N-body Problems in our Solar System: Various Numerical Computing Approaches

PHYS 512 Final Project Report

Babic, Sara (260987990)
Camara, Luca (261051275)
Zhang, Shu (260944682)

December 5, 2023

Abstract

Based on the object-orientated style of coding, we studied and simulated the unique motions of solar objects: Mercury precession and grouping of Trojan Asteroid belt. We implemented Barnes-hut algorithm to handle a large number of objects in the system. We successfully mimic the scenario that is consistent with real observed behaviours.

1 Introduction

The N-body problem has been a topic of interest amongst mathematicians and physicists alike since its conception in the late 17th century, whether it be for its fascinating chaotic nonlinear dynamics or for its applicability to orbital mechanics and cosmological evolution. In this project, we investigate the orbital dynamics of the inner solar system, specifically, the feasibility and accuracy of simulating known phenomena such as the main asteroid belt, the Jupiter Trojans, and the precession of Mercury's orbit. We also analyze the computational methods required for each and their differences. Many of the issues encountered in N-body simulations are scenario-specific, requiring the implementation of different algorithms to suit the required purposes. We explore the uses of particle-particle, both Newtonian and with relativistic corrections, as well as the famous Barnes-Hut algorithm, which some may classify as a particle-mesh method.

1.1 Newtonian Gravitation

Newton's second law states that the sum of the forces acting on an object are equal to the mass of the object times the acceleration of the object. Vector calculus also tells us that we can describe conservative vector fields as the gradient of some potential function. Using these facts, and the well defined gravitational potential energy $U_i = -\frac{GMm_i}{r_i}$, we can form an equation to obtain the acceleration $\vec{a}_i$ of an object of mass m_i due to a certain massive object M at a distance $\vec{r}_i = r_i\hat{r}_i$ as:

$$\vec{F}_i = -\nabla U_i = m_i\vec{a}_i \implies \vec{a}_i = \frac{GM_i}{|r_i|^2}\hat{r}_i\,.$$

Since each celestial object interacts with every other object in the given space, the total force will be a sum of each contribution, where we introduce a new parameter $\mu_i = GM_i$ that we call the associated gravitational constant for simplicity:

$$\vec{a} = \sum_i^n \vec{a}_i = \sum_i^n \frac{\mu_i}{|r_i|^2}\hat{r}_i\,.$$

1.2 Lagrange Points, the Jupiter Trojans, and the Hilda asteroids

Trojans are small celestial bodies that share their orbit with another more massive body trailing 60° degrees ahead or behind of it. The Jupiter Trojans are a group of an estimated tens of thousands to millions of asteroids that fit this criteria for Jupiter. The Hildas are another group of asteroids in Jupiters orbit that form a characteristic triangular 'dance' around the inner planets with the corners of the triangle at 60°, -60°, and 180° from the position of Jupiter, due to a 3:2 mean motion resonance, that is, for every 3 orbits of the Hildas, there is two orbits of Jupiter, leading to this apparent triangular motion. The angular positions of the two groups of asteroids are no coincidence, the positions at $+60^\circ$, -60°, $+180^\circ$ are commonly referred to as the L_4, L_5, and L_3 Lagrange points respectively [1]. These are points in the gravitational field where small objects can get trapped in a gravitational potential well, as these points define positions of balanced gravitational forces from a system of two much more massive gravitating bodies, i.e the Sun and Jupiter.

1.3 Relativistic Gravitation

For situations involving very strong gravitational fields, Newtonian gravity fails to describe the motion of bodies accurately. In these cases we must not neglect relativistic interactions. Although the sun is not a strongly gravitating object due to its relatively low mass on a cosmological scale, Mercury's orbit is so close to it that the space it travels through at perihelion is slightly warped. This yields a relativistic boost, if you will, which causes one portion of the precession of Mercury's orbit, which is inexplicable without this relativistic correction [2].

1.3.1 Gravitational time dilation

The gravitational redshift is given by: $\frac{\Delta\nu}{\nu_0} = -\frac{\nu}{c^2} = -\frac{g\Delta x}{c^2}$. If we sub g using Newtonian gravity $g = GM/r^2$, and spacial distance with dr for a spherical object of mass M. Then integrating it over all the frequency

(ν_0 to ν_{inf}) and the entire space (r_0 to r_{inf}). Finally with the assumption that $r_0/r_c = GM/r_0c^2 \ll 1$, and approximation with Taylor expansion, we finally obtain:

$$\frac{\nu_{\text{inf}}}{\nu_0} \simeq e^{-GM/2r_0c^2} \simeq (1 - \frac{GM}{r_0c^2})^{1/2}.$$

Thus, gravitational time dilation is:

$$\frac{\Delta t_0}{\Delta t_{\text{inf}}} = \frac{\nu_{\text{inf}}}{\nu_0} = (1 - \frac{GM}{r_0c^2})^{1/2}.$$

1.3.2 The Schwarzschild Metric

In a 4-coordinate, spacetime coordinate, metric is not defined to be the same as 3D coordinates: $ds^2 = dx^2 + dy^2 + dx^2$. Instead, 3D coordination suggests the proper length Δl, where the metric for flat spacetime is then: $ds^2 = (cdt)^2 - dl^2 = (cdt)^2 - (dx^2 + dy^2 + dx^2)$. In spherical coordinates, it reads: $ds^2 = (cdt)^2 - dl^2 = (cdt)^2 - dr^2 - (rd\theta)^2 - (rsin\theta d\phi)^2$. It to encounter the gravitational time dilation fraction which is derived previously (see section 1.3.1) for $dt = d\tau$, the Schwarzschild Metric written in Schwarzschild radius $r_s = 2GM/c^2$ is given by :

$$ds^2 = (cdt\sqrt{1 - \frac{r_s}{r}})^2 - (\frac{dr}{\sqrt{1 - \frac{r_s}{r}}})^2 - r^2(d\theta^2 + sin^2\theta d\phi^2).$$

1.3.3 Equation of Motion*

Given the Schwarzschild metric derived previously, combined with the Euler-Lagrange equation:

$$\frac{\partial \mathcal{L}}{\partial x^i} = \frac{d}{d\tau}\frac{\partial \mathcal{L}}{\partial \dot{x^i}},$$

using conservation of angular momentum, and substituting Lagrangian with energy term: $\mathcal{L} = \epsilon$, one will reach the equation:

$$(\frac{dU}{d\phi})^2 + U^2(1 - \frac{2GM}{c^2}U) - \frac{2GM\epsilon^2}{h^2}U = \frac{c^2}{h^2}(k^2 - \epsilon^2)$$

where $U = 1/r$, $r^2\dot{\phi} = h$. Therefore equation of motion in differential form could be obtained:

$$\frac{d^2U}{d\phi^2} + U = \frac{GM\epsilon^2}{h^2} + \frac{3GM}{c^2}U^2$$

which could be further rewritten [4]:

$$\ddot{r} = -\frac{GM}{r^2} + \frac{2GM}{c^2r^3}. \quad (1)$$

2 Numerical Methods

The numerical solutions describing a dynamical system of N bodies/particles interacting gravitationally are powerful tools for the study of astronomical systems. The challenge in *N-body simulations* lies in updating this system where each body's state affects every other in a computationally efficient manner, as the system is scaled up.

A direct integration approach, also known as the naive, brute-force, or particle-particle method, is computationally expensive. At each time step, each pairwise interaction is computed without introducing approximations in the equations of motion, resulting in an algorithm that runs in quadratic time $\mathcal{O}(N^2)$. We explore an alternative method, the Barnes-Hut algorithm.

The system of N bodies interact under Newtonian gravity. In some cases, Post-Newtonian corrections involving a correction factor that approximates the effect of general relativity (GR) are necessary.

2.1 Correction factors

According to Einstein's equation of motion (see Equation 1), the acceleration (or the force) acting on a certain object in the rest frame is fixed by a factor:

$$g = g\prime(1 - \frac{2}{rc^2})$$

The precession of Mercury which is caused by Sun's gravity is much larger comparing to the effect from other solar objects. For example, Jupiter is massive, it also has a significant impact on Mercury's orbit, but still, merely 1/20 of what the Sun can do. In order to save computation time, we choose to only include Mercury and Sun in the sandbox. And we uses particle-particle interaction to increase the precision of the results.

*Derivation inspired by [3].

Unlike the following section of Barnes-Hut algorithm, which aims for efficiency, this section aims for high precision in evaluating the position of Mercury. Thus we use Euler's method and Runge Kutta method to update the acceleration, in the cost of computation power.

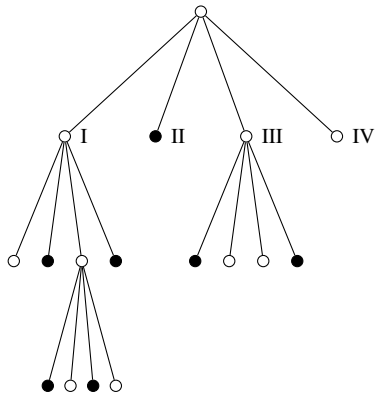
2.2 Barnes-Hut Algorithm

When it comes to N-body simulations, it is a well known fact that effective runtime management becomes a formidable challenge as the number of bodies involved escalates. Some attempts to mitigate this may involve neglecting interactions from relatively far away or relatively insignificant masses within the system. However, it is important to recognize that in dynamic and especially chaotic systems, even seemingly minor perturbations or their absence can exert profound influences on the system's dynamics over cosmological timescales. In 1986, J. Barnes and P. Hut developed their namesake algorithm that ingeniously strikes a delicate balance by incorporating minute interactions while maintaining computational efficiency [5].

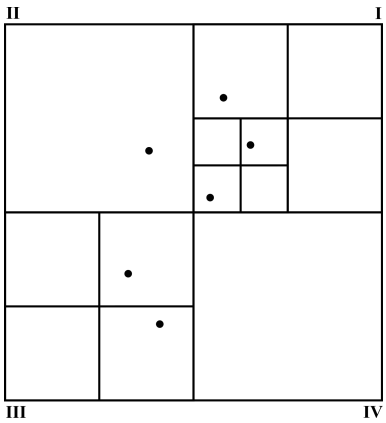
The Barnes-Hut algorithm leverages a hierarchical tree structure, specifically a Quadtree in two dimensions or an Octree in three dimensions, to recursively partition the simulation domain while storing data about the objects in each partition at each level of the tree. In this Quadtree, each external leaf node signifies a quadrant containing at most one object, whereas internal branch nodes represent more populated quadrants encapsulating cumulative information about the nodes/quadrants beneath/within them, including total mass and center of mass. The force calculation is then simply a matter of iterating through each object, but as opposed to computing the contributions from each other individual object, we traverse the tree depth-first and at each internal node, a crucial ratio between the size of the quadrant D and the distance d from the object to the quadrants center of mass is computed. Should this ratio fall below a specified threshold parameter, denoted by criterion-θ (typically 0.5), the force calculation is carried out using the center of mass and total mass of the current quadrant, which approximates the force contribution from that quadrant as that from a single body. In other words, this adjustable criterion determines which bodies are sufficiently far-way ($D/d < \theta$).

This strategic approximation significantly diminishes the computational burden from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log_4 N)$, since for each body we need not interact it with every other body but rather a subset of bodies proportional to the θ parameter and the depth of the tree. The depth of the tree is approximated to be $\log_4 N$ since for a full tree the number of leaves and thus the number of bodies would be $4^{\text{depth}} = 4^{\log_4 N} = N$. Another benefit to this is that the force computation has the same asymptotic complexity as building the Quadtree itself (same logic as force calculation minus θ factor) making it essentially a free operation in terms of asymptotic complexity. Thus many implementations of this algorithm, including ours, opt to rebuild the Quadtree at each timestep instead of worrying about things like tree rotations.

This algorithm is intuitively understood graphically as shown in Figure 1.



(a) Quad tree structure where each solid node represents a body.



(b) Quadrant space where each point represents a body.

Figure 1: Schematic showing Barnes-Hut simulation hierarchical structures.

2.3 Code Description

We use an object-oriented programming approach for its modularity, flexibility, and readability. The code is based on two main class objects:

- `Body`
  This parent class describes each individual body/particle in our simulation. It stores their mass, their position, their velocity and even their trajectory (i.e. position history throughout integration time). A `Planet(Body)` class inherits these attributes while adding new information (name, object radius, GM, orbit semi-major axis, orbit eccentricity) useful for plotting and simulating known/identified celestial objects.

- `SolarSystem`
  This parent class describes the system as a whole. Its methods involve evolving the system according to a specified integrator (Euler, RK4, Leapfrog) that loops calls to the `Body` methods. It has dedicated subclasses for each simulation method: `SolarSystemPP` for particle-particle, `SolarSystemPMBH` for particle mesh Barnes-Hut.

  To implement the Barnes-Hut algorithm, we built a supporting `Node` class. To build the Quadtree, it uses an insertion method to recursively insert bodies as new nodes and a modified depth-first search for its acceleration calculation method.

For reusability and clarity of code, these classes are all equipped with the standard getters and setters to access attributes as well as initializers to reset environments.

2.4 Initial Conditions

When simulating our solar system, we initialize all celestial objects as `Body` (or `Planet`) objects.

We start with adding familiar celestial bodies : the Sun, Mercury, Venus, Earth, Mars, and Jupiter. Their exact masses, and position and velocity vectors are set using NASA's Horizons System [6] using 01/01/2000 00:00 as epoch and the Solar System Barycenter as the origin.

The asteroid belts were simulated using two different strategies. The first involved simulating only the Jupiter Trojans as well as the main asteroid belt. The Trojans are initialized using normal distributions in radius and angle on limited domains which are chosen to be a radial differential of 1 AU centered on 5.2 AU, and an angular differential of 30° centered on $\pm 60^\circ$ relative to Jupiter's position in its orbit. Few Hildas were also added similarly at 180°, although their count is limited to keep the object ratio between the asteroid bundles realistic. For the main asteroid belt we use a similar method, with a radial differential of 1.2AU centered on 2.6 AU which is a distribution that obeys the Kirkwood gap; a gap in semi-major axes for main belt asteroids. The velocities of these bodies were difficult to come across, thus a simple approximation of the tangential velocity using Newtons second law applied to the radial forces on the asteroids neglecting all forces besides those from the Sun:

$$F_r^g = F^{cent} \implies \frac{GMm}{r^2} = m\frac{v^2}{r} \implies v = \sqrt{\frac{GM}{r}}\,.$$

This velocity is then broken into components depending on the position of the mass in question. Lastly, we justify using the higher end of the mass spectrum (10^{18} kg) for the asteroids masses as those will be the ones to influence each others dynamics, and our number of objects is still limited by computation time.

Subsequent simulations omitted the main belt asteroids, and thus doubled the starting number of Trojans. Slight modifications to the parameters of the normal distributions used were made to accompany this change.

3 Results and Analysis

3.1 Code Test

The vast majority of code testing was done via visual verification and intuition of known physical laws. Both simulation methods, particle-particle and Barnes-Hut, were to verified by observing the simple case of the orbits of the planets which was first verified before introducing asteroids. This test is run for a full Jupiter period to ensure it completes its orbit. In Figure 2, the test result is shown for one of the methods. Both methods produced visually indistinguishable results. More careful error analysis and comparison is discussed in later sections.

For the relativistic correction, the characteristic precession of Mercury was looked for to ensure the appropriate relativistic kick was added at perihelion. For the Barnes-Hut method, the function `displaySystem()`, which was debugged using chatGPT3.5, was used to verify that the quad tree creation was working as intended. As seen in Figure 3, each body is indeed in its own quadrant.

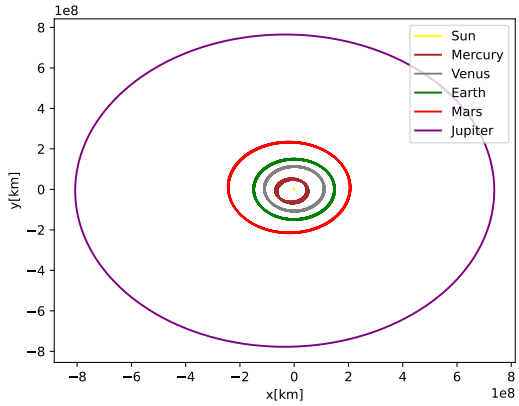


Figure 2: Orbital trajectories of planets in particle-particle simulation integrated over 12 years (i.e. Jupiter's period).

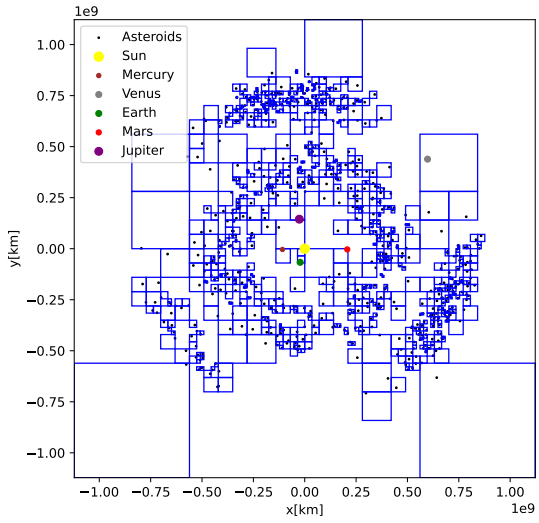


Figure 3: Barnes-Hut algorithm grid segmentation.

Additionally, verification of characteristic centers of mass such as the Solar System Barycenter were confirmed to be consistent with expected values. Lastly, things like using a debugger to verify that the logic was performing as it was intended as well as the insertion of raise and print statements at problematic checkpoints in the code to catch bugs that could not be anticipated.

3.2 Mercury Precession

From the observation data, the precession of Mercury's orbit is known to have a rate of 574 arcseconds per 100 years [7]. Our simulation is carried out with the condition: that we put only Mercury and Sun in the sandbox to study the shift of Mercury's orbital axis. It is tough to visualize such a small shift($\sim 0.003°$) in the scale of Mercury's orbit (see Figure 4). Both Euler's and RK4 methods converge to the expected value, shown in Figure 5, with RK4 converging at a much faster rate than Euler's.

Although the results demonstrate an agreement between our simulation and theory/observation, however, it is not strong. Upon converging to the expected value, the results still show large fluctuations (recall that Figure 5 is in log-log scale). We suspect that the dominant error source is the limitation of numerical simulations. The effect of sun gravity is not that significant such that GR factor of acceleration is very small. In this case, the round-off error will have a large impact on our simulation.

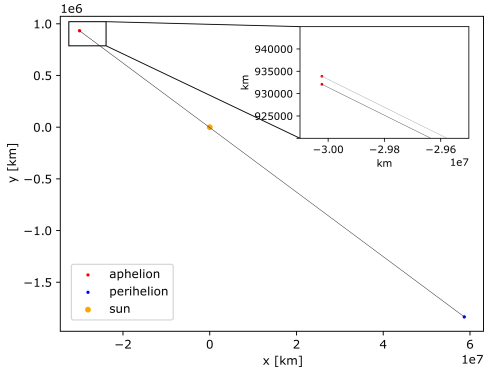


Figure 4: Visualization of the Mercury precession in the scale of its orbit. This simulation is taken with a total orbital time of 2 Julian years with $10^{6.5}$ steps.

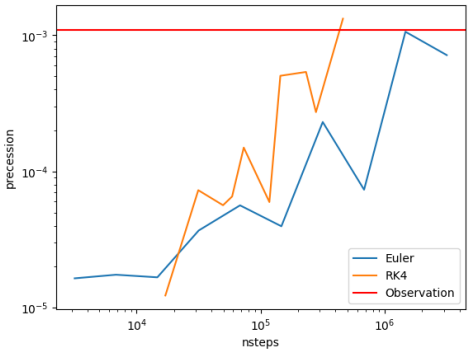


Figure 5: Precession with different numbers of timesteps(all over 1 Julian year). The figure shows Euler method and RK4 converge to the expected value$\sim 0.00119°$.

3.3 Motion of the Trojan Asteroids

The motion of the Trojan asteroids was simulated in many different cases as can be seen in Figure 7. Some overall observations that were seen across all simulations is the remarkable stability of the L_4 and L_5 Lagrange points, and to an extent L_3 as well, as all simulations showed a clear population of those areas after their respective integrations.

A preliminary (not recorded) twelve year (one Jupiter orbit) integration test involved densely and uniformly populated regions of asteroids set up $60°$ ahead and behind Jupiter in its orbital path, spanning only a few degrees each. The result of these tests were asteroids spanning a much larger angles of the orbit than they did initially, however still keeping in line with Jupiter's orbit.

A following simulation (Figure 7 subplot (**b**)) was then run with altered initial conditions, setting the initial span of each group of asteroids to be wider than the last (approx. $30°$) and also widening the radial distribution to approximately 1 AU across Jupiter's orbit. We also now switch to normal distributions from uniform ones in the hopes of seeing a smaller spreading of the asteroid clusters over the integration. The result of this second test mimicked the first, with the asteroids thinning out radially and spreading out angularly, however the relative normal distribution appears to be preserved. Next, the main asteroid belt was added with the idea that since it carries as much mass or more as the Jupiter Trojans, that it may actually have an influence on them, particularly in spreading them out radially to almost connect with the main belt. Results of this simulation (Figure 7 subplot (**a**)) did not yield answers to this question but instead were a great demonstration of just how stable the Jupiter Trojans are, as main belt objects were chaotically ejected from the system or had absurdly eccentric orbits when given equivalent initial conditions, the final configuration of the Trojan asteroids did not change from the previous test in this case. As a final testament of the stability of this system, a 400 year integration was performed with refined initial conditions, that confirmed again the stability of these orbits as a clustering around L_3, L_4, L_5 is clearly maintained (Figure 7 subplot (**c**)). An analysis of the paths of these asteroids with time reveals that there are huge variations in the eccentricities of some of these orbits, but unlike the main belt objects, none were fully ejected from the solar system in this case.

3.4 Simulation Time Complexity

We compare the time complexities of both our simulation methods: Barnes-Hut and particle-particle. The results shown in Figure 6 show that the asymptotic behaviours of Barnes-Hut and particle-particle agree with the expected complexities of $\mathcal{O}(N \log N)$ and (N^2) respectively. For a system with a large number of bodies, the Barnes-Hut simulation is more efficient. However, for a system with very few bodies the particle-particle simulation is faster: the cost of building the Quadtree data structure in the Barnes-Hut algorithm is not insignificant at small N.

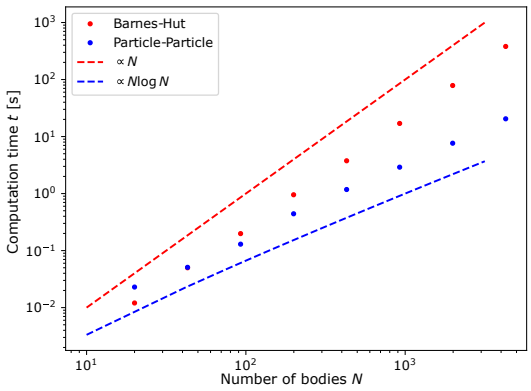


Figure 6: Comparison of computation time as a function of the number of bodies of Barnes-Hut and particle-particle simulations. These simulations were for one (1-day) timestep with the same initial conditions, only changing the number of bodies N for each simulation method. For the Barnes-Hut simulation we set criterion-$\theta = 0.5$.

Figure 8 shows how the criterion-θ affects the force estimation of the Barnes-Hut algorithm. Higher values of θ lead to coarser approximations: more bodies (leaf nodes in the Quadtree) are approximates by the center of mass and total mass of grouped bodies (branch nodes in the Quadtree). Inversely, lower values of θ result in more accurate calculations by considering smaller groups of bodies. However, this increases the computational cost as more nodes are analyzed and fewer approximations are made.

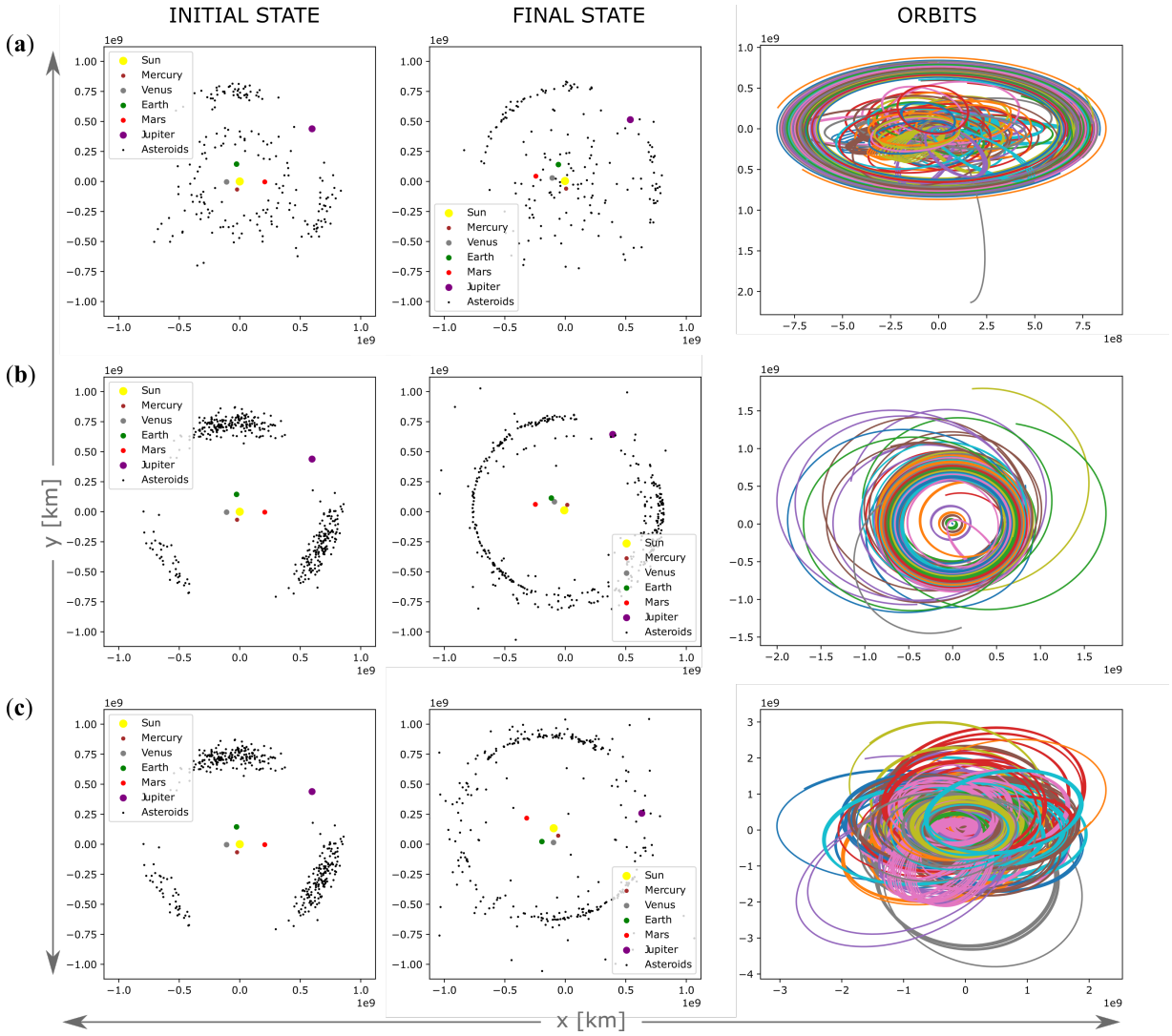


Figure 7: Summary of N-body simulations using the Barnes-Hut algorithm. Each row represents a simulation run with different defined parameters. For left to right, the columns represent the initial state of all bodies in the system, their final state and their integrated trajectories[†]. For all runs, we set criterion-$\theta = 0.5$, and use a Leapfrog integrator.
(**a**) The asteroids are composed of $N = 400$ equal-mass bodies, distributed such as to imitate the main and trojan asteroid belts. The integration time is of 36 years.
(**b**) The asteroids are composed of $N = 400$ equal-mass bodies, distributed such as to imitate the trojan asteroid belt.
(**c**) The asteroids initial state is the same as in (b). The integration time is of 400 years.

[†]We do no include the trajectories of objects whose orbits were highly eccentric or escaped the solar system, we attribute these to initial condition errors.

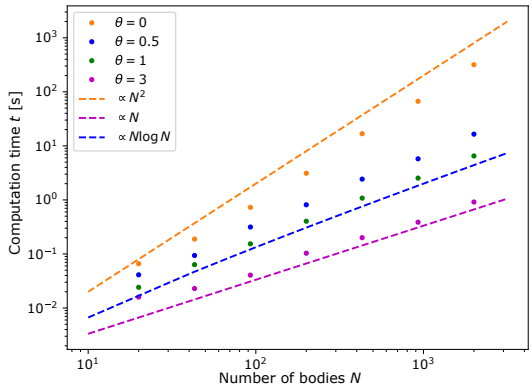


Figure 8: Comparison of computation time as a function of the number of bodies of Barnes-Hut and particle-particle simulations. These simulations were for one (1-day) timestep with the same initial conditions, only changing the number of bodies N for each simulation method. For the Barnes-Hut simulation we set criterion-$\theta = 0.5$.

3.5 Simulation Accuracy

Particle-particle methods introduce no approximations, providing accurate results for all interactions, limited only by roundoff error as timesteps go to zero, whereas Barnes-Hut introduces approximations, especially for distant particle interactions, which may reduce precision. An evaluation of the mean positional error relative to the particle-particle method versus the number of bodies was performed which yielded results that could not be explained nor interpreted. We do believe the error calculation to be correct however as θ's close together overlap, and the one large θ introduces more error than the smaller ones as it should.

Also, setting $\theta = 0$ in Barnes-Hut should be equivalent to a particle-particle simulation. Though $\theta = 0$ relative error is expected to be near machine error, this is not what we observed as illustrated by Figure 9. We suspect this is due to a mistake in our relative error calculation. Unfortunately, we do not have the time necessary to verify this and re-run this computationally expensive simulation.

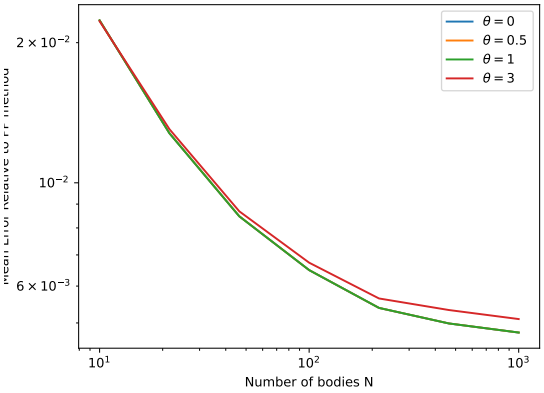


Figure 9: Relative error analysis of Barnes-Hut compared to particle-particle simualtion as a function of the number of bodies N, with various criterion-$\theta = \{0, 0.5, 1, 3\}$. This error is the average error per body.

4 Discussion

4.1 Errors and Flaws

The code we wrote is so computationally heavy that in many cases where we want higher precision, we will face large computing time that is in the scale of days. Especially in section 3.2, shown in 5, we don't have sufficiently enough data to prove that the results of our simulation will be stable after reaching observation data. We could only prove that the data is converging to the expected data. Beyond this point, the computation time is too long for us to execute the code. This flaw could potentially make some errors in our code blind to us. Thus we urge optimization for the coding, which will be mentioned in Section 4.3.

4.2 Potential Improvements

Some future pursuits for this project would be to potentially vectorize some of the operations performed, although the main algorithm does not appear to be vectorizable, and to re-investigate the use of particle-mesh methods as our current implementation remained unfinished due to the difficulty in implementing an adaptive mesh method since a uniform mesh was much too coarse to properly resolve a domain with such large areas of empty space. Additionally, carrying out simulations over cosmological timescales, and perhaps trying to recreate the capture of the Trojan asteroids, would have been an interesting experiment to do as the longest timescale

simulated was only 400 Julian years, which is not long enough to view this phenomena unfold. It would have also been interesting and informative to investigate how the change in runtime and error with number of bodies changes as the configuration of said bodies changes, for example, in our case, we only investigate a configuration that has two areas of high body density (the Jupiter Trojans), which we theorize could have an effect on the computation time and precision.

For particle-particle interaction, there are numerous methods that updates the acceleration of celestial bodies in very different ways. In a much more elevated situation, we could try to implement and manipulate those methods to our functions.

4.3 Optimization

Astronomical environments are often very complicated and largely scaled. Numerically simulating these, or even only one particularly interested, scenarios will involve large-scale numbers. Especially when the information is spread in the speed of light, but evolutions of celestial bodies usually happen on a scale of decades or more, the simulation becomes a race of computation power. This is the exact situation we have encountered in our simulations. We have to take an insane amount of steps to reach the desired precision that matches the behaviours of real observation values. On top of that, the errors (round-off and others) always accumulate. This means we need both a large number of steps and larger bits to store our values. No matter which way to pursue, there is an increasing need of computational power.

Therefore, optimizing the functions will be a good option. Parallel calculations are also a favourable solution to the dilemma. Both of which is beyond our capability, thus to be a further potential improvement.

5 Conclusion

We have successfully implemented two approaches to N-body simulations: the particle-particle method and the Barnes-Hut algorithm, for two different inner solar system phenomena. For the Trojan asteroids, a replication of real phenomena was observed using the Barnes-Hut algorithm as the asteroids remained relatively close to their analytic distributions, and stayed stable in their orbits for as long as a 400 year integration. It was also determined that the main belt asteroids behaviour is much less stable than the Trojans. Analysis of the Barnes-Hut algorithm was also successful as previously documented complexities for various configurations and their limiting cases were correctly identified and verified.

We found that Barnes-Hut was more efficient (with $\mathcal{O}(N \log N)$) than particle-particle (with $\mathcal{O}(N^2)$). However, there is of course a trade-off on accuracy as Barnes-Hut introduces approximations (tuned with criterion-θ) for far-away objects.

We are able to simulate the perihelion shift of Mercury with the extra factor due to relativity introduced to Newton's equations. See the exaggerated effect in file "test.mp4", and we apply two different methods to calculate the particle-particle interaction: Euler's and Runge-Kutta, and successfully shows that RK4 converges faster to expected value than Euler's.

Contributions

We have contributed to writing the code and the report. A more detailed breakdown follows.

- Sara Babic:
  Classes Body, Planet, SolarSystemBH, Node; Sections 2, 2.2, 2.3, 2.4, 3.1, 3.4, 3.5; Figs 1, 4, 9
- Luca Camarra: Classes Body, SolarSystemPP, SolarSystemPM, SolarSystemBH, Node; Sections 1, 1.1, 1.2, 2.2, 2.4, 3.1, 3.3, 3.4, 3.5, 4.2, 5; Figs 2, 3, 6, 7, 8, 9
- Shu Zhang: Code: Mercury precession, SolarsystemPP, Planet; Section: 1.1, 1.2, 1.3, 2.1, 2.3, 2.4, 3.1, 3.2, 4.1, 4.2, 4.3; Figure: 4, 5

References

[1]P. Robutel and J. Souchay, "An introduction to the dynamics of trojan asteroids", in *Dynamics of small solar system bodies and exoplanets*, edited by J. J. Souchay and R. Dvorak (Springer Berlin Heidelberg, Berlin, Heidelberg, 2010), pp. 195–227.

[2]A. Einstein, "The foundation of the general theory of relativity", (1916).

[3]S. Heefer, *The Schwarzschild Solution, lecture notes in General Relativity*, June 2023.

[4]A. Sfarti, "Application of the euler–lagrange method in determination of the coordinate acceleration", European Journal of Physics **37**, 032001 (2016).

[5]J. Barnes and P. Hut, "A hierarchical O(N log N) force-calculation algorithm", Nature **324**, 446–449 (1986).

[6]NASA Horizons System, *Ephemeris*, data retrieved from World Development Indicators, `https://ssd.jpl.nasa.gov/horizons/`, 2023.

[7]N. I. Amelkin, "The perihelion shift of mercury's orbit", Mechanics of Solids **54**, 1131–1137 (2019).