



universität
wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

**'Estimating Water Transfer during Planetary Formation
using Interpolated Results from SPH Collision Catalogues'**

verfasst von / submitted by

Lukas Winkler, BSc

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2022 / Vienna, 2022

Studienkennzahl lt. Studienblatt /
degree programme code as it appears on
the student record sheet:

UA 066 861

Studienrichtung lt. Studienblatt /
degree programme as it appears on
the student record sheet:

Masterstudium Astronomie

Betreut von / Supervisor:

ao. Univ.-Prof. i.R. tit. Univ.-Prof. Dr. Rudolf Dvorak

Contents

Abstract	3
Zusammenfassung	3
Acknowledgements	4
1 Introduction	5
2 SPH-based collision simulation	7
3 Simulation Setup	9
3.1 Initial Conditions	9
3.2 N-Body Integrator	10
3.2.1 Hybrid Integrator	11
3.2.2 Ejections and Solar Encounters	12
3.2.3 Radii	13
3.3 Collision Handling	13
3.4 Output Management	15
3.4.1 Reproducibility	15
3.5 Scenarios	16
4 Mass Loss Estimation	17
4.1 Perfect Merging	17
4.2 Randomized Mass Loss	17
4.3 RBF-based Mass Loss Estimation	18
4.4 Neural Network Mass Loss Estimation	20
5 Results	23
5.1 Randomized Mass Loss	32
6 Discussion and Conclusions	34
6.1 Data Availability	35
6.2 Conclusions	35
List of Figures	37
List of Tables	37
List of Code Fragments	38
Abbreviations	39
Bibliography	39

Abstract

To understand how water came to Earth, we need to better understand the late stage of terrestrial planetary formation. Many N-Body simulations were run in the past describing the mass and water transport and accretion of mass when smaller planetesimals and protoplanetary embryos collide to form the final planets. One major detail that is only considered more accurately recently is the fact that during these collisions mass is lost into space, which affects the final mass and water content of the solar system. But often this is neglected in favour of assuming bodies merge perfectly without mass loss. In this thesis, we propose two alternative methods to estimate this mass loss without having to simulate the physical collision in high resolution by using an existing dataset of collision outcomes. We run a total of 98 N-Body simulations implementing these water loss estimation methods in addition to a randomized mass loss and assuming perfect merging. When comparing our results with existing research, we can reproduce most effects compared to more complex mass loss setups. When considering mass loss, final bodies are less massive and contain significantly less water. In addition, our methods implicitly takes into consideration that earlier collisions are less impactful and shows that most water is accreted in a small number of later collisions. Finally, we confirm that without considering Hit-and-Run collision scenarios, the duration of terrestrial planetary formation cannot be accurately modelled.

Zusammenfassung

Um zu verstehen, wie Wasser auf die Erde gekommen ist, brauchen wir ein besseres Verständnis der Spätphasen der Planetenentstehung. Viele N-Body Simulationen wurden in der Vergangenheit durchgeführt um den Massen- und Wassertransport und den Massenzuwachs, wenn protoplanetare Körper miteinander kollidieren, um die finalen Planeten zu bilden, besser zu beschreiben. Ein grundlegendes Detail, welches erst vor kurzem genauer betrachtet wurde, ist, dass während Kollisionen Masse in den Weltraum verloren geht, was die Masse und den Wasseranteil des fertigen Sonnensystems beeinflusst. Dies wird aber oft vernachlässigt und stattdessen eine perfekte Kollision ohne Massenverlust angenommen. In dieser Arbeit stellen wir zwei alternative Methoden vor, um diesen Massenverlust abzuschätzen ohne die physikalischen Details der Kollision in hoher Auflösung simulieren zu müssen. Hierzu verwenden wir ein existierendes Datenset an Kollisionsergebnissen und implementieren damit die Wasserverlustabschätzungsmethoden zusätzlich zu Perfect Merging und einem zufälligen Massenverlust. Wenn wir das Ergebnis unserer 98 N-Body Simulationen mit existierenden Ergebnissen vergleichen, können wir die meisten Effekte aus komplexeren Simulations-Setups reproduzieren. Wenn Massenverlust berücksichtigt wird, sind die gebildeten Planeten masseärmer und enthalten weniger Wasser. Außerdem berücksichtigt unsere Methoden implizit, dass frühere Kollisionen schwächer sind und zeigen, dass das meiste Wasser in wenigen späten Kollisionen auf Planeten kommt. Zuletzt können wir bestätigen, dass ohne Berücksichtigung von Hit-and-Run Kollisionsszenarien die Dauer der Bildung terrestrischer Planeten nicht zuverlässig bestimmt werden kann.

Acknowledgements

I would like to thank everyone who helped me make this thesis possible. Starting with my supervisor Rudolf Dvorak who made it possible in the first place to write my thesis about this topic and encouraged me about this work. Thanks also to Christoph Burger for providing the scientific foundation this thesis builds on and sharing his dataset and many helpful insights. I would also like to thank Hanno Rein for REBOUND which made the simulation setup of this thesis possible to implement, for helping with misconceptions and for fixing bugs whenever they were found. Similarly, I would like to thank Thomas Maindl who first introduced me to the topic of simulating planetary collisions and who gave encouraging feedback on earlier drafts of this thesis. I also have to thank Markus Rockenbauer and the rest of the Department of Astrophysics at the University of Vienna for allowing me to use one of their servers for the long simulation times. And also many thanks to Stefanie Reiter, Simon Schleich and Fridolin Glatter for proofreading and commenting on this thesis. Finally, I want to thank all of my friends and my family for supporting me during the time it took me to finish this thesis.

1 Introduction

The source of water on Earth and the processes that caused water to end up in the inner parts of the solar system are two of the major questions in planetary formation. During the early stages of the solar system, the innermost parts were too hot for water to condense into ice (Hayashi 1981). While the exact position of this so-called snowline (the border where water ice could form) varies depending on the model (Podolak and Zucker 2004 indicates a minimum radius of 3.2 AU), the earth’s orbit was most likely inside the area where solid water ice was not possible (unless other effects like in Martin and Livio 2012 make ice close to the sun possible). This means water must have moved inside during later stages of planetary formation. While comets were originally thought to be a large source of water (Oró 1961; Owen and Bar-Nun 2001), observations of 67P/Churyumov-Gerasimenko indicated that their isotope signature does not match the one from Earth’s water (Rubin et al. 2015). In addition, dynamical studies like Morbidelli et al. 2000 show that only a small fraction of Earth’s current water mass can come from asteroids. Instead, the majority is accreted in planetary embryos which transferred it to Earth via collisions in the late stages of formation.

To describe this process of collision between planetary embryos during the later stages of planetary formation and the water transfer that is caused by it, numerical N-Body simulations like those in O’Brien, Morbidelli and Levison 2006 are set up. In these, a stage in the planetary formation is considered, where Jupiter and Saturn are on their current orbits (or a circular orbit). The solid mass is split up into a smaller number of more massive planetary embryos and a larger number of smaller planetesimals. An N-Body integrator like MERCURY (Chambers 1999) is then used to simulate the orbital dynamics over 200 million years, during which the bodies will collide and more massive planets form. One limitation of this approach, especially when focusing on water transport, is that collisions are mostly modelled by perfect merging in inelastic mergers, where the linear momentum is conserved, but all mass of the colliding bodies remains in the newly formed body (Izidoro, Morbidelli and Raymond 2014; O’Brien, Morbidelli and Levison 2006; Raymond, Quinn and Lunine 2006). This is a reasonable approximation for low impact collisions, but as we will see later on, stronger collisions are common. The assumption of perfect merging will always overestimate the amount of mass (especially water mass) remaining after a collision which also influences the amount of (water) mass remaining in the final bodies. In this thesis, we will be looking at various methods on how collisions can be handled more realistically.

In Leinhardt and Stewart 2012; Stewart and Leinhardt 2012, a collection of collisions is simulated using the high resolution N-Body integrator PKDGRAV¹. The resulting remnants are analysed and used to group the collisions as perfect merging, partial accretion, net erosion, hit-and-run and graze-and-merge. Also scaling laws are derived and used to create an algorithm on how to handle collisions, that is further improved in Mustill, Davies and Johansen 2018. The major difference of this approach to everything else in this thesis, is that the bodies used had a radius of 10 km and a mass of $4.2 \cdot 10^{15}$ kg. They are therefore many orders of magnitudes smaller, which isn’t representative of the bodies in the final stages of planetary formation.

Another much newer work that is closer to this scenario is Cambioni, Asphaug et al. 2019, in which 769 collisions are simulated using *Smoothed Particle Hydrodynamics* (SPH). The simulations outcomes are then used to train two separate neural networks. One of them tries to predict collision outcomes such as the masses of the remnants from the impact parameters in a similar way as described in Section 4.4. The other network tries to classify the type of collision scenario. Emsenhuber et al. 2020 improves this

¹<https://faculty.washington.edu/trq/hpcc/faculty/trq/brandon/pkdgrav.html>, Stadel 2001

method by adding a third network that predicts the orbital parameters of the two remnants in case of a hit-and-run scenario. And the latest work in Cambioni, Jacobson et al. 2021 further expands this method to focus more precisely on the planetary composition.

But while all methods described until now greatly improve how the collision process is modelled and therefore give much more accurate results than just assuming perfect merging, they don't include water ice mantles in their models and can therefore not give any information on how these processes affect the water transport in the late stage planetary formation. This changes with Burger, Bazzó and Schäfer 2020 (henceforth cited as ChB), in which all bodies consist of an iron core, a basalt mantle and a water ice shell. While the initial conditions are similar to the previously mentioned works, the major difference is that for every collision of two bodies, the N-Body integration is halted, and a dedicated SPH-based simulation of the collision is started (as will be explained in Section 2). These results are then used to resolve the collision and continue the N-Body integration.

Another similar paper is L. Zhou, Dvorak and L.-Y. Zhou 2021 (henceforth cited as ZhL), which replaces the estimation of the mass loss of the three materials during collisions with a random value in a predefined range, while still more realistically reducing the masses of bodies than assuming perfect merging. See Section 4.2 for more details on this method.

Yet another very recent work is Haghighipour and Maindl 2022 which also argues that perfect merging is not an acceptable approximation when simulating planetary formation. For this the authors show that the mass losses in each simulation accumulate and affect the final planets' mass strongly.

In this thesis, we will describe our own N-Body simulation setup based on the ideas of ChB and using the REBOUND package of integrators (Rein and Liu 2012). Then, we will introduce two new mass loss estimation methods that are using the dataset of collisions from ChB to instantaneously estimate a realistic mass loss and compare their impact on the formed planets with simulations assuming perfect merging and the results from ChB and ZhL.

Chapter 2 introduces the high-resolution SPH-based collision simulation used in ChB. Chapter 3 explains the setup of our N-Body simulations and how ejections and collisions are handled in them while Chapter 4 focuses on the two new mass loss estimation methods and how they are implemented together with perfect merging and our implementation of the method from ZhL. The results of the collection of simulations are then shown in Chapter 5 and are discussed and compared in more detail in Chapter 6.

2 SPH-based collision simulation

In order to accurately predict the outcome of collisions between large bodies, we need to be able to accurately resolve their interactions in a high resolution. One method that allows us to model the two bodies using a large amount of particles is using 3D *Smoothed Particle Hydrodynamics* (SPH). This way we can look at a specific collision between two bodies with a set of initial conditions, let them collide and afterwards observe the physical properties of the remnants. SPH has already been used before to estimate collision outcomes as mentioned in the introduction. This thesis uses the outcomes from ChB which are described together with their simulation setup in the following and are based on the work in Burger, Maindl and Schäfer 2018.

ChB uses the SPH code `miluphcuda`² (Schäfer, Riecker et al. 2016; Schäfer, Wandel et al. 2020) which uses CUDA to parallelize the calculations on GPUs. `miluphcuda` allows simulating the collision of two bodies using 20 000 to 75 000 SPH particles depending on the mass ratio between them. Both bodies are made of an iron core, a basalt stone mantle and a water-ice shell with material properties from Melosh 1989 for iron and Benz and Asphaug 1999 for basalt and ice. They are already in a relaxed state to avoid inaccuracies during the collision.

The primary input parameters for the simulations are the impact velocity v_{esc} in units of the mutual escape velocity, the impact angle α , the projectile and target masses and the water and core mass fractions of both bodies. The main output values of the simulations are the masses of the largest and second-largest object³ and the water and core mass fractions of the largest two bodies. These parameters can be seen again in Section 4.3 when we try to replace these simulations with interpolated results.

The dataset used in this thesis consists of the export of 10 000 individual collisions from 49 N-Body simulation runs. A very small fraction of collisions had values that could not be used, leaving a remaining dataset of 9977 collisions outcomes with their corresponding collision parameters. We can take a very general look at it by calculating the Pearson correlation coefficient between the input parameters and the water mass fraction to see which parameters influence water loss most (Figure 2.2). As expected collision speed and impact angle matter most, with the masses and mass fractions between the two objects also having an influence on the result. The water mass fraction of the two bodies does not have a significant impact on the percentage of water that is lost during the collisions.

²The code is available at <https://github.com/christophmschaefer/miluphcuda>.

³This is done by using a friends-of-friends algorithm to group the particles into fragments and aggregate the masses of gravitationally bound fragments onto each other.

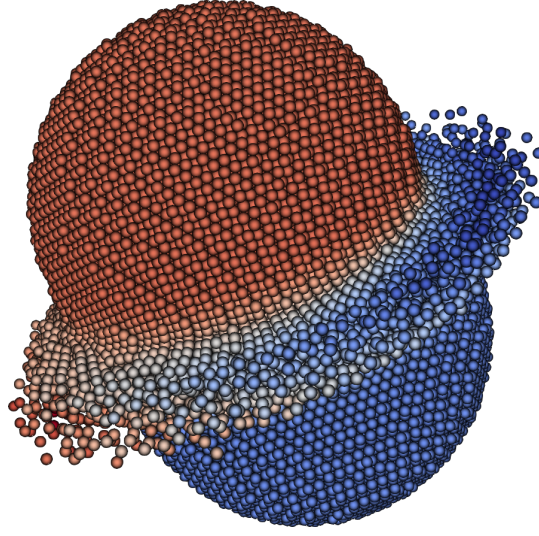


Figure 2.1: A snapshot of an SPH simulation like the ones used here. The colour indicates the velocity in the direction of the collision.

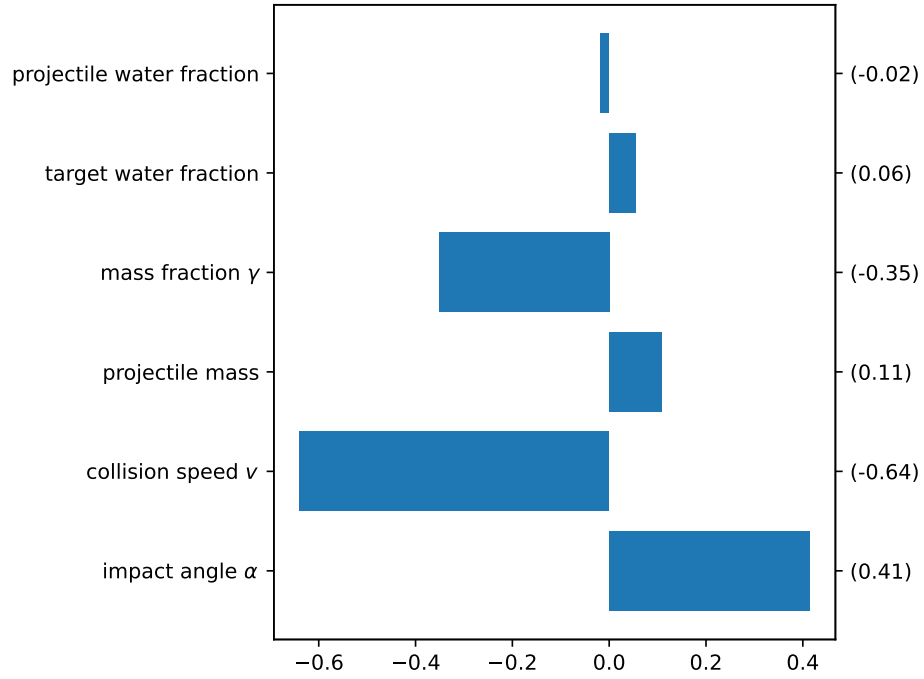


Figure 2.2: The Pearson correlation coefficient between the input parameters and the water mass fraction.

3 Simulation Setup

3.1 Initial Conditions

To keep the results of the simulation comparable with [ChB](#), the initial conditions of the N-Body integration are kept as similar as possible. The same script is used to generate them and deviations in the parameters are described in the following.

The simulation starts with an initial debris disk consisting of both *planetary embryos* and *planetesimals* orbiting around the sun during the late-stage accretion phase of terrestrial planet formation. Both of them together follow a disk profile starting at 0.5 AU and ending at 4.0 AU with the surface density Σ described in [\(3.1\)](#). The total solid mass in the disk is around $5 M_{\oplus}$.

$$\Sigma_{\text{solid}}(r) = \Sigma_0 \left(\frac{r}{1 \text{ AU}} \right)^{-\alpha} \quad (3.1)$$

$$\alpha = 1.5 \quad (3.2)$$

$$\Sigma_0 = 1.125 \cdot 10^{-6} M_{\odot} / \text{AU}^2 = 10 \frac{\text{g}}{\text{cm}^2} \quad (3.3)$$

The embryos are the more massive population and make up 70 % of the total mass. They are placed 10 times their mutual hill radii [\(3.4\)](#) away from each other with a and m being their semi-major axes and masses, respectively, and M_0 being the mass of the central star. The mass of the embryos is the isolation mass M_{iso} which describes the mass available in the annulus between two particles [\(3.5\)](#). With these conditions, around 34 planetary embryos are generated.

$$R_H = \frac{a_1 + a_2}{2} \left(\frac{m_1 + m_2}{3M_0} \right)^{\frac{1}{3}} \quad (3.4)$$

$$M_{\text{iso}} = 2\pi ab \Sigma_{\text{solid}}(a) \quad (3.5)$$

The remaining 30 % of the mass is assumed to have not yet been accreted and is split up into 250 planetesimals. This results in each of them having a mass of around $4.85 \cdot 10^{22} \text{ kg}$ ($\approx 50 M_{\text{Ceres}} \approx 0.66 M_{\text{Moon}}$). This is about an order of magnitude less than the mass of the embryos which ranges from $3.42 \cdot 10^{23} \text{ kg}$ to $1.43 \cdot 10^{24} \text{ kg}$.

With the mass m and semi-major axis a of all bodies defined, the eccentricities and inclinations of all bodies are distributed in a Rayleigh distribution (Code Fragment [3.1](#) shows the parameters for embryos⁴). The remaining orbital parameters are uniformly distributed between 0° and 360° .

The water mass fraction (WMF) of the disk bodies follows the relation from Raymond, Quinn and Lunine [2004](#) as shown in [\(3.6\)](#). This splits the solar system into a water-rich outer part and a water poor inner region with an intermediate in-between (Figure [3.1](#)). The core mass fraction (CMF) is set to a fixed value of 0.25, with the mantle mass fraction (MMF) making up the rest of the bodies' mass.

⁴They are called ‘minor bodies’ in the code generating the initial conditions.

```

286 # eccentricities
287 ecc = np.random.rayleigh(0.0005, n_minor)
288 # inclinations [rad]
289 inc = np.random.rayleigh(0.05 * deg2rad, n_minor)
290 # arguments of perihelion [rad]
291 aph = np.random.uniform(0.0, 2.0 * np.pi, n_minor)
292 # longitudes of ascending node [rad]
293 lan = np.random.uniform(0.0, 2.0 * np.pi, n_minor)
294 # mean anomalies [rad]
295 man = np.random.uniform(0.0, 2.0 * np.pi, n_minor)

```

Code Fragment 3.1: initcon/InitCondGen.py:286-299 (simplified)

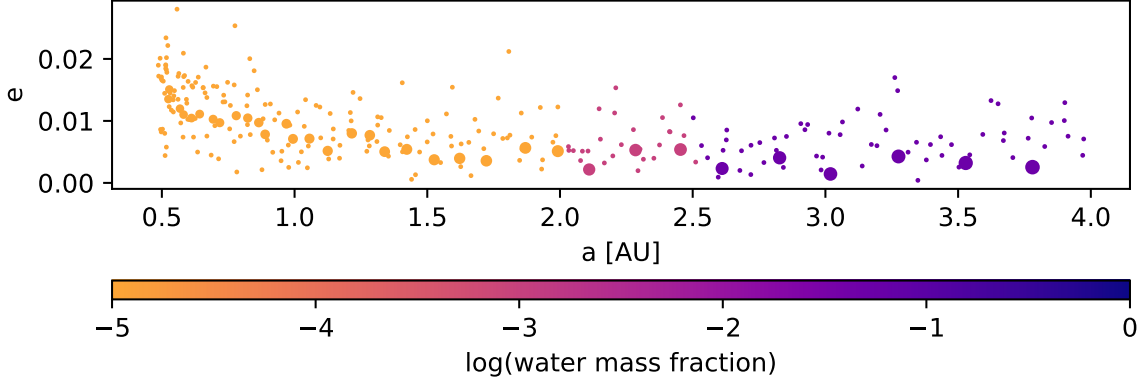


Figure 3.1: Minor bodies in first set of initial conditions (see Section 3.5).

$$\text{WMF} = \begin{cases} 10^{-5} & a < 2 \text{ AU} \\ 10^{-3} & 2 \text{ AU} \leq a < 2.5 \text{ AU} \\ 0.05 & 2.5 \text{ AU} \leq a \end{cases} \quad (3.6)$$

In addition to the around 284 bodies described here, we add a central star with the properties of the sun, and Jupiter and Saturn with their current orbits, but their eccentricities set to 0, so identical to the circular orbits in the cJS scenarios in ChB and similar to the Nice Model (Tsiganis et al. 2005).

3.2 N-Body Integrator

To integrate the N-Body interactions between the bodies in the system, the REBOUND package of integrators (Rein and Liu 2012)⁵ is used, which bundles multiple integrators and other useful tools in one easily extensible package which can be used in Python. The hybrid integrator MERCURIUS (Rein, Hernandez et al. 2019) is chosen as a compromise between runtime and physical accuracy. It uses the fast symplectic integrator WHFAST (Rein and Tamayo 2015) for the main calculation of the gravitational interactions between bodies, but uses a switching function inspired by MERCURY (Chambers 1999) to change to the non-symplectic integrator IAS15 (Rein and Spiegel 2015) with an adaptive timestep whenever two bodies come close to each other. This allows to resolve collisions between bodies accurately, while at the same time benefiting from the faster integration time of WHFAST. To drastically speed up the run time of the simulation by reducing the number of N^2 -Interactions, planetesimals are not interacting with each other,

⁵The source code of REBOUND can be found at <https://github.com/hannorein/rebound>.

only with other embryos, gas giants and the sun, while planetary embryos are gravitationally interacting with all other bodies. This is equivalent to the `pp0` scenarios from [ChB](#) which seem to achieve nearly the same results as scenarios with planetesimal interaction (`pp1`).

The time step of WHFAST is set to 0.01 years = 3.65 days while at the same time care is taken that no body has an orbital period of below 20 timesteps (this factor will be called S in Section 3.2.2). Every simulation is run for 200 Myr with intermediate results being saved every 20 000 years (see Section 3.4).

3.2.1 Hybrid Integrator

One of the most important parts of the N-Body setup is the switching between the fixed-timestep symplectic integrator and the adaptive timestep integrator, as it ensures the accuracy of the collision handling. This is handled in MERCURIUS (Rein, Hernandez et al. 2019) by the (force) switching function L which determines which method is used depending on the distance of two bodies r in units of the critical distance r_{crit} . This critical distance depends on a few parameters (Code Fragment 3.2), but mostly is `rim→hillfac` times the Hill radius. This factor is set to the REBOUND default of 3 in all simulations.

```

416 double dcrit = 0;
417 // Criteria 1: average velocity
418 dcrit = MAX(dcrit, vc*0.4*r→dt);
419 // Criteria 2: current velocity
420 dcrit = MAX(dcrit, sqrt(v2)*0.4*r→dt);
421 // Criteria 3: Hill radius
422 dcrit = MAX(dcrit, rim→hillfac*a*cbrt(r→particles[i].m/(3.*r→particles[0].m)));
423 // Criteria 4: physical radius
424 dcrit = MAX(dcrit, 2.*r→particles[i].r);

```

Code Fragment 3.2: REBOUND: main/src/integrator_mercurius.c:416-424 ([1074be9583](#))

The actual switching function L is then calculated according to (3.7) with y being defined in (3.8), which is the same as in MERCURY (Chambers 1999) and implemented as shown in Code Fragment 3.3.

$$L_{\text{merc}}(r) = \begin{cases} 0 & y \leq 0 \\ 10y^3 - 15y^4 + 6y^5 & 0 < y < 1 \\ 1 & y \geq 1 \end{cases} \quad (3.7)$$

$$y = \frac{r - 0.1r_{\text{crit}}}{0.9r_{\text{crit}}} \quad (3.8)$$

```

44 double reb_integrator_mercurius_L_mercury(const struct reb_simulation* const r, double d, double dcrit){
45     // This is the changeover function used by the Mercury integrator.
46     double y = (d-0.1*dcrit)/(0.9*dcrit);
47     if (y<0.){
48         return 0.;
49     }else if (y>1.){
50         return 1.;
51     }else{
52         return 10.*(y*y*y) - 15.*(y*y*y*y) + 6.*(y*y*y*y*y);
53     }
54 }

```

Code Fragment 3.3: REBOUND: main/src/integrator_mercurius.c:44-54 ([1074be9583](#))

Using MERCURIUS instead of manually switching between the two integrators is one of the more significant differences in this work compared to [ChB](#).

3.2.2 Ejections and Solar Encounters

At the beginning of the simulation, the innermost body has a semi-major axis of 0.5 AU. This means that according to Kepler's third law, it has an orbital period of 0.35 yr^6 , which means that with a timestep of 0.01 yr, more than the targeted 20 ($= S$) timesteps are used to resolve one orbital period.

But the orbits do not stay that way and due to collisions and other influences, orbital bodies will get thrown on orbits that are closer to the sun (without necessarily colliding with it). This means that we either simulate our system with a timestep that is too high to accurately resolve it, resulting in an inaccurate simulation, or we will use IAS15 for the majority of the simulation time and lose the performance of the non-symplectic integrator.

The solution for this problem is ignoring objects that come too close to the sun (closer than a_{cutoff}) and assuming they have merged with the sun instantaneously.

$$a_{\text{cutoff}} = (\Delta t \cdot S)^{\frac{2}{3}} = 0.20^{\frac{2}{3}} \approx 0.34 \text{ AU} \quad (3.9)$$

This is implemented using a C function that is called on every timestep. Every 100 timesteps, it checks if any body is too close to the sun and deletes it. This is done by both comparing the current distance and the perihelion distance (Code Fragment 3.4). The same method also allows us to check for bodies that have been ejected from the system (more than 150 AU from coordinate origin) at the same time, remove them and record this separately. Additionally, bodies are sometimes scattered outside the inner system to perihelion distances beyond 11 AU, but are not immediately getting unstable enough to be removed by the other two checks. Therefore, they are also removed by an explicit third check, so they don't slow down the simulation.

```

1 void heartbeat(struct reb_simulation *sim) {
2     if ((sim->steps_done % 100) == 0) {
3         const struct reb_particle *const particles = sim->particles;
4         int N = sim->N - sim->N_var;
5         for (int i = 1; i < N; i++) { // skip sun
6             struct reb_particle p = particles[i];
7             double distance_squared = p.x * p.x + p.y * p.y + p.z * p.z;
8             struct reb_orbit tmp_orbit = reb_tools_particle_to_orbit(sim->G, p, sim->particles[0]);
9             double perihelion_dist = tmp_orbit.a * (1.0 - tmp_orbit.e);
10            if (distance_squared > max_distance_from_sun_squared) {
11                reb_remove_by_hash(sim, p.hash, 1);
12            } else if (distance_squared < min_distance_from_sun_squared ||
13                      (tmp_orbit.e < 1.0 &&
14                       perihelion_dist * perihelion_dist <
15                       min_distance_from_sun_squared)
16            ) {
17                reb_remove_by_hash(sim, p.hash, 1);
18            }
19            // add mass of deleted particle to sun
20            struct reb_particle sun = sim->particles[0];
21            sun.m += mass;
22        } else if (tmp_orbit.e < 1.0 && perihelion_dist > 11.) {
23            reb_remove_by_hash(sim, p.hash, 1);
24        }
25        // [...] (store information and synchronize integrator)
26    }
27 }
28 }
```

Code Fragment 3.4: heartbeat/heartbeat.c (simplified)

⁶ $= \sqrt{\left(\frac{0.5 \text{ AU}}{1 \text{ AU}}\right)^3} 1 \text{ yr}$

3.2.3 Radii

We consider a collision between two bodies if they come closer than the sum of their radii and are therefore touching each other. But for this to work, we need to estimate the radius of all our bodies. The approach used is a simplified one, which allows an easier implementation. All three layers of the bodies are assumed to be made of a uniform density of iron ($\rho = 7800 \frac{\text{kg}}{\text{m}^3}$), basalt ($\rho = 2700 \frac{\text{kg}}{\text{m}^3}$) and ice ($\rho = 917 \frac{\text{kg}}{\text{m}^3}$)⁷. This is not true in reality as the bodies are in a hydrostatic equilibrium and the SPH simulations consider this by running a relaxation procedure (as described in Appendix A. of Burger, Maindl and Schäfer 2018) before the start of the simulation. This simplified assumption makes it very easy to assign a radius to every body in the simulation by following equations (3.10), (3.11) and (3.12) as implemented in Code Fragment 3.5.

$$R_{\text{core}} = \sqrt[3]{\frac{m_{\text{core}}}{\rho_{\text{iron}}} \frac{3}{4\pi}} \quad (3.10)$$

$$R_{\text{mantle}} = \sqrt[3]{\frac{m_{\text{mantle}}}{\rho_{\text{basalt}}} \frac{3}{4\pi} + R_{\text{core}}^3} \quad (3.11)$$

$$R = \sqrt[3]{\frac{m_{\text{shell}}}{\rho_{\text{ice}}} \frac{3}{4\pi} + R_{\text{mantle}}^3} \quad (3.12)$$

```

1 class PlanetaryRadius:
2     # [ ... ]
3     @property
4     def core_radius(self) → float:
5         return (self.core_mass / self.iron_density * 3 / 4 / pi) ** (1 / 3)
6
7     @property
8     def mantle_radius(self) → float:
9         return (self.mantle_mass / self.basalt_density * 3 / 4 / pi + self.core_radius ** 3) ** (1 / 3)
10
11     @property
12     def total_radius(self) → float:
13         return (self.shell_mass / self.ice_density * 3 / 4 / pi + self.mantle_radius ** 3) ** (1 / 3)

```

Code Fragment 3.5: utils/radius.py (simplified)

3.3 Collision Handling

The direct collision detection module (REB_COLLISION_DIRECT) of REBOUND is used to detect whenever two bodies collide with each other. It is a simple $\mathcal{O}(n^2)$ search that after every timestep checks the distance from every body to every other body to see if they overlap. If this is the case, `collision_resolve` will be called, for which we provide a custom Python function.

Besides the simulation object, the function also gets passed the particle indices of the two involved bodies. As `collision.p1` and `collision.p2` are assigned randomly, the first step is to name the more massive body ‘target’ and the other body ‘projectile’. This makes sure that we can later on assume that the mass fraction $\gamma = \frac{M_{\text{projectile}}}{M_{\text{target}}}$ is always below 1. Also, the resulting body will always continue to be of the same ‘type’ (‘star’, ‘gas giant’, ‘embryo’ or ‘planetesimal’) as the target of the collision.

With the position and velocity vectors and the masses of the two bodies we can calculate the collision angle α using (3.13), the impact velocity v_{esc} in units of the mutual escape velocity using the escape

⁷The same materials and densities as in the SPH simulations are used here.

3 Simulation Setup

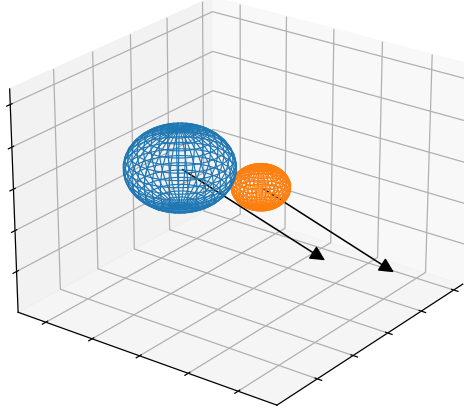


Figure 3.2: Visualization of a collision with $\alpha = 20.29^\circ$, $\frac{v}{v_{esc}} = 1.02$ and $\gamma = 0.14$.

velocity from (3.14) and the mass fraction γ . An example of a collision that happened during the simulations can be seen in Figure 3.2. It is important to keep in mind that collision angle α does not describe the angle between the bodies velocity vectors, but rather how central the bodies hit each other. $\alpha = 0^\circ$ means that it is a perfect hit in the centre of the body, while at $\alpha = 90^\circ$ the two bodies are barely touching each other (Figure 3.3).

$$\alpha = \arccos\left(\frac{\Delta\vec{r}\Delta\vec{v}}{|\Delta\vec{r}||\Delta\vec{v}|}\right) \quad (3.13)$$

$$v_{\text{escape}} = \sqrt{\frac{2G(M_{\text{target}} + M_{\text{projectile}})}{r_{\text{target}} + r_{\text{projectile}}}} \quad (3.14)$$

$$\gamma = \frac{M_{\text{projectile}}}{M_{\text{target}}} \quad (3.15)$$

These three parameters, the projectile mass $M_{\text{projectile}}$ and the water fraction of the two bodies are then passed to the mass loss estimation module, which returns the fractions of core (iron), mantle (basalt) and shell (water ice) mass that is lost during the impact. The different approaches for this estimation are described in Section 4.

After the estimation, the total mass of core, mantle and shell, each reduced by the mass loss, is merged into a new body which is given a velocity conserving momentum. Also, the new core and shell mass fractions of the merged body are recorded, and all properties of the collision are stored for later analysis.

$$v_{\text{merged}} = \frac{v_{\text{target}}M_{\text{target}} + v_{\text{projectile}}M_{\text{projectile}}}{M_{\text{merged}}} \quad (3.16)$$

As a last step, the colliding body with the lower particle index⁸ is replaced with the newly formed body and REBOUND is instructed to delete the other, move all particles in the simulation to coordinates with the new centre of mass at the origin and synchronize the integrator.

⁸As we are using both active and semi-active particles (which are last in the particle index), we need to always keep the object with the lower index (which will always be an active particle, as two semi-active particles cannot collide) to make sure the collision result stays an active particle.

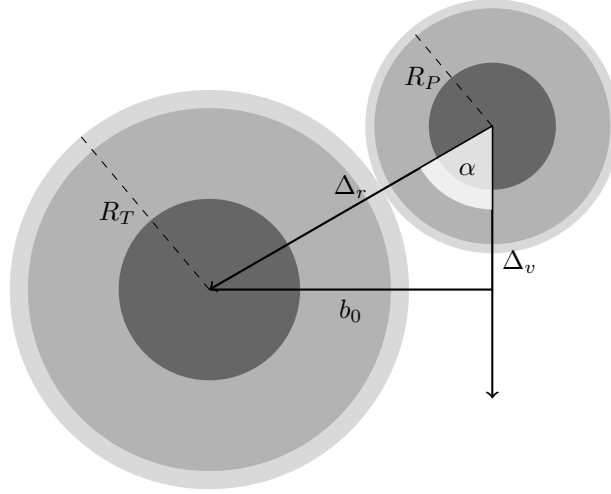


Figure 3.3: Visualization of the collision angle α (here $\alpha = 60^\circ$), based on Figure 1 in [ChB](#).

3.4 Output Management

To be able to analyse the results easily, it is important to save the state of the simulation in much detail. REBOUND facilitates this by allowing to write the whole state of the integrator to a file (Rein and Tamayo 2017). This way, we can save 10 000 full snapshots during each simulation run and can afterwards read the properties of all bodies during the integration, making it easy to plot values like in Figure 5.2a without explicitly logging these values during the simulation run. In addition to the simulation state, we are also storing the total energy every 100 years, the initial conditions of the simulation, the additional properties of each body (water and core mass fractions, type and time of escape or collision with the sun) and the properties of every collision.

3.4.1 Reproducibility

The outcome of individual simulations is very chaotic as a slightly different outcome in an early collision can completely change how stable later orbits are. This not only means we need to take a look at the results of a larger number of simulations to see the influence of our mass loss handling (see Section 3.5), but also that we need to take care to make the simulation as reproducible as possible in order to allow comparable test runs. As an example, all water loss methods (apart from the randomized method) need to always give the same output for the same collision parameters. This also affects collision handling as while REBOUND randomizes the order of the two particles affected⁹, we then reorder them and call the more massive body ‘target’ to always keep the collision outcomes and numbering of child bodies the same (see Section 3.3 for more details). Also, we make sure bodies get the same IDs in every simulation run by assigning a global counter which is a number that is incremented for every newly formed body. All of this assures that while slightly different initial conditions might result in completely different planets formed, identical initial conditions will always give byte-for-byte identical REBOUND snapshots and identical output files apart from the wall-time (the amount of time the simulation takes).

In addition, all simulation runs in this thesis were run on the same computer with no dependencies updated while running them. For these simulations version `d2f2ca44d828b3687ffc759ff1f8f9584fac4edb`¹⁰ and

⁹This is done to avoid statistical effects in simulations with magnitudes higher numbers of collisions like Saturn’s rings.

¹⁰These hashes point at specific versions of the in the version control system used which can be found at <https://lw1.at/r/masters-thesis-code>.

[1f09310b2efd7a411efb29d0a46adaec07689bf8](#)¹¹ of the software created in this thesis were used.

3.5 Scenarios

As the initial conditions are randomized and the collision process is chaotic, a large number of simulation runs is required to get an accurate picture over the properties of the resulting planets. Contrary to this, the long simulation run time¹² is detrimental to running a large number of simulations (ChB used 20 SPH and 16 perfect merging simulations split over four simulation setups each). As a compromise between those two aspects, 40 initial conditions were created as described in Section 3.1 and one simulation using each mass loss estimation method from Chapter 4 (RBF, NN, LZ, PM) was run for the first 20 initial conditions. Two of the LZ simulations (7 and 8) failed due to issues unrelated to the code, so there are only 18 simulations in the LZ dataset. In addition, 20 more simulations were run with the remaining initial conditions and only the RBF method. To keep comparisons fair even though the RBF set contains more simulations, a RBF_SM subset is defined consisting only of the first 20 simulation runs.

¹¹This version includes additional fixes for the randomized mass loss method

¹²The exact run time depends on how fast bodies collide and the number of particles (and therefore N^2 -interactions) decreases (Figure 5.2), but averages 139 ± 20 CPU-hours for the non-LZ scenarios. Due to the fewer collisions in the LZ scenarios (Section 5.1) these simulations took significantly longer with 380 ± 64 CPU-hours. The randomized input is also responsible for a larger spread in this value with the longest simulation taking 557 CPU-hours (about 23 days).

4 Mass Loss Estimation

Four different methods for estimating the amount of mass of each of the three materials lost during a collision are described in this thesis. Their influence on the final simulation outcomes is described in Section 5.

4.1 Perfect Merging

This is by far the simplest approach to implement as we just assume that no mass will be lost and the core/mantle/shell mass retention fraction is always set to 1, independent of the collision parameters. Afterwards these values will be used in the collision handling to calculate the mass and velocity of the new merged body as described in Section 3.3. Implementing perfect merging this way allows us to compare the exact same simulation with and without perfect merging and therefore see the impact of more realistic collision modelling.

```
6 class PerfectMerging(Massloss):
7     name = "perfectmerging"
8
9     def estimate(self, alpha, velocity, projectile_mass, gamma) → Tuple[float, float, float]:
10         return 1, 1, 1
```

Code Fragment 4.1: massloss/perfect_merging.py:6-10

4.2 Randomized Mass Loss

This method reuses the approach introduced in ZhL. It is similar to the perfect merging approach in that it works independently of the collision parameters and is very easy to implement. The mass retention fraction r is estimated as a random number in a defined range, with R being a random float between 0 and 1.

$$r_{\text{core}} = \delta_{\text{core}}^{\text{low}} + R_{\text{core}}(\delta_{\text{core}}^{\text{up}} - \delta_{\text{core}}^{\text{low}}) \quad (4.1)$$

$$r_{\text{water}} = \delta_{\text{water}}^{\text{low}} + R_{\text{water}}(\delta_{\text{water}}^{\text{up}} - \delta_{\text{water}}^{\text{low}}) \quad (4.2)$$

$$r_{\text{mantle}} = r_{\text{core}} \quad (4.3)$$

As the bodies in ZhL are only made up of a core and a water ice shell, the mass retention fraction for the mantle is set to the same as the one for the core. The range for the randomized values is taken from the original paper. While they might not be applicable to the slightly different scenario here, they nevertheless show the influence of adding mass loss to the formed system.

$$\begin{aligned}\delta_{\text{core}}^{\text{low}} &= 0.01 \\ \delta_{\text{water}}^{\text{low}} &= 0.01 \\ \delta_{\text{core}}^{\text{up}} &= 0.08 \\ \delta_{\text{water}}^{\text{up}} &= 0.10\end{aligned}$$

4.3 RBF-based Mass Loss Estimation

This is the main mass loss estimation method of this thesis, and unlike the previous methods, it takes the properties of the collision in consideration. For this, it uses the dataset of around 10 000 collisions from [ChB](#) as described in Section 2.

The collision parameter set of the dataset consists of a 9977×6 array of the collision input variables impact angle α , collision velocity v , projectile mass $M_{\text{projectile}}$, mass ratio between the two bodies γ , target water mass fraction and projectile water mass fraction for each of the 9977 collisions used. From now on the array of these six parameters will be called X . The output dataset consists of the water/mantle/core mass retention fraction for the same collisions. They are defined as the fraction between the total mass of a material in the two most massive remnants of a collision and the total mass of the material in the two initial bodies. The combination of the three mass retention fractions for a collision will from now on be called Y . The goal for a mass loss estimation method is now to find a method f that can accurately estimate Y given an arbitrary set of X ($f(X) = Y$). One initial step that helps with this is to calculate the mean and standard deviation for every collision parameter over the dataset and use them to then normalize any arbitrary X before continuing. Otherwise, some parameters like the projectile mass would dominate any estimation method purely by being orders of magnitude larger than the others.

The method that is then used to estimate a Y for every X given the dataset of existing collisions is based on *Radial Basis Functions* as described in Du Toit 2008 and was already successfully applied to a much smaller dataset in the Bachelor's Thesis Winkler 2019. Functions ϕ in x , which only depend on the distance $|x|$ (so $\phi(x) = \phi(|x|)$), are called *radial* functions. To be able to use this for interpolation, we need to find an interpolation function $s(x)$ that is the same as our given Y for every X from the dataset (4.4).

$$s(X_i) = Y_i, \quad i = 1, 2, \dots, n \quad (4.4)$$

For all other values, we use a linear combination of $\phi(|x - X_i|)$ with the radial function ϕ we will choose later and n constants λ_i .

$$s(x) = \sum_{i=1}^n \lambda_i \phi(\|x - x_i\|) \quad (4.5)$$

$$Y_j = \sum_{i=1}^n \lambda_i \phi(\|X_j - x_i\|), \quad j = 1, 2, \dots, n \quad (4.6)$$

This allows us to write the same relation as a linear matrix equation

4 Mass Loss Estimation

```

1 class RbfMassloss(Massloss):
2     name = "rbf"
3
4     def __init__(self):
5         # [...] (load the collision dataset)
6         self.scaler = CustomScaler()
7         self.scaler.fit(simulations.X)
8         scaled_data = self.scaler.transform_data(simulations.X)
9
10        self.interpolator = Rbf(*scaled_data.T, simulations.Y.T, function="linear", mode="N-D")
11
12    def estimate(self, alpha, velocity, projectile_mass, gamma) → Tuple[float, float, float]:
13        hard_coded_water_mass_fraction = 1e-5
14        estimation_input = [alpha, velocity, projectile_mass, gamma,
15        hard_coded_water_mass_fraction, hard_coded_water_mass_fraction]
16
17        scaled_input = list(self.scaler.transform_parameters(testinput))
18        water_retention, mantle_retention, core_retention = self.interpolator(*scaled_input)
19        return float(water_retention), float(mantle_retention), float(core_retention)

```

Code Fragment 4.2: massloss/rbf_massloss.py (simplified)

$$\begin{bmatrix} \phi(\|x_1 - x_1\|) & \phi(\|x_2 - x_1\|) & \dots & \phi(\|x_n - x_1\|) \\ \phi(\|x_1 - x_2\|) & \phi(\|x_2 - x_2\|) & \dots & \phi(\|x_n - x_2\|) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(\|x_1 - x_n\|) & \phi(\|x_2 - x_n\|) & \dots & \phi(\|x_n - x_n\|) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} \quad (4.7)$$

or simply

$$\Phi \lambda = Y \quad (4.8)$$

with the symmetric¹³ $x \times n$ matrix Φ . This also means that we only have to solve this (9977×9977) matrix equation once at the beginning of the simulation to get λ and can then rather quickly estimate values. The whole method can be used in m dimension by using an $x \in \mathbb{R}^m$ and a norm in $\mathbb{R}^m$ for $\| \cdot \|$. In our case, $m = 6$ for the six impact parameters in X and we can repeat the method three times for each of the three mass retention fractions in Y .

We implement this method using `interpolate.Rbf()` from the `scipy` Python module (Virtanen et al. 2020), as seen in Code Fragment 4.2. Many different radial functions are supported, but we pick the simplest of them, the linear function $\Phi(r) = r$.

As the water mass fractions in the SPH simulation set and the ones in the N-Body simulations are not comparable¹⁴, we cannot just use them in the interpolation. But as we can see in Figure 2.2, the water mass fractions hardly influence the estimation outcome, so we can simply set them to 10^{-5} and ignore the further details.

Figure 4.1 shows the output of the interpolation for water mass retention¹⁵ applied to two different projectile masses covering the main part of the parameter space of collision angle and velocity.

¹³as $\|x_j - x_i\| = \|x_i - x_j\|$

¹⁴If a realistic water fraction of 10^{-5} was used in the SPH collision, the resolution of around 50 000 particles would mean that there was not even a full layer of water particles and the results would be unrealistic because of this.

¹⁵The same figure for mantle and core material looks similar, but the border moves to the top right as it needs a stronger impact to lose mass from the inner parts of the bodies.

4 Mass Loss Estimation

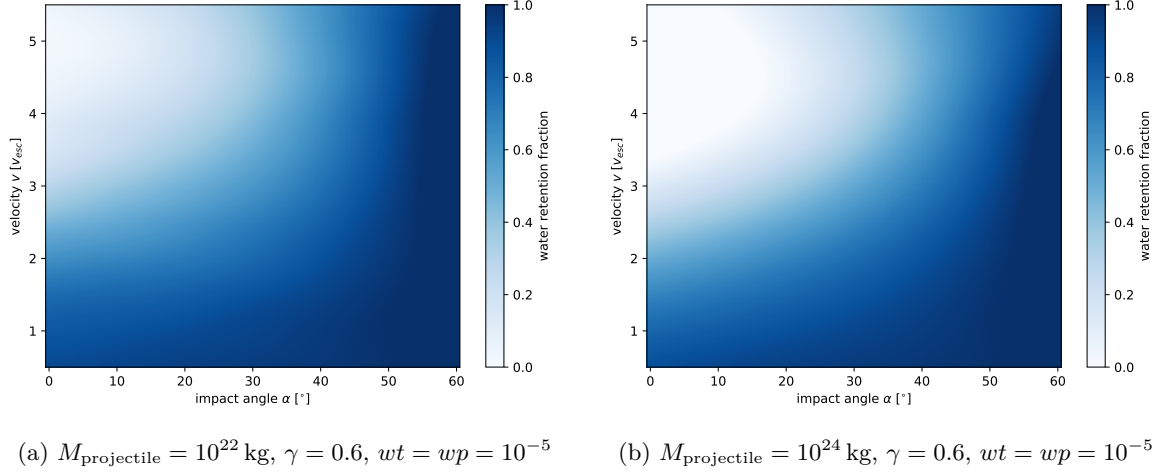


Figure 4.1: Interpolation result using Radial Basis Functions.

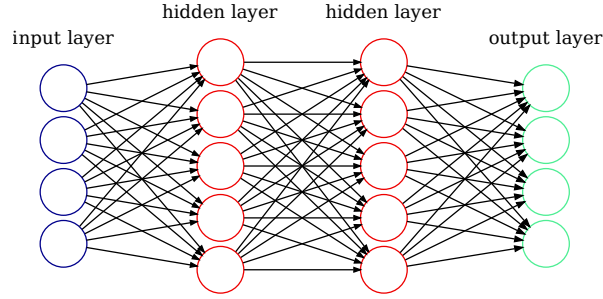


Figure 4.2: An example of a simple Neural Network.

4.4 Neural Network Mass Loss Estimation

Another method to estimate the mass loss is using a really simple *Artificial Neural Network*. This method is a significantly improved version of the one described in Winkler 2019 and uses the exact same dataset as in Section 4.3 for training.

Such a simple Neural Network (Figure 4.2) consists of an input layer (in our case the six input parameters), one or multiple hidden layers and one output layer (consisting of the three mass loss fractions and the fraction between the mass lost in the most massive and the second most massive remnant¹⁶).

Each node z consists of the linear combination of all nodes from the previous layer combined with the weight w of the connection between them. Afterwards, the activation function g is applied to get the prediction $\hat{y}$.

$$z = \sum_i w_i x_i \quad \hat{y} = g(z) \quad (4.9)$$

¹⁶This is not used any further in this thesis, but could be used to distinguish hit-and-run scenarios in the future.

```

1 from torch import nn
2
3 class Network(nn.Module):
4     def __init__(self):
5         super().__init__()
6         self.hidden = nn.Linear(6, 50)
7         self.output = nn.Linear(50, 4)
8
9         self.sigmoid = nn.Sigmoid()
10        self.relu = nn.ReLU()
11
12    def forward(self, x):
13        x = self.hidden(x)
14        x = self.relu(x)
15        x = self.output(x)
16        x = self.sigmoid(x)

```

Code Fragment 4.3: `bac/network.py`

For the activation function g , we are using the rectified linear unit (*ReLU*) $g(x) = \max(0, x)$ (Nair and Hinton 2010) on the hidden layer and the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$ on the output layer, as all predicted values are in the range $[0, 1]$. These non-linear activation functions are needed for the neural network to not just be a linear combination of values, but to also be able to approximate non-linear functions.

During the training of the neural network, this step (the *feedforward*) is followed by the *backpropagation*, where the predictions $\hat{y}$ are compared with the real output y in the training set. The function describing the error is called the Loss function L and the one used in this work is the mean squared error.

$$L(\hat{y}, y) = \sum_i (\hat{y}_i - y_i)^2 \quad (4.10)$$

To train the neural network, `pytorch` (Paszke et al. 2019) is used to describe the network as seen in Code Snippet 4.3. The hidden layer of the network consists of 70 nodes connected to the 6 nodes in the input layer and 4 nodes in the output layer. For training, the existing dataset of collisions was split into a training set containing 80 % of the collisions (7982 entries) and a validation set containing the remaining 1995 entries. After normalizing the dataset as described in Section 4.3, we use the Adam optimizer (Kingma and Ba 2015) to train the dataset for 200 epochs. The output of the loss function on the training dataset and the validation decreases monotonously, as can be seen in Figure 4.3. The final state of the network including the means and standard deviations used by the scaler are then saved into an output file. One theoretical advantage of this method is that unlike the RBF-based method or when creating three separate neural networks the correlations between output values is considered as the training tries to find constants that are able to ‘explain’ all of them at once.

To keep the simulation setup simple, `pytorch` is not used to evaluate the neural network in simulations, but instead the process described above is implemented directly in Python. In addition, to help visualize the parameter space, a second implementation in JavaScript is used to create an interactive evaluation of the network available at <https://mt-nn.lw1.at>.

Figure 4.4 shows the output of the neural network for water mass retention applied to two different projectile masses with the same parameters as in Figure 4.1. Similar to the output of the RBF-based Mass Loss Estimation, a smooth gradient between high and low impact velocities can be seen, even though there are some minor artefacts visible, that are caused by the size of the hidden layer and other hyperparameters. Another way to see the accuracy of this model is by comparing the real data with the output of the model for the training and validation dataset, as can be seen in Figure 4.5.

4 Mass Loss Estimation

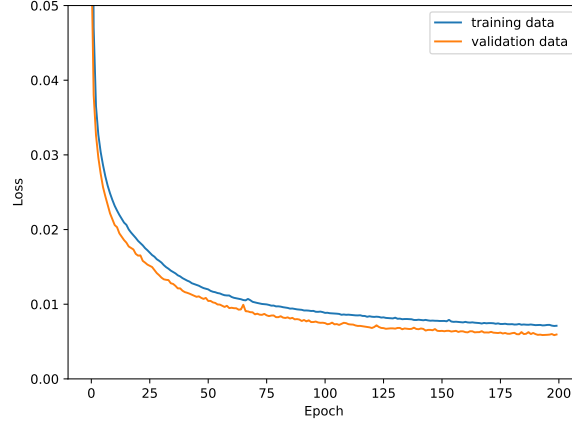


Figure 4.3: Evolution of loss function applied to training and validation dataset during training.

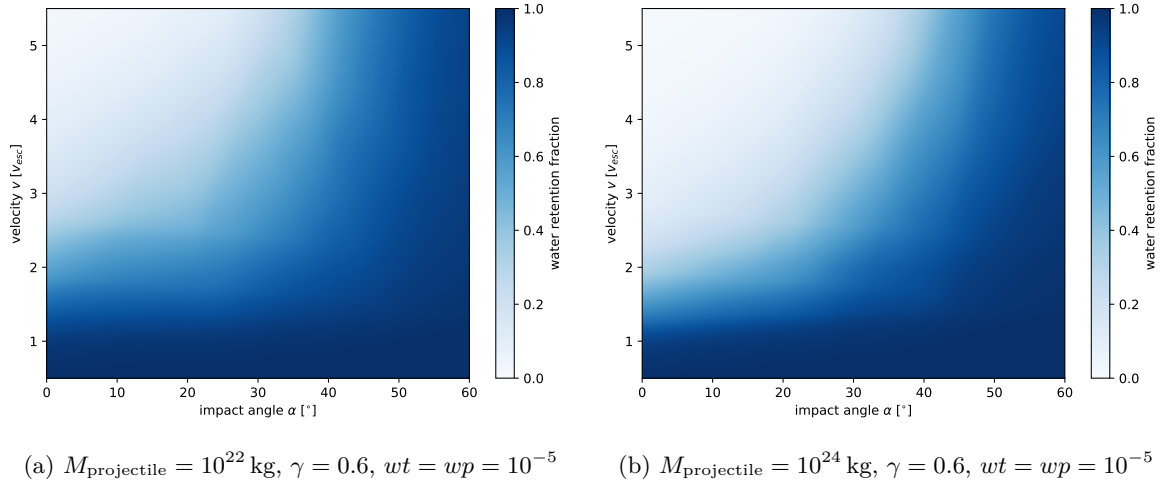


Figure 4.4: Interpolation result using the neural network.

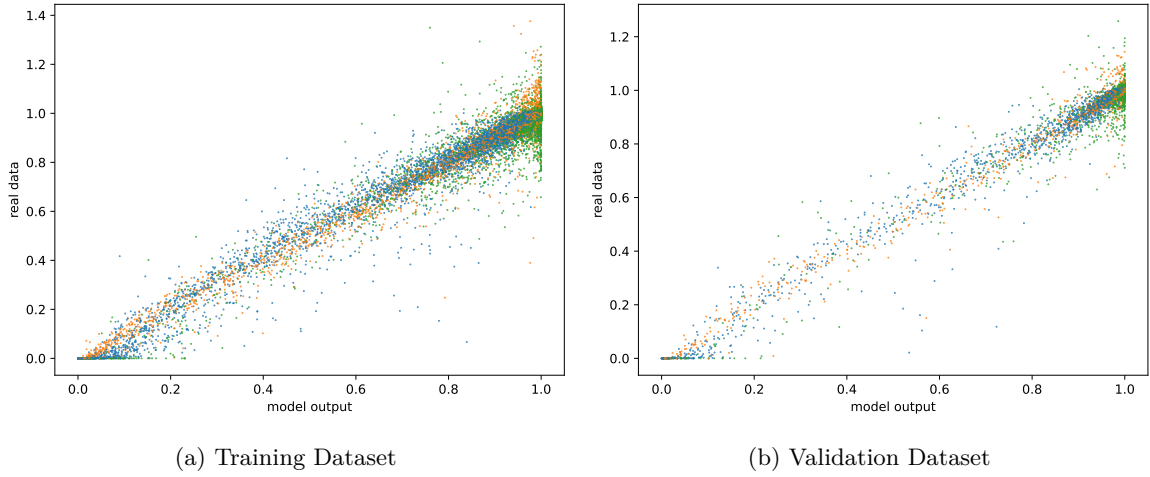


Figure 4.5: Prediction of the neural network for shell (•), mantle (•) and core (•) compared with their actual value in the collision dataset.

5 Results

After running all simulation scenarios described in Section 3.5, we can now analyse the outcomes and compare them to the ones from ChB and ZhL. One of the most straightforward results we can look at to learn more about the simulation outcomes are the properties of the formed bodies that remained at the end of all simulations (Figure 5.1). Figure 5.1a shows the sum of all final systems of the RBF-based simulation runs. The orbital parameters are averaged over the last 10 Myr (corresponding to 1000 simulation saves) to avoid effects by oscillations. Leftover planetesimals that never collided with any other body are left out in all results. The iron core mass is drawn in black, while the colour of total mass of the planets¹⁷ indicates the water mass fraction of the body. For this, a unified colour scale is used throughout this thesis (which was already seen before in Figure 3.1). In addition, the habitable zone around the sun is filled in light grey in the background. For this thesis, just like in ChB, the habitable zone is simplified as the area between 0.75 AU and 1.5 AU which roughly matches the area from Kopparapu et al. 2013. Finally, the real orbital parameters and masses of the inner planets of the solar system are drawn as black circles with the water mass fraction of Earth filled in. The total mass of water on primordial Earth $M_{w,\oplus}$ is assumed to be $1.7 \cdot 10^{21}$ kg (Lécuyer, Gillet and Robert 1998)¹⁸ for this. Unless mentioned otherwise, whenever we look at a single simulation outcome (like in Figure 5.1f), we will pick the one based on the first set of initial conditions to avoid cherry-picking outcomes. Also, it is important to note for all results that all simulations of planetary formation are a stochastic and chaotic process and while the number of simulations per method is a lot larger than in ChB (40, 20 and 18 compared to 5 and 3) the outcomes still only show general trends instead of precise statistical outcomes.¹⁹

Looking at the results in Figure 5.1, we can see that in all scenarios (apart from LZ which will be explained later on), planets with similar properties to the ones in the inner solar system were formed in addition to many others. Also, it is faintly visible that the perfect merging method results in more planets with more water. This is a lot more visible when looking at the average values for the simulation outcomes by mass loss estimation method (Table 5.1 and Table 5.2 for minimum and maximum values).

Using RBF-based mass loss we can see that 1 to 3 planets are formed (N_{planets}), of which 0 to 2 ($N_{\text{planets,pot}}$) are inside the potentially habitable zone. As expected, the largest difference can be seen when comparing the total mass M_{planets} and the total mass of the water M_{water} of these planets. Comparing the RBF dataset with the perfect merging dataset shows that the total mass is about 40 % less and the water mass is about 50 % less. Most of this larger mass loss is of course explained by the mass lost in the collision handling M_{col} , but it seems like part of it is also caused by a larger amount of mass lost in collisions with the sun M_{sun} (Section 3.2.2). Interestingly, both in this thesis and in ChB the amount of mass lost via ejections from the solar system M_{esc} is very similar between the different methods. But between this thesis and ChB, there is a large difference in M_{esc} which might be explained by small differences in the simulation setup and implementation. Compared to this, the mass accreted onto gas giants $M_{\text{gas-giant}}$ is quite negligible.

¹⁷The size of bodies in all plots areas are scaled so that the diameter of the circles scales with $\sqrt[3]{\text{mass}}$ or their area with $\text{mass}^{2/3}$.

¹⁸The exact value is not accurately known as it is not clear how much water is in Earth's mantle. The better known value is the total amount of water in Earth's oceans, which is $(1.335 \pm 0.013) \cdot 10^9 \text{ km}^3 = 1.335 \cdot 10^{21} \text{ kg}$ (Eakins and Sharman 2010).

¹⁹This also means that while some individual results might resemble the planets in the current inner solar system, this is not expected of all simulation outcomes.

5 Results

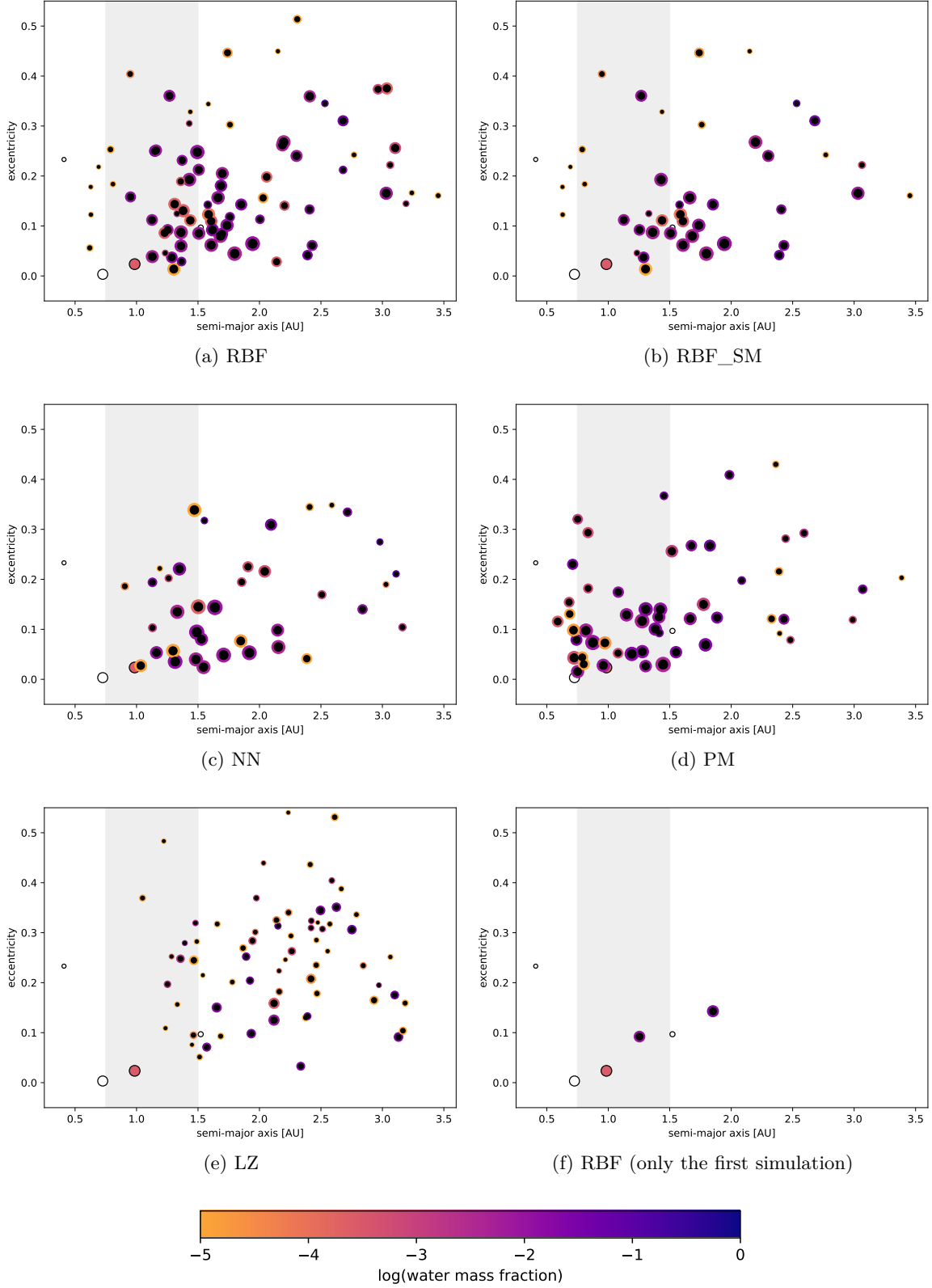


Figure 5.1: All final bodies formed after 200 Myr simulations.

	RBF	RBF_SM	NN	LZ	PM	ChB	ChB PM
N_{planets} [1]	1.9 ± 0.7	2.1 ± 0.7	1.9 ± 0.8	3.8 ± 1.7	2.6 ± 0.6	2.6 ± 0.5	2.7 ± 0.5
$N_{\text{planets,pot}}$ [1]	0.7 ± 0.6	0.8 ± 0.6	0.8 ± 0.4	0.8 ± 0.9	1.1 ± 0.6	0.8 ± 0.4	1.0
M_{planets} [$M_{\oplus}$]	1.7 ± 0.5	1.8 ± 0.4	2.1 ± 0.5	1.0 ± 0.3	2.8 ± 0.4	2.2 ± 0.4	3.6 ± 0.2
$M_{\text{planets,pot}}$ [$M_{\oplus}$]	0.6 ± 0.7	0.6 ± 0.7	1.0 ± 0.9	0.1 ± 0.2	1.6 ± 0.8	1.1 ± 0.6	1.8 ± 0.7
M_{water} [$M_{w,\oplus}$]	44.4 ± 30.8	49.3 ± 32.7	52.2 ± 38.6	27.9 ± 25.1	87.1 ± 49.6	34.3 ± 16.9	132.3 ± 19.6
$M_{\text{water,pot}}$ [$M_{w,\oplus}$]	15.1 ± 23.5	13.0 ± 18.7	21.6 ± 34.4	0.5 ± 1.1	45.9 ± 38.0	25.8 ± 16.7	51.3 ± 46.7
M_{esc} [$M_{\oplus}$]	1.4 ± 0.4	1.4 ± 0.4	1.4 ± 0.4	1.1 ± 0.3	1.5 ± 0.4	0.2 ± 0.1	0.2 ± 0.2
$M_{\text{esc,water}}$ [$M_{w,\oplus}$]	36.9 ± 25.5	35.0 ± 22.6	26.6 ± 16.0	22.2 ± 10.1	37.5 ± 29.7	2.8 ± 2.6	2.8 ± 1.0
M_{sun} [$M_{\oplus}$]	2.4 ± 0.5	2.3 ± 0.5	2.2 ± 0.4	3.0 ± 0.4	1.8 ± 0.3	2.6 ± 0.2	2.2 ± 0.2
$M_{\text{sun,water}}$ [$M_{w,\oplus}$]	192.3 ± 39.5	188.8 ± 40.8	205.1 ± 41.5	233.1 ± 29.9	180.8 ± 43.6	257.0	210.0
$M_{\text{gas-giant}}$ [$M_{\oplus}$]	0.0 ± 0.1	0.0	0.0	0.1 ± 0.2	0.0	-	-
$M_{\text{gas-giant,water}}$ [$M_{w,\oplus}$]	3.9 ± 9.0	2.7 ± 7.0	1.4 ± 1.4	14.2 ± 25.2	1.5 ± 1.3	-	-
M_{col} [$M_{\oplus}$]	0.6 ± 0.7	0.5 ± 0.5	0.7 ± 0.4	0.6 ± 0.1	0.0	1.1 ± 0.3	0.0
$M_{\text{col,water}}$ [$M_{w,\oplus}$]	23.0 ± 11.2	23.3 ± 11.6	22.0 ± 8.9	10.8 ± 3.3	0.0	-	-
$t_{\text{last-col}}$ [Myr]	85.8 ± 45.5	76.3 ± 45.0	94.7 ± 50.4	131.1 ± 36.0	102.8 ± 39.8	295.6 ± 55.5	127.3 ± 34.0

Table 5.1: Aggregated results and their standard deviation for all remaining bodies per mass loss estimation method compared with the results from ChB. ‘pot’ refers to the properties of only bodies inside the potentially habitable zone and ‘water’ to the water mass. $M_{\oplus}$ are Earth masses while $M_{w,\oplus}$ are Earth water masses. A detailed description of all rows can be found in the text on page 23.

N_{planets} [1]	1.0 – 3.0	1.0 – 3.0	1.0 – 4.0	1.0 – 7.0	1.0 – 3.0	2.0 – 3.0	2.0 – 3.0
$N_{\text{planets,pot}}$ [1]	0.0 – 2.0	0.0 – 2.0	0.0 – 1.0	0.0 – 3.0	0.0 – 2.0	0.0 – 1.0	1.0 – 1.0
M_{planets} [$M_{\oplus}$]	0.7 – 3.0	1.2 – 3.0	1.3 – 3.2	0.4 – 1.4	2.1 – 3.4	1.4 – 2.4	3.4 – 3.9
$M_{\text{planets,pot}}$ [$M_{\oplus}$]	0.0 – 1.9	0.0 – 1.8	0.0 – 2.3	0.0 – 0.6	0.0 – 2.7	0.0 – 1.5	0.9 – 2.5
M_{water} [$M_{w,\oplus}$]	0.0 – 105.7	0.0 – 101.8	0.0 – 126.8	0.0 – 66.5	7.0 – 192.1	6.8 – 53.0	117.0 – 160.0
$M_{\text{water,pot}}$ [$M_{w,\oplus}$]	0.0 – 104.0	0.0 – 54.1	0.0 – 96.7	0.0 – 3.0	0.0 – 120.4	0.0 – 50.2	0.0 – 113.0
M_{esc} [$M_{\oplus}$]	0.8 – 2.7	0.8 – 2.0	0.8 – 2.2	0.6 – 1.7	0.8 – 2.4	0.1 – 0.3	0.1 – 0.5
$M_{\text{esc,water}}$ [$M_{w,\oplus}$]	10.4 – 124.5	11.6 – 82.7	14.5 – 68.9	12.4 – 49.6	10.2 – 107.7	0.0 – 6.8	2.1 – 4.3
M_{sun} [$M_{\oplus}$]	1.4 – 3.3	1.4 – 3.2	1.4 – 2.8	2.4 – 3.6	1.1 – 2.4	2.2 – 2.7	2.1 – 2.6
$M_{\text{sun,water}}$ [$M_{w,\oplus}$]	117.4 – 266.6	117.4 – 255.6	140.3 – 287.0	183.7 – 280.1	96.5 – 252.7	-	-
$M_{\text{gas-giant}}$ [$M_{\oplus}$]	0.0 – 0.2	0.0 – 0.2	0.0 – 0.2	0.0 – 0.9	0.0 – 0.1	-	-
$M_{\text{gas-giant,water}}$ [$M_{w,\oplus}$]	0.0 – 37.0	0.0 – 32.0	0.0 – 4.3	0.0 – 75.6	0.0 – 4.3	-	-
M_{col} [$M_{\oplus}$]	0.3 – 4.2	0.3 – 2.7	0.2 – 1.3	0.4 – 0.7	0.0 – 0.0	0.7 – 1.6	0.0 – 0.0
$M_{\text{col,water}}$ [$M_{w,\oplus}$]	5.5 – 51.4	5.5 – 44.8	6.1 – 39.6	6.8 – 17.2	0.0 – 0.0	-	-
$t_{\text{last-col}}$ [Myr]	23.6 – 195.0	23.6 – 186.2	25.4 – 199.0	80.3 – 192.1	46.9 – 196.3	217.0 – 369.0	80.0 – 158.0

Table 5.2: Same as Table 5.1, but showing the minimum and maximum of the range of values.

5 Results

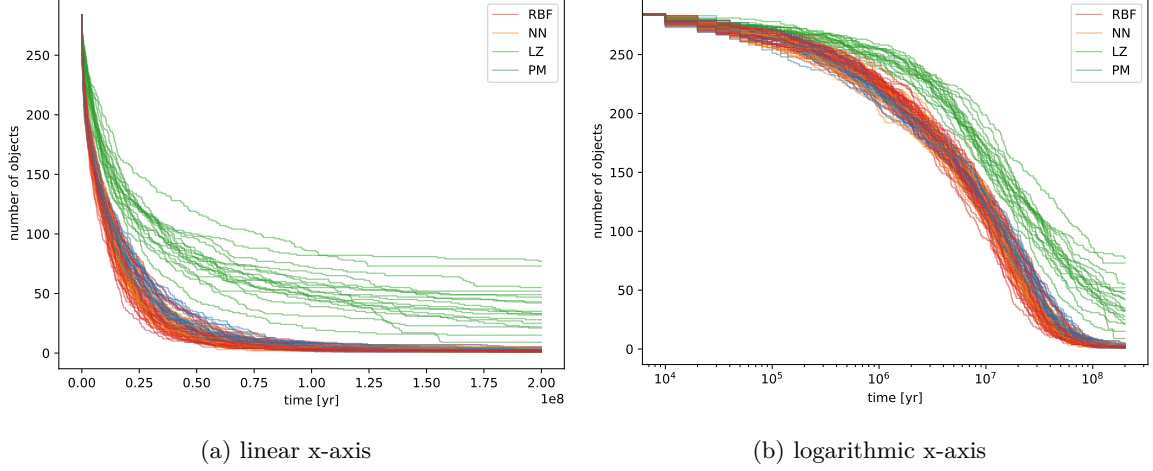


Figure 5.2: Number of bodies in the simulations over time.

Unlike in [ChB](#), we can hardly see any difference in the time of the last collision $t_{\text{last-col}}$. This is even more clearly visible when we take a closer look at the evolution of the number of particles in the simulations. In [Figure 5.2](#), we plot the number of bodies in the simulation against the time. All simulations start with 284 bodies, but as they collide and merge, the number quickly decreases. Apart from the randomized mass loss method (LZ), all simulation in all variants follow nearly the exact same curve. This is expected as hit-and-run collisions are not handled separately unlike in [ChB](#) and therefore any collision causes a reduction of the number of particles in the simulation by one. This reduction of bodies and therefore N^2 -interactions also allows us to finish the simulation in a reasonable time. Quite similarly we can plot the reduction of the total mass²⁰ and water mass over time in [Figure 5.3](#). At a first glance, this looks similar to [Figure 5.2b](#), but there are two major differences. The spread between the final values is a lot higher as, while nearly all simulations end up with only a few remaining particles, the amount of mass lost over time depends on the (chaotic) properties of the collision histories and ejections. Additionally, we can see the influence of mass loss in collisions as the masses are generally higher in the perfect merging simulations.

We can also take a closer look at what happens during individual simulations. [Figure 5.4](#) shows five snapshots at different times in singular simulations based on the first initial conditions. We can see that in the first Myr the eccentricity of all bodies increases strongly and collisions are quite common. At 5 Myr, the number of bodies has already decreased significantly (as also visible in [Figure 5.2b](#)) and larger bodies are starting to form. In the scenarios using more realistic mass loss, water is still mostly in the outer parts of the system. At 20 Myr, this changes with the water mass fraction increasing in the more massive protoplanets due to water-rich collisions. This process continues, as visible in the 50 Myr plot, until only few bodies are remaining and collisions become rarer.

Another way to analyse the outcomes of singular simulation runs is by looking at the remaining planets and then following their collision history back in time, always looking at the more massive parent body. The results for the first simulation with all four methods (which are the same as in [Figure 5.4](#)) can be seen in [Figure 5.5](#). The upper plot traces the mass of the bodies and how it changes through accretion, while the lower plot shows the change of the water mass fraction over time. Remaining bodies that had less than three collisions during their formations are hidden and the water mass of Earth is indicated by a dotted line²¹. We can see that while the mass of the bodies has to always monotonically increase over time when assuming perfect merging, the other methods allow collisions to happen that are strong

²⁰Only mass in planetesimals and planetary embryos is counted.

²¹As mentioned earlier in this section, this value is not to be taken as a precise measurement.

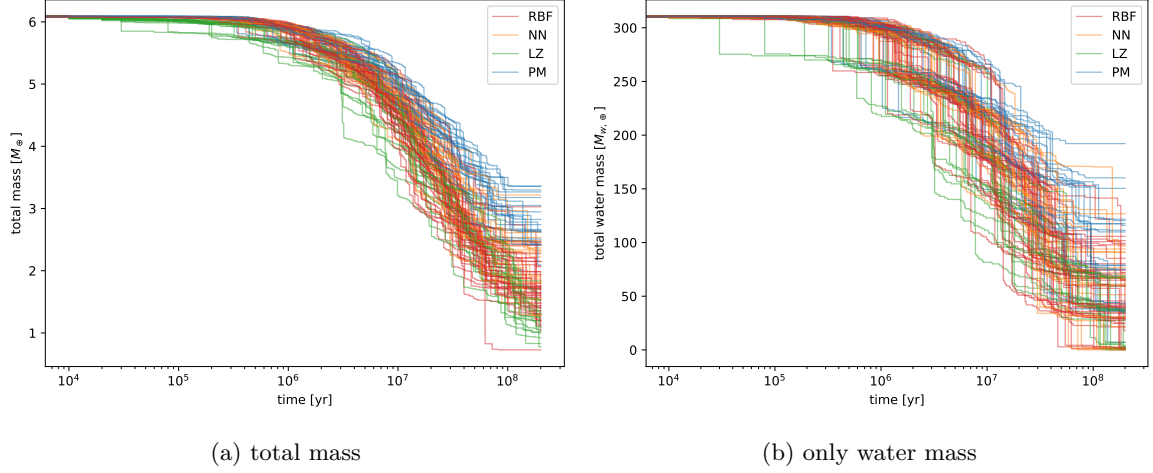


Figure 5.3: Total mass in all planetesimals and planetary embryos over time.

enough that the mass loss is higher than the mass gain through merging two bodies. Because of this the newly formed body has a smaller mass than the two parent bodies each had before the collision.

It is important to note that due to the chaotic processes during planetary formation, the simulation outcomes as shown in Figure 5.1, 5.4 and 5.5 are only examples of possible simulation outcomes. The differences in-between them are not only caused by the difference in mass loss estimation, but mostly by minor differences in the beginning of the simulation, making the chaotic processes result in strongly different final bodies. We can get some more generalized results by overlaying these collision histories for all resulting bodies in all simulations run in Figure 5.6, while keeping in mind that the RBF dataset consists of twice as many simulations. Then we can (once again ignoring LZ for now) for example see that in most cases the mass starts to slowly increase already early on when the first bodies collide after a few ten thousand years and continues to do so until only few bodies remain at around 100 Myr to 150 Myr. Compared to that, it takes a lot longer for the water mass fractions to change, as it takes bodies around 10 Myr to 15 Myr to travel far enough to collide with ones with other water mass fractions (see Figure 3.1 for the initial conditions). One effect that is mentioned in ChB, p. 18 can also be seen here: in many simulations, an early collision delivers water and increases the water mass fraction by a bit to around 0.5 % with a later water-rich-collision delivering the majority of the water mass. Therefore, in many cases, only the later water-rich collision is enough for a water-rich planet to form, but when only the earlier collision occurs, it stays less water-rich. One limitation of this visualisation is that we only follow the history of the more massive progenitor and therefore lose the details of water transport if most water comes from a less massive collision body.

We can also take an even closer look at the properties of a specific formed planet in the first simulation of the RBF dataset. If we only consider the collision history of planet 378 (blue line in Figure 5.5a and inner larger body in the bottom right of Figure 5.4), we can follow the full tree of all bodies whose mass contributed to the final planet and plot it as a graph (Figure 5.7). Planetesimals (using the definition from Section 3.1) are displayed as an ellipse, while planetary embryos are displayed as boxes. Both contain the total mass and water mass fraction of the body and are filled-in according to the same water-mass-fraction colour scheme as used in the rest of the thesis. The diamond-shaped boxes between them indicate collisions and show the fraction of water-shell, mantle and core mass remaining after the collision. Once again, we can see how the majority of the water is accreted in late collisions and that later collisions are more severe, which is also visible in Figure 5.8b.

A completely different view on the influence of the mass loss estimation method on the collision outcomes can be seen by looking at all collisions that happened during the simulations (8429 in total). When

5 Results

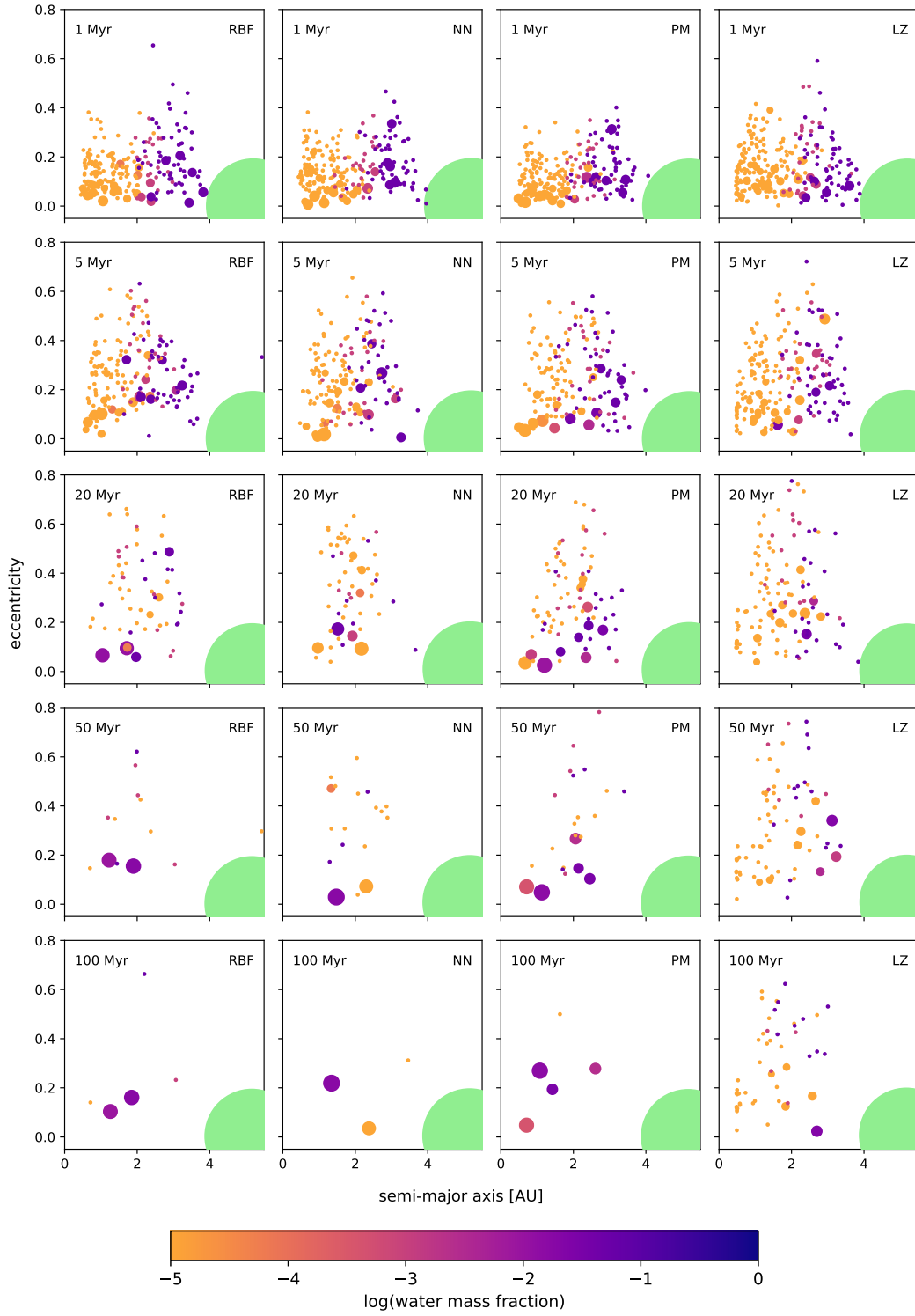


Figure 5.4: Snapshots of four simulation runs at five different times. The size of Jupiter (light green) is scaled according to its mass like with the other bodies.

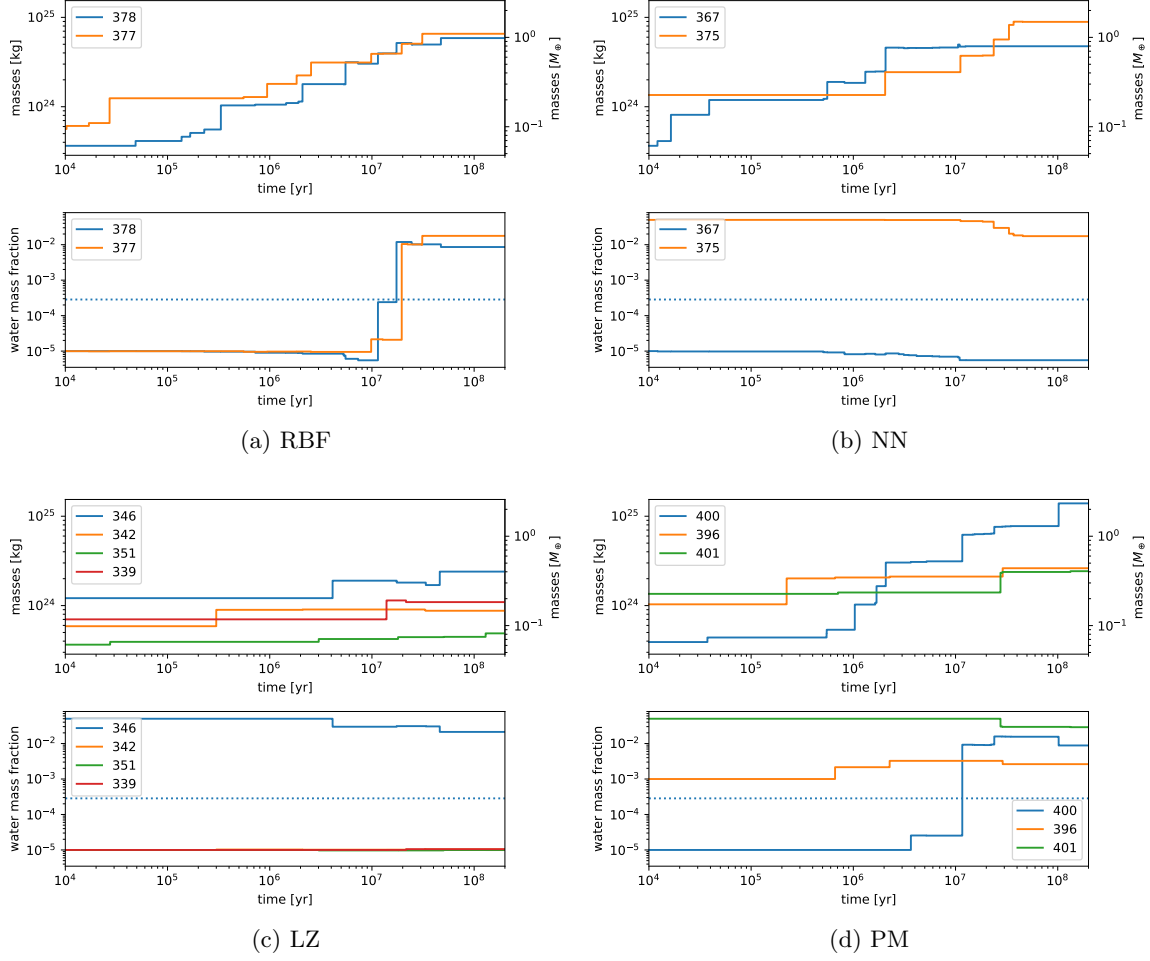


Figure 5.5: Mass and water mass fraction changes of resulting planets over their collision history.

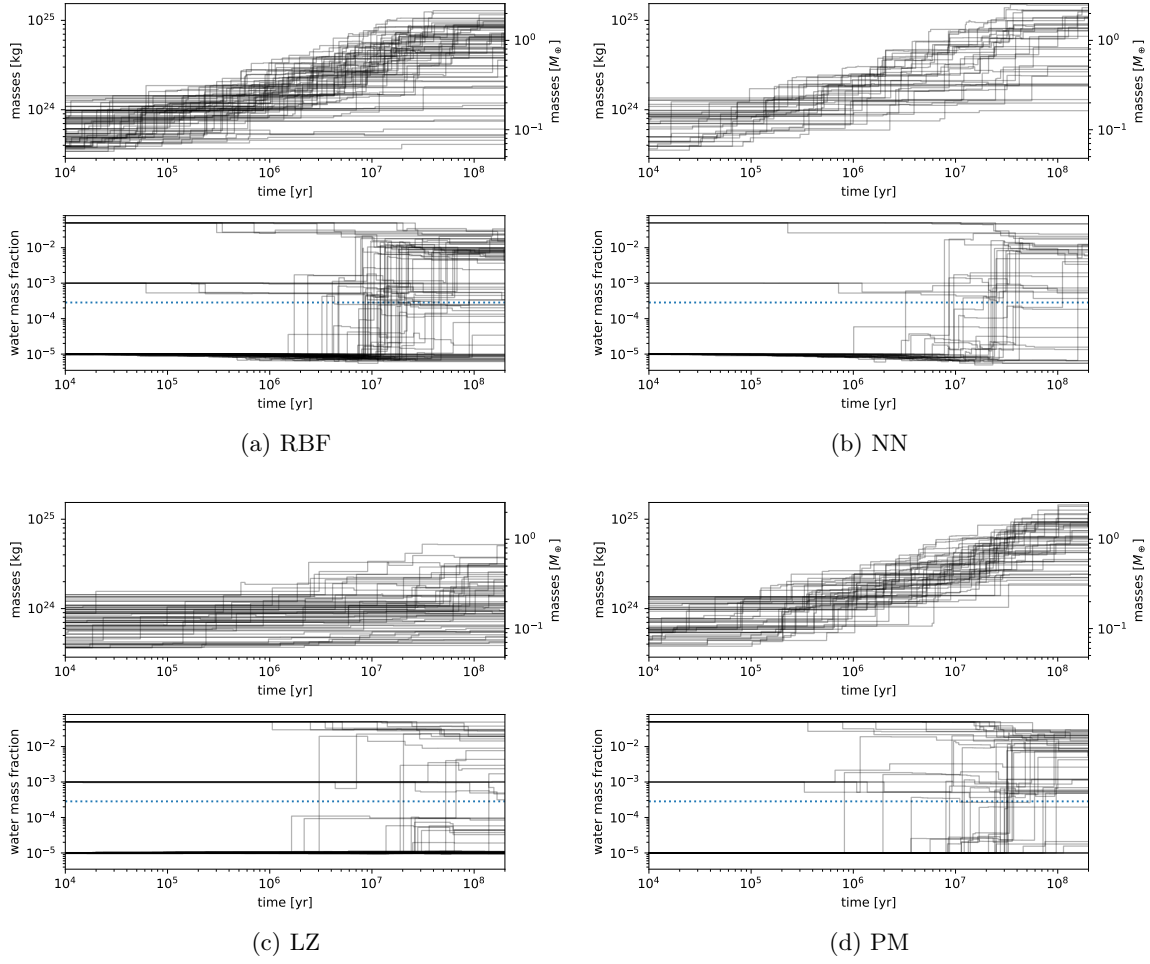


Figure 5.6: Same visualisation as Figure 5.5, but for all simulations of one type.

5 Results

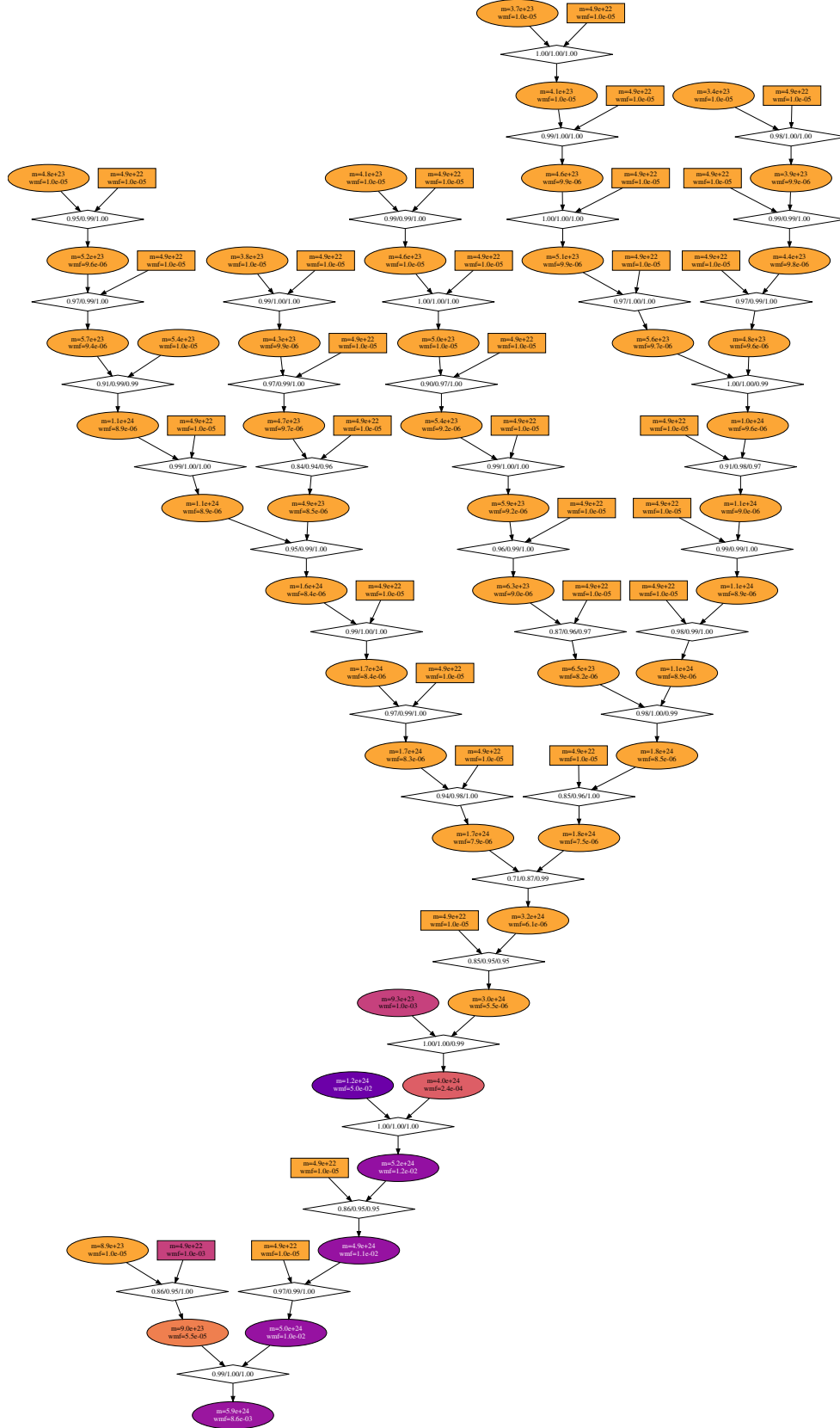


Figure 5.7: Collision graph of one of the final planets in the first RBF simulation.

plotting the impact angle and velocity of the collisions (Figure 5.8a), we can see that while the majority of collisions has a collision velocity close to the escape velocity v_{esc} , there is still a significant amount of stronger collisions. There also does not seem to be any significant difference in the type of collisions that occur in the simulations between the different mass loss estimation methods. Figure 5.8b again plots the collision velocity but this time against the time in the simulation. We can see that at the beginning of the simulation (up to 100 000 years), the velocities of nearly all collisions are rather low as the system starts with very low eccentricities. Only over time and as more and more bodies collide, collisions become more impactful and are able to reach up to 10 times the mutual escape velocity. Once again, the final million years have a significantly lower number of collisions, as the number of bodies becomes rather low. If we now consider the impact of the collisions in the mass loss like we do in the RBF and NN datasets, then we can also see this effect in the amount of water (Figure 5.8c) and iron core mass (Figure 5.8d) lost in each collision. So while hardly any water is lost during the first 100 000 years, water loss in collisions becomes significant later on with common collisions where more than 10 % water mass lost. The same is true for the mantle and core mass loss, but offset to lower values, as collisions impact the outer layers of planets more. As we would expect, the randomized mass loss method in green gives random outcomes in the defined range. We can also analyse the distributions of the impact angle α as defined in Figure 3.3 (Figure 5.8e and 5.8f) and as expected, there is no change in the distribution over time and 45° collisions are most common ($\bar{\alpha} = 45.4^\circ$) with a symmetric decrease towards 0° and 90° . More precisely, the distribution matches $\sin(2x)$ (orange line) very well.

5.1 Randomized Mass Loss

For now we mostly ignored the outcomes of the LZ simulations, which used the randomized mass loss inspired by ZhL as described in Section 4.2. That's because they appear to have unphysically seeming results that differ strongly from both perfect merging and the more realistic mass loss estimation methods introduced in this thesis. We can see in Figure 5.1e, Figure 5.4 and Table 5.1 that the final bodies are significantly smaller and simulations finish with significantly more final planets. Figure 5.2a and 5.6c also show that collisions occur either much slower to the point where up to a third of all bodies still exist after 100 Myr. But even in the following 100 Myr the number does not decrease much further, indicating that the process is not just slowed down, but halted.

The reason for this seems to be two processes that can be seen when looking at the properties of all collisions over time. Figure 5.8c shows how unlike in every other method the amount of water mass lost in collisions cannot slowly increase over time, but is per definition between 1 % and 10 %. And this means that with a bit of bad luck a very early collision can already cause a large amount of mass loss. This is made worse by the fact that, as visible in Figure 5.8d, the difference is even larger in the solid mass loss in mantle and core. The randomized range of 1 % to 8 % (which was not intended for simulating the early solar system, but 55 Cancri in ZhL) is far higher than the mass loss that occurs in the other methods where it stays below 1 % for a long time. In combination, these two effects mean that early collisions between bodies can already be strong enough that mass growth is hindered and the decreased mass reduces the chances that bodies hit each other again later causing too many small bodies to remain.

The latter issue could be solved by picking a range for the randomized mass loss that closer resembles the loss that seem to occur based on the SPH-based collision modelling. But this still leaves the issue that the mass loss in early collisions is overestimated. One could consider adding a dependency on the time in the simulation to fix this, but at this point one might also need to add a dependency with the impact angle and velocity to not accidentally estimate a large amount of mass loss in a weak collision. And this would bring the complexity of the implementation to a similar level as the other methods introduced in this paper which are also very computationally inexpensive.

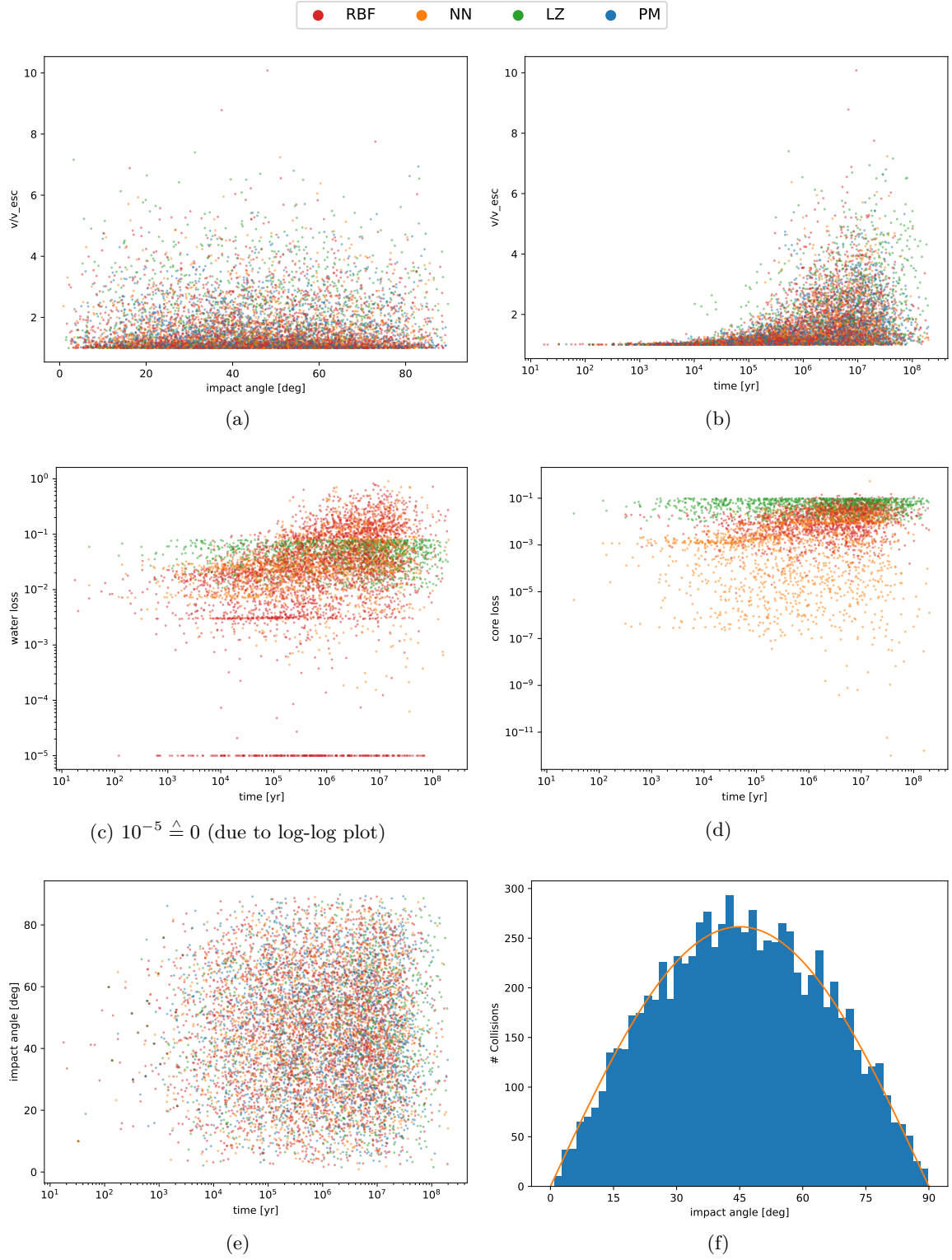


Figure 5.8: Statistics over all collisions that occurred in the simulations.

6 Discussion and Conclusions

As mentioned in the introduction, water transport is an important process during the late stages of terrestrial planetary formation. It determines the amount of water that can be found on the planets in a solar system, and we expect it to be strongly affected by the details of how we model the loss of water in the collisions of protoplanetary embryos and planetesimals. When looking at the results of our simulation setup, we can confirm the primary result of ChB, namely that assuming no water or other mass will be lost in collisions and that bodies are merged perfectly is a large oversimplification. It causes planets to be more massive and water rich and slightly more common. When looking at the exact values of Tables 5.1 and 5.2 to compare our final parameters of the solar systems with the ones from ChB, we can see that while the results in the perfect merging cases differ in our simulations, possibly because of the slightly different simulation setup, the influence of the mass loss estimation methods introduced in this thesis on the simulation outcomes is rather similar. We can also reproduce other effects, like the fact that most planets first accrete their mass while staying dry until one or a few major collisions deliver a large fraction of the final water content (Figure 5.5). Our model also correctly displays the fact that early collisions are weaker and have far less water loss than collisions in the final stages of planetary formation (Figure 5.8c).

As we are not comparing multiple N-Body setups, we can afford to run more simulations per method (40, 20 or 18 depending on the method compared to 5 in ChB). As the outcomes of the N-Body simulations are always strongly affected by the chaotic effects of gravitational interactions, this allows to see the influence of the specific mass loss estimation methods more clearly in comparison. Nevertheless, the standard deviations of all simulation outcomes in Table 5.1 are consistently higher in our work. One possible explanation for this is that there is no ‘true value’ for e.g. M_{planets} that we would converge to if we ran orders of magnitude more simulations. Instead, both the large influence minor differences in singular collision events can have on the outcome and the fact that our initial conditions are randomized mean that there is a wider range of possible formed solar systems at the end of our simulations. Additionally, if we only run a very small amount of simulations, the standard deviation might just appear smaller, as we are unable to cover the full extent of possible formed solar systems. This can be seen when comparing the RBF results with the ones from RMF_SM, which is just the first 20 simulations of RBF.

In contrast, the simulation setup of this thesis has a major limitation compared to ChB: Every single collision between two bodies results in them merging and creating a new body (even if we consider the mass lost during this process). But with an impact angle high enough, it might be possible that both bodies lose mass without merging, but rather continue their trajectory independently. These Hit-and-Run collisions are not unlikely, but slow down the formation process when considered properly as they don’t reduce the number of collisions. In ChB, this causes the accretion time to be nearly doubled while in our results the speed of accretion is hardly affected. This means that while the final simulation outcome might be comparable, the time at which it is reached is far too early. In a future work, our approach could be extended to also correctly handle Hit-and-Run scenarios. For this, the mass ratio between the two most massive bodies after the collision would also need to be estimated (the work for this is already started in Section 4.4). We would also need to find a way to estimate the orbital parameters of the two resulting bodies (potentially similar to Emsenhuber et al. 2020), as just using conservation of momentum is no longer enough to define their resulting velocities. This could be brought even further by including the results of the ongoing work of Winter et al. 2022, in which more complex neural networks are used to not just estimate the output from the input, but also model the whole process in-between more accurately.

Still, the approach of estimating collision outcomes based on existing datasets has the major advantage of being easier to implement and faster by orders of magnitude to run per collision (even though this might not be as relevant considering the very long run times of the N-Body simulation itself). Also, it does not depend on the availability of GPUs which are needed to run the SPH-based collision simulations using

miluphcuda. The exact method of estimation (linear interpolation or simple neural network) does not really matter in comparison to the much larger sources of uncertainties in the N-Body simulation and SPH setup, as long as they are able to reproduce the original dataset in all input and output variables.

6.1 Data Availability

Most code used in this thesis (both for simulations and analysis of results) is available at <https://lw1.at/r/masters-thesis-code> and the code related to reading the existing dataset of collisions at <https://lw1.at/r/masters-thesis-dataset-code>. The raw data from all simulations is available on request from the author.

6.2 Conclusions

Late stage formation of terrestrial planets is still a topic with many unknown details, in which many teams try to describe the process as accurately as possible with more and more detailed simulations. In this thesis, we tried to focus on one particular aspect of this process, the influence of mass and especially water mass loss during collisions on the formed solar systems. While many existing simulations simply assume that all mass of two protoplanetary bodies colliding with each other is merged into the resulting body, higher resolution, SPH-based simulations of individual collisions show that this is only true for low velocity, low impact angle collisions. As we can reproduce in this thesis, these collisions are only common in the early stages of simulations with collisions becoming more and more impactful over time. To make the collision handling more accurate, we need to include the additional knowledge about collision outcomes into our N-Body simulation of the planetary formation. For this, we created an alternative method to ChB which uses a dataset of about 10 000 individually resolved collisions from ChB to estimate the mass loss in our collisions based on their properties. Estimation methods based on linear interpolation and artificial neural networks are compared with a randomized mass loss method inspired by ZhL and perfect merging. Using each method, 18–40 simulations are run with initial conditions corresponding to the late stage of the formation of the solar system. The main results of these simulations are:

- We can reproduce the fact that a more accurate handling of mass loss in collisions has a strong influence on the final solar system. Planets are less massive, contain less water and are fewer in numbers (both inside the potentially habitable zone and in general).
- While the number of bodies decreases and their mass increases strongly in the first 10 Myr, it takes longer for water to be transported into the inner parts of the solar system. Related, most water is accreted onto the final planets in a small amount of collisions with water rich protoplanetary embryos.
- Increasing the number of simulations per method helps limit the influence of the chaotic nature of N-Body simulations and allows giving more precise results.
- Using estimations for the mass loss based on existing SPH-based simulations instead of running a dedicated simulation for every collision like in ChB seems to give quite comparable results independently of the exact estimation method.
- The lack of consideration for Hit-and-Run collision scenarios remains a major limitation of this work and results in too fast accretion, as every collision reduces the number of bodies in the system.
- A modern hybrid integrator like MERCURIUS works very well in this scenario with a long simulation time where collisions still need to be resolved in a high resolution.

6 *Discussion and Conclusions*

- A randomized mass loss similar to [ZhL](#) can work, but it is important to use the correct distribution for the scenario and take the time of collision into consideration as otherwise too much mass is lost very early, making later processes impossible.

List of Figures

2.1	A snapshot of an SPH simulation like the ones used here. The colour indicates the velocity in the direction of the collision.	8
2.2	The Pearson correlation coefficient between the input parameters and the water mass fraction.	8
3.1	Minor bodies in first set of initial conditions (see Section 3.5).	10
3.2	Visualization of a collision with $\alpha = 20.29^\circ$, $\frac{v}{v_{esc}} = 1.02$ and $\gamma = 0.14$	14
3.3	Visualization of the collision angle α (here $\alpha = 60^\circ$), based on Figure 1 in ChB.	15
4.1	Interpolation result using Radial Basis Functions.	20
4.2	An example of a simple Neural Network.	20
4.3	Evolution of loss function applied to training and validation dataset during training. . .	22
4.4	Interpolation result using the neural network.	22
4.5	Prediction of the neural network for shell ($\bullet$), mantle ($\bullet$) and core ($\bullet$) compared with their actual value in the collision dataset.	22
5.1	All final bodies formed after 200 Myr simulations.	24
5.2	Number of bodies in the simulations over time.	26
5.3	Total mass in all planetesimals and planetary embryos over time.	27
5.4	Snapshots of four simulation runs at five different times. The size of Jupiter (light green) is scaled according to its mass like with the other bodies.	28
5.5	Mass and water mass fraction changes of resulting planets over their collision history. . .	29
5.6	Same visualisation as Figure 5.5, but for all simulations of one type.	30
5.7	Collision graph of one of the final planets in the first RBF simulation.	31
5.8	Statistics over all collisions that occurred in the simulations.	33

List of Tables

5.1	Aggregated results and their standard deviation for all remaining bodies per mass loss estimation method compared with the results from ChB. ‘pot’ refers to the properties of only bodies inside the potentially habitable zone and ‘water’ to the water mass. $M_\oplus$ are Earth masses while $M_{w,\oplus}$ are Earth water masses. A detailed description of all rows can be found in the text on page 23.	25
5.2	Same as Table 5.1, but showing the minimum and maximum of the range of values.	25

List of Code Fragments

3.1	initcon/InitCondGen.py:286-299 (simplified)	10
3.2	REBOUND: main/src/integrator_mercurius.c:416-424 (1074be9583)	11
3.3	REBOUND: main/src/integrator_mercurius.c:44-54 (1074be9583)	11
3.4	heartbeat/heartbeat.c (simplified)	12
3.5	utils/radius.py (simplified)	13
4.1	massloss/perfect_merging.py:6-10	17
4.2	massloss/rbf_massloss.py (simplified)	19
4.3	bac/network.py	21

Abbreviations

- ChB C. Burger, Á. Bazsó and C. M. Schäfer (Feb. 2020). ‘Realistic collisional water transport during terrestrial planet formation. Self-consistent modeling by an N-body-SPH hybrid code’. In: *A&A* 634, A76, A76. DOI: [10.1051/0004-6361/201936366](https://doi.org/10.1051/0004-6361/201936366). arXiv: [1910.14334](https://arxiv.org/abs/1910.14334) [astro-ph.EP].
- ZhL L. Zhou, R. Dvorak and L.-Y. Zhou (Aug. 2021). ‘On the formation of terrestrial planets between two massive planets: the case of 55 Cancri’. In: *MNRAS* 505.3, pp. 4571–4585. DOI: [10.1093/mnras/stab1534](https://doi.org/10.1093/mnras/stab1534). arXiv: [2105.10105](https://arxiv.org/abs/2105.10105) [astro-ph.EP].

Bibliography

- Benz, W. and E. Asphaug (Nov. 1999). ‘Catastrophic Disruptions Revisited’. In: *Icarus* 142.1, pp. 5–20. DOI: [10.1006/icar.1999.6204](https://doi.org/10.1006/icar.1999.6204). arXiv: [astro-ph/9907117](https://arxiv.org/abs/astro-ph/9907117) [astro-ph] (cit. on p. 7).
- Burger, C., Á. Bazsó and C. M. Schäfer (Feb. 2020). ‘Realistic collisional water transport during terrestrial planet formation. Self-consistent modeling by an N-body-SPH hybrid code’. In: *A&A* 634, A76, A76. DOI: [10.1051/0004-6361/201936366](https://doi.org/10.1051/0004-6361/201936366). arXiv: [1910.14334](https://arxiv.org/abs/1910.14334) [astro-ph.EP] (cit. on pp. 6, 7, 9–11, 15, 16, 18, 23, 25–27, 34, 35).
- Burger, C., T. I. Maindl and C. M. Schäfer (Jan. 2018). ‘Transfer, loss and physical processing of water in hit-and-run collisions of planetary embryos’. In: *Celestial Mechanics and Dynamical Astronomy* 130.1, 2, p. 2. DOI: [10.1007/s10569-017-9795-3](https://doi.org/10.1007/s10569-017-9795-3). arXiv: [1710.03669](https://arxiv.org/abs/1710.03669) [astro-ph.EP] (cit. on pp. 7, 13).
- Cambioni, S., E. Asphaug et al. (Apr. 2019). ‘Realistic On-the-fly Outcomes of Planetary Collisions: Machine Learning Applied to Simulations of Giant Impacts’. In: *ApJ* 875.1, 40, p. 40. DOI: [10.3847/1538-4357/ab0e8a](https://doi.org/10.3847/1538-4357/ab0e8a). arXiv: [1903.04507](https://arxiv.org/abs/1903.04507) [astro-ph.EP] (cit. on p. 5).
- Cambioni, S., S. A. Jacobson et al. (June 2021). ‘The Effect of Inefficient Accretion on Planetary Differentiation’. In: *The Planetary Science Journal* 2.3, 93, p. 93. DOI: [10.3847/PSJ/abf0ad](https://doi.org/10.3847/PSJ/abf0ad). arXiv: [2106.07680](https://arxiv.org/abs/2106.07680) [astro-ph.EP] (cit. on p. 6).
- Chambers, J. E. (Apr. 1999). ‘A hybrid symplectic integrator that permits close encounters between massive bodies’. In: *MNRAS* 304.4, pp. 793–799. DOI: [10.1046/j.1365-8711.1999.02379.x](https://doi.org/10.1046/j.1365-8711.1999.02379.x) (cit. on pp. 5, 10, 11).
- Du Toit, W. (Mar. 2008). ‘Radial basis function interpolation’. MA thesis. Stellenbosch: Stellenbosch University. URL: <https://scholar.sun.ac.za/handle/10019.1/2002> (cit. on p. 18).
- Eakins, B. W. and G. F. Sharman (2010). *Volumes of the World’s Oceans from ETOPO1*. NOAA National Geophysical Data Center, Boulder, CO. URL: https://www.ngdc.noaa.gov/mgg/global/etopo1_ocean_volumes.html (visited on 01/01/2022) (cit. on p. 23).
- Emsenhuber, A. et al. (Mar. 2020). ‘Realistic On-the-fly Outcomes of Planetary Collisions. II. Bringing Machine Learning to N-body Simulations’. In: *ApJ* 891.1, 6, p. 6. DOI: [10.3847/1538-4357/ab6de5](https://doi.org/10.3847/1538-4357/ab6de5). arXiv: [2001.00951](https://arxiv.org/abs/2001.00951) [astro-ph.EP] (cit. on pp. 5, 34).
- Haghighipour, N. and T. I. Maindl (Jan. 2022). ‘Building Terrestrial Planets: Why results of perfect-merging simulations are not quantitatively reliable approximations to accurate modeling of terrestrial planet formation’. In: *arXiv e-prints*. arXiv: [2201.06702](https://arxiv.org/abs/2201.06702) [astro-ph.EP]. Pre-published (cit. on p. 6).

- Hayashi, C. (Jan. 1981). ‘Structure of the Solar Nebula, Growth and Decay of Magnetic Fields and Effects of Magnetic and Turbulent Viscosities on the Nebula’. In: *Progress of Theoretical Physics Supplement* 70, pp. 35–53. DOI: [10.1143/PTPS.70.35](https://doi.org/10.1143/PTPS.70.35) (cit. on p. 5).
- Izidoro, A., A. Morbidelli and S. N. Raymond (Oct. 2014). ‘Terrestrial Planet Formation in the Presence of Migrating Super-Earths’. In: *ApJ* 794.1, 11, p. 11. DOI: [10.1088/0004-637X/794/1/11](https://doi.org/10.1088/0004-637X/794/1/11). arXiv: [1408.1215](https://arxiv.org/abs/1408.1215) [astro-ph.EP] (cit. on p. 5).
- Kingma, D. P. and J. Ba (2015). ‘Adam: A Method for Stochastic Optimization’. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. arXiv: [1412.6980](https://arxiv.org/abs/1412.6980) (cit. on p. 21).
- Kopparapu, R. K. et al. (Mar. 2013). ‘Habitable Zones around Main-sequence Stars: New Estimates’. In: *ApJ* 765.2, 131, p. 131. DOI: [10.1088/0004-637X/765/2/131](https://doi.org/10.1088/0004-637X/765/2/131). arXiv: [1301.6674](https://arxiv.org/abs/1301.6674) [astro-ph.EP] (cit. on p. 23).
- Lécuyer, C., P. Gillet and F. Robert (1998). ‘The hydrogen isotope composition of seawater and the global water cycle’. In: *Chemical Geology* 145.3, pp. 249–261. ISSN: 0009-2541. DOI: [https://doi.org/10.1016/S0009-2541\(97\)00146-0](https://doi.org/10.1016/S0009-2541(97)00146-0) (cit. on p. 23).
- Leinhardt, Z. M. and S. T. Stewart (Jan. 2012). ‘Collisions between Gravity-dominated Bodies. I. Outcome Regimes and Scaling Laws’. In: *ApJ* 745.1, 79, p. 79. DOI: [10.1088/0004-637X/745/1/79](https://doi.org/10.1088/0004-637X/745/1/79). arXiv: [1106.6084](https://arxiv.org/abs/1106.6084) [astro-ph.EP] (cit. on p. 5).
- Martin, R. G. and M. Livio (Sept. 2012). ‘On the evolution of the snow line in protoplanetary discs’. In: *MNRAS* 425.1, pp. L6–L9. DOI: [10.1111/j.1745-3933.2012.01290.x](https://doi.org/10.1111/j.1745-3933.2012.01290.x). arXiv: [1207.4284](https://arxiv.org/abs/1207.4284) [astro-ph.EP] (cit. on p. 5).
- Melosh, H. J. (1989). *Impact Cratering: A Geologic Process*. New York: Oxford University Press. ISBN: 9780195042849 (cit. on p. 7).
- Morbidelli, A. et al. (Nov. 2000). ‘Source regions and time scales for the delivery of water to Earth’. In: *Meteoritics & Planetary Science* 35.6, pp. 1309–1320. DOI: [10.1111/j.1945-5100.2000.tb01518.x](https://doi.org/10.1111/j.1945-5100.2000.tb01518.x) (cit. on p. 5).
- Mustill, A. J., M. B. Davies and A. Johansen (Aug. 2018). ‘The dynamical evolution of transiting planetary systems including a realistic collision prescription’. In: *MNRAS* 478.3, pp. 2896–2908. DOI: [10.1093/mnras/sty1273](https://doi.org/10.1093/mnras/sty1273). arXiv: [1708.08939](https://arxiv.org/abs/1708.08939) [astro-ph.EP] (cit. on p. 5).
- Nair, V. and G. E. Hinton (2010). ‘Rectified Linear Units Improve Restricted Boltzmann Machines’. In: *Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML’10*. Haifa, Israel: Omnipress, pp. 807–814. ISBN: 9781605589077. URL: <https://dl.acm.org/doi/10.5555/3104322.3104425> (cit. on p. 21).
- O’Brien, D. P., A. Morbidelli and H. F. Levison (Sept. 2006). ‘Terrestrial planet formation with strong dynamical friction’. In: *Icarus* 184.1, pp. 39–58. DOI: [10.1016/j.icarus.2006.04.005](https://doi.org/10.1016/j.icarus.2006.04.005) (cit. on p. 5).
- Oró, J. (1961). ‘Comets and the formation of biochemical compounds on the primitive Earth’. In: *Nature* 190.4774, pp. 389–390 (cit. on p. 5).
- Owen, T. C. and A. Bar-Nun (Aug. 2001). ‘Contributions of Icy Planetesimals to the Earth’s Early Atmosphere’. In: *Origins of Life and Evolution of the Biosphere* 31, pp. 435–458. DOI: [10.1023/A:1011809412925](https://doi.org/10.1023/A:1011809412925) (cit. on p. 5).
- Paszke, A. et al. (2019). ‘PyTorch: An Imperative Style, High-Performance Deep Learning Library’. In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., pp. 8024–8035. URL: <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf> (cit. on p. 21).
- Podolak, M. and S. Zucker (Nov. 2004). ‘A note on the snow line in protostellar accretion disks’. In: *Meteoritics & Planetary Science* 39.11, pp. 1859–1868. DOI: [10.1111/j.1945-5100.2004.tb00081.x](https://doi.org/10.1111/j.1945-5100.2004.tb00081.x) (cit. on p. 5).
- Raymond, S. N., T. Quinn and J. I. Lunine (Mar. 2004). ‘Making other earths: dynamical simulations of terrestrial planet formation and water delivery’. In: *Icarus* 168.1, pp. 1–17. DOI: [10.1016/j.icarus.2003.11.019](https://doi.org/10.1016/j.icarus.2003.11.019). arXiv: [astro-ph/0308159](https://arxiv.org/abs/astro-ph/0308159) [astro-ph] (cit. on p. 9).
- (Aug. 2006). ‘High-resolution simulations of the final assembly of Earth-like planets I. Terrestrial accretion and dynamics’. In: *Icarus* 183.2, pp. 265–282. DOI: [10.1016/j.icarus.2006.03.011](https://doi.org/10.1016/j.icarus.2006.03.011). arXiv: [astro-ph/0510284](https://arxiv.org/abs/astro-ph/0510284) [astro-ph] (cit. on p. 5).

- Rein, H., D. M. Hernandez et al. (June 2019). ‘Hybrid symplectic integrators for planetary dynamics’. In: *MNRAS* 485.4, pp. 5490–5497. DOI: [10.1093/mnras/stz769](https://doi.org/10.1093/mnras/stz769). arXiv: [1903.04972](https://arxiv.org/abs/1903.04972) [astro-ph.EP] (cit. on pp. 10, 11).
- Rein, H. and S.-F. Liu (Jan. 2012). ‘REBOUND: an open-source multi-purpose N-body code for collisional dynamics’. In: *A&A* 537, A128, A128. DOI: [10.1051/0004-6361/201118085](https://doi.org/10.1051/0004-6361/201118085). arXiv: [1110.4876](https://arxiv.org/abs/1110.4876) [astro-ph.EP] (cit. on pp. 6, 10).
- Rein, H. and D. S. Spiegel (Jan. 2015). ‘IAS15: a fast, adaptive, high-order integrator for gravitational dynamics, accurate to machine precision over a billion orbits’. In: *MNRAS* 446.2, pp. 1424–1437. DOI: [10.1093/mnras/stu2164](https://doi.org/10.1093/mnras/stu2164). arXiv: [1409.4779](https://arxiv.org/abs/1409.4779) [astro-ph.EP] (cit. on p. 10).
- Rein, H. and D. Tamayo (Sept. 2015). ‘WHFAST: a fast and unbiased implementation of a symplectic Wisdom-Holman integrator for long-term gravitational simulations’. In: *MNRAS* 452.1, pp. 376–388. DOI: [10.1093/mnras/stv1257](https://doi.org/10.1093/mnras/stv1257). arXiv: [1506.01084](https://arxiv.org/abs/1506.01084) [astro-ph.EP] (cit. on p. 10).
- (May 2017). ‘A new paradigm for reproducing and analyzing N-body simulations of planetary systems’. In: *MNRAS* 467.2, pp. 2377–2383. DOI: [10.1093/mnras/stx232](https://doi.org/10.1093/mnras/stx232). arXiv: [1701.07423](https://arxiv.org/abs/1701.07423) [astro-ph.EP] (cit. on p. 15).
- Rubin, M. et al. (Apr. 2015). ‘Molecular nitrogen in comet 67P/Churyumov-Gerasimenko indicates a low formation temperature’. In: *Science* 348.6231, pp. 232–235. DOI: [10.1126/science.aaa6100](https://doi.org/10.1126/science.aaa6100) (cit. on p. 5).
- Schäfer, C. M., S. Riecker et al. (May 2016). ‘A smooth particle hydrodynamics code to model collisions between solid, self-gravitating objects’. In: *A&A* 590, A19, A19. DOI: [10.1051/0004-6361/201528060](https://doi.org/10.1051/0004-6361/201528060). arXiv: [1604.03290](https://arxiv.org/abs/1604.03290) [astro-ph.EP] (cit. on p. 7).
- Schäfer, C. M., O. J. Wandel et al. (2020). ‘A versatile smoothed particle hydrodynamics code for graphic cards’. In: *Astronomy and Computing* 33, p. 100410. ISSN: 2213-1337. DOI: [10.1016/j.ascom.2020.100410](https://doi.org/10.1016/j.ascom.2020.100410). URL: <https://www.sciencedirect.com/science/article/pii/S2213133720300640> (cit. on p. 7).
- Stadel, J. G. (Jan. 2001). ‘Cosmological N-body simulations and their analysis’. PhD thesis. University of Washington, United States. URL: <http://carma.astro.umd.edu/nemo/pitp/papers/Stadel2001-thesis.ps> (cit. on p. 5).
- Stewart, S. T. and Z. M. Leinhardt (May 2012). ‘Collisions between Gravity-dominated Bodies. II. The Diversity of Impact Outcomes during the End Stage of Planet Formation’. In: *ApJ* 751.1, 32, p. 32. DOI: [10.1088/0004-637X/751/1/32](https://doi.org/10.1088/0004-637X/751/1/32). arXiv: [1109.4588](https://arxiv.org/abs/1109.4588) [astro-ph.EP] (cit. on p. 5).
- Tsiganis, K. et al. (May 2005). ‘Origin of the orbital architecture of the giant planets of the Solar System’. In: *Nature* 435.7041, pp. 459–461. DOI: [10.1038/nature03539](https://doi.org/10.1038/nature03539) (cit. on p. 10).
- Virtanen, P. et al. (2020). ‘SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python’. In: *Nature Methods* 17, pp. 261–272. DOI: [10.1038/s41592-019-0686-2](https://doi.org/10.1038/s41592-019-0686-2) (cit. on p. 19).
- Winkler, L. (1st Sept. 2019). ‘Interpolated water retention after two-body collisions using Neural Networks and linear interpolation methods’. English. BA thesis. University of Vienna. 19 pp. URL: <https://lw1.at/s/astro/Bachelorarbeit.pdf> (cit. on pp. 18, 20).
- Winter, P. M. et al. (2022). ‘Residual Neural Networks for the Prediction of Planetary Collision Outcomes’. In preparation (cit. on p. 34).
- Zhou, L., R. Dvorak and L.-Y. Zhou (Aug. 2021). ‘On the formation of terrestrial planets between two massive planets: the case of 55 Cancri’. In: *MNRAS* 505.3, pp. 4571–4585. DOI: [10.1093/mnras/stab1534](https://doi.org/10.1093/mnras/stab1534). arXiv: [2105.10105](https://arxiv.org/abs/2105.10105) [astro-ph.EP] (cit. on pp. 6, 17, 23, 32, 35, 36).