A Keplerian method for Collision Detection in Protoplanetary Disk Simulations

Calculating the computational efficiency and the consecutive collision times in N-body systems as a function of Keplerian parameters.

by

Martijn Moorlag

to obtain the degree of Bachelor of Science
at the Delft University of Technology
to be defended publicly on Monday September 7, 2022 at 13:30.

Student number: 4866363
Bachelor End Project
Applied Physics & Applied Mathematics

Delft, 31 August 2022
TU Delft
Supervisors: Dr. Paul Visser
Prof. Dr. Jos Thijssen

Table of contents

Abstract

This report has been made to gain more insight and knowledge of the Keplerian method for collision detection simulations by simplifying N-body systems to a list or particles with orbital parameters. Collision detection is an essential part in modeling the evolution of protoplanetary disk becoming planetary systems by the merging of planetesimal objects. Presumably Keplerian systems allow for a computationally efficient algorithm, having its computational time influenced by orbital parameters, unlike the efficient octree method which only scales with the number of particles.

The aim of this research was to create a collision detection model for a simplified protoplanetary disk system using Kepler systems and analyse the results of the model to check the reliability of this simulation method, i.e. if the quality of the results are trustworthy, performing consistently well, and are accurate. In addition, the method was tested on its computational efficiency by investigating the run time of separate parts of the algorithm.

In the initialisation of the algorithm a particle list is created and sorted by increasing apoapsis. Then by a sweep and prune filter the list is checked for possible collision pairs. For pairs that have a small enough minimal orbital intersection distance that they can collide, the collision time is calculated and added to a time-ranked collision list. The algorithm keeps merging the pairs on top op the list, creating a new particle with new possible collision pairs and new collision times, and updates the time-ranked collision list after each merge. An empty collision list implies the end of the simulation. Various parameters as well as length of the lists are saved within the algorithm for subsequent examination.

The results of the Keplerian model found showed similarities to other simulation studies, and include the perceived critical eccentricity of 0.02, preservation of shape of body radius distribution, and the formation of large bodies. Simulations showed three stages of collisions, characterised by collisions occurring homogeneously throughout the disk, large body collisions, a final stage of small body collisions. The performance of the algorithm could be summarised by the number of checks by the sweep and prune filter *#checks* $\propto N^2$ and *#checks* $\propto \epsilon$ for eccentricity $\epsilon < 0.1$, the number of pairs in the collision pair list was found to be proportional to the number of starting particles N, the body radius s, the inverse of the inclination $1/I$, and the eccentricity ϵ, so *#pairs* $\propto N^2$, s, $1/I$, ϵ. For $N < 1000$ the computation time of the algorithm appeared to be proportional to N^2, but for $N < 10.000$ proportional to to a higher power. For $\epsilon < 10^{-9}$ and $N < 10.000$ the computation time was found to be $runtime \propto s$.

The conclusion of this research is that the Keplerian model could be used to model N-body systems for collision detection because of its behavioural resemblance to observable astrophysical systems. The computation time of the Keplerian system scales with a higher order per N than for instance the three code algorithm, but has showed to be influenced by orbital parameters in which this method differentiates itself from other simulation methods, with a remarkable influence on the total computation time by the body radius s of the system. More research could be done for higher orders of N to better model reality as well as providing more insight on the proportionality of the computation time. Recommendations to the algorithm consist of adding planetary spin as a particle parameter, and methods to incorporate apsidal precession in a competent manner.

List of Symbols

Symbol explanation:

a	semi-major axis
b	semi-minor axis
c	semi-focal separation
E	eccentric anomaly
G	gravitational constant
I	inclination
k, l	rounds to collisioin (integer)
m	mass
M	central mass
N	number of particles
s	body radius
T	orbital period
t^0	creation time
Ω	ascending node
ϵ	eccentricity
$\mathscr{R}$	rotational matrix
ω	angular speed
ϖ	argument of periapsis
$\vec{K}$	nodal line vector
$\vec{L}$	angular momentum
$\vec{r}$	position vector
$\vec{v}$	velocity
ξ	uniform distribution [0,1]

1. Introduction

> *“If the planetary structure, with all its order and beauty, is only an effect of the universal laws of motion in matter left to itself, if the blind mechanism of natural forces knows how to develop itself out of chaos in such a marvellous way and to reach such perfection on its own, . . . we derive from a glance at the beauty of the cosmic structure, . . . nature is self-sufficient, the divine rule is unnecessary”*
>
> *–Immanuel Kant,*
> *"Universal Natural History and Theory of the Heavens"*

The contemplation of how one has and, considering the vastness of things, how everything has come to be, has startled the mind and pushed the imagination of us humans probably long before the preservation of written language. Today, the question "How did the Earth form?" does have a pretty well formulated answer. The well established and widely used Solar Nebular Disc Model hypothesised by Safronov (1969) [1] gives a decent answer to begin with: Giant collapsing molecular clouds start spinning by the conservation of angular momentum and heat up while matter is continuously accumulated at the center, forming a protostar slowly growing in size. Gravity, electromagnetic forces and other forces will flatten the cloud that is rotating around the protostar, known as a protoplanetary disk. In order to form a planetary system the disk needs to undergo three stages of transformation. This evolutionary process starts with the accretion of small dust grains(μm) to planetesimals(km) (by coagulation), then these planetesimals(km) form protoplanets($10^3 km$) (by inelastic collision). Then in the late third stage a final assemblage will take place where protoplanets($10^3 km$) will form planets (planetary cores) (by gravity-assisted merging) [2][3].

To test this hypothesis and investigate specific processes in the evolution of these celestial systems, simulations can be made. As described above, the formation of planetary systems is primarily characterised by the merging of particles and their growth in size. In order to replicate the transformation of the disk, it is of the importance to keep track of *if, when,* and *where* two particles are going to come into contact, collide, and merge. Hence the title: Collision Detection, a field of study characterized by answering those three questions. The protoplanetary disk can be considered an N-body system, where every particle is in motion and interacting with every other particle. To keep track of the every body in the system is computationally exhaustive as the number of particles increase. For an N-body system this would result in simulating N^2 interactions. When simulating the motion of these particles with brute force, these N^2 interactions have to be calculated for every time step, making this method more and more computationally expensive. One of the most efficient is the octree code, which has a time complexity of, big O notation, $O(N \log(N))$ [4], and when considering the simulations time step with a fixed integration time it has a time complexity of $O(N^{4/3} \log(N))$ [5].

In this research, the model is simplified to a Keplerian system, where every body is described in a few orbital parameters [6][7][8], omitting the small effects of interpartical interaction and time step determination. The orbits of particles around a central mass are considered fixed, disregarding the observed precession of orbits [6]. The Keplerian method is expected to have a computation time of $N^2\epsilon + (N^4 s^3)/(Ia^3)$ [5]. Astrophysical disk systems that can potentially be simulated by this Keplerian model include but are not limited to protoplanetary disks, accretions disks, debris disks, asteroid belts, planetary systems and satellites in orbit.

In chapter 2 of this report the literature study will be discussed, followed by the experimental method in chapter 3. Chapter 4 is where the results will be presented and is where these results will be discussed. Then, finally, the conclusion of this report will be described in chapter 5. This research report is part of the bachelors program Applied Mathematics and Applied of the Technical University Delft and is an attempt to obtain the degree of Bachelor of Science.

2. Literature

2.1. Protoplanetary Disks and the Formation of Planetary Systems

In 1734 Swedenborg proposed the first theory that is now known as the nebular hypothesis, stating that stars and planetary systems are formed by gas and dust [9][10]. Immanuel Kant developed his hypothesis further in his own work on the mechanisms involved in the process [11]. Even with this theory proposed centuries ago, it still remains a prominent commitment for astronomers today to study planetary formation [12][1]. The formation starts with a giant molecular cloud (*nebula* [13]) collapsing into itself by gravitational pull [11].

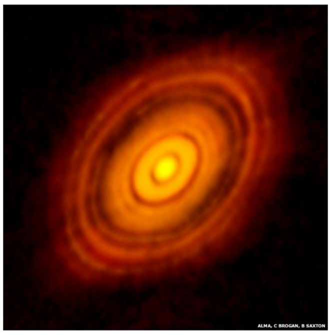


Figure 1: Protoplanetary disk formed around a young star HL Tauri. ALMA 2014 [14]

By the conservation of angular momentum, the collapsing gas cloud will start spinning. Temperature increases as more and more particles collide, and a protostar will form at the centre where most of the matter is accumulated [15]. By a combination of gravity, magnetic forces, pressure and other mechanical phenomena, the remaining spinning gas cloud will eventually flatten and form a rotating disk around the centered star. The resulting structure is called a **protoplanetary disk** [12][16][17] (figure 1). These disks primarily consist of gas and dust, but due to turbulence within the gas the dust and grain particles will clump together via chemical condensation, van der Waals bonding and electrostatic/magnetic forces [2]. This clumping together is called accretion [18]. Then, by gravitational scattering, paths of particles are perturbed enough by bigger clumps resulting in more collisions. By inelastic collision, planetesimals will form and the last stage of planet formation begins [19]. In this last stage the large solids will clump together by mutual gravitational interaction to form large planetary-sized objects. These have been individually observed [20].

Protoplanetary disks are thin structures, meaning that their height is much smaller than their radius, and the rotating disks are only a fraction of the mass of the central protostar [17]. Protoplanetary disks can be described as N-body systems, and to understand their behavior simulations can be run and analysed [21].

2.2. Collision Detection

The subject of collision detection (synonym for: collision determination/intersection detection/interface detection) can be categorized by three main questions; *if*, *when*, and *where* two particles are going to come into contact [22]. To answer these question, you have to keep track of all the particles in the system. Even for a low number of particles in the system this task is surprisingly hard. For instance, Isaac Newton kept getting inconsistent results when trying to explain the motion of the Moon [23]. This is because the Sun-Earth-Moon system could be considered a Three-body problem. It took another century before Poincaré proved that the three-body system is unsolvable: The motion of three bodies is usually non-periodic, with exceptions in special cases [24][25]. When considering N-body systems, collision detection becomes increasingly complex and computationally expensive in the order given [22].

Collision detection plays an essential part in computer-aided machining and environmental simulations for motion planning, engineering analysis, virtual prototyping, robotics, etc. [26]. These simulations are used in astronomy to find if, when and where asteroids, planets or satellites collide. The interpenetration of protoplanetary objects is fundamental during the formation phase of a planetary system [27], and collisions continue to occur in the formed planetary system. An important example would be the collision responsible for the formation of the Moon [28][29].

2.3. Kepler Orbit

As stated before, N-body systems can become increasingly complex to simulate. For every particle in the system the interaction between that particle and every other particle should be taken into account. This method very quickly becomes computationally exhaustive as the number of particles increase [30][31][32].

One method to simplify the N-body system is to assume Keplerian orbits. In a Keplerian system one assumes elliptical orbits around a stationary central mass which occupies most of the mass of the system; *central mass* $\gg$ *mass disk* [33], i.e. planetary systems and planet ring systems. This is not the case for star systems [34]. To understand the motion of the particles in orbit there are some common principles to be acquainted with; Kepler's Laws [35][36], together with some basic elliptical parameters (or Keplerian elements) [6][7][8]:

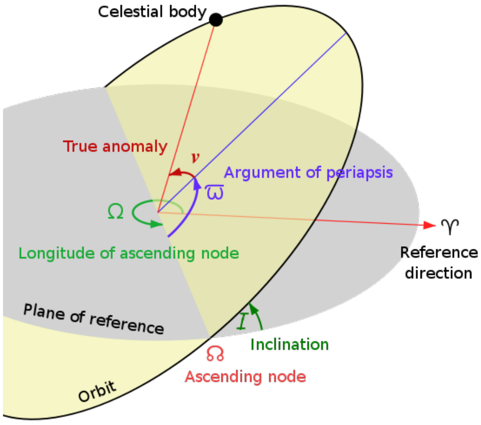


Figure 2: Diagram illustrating and explaining various terms in relation to Orbits of Celestial bodies. [37]

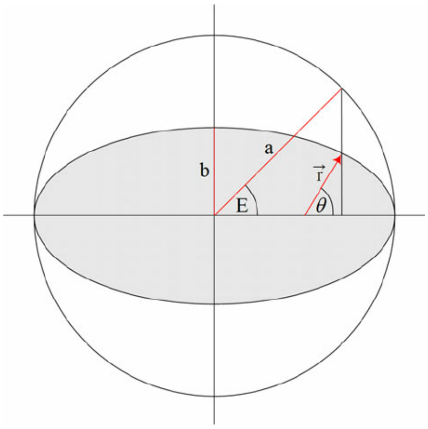


Figure 3: Diagram clarifying the eccentric anomaly E [38]

The elliptical elements: The semi-major axis a is and the semi-minor axis b are the width and the height parameters of the ellipse. The semi-focal separation c is the distance from the centre tot the focal points. The eccentricity ϵ of an orbit describes the elongation of the orbit. These parameters are related by $a^2 = b^2 + c^2, \quad c = a\epsilon, \quad b = a\sqrt{1-\epsilon^2}$. The eccentricity is quantized as follows: $\epsilon = 0$ is a circle, $0 < \epsilon < 1$ is an ellipse, $\epsilon = 1$ is a parabola, and $\epsilon > 1$ is a hyperbola.

The orbital orientation elements: As seen in figure 2, the inclination I is the tilt with respect to the plane of reference. The angle is measured from the ascending node ☊, where the orbit passes upward through the plane of reference. The ascending nodes are on their turn positioned by the Longitude of ascending node. Because these two quantities ultimately report the same information, only the ascending node ☊ will be used in this report, with values ranging from 0 to 2π.

Two remaining elements: The argument of periapsis ϖ describes the orientation of the ellipse in the orbital plane. The eccentric anomaly E describes the position of orbiting particle at time t_0, see figure 3. To know the starting position of your elements is essential for collision detection. The equation of a centred ellipse is $(x/a + \epsilon)^2 + (y/b)^2 = 1$, and the projections in the horizontal and vertical directions are

$$x = a(\cos E - \epsilon) \quad \text{and} \quad y = a\sqrt{1-\epsilon^2}\sin E \tag{1}$$

The ellipse is here is rotated and oriented by the orbital elements mentioned above. So in the reference plane of the ellipse, x is the horizontal position and y is the vertical position. To transform from the orbital coordinate systems with the orbital angels $(\text{☊}, I, \varpi)$ to the Cartesian coordinate system (x, y, z), the use of the constant rotation matrix $\mathscr{R}$ is needed:

$$\mathscr{R} = \begin{pmatrix} \cos\varpi\cos\text{☊} - \sin\varpi\sin\text{☊}\cos I & -\cos\varpi\sin\text{☊} - \sin\varpi\cos\text{☊}\cos I & \sin\varpi\sin I \\ \sin\varpi\cos\text{☊} + \cos\varpi\sin\text{☊}\cos I & -\sin\varpi\sin\text{☊} + \cos\varpi\cos\text{☊}\cos I & -\cos\varpi\sin I \\ \sin\text{☊}\sin I & \cos\text{☊}\sin I & \cos I \end{pmatrix} \tag{2}$$

Now the position vector with Cartesian coordinates can be expressed with the orbital elements mentioned above.

$$\vec{r} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathscr{R}(\text{☊}, I, \varpi) \begin{pmatrix} a(\cos E - \epsilon) \\ a\sqrt{1-\epsilon^2}\sin E \\ 0 \end{pmatrix} \tag{3}$$

Here $\vec{r}$ is the position vector in the reference frame of the large centered mass (e.g. the sun). To calculate the velocity we need the radial distance, the time elapsed since periapsis, and Kepler's third law. These are respectively

$$r = a(1 - \epsilon\cos E), \quad t = \frac{E - \epsilon\sin E}{\omega}, \quad \omega = \sqrt{\frac{GM}{a^3}} \tag{4}$$

Here the radial distance r is obtained by squaring both the equations in (1) and then taking the root of the sum. ω is the mean motion or angular speed $\omega = 2\pi/T$, where T is the orbital period, G is the gravitational constant, M is the mass of the large centered mass, t is the time elapsed since periapsis, E is the eccentric anomaly, and ϵ is the eccentricity the ellipse. The velocity $\vec{v}$ can be expressed as

$$\vec{v} = \frac{d\vec{r}}{dt} = \frac{\omega a}{r}\mathscr{R} \begin{pmatrix} -a\sin E \\ a\sqrt{1-e^2}\cos E \\ 0 \end{pmatrix} \tag{5}$$

The angular momentum $\vec{L} = \vec{r} \times m\vec{v}$, with m being the mass of the body in orbit, is equal to

$$\vec{L} = m\omega ab\mathscr{R} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = m\omega a^2\sqrt{1-\epsilon^2}\mathscr{R} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{6}$$

With ω as the mean motion of the orbit, and a and b being the semi-major and semi-minor axis respectively. The last parameter to help describe a Keplerian orbit is the eccentricity vector $\vec{\epsilon}$.

$$\vec{\epsilon} = \frac{\overrightarrow{LRL}}{GMm^2} = \frac{\vec{v} \times \vec{L}}{GMm} - \frac{\vec{r}}{r} = \epsilon \mathscr{R} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{7}$$

The eccentricity vector $\vec{\epsilon}$ points from apoapsis to periapsis and with magnitude ϵ. $\overrightarrow{LRL}$ is the Laplace-Runge-Lenz vector [39], $\vec{v}$ is the velocity vector, $\vec{L}$ is angular momentum, G is the gravitational constant, M is the mass of the large centered mass, m is the mass of the body in orbit, $\vec{r}$ is the position vector, r the radial distance, and $\mathscr{R}$ is the constant rotation matrix [6][7][8].

The orbits of the particles are also assumed to be stationary, i.e. the argument of periapsis ϖ is assumed to be constant. In reality the orbits of celestial bodies are found to be non-stationary, and actually rotate by a phenomenon called apsidal precession [6]. This precession is caused by a combination of different factors, including general relativity [40], rotational quadrupole bulges on the planet, tides raised on the planet and the star [41], stellar quadrupole moment [6], the Kozai mechanism [42], and perturbing planets [43]. The last one of which is expected to have the most dominant effect of the overall apsidal precession [44]. These effects are not monitored in this research's algorithm. Although the precession rate of planets in our solar system are of the order of magnitude of $\sim 10^5$ years, but is occasionally observed to be a couple of days in rare cases of exoplanets [45]. In addition, the formation of planetary system take multiple Myears to develop. The neglection of this apsidal precession may therefore be considered as the Achilles' heel of the Keplerian model.

This concludes the literature review. In the next section the methods for how these parameters are used in the simulations of this research will be stated.

3. Experimental Method

The algorithm and model is based on the Keplerian orbit methods created by P.M.Visser (2022)[5]. The model is three dimensional. The desired protoplanetary disk is therefore not flat, but more of a triangular torus shape, with its height gradually expanding (Class II protoplanetary disk), as hypothesised in [20][46][47]. The appearances of protoplanetary disks decrease with the stellar age of the central star, therefore the maximum simulation time is set to 10 Myears [48].

3.1. The initialization of the algorithm

The algorithm starts by creating a triangular torus shape disk with a uniform particle distribution. In nature, the densities observed are not at all uniform [49][50]. But in this research it was preferred to keep the model simple, so the algorithm can be better tested, than to go for a more realistic model. Also, disk structure simulations will many times approach approximately identical solutions, regardless of specific initial distributions [51]. But it is of importance that the model is not completely circular symmetric, because in studies it is hypothesised that nonuniformity is necessary to create planetesimals [47].

3.1.1. Initialization, N-body system

Firstly, a few global constants are introduced to the code: gravitational constant $G = 6.674\,28 \times 10^{-11}$N, mass of the sun $M = 1.988\,47 \times 10^{30}$kg, astronomical unit $AU = 1.495\,978\,707 \times 10^{11}$m, radius of the sun $S = 0.00465047$AU, mass of Jupiter $M_J = 1.898 \times 10^{27}$kg, radius of Jupiter $S_J = 0.00047789$AU.

Then, a system is created with N bodies. For each body $j = 1, ..., N$, a list of parameters is chosen and stored: $[t_j^0, a_j, \epsilon_j, s_j, m_j, \vec{r_j^0}, \vec{L_j}, \vec{\epsilon_j}, \omega_j]$. Here t^0 is the time of creation, a is the length of the semi-major axis, ϵ is the eccentricity of the body's orbit, s is the radius of the body, m is the mass of the body, $\vec{r^0}$ is its position at time t_0, $\vec{L}$ is its angular momentum, $\vec{\epsilon}$ is the eccentricity vector, and ω is its mean motion or angular speed.

The procedure of appointing values to these parameters is as follows: For every body j the time of creation in set to $t^0 = 0$. After all, the time of collision needs to be calculated for this collision detection system.

Since the desired disk has a uniform density, the semi-major axis a should not be chosen from a uniform distribution. The further one gets from the sun, the more space there is to be occupied. Let ξ be a random uniform distribution $[0, 1]$, then a is distributed as follows

$$a = \sqrt[3]{(a_+^3 - a_-^3)\xi + a_-^3} \tag{8}$$

The minimal semi-major axis is set at $a_- = 1$AU, because for lower values dust particles are subject to the star's evaporation zone and almost no big protoplanetary objects can form [52][53]. The maximum value is set as $a_+ = 10$AU. At distances larger than 10AU the surface density as well as the average particle size becomes so small that they are dismissed from scope of this research [20] [54]. To maintain this uniform density distribution throughout the entirety of the disk, the inclination I must also be chosen in a specific way. It suffices to take

$$I = \arccos(\xi[1 - \cos(I_{max})] + \cos(I_{max})) \tag{9}$$

Derivations can be found in the appendix. In this research $I_{max} = 0.1$, this was chosen because the inclinations of protoplanetary disks are fairly low [55][28]. In the initialization, the eccentricity ϵ of body j is picked from a uniform distribution: $\epsilon = \epsilon_{max}\xi$, with $\xi = [0, 1]$ a uniform distribution and $\epsilon_{max} = 0.1$. In the solar system, the eccentricities of the planets are relatively small ($\epsilon < 0.1$), with mercury as the exception with $\epsilon = 0.2056$. Various studies have shown that the eccentricities of the orbital bodies are damped over time [55][56][57]. This, together with the appeal to keep the model simple, gives the model enough room to choose its eccentricity ϵ from a $\epsilon = \epsilon_{max}\xi$ distribution.

Because of the many different sizes of bodies during planetary formation, the radius s of body j is also picked at random. The radius of Jupiter $R_J = 4.779 \times 10^{-4}$AU is a good measure for the radii of the other bodies, since it is the largest planet in the solar system. Therefore s is chosen around that radius with the distribution $s = 1\text{AU} \cdot 10^{-7+4\xi}$.

The mass of each body is chosen to be the mass of Jupiter $M_J = 1.898 \times 10^{27}$kg divided by the number of elements in the N-body system. so $m = M_J/N$. The mass of Jupiter accounts for almost all the mass in the solar system, therefor its mass is distributed among the bodies in the system to create a pre-solar-system-like environment.

To ensure axisymmetry, both the argument of periapsis ϖ, the ascending node ☊, and the true anomaly E are all separately sampled from a uniform distribution with range $[0, 2\pi]$.

The initial position vector $\vec{r^0}$ is calculated with a, ϵ, E, ϖ, ☊, and I using (3), and the angular speed ω is calculated with G,M, and a using (4). Then at last the angular momentum $\vec{L}$ is calculated with m, ω, a, ϵ, ϖ, ☊, and I using (6), and $\vec{\epsilon}$ with ϵ, ϖ, ☊, and I using (7)

This process of sampling and calculating is repeated for every j until N lists of parameters are filled. The last step of creating the N-body system is by *ordering* the set of bodies by increasing periapsis, including planet radius. Hence by increasing value of $(a_j - \epsilon_j a_j - s_j)$. All listed bodies then lose their original numbering, and obtain their new numbering j, synonym for their rank in the list e.g. $j = 1$ has the smallest value for $(a_j - \epsilon_j a_j - s_j)$ and $j = N$ has the largest. This finalizes the initialization of the **particle list**.

3.1.2. Initialization, collision pairs

Up next, a pair list will be made of all pairs that can possibly collide. Instead of checking each possible pair for each particle, the algorithm only checks the a few pairs within a vicinity of the particle's periapsis. To be more specific: the reach of a particle j is limited by its semi-major axis a_j, its eccentricity ϵ_j, and its radius s_j. It has an inward reach (periapsis) of $(a_j - \epsilon_j a_j - s_j)$ and an outward reach (apoapsis) of $(a_j + \epsilon_j a_j + s_j)$. So particle j reaches values between the interval

$$[a_j - \epsilon_j a_j - s_j, a_j + \epsilon_j a_j + s_j] \tag{10}$$

This means that the algorithm only has to check those particles with overlapping reach. And since the particle list is sorted; piece of cake.

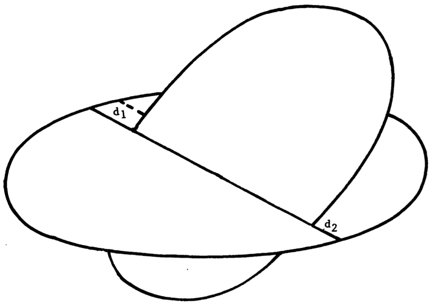


Figure 4: Illustration of two orbits. Assuming their radii are much smaller than their orbits, collisions may occur near their mutual nodal line. This line is of the intersection of the two orbital planes. [58]

Let $i = 1...N$ and let $j = i + 1, i + 2, ..., N$. Because of the list being well-ordered, it can be seen that $(a_i - \epsilon_i a_i - s_i) \leq (a_j - \epsilon_j a_j - s_j)$ [59].

So while $(a_i + \epsilon_i a_i + s_i) \geq (a_j - \epsilon_j a_j - s_j)$ holds, i and j are a candidate for collision and their orbits have to be analysed (later explained) [58]. j can be increased by 1 and check if the inequality hold. Once $(a_i + \epsilon_i a_i + s_i) > (a_j - \epsilon_j a_j - s_j)$ the body j is out of reach and so will all other bodies with a numbering higher than j, so no more collisions. i can be increased by 1 and repeat the checking. This process is called *sweep and prune* [60].

If a pair i, j is a candidate for collision, their minimum orbit intersection distance (MOID) is calculated [61]. This distance is analytically evaluated for a point near the nodal lines, see figure 4 for clarification. In the case where the MOID is smaller than $s_i + s_j$, then a collision is most likely to occur somewhere in the future. Because we assume all radii to be smaller than their orbits $s \ll a$, the MOID needs to be close to the nodal line, i.e. the line of intersection of their orbital planes. The nodal line is determined by [5] [58] [62]

$$\vec{K} = \pm \vec{L_i} \times \vec{L_j} \tag{11}$$

where $\vec{K}$ is the directional vector of the nodal line and $\vec{L_i}, \vec{L_i}$ are the angular momenta of bodies i, j. The ± indicate the two opposite directions in which the vector can point, denoting the two regions where the orbits intersect and where collision may occur. The intersection points are obtained by

$$\vec{r_i} = \frac{(1-\epsilon_i^2) a_i \vec{L_i} \times \vec{L_j}}{\pm|\vec{L_i} \times \vec{L_j}| + \vec{\epsilon_i} \bullet (\vec{L_i} \times \vec{L_j})} \tag{12}$$

$$\vec{r_j} = \frac{(1-\epsilon_j^2) a_j \vec{L_i} \times \vec{L_j}}{\pm|\vec{L_i} \times \vec{L_j}| + \vec{\epsilon_j} \bullet (\vec{L_i} \times \vec{L_j})} \tag{13}$$

Here $\vec{r_i}$ is the intersection point vector, a_i the semi-major axis, ϵ_i the eccentricity, and $\vec{\epsilon_i}$ the eccentricity vector of orbit i. Their velocities $\vec{v}$ can be expressed by [39]:

$$\vec{v} = \frac{GMm}{\vec{L} \bullet \vec{L}} \vec{L} \times \left(\vec{\epsilon} + \frac{\vec{r}}{r}\right) \tag{14}$$

Here G is the gravitational constant, M is is the mass of the heavy central mass, m the mass of the orbiting body, $\vec{L}$ the bodies angular momentum, $\vec{r}$ as positional vector of the body, and r the radial distance as described in (4). To approximate the MOID, the motion of i and j is linearized, this can be done because again: $s \ll a$. The tangent line at $\vec{r_i}$ points in the direction of $\vec{v_i}$. The same applies to j. If we follow those tangent lines, hence the linearization, we find that the distance between the two lines is given by

$$d = \left| (\vec{r_j} - \vec{r_i}) \bullet \frac{\vec{v_i} \times \vec{v_j}}{|\vec{v_i} \times \vec{v_j}|} \right| \tag{15}$$

Now the positions where the distance is minimal are expressed as

$$\vec{r_i}^{min} = \vec{r_i} + \left((\vec{r_j} - \vec{r_i}) \bullet \frac{\vec{v_j} \times (\vec{v_i} \times \vec{v_j})}{|\vec{v_i} \times \vec{v_j}|^2} \right) \vec{v_i} \tag{16}$$

$$\vec{r_j}^{min} = \vec{r_j} + \left((\vec{r_j} - \vec{r_i}) \bullet \frac{\vec{v_i} \times (\vec{v_i} \times \vec{v_j})}{|\vec{v_i} \times \vec{v_j}|^2} \right) \vec{v_j} \tag{17}$$

$$|\vec{r_i}^{min} - \vec{r_j}^{min}| < s_i + s_j \tag{18}$$

The last equation states the condition that whenever the distance between these two points is smaller than the sum of their radii, then bodies i and j collide [5]. One needs to keep the ± sign in mind; for every pair, two locations have to be checked.

After applying the sweep and prune filter and checking for all remaining pairs if i and j collide, all pairs that satisfied condition (18) can be stored in a **collision pair list**. Up next is the calculation of the collision time.

3.1.3. Initialization, time of collision

The time it takes for body to move from position $\vec{r}^0$ to position $\vec{r}^1$ is given by [5]

$$t^1 - t^0 = \frac{E^1 - E^0}{\omega} - \frac{\vec{\epsilon} \times (\vec{r}^1 - \vec{r}^0)}{1 - \vec{\epsilon} \bullet \vec{\epsilon}} \bullet \frac{\vec{L}}{GMm} \tag{19}$$

Here t^1 and t^0 are the times at $\vec{r}^1$ and $\vec{r}^1$, E^1 and E^0 are the eccentric anomalies of $\vec{r}^1$ and $\vec{r}^1$, ω is the mean motion of the body, $\vec{\epsilon}$ is the eccentricity of the orbit, $\vec{L}$ is the angular momentum, G is the gravitational constant, M is the mass of the heavy central mass, and m is the mass of the body. To calculate the difference in eccentric anomalies $E^1 - E^0$ the following expressions can be used.

$$E^1 - E^0 = \arg(z) \tag{20}$$

$$z = \left(\frac{\vec{r}^1}{a} - \frac{i|\vec{r}^1|\vec{v}}{a^2\omega}\right) \bullet \left(\frac{\vec{r}^0 - \vec{\epsilon}\vec{\epsilon} \bullet \vec{r}^0}{a - a\vec{\epsilon} \bullet \vec{\epsilon}} + \vec{\epsilon}\right) \frac{\vec{r}^0 \bullet \vec{\epsilon}}{a} + \vec{\epsilon} \bullet \vec{\epsilon} \tag{21}$$

When taking $\vec{r}^{min}$ for $\vec{r}^1$, the first passage of crossing t^1 can be calculated. These can differ for the two colliding bodies, so t_i^1 and t_j^1 can differ. From here the collision time $t_{i,j}$ can be calculated. For simplicity we take $i = 1$ and $j = 2$. An equality may look like this

$$t_{1,2} = t_1^1 + T_1 k + dt_1 = t_2^1 + T_2 l + dt_2 \tag{22}$$

t_1^1 and t_2^1 are the times of first passage of crossing, T_1 and T_2 are the orbit period of body 1 and 2, with $T = 2\pi/\omega$ and ω being the bodies mean motion, k and l are integers indicating the number of completed orbits of the bodies, and dt_1 and dt_2 are small margins that allow for the fact that the bodies do not need to be exactly at $\vec{r}^{min}$ when the bodies collide. The two bodies will ultimately collide, unless they are in mean-motion resonance [63].

To find smallest k and l, a combination must be found that satisfies the following condition

$$\left|t_1^1 + T_1 k - t_2^1 - T_2 l\right| < \frac{|\vec{v}_{12}|\sqrt{(s_1 + s_2)^2 - |\vec{r}_2^{min} - \vec{r}_1^{min}|^2}}{|\vec{v}_1 \times \vec{v}_2|} = \delta|t_1^1 - t_2^1| \tag{23}$$

Where s_1, s_2 are the radii of the bodies and $\vec{v}_{12} = \vec{v}_2 - \vec{v}_1$, and

$$p = \frac{T_1}{|t_1^1 - t_2^1|}, \quad q = \frac{T_2}{|t_1^1 - t_2^1|} \tag{24}$$

p, q, and δ are dimensionless parameters. This allows the problem to be rewritten as

$$1 - \delta < kp - lq < 1 + \delta \tag{25}$$

This means that k and l are the smallest integer points that lie in between the lines

$$x = \frac{q}{p}y + \frac{1-\delta}{p} \quad \text{and} \quad x = \frac{q}{p}y + \frac{1+\delta}{p} \tag{26}$$

To find k and l an algorithm is implemented that makes use of the continued-fraction representation of p/q [64]. The iteration scheme is similar to *lattice base reduction* [65] and works by searching entire parallelograms before changing the basis. The algorithm works as followed [5]:

Start with $n = 0$

$$\begin{aligned}
&q_0 = \frac{2\pi}{\omega_1|t_1^1 - t_2^1|}, && k_0 = 1 \\
&q_1 = \frac{2\pi}{\omega_2|t_1^1 - t_2^1|}, && k_1 = 0 \\
&\alpha_{2n} = \left\lfloor \frac{q_{2n}}{q_{2n+1}} \right\rfloor \\
&q_{2n+2} = q_{2n} - \alpha_{2n}q_{2n+1} \\
&\text{if } q_{2n+2} = 0 \implies \text{next pair} && \text{(can be neglected)}\,^{1} \\
&k_{2n+2} = k_{2n} - \alpha_{2n}k_{2n+1} \\
&\alpha_{2n+1} = \left\lfloor \frac{q_{2n+1}}{q_{2n+2}} \right\rfloor \\
&q_{2n+3} = q_{2n+1} - \alpha_{2n+1}q_{2n+2} \\
&\text{if } q_{2n+3} = 0 \implies \text{next pair} && \text{(can be neglected)} \\
&k_{2n+3} = k_{2n+1} - \alpha_{2n+1}k_{2n+2}
\end{aligned}$$

and then test the coordinates [67]

$$x = \left\lceil \frac{1-\delta}{q_{2n}} \right\rceil, \ldots, \left\lfloor \frac{1+\delta}{q_{2n+2}} \right\rfloor, \qquad y = \max\left(0, \left\lceil \frac{q_{2n}x - 1 - \delta}{q_{2n+1}} \right\rceil\right) \tag{27}$$

whether they satisfy

$$xq_{2n} - yq_{2n+1} > 1 - \delta \tag{28}$$

If none of the points (x, y) from (27) satisfy this inequality, increase n by 1 and start from top. If the inequality is satisfied, the solution becomes

$$k = xk_{2n} - yk_{2n+1}, \qquad t^0 = t_1^1 + \frac{2\pi}{\omega_1} \tag{29}$$

Where t^0 is the collision time between body 1 and 2.

This process needs to be repeated for every pair in the collision list. Then, every pair i, j together with their collision time t^0 are stored in a **time-ranked collision list**. This finalizes the initialisation of the algorithm.

[1] probability of picking a rational number from a real number line = 0 [66]

3.2. Creating a new particle from a merger.

Every collision in the simulation was assumed to be inelastic, so calculations for in a simple merger were used. This however does not portray reality for bodies with radii $s > 10^3$km. Oblique collisions transfer orbital angular momentum into planetary spin, combined with the fact that fragmentation occurs, indicate that the simple mergers method for these collisions is not desirable [68]. Yet again; in this research it was preferred to keep the model simple, so the algorithm can be better tested, than to go for a more realistic model. So when two bodies i and j collide, we assume a simple merger. The creation of the new particle follows simple conservation laws [5]:

$$\begin{aligned} s_{new} &= \sqrt[3]{s_i^3 + s_j^3} \\ m_{new} &= m_i + m_j \\ \vec{r}_{new} &= \frac{m_i\vec{r}_i + m_j\vec{r}_j}{m_{new}} \\ \vec{v}_{new} &= \frac{m_i\vec{v}_i + m_j\vec{v}_j}{m_{new}} \\ \vec{L}_{new} &= \vec{L}_i + \vec{L}_j \end{aligned} \tag{30}$$

$$\begin{aligned} l_{new} &= \frac{\vec{L}_{new} \bullet \vec{L}_{new}}{GMm_{new}^2} \\ r_{new} &= \sqrt{\vec{r}_{new} \bullet \vec{r}_{new}} \\ \vec{\epsilon}_{new} &= \frac{\vec{v}_{new} \times \vec{L}_{new}}{GMm_{new}} - \frac{\vec{r}_{new}}{r_{new}} \\ \epsilon_{new} &= \sqrt{\vec{\epsilon}_{new} \bullet \vec{\epsilon}_{new}} \end{aligned} \tag{31}$$

Here, s is the body radius, m is the body's mass, $\vec{r}$ is the position vector, $\vec{v}$ is the body's momentum, $\vec{L}$ is angular momentum, l is the semi-latus rectum of the orbit. r is the radial distance, $\vec{\epsilon}$ is the eccentricity vector with ϵ as the eccentricity of the new particle. Before we introduce this new body to the particle list, first check conditions

$$l_{new} > (1 + \epsilon_{new})S \quad \textbf{and} \quad \epsilon_{new} < 1 \tag{32}$$

Only then the new particle stays in the system. S is the radius of the central mass. If $l \leq (1+\epsilon)S$ the body collides with the central mass, and if $\epsilon \geq 1$ the body escapes the system. Regardless of the satisfaction of the conditions, the bodies i and j are deleted from the particle list. When both conditions of (32) are satisfied the new body can be added to the particle list with its new parameters

$$a = \frac{l}{1 - \epsilon^2}, \quad \omega = \sqrt{\frac{GM}{a^3}} \tag{33}$$

3.3. The Main loop of the algorithm

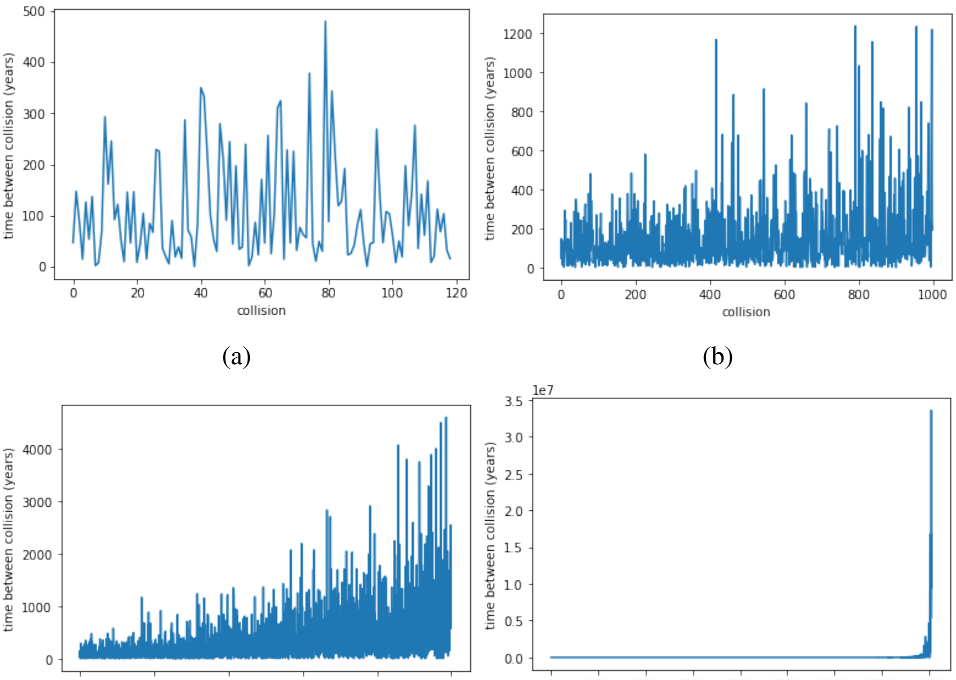
1. If the time-ranked collision list is empty, end the simulation.
2. Take the collision pair i, j on top of the timed collision list, so the smallest collision time and add this to the **chronological collision list**.
3. With the time-ranked collision and the particle list, calculate a new time-ranked collision list and particle list. This is done by:
    (a) Temporary store the data of the i, j pair from the time-ranked collision list.
    (b) Remove the pair and remove any pair containing either i or j from the time-ranked collision list.
    (c) In the time-ranked collision list, for any pair containing an index $\geq i$, subtract that index by 1.
    (d) In the time-ranked collision list, for any pair containing an index $\geq j$, subtract that index by 1.
    (e) Create a new body from the collision of i, j with its time of creation t^0 equal to the time of collision of bodies i, j.
    (f) If the orbit of the new body collides with the central mass or leaves the system, go back to 1. of the main loop.
    (g) Put the new body in the particle list, sort this list again and mark the new particles numbering k.
    (h) Calculate all new possible collision pairs with respect to k. Check for all bodies l with a smaller apsiosis $(a_k + \epsilon_k a_k + s_k \geq a_l - \epsilon_l a_l - s_l)$, but this time, start at $l = 0$. When $(a_k + \epsilon_k a_k + s_k < a_l - \epsilon_l a_l - s_l)$ no more new pairs shall be detected.
    (i) For the new pairs k, l_n, calculate the collision time.
    (j) In the time-ranked collision list, for any pair containing an index $\geq k$, add 1 to that index.
    (k) Add all new pairs k, l_n to the time-ranked collision list, and order the list by collision time.
4. Store values of interest.

4. Results and Discussion

First the results and findings of the simulation are presented and analysed, going over the behaviour in changes of different Keplerian parameters, as well as collision time and the change in the number of particles in the system. In the next section the results of the method itself are analysed, going over the number of checks, pairs, and collisions for different sizes of the N-body system, as well as runtime for the initialisation, the main loop, and the total run time for different sizes of the N-body system.

4.1. Results of the simulation

Figures 5-11 show the results of a system created with $N = 10.000$. The semi-major axis a was picked from distribution (8) with $a_- = 1\text{AU}$ and $a_+ = 10\text{AU}$, I from (9) with $I_{max} = 0.1$, ϵ from uniform distribution [0,0.1] such that the mean eccentricity is 0.05 as in [69], s from $s = 1\text{AU} \cdot 10^{-7+4\xi}$ as described in section 3.1.1. In the model it was chosen to uniformly distribute the bodies throughout the disk.



(a)

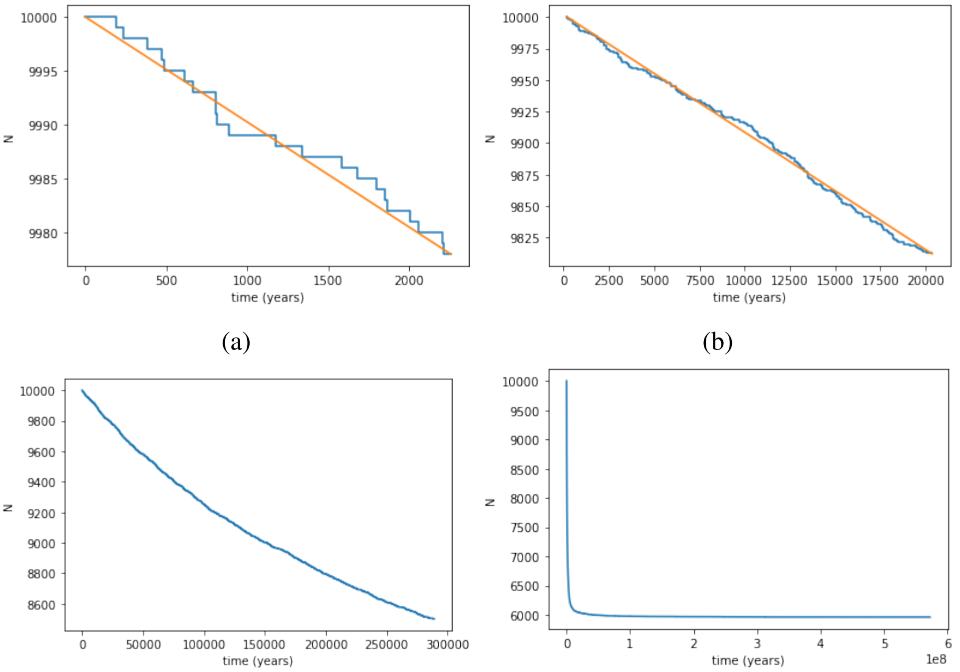
(b)

(c)

(d)

Figure 5: The time between collisions plotted for a number of consecutive collisions for a system starting with $N = 10.000$ bodies, and a total of 4037 collisions. (a) shows the time between collisions for the first 120 collisions, (b) for the first 1000 collisions, (c) for the first 2500, and (d) for the first 4030.

In figure 5 the collision times are shown. In each of the sub-figures the times between the consecutive collisions is displayed. Placed on the horizontal axis is the index of the collision. It can be seen that in the beginning of the simulation the time between collisions seems quite random. The time between collisions fluctuates about a hundred years and does not seem to de- or increase. Only after a mere thousand collisions a perceivable increase can be noted, which when the scope is broadened, increases in an exponential manner. Figure 5(d) is added to portray the explosive behaviour of the trend. The collision time is further displayed in figure 6. Both figures are in agreement with similar studies [70].



(a) (b)

(c) (d)

Figure 6: The time of collision measured from the start of the simulation plotted for a number of consecutive collisions for a system starting with $N = 10.000$ bodies, and a total of 4037 collisions, revealing mostly the same as figure 5. (a) shows the number of particles in the system versus time for the first 22 collisions, (b) for the first 190 collisions, (c) for the first 1500 collisions, and (d) for all 4037 collisions. A line is added to (a) and (b) to exhibit linearity. The slope in (a) corresponds to 1 collision every 102.6 ± 3.1 years, and in (b) to 1 collision every $108.3 \pm 4 \times 10^{-5}$.

In figure 6 the number of particles in the system is plotted against time. The decrease seems random in the beginning, which is consistent with figure 5, with an average of 102.6 ± 3 years between each collision for the first 22 collisions. This number says fairly constant with an average of $108.3 \pm 4 \times 10^{-5}$ years between each collision for the first 190 collisions. This matches what is shown in figure 5(a,b). Later the trend shows the exponential increase in collision time as seen the previous figure. Obviously the number of particles in the system decreases by one after each collision. The system that is left after a substantial number of collisions is therefore more empty. Thus the chances of a collision occurring drops, resulting in longer time between collisions.

4.1.1. Keplerian parameters

To display the distribution of the orbital profile of the protoplanetary disk the semi-major axis a of every particle in the system is plotted against its eccentricity ϵ in figure 7. On the left (blue) the initial state is displayed, with the semi-major axes distributed as in (8), and a clear

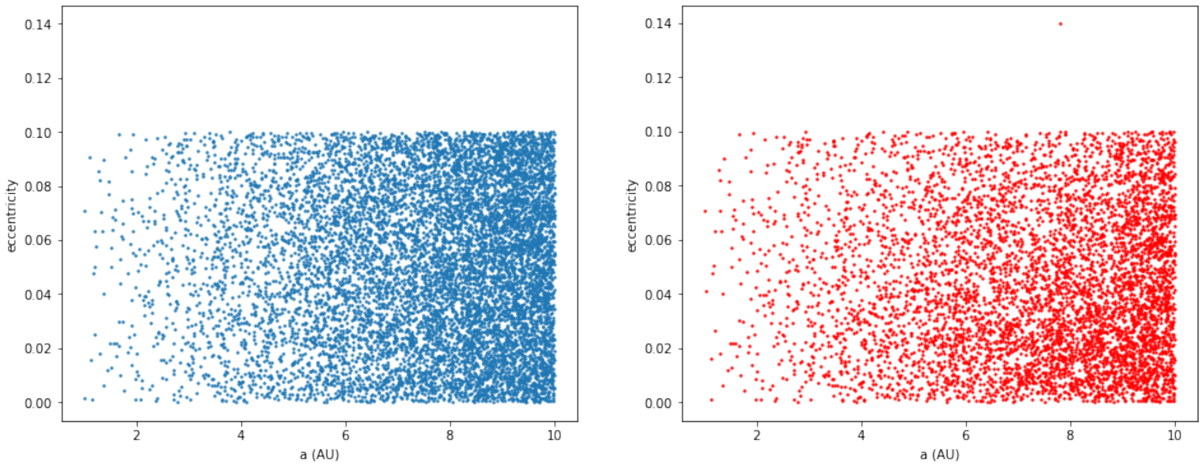


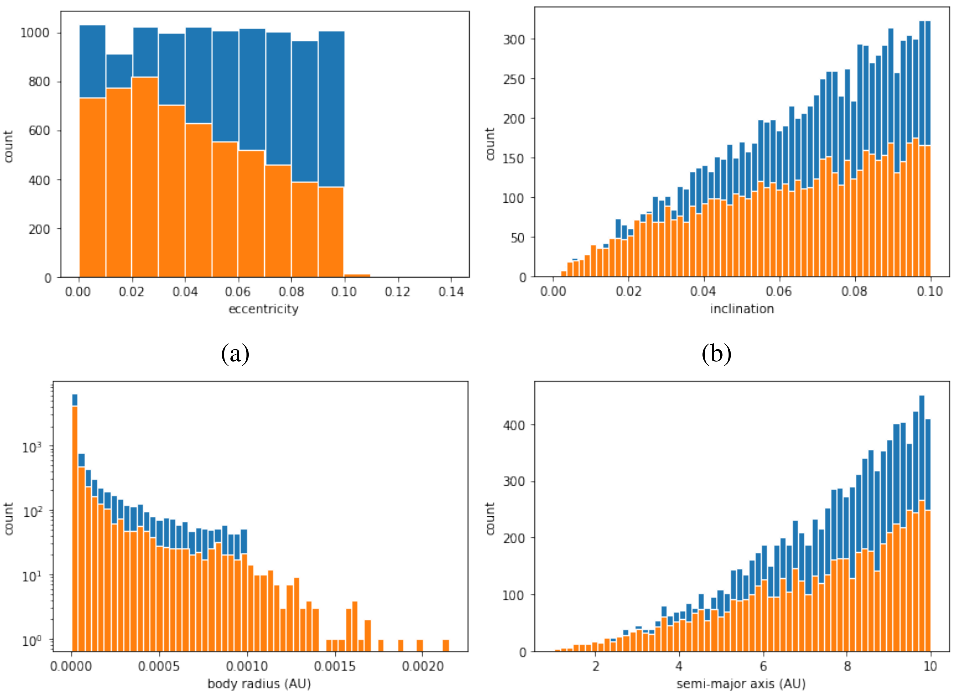
Figure 7: Semi-major axis versus eccentricity for initial system with $N = 10.000$ (left, blue) and final system with $N = 5.963$ (right,red). For the initial state: The distribution of ϵ is clearly uniform as well as the increasing probability of a. For the final state: a decrease in overall density is seen, especially for higher values for eccentricity.

uniform distribution for the eccentricity. On the right (red) the final distribution of these orbital parameters. A moderate reduction in the number of particles displayed can be seen, especially for eccentricities in the upper half (above 0.05). One outlier with an an eccentricity of 0.14 is displayed as well. Eccentricity build-up is a common effect in disk evolution [71]. These distribution changes are more extensively displayed in figure 8 with some additional parameters of interest.

The eccentricity, starting with an almost uniform distribution, shows high damping activity during the simulation, shown in figure 8(a). Bodies with higher eccentricities collide more because their broader range in (10) gives them more possible collision pairs, and thus have a greater tendency to collide [72]. The distribution of eccentricity at the end of the simulation shows a peak at an eccentricity around 0.02, hinting at a critical eccentricity of around 0.025 [55]. When two bodies collide the remaining merged body establishes an eccentricity close/-closer to 0.02. This results in a less damped effect in eccentricity around this point. The critical eccentricity highly influences the eccentricital damping of the system [72].A critical eccentricity of 0.02 is the same order of magnitude as those found in similar studies; 0.0315 [55], 0.027, and 0.015 [73]. The shape of the eccentricity distribution slowly takes shape of that of observed distributions in asteroid belts [74].

The distribution of the body radius shows an overall decrease from the initial distribution, with the newly form bodies with larger radii added to the final distribution. A few bodies are displayed with larger radii, standing out from the rest of the distribution. These are probably the result of a number of collisions, with a larger radius after every consecutive collision, and with less particles around to collide with, hence their isolated position. The distribution perserves its shape, and is only roughly reduced by a constant factor ($\approx 1/4$). The change in shape of the distribution is comparable with those in related studies [75][76].

The distribution of semi-major axes shows a decrease especially at larger axes. This corresponds with figure 9(g), where a decrease in overall semi-major axes is seen. At smaller axes, barely any decrease is seen in the distribution, which on its turn corresponds to figure 7. For these values, there are too few bodies are around to collide with.



(a)

(b)

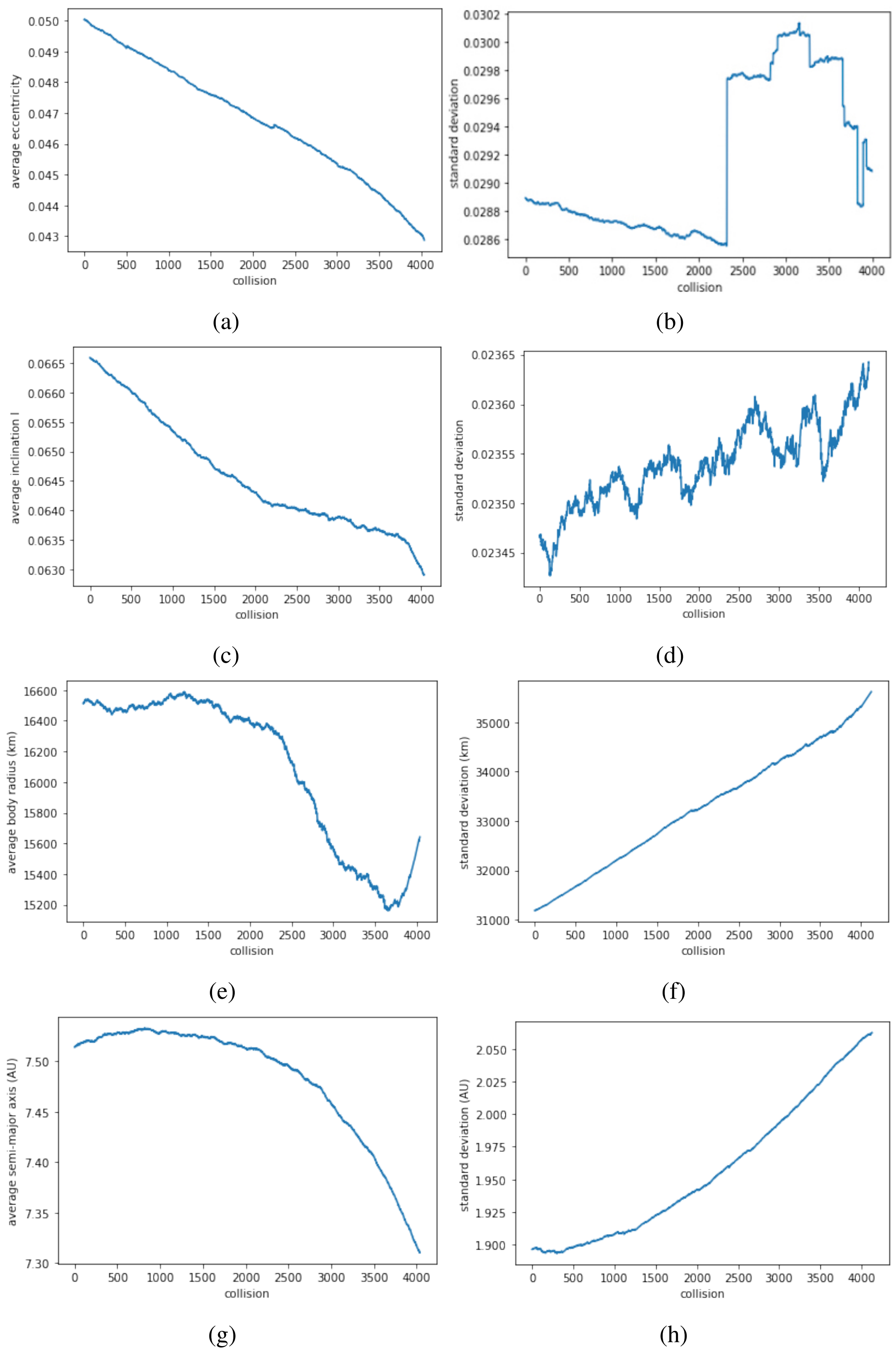
(c)

(d)

Figure 8: The distribution of some important orbital parameters, plotted at the start of the simulation (blue) and after no collisions were left in the collision list (orange). System started with $N = 10.000$ and ended with $N = 5.963$. Displayed are the distributions of (a) the eccentricity with binsize 14, (b) the inclination with binsize 63, (c) the body radius with binsize 58, and (d) the semi-major axis with binsize 60.

In order to understand how the system behaves and how it changes after every consecutive collision figure 9 is portrayed how the averages and standard deviations change after the collisions. Note; the x-axis does not display the time but the index of every collision. To see the same parameters set out over time, regard figure 10. The eccentricity together with the inclination, gradually but steadily declines over time. The average body radius is more or less constant in the beginning, then declines rapidly, followed by a steep growth. The average semi-major axis increases with a constant rate for the first few collisions, and then exponentially decreases for the following collisions. All standard deviations have increased from their initial value. The standard deviation of the eccentricity first shows a steady decrease, but increases considerately by certain jumps, relating to the high eccentricities that can manifest from certain merges as seen in figure 7. The standard deviation of the inclination stays almost the same when considering the scale of the graph. The standard deviation of the body radius as well as the semi major axis seem to gradually increase, The overall growth of the bodies in the system (as depicted in figure 8(c)) accounts for the increase in standard deviation of the body radius, and the 'flattening' of the semi-major axis distribution from figure 8(d) accounts for the increase in standard deviation of the semi-major axis. The shapes of the average inclination graph and the average body radius graph suggest that collisions take place in a 'three-stage-rocket' sequence, split up in; the first 2200 collisions, then from 2200 up to 3700, and then the last collisions after 3700.

[2] the standard deviation of ϵ is given by $\sqrt{\langle\epsilon^2\rangle - \langle\epsilon\rangle^2}$



(a)

(b)

(c)

(d)

(e)

(f)

(g)

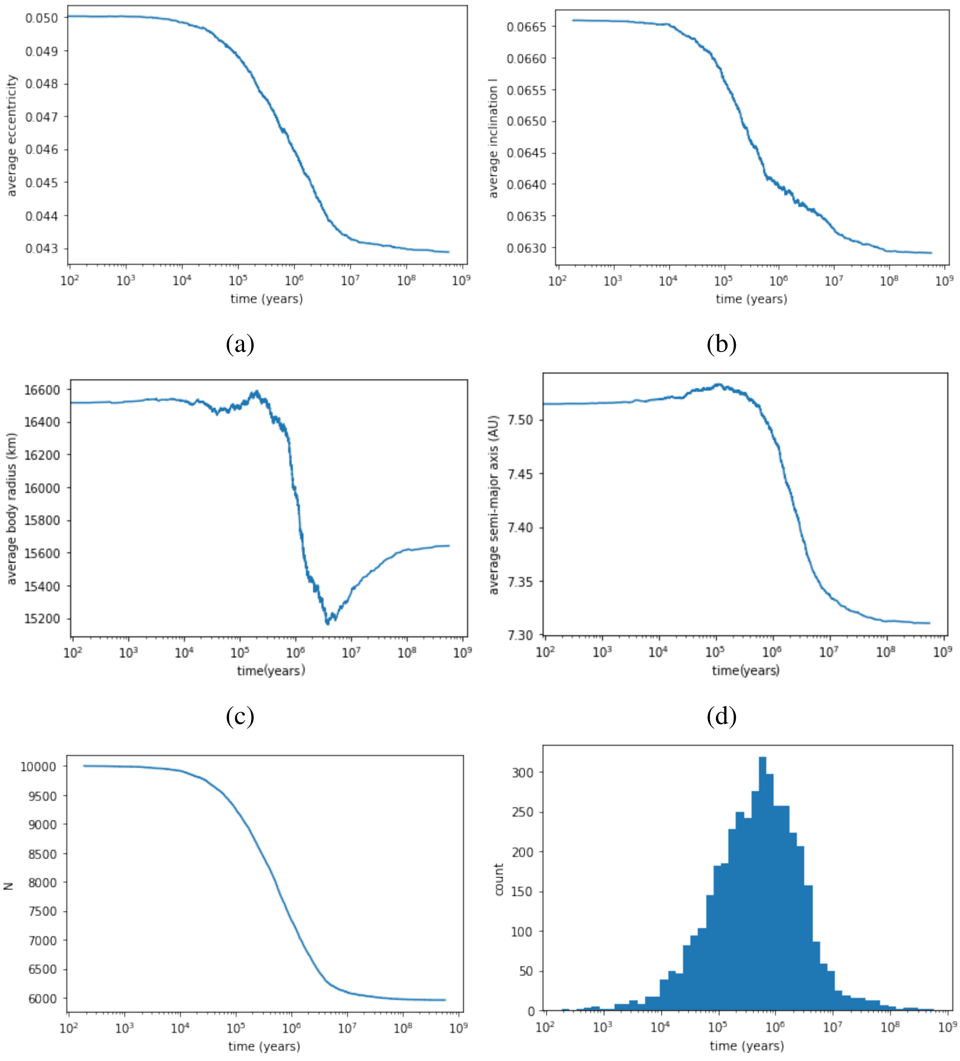
(h)

Figure 9: The averages and standard deviations of some important orbital parameters calculated after all of the 4037 collisions. System started with $N = 10.000$. (a) shows the average eccentricity $\langle \epsilon \rangle$ after every collision, (b) the standard deviation[2]of ϵ, (c) the average inclination $\langle I \rangle$, (d) the standard deviation of I, (e) the average body radius $\langle s \rangle$, (f) the standard deviation of s, (g) the average semi-major axis $\langle a \rangle$, and (h) the standard deviation of a. All values are plotted against collision index.

For the first stage: The almost constant average body radius graph and average semi-major axis graph suggest that collisions happen quite randomly in the system, regardless of size or orbital radius. Ergo, collisions happen rather uniformly throughout the disk regardless of specific parameters. Presumably a little bit more collisions occur between bodies with smaller orbits, hence the small increase in semi-major axis.

Then for the second stage: The system is a little bit more empty due to the number of bodies that have merged in the first stage. Now only larger bodies collide, since they have a larger probability of colliding. These events can be seen in the steep decline in average body radius. The less steep decline of the average inclination suggests that the large bodies colliding already have collided a few times already, with a less prominent damping effect as result.

When considering the last stage, the shape of the graph in 9(g) has to be noted: Bodies with large semi-major axes are colliding in this stage. These collisions take an extremely long time to occur. This information resonates with that which is seen on the graphs in this figure. A steep decline in inclination relates to a larger angle between the orbital planes of the two colliding bodies, hence a smaller dt's in (22) hence a smaller probability of colliding. Also, larger inclination means that the body spends more time in rarefied regions of the disc. The steep increase in body radius insinuates that the smaller bodies in the system finally collided with each other. And the ever so steep decrease in semi-major axis suggests that the small bodies far from the central body with the largest orbital time finally coincided with each other.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 10: The averages some important orbital parameters plotted against a logarithmic time scale. The (a) average eccentricity $\langle \epsilon \rangle$, (b) average inclination $\langle I \rangle$, (c) average body radius $\langle s \rangle$, and (d) average semi-major axis $\langle a \rangle$ are plotted against a logarithmic timescale. For reference (e) the number of particles N over time and (d) the distribution of collision time with binsize 50 (Note: log-scaled) are also displayed. The system started with $N = 10.000$.

Figure 10 portrays the same information as figure 9 but with different scaling. For reference the number of particles over time is also displayed. To some degree, the average eccentricity (a) and inclination (b) decline with the same shape as the number of particles (e), with the exception around 1Myear to 10Myear for the inclination. This, together with the concomitant decline in average body radius suggests a period of collisions with particles that have previously collided a few times already. Figure (e) and (f) are different representations of the same information.

When comparing the average body radius to the collision time distribution, some insights can be made. From the start the average body radius is rather constant and only starts to drop significantly around 1 Myears The distribution of collision time shows an exponential increase up until 1 Myears (logical consequence of logarithmic scale). This together with figure 22(c) makes a parallel to the first stage of the process as described on previous pages, where a random collision happen quite uniformly throughout the disk. Stage two would be between 1 Myears and 4 Myears, regarding the shape of both figure 10(c) and 10(f). Then the last stage starting from 4 Myears, sweeping a substantial amount of particles with a small body radius. From time from 1 Myears to 10 Myears, the average semi-major is in decline. This resonates with asteroid belts diameters decreasing over large timescales [77][78], as well as the phenomenon of migration of planetary objects [79].

So far the results shown had been produced by a system of $N = 10.000$ starting particles. To check the consistency of the system as well as the behaviour of the system for smaller starting particles, the simulation was run multiple times for $N = 5.000$. The results of these runs are illustrated in figure 12.

The systems shows very similar behaviour compared to the $N = 10.000$ displayed in figure 9 and 10. Averages of eccentricity, inclination and semi-major axis are increasing over time, and the standard deviations of the inclination, body radius, and semi-major axis are also increasing.

Differences with respect to the $N = 10.000$-system are the standard deviation of the eccentricity and the average body radius. The standard deviation of the eccentricity in figure 12 shows an overall decreasing behaviour among all 5 runs, but continues to express the same jumping behaviour as in figure 9(b). The average body radius exhibits an overall increase in value. The drop seen in figure 9(c) is almost nonexistent in the five runs showcased here. Hence it can be extrapolated that the second stage, characterised by large body merges, is less prominently dominant in the previously described three-stage process. Because of a lower number of starting particles, there are fewer collisions, causing less growth for the larger bodies, and hence a lower probability of merging with particles with nearby apoapses. However, the last stage, characterised by small body merges and long collision times, remains strongly present. This causes the average body radius to go up in the last stage, resulting in an overall larger body radius throughout the system.

4.1.2. Planetary formation

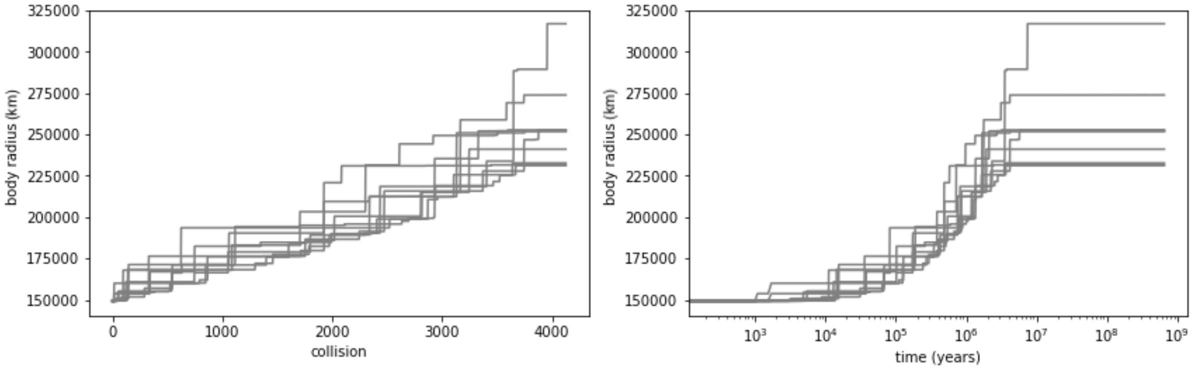
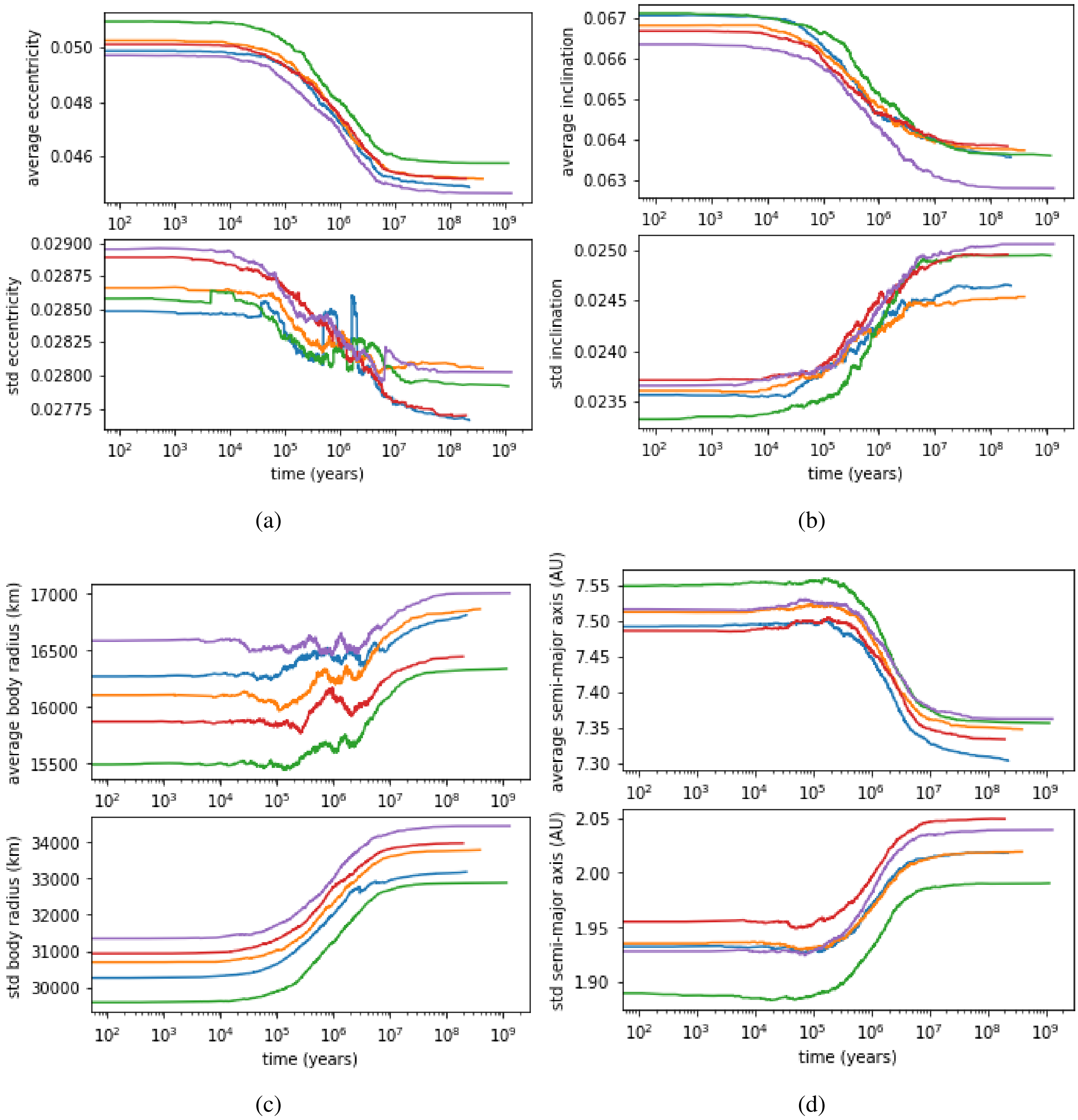


Figure 11: The largest 10 bodies and their growth plotted over collision index (left) and plotted over a logarithmic timescale (right). The system started with $N = 10.000$.

In order to gain insight on whether the model was able to produce significantly large planetary objects, the model kept an eye on the 10 largest bodies. In figure 11 the growth of the 10 largest bodies are displayed to portray their development during the simulation. It can be seen that even though the average body radius of the system decreased, the largest objects all increased a substantial amount, starting around 1.5×10^5 km and reaching body radius of 2.3×10^5 for the 10th largest body and the largest body 3.2×10^5 km. As portrayed, large bodies are accountable of merging with a lot of smaller bodies, depicted by small jumps in body radius, but substantial growth is mostly caused by two larger bodies merging, depicted by the larger jumps. Both phenomena have appeared in similar studies where it is hypothesised that larger planetesimals grow faster than small ones [3] [75][80]. The small merges happen more frequent in the beginning, while the most significant large merges are the last few merges that happen [81]. After 10 Myears none of the 10 largest bodies are involved in a collision. This again is analogous to the third stage of the described process, representing a long period where only smaller bodies collide. The shape of the figure matches those in similar studies [82].

The number of particles leaving the system because condition (32) was not satisfied, was monitored in all of the simulations that were run in this research. The only occasions this came about was in cases of extremely large body radii simulations, shown in figure 21. In all other merging events the resulting particle stayed in orbit around the central mass.



(a)

(b)

(c)

(d)

Figure 12: For 5 different runs some important orbital parameters are portrayed. The subfigures show the average and standard deviation of (a) the eccentricity ϵ, (b) the inclination I, (c) the body radius s, (d) the semi-major a, all plotted against a log-scaled time axis. All five runs started with $N = 5.000$ bodies.

4.2. Results of the method

Now the performance of the algorithm will be analysed, going over the number of checks performed by the sweep and prune filter, the size of the collision pair list and the run time of different parts of the algorithm. The different parts of the algorithm that are studied are (1) the creation and sorting of the particle list, (2) the initialisation of the system, including the creation and sorting of the particle list, the creation of the collision pair list and the creation of the time-ranked collision list, (3) the main loop, starting after the initialisation until the end of the simulation, consisting of merging pairs, placing new particle at right place in particle list, calculating new collision pairs with the new particle and calculating the collision times, and repeating this process until the time-ranked collision list is empty, and (4) the entire simulation.

4.2.1. Number of checks and pairs

As described in the method, the algorithm checks during the initialisation whether two particles are a candidate for collision via a process called sweep and prune. It then checks if the MOID of two particles is small enough for a collision to take place, and if so the pair is added to the collision list.

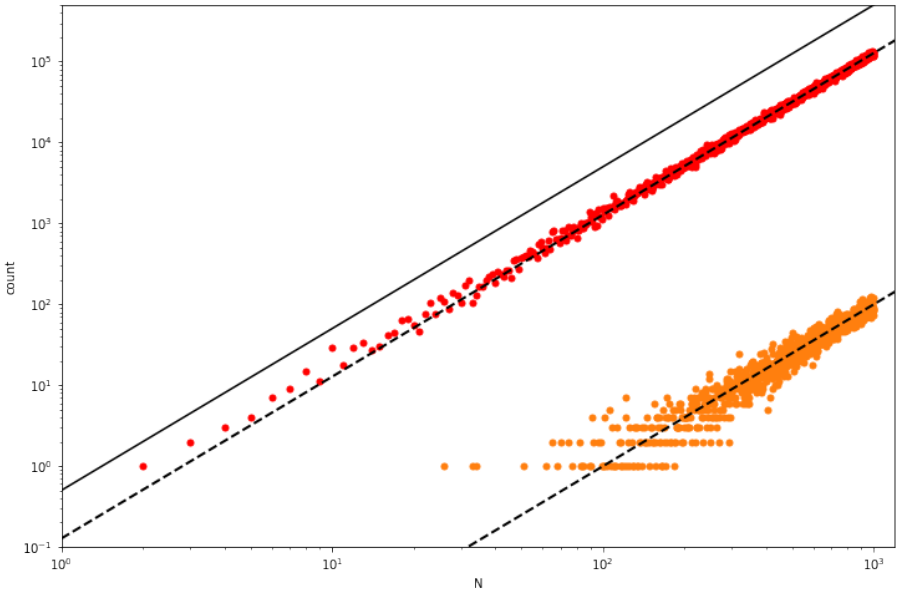


Figure 13: Outcomes of different components of the algorithm plotted for a range of starting particles $N = 1...1000$. The orange dots are the number of pairs in the collision pair list after initialisation, the dotted line through the data represents the fit given by $y = 9.93 \times 10^{-5} \pm 4.8 \times 10^{-7} x^2$. The red dots are the number of apoapsis checks done for the sweep en prune routine. The black dotted line going through the red points is given by $y = 0.1281 \pm 2 \times 10^{-3} x^2$. The solid line is given by $y = 1/2 * x^2$ and is added for comparison, and represents the total number of theoretical pairs.

To understand how the number of checks, as well as how the size of the collision pair list behave and develop under an increasing number of starting bodies N, the model is run for varying N. The number of checks and the size of the pairlist are extracted after the initialisation. The collected data is fitted and the results are shown in figure 13 on a log-log scale. The black solid line is given by $y = 1/2 * x^2$ and represents the total number of theoretical pairs given by, in big *O* notation, $O(N^2/2)$, and is added so that comparisons to the data can be made.

The number of pairs in the collision pair list after the initialisation (orange in figure 13) increases exponentially. The fit used in the figure was $y = ax^2$ and gave a result of $a = 9.93 \times 10^{-5} \pm 4.8 \times 10^{-7}$. The data was also fitted to $y = ax^b$ which gave a result of $a = 9.3 \times 10^{-5} \pm 1.5 \times 10^{-5}$ and $b = 2.010 \pm 2.4 \times 10^{-2}$. This gives enough ground to believe that the size of the collision pair list is proportional to N^2, i.e. *#pairs* $\propto N^2$. Now to understand where this growth rate comes from a number of test were conducted.

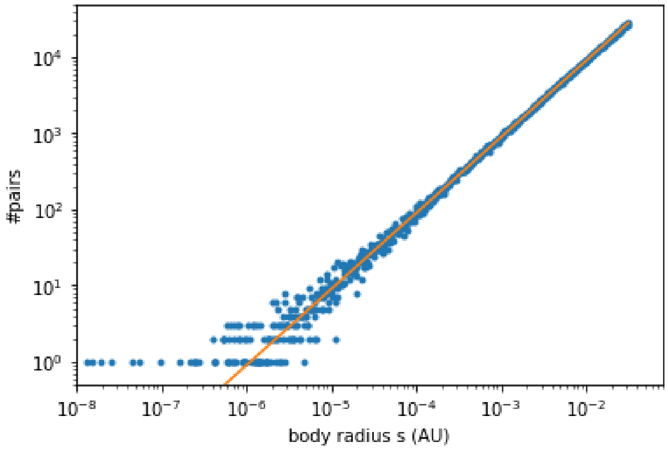


Figure 14: Plotted are the number of pairs in the collision pair list at the end of the initialisation for different values of body radii, shown on a log-log scale. The model is run several times. In each run, the number of starting bodies is $N = 1000$, the distribution of the semi-major axis is $a = 10 \cdot AU \cdot \sqrt[3]{\xi}$ with $\xi = [0, 1]$ uniform, and every particle has the same body radius. The size of this body radius is varied over all runs with the range $s = 10^{\xi_s} AU$ with $\xi_s = [-9, -1.5]$ uniform. The fitted line has function $y = ax^b$ with slope $a = 8.484 \times 10^5 \pm 5.5 \times 10^3$ and power $b = 0.987 \pm 0.002$

To check whether the number of pairs is proportional to the body radius, the model was run several times for varying body radius s. In each run, the number of starting bodies is $N = 1000$, the distribution of the semi-major axis is $a = 10 \cdot AU \cdot \sqrt[3]{\xi}$ with $\xi = [0, 1]$ uniform, and every particle has the same body radius. The size of this body radius varied over all runs with the range $s = 10^{\xi_s} AU$ with $\xi_s = [-9, -1.5]$ uniform. Plotted are the number of pairs in the collision pair list at the end of the initialisation for different values of body radii. The fitted line has function $y = ax^b$ with slope $a = 8.484 \times 10^5 \pm 5.5 \times 10^3$ and power $b = 0.987 \pm 0.002$. The data was fitted once more to $y = ax$ which gave a result of $a = 8.9266 \times 10^5 \pm 7.7 \times 10^2$ By keeping N the same an varying the body radius, there is enough ground to assume *#pairs* $\propto s$

In order to determine if the number of pairs is influenced by the inclination of the orbits in the system, the simulation was run for different values of inclination. In each run, all the orbits of the particles had the same inclination. In figure 15 the number of pairs in the collision pair list at the end of the initialisation is plotted against of different values of inclination. The value of this inclination was changed with each run with the range $I = 10^{\xi_I}$ with $\xi_I = [-5, 0$ uniform. For every run of the model the number of starting bodies is $N = 500$, the distribution of the semi-major axis is equal to $a = \sqrt[3]{(a_+^3 - a_-^3)\xi + a_-^3}$ with $\xi = [0, 1]$ a uniform distribution, as described in section 3.1.1. The body radius is distributed by $s = 10^{\xi_s} AU$ with $\xi_s = [-4, -2.5]$ uniform. Results are shown in figure 15 on a double logarithmic scale. The data exhibits a decreasing slope in the middle of the figure, flattening out on the sides. The orange line is the data fitted between $I = [0.003, 0.1]$ with the function $y = ax^b$, which gave a result of $a = 53 \pm 3$ and $b = -0.73 \pm 0.01$. To grasp the shape of the curvature, a function was manually fitted to the data. The resulting function lead to $y = a/(bx^c + d) + 140$, because as well as for low as for high inclination the curve must flatten. Filling in the values $a = 40$, $b = 2$, $c = 0.896$ and

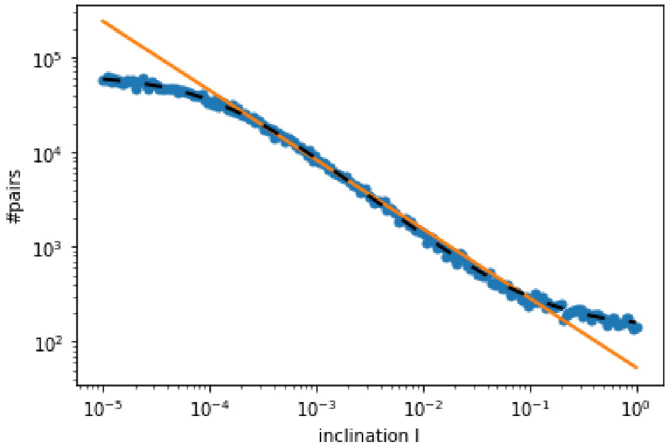


Figure 15: Plotted are the number of pairs in the collision pair list at the end of the initialisation for different values of Inclination, shown on a log-log scale. The model is run several times. In each run, the number of starting bodies is $N = 500$, the distribution of the semi-major axis a is as described in section 3.1.1, the body radius is distributed with $s = 10^{\xi_s}$ with $\xi_s = [-4, -2.5]$ uniform, and every particle has the same inclination I of its orbit. The inclination is varied over all runs with the range $I = 10^{\xi_I}$ with $\xi_I = [-5, 0]$. The orange line is the data fitted between $I = [0.003, 0.1]$ with the function $y = ax^b$, which gave a result of $a = 53 \pm 3$ and $b = -0.73 \pm 0.01$. The dotted line is the function $y = a/(bx^c + d) + 140$, with fitting result $a = 40 \pm 8 \times 10^6$, $b = 2 \pm 4 \times 10^5$, $c = 0.896 \pm 1.8 \times 10^{-2}$ and $d = 6 \times 10^{-4} \pm 1 \times 10^2$.

$d = 6 \times 10^{-4}$ the curve visually follows the data. In the caption of figure 15 the fitted results of this function are written. The uncertainties of which are too high to draw conclusions, except for the power on x, which is $c = 0.896 \pm 1.8 \times 10^{-2}$. Considering this result and that of the orange fit, it is suspected that between bends of the curve the the size of the collision pair list is inversely proportional to the inclination, i.e. *#pairs* $\propto 1/I$.

To research the influence of the eccentricity on the size of the pair list, the model is again run multiple times, this time varying the eccentricity with every run. The number of starting bodies was$N = 500$, body radius was distributed with $s = 10^{\xi_s}$ with $\xi_s = [-4, -2.5]$, and the eccentricity ranged from $\epsilon = 0...0.5$. Results are portrayed in figure 16, where the number of pairs in the collision pair list is plotted against the eccentricity of the system. The data appears to be linear and when a fitting to $y = a + bx^c$ was executed the outcome was $a = 145 \pm 7$, $b = 1107 \pm 27$, $c = 1.025 \pm 0.04$.

The number of apoapsis checks done by the sweep and prune filter to determine whether two orbits overlap, is displayed in red in figure 13. Using $y = ax^b$ for the fitting of the data the parameter resulted in $a = 0.132 \pm 0.005$ and $b = 1.996 \pm 0.006$, already hinting at a at a $y = ax^2$ correlation, which when used for fitting gave $a = 0.1281 \pm 2 \times 10^{-3}$ which is displayed in the black dotted line. From this it can be concluded that the growth factor on $y = ax^2$ is close to 0.128. This is a significant decrease when comparing this sweep and prune method to a method of checking all pairs with brute force, which scales with $O(N^2/2)$. The sweep and prune method only checks $25.6\% \pm 0.2\%$ of all possible pairs.

The peri- to apoapsis interval in 10, where the sweep and prune filter depends on, is constructed semi-major axis a, the eccentricity ϵ and the body radius s of a particle, where the

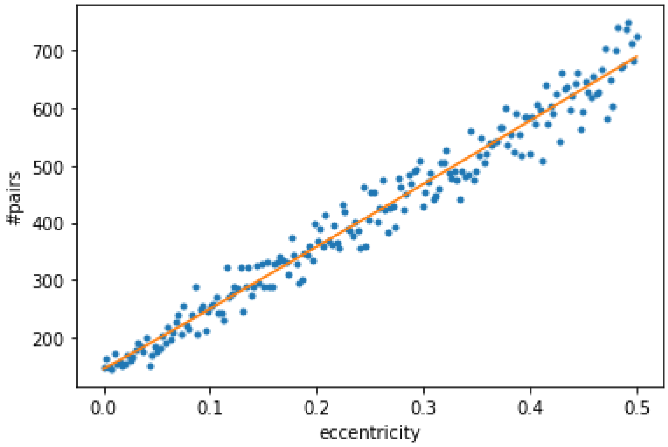


Figure 16: Plotted are the number of pairs in the collision pair list at the end of the initialisation for different values of eccentricity. The model is run several times. In each run, the number of starting bodies is $N = 500$, the distribution of the semi-major axis a is as described in section 3.1.1, the body radius is distributed with $s = 10^{\xi_s}$ with $\xi_s = [-4, -2.5]$ uniform, and every particle has the same eccentricity ϵ for its orbit. The eccentricity is varied over all runs with values ranging from $\epsilon = [0, 0.5]$. The orange line is the data fitted with the function $y = a + bx^c$, which gave a result of $a = 145 \pm 7$, $b = 1107 \pm 27$, $c = 1.025 \pm 0.04$.

eccentricity and body radius make up for the spread of the interval. Since $a >> s$, the influence of the eccentricity ϵ should be significant. To quantize this influence, the model is run for different values of eccentricity, similar to figure 16. The parameters chosen were $N = 3000$, $s = 10^{\xi_s}$ with $\xi_s = [-4, -2.5]$, the distribution of the semi-major axis a and inclination I is as described in section 3.1.1. All particles in the system were set to have equal orbital eccentricity. The eccentricity per run varied with range $\epsilon = 0...0.63$. The results are shown in figure 17. On the left is the data is exhibited for eccentricities up until 0.0025 and on the right for eccentricities for over 0.6. The small eccentricity $\epsilon < 0.1$, the number of checks appears to be linearly proportional to the eccentricity. For larger eccentricities, the curve flattens towards the maximum possible pairs $N^2/2$, implying that the filter is no longer beneficial. The data was fitted to the function $y = a + bx^c$, which gave a result of $a = 5954 \pm 47$, $b = 2.667 \times 10^6 \pm 6.2 \times 10^4$, and $c = 0.986 \pm 5 \times 10^{-3}$. For an eccentricity of $\epsilon = 0.05$ the data in the figure indicates to have done a little bit over 1×10^6 checks, which is about a quarter of the maximum possible pairs. This resonates with what is mentioned above about figure 16, where the average eccentricity was $\langle \epsilon \rangle = 0.05$ and gave rise to a checking efficiency of $25.6\% \pm 0.2\%$.

4.2.2. Run time

To obtain a better judgement of how the computation time is influenced by the number of bodies, the model was run multiple times for different number of starting particles. The results are showcased in figure 18. To understand which components of the algorithm are responsible for the total running time, various components and their share of the computation time are displayed.

In figure 18 the exponential growth of the computation time can be observed. The creation and the sorting of the particle list takes very little to almost none of the total running time. The run time of the main loop has a very short computation time for for small N, but growth

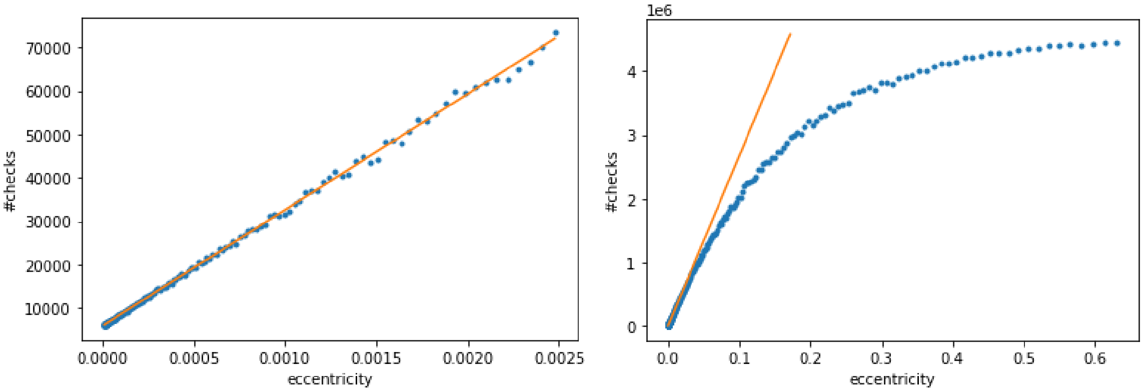


Figure 17: Plotted are the number of checks done by the sweep and prune filter for different values of eccentricity. Left are the eccentricities $\epsilon = 0...0.0025$, and right is zoomed out with eccentricities $\epsilon = 0...0.6$. The model is run several times. In each run, the number of starting bodies is $N = 3000$, the distribution of the semi-major axis a and inclination I is as described in section 3.1.1, the body radius is distributed with $s = 10^{\xi_s}$ with $\xi_s = [-4, -2.5]$ uniform, and every particle has the same eccentricity ϵ for its orbit. The eccentricity is varied over all runs with values ranging from $\epsilon = [0, 0.6]$. The orange line is the data fitted with the function $y = a + bx^c$, which gave a result of $a = 5954 \pm 47$, $b = 2.667 \times 10^6 \pm 6.2 \times 10^4$, and $c = 0.986 \pm 5 \times 10^{-3}$.

substantially to about one third of the total run time. Most of the total run time is caused by the initialisation of the model. In the first half being responsible for almost all- but then by $N = 1.000$ still about two thirds of the total computation time.

To comprehend the curvature of the growth of the computation times, the data has been fitted and the results have been displayed in figure 19. All the data was fitted the the function $y = ax^b$ to find the power by which the run time increases by N. the The creation and sorting of the particle list was fitted this function with the resulting values $a = 5.919 \times 10^{-5} \pm 9.0 \times 10^{-7}$ and $b = 1.0633 \pm 2.2 \times 10^{-3}$. These results hint at a linear increase of the run time for the creation and sorting of the particle list *#pairs* $\propto N$, however $b > 1$. The the creation of the particle list has a time complexity of $O(N)$ and the algorithm uses *Quick Sort* for sorting by apoapsis which has a time complexity of $O(N \log(N))$ [83]. This last term is likely the reason why $b > 1$. The function $y = ax\log(x) + bx$ was also fitted and gave a result of $a = 1.272 \times 10^{-5} \pm 4.4 \times 10^{-8}$ and $b = 5.34 \times 10^{-5} \pm 1.2 \times 10^{-6}$.

The results of the $y = ax^b$ fit of the initialisation gave back $a = 4.22 \times 10^{-5} \pm 2.0 \times 10^{-6}$ and $b = 1.980 \pm 7.2 \times 10^{-3}$, hinting at a relation *#pairs* $\propto N^2$. Hypothesised is that the initialisation of the system would increase with $y = \alpha N^2 \log(\alpha N^2)$ [5], with $alpha = s/a$, the average body radius over the average semi-major axis. When this function was fitted, the resulting value was $\alpha = 2.85 \times 10^{-5} \pm 1 \times 10^{-7}$. When dividing the averages the outcome was $\langle s \rangle / \langle a \rangle = 1.57 \times 10^{-5}$ having the same order of magnitude.

The function $y = ax^b$ was also fitted to the main loop and the total run time of the entire algorithm in figure 19. The outcome of the fit to the main loop was $a = 7.6 \times 10^{-8} \pm 1.6 \times 10^{-8}$ and $b = 2.792 \pm 3.1 \times 10^{-2}$, and for the entire algorithm the outcome was $a = 1.372 \times 10^{-5} \pm 9.4 \times 10^{-7}$ and $b = 2.199 \pm 1.0 \times 10^{-2}$. To further research the growth of the computation time, the model was run for even larger N. Results are shown in figure 20. Here the fit gave back values of $a = 3.4 \times 10^{-9} \pm 2.3 \times 10^{-9}$, $b = 2.834 \pm 7.6 \times 10^{-2}$, suggesting a faster increase

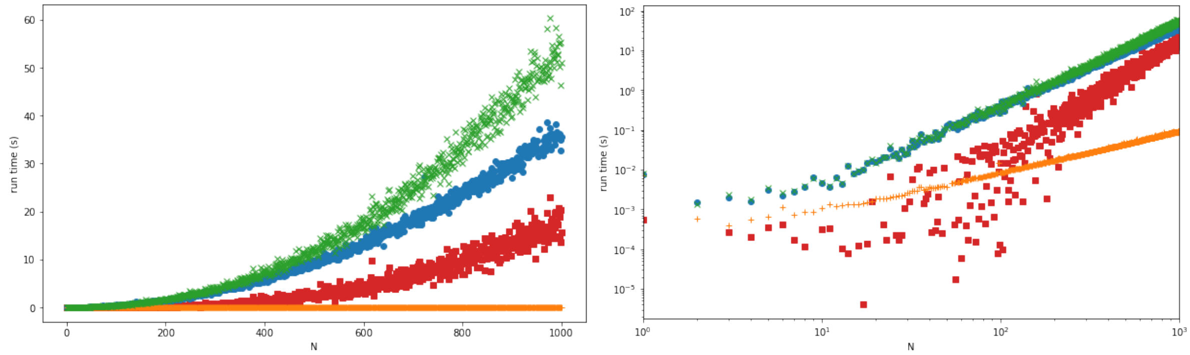


Figure 18: The computation time of different aspects of the algorithm are plotted against the number of starting particles. The total computation time until the end of the simulation is plotted in green x. The run time of the initialisation is plotted in blue ○. the run time of the main loop is plotted in red □. The run time of the creation and sorting of the particle list is plotted in orange +. On the left the exponential growth of the run time can be seen. On the right the same data is plotted on a double logarithmic scale.

in computation time per N than the fit in figure 19. The data for larger N was also fitted to $y = ax^2 + bx^4$, giving a result of $a = 3.30 \times 10^{-6} \pm 2.7 \times 10^{-7}$, $b = 4.15 \times 10^{-14} \pm 3.4 \times 10^{-15}$. It is plausible that the total computation time actually increases by a time complexity of $O)(N^4)$, but because of its small growth factor is only a manifest addition for very large number of starting particles $N > 10.000$.

For the purpose of understanding how the total run time is dependant of the body radius of particles in the system, the model was run for varying s. The results are visualized in figure 21. The total computation time is plotted against the body radius. The number of starting bodies was $N = 2000$ and the eccentricity was distributed by $\epsilon = [0, 10^{-9}]$ uniform. The body radius of every particle in the system was set to be equal, and varied with every run of the model. The The blue line is the data fitted with the function $y = a + bx^c$, which gave a result of $a = -1.0 \pm 0.4$, $b = 7.2 \times 10^{-3} \pm 1.8 \times 10^{-3}$, $c = 0.905 \pm 0.026$. Here a could be the small offset of the computation time when there are little to no collisions, and c being the power of which the computation time scales with s. Considering that the runtime was proportional to $N^2 = 4.000.000$, the impact the body radius on the total computation time could be regarded as significant. The curve of the data eventually flattens around 400 seconds, where almost all bodies have collided with each other. This gives evidence for a maximal computation time of the Keplerian method of N-body system simulations.

In addition, the number of particles leaving the system is also shown in figure 21. This is the result of condition (32) not being satisfied after the merging of two particles. All cases are the result of a collision with the central mass.

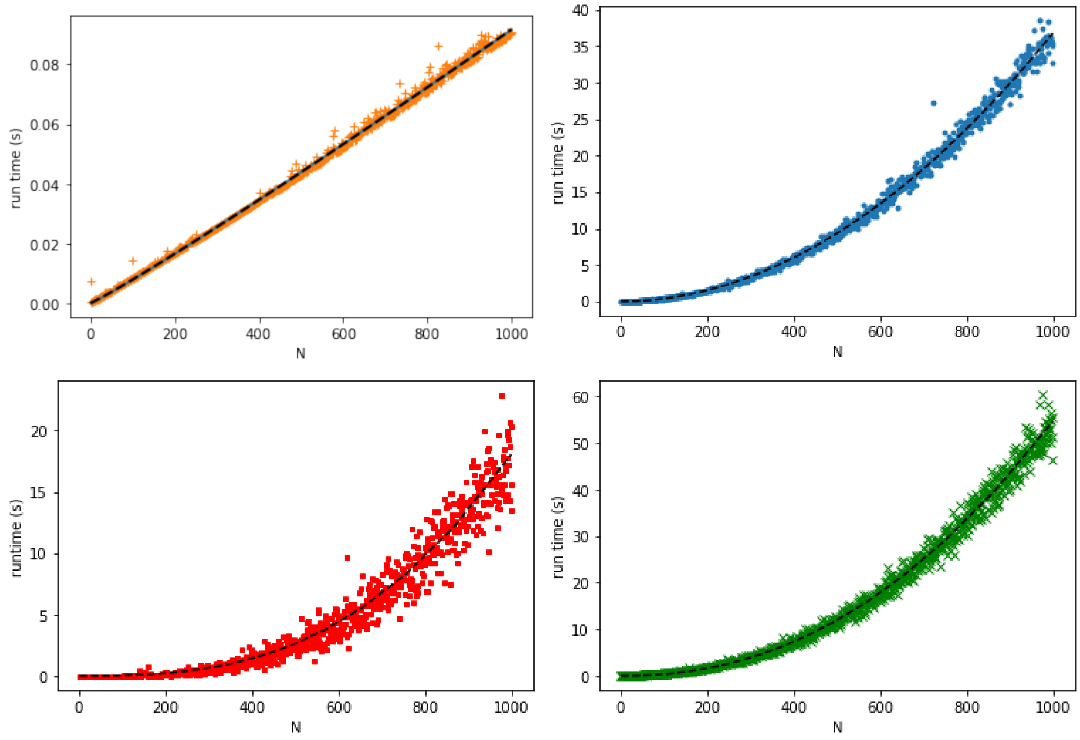


Figure 19: Run time is plotted against the number of starting particles. All data is fitted with the function $y = ax^b$ in this figure. Shown is the data of the run time of (orange) the creation and sorting of the particle list with fit result $a = 5.919 \times 10^{-5} \pm 9.0 \times 10^{-7}$ and $b = 1.0633 \pm 2.2 \times 10^{-3}$, (blue) the initialisation, consisting of the creation and sorting of the particle list, and the creation and sorting the time-ranked collision list with fit result $a = 4.22 \times 10^{-5} \pm 2.0 \times 10^{-6}$ and $b = 1.980 \pm 7.2 \times 10^{-3}$, (red) the main loop, consisting of merging pairs, placing new particle at right place in particle list, calculating new collision pairs with the new particle and calculating the collision times, and repeating this process until the time-ranked collision list is empty, with a resulting fit of $a = 7.6 \times 10^{-8} \pm 1.6 \times 10^{-8}$ and $b = 2.792 \pm 3.1 \times 10^{-2}$, and (green) the computation time for the entire algorithm with a resulting fit of $a = 1.378 \times 10^{-5} \pm 9.4 \times 10^{-7}$ and $b = 2.199 \pm 1.0 \times 10^{-2}$. Parameters are distributed as described in section 3.1.1, with averages $\langle \epsilon \rangle \approx 0.05$, $\langle I \rangle \approx 0.067$, $\langle s \rangle \approx 16000$km, and $\langle a \rangle \approx 7.51$AU.

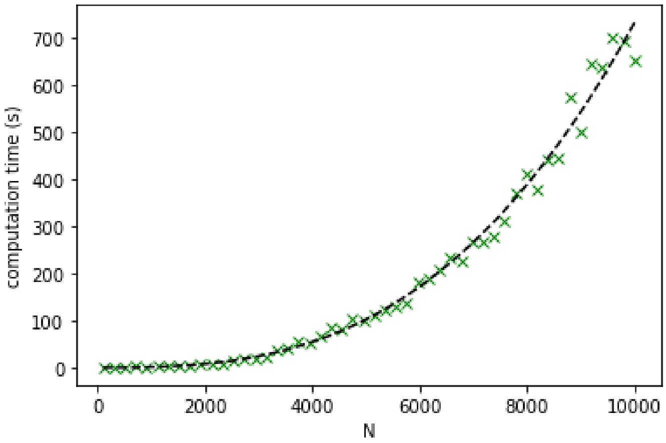


Figure 20: Plotted is the total computation time versus the number of starting particles N. Eccentricity was $\epsilon = 0$ and body radius was distributed by $s = 10^{\xi_s}$AU with $\xi_s = [-8, -4]$ a uniform distribution. The dotted line fitted function $y = ax^b$ with values $a = 3.4 \times 10^{-9} \pm 2.3 \times 10^{-9}$, $b = 2.834 \pm 7.6 \times 10^{-2}$.

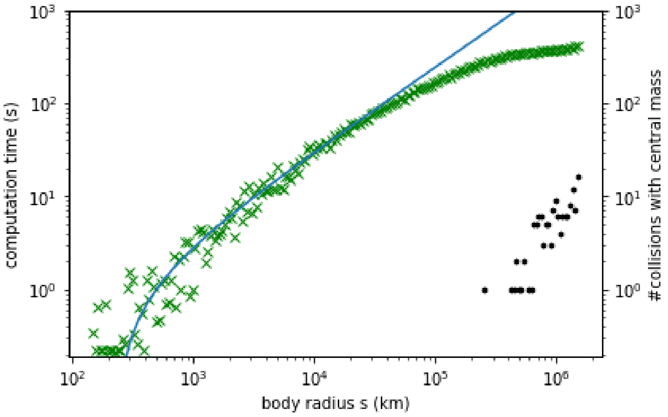


Figure 21: Plotted in green is the total computation time for one run of the algorithm against the body radius of the particles in the system, in blue its fitted line, and in black the number of collisions with the central mass. The model is run several times. In each run, the number of starting bodies is $N = 2000$, the distribution of the semi-major axis a and the inclination I are as described in section 3.1.1, the eccentricity is distributed by $\epsilon = [0, 10^{-9}]$ uniform, and every particle in the system has an equal body radius. The body radius is varied with each run by $s = 10^{-6}...10^{-2}$. The blue line is the data fitted with the function $y = a + bx^c$, which gave a result of $a = -1.0 \pm 0.4$, $b = 7.2 \times 10^{-3} \pm 1.8 \times 10^{-3}$, $c = 0.905 \pm 0.026$.

5. Discussion

In this section the results and foremost limitations and imperfections of the Keplerian model studied in this research will be discussed, accompanied by recommendations for future research in this field of study.

The results could have been affected by errors induced by the computer program. Small uncertainties in machine precision could accumulate to become large errors in the produced results. Also, the coded computer program may contain unforeseen errors that influence the calculations. However, the results produced by the code were analysed and showed to be reliable and consistent outcomes.

The number of pairs as well as the total computation time showed to be proportional to N^2 or even higher orders, which induces a less efficient method than the octree code with a time complexity of $O(N^{4/3}\log(N))$. However, this method proved to be dependent of eccentricity, body radius, and inclination, in which this method differentiates itself from other methods. Because the body radius for planetesimals is very small compared to their semi major axis, the run time stays remarkably low for $N < 2000$. The influence of these parameters for higher orders of N is hypothesised in [5] and could results in a preference for the Keplerian method. Future research could include these higher orders of starting particles.

In the model it was chosen to uniformly distribute the bodies throughout the disk. But in studies it it hypothesised that non-uniformity is necessary to create planetesimals [47]. The body radius of the particles in the system was independently distributed described in section 3.1.1, but is observed to be inversely proportional to the particle's semi major axis [84]. The eccentricity and the semi-major axis are also distributed independently of each other, while observed celestial systems show correlations [85]. The simulations in this research were run with a relatively small number of particles $N \leq 10.000$, which is considered very small considering there are at least 1 million observed asteroids in the solar system [86]. Increasing the number of starting particles would offer this research a more accurate description of the formation of planetary system via Keplerian models. The tests on the computation time would also be more interesting with a higher N.

Every collision in the simulation was assumed to be inelastic, so calculations for in a simple merger were used. This however does not portray reality for bodies with radii $s > 10^3$km. Oblique collisions transfer orbital angular momentum into planetary spin, combined with the fact that fragmentation occurs, indicate that the simple mergers method for these collisions is not desirable [68]. A recommendation for future research would be to add a spin parameter to the list of parameters for each particle. Other consequences of collision could be one or a combination of bouncing, erosion, fragmentation, or mass transfer [87].

The model did not incorporate mutual gravitational interaction between the particles, which in fact produce large effects on the trajectory of the particles. For further research it may be recommended to include gravitational scattering of nearby passages. These could be modelled as elastic collisions. Other influential disk interaction that were not included in the model include but are not limited to disk turbulence [88], radiative transfer [89], gravitational perturbation [90], magnetic disk wind [91], planetary migration and eccentric bending [92].

The orbits of the particles are also assumed to be stationary, i.e. the argument of periapsis ϖ is chosen to be constant. In reality the orbits of celestial bodies are found to be non-stationary, and actually rotate by a phenomenon called apsidal precession [6]. This precession is caused by a combination of different factors [44], all of which are not monitored in this research's algorithm. Although the precession rate of planets in our solar system are of the order of magnitude of $\sim 10^5$ years, it is occasionally observed to be a small number of days in rare cases of exoplanets [45]. In addition, the formation of planetary systems takes multiple Myears to develop. In future research this precession could be included as a particle parameter, but because of its dependence on fluctuating external factors this would be shorthand solution. The neglection of the apsidal precession may be considered as the Achilles' heel of the Keplerian model.

6. Conclusion

The aim of this research was to create an effective collision detection model for a simplified protoplanetary disk system using Kepler systems and analyse the results of the model to check the reliability of this simulation method, i.e. if the quality of the results are trustworthy, performing consistently well, and can be depended on to be accurate. In addition, the method had to be computationally efficient, having to reach time complexities below that of other methods in similar studies i.e. tree code, which has a time complexity of $O(N^{4/3}\log(N))$.

To keep track of the every body in the system is computationally exhaustive as the number of particles increase. For an N-body system this would result in simulating N^2 interactions. In Keplerian models simplifies N-body systems by describing them in a few orbital parameters and neglecting the small effects of interparticle interaction and timestep determination. This method is hypothesised to have a total computation time of $N^2\epsilon + (N^4 s^3)/(Ia^3)$ [5]. Where the power of N is greater than in the tree code, but because of it's body radius s dependency it could be considered as the faster method.

It was preferred to keep the model simple, so the algorithm could be better tested, rather than to go for a more realistic model. Various realizations of N-body systems were modelled using Kepler orbits and their orbital elements. In the initialisation the parameters were picked from distribution to form a triangular torus disk with a uniform particle distribution. For the pair list initialisation, the peri- and apoapsis of each particle were checked to perform a sweep and prune filter, where after all possible collision pairs are calculated using the minimum orbit intersection distance, and lastly determining the collision time for each collision and sorting them in a time-ranked collision list. In the main loop of the algorithm takes the top two particles from the collision list, merges them in the particle list, creating a new body in the particle list and updates the collision list. Various parameters are stored during the simulation.

The model showed behaviour of that of other simulation studies, as well as observational data. Worthy of note are the shape of the decreasing total number of particles in the system over time [70], the damping effect of the eccentricity and the inclination [72], the graduate shape transformation of the eccentricity distribution to that of an asteroid belt [74], the transition of the orbits in the model to move towards a critical eccentricity of 0.02 [55][73], the preservation of shape of the body radius distribution of the system [75][76], the decrease in average semi-major axis over large timescales [77][78], the number of small bodies colliding with only a few large bodies [75][80], and the collisions between two large bodies responsible for the growth of the largest bodies [81].

It found that the model encountered three stages of collisions. The first stage being one where collisions happen rather uniformly throughout the disk, regardless of body radius or semi-major axis. The averages of these two parameters stay about the same value during this stage, only the average semi-major increases a little, implying a little more collisions with bodies with smaller orbits. The second stage is dominated by bodies that already collided before and hence have larger radii, giving them higher probabilities of colliding again. The third and last stage is characterised by bodies with large semi-major axes and small body radius. In this last stage the average body radius of the system increases significantly because of these small body collisions. The first stage takes place during the first 1 Myears of the simulation, the second stage between 1 Myears up until 10 Myears, and after 10 Myears the third stage starts.

The decomposition of the performance of the algorithm gave the following results: The number of checks performed by the sweep and prune filter as well as the number of pairs in the collision pair list after initialisation is exponentially proportional to the number of starting bodies of the system, so *#checks* $\propto N^2$ and *#pairs* $\propto N^2$. For a system with $N = 3.000$, the number of checks performed by the sweep and prune filter was proportional to the eccentricity of the system for small eccentricities $\epsilon < 0.1$ and when fitted showed a result of *#checks* $\propto \epsilon$. For larger eccentricities the number of checks flattens towards the total possible pairs $N^2/2$. The total number of pairs in the collision pair list created in the initialisation of the system appeared to be proportional to the body radius s of the system as well as the eccentricity ϵ and inversely proportional to I, so *#pairs* $\propto s$, *#pairs* $\propto 1/I$, and *#pairs* $\propto \epsilon$.

Finally , the computation time of different parts of the algorithm plotted against the number of starting particles $N = 1...1.000$, and the results were analysed. All data is fitted with the function $y = ax^b$. It was found that the run time of the creation and sorting of the particle list showed to be proportional to N, the initialisation, consisting of the creation and sorting of the particle list, and the creation and sorting the time-ranked collision list to be proportional to N^2, the main loop, consisting of merging pairs, placing new particle at right place in particle list, calculating new collision pairs with the new particle and calculating the collision times, and repeating this process until the time-ranked collision list is empty to be roughly proportional to $N^{2.8}$, and the computation time for the entire algorithm to be also roughly $N^{2.8}$. The last of these results imply a proportionality of a higher power.

The relationship between body radius and run time of the entire algorithm was analysed while the eccentricity was kept small ($\epsilon < 10^{-9}$) and $N = 2.000$. The data was fitted to $y = a+bx^c$ and a gave result of $a = -1.0\pm0.4$, $b = 7.2\times10^{-3}\pm1.8\times10^{-3}$, $c = 0.905\pm0.026$. So for $N < 10.000$, the computation time is therefore linearly proportional to the body radius, *runtime* $\propto s$ and could be considered as a significant influence to the run time.

To conclude this research, the Keplerian model could be used to model N-body systems for collision detection because of its behavioural resemblance to obserable astrophysical systems. The computation time of the Keplerian system scales with a higher order per N than for instance the three code algorithm, but has showed to be influenced by orbital parameters in which this method differentiates itself from other simulation methods, with a remarkable influence on the total computation time by the body radius s of the system, as well as the eccentricity ϵ and inclination I. More research could be done for higher orders of N to better mimic reality as well as providing more insight on the proportionality of the computation time. Recommendations to the algorithm consist of adding planetary spin as a particle parameter, and methods to incorporate apsidal precession in a competent manner.

Bibliography

[1] Viktor Sergeevich Safronov. Evolution of the protoplanetary cloud and formation of the earth and the planets. 1972.

[2] WR Ward. Planetary accretion. *Completing the Inventory of the solar system*, 107:337–361, 1996.

[3] Eiichiro Kokubo, Junko Kominami, and Shigeru Ida. Formation of terrestrial planets from protoplanets. i. statistics of basic dynamical properties. *The Astrophysical Journal*, 642(2):1131, 2006.

[4] Donald Meagher. Geometric modeling using octree encoding. *Computer graphics and image processing*, 19(2):129–147, 1982.

[5] Visser P M. Collision detection for n-body kepler systems. *Astronomy & Astrophysics*, submitted 2022.

[6] Carl D Murray and Stanley F Dermott. *Solar system dynamics*. Cambridge university press, 1999.

[7] Wikipedia contributors. Orbital elements, 2021. [Online; accessed 14-June-2022].

[8] David A Vallado. *Fundamentals of astrodynamics and applications*, volume 12. Springer Science & Business Media, 2001.

[9] H Stroh. Swedenborg's position in the history of philosophy. In *Atti del IV Congresso Internazionale di Filosofia*, volume 2, pages 290–296, 1911.

[10] Emanuel Swedenborg. *Emanuelis Swedenborgii Opera philisophica et mineralia*, volume 1. 1734.

[11] Immanuel Kant. *Allgemeine Naturgeschichte und Theorie des Himmels, nach Newtonischen Grundsätzen abgehandelt*. 1755.

[12] T Birnstiel, CP Dullemond, and F Brauer. Gas-and dust evolution in protoplanetary disks. *Astronomy & Astrophysics*, 513:A79, 2010.

[13] Shrinivas R Kulkarni and J Jeff Hester. Discovery of a nebula around psr1957+ 20. *Nature*, 335(6193):801–803, 1988.

[14] CL Brogan, LM Pérez, TR Hunter, WRF Dent, AS Hales, RE Hills, S Corder, EB Fomalont, C Vlahakis, Y Asaki, et al. The 2014 alma long baseline campaign: first results from high angular resolution observations toward the hl tau region. *The Astrophysical journal letters*, 808(1):L3, 2015.

[15] Jane S Greaves. Disks around stars and the growth of planetary systems. *Science*, 307(5706):68–71, 2005.

[16] Thierry Montmerle, Jean-Charles Augereau, Marc Chaussidon, Mathieu Gounelle, Bernard Marty, and Alessandro Morbidelli. 3. solar system formation and early evolution: the first 100 million years. *Earth, Moon, and Planets*, 98(1):39–95, 2006.

[17] Jonathan P Williams and Lucas A Cieza. Protoplanetary disks and their evolution. *arXiv preprint arXiv:1103.0556*, 2011.

[18] Lee Hartmann. *Accretion processes in star formation*, volume 32. Cambridge University Press, 2000.

[19] Philip J Armitage. Dynamics of protoplanetary disks. *arXiv preprint arXiv:1011.1496*, 2010.

[20] Leonardo Testi, Tilman Birnstiel, Luca Ricci, Sean M. Andrews, Juergen Blum, John M. Carpenter, Carsten Dominik, Andrea Isella, Antonella Natta, Jonathan P. Williams, and David J. Wilner. Dust evolution in protoplanetary disks. *arXiv: Solar and Stellar Astrophysics*, pages 2205–2220, 2018.

[21] Sanson Tsun Sum Poon. *Formation and dynamical evolution of planetary systems*. PhD thesis, Queen Mary University of London, 2021.

[22] Christer Ericson. *Real-time collision detection*. Crc Press, 2004.

[23] Isaac Newton and David Gregory. *A NEW MOST ACCURATE THEORY OF THE MOON'S MOTION;...: And Publ. in Latin by David Gregory*. Baldwin, 1702.

[24] Henri Poincaré. *Les méthodes nouvelles de la mécanique céleste*, volume 3. Gauthier-Villars, 1899.

[25] Gustavo Gargioni, David Alexandre, Marco Peterson, and Kevin Schroeder. Multiple asteroid retrieval mission from lunar orbital platform-gateway using reusable spacecrafts. pages 1–13, 03 2019.

[26] Ming Lin and Stefan Gottschalk. Collision detection between geometric models: A survey. In *Proc. of IMA conference on mathematics of surfaces*, volume 1, page 31. Citeseer, 1998.

[27] Yann Alibert, Frédéric Carron, Andrea Fortier, Samuel Pfyffer, Willy Benz, Christoph Mordasini, and David Swoboda. Theoretical models of planetary system formation: mass vs. semi-major axis. *Astronomy & Astrophysics*, 558:A109, 2013.

[28] Michiel Lambrechts, Alessandro Morbidelli, Seth A Jacobson, Anders Johansen, Bertram Bitsch, Andre Izidoro, and Sean N Raymond. Formation of planetary systems by pebble accretion and migration-how the radial pebble flux determines a terrestrial-planet or super-earth growth mode. *Astronomy & Astrophysics*, 627:A83, 2019.

[29] William K Hartmann and Donald R Davis. Satellite-sized planetesimals and lunar origin. *Icarus*, 24(4):504–515, 1975.

[30] Walter Dehnen and Justin I Read. N-body simulations of gravitational dynamics. *The European Physical Journal Plus*, 126(5):1–28, 2011.

[31] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[32] Josh Barnes and Piet Hut. A hierarchical o (n log n) force-calculation algorithm. *nature*, 324(6096):446–449, 1986.

[33] Takuya Ohtani and Toru Tsuribe. Growth of a protostar and a young circumstellar disk with a high mass-accretion rate onto the disk. *Publications of the Astronomical Society of Japan*, 65(4), 2013.

[34] W Kapferer, C Sluka, S Schindler, C Ferrari, and B Ziegler. The effect of ram pressure on the star formation, mass distribution and morphology of galaxies. *Astronomy & Astrophysics*, 499(1):87–102, 2009.

[35] Johannes Kepler. Astronomia nova..., seu physica coelestis, tradita commentariis de motibus stellae martis. *Astronomia nova*.

[36] Johannes Kepler. Harmonices mundi (frankfurt, 1619). *Book II (English translation by EJ Aiton, AM Duncan, and JV Field, American Philosophical Society, 1997).*

[37] Lasunncty at the English Wikipedia. Diagram illustrating and explaining various terms in relation to orbits of celestial bodies., 2007. [Online; accessed 14-June-2022].

[38] Marc Murison. A practical method for solving the kepler equation. 11 2006.

[39] Herbert Goldstein, Ch Poole, and J Safko. *Classical mechanics addison-wesley*, volume 426. 1980.

[40] Albert Einstein. The field equations of gravitation. *Sitzungsber. Preuss. Akad. Wiss. Berlin (Math. Phys.)*, 1915:844–847, 1915.

[41] Theodore Eugene Sterne. Apsidal motion in binary stars. *Monthly Notices of the Royal Astronomical Society*, 99:451–462, 1939.

[42] Yoshihide Kozai. Secular perturbations of asteroids with high inclination and eccentricity. *The Astronomical Journal*, 67:591–598, 1962.

[43] Jordi Miralda-Escudé. Orbital perturbations of transiting planets: a possible method to measure stellar quadrupoles and to detect earth-mass planets. *The Astrophysical Journal*, 564(2):1019, 2002.

[44] David M Kipping. *The transits of extrasolar planets with moons*. Springer Science & Business Media, 2011.

[45] Darin Ragozzine and Aaron S Wolf. Probing the interiors of very hot jupiters using transit light curves. *The Astrophysical Journal*, 698(2):1778, 2009.

[46] C Thalmann. Imaging of a transitional disk gap in reflected light thalmann, c.; grady, ca; goto, m.; wisniewski, jp; janson, m.; henning, t.; honda, m.; mulders, gd; min, m.; fukagawa, m.

[47] Taichi Uyama. *Direct Imaging and Characterizations of.* PhD thesis, School of Science, The University of Tokyo, 2018.

[48] Eric E Mamajek. Initial conditions of planet formation: lifetimes of primordial disks. In *AIP Conference Proceedings*, volume 1158, pages 3–10. American Institute of Physics, 2009.

[49] James E Pringle. Accretion discs in astrophysics. *Annual review of astronomy and astrophysics*, 19:137–162, 1981.

[50] Anusha Kalyaan, Steven J Desch, and Nikhil Monga. External photoevaporation of the solar nebula. ii. effects on disk structure and evolution with non-uniform turbulent viscosity due to the magnetorotational instability. *The Astrophysical Journal*, 815(2):112, 2015.

[51] Lee Hartmann, Nuria Calvet, Erik Gullbring, and Paola D'Alessio. Accretion and the evolution of t tauri disks. *The Astrophysical Journal*, 495(1):385, 1998.

[52] Andrea Isella, Leonardo Testi, and Antonella Natta. Large dust grains in the inner region of circumstellar disks. *Astronomy & Astrophysics*, 451(3):951–959, 2006.

[53] CP Dullemond and JD Monnier. The inner regions of protoplanetary disks. *arXiv preprint arXiv:1006.3485*, 2010.

[54] Natalia Dzyurkevich, Neal J Turner, Thomas Henning, and Wilhelm Kley. Magnetized accretion and dead zones in protostellar disks. *The Astrophysical Journal*, 765(2):114, 2013.

[55] Henrik Eklund and Frédéric S Masset. Evolution of eccentricity and inclination of hot protoplanets embedded in radiative discs. *Monthly Notices of the Royal Astronomical Society*, 469(1):206–217, 2017.

[56] Bertram Bitsch, Trifon Trifonov, and Andre Izidoro. The eccentricity distribution of giant planets and their relation to super-earths in the pebble accretion scenario. *Astronomy & Astrophysics*, 643:A66, 2020.

[57] Alibert, Y., Carron, F., Fortier, A., Pfyffer, S., Benz, W., Mordasini, C., and Swoboda, D. Theoretical models of planetary system formation: mass vs. semi-major axis. *A&A*, 558:A109, 2013.

[58] Felix R Hoots, Linda L Crawford, and Ronald L Roehrich. An analytic method to determine future close approaches between satellites. *Celestial mechanics*, 33(2):143–158, 1984.

[59] Ernst Julius Öpik. Collision probabilities with the planets and the distribution of interplanetary matter. In *Proceedings of the Royal Irish Academy. Section A: Mathematical and Physical Sciences*, volume 54, pages 165–199. JSTOR, 1951.

[60] Jonathan D Cohen, Ming C Lin, Dinesh Manocha, and Madhav Ponamgi. I-collide: An interactive and exact collision detection system for large-scale environments. In *Proceedings of the 1995 symposium on Interactive 3D graphics*, pages 189–ff, 1995.

[61] Andrea Milani and Giovanni Gronchi. *Theory of orbit determination*. Cambridge University Press, 2010.

[62] SP Manley, F Migliorini, and ME Bailey. An algorithm for determining collision probabilities between small solar system bodies. *Astronomy and Astrophysics Supplement Series*, 133(3):437–444, 1998.

[63] Hiroshi Kinoshita and Hiroshi Nakai. Motions of the perihelions of neptune and pluto. *Celestial Mechanics*, 34(1):203–217, 1984.

[64] Andrew M Rockett et al. *Continued fractions*. World Scientific, 1992.

[65] Murray Bremner. *Lattice basis reduction*. CRC Press New York, 2011.

[66] Natasa Sirotic and Andrina Zazkis. Irrational numbers: The gap between formal and intuitive knowledge. *Educational Studies in Mathematics*, 65(1):49–76, 2007.

[67] Aron Shouten. Collision detection using continued fractions, 2022.

[68] Craig Agnor and Erik Asphaug. Accretion efficiency during planetary collisions. *The Astrophysical Journal*, 613(2):L157, 2004.

[69] Paul Cresswell and Richard P Nelson. On the evolution of multiple protoplanets embedded in a protostellar disc. *Astronomy & Astrophysics*, 450(2):833–853, 2006.

[70] John CB Papaloizou and John D Larwood. On the orbital evolution and growth of protoplanets embedded in a gaseous disc. *Monthly Notices of the Royal Astronomical Society*, 315(4):823–833, 2000.

[71] Huan YA Meng, Kate YL Su, George H Rieke, Wiphu Rujopakarn, Gordon Myers, Michael Cook, Emery Erdelyi, Chris Maloney, James McMath, Gerald Persha, et al. Planetary collisions outside the solar system: Time domain characterization of extreme debris disks. *The Astrophysical Journal*, 805(1):77, 2015.

[72] M Nagasawa, DNC Lin, and E Thommes. Dynamical shake-up of planetary systems. i. embryo trapping and induced collisions by the sweeping secular resonance and embryo-disk tidal interaction. *The Astrophysical Journal*, 635(1):578, 2005.

[73] Bertram Bitsch and Wilhelm Kley. Orbital evolution of eccentric planets in radiative discs. *Astronomy & Astrophysics*, 523:A30, 2010.

[74] Sean N Raymond and David Nesvorny. Origin and dynamical evolution of the asteroid belt. *arXiv preprint arXiv:2012.07932*, 2020.

[75] Anders Johansen, Mordecai-Mark Mac Low, Pedro Lacerda, and Martin Bizzarro. Growth of asteroids, planetary embryos, and kuiper belt objects by chondrule accretion. *Science Advances*, 1(3):e1500109, 2015.

[76] Stephen J Kortenkamp, George W Wetherill, and Satoshi Inaba. Runaway growth of planetary embryos facilitated by massive bodies in a protoplanetary disk. *Science*, 293(5532):1127–1129, 2001.

[77] William F Bottke Jr, Daniel D Durda, David Nesvornỳ, Robert Jedicke, Alessandro Morbidelli, David Vokrouhlickỳ, and Harold F Levison. Linking the collisional history of the main asteroid belt to its dynamical excitation and depletion. *Icarus*, 179(1):63–94, 2005.

[78] Matthew S Clement, Sean N Raymond, and Nathan A Kaib. Excitation and depletion of the asteroid belt in the early instability scenario. *The Astronomical Journal*, 157(1):38, 2019.

[79] S Ida, DNC Lin, and M Nagasawa. Toward a deterministic model of planetary formation. vii. eccentricity distribution of gas giants. *The Astrophysical Journal*, 775(1):42, 2013.

[80] Alessandro Morbidelli, William F Bottke, David Nesvornỳ, and Harold F Levison. Asteroids were born big. *Icarus*, 204(2):558–573, 2009.

[81] Adrián Brunini and Omar G Benvenuto. On oligarchic growth of planets in protoplanetary disks. *Icarus*, 194(2):800–810, 2008.

[82] Zoë M Leinhardt and Derek C Richardson. Planetesimals to protoplanets. i. effect of fragmentation on terrestrial planet formation. *The Astrophysical Journal*, 625(1):427, 2005.

[83] Diwakar Gupta NumPy. https://numpy.org/doc/stable/reference/generated/numpy.sort.html. consulted 11 augustus 2022.

[84] Guillaume Laibe, J-F Gonzalez, Laure Fouchet, and Sarah T Maddison. Sph simulations of grain growth in protoplanetary disks. *Astronomy & Astrophysics*, 487(1):265–270, 2008.

[85] Soko Matsumura, Edward W Thommes, Sourav Chatterjee, and Frederic A Rasio. Unstable planetary systems emerging out of gas disks. *The Astrophysical Journal*, 714(1):194, 2010.

[86] NASA. https://solarsystem.nasa.gov/asteroids-comets-and-meteors/asteroids/in-depth/. consulted 15 August 2022.

[87] T Birnstiel, M Fang, and A Johansen. Dust evolution and the formation of planetesimals. *Space Science Reviews*, 205(1):41–75, 2016.

[88] Hiroshi Kobayashi, Hidekazu Tanaka, and Satoshi Okuzumi. From planetesimals to planets in turbulent protoplanetary disks. i. onset of runaway growth. *The Astrophysical Journal*, 817(2):105, 2016.

[89] Nicholas P Ballering and Josh A Eisner. Protoplanetary disk masses from radiative transfer modeling: A case study in taurus. *The Astronomical Journal*, 157(4):144, 2019.

[90] Tatsuya Okamura and Hiroshi Kobayashi. The growth of protoplanets via the accretion of small bodies in disks perturbed by the planetary gravity. *The Astrophysical Journal*, 916(2):109, 2021.

[91] Sanemichi Z Takahashi and Takayuki Muto. Structure formation in a young protoplanetary disk by a magnetic disk wind. *The Astrophysical Journal*, 865(2):102, 2018.

[92] Hidekazu Tanaka, Taku Takeuchi, and William R Ward. Three-dimensional interaction between a planet and an isothermal gaseous disk. i. corotation and lindblad torques and planet migration. *The Astrophysical Journal*, 565(2):1257, 2002.

7. Appendix

7.1. Derivations

The further one gets from the sun, the more space there is to be occupied. And because we simulate in a 3d model, the probability of picking a semi-major axis a is $P(a) = ca^2$. Normalize the function to obtain c:

$$1 = \int_{a_-}^{a^+} P(a)da \longrightarrow c = \frac{3}{a_+^3 - a_-^3} \tag{34}$$

a_+ and a_- being the maximum and minimum value of a. Set the probability density function equal to a random uniform number ξ.

$$f(a) = \int_{a_-}^{a} P(a')da' = \frac{a^3 - a_-^3}{a_+^3 - a_-^3} = \xi \tag{35}$$

Then a becomes

$$a = \sqrt[3]{(a_+^3 - a_-^3)\xi + a_-^3} \tag{36}$$

The probability for picking the inclination I is $P(I) = c\sin(I)$. Normalize to obtain c

$$1 = \int_{0}^{I_{max}} P(I)dI \longrightarrow c = \frac{1}{1 - \cos(I_{max})} \tag{37}$$

Set the probability density function equal to a random uniform number ξ.

$$f(I) = \int_{0}^{I} P(I')dI' = \frac{\cos(I) - \cos(I_{max})}{1 - \cos(I_{max})} = \xi \tag{38}$$

Then I becomes

$$I = \arccos(\xi[1 - \cos(I_{max})] + \cos(I_{max})) \tag{39}$$

7.2. Code

```python
import numpy as np
import matplotlib.pyplot as plt
import math
import random
import scipy.misc
import scipy.special
import scipy.stats
import os
from scipy.optimize import curve_fit
import time

G = 6.67428e-11
M = 1.98847e30
AU = 1.495978707e11
S = 0.00465047*AU
MJ = 1.898e27
SJ = 0.000477895*AU
year = 60*60*24*365

def R(a,b,c):
    return np.array([[np.cos(a)*np.cos(b)-np.sin(a)*np.sin(b)*np.cos(c),-np.cos(a)*np.sin(b)-np.sin(a)*np.cos(b)*np.cos(c),np.sin(a)*np.sin(c)],
                    [np.sin(a)*np.cos(b)+np.cos(a)*np.sin(b)*np.cos(c),-np.sin(a)*np.sin(b)+np.cos(a)*np.cos(b)*np.cos(c), -np.cos(a)*np.sin(c)],
                    [np.sin(b)*np.sin(c),np.cos(b)*np.sin(c),np.cos(c)]])

def gen(N):
    arr = np.array([])
    for n in range(N):
        a,s = np.cbrt(random.uniform(0,1)*(10**3-1**3)+1**3)*AU,10**(random.uniform(-7,-3))*AU
        I = np.arccos(random.uniform(0,1)*(1-np.cos(0.1))+np.cos(0.1))
        e,m,wbar,W,E = random.uniform(0,0.1),(s/SJ)**3*MJ,random.uniform(0,2*np.pi),random.uniform(0,2*np.pi),random.uniform(0,2*np.pi)
        r0,w = np.matmul(R(wbar,W,I),np.array([a-np.cos(E)*e/a,np.sqrt(1-e**2)*a*np.sin(E),0])), np.sqrt(G*M/a**3)
        Lbar, ebar = m*w*a**2*np.sqrt(1-e**2)*R(wbar,W,I)[2], e*R(wbar,W,I)[0]
        arr = np.append(arr,[0,a,e,s,m,r0,Lbar,ebar,w,I])
    arr = arr.reshape(N,10)
    return arr[(arr[:, 1]-arr[:, 1]*arr[:,2]-arr[:,3]).argsort()]


def r_colision(p1,p2):
    Kbar = np.cross(p1[6],p2[6])
    K = np.sqrt(np.dot(Kbar,Kbar))
    l1 = np.dot(p1[6],p1[6])/(G*M*p1[4]**2)
    l2 = np.dot(p2[6],p2[6])/(G*M*p2[4]**2)
    rbar1 = np.array([Kbar*l1/(K+np.dot(p1[7],Kbar)),Kbar*l1/(-K+np.dot(p1[7],Kbar))])
    rbar2 = np.array([Kbar*l2/(K+np.dot(p2[7],Kbar)),Kbar*l2/(-K+np.dot(p2[7],Kbar))])
    r1 = np.array([np.sqrt(np.dot(rbar1[0],rbar1[0])),np.sqrt(np.dot(rbar1[1],rbar1[1]))])
    r2 = np.array([np.sqrt(np.dot(rbar2[0],rbar2[0])),np.sqrt(np.dot(rbar2[1],rbar2[1]))])
    vbar1 = np.array([np.cross(p1[6]/(p1[4]*l1),p1[7]+rbar1[0]/r1[0]),np.cross(p1[6]/(p1[4]*l1),p1[7]+rbar1[1]/r1[1])])
    vbar2 = np.array([np.cross(p2[6]/(p2[4]*l2),p2[7]+rbar2[0]/r2[0]),np.cross(p2[6]/(p2[4]*l2),p2[7]+rbar2[1]/r2[1])])
    d = rbar2 - rbar1
    worbit = np.array([np.cross(vbar1[0],vbar2[0]),np.cross(vbar1[1],vbar2[1])])
    wsq = np.array([np.dot(worbit[0],worbit[0]),np.dot(worbit[1],worbit[1])])
    rcol1 = np.array([rbar1[0] + (np.dot(d[0],np.cross(vbar2[0],worbit[0])/wsq[0]))*vbar1[0],rbar1[1] + (np.dot(d[1],np.cross(vbar2[1],worbit[1])/wsq[1]))*vbar1[1]])
    rcol2 = np.array([rbar2[0] + (np.dot(d[0],np.cross(vbar1[0],worbit[0])/wsq[0]))*vbar2[0],rbar2[1] + (np.dot(d[1],np.cross(vbar1[1],worbit[1])/wsq[1]))*vbar2[1]])
    dcol = np.array([np.sqrt(np.dot(rcol2[0]-rcol1[0],rcol2[0]-rcol1[0])),np.sqrt(np.dot(rcol2[1]-rcol1[1],rcol2[1]-rcol1[1]))])
    arr = np.array([])
    if dcol[0] < p1[3]+p2[3]:
        arr = np.append(arr, [dcol[0],rcol1[0],rcol2[0],vbar1[0],vbar2[0]])
    if dcol[1] < p1[3]+p2[3]:
        arr = np.append(arr, [dcol[1],rcol1[1],rcol2[1],vbar1[1],vbar2[1]])
    if dcol[0] < p1[3]+p2[3] or dcol[1] < p1[3]+p2[3]:
        return arr.reshape(int(len(arr)/5),5)

def pairlist(sys):
    arr = np.array([])
    for i in range(len(sys)):
        for j in range(i+1,len(sys)):
            if sys[i][1]+sys[i][1]*sys[i][2]+sys[i][3] >= sys[j][1]-sys[j][1]*sys[j][2]-sys[j][3]:
                k = r_colision(sys[i],sys[j])
                if type(k) != type(None):
                    for x in k:
                        arr = np.append(arr, [i,j,x])
            else:
                break
    return arr.reshape(int(len(arr)/3),3)

def newpairs(new,nsys,x):
    a =list(range(len(nsys)))
    a.remove(x)
    arr = np.array([])
    for i in a:
        if new[1]+new[1]*new[2]+new[3]>=nsys[i][1]-nsys[i][1]*nsys[i][2]-nsys[i][3]:
            k = r_colision(new,nsys[i])
            if type(k) != type(None):
                for xi in k:
                    arr = np.append(arr, [x,i,xi])
    return arr.reshape(int(len(arr)/3),3)

def t_colision(sys, cl):
    arr = np.array([],dtype=object)
    for i,j,k in cl:
        e1sq, e2sq = np.dot(sys[i][7],sys[i][7]), np.dot(sys[j][7],sys[j][7])
        r1,r2 = np.sqrt(np.dot(k[1],k[1])),np.sqrt(np.dot(k[2],k[2]))
        z1 = np.dot(k[1]/sys[i][1]-1j*(r1*k[3]/(sys[i][1]**2*sys[i][8])),(sys[i][5]-sys[i][7]*np.dot(sys[i][7],sys[i][5]))/(sys[i][1]-e1sq*sys[i][1])+sys[i][7])+(np.dot(sys[i][5],sys[i][7]))/sys[i][1]+e1sq
        z2 = np.dot(k[2]/sys[j][1]-1j*(r2*k[4]/(sys[j][1]**2*sys[j][8])),(sys[j][5]-sys[j][7]*np.dot(sys[j][7],sys[j][5]))/(sys[j][1]-e2sq*sys[j][1])+sys[j][7])+(np.dot(sys[j][5],sys[j][7]))/sys[j][1]+e2sq
        DE1, DE2 = np.angle(z1)%(2*np.pi), np.angle(z2)%(2*np.pi)
        t11 = sys[i][0]+DE1/sys[i][8]-np.dot(np.cross(sys[i][7],k[1]-sys[i][5])/(1-e1sq),sys[i][6]/(G*M*sys[i][4]))
        t21 = sys[j][0]+DE2/sys[j][8]-np.dot(np.cross(sys[j][7],k[2]-sys[j][5])/(1-e2sq),sys[j][6]/(G*M*sys[j][4]))
        ubar, wsq = k[4]-k[3], np.dot(np.cross(k[3],k[4]),np.cross(k[3],k[4]))
        delta = 1/np.abs(t11-t21)*np.sqrt(np.dot(ubar,ubar)*((sys[i][3]+sys[j][3])**2-k[0]**2)/wsq)
        q0,q1,k0,k1 = 2*np.pi/(sys[i][8]*np.abs(t11-t21)),2*np.pi/(sys[j][8]*np.abs(t11-t21)),1,0
        q,alpha,ks,n= np.array([q0,q1]),np.array([]),np.array([k0,k1]),0
        while n <=8:
            q, alpha, ks = np.append(q,[0,0]), np.append(alpha, [0,0]), np.append(ks, [0,0])
            alpha[2*n] = np.floor(q[2*n]/q[2*n+1])
            q[2*n+2] = q[2*n]-alpha[2*n]*q[2*n+1]
            if q[2*n+2] == 0:
                print('iets')
                break
            ks[2*n+2] = ks[2*n] -alpha[2*n]*ks[2*n+1]
            alpha[2*n+1] = np.floor(q[2*n+1]/q[2*n+2])
            q[2*n+3] = q[2*n+1]-alpha[2*n+1]*q[2*n+2]
            if q[2*n+3] == 0:
                print('niets')
                break
            ks[2*n+3] = ks[2*n+1] -alpha[2*n+1]*ks[2*n+2]
            xmax = np.floor((1+delta)/q[2*n+2])+1
            if xmax > 1000000:
                break
            x = np.arange(np.ceil((1-delta)/q[2*n]),xmax,1)
            y = np.maximum(np.zeros(len(x)), np.ceil(q[2*n]*x/q[2*n+1]-(1+delta)/q[2*n+1]))
            for xi in range(len(x)):
                if x[xi]*q[2*n]-y[xi]*q[2*n+1] > 1- delta:
                    sol = x[xi]*ks[2*n]-y[xi]*ks[2*n+1]
                    tnew = t11+2*np.pi*sol/sys[i][8]
                    arr = np.append(arr,[i,j,tnew,k[1:]])
                    n = 10
                    break
            n=n+1
    arr = arr.reshape(int(len(arr)/4),4)
    return arr[(arr[:,2]).argsort()]

def colision(sys,tl):
    i,j,t,k = tl[0]
    removed = tl[~(tl[:,0] == i),:]
    removed = removed[~(removed[:,1] == i),:]
    removed = removed[~(removed[:,0] == j),:]
    removed = removed[~(removed[:,1] == j),:]
    for ii in range(len(removed)):
        if removed[ii][0]>= i:
            removed[ii][0] = removed[ii][0]-1
        if removed[ii][0] >= j:
            removed[ii][0] = removed[ii][0]-1
        if removed[ii][1]>= i:
            removed[ii][1] = removed[ii][1]-1
        if removed[ii][1] >= j:
            removed[ii][1] = removed[ii][1]-1
    s = (sys[i][3]**3+sys[j][3]**3)**(1/3)
    m = sys[i][4]+sys[j][4]
    rbar = (sys[i][4]*k[0]+sys[j][4]*k[1])/m
    v = (sys[i][4]*k[2]+sys[j][4]*k[3])/m
    L = sys[i][6]+sys[j][6]
    l = np.dot(L,L)/(G*M*m**2)
    r = np.sqrt(np.dot(rbar,rbar))
    ebar = np.cross(v,L)/(G*M*m)-rbar/r
    e = np.sqrt(np.dot(ebar,ebar))
    if l>(1+e)*S and e < 1:
        a = l/(1-e**2)
        w = np.sqrt(G*M/(a**3))
        I = np.arccos(L[2]/(m*w*a**2*np.sqrt(1-e**2)))
        new = np.array([t,a,e,s,m,rbar,L,ebar,w,I])
        nsys = np.delete(sys,[i,j],axis=0)
        nsys = np.append(nsys,[t,a,e,s,m,rbar,L,ebar,w,I])
        nsys = nsys.reshape(int(len(nsys)/10),10)
        x = np.where((nsys[:, 1]-nsys[:, 1]*nsys[:,2]-nsys[:,3]).argsort() == len(nsys)-1)
        #print("x = "+str(int(x[0])))
        nsys = nsys[(nsys[:, 1]-nsys[:, 1]*nsys[:,2]-nsys[:,3]).argsort()]
        ncl = newpairs(new,nsys,int(x[0]))
        ntl = t_colision(nsys,ncl)
        for ii in range(len(removed)):
            if removed[ii][0]>= int(x[0]):
                removed[ii][0] = removed[ii][0] + 1
            if removed[ii][1]>= int(x[0]):
                removed[ii][1] = removed[ii][1] + 1
        merged = np.append(removed,ntl)
        merged = merged.reshape(int(len(merged)/4),4)
        arr = merged[(merged[:,2]).argsort()]
        return nsys,arr
    else:
        print('byebye')
        nsys = np.delete(sys,[i,j],axis=0)
        print(len(nsys))
        return nsys,removed

def mainloop(sys):
    n = 1
    nsys = sys
    CL = np.array([])
    Nt = np.array([])
    cl = pairlist(sys)
    tl = t_colision(sys,cl)
    waardes = np.array([np.average(nsys[:,1]),np.average(nsys[:,2]),np.average(nsys[:,3]),np.average(nsys[:,9]),np.std(nsys[:,1]),np.std(nsys[:,2]),np.std(nsys[:,3]),np.std(nsys[:,9]),nsys[nsys[:,3].argsort()][:,3][-10:]])
    while 1>0:
        if len(tl) == 0 :
            break
        CL = np.append(CL,tl[0])
        Nt = np.append(Nt,len(nsys))
        nsys,tl = colision(nsys,tl)
        n= n+1
        waardes = np.append(waardes,[np.average(nsys[:,1]),np.average(nsys[:,2]),np.average(nsys[:,3]),np.average(nsys[:,9]),np.std(nsys[:,1]),np.std(nsys[:,2]),np.std(nsys[:,3]),np.std(nsys[:,9]),nsys[nsys[:,3].argsort()][:,3][-10:]])
    CL = CL.reshape(int(len(CL)/4),4)
    waardes = waardes.reshape(int(len(waardes)/9),9)
    return nsys, CL, Nt, waardes
```