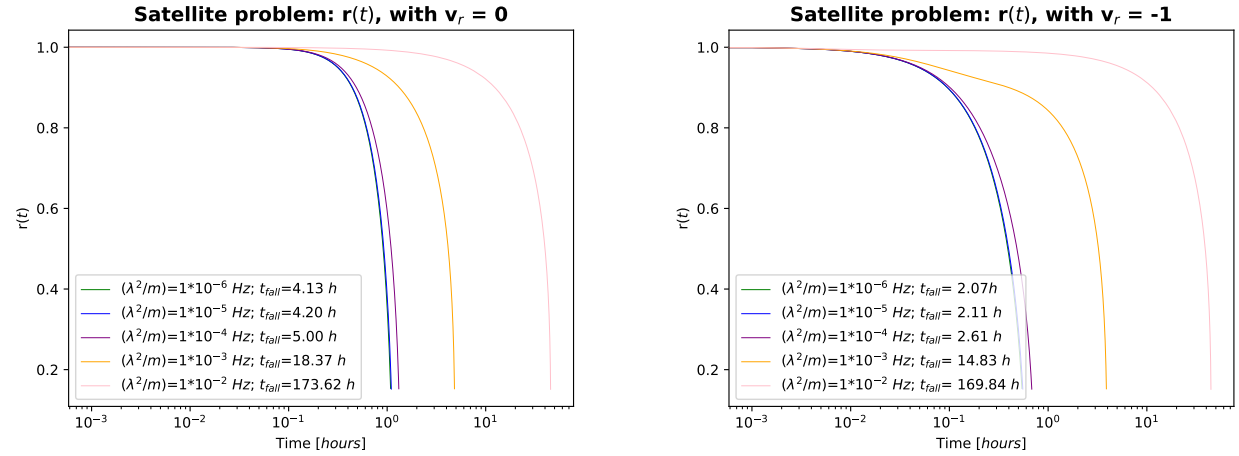


Figure 8: $r(t)$ slope, on the left with initial $v_r = 0$, on the right with $v_r = -1$.

In this case $t_{fall} = \Delta t_-$ and it always increases as $\frac{\lambda^2}{m}$ increases.

2.4 Satellite mass loss

In this section we will focus on the satellite motion, comparing its trajectory with $\dot{m} = \frac{dm}{dt} = 0$ and with $\dot{m} = const \neq 0$ (which means that the satellite loses mass as it approaches Earth), under the assumption of a 1^{st} order loss: $m(t) = m_0 + \dot{m}t$ with $\ddot{m} = 0$.
In this case, the equation of motion is represented by dimensionless eq.3, where $\beta(t)$ can be positive or negative depending on the choice of $\dot{m}$ value:

$$\ddot{\boldsymbol{r}}' = -\left(\frac{1}{r'^3}\right)\boldsymbol{r}' - \beta(t)\dot{\boldsymbol{r}}'$$

Mass loss

In order to interpret our numerical results, we need to distinguish between two cases, always assuming $\dot{m} < 0$:

- $\beta(t) > 0 \rightarrow |\dot{m}| < \lambda^2$

  This is a very restricting case in which we have low relative mass loss during the falling, in fact $\frac{\Delta m}{m_0} = \frac{|\dot{m}|}{m_0}\ \Delta t$ and the less is $\dot{m}$, the larger has to be our integration time Δt in order to observe a significant mass loss, otherwise we recover a very similar trajectory to the one with $\dot{m} = 0$.
  Furthermore this is also the same type of problem discussed before, where the acceleration is proportional to $-\boldsymbol{v}$, and the radius of the orbit decreases with time, with a slightly different drag than $\frac{\lambda^2}{m_0}$.

- $\beta(t) < 0 \rightarrow |\dot{m}| > \lambda^2$

  This is a more realistic case, where we can change $\dot{m}$ *a posteriori* independently from the value of the drag coefficient, and reproduce a more realistic mass loss in the satellite journey. In this situation, the acceleration is proportional to $\boldsymbol{v}$ and so the radius of the orbit increases with time instead of decrease.

Firstly we want to rewrite the dimensionless time dependent coefficient $\beta(t)$:

$$\beta(t) = \frac{\beta_{new}}{\alpha(t)}\sqrt{\frac{r_{geo}^3}{GM_E}} \propto \frac{1}{\alpha(t)}$$

where $\beta_{new} = \frac{\lambda^2}{m_0} + \frac{\dot{m}}{m_0} = constant$ and $\alpha(t) = 1 + \frac{\dot{m}}{m_0}t < 1\ \forall t$.
So $\beta(t)$ is an increasing function of time during the satellite fall and its sign depends on the sign of β_{new}.
Using this notation, we can also express the energy per unit mass as:

$$\frac{E(t)}{m(t)} = \frac{E(t)}{m_0} \cdot \frac{1}{\alpha(t)}$$

and in fact if $\alpha(t) = 1$ we recover the expression of the energy in the case of only drag coefficient ($\dot{m} = 0$).
We mention also that in the limit of $\frac{|\dot{m}|}{m_0}t \ll 1$ (small relative loss of mass) we recover the situation where $\beta(t) \simeq \beta_{new}$, and $\frac{\dot{m}}{m}$ represents only a constant negative correction to the drag coefficient $\frac{\lambda^2}{m}$ in a motion where $m \simeq const$. In this case we recover, approximately, the solution in eq.5 in which $\beta = \beta_{new}$:

$$r(t) = e^{-2\beta_{new}t}$$

with the assumption of circular velocity[3].

2.4.1 $\beta_{new} < 0$

- $\frac{\lambda^2}{m_0} = 0$

  For simplicity, we begin our analysis without the presence of the atmospheric drag. We want to test the validity of our numerical code and, assuming this and $\beta_{new} < 0$, we expect an increasing trajectory radius. We show, in the following fig.9 and 10, the trajectories of the satellite with four different $\frac{\dot{m}}{m_0}$ values:

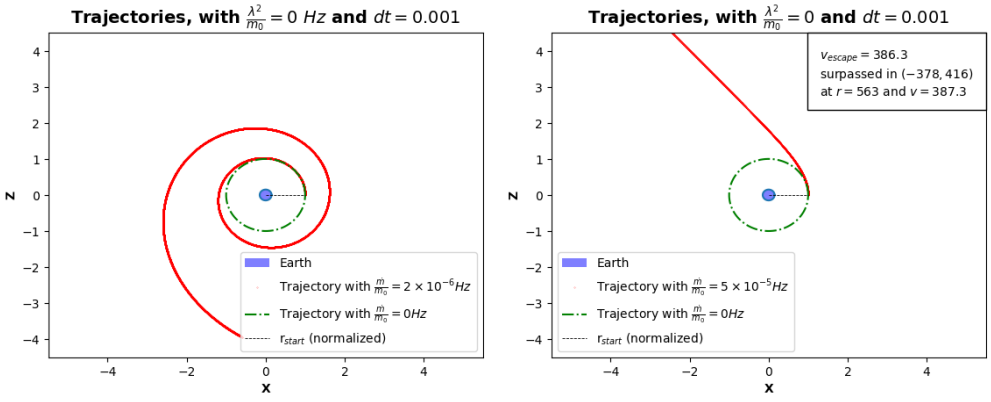


Figure 9: Respectively, from left to right, $\left|\frac{\dot{m}}{m_0}\right|$ is increasing from $2 \times 10^{-6}\ Hz$ to $5 \times 10^{-4}\ Hz$.

[3]See subsec.2.2

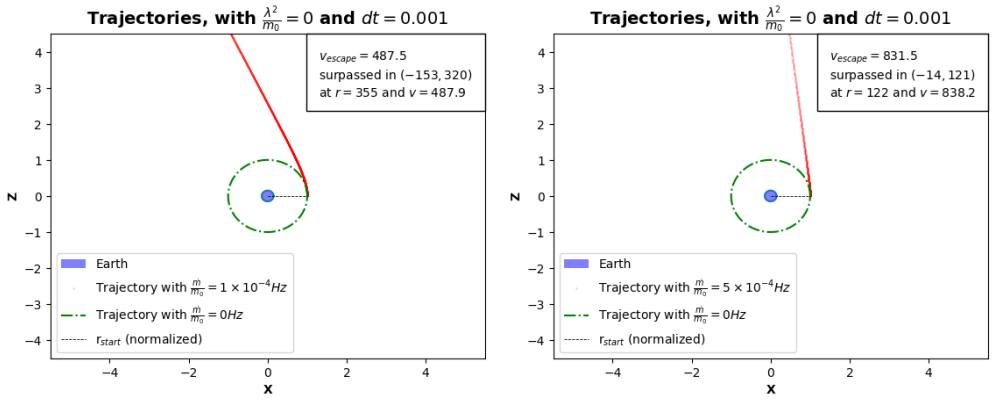


Figure 10: Respectively, from left to right, $\left|\frac{\dot{m}}{m_0}\right|$ is increasing from 1×10^{-4} Hz to 5×10^{-4} Hz.

We notice how an increment in the mass loss affects the curvature of the trajectory. It tends to be straighter and parallel to the initial circular velocity as $\left|\frac{\dot{m}}{m_0}\right|$ increases.
The same qualitative argument for the choice of the drag coefficient value (sec.2.2) could be generalized also for the choice of the relative mass loss per unit mass. In fact the equation of motion is exactly the eq.4 a part from the fact that $\beta = \beta_{new}$ is negative.

Considering $\left|\frac{\dot{m}}{m_0}\right| \lesssim \left|\frac{\dot{m}}{m_0}\right|^* \simeq 1.2 \times 10^{-5}$ Hz, the radial solution is well approximated by an increasing exponential function (left fig.9), but if $\left|\frac{\dot{m}}{m_0}\right| > \left|\frac{\dot{m}}{m_0}\right|^*$ our approximation is no more valid and the motion enters in a new regime (right fig.9 and fig.10). The radial gravity acceleration is dominated by the components parallel to $\boldsymbol{v}$ and in the limit of $\left|\frac{\dot{m}}{m_0}\right| \gg \left|\frac{\dot{m}}{m_0}\right|^*$ the trajectory would be a straight line going outwards with increasing velocity.
The system is no more bound, in fact $r(t) \to \infty$ and we compute the point where the velocity becomes greater than the escape velocity (top right in the figures); the radial distance from this point decreases as $\left|\frac{\dot{m}}{m_0}\right|$ increases (read the end of the section for a more accurate evaluation on the escape velocity).

- $\frac{\lambda^2}{m_0} \neq 0$

  In the simpler case, where $\frac{\lambda^2}{m_0} = \left|\frac{\dot{m}}{m_0}\right|$ we get $\beta_{new} = 0$ and we expect a circular motion where the effect of the mass loss cancel the loss of energy due to the drag. This is another aspect that confirms the validity of our numerical integration and in fig.11 we show the trajectory:

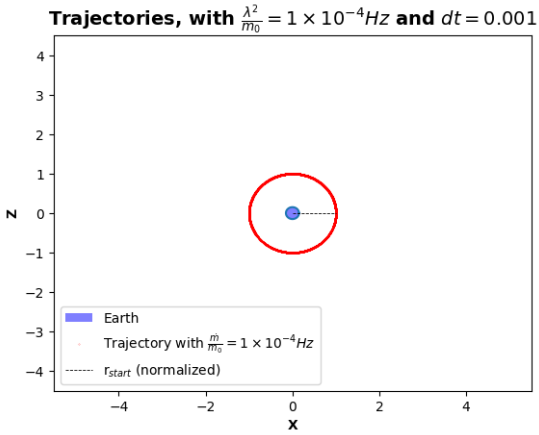


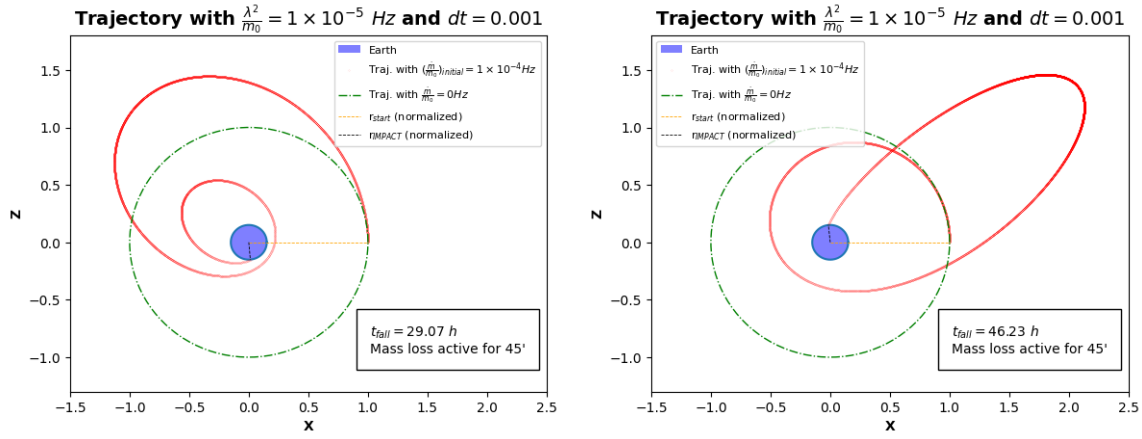
Figure 11: Circular motion with atmospheric drag and mass loss

Our last scope is studying the case when the two contributes (mass loss and drag) are different, and trying to understand how they affect the motion together.

- $\frac{|\dot{m}|}{m_0} > \frac{\lambda^2}{m_0}$

We finally studied the case in which the value of $\left|\frac{\dot{m}}{m_0}\right|$ is greater than the drag coefficient value per unit mass. So, we focused on **two different cases**.

In the **first case**, we activated an isotropic mass loss (per unit mass) for only 45 minutes and studied how this would affect our satellite trajectory. What happened is shown in figure 12.



(a) Mass loss activated for the first 45 minutes of flight.

(b) Mass loss activated for 45 minutes, after the 10^{th} hour of flight.

Figure 12: Trajectories, with $\frac{\lambda^2}{m_0} = 1 \times 10^{-5}\ Hz$, and $\left|\frac{\dot{m}}{m_0}\right| = 1 \times 10^{-4}\ Hz$, activated in different flight moments.

In fig.12a the satellite starts moving away from Earth ($\beta_{new} = \frac{\lambda^2}{m_0} + \frac{\dot{m}}{m_0} > 0$), until the mass loss is deactivated and, therefore, it starts falling, since the value of β_{new} becomes negative.
In fig.12b the satellite starts falling toward Earth ($\beta_{new} < 0$); after 10 hours $\left|\frac{\dot{m}}{m_0}\right|$ is activated, so the satellite starts moving away from Earth ($\beta_{new} > 0$), until the mass loss is once again deactivated and β_{new} returns negative.

If the satellite reaches the escape velocity, at the time when β_{new} is positive, it will obviously never come back. The **second case** we studied is indeed when the mass loss is activated at a certain time and never deactivated, until the satellite would lose the 10 % of its mass.

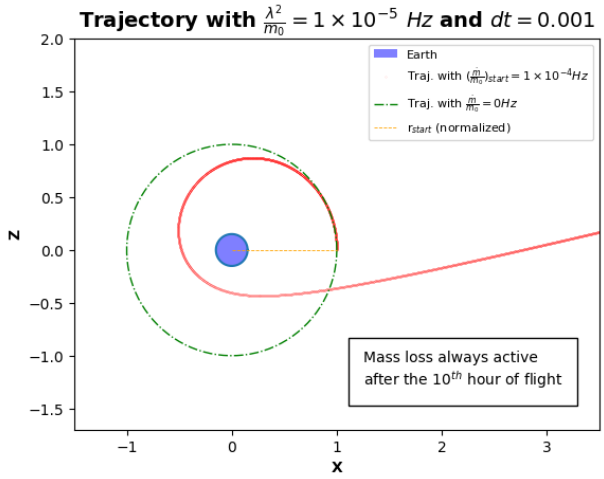


Figure 13: $\frac{\lambda^2}{m_0} = 1 \times 10^{-5}\ Hz$, and $\left|\frac{\dot{m}}{m_0}\right| = 1 \times 10^{-4}\ Hz$, active from the 10^{th} hour on.

We note that, after the activation of the mass loss, the satellite's velocity becomes greater and greater as the satellites moves forward from Earth. In fact the gravitational attraction $\frac{1}{r^2}$ decreases with time while $\beta(t)$ is increasing, the acceleration becomes parallel to the outgoing velocity.
At some time the latter could become greater than the escape velocity and so the satellite would not come back. However this situation is non trivial. After the loss of the the 10 % of mass, $\beta_{new} > 0$ and there is energy dissipation again, $v_{esc} = \sqrt{2U}$ is only an under estimate.
In order to estimate the escape velocity more precisely, we would have to simulate our motion (with a fixed

value of $\frac{\lambda^2}{m_0}$) varying the initial velocity until finding a motion where $\dot{r}(t) > 0 \ \forall t$ (see fig.7). At this point we would encountered another problem which is the fact that our drag is non zero also at infinity, and there would always be a time where the energy returns < 0. Fortunately, in a real astrophysical situation this is not true and $\frac{\lambda^2}{m_0} = 0$ after a certain r^* (beginning of empty space).

3 N-Body numerical solutions

In order to check the validity of our numerical code, simulating the gravitational interaction between N-body, we need to compare our results with trajectories that can be obtained and discussed easily. This is the case of the 2-body problem (sec.3.1), but also the case of a typical and simple 3-body problem called "Pythagorean 3-body problem" (sec.3.2).

3.1 2-Body

The equations of motion can be written as eq.2, where $i, j = 1, 2$:

$$\ddot{\boldsymbol{r}}_{\boldsymbol{i}} = \boldsymbol{F}_{\boldsymbol{i}} = \sum_{\substack{j=1 \\ i \neq j}}^{2} \frac{\boldsymbol{r}_{\boldsymbol{j}} - \boldsymbol{r}_{\boldsymbol{i}}}{r_{12}^3} \tag{8}$$

where we use a dimensionless re-scaling where $G = m_i = m_j = 1$ ($\forall \ i, j$), in order to obtain dimensionless results for the positions and velocities.
The aim of our integration is to find the trajectories $\boldsymbol{r}_{\boldsymbol{i}}$ with different initial conditions $\boldsymbol{r}_{\boldsymbol{i}}(0)$ and $\boldsymbol{v}_{\boldsymbol{i}}(0)$, solving the system of two second order differential equations.

We will show some plots and link some videos, with different initial velocities, that represent the evolution over time of our 2 bodies' planar motion.
The initial positions and velocities (in two dimensions) of the two bodies are:

$$\boldsymbol{r}_{\boldsymbol{0}1} = (1, 0) = -\boldsymbol{r}_{\boldsymbol{0}2}$$
$$\boldsymbol{v}_{\boldsymbol{0}1} = -\boldsymbol{v}_{\boldsymbol{0}2}$$

We can compute the initial energy of the system as:

$$E = \frac{1}{2}v_1^2 + \frac{1}{2}v_2^2 - \frac{1}{r_{12}} = v^2 - \frac{1}{r_{12}} \ , \text{ where: } r_{12} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + \epsilon^2} \tag{9}$$

where ϵ is the softening parameter.
The initial velocities of the bodies have to be $v_i < v_{esc} = \sqrt{\frac{1}{r_{12}}} \simeq 0.7$, otherwise the system is not bounded.

Softening parameter

We decided to choose $\epsilon = 0.4$, in order to avoid the collision between the two bodies and control the acceleration in the UV phases of motion. In fact if $r_{12} \to 0$ the potential U_i diverges and so the accelerations a_i; the motion is no more controlled by our integration.
Actually, the value of ϵ is connected with our numerical time step dt. As discussed in sec. 4.2, the velocity computed by *leapfrog* depends on the term $a(t)dt$, and we could decrease dt in order to afford a higher value of the acceleration (smaller ϵ) and study a more realistic interaction.
We decided to use $dt = 0.1$ which permits us to represent our sketch of the motion in a more faster way. After some trials we realize that the value of $\epsilon = 0.4$ is the lower value which let our integration for the 2-body problem to be stable.

3.1.1 Displaying results

Below, we show how the system evolves when the initial velocity value v_i would be changed.

- Null initial velocities ($v_{0i} = 0$, $\forall i = 1, 2$)
  The motion is on *YouTube* video linked.
- Bound initial velocities ($v_{0i} = 0.3$, $\forall i = 1, 2$)
- Circular initial velocities ($v_{0i} = 0.5$, $\forall i = 1, 2$)
  and the full motion is on *YouTube* video linked.

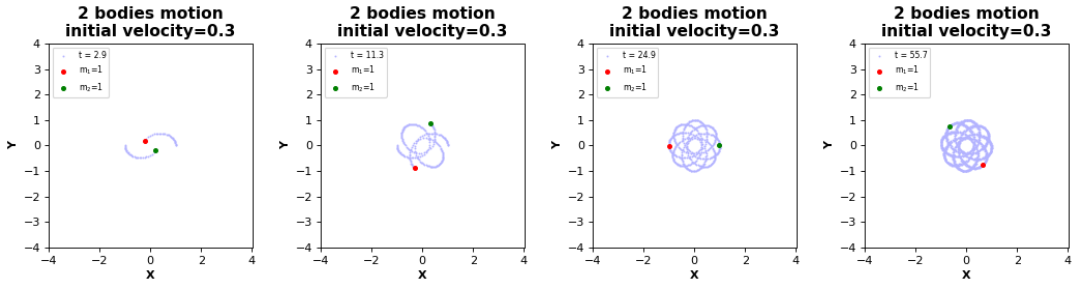


Figure 14: 2-body sketches of the motion over time (top left) with $v_{0i} = 0.3$.

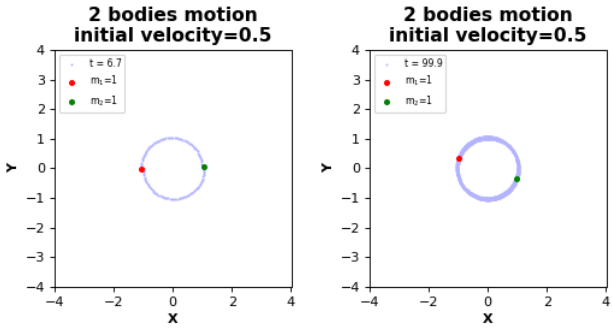


Figure 15: 2-body sketches of the motion over time with $v_{0i} = 0.5$.

- Bound initial velocities ($v_{0i} = 0.6$, $\forall i = 1, 2$)

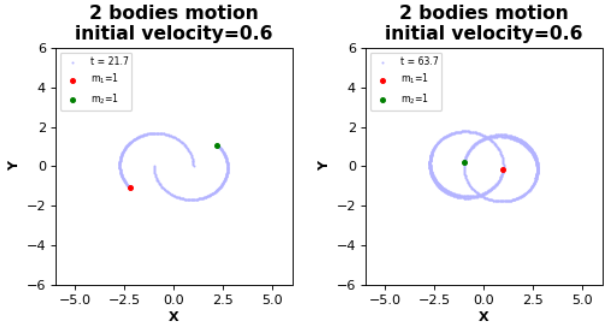


Figure 16: 2-body sketches of the motion over time with $v_{0i} = 0.6$

- Escape initial velocities ($v_{0i} = 0.8$, $\forall i = 1, 2$)

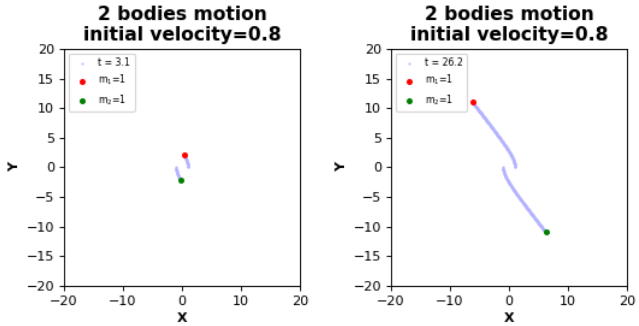


Figure 17: 3-body motion with $v_{0i} = 0.8$.

In this case, we have: $v_{0i} = 0.8 > v_{esc} \simeq 0.7$. The system is indeed not bounded.

3.2 3-Body

In this case the dimensionless equations of motion are:

$$\ddot{\boldsymbol{r}}_{\boldsymbol{i}} = \boldsymbol{F}_{\boldsymbol{i}} = \sum_{\substack{j=1 \\ i \neq j}}^{3} \frac{\boldsymbol{r}_{\boldsymbol{j}} - \boldsymbol{r}_{\boldsymbol{i}}}{r_{12}^3}, \text{ where } i, j = 1, 2, 3 \tag{10}$$

The initial positions (in two dimensions) of our three bodies satisfies the "Pythagorean 3-body problem", in fact the bodies are initially at the vertices $\{(0,0),(3,0),(0,4)\}$ of the Pythagorean triangle, and with null velocities. We decided to use, after some trials, $\epsilon = 0.5$ and $dt = 0.1$.

The theoretical behavior we tried to reconstruct is shown in *this* linked video.
Our simulation ($v_i = 0$, $\forall i = 1, 2, 3$) is displayed in this *YouTube* video linked and we show a sketch of it down below:

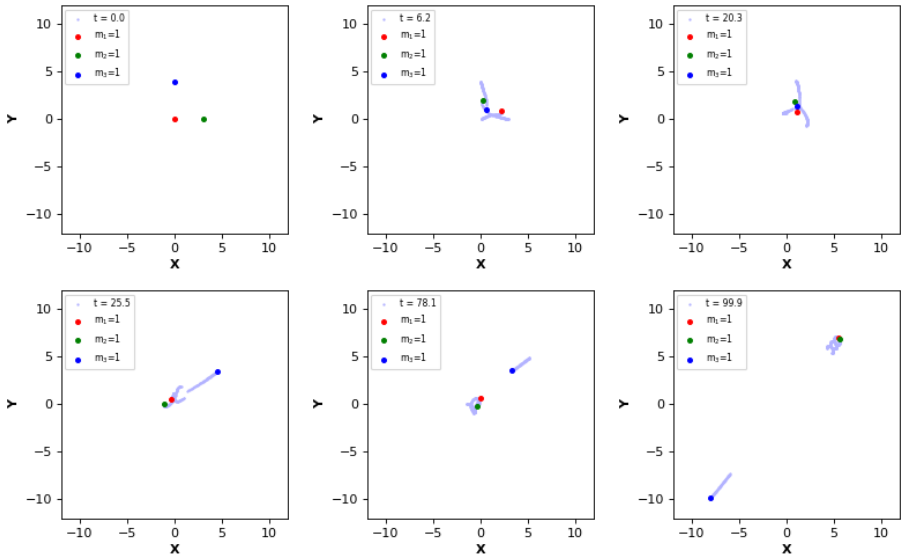


Figure 18: Motion of 3 bodies with $v_i = 0$.

The motion of three free bodies in gravitational interaction is one of the simplest examples of chaotic behavior in nature. Such behavior is characterized by complex non-periodic orbits, which usually exhibit high sensitivity to small variations in initial conditions, a feature known as the *butterfly effect*.

As in all 3-body problems, one of the bodies is eventually ejected (like the blue one in fig.18) from the system, while the other two remain stuck together forever.

3.3 Dimensionless re-scaling

In order to deal with an equation which has to be integrated numerically with *leapfrog*, we need to re-scale the equations of motion using dimensionless variables, which we define as:

$$\begin{cases} m' \coloneqq \dfrac{m}{M_\odot} \\ \boldsymbol{r'} \coloneqq \dfrac{\boldsymbol{r}}{ly} \\ t' \coloneqq \sqrt{\dfrac{GM_\odot}{ly^3}}\, t \simeq (9.72 \times 10^{-10} Hz) \cdot t \end{cases}$$

where $ly \simeq 9.461 \times 10^{12}\ km$ is the *light year*[4], and $M_\odot \simeq 1.989 \times 10^{30}\ kg$ is the *Solar mass*.

We can also obtain the dimensionless kinetic, potential and total energy of the system:

$$\begin{cases} K' = \dfrac{1}{2} \displaystyle\sum_{i=1}^{N} m_i'\, \dot{r}_i'^2 \\ U' = -\dfrac{1}{2} \displaystyle\sum_{\substack{i,j=1 \\ i \neq j}}^{N} \dfrac{m_i'\, m_j'}{|r_i' - r_j'|} \\ E' = K' + U' \end{cases}$$

[4]For an open cluster the typical radius R is about $10 \div 20\ ly$ and $t_{cross} = \sqrt{\frac{R^3}{GM_\odot}} \simeq 3.25 \times 10^{10}\ s \simeq 1000\ years$ is the typical crossing time

where, with the dot on top of r', we mean the derivative with respect to t'.

3.4 Virial theorem

If we consider a system made up of N bodies, each of them having a position $\boldsymbol{r}_i$ (with $i = 1, .., N$), a $\boldsymbol{v}_i = \dot{\boldsymbol{r}}_{\boldsymbol{i}}$ velocity and a mass m_i, we can compute the moment of inertia of the system I:

$$I = \sum_i m_i \ \boldsymbol{r_i} \cdot \boldsymbol{r_i} \tag{11}$$

If we compute its first derivative with respect to time, $\dot{I} = 2\sum_i m_i \ \dot{\boldsymbol{r}}_{\boldsymbol{i}} \cdot \boldsymbol{r_i}$, we can define a *collapsing system* as a system for which we have $\dot{I} < 0$. On the other hand a system with $\dot{I} > 0$ is said to be *expanding*.

As states before, if we assume the existence of a *potential* for this N-body system, such that $\boldsymbol{F_i} = \boldsymbol{\nabla}_i U$, we can easily show (taken the second Newton's law):

$$\sum_i m_i \ddot{\boldsymbol{r}}_{\boldsymbol{i}} \cdot \boldsymbol{r_i} = \sum_i \boldsymbol{r_i} \cdot \boldsymbol{\nabla}_i U \tag{12}$$

which, after some computation can be written as:

$$\frac{1}{2}\ddot{I} = 2K + U \tag{13}$$

being $K = \frac{1}{2}\sum_i m_i \ \dot{\boldsymbol{r}}_{\boldsymbol{i}} \cdot \dot{\boldsymbol{r}}_{\boldsymbol{i}}$ the kinetic energy of the system, U the total potential energy, and $\ddot{I}$ the second time derivative of the momentum of inertia. This is know as the *virial theorem*, and it gives information about the state of the system. If we define the *virial ratio* as:

$$\boxed{Q \coloneqq \frac{2K}{|U|}} \tag{14}$$

we say a system is *virialized* if $\ddot{I} = 0$, then it is in equilibrium and $Q = 1$. So we can summarize the state of the system with respect to the value of the virial ratio:

$$\begin{cases} Q > 1 & (expansion) \\ Q < 1 & (compression) \\ Q = 1 & (equilibrium) \end{cases}$$

The value of Q can change over time, as the system evolves. In our analysis, we will show how the virial ratio of a N-body system can oscillate and finally increase (or decrease) up to a value of $Q \approx 1$, as a gravitational system is supposed to reach an equilibrium state after some integration time, evolving to a virialized system.

In addition, it can be shown that in a *bounded system* such as a gravitational one, where the total energy $E = K + U$ is negative, the relation $Q \leq 2$ is necessary.

3.4.1 Softening parameter

As mentioned before (sec.3.1), it is useful to introduce a softening parameter in the definition of the potential of the system, in order to avoid the so called *UV problem*. We introduce a parameter ϵ, such that the potential energy between m_i and m_j becomes:

$$U_{ij} = -\frac{m_i \ m_j}{\sqrt{r_{ij}^2 + \epsilon^2}} \tag{15}$$

being r_{ij} the relative distance between the two point mass.

For our purposes, we chose to use a value for ϵ such that:

$$\boxed{\epsilon = \frac{R}{2\sqrt[3]{N}}} \tag{16}$$

where R is the radius of the initial homogeneous density sphere, as discussed in sec. 3.5. The softening parameter in this assumption is then equal to half of the *average closest distance*. We have to remember that the softening parameter represents a "lower limit", in order to avoid a major value of the gravitational potential which would cause a greater value of the acceleration to be computed, leading to a possible decrease in the integration's efficiency.

The factor 2 act as a "safety correction": since $\langle d \rangle = \frac{R}{\sqrt[3]{N}}$ is the real *average closest distance*, this is one possible way of including also bodies which appear to be nearer than $\langle d \rangle$.

3.5 Initial Conditions

We have chosen to study the case of constant spherical mass density, specifically, with all masses initialized as $m_i' = 1$, the density corresponds to $\rho = \frac{M}{V}$, so to find the **initial positions** of our problem, we have to deal with spherical symmetry and $\rho(r) = \rho_0$. To get the distribution on a sphere, where the density is only dependent by the distance from the center r, we start by computing the so called *Cumulated Mass Distribution* $F(< r) = \frac{M(r)}{M}$. Defined this way, the *CMD* has values between $0 \leq F \leq 1$.

$$F(< r) = \frac{\int_0^r \rho(r')r'^2 dr'}{\int_0^R \rho(r')r'^2 dr'} \tag{17}$$

where R is the radius of the sphere. And in the case of constant density, after some simple computations it becomes: $F(< r) = \frac{r^3}{R^3}$.

Using this method we obtain a simple law to sample the positions of the N bodies:

$$r = \sqrt[3]{F}R \tag{18}$$

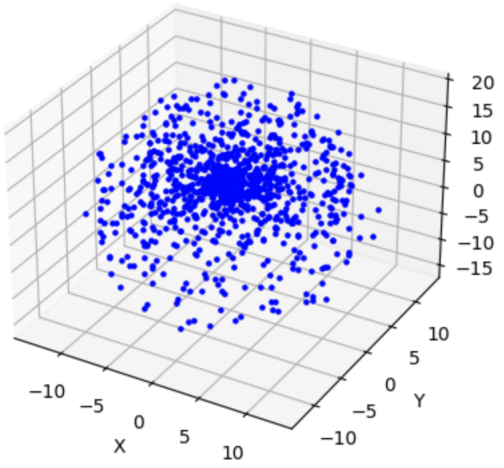


Figure 19: Initial positions of $N = 1000$ bodies on a sphere of radius $R = 10$.

After the positions, we also have to determine the **initial velocities** of our problem. In this case we are not taking total random velocities, because we would have many "escapers". In order to avoid this problem, we decided to initially sample the velocities with values between $0 \leq v' < v'_{esc}$, where v'_{esc} is the velocity needed by a particle to escape from the gravitational potential to which it is subject. The *escape velocity* so is defined as:

$$v'_{esc_i} = \sqrt{2U'_i}$$

We also decided to re-scale all the velocities using an initial *virial value* Q_{ini}, in fact we choose our effective velocity as $v = \sqrt{\frac{Q_{ini}}{Q'}}v'$. In this way, we could see how the system evolves, using different initial conditions, and see if its behaviour agrees with the theoretical expectations. In particular the initial value of Q could be set at $Q_{ini} = 0.001$, to have an initial phase of *compression*.

Escaper correction

When, in an N-body simulation, we are studying quantities such as the energy (E, U, K) we usually have to deal with the problem of the **escapers**. Despite the initial conditions, indeed, there will inevitably be particles that, during the evolution of the system, will tend to escape, and it is important to take this into account when computing the energies.

In this case, we decided to count as an "escaper" a particle which has a total energy $E_i \geq 0$ and a distance from the center of the system greater than k times the average distance from the centers of all the particles but the escapers, also known as *half mass distance* (see sec.3.6.2) We are imposing a double implication because just the condition of $E_i \geq 0$ is not sufficient. If we think, for example, of a particle in the inner regions, the fact that it has positive energy doesn't necessary mean that it will escape, since it may interact with other particles and lose energy.

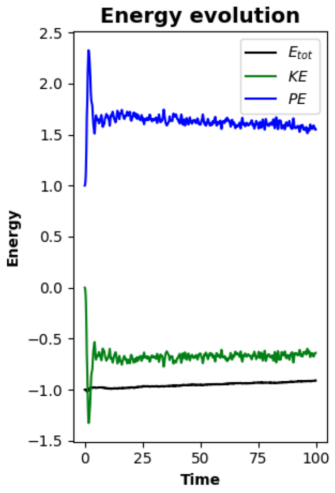
By simple computation, under the assumption of homogeneous density, if all the particle are enclosed on a sphere of radius R, the half mass distance is $r_{hm} = \frac{R}{\sqrt[3]{2}} \approx 0.8 \cdot R$. So we are neglecting the contribution of all particles with distance of $r_{escapers} > k \cdot r_{hm} \approx 0.8 \cdot k \cdot R$ and with a positive energy.

This correction is also very important while studying the evolution of $Q(t)$, since it is defined as $Q = \frac{2K}{|U|}$. Therefore, counting the energy of every single particle of the system will lead to a greater value of Q, since an *escaper* has a negligible potential energy U, respect to the kinetic energy K.

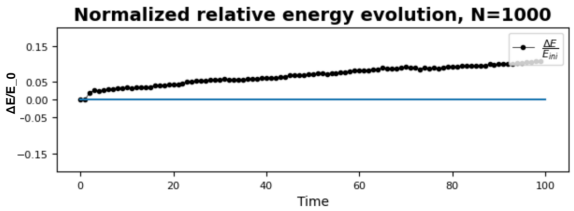
3.6 Numerical results

3.6.1 1000-Body

The first thing to check in a simulation like this is the behaviour of the energy. We know from the theory that the total energy should remain constant, but since the code has finite time steps we will have some fly-by errors, due to integration error when two particles being too close. This process will increase the total energy of the system, so depending on the initial conditions, we could have different evolutions of the total energy.



(a) Evolution of the normalized kinetic (green), potential (blue), and total energy (black).



(b) Plotting the relative energy error evolution over time: $\frac{E_0 - E(t)}{E_0}$.

Figure 20: 1000-body simulation, with $R = 10$, $dt = 0.1$ which corresponds to 30 *years*. All masses are equal: $m_i = 1.0$, $\forall i = 1, ..., N$; the total simulation time is $t_{sim} = 100$, which corresponds to 30 000 *years*. The initial value of Q is 0.8.

In the previous image 20a the normalized energies are computed dividing by the absolute value of the initial total energy. In the right plot 20b, we computed the *relative energy error* evolution, studying $\frac{\Delta E}{E_0}$ as a function of time, to evaluate the accuracy of the integration algorithm. It can be noted that the error appears to grow as time passes: at $t_{sim} = 100.0$, it reaches a value of about[5] 15%.

[5]It's important to underline that the *escaper correction* is not taken into account, so all the energies shown are the total energies, counting the energy of every single particle.

During the temporal evolution, we expect the system, initially under compression, to rapidly increase the value of Q and oscillate around a value of the virial ratio, such that the system becomes virialized, $Q_{vir} = 1$. That behaviour is the same even if we change the initial value of $Q_{ini} \leq 1$. If we start with a value of $Q_{ini} > 1$, the system has otherwise an initial expansion.

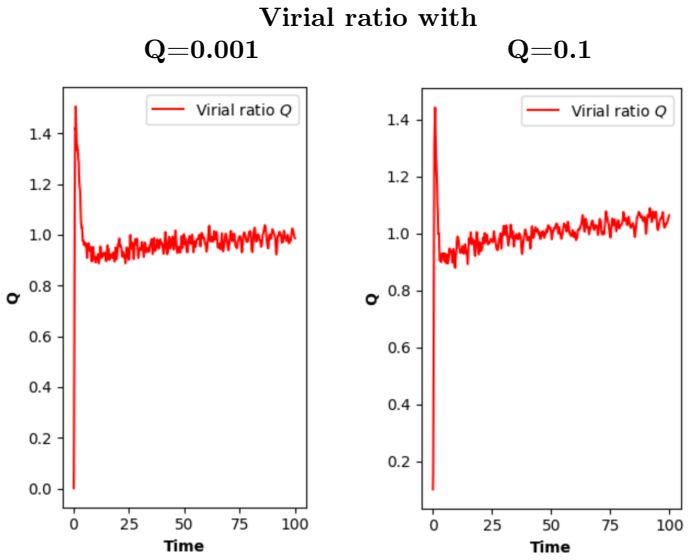


Figure 21: Evolution of virial ratio, defined in eq.14, of a 1000-body simulation, with $R = 10$, $dt = 0.1$. In the first case, the initial virial ratio value is $Q = 0.001$; in the second case we have $Q = 0.1$.

In this plot we can see that for t_{sim} approaching to 100, the value of Q is a little bigger than 1, this behaviour can be explained by a non proper choice of the value k, used to estimate the escaper correction. In this simulation we used a value of $k = 2.0$, and as we will see in the next section, this is a little overstimate of the value.

3.6.2 Computing the escapers

In this stage of our N-body problem computation, our focus shifted towards simulating various system evolution by employing different criteria for determining escape distances. The objective was to investigate the behavior of the Q factor as we varied the distance (relative to the r_h value) used to identify escapers, with the aim of finding the criteria that best replicated the behavior of a well-virialized system (with $Q \approx 1$).

In our program, we also recorded the total number of escapers at the conclusion of the computation.

The following plots present the results of a simulation involving $N = 1000$ bodies and $t_{sim} = 100.0$, using different values of $r_{escapers}$: 0.5, 1.0, 1.5, and 10.0 times the value of the half mass system. Additionally, each simulation includes the total number of bodies that appeared to have escaped by the end of the computation.

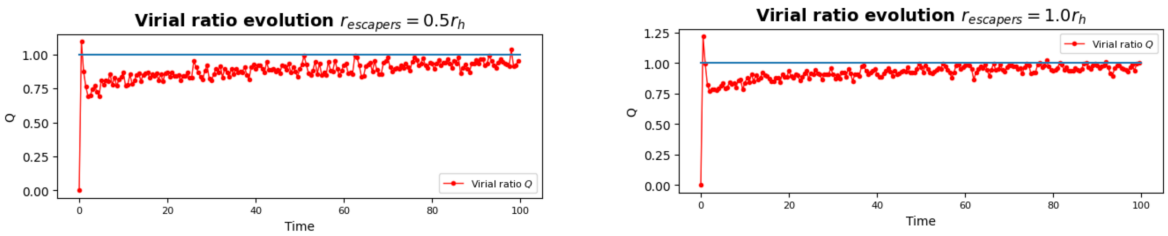


Figure 22: 1000-body, in the first case the number of escapers in $N = 533$, in the second one $N = 386$.

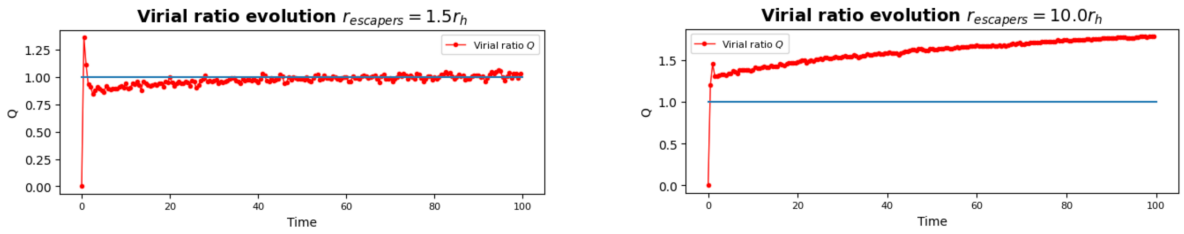


Figure 23: 1000-body, in the third is $N = 339$ and in the last one $N = 91$

It is evident that underestimating the number of escapers, as observed in the last simulation, results in an overestimate of the Q value, as anticipated. Conversely, opting for a highly "selective" $r_{escapers}$ value would lead to an underestimation of the virial parameter. Based on our simulation, we can confidently state that an appropriate value for the escaper distance criteria would be approximately $1.5 \cdot r_h$. We will use this value in all the following simulations.

3.6.3 100-Body, with equal masses

In order to understand how **100 bodies** behave (all **masses are equal**, $m_i = 1.0$, $\forall i = 1, ..., N$), we show in fig.24, some screen captures of these 100 bodies positions.

Initially, all the bodies are randomly positioned in a uniform sphere of radius $R = 10$, and the softening parameter turns out to be (from eq.16): $\epsilon = \frac{R}{2\sqrt[3]{N}} \simeq 1.08$. We chose $Q_{ini} = 0.8$, and studied the evolution over a time $t = 1000$ (the dimensionless numerical time step is $dt = 0.1$).

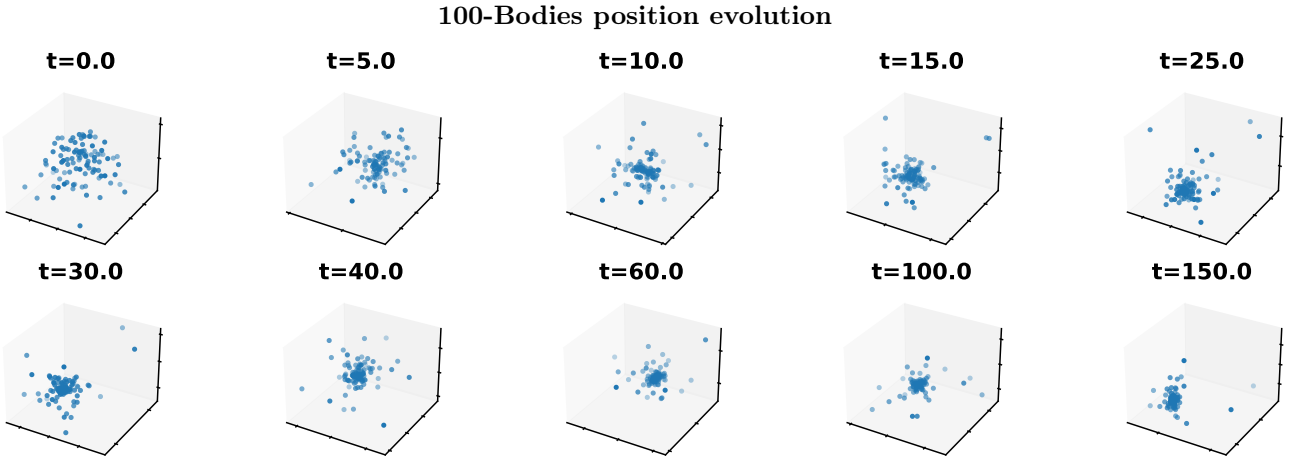


Figure 24: Screen captured 100-body position.

The total duration of our simulation corresponds to $t_{sim} \simeq 100\,000$ *years*, and the time step to $dt \simeq 30$ *years*.

The corresponding virial ratio trend is shown below, in fig.25:

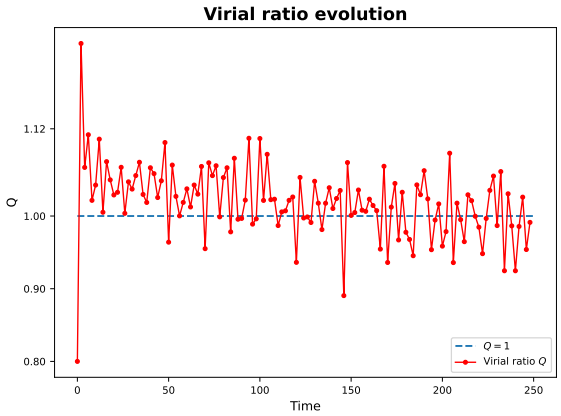


Figure 25: 100-body virial ratio evolution, with a $Q_{ini} = 0.8$.

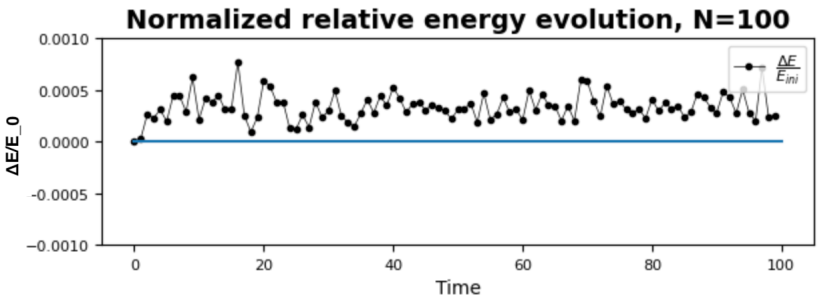


Figure 26: 100-body Energy evolution, with a $dt = 0.01$.

We chose $Q_{ini} = 0.8$ and, as we can observe from the red line indicating the evolution of Q, we see how its value tends to oscillate around a number which is slightly above 1. We can indeed notice (from fig.24) how the system undergoes a slight expansion.

It should be stressed that, as we can see in fig.24, the center of the higher density region could not coincide with the origin of our initial system. This could be given by a relevant initial anisotropy, due to the fact that the system is not a continuum. The value of the *half mass radius* r_{hm} is computed starting from the **center of mass** of the higher density region (which, instead of the center of mass of the total system, is not a conserved quantity), in order to have a more precise evaluation of the *escapers correction*.

In this case, the relative error of the energy is $3 \div 4$ orders of magnitude lower than the 1000-Body case. This can be explained using a lower time step $dt = 0.01$ (against the previous $dt = 0.1$), so there are less integration errors. Also the fact that since there are less bodies there are less *strong interactions* between them, and this leads to less *fly-by* errors.

The total number of escapers is $N_{esc} = 35$, the 35% of the bodies escape.

3.6.4 99-Body, with different masses

We now want to understand how **99 bodies**, with **different masses**, behave. Therefore, we divide them in 3 groups of 33 bodies. Each group has different mass values; in green we plot all the bodies with $m = 10.0$; in red we represent the bodies with $m = 5.0$ and in blue $m = 1.0$, the lightest ones.

Ten screen captures of these 99 bodies positions are shown in fig.27. Initially, all the bodies are randomly positioned in a uniform sphere of radius $R = 10$, and the softening parameter turns out to be: $\epsilon = \frac{R}{2\sqrt[3]{N}} \simeq 1.08$.

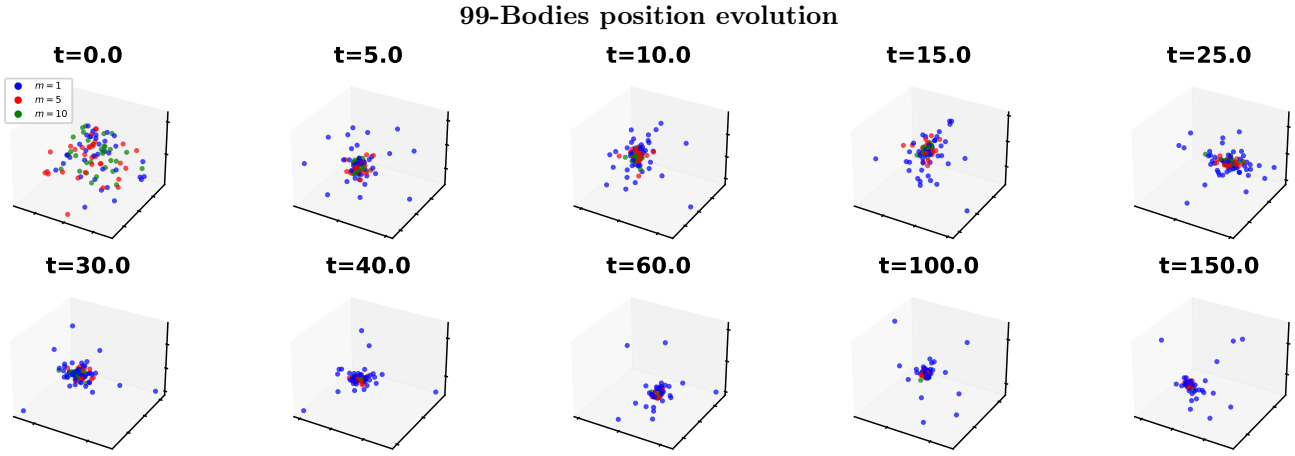


Figure 27: Screen captured 99-body position

As the system evolves, we notice how the heavier masses centralize (the green ones are all condensed in the center), while the lighter masses are arranged further outward; the blue ones, which are the lightest, are the furthest out. This phenomenon is called *segregation of masses* which is typical of clusters.

The corresponding virial ratio trend is shown in fig.28 down below:

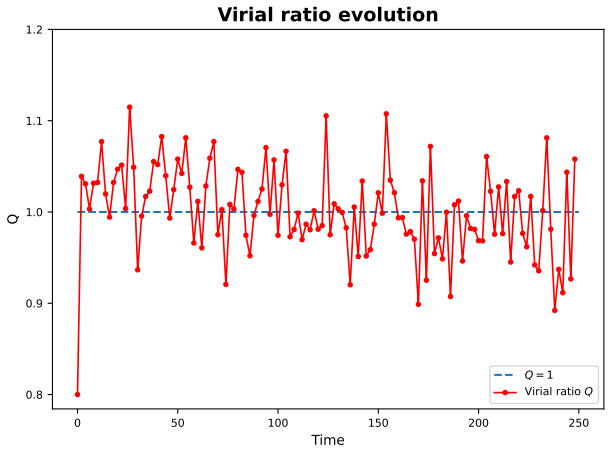


Figure 28: 99-body virial ratio evolution, with a $Q_{ini} = 0.8$.

We chose $Q_{ini} = 0.8$ and we see how its value tends to oscillate around a number which is around 1.

The total number of escapers turns out to be $N_{esc} = 30$, and we can see from fig.27 that the escapers components are dominated by the lighter masses.

3.6.5 Without Softening

A further analysis of the N-body code was conducted, specifically examining the scenario where the softening parameter ϵ is set to zero. For this investigation, a simulation with 100 bodies, a total integration time of $t_{sim} = 100.0$, and a time step of $dt = 0.01$ were chosen. The disparity in results is striking: when the softening parameter is not considered, the Q value exhibits significant growth from the early stages of the evolution, eventually reaching approximately 40 at the conclusion of the simulation.

This outcome serves as evidence supporting the importance of introducing a softening parameter to mitigate the effects of granularity, such as close encounters that would otherwise lead to unrealistic motion of particles and higher acceleration values.

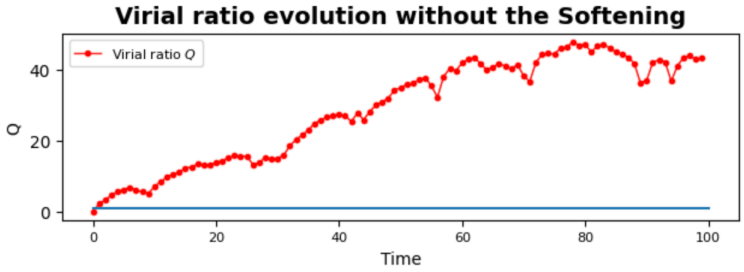


Figure 29: The increase in the Q value when imposing $\epsilon = 0$ (the blue line represents $Q = 1$).

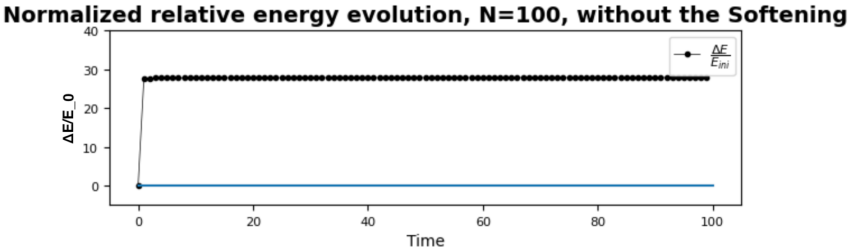


Figure 30: Relative $\frac{\Delta E}{E_0}$ value evolution when the softening parameter is null.

Under this hypothesis, we can also see that the relative energy error increases dramatically in the first stages of the evolution, when the bodies appear to be closer, and the effect of granularity is more significant. In the first instants, the fly-by effect increase the error on the energy, and after an "explosion" (the system starts with an initial phase of compression $Q_{ini} = 0.001$) the error remains constant because the bodies are basically not interacting anymore.

4 Numerical methods

4.1 Runge-Kutta 4 algorithm description

The *Runge-Kutta 4* (*RK4*) algorithm is a numerical method for solving ordinary differential equations (*ODEs*), commonly used in scientific computing applications. This algorithm involves four steps for each "time step" of the *ODE* solution. Given an initial value problem of the form $\dot{y}(t) = f\big(t, y(t)\big)$ and an initial condition $y(t_0) = y_0$, the *RK4* method computes the solution at the next time step $y(t_1) = y_0 + k$, where k is the estimate of the slope of the solution curve at t_0.

The *RK4* method computes k using four intermediate slopes (k_1, k_2, k_3, k_4) evaluated at different points within the time step. These intermediate slopes are calculated as follows:

$$\begin{cases} k_1 = f(t_0, y_0) \\ k_2 = f(t_0 + \frac{dt}{2}, y_0 + k_1 \cdot \frac{dt}{2}) \\ k_3 = f(t_0 + \frac{dt}{2}, y_0 + k_2 \cdot \frac{dt}{2}) \\ k_4 = f(t_0 + dt, y_0 + k_3 \cdot dt) \end{cases}$$

Where dt is the size of the time step.
The estimate of the slope k is then calculated as a weighted sum of these intermediate slopes:

$$k = \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

Finally, the solution at the next time step is computed as:

$$y_1 = y_0 + k \cdot dt$$

The *RK4* algorithm is a higher-order method that provides more accurate results than simpler methods (such as *Euler's method* or the *Runge-Kutta 2* algorithm). *RK4* is more accurate than the *RK2* because it uses four intermediate slopes instead of two. This allows it to capture more of the curvature of the solution curve and produce a more accurate estimate of the slope at each time step. Additionally, the *RK4* algorithm has a higher order of convergence, which means that it converges to the exact solution at a faster rate than the *RK2* algorithm. However, the *RK4* algorithm requires more computational effort than the *RK2* algorithm due to the additional intermediate steps.

4.2 Leapfrog algorithm description

In numerical analysis the *leapfrog* algorithm is an integration method used to solve ordinary differential equations (*ODEs*). The idea behind the *leapfrog* algorithm is to update the positions and velocities of the particles in a staggered way. Specifically, the positions are updated at integer time steps ($t = 0, 1, 2, ...$), while the velocities are updated at half-integer time steps ($t = \frac{1}{2}, \frac{3}{2}, \frac{5}{2}, ...$).

To implement the *leapfrog* algorithm, we start with the initial positions and velocities of the particles. Then, at each time step t, we do the following:

$$\begin{cases} v(t + \frac{1}{2}) = v(t - \frac{1}{2}) + a(t) \cdot dt \\ x(t + 1) = x(t) + v(t + \frac{1}{2}) \cdot dt \end{cases}$$

where $x(t)$ is the position of the particle at step t, $v(t + \frac{1}{2})$ is the velocity, or first derivative of x, at step $(t + \frac{1}{2})$, $a(t)$ is the acceleration and dt is the size of each time step.

The *leapfrog* algorithm is often preferred over other numerical integration methods because it is time-reversible and symplectic, meaning that it conserves the energy and momentum, making it a reliable method for simulating physical systems that involve the conservation of these properties.

4.3 Possible more advanced numerical methods

With the two numerical methods mentioned before, the computational effort turns out to be $\mathcal{O}(N^2)$. In fact, as we can see from eq.2, in order to compute force over the i-body, we first have to obtain all the forces between i and j and sum them, each time $i \neq j$.
For example, if we take into account 9 particles (fig.31), there will be needed $\boldsymbol{N \cdot (N-1)} = 9 \cdot 8 = 72$ **calculations**. As we can see, the complexity grows as $\boldsymbol{N^2}$.

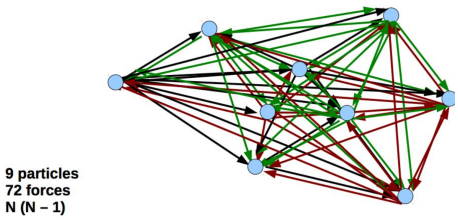


Figure 31: *Direct summation*, example for $N = 9$ particles.

We are now going to briefly present some alternative numerical methods that require a less "amount" of computational effort.

4.3.1 Tree Codes

A *Tree Code* is a method which can be used to solve the N-body problem. It uses a hierarchical tree structure to efficiently calculate interactions between particles. It approximates distant interactions and reduces computational complexity (computational effort $\simeq \mathcal{O}(N \log N)$).

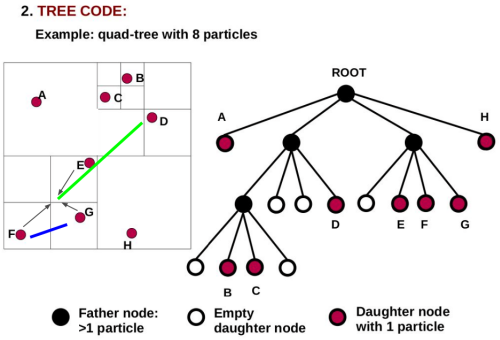


Figure 32: *Tree Code*, example for $N = 8$ particles. As can be seen, it divides the space into cubic cells until they are occupied by a single particle. Then, it organizes particles into tree-like structures.

4.3.2 Particle Mesh

A *Particle Mesh* is another method that can be used to solve the N-body problem.
The particles' positions are discretized and mapped onto a grid (*mesh*). The potential equations are solved on the *grid points* ($G < N$). Then, the forces that these potentials exert on other particles are evaluated.

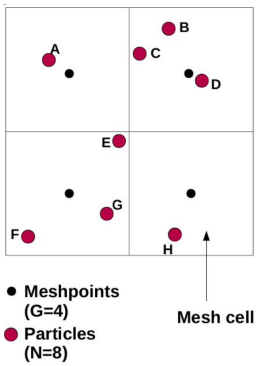


Figure 33: *Particle Mesh*, example for $N = 8$ particles, in $G = 4$ *meshpoints* (or *grid points*).

Using a regular grid allows the problem to be solved in the Fourier space (*FFT*-Fast Fourier transform). The computational cost turns out to be $\mathcal{O}(G \log G)$. A limitation that may occur is the grid spacing.

Table 3: **Comparison between different types of algorithms**

Type	Computational effort	Information
Direct summation	$\mathcal{O}(N^2)$	Practical for $N < 10^4$
Particle Mesh	$\mathcal{O}(N \log N)$	*FFT*s to invert Poisson equation
Tree Codes	$\mathcal{O}(G \log G)$	Multipole expansion