Motion of icy grains originating in Saturn's D-ring

Apostolos Kolokotronis

Supervisors: Michiko Morooka, Erik Vigren

June 29, 2023

Abstract

Saturn's rings have long been thought to provide the planetary atmosphere with infalling material, affecting both its chemical composition and physical properties (e.g. temperature). As in-situ measurements of this material have been realized during the Grand Finale orbits of the Cassini spacecraft, icy grains falling from the innermost, D ring, towards the planet have been detected. In this work we are simulating the trajectories of these grains, starting from the D ring, moving under the influence of the planetary magnetic and gravitational fields. The atmospheric drag force that acts on them while traveling through the Saturn's atmosphere is also implemented and the grains apart from losing kinetic energy, are also getting sublimated. Our results suggest that in a constant electric potential of typical values for the ambient plasma around Saturn, the grains tend to move towards the southern hemisphere, with the negatively charged ones ending up in low to mid latitudes, whereas the positive ones follow a spiraling motion and fall close to the south pole.

1 Introduction

Material from Saturn's rings was thought for a long time to be falling into the planetary atmosphere and ionosphere, modifying their physical properties and chemical composition. During the end of the Cassini mission, the so-called Grand Finale, the Cassini spacecraft performed 22 orbits between the innermost ring (D ring) and the outermost part of the atmosphere. These orbits gave the opportunity to measure neutral, ion and electron densities close to the planet. It has also detected larger, nanometer sized dust grains of water ice density, with their population peaking close to the ring plane, in altitudes lower than 4000 km (Mitchell et al. 2018). These grains most likely originate from the D ring and the mechanisms that removes them from it are under investigation. This icy material has long been suspected to affect the chemistry of the planet's upper atmosphere (Connerney and Waite 1984).

Dust meteorites also fall down to the Earth's atmosphere and affect the upper atmosphere. Such an example is the polar noctilucent clouds, also called polar mesospheric clouds (PMCs), which often appear during the summer at high latitudes. Little is known about the cause of PMCs since in-situ observations of the mesosphere are difficult to conduct. Only the short time interval rocket can go through this area. Therefore, it is important to investigate Saturn's case, where the dust infall is continuous, and compare it to that of Earth. As a first step towards this, it would be valuable to understand how the icy grains fall from the D ring, where they end up and the amount of material deposited into the atmosphere.

At the distance of the D ring, the atmosphere of Saturn is probably not dense enough to decelerate the grains, but in Ip et al. 2016, it was shown that through collisions, charged grains can get ejected perpendicular to the ring plane, and thereafter follow the magnetic field lines, to distances closer to the planet. Then, collisions with molecules in the hydrogen-rich atmosphere will become more and more frequent, leading to losses in kinetic energy and mass, as shown in Hamil et al. 2018. The goal of this project is to combine the methods used in these two works and simulate the motion of charged icy grains from the D ring. We are accounting for the forces exerted by the gravitational field, the planetary magnetic field, the convective electric field and the atmospheric molecules. The initial physical properties (size/mass and charge) of the grains will be important for their fate, therefore we will consider different values for them.

2 Methods

In order to simulate the motion of charged icy nano-grains, launched from Saturn's D ring we consider the gravitational force of the planet, the atmospheric drag force due to neutral H_2 and the electromagnetic force of a dipole magnetic field. These forces act on the icy grains that are generally expected to be charged, either by collisions with the ambient plasma's particles or by solar irradiance. We will consider a simple case of constant plasma potential, which will define the initial charge of the grains. In our modeling, the grains, starting from an equatorial latitude on the ring plane, eventually reach the denser parts of the atmosphere, where atmospheric heating becomes important and results into sublimation of the particles, when moving with high enough velocities. The simulation is terminated when the grain either loses its mass, reaches the planet (1-bar level) or loses all of its kinetic energy and stops moving.

A table with all the constants and parameters used for the following simulations is provided in Appendix A and the codes used for this study are presented in Appendix B.

2.1 Drag force and sublimation

The modeling of the drag force and sublimation due to atmospheric heating has been done by following the method from Hamil et al. 2018. We consider spherical grains that are comprised of water ice and a neutral atmosphere of H_2 molecules. In our modeling we will include the magnetic field (in section 2.2), so we will use a three dimensional Cartesian coordinate system, where Saturn's D ring lies on the $x - y$ plane and z is parallel to Saturn's rotation axis. In this coordinate system the equation of motion of a grain under the influence of gravity and the drag force is:

$$\frac{d\vec{v}}{dt} = -\frac{GM_s}{r^2}\hat{r} - \Gamma|\vec{v}|\frac{3\rho_a}{4\rho_m}\left(\frac{4\pi\rho_m}{3m}\right)^{\frac{1}{3}}\vec{v} \tag{1}$$

where $\vec{v}$ is the velocity vector in Cartesian coordinates and r is the distance to the planetary center. The dimensionless "free-molecular drag" coefficient $\Gamma = 1$ is chosen similarly to Vondrak et al. 2008, G is the gravitational constant, M_s is Saturn's mass and m is the mass of the grain. The mass density of the grain, ρ_m is equal to that of water ice and that of the neutral atmosphere is calculated by:

$$\rho_a = n_{1/2}e^{-a_1(r-r_{1/2})}\frac{M_{H_2}}{N_A} \tag{2}$$

where N_A is the Avogadro number, M_{H_2} is the molecular mass of H_2 and the density profile formula is given by Koskinen et al. 2013, with $r_{1/2}$ and $n_{1/2}$ being the half-light distance (where the optical depth of the O V line measured from solar occultations is $\tau = 0.69$) and number density respectively and $a_1 = H^{-1}$ is the inverse scale height of an isothermal atmosphere.

The particle has a temperature T, which changes as it moves through the atmosphere and affects the mass loss rate of the grain, given by equation:

$$\frac{dm}{dt} = 4\pi \left(\frac{3m}{4\pi\rho_m}\right)^{\frac{2}{3}} e(T)\sqrt{\frac{\mu_i}{2\pi k_B T}} \tag{3}$$

with μ_i being the molecular mass of ice, k_B the Boltzmann constant and $e(T)$ is the saturation vapor pressure of ice:

$$e(T) = e^{9.550426 - \frac{5723.265}{T} + 3.53068 \ln T - 0.00728332T} \tag{4}$$

In order to calculate the temperature of the grain in each position, we use the energy-balance equation given by Vondrak et al. 2008:

$$\frac{1}{2}\pi R_{gr}^2 |v|^3 \rho_a \Lambda = 4\pi R_{gr}^2 \epsilon\sigma(T^4 - T_{\text{env}}^4) + \frac{4}{3}\pi R_{gr}^3 \rho_m C \frac{dT}{dt} + L(T)\frac{dm}{dt} \tag{5}$$

where R_{gr} is the grain radius, Λ the free molecular heat transfer coefficient, ϵ the emissivity of ice, σ the Stefan-Boltzmann constant, T_{env} is the temperature of the ring and C the heat capacity of ice. By assuming the grain to be in radiative equilibrium, the solar radiation absorbed gives a starting temperature, $T_{\text{env}} = 85\,\text{K}$. The latent heat of ice sublimation L is given by:

$$L(T) = 1000(2834.1 - 0.29T - 0.004T^2) \tag{6}$$

and the radius of the grain, assuming a spherical grain:

$$R_{gr} = \left(\frac{3m}{4\pi\rho_m}\right)^{1/3} \tag{7}$$

In Eq. (5) the left-hand side term is the energy gained by the grain due to frictional heating. On the right-hand side we have the energy loss terms, namely, the radiation loss in the first term, the energy used to heat the grain in the second one and the energy that goes into sublimating the grain in the third one. As in Hamil et al. 2018, we make the simplifying assumption that dT/dt is negligible and the grain reaches the environment's temperature quickly. Then substituting the mass loss rate from Eq. (3) into Eq. (5), we have:

$$\frac{1}{8}|\vec{v}|^3 \rho_a \Lambda = \epsilon\sigma(T^4 - T_{\text{env}}^4) + e(T)L(T)\sqrt{\frac{\mu_i}{2\pi k_B T}} \tag{8}$$

which can be solved numerically for the temperature in every position of the grain, so for different values of the velocity magnitude $|\vec{v}|$ and the atmosphere density ρ_a.

In order to compute the trajectory of the grain we implement a fourth-order Runge-Kutta method to solve Eq. (1). In each time step, the temperature is calculated by solving Eq. (8) and then used to calculate the mass of the grain, after it has lost mass due to sublimation, given by Eq. (3). Then the new position and velocity of the grain is calculated again by solving the equations of motion and so on.

For a grain starting with keplerian velocity, far enough from the planet (altitude above $r_{1/2}$, from the 1-bar level):

$$v_{\text{kep}} = \sqrt{\frac{GM_s}{r}} \tag{9}$$

the trajectory will be a circular motion around the planet, on the ring plane. If we, on the other hand, give the grain also a small initial velocity component towards the planetary center, then the grain spirals inwards, due to the gravitational force, until it reaches a distance $\sim r_{1/2}$. Around this distance the deceleration due to the drag force and the consequent heating and sublimation start to become noticeable. Then the time it takes for grains of the same initial radius to fall into the planet, depends on the initial velocity component towards the center, as well as their radii.

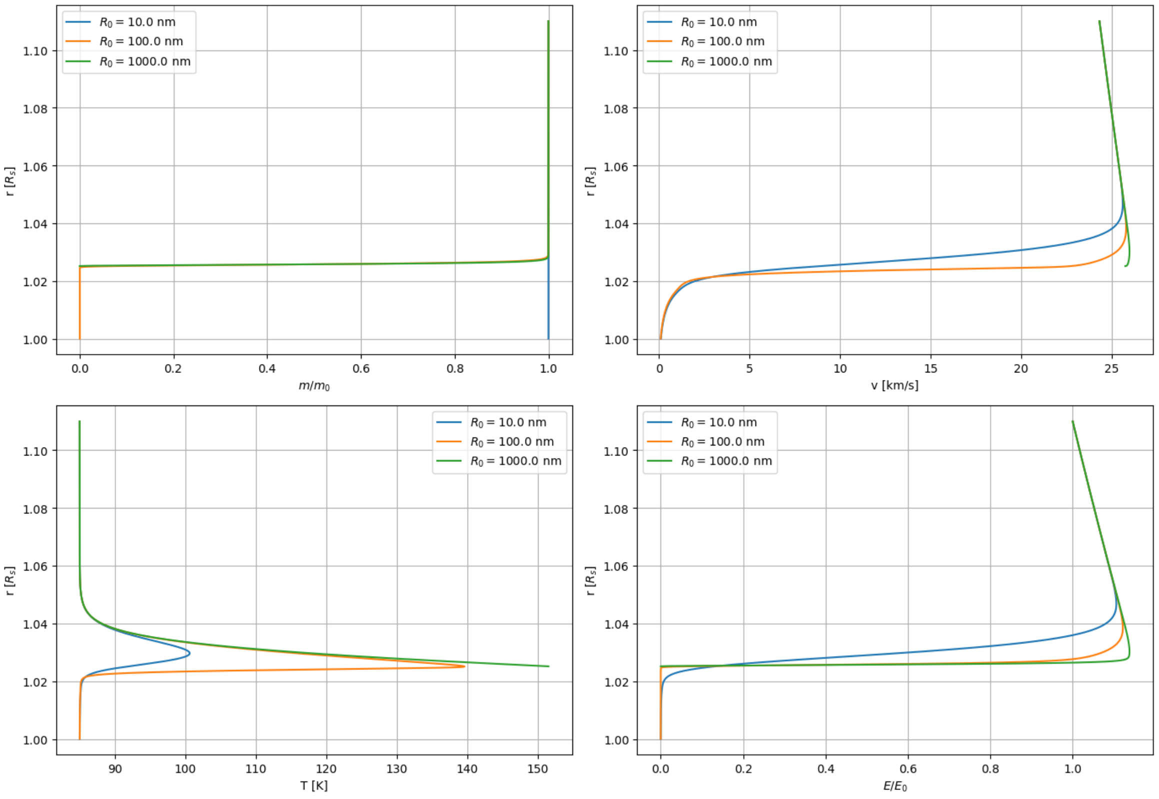

Figure 1: Grain mass, normalized to the initial mass (top left), grain velocity (top right), grain temperature (bottom left) and grain total energy, normalized to its initial energy (bottom right), against the distance to the planetary surface, for different initial grain sizes.

In Fig. 1 we can see how the mass, velocity, temperature and total energy change with the distance to the planetary surface, for three grains of different size, launched from the D ring with keplerian velocity in the tangential direction of the ring and a perturbation 5 km/s towards the planetary center. The energy of the grain, when it has a mass m, velocity $\vec{v}$ and is at a distance r from the planetary center is the sum of its kinetic and gravitational potential energy, given by:

$$E = \frac{1}{2}m|\vec{v}|^2 + m\frac{GM_s}{R_s^2}(r - R_s) \tag{10}$$

The smallest grain does not lose any mass, whereas the bigger ones get sublimated completely. The largest, initially 1000 nm grain gets sublimated before reaching the planet, when it essentially enters the thicker parts of the atmosphere at $\sim 1.03\,\mathrm{R_s}$. Even though the mass loss rate (Eq. 3) of the grain is a complicated function of the grain radius and the temperature (which in turn is the solution of another complicated equation, Eq. 8), it seems that the altitude where most of the energy has been lost is similar for all grain sizes. The location of maximal energy loss corresponds to the point where the grain has the maximum temperature. The smaller grain, since it maintains its mass, loses kinetic energy only because of its velocity decreasing, whereas the larger ones lose kinetic energy because of mass losses too.

2.2 Planetary magnetic field

Next we will simulate the motion of a charged grain, in the dipole magnetic field of Saturn, as in Ip et al. 2016. Saturn's magnetic field is a dipole shifted by $z_s = 0.04\,\mathrm{R_s}$ towards the north pole and has its magnetic dipole moment aligned with the planet's spin axis. In Cartesian coordinates it is given by:

$$\begin{aligned} B_x &= 3M_0\frac{x(z - z_s)}{r_s^5} \\ B_y &= 3M_0\frac{y(z - z_s)}{r_s^5} \\ B_z &= M_0\frac{3(z - z_s)^2 - r_s^2}{r_s^5} \end{aligned} \tag{11}$$

Where M_0 is the equatorial magnetic moment of Saturn and r_s is the distance of a particle relative to the shifted center of the magnetic field:

$$r_s = \sqrt{x^2 + y^2 + (z - z_s)^2} \tag{12}$$

In the equation of motion we include the force to the charged grain, with charge Q, from the corotation electric field:

$$\vec{E} = -\left(\vec{\Omega} \times \vec{r}\right) \times \vec{B} \tag{13}$$

where $\vec{\Omega} = \Omega\hat{z}$ is the angular frequency of Saturn's rotation. The equation of motion then becomes:

$$\frac{d\vec{v}}{dt} = -\frac{GM_s}{r^2}\hat{r} + \frac{Q}{m}(\vec{E} + \vec{v} \times \vec{B}) \tag{14}$$

The grain charge, Q is calculated by assuming a grain potential U. Then the charge is given by:

$$Q = C_{gr}U \tag{15}$$

where C_{gr} is the capacitance of a spherical grain, of radius R_{gr}:

$$C_{gr} = \epsilon_r C_{gr,0} = \epsilon_r 4\pi\epsilon_0 R_{gr} \tag{16}$$

where we assume that the relative permittivity $\epsilon_r = 1$ and ϵ_0 is the vacuum permittivity. Grains launched with keplerian velocity from outside the synchronous orbit radius:

$$R_{synch} = \left(\frac{GM_s}{\Omega^2}\right)^{\frac{1}{3}} \tag{17}$$

will move outwards, whereas those launched from a distance $r < R_{synch}$ will either stay at roughly the initial distance from the planet, but following an oscillatory motion between lower and higher latitudes, or move spirally inwards. The fate of these grains depends, apart from their initial state (position, velocity), also on their charge-to-mass ratio, which regulates how strong the Lorentz force is relative to the gravitational force.

The charge-to-mass ratio is determined by the initial grain radius and the electric potential, which is chosen to have values ranging from $-4\,\mathrm{V}$ to $4\,\mathrm{V}$. Multiple factors dictate the potential and typically it varies with the exposure to solar radiation and the properties (e.g. density and temperature) of the ambient plasma, which also depend on the distance from the planet. In the plasma disk region (the region around the ring), where the plasma density is relatively high, the potential is expected to be mostly negative. However, some of the Grand Finale measurements suggested that the potential can also be positive (Ye et al. 2018 and Morooka et al. 2019). Therefore, will consider both negative and positive constant potentials, as an ideal scenario.

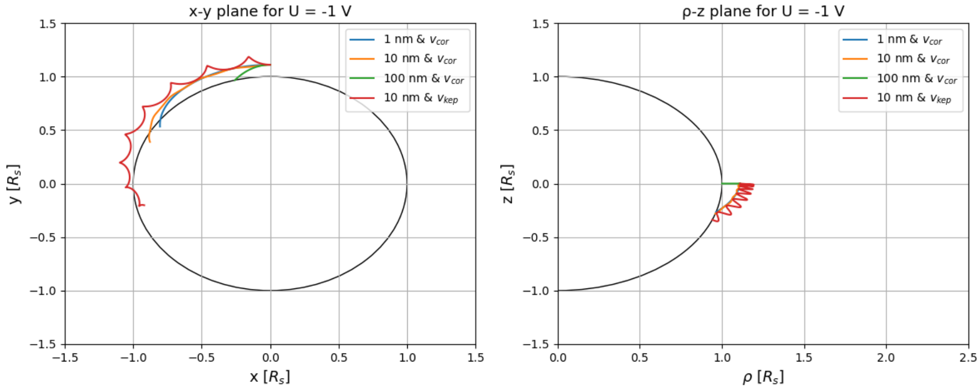

Figure 2: Trajectories of different sized grains, starting with corotation velocity from the inner radius of the D ring, moving under the influence of gravity and the planetary dipole magnetic field, for an electric potential $U = -1\,\mathrm{V}$. For the 10 nm grain we also include one that initially has keplerian velocity. On the left, the $x-y$ plane trajectories are plotted and on the right the $\rho-z$ plane ones, where ρ is the projection of the grain's position on the $x-y$ plane.

In implementing this model, we reproduced the results of Ip et al. 2016. As an example, in Fig. 2 and Fig. 3 we can see the trajectories followed by the grains under the gravitational and Lorentz forces as described above, for a negative and a positive electric potential. In the plots, grains with radii 1, 10 and 100 nm, with initially corotation velocity:

$$v_{cor} = \Omega r_0 \tag{18}$$

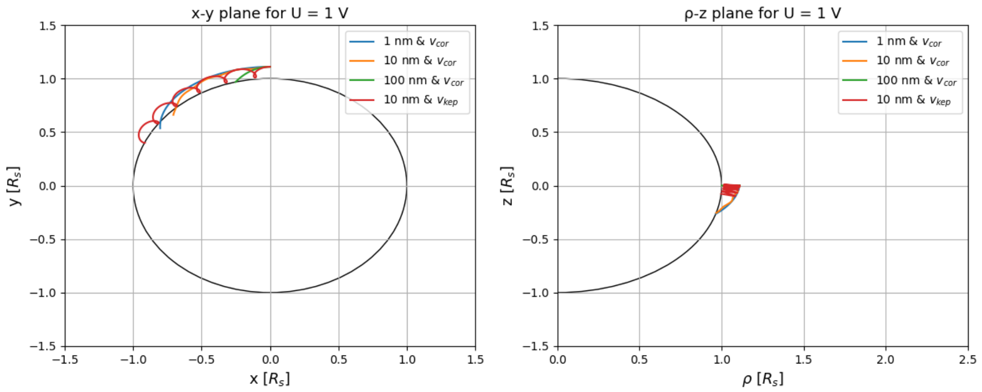

Figure 3: Similar to Fig. 2, for $U = 1\,\mathrm{V}$.

are shown. The initial distance of the grain to the planetary center, r_0 is in this case the inner radius of the D ring. This velocity is chosen, because grains that get charged may be expected to eventually follow the motion of the plasma rotating with the planetary dipole magnetic field. We also plot the trajectory of a 10 nm charged grain, that maintains its keplerian velocity at the start of the simulation, for comparison. The difference is that it has a larger gyro-radius, as expected and reaches at larger longitudes, before falling in the planet.

In constrast to the results of Ip et al. 2016, all the grains end up at $1\,\mathrm{R_s}$ after the electromagnetic force picks them up from the ring plane and leads them to the southern hemisphere. This difference is a result of the launch distance we used ($1.11\,\mathrm{R_s}$), which is much smaller than the so-called Northrop-Hill radius (Northrop and Hill 1983) for all the grain potentials. In addition, we did not eject the grains vertically to the ring plane. For the large grain, the Lorentz force is not strong enough to dominate over gravity and since the corotation velocity is not enough to maintain a stable orbit, it falls fast inwards and at a latitude very close to that of the ring plane. As highlighted in Ip et al. 2016, the launch distance can determine the hemisphere in which a grain ends. For both positive and negative potentials, all the grains end up in the southern hemisphere, suggesting that for launch distances close to the planet, the potential of the grain does not have a strong effect on the hemisphere it will move towards.

2.3 Combining the drag force with the magnetic field

When combining the four forces, gravity, atmospheric drag, magnetic field force and corotation electric field force mentioned in sections 2.1 and 2.2, the equation of motion becomes:

$$\frac{d\vec{v}}{dt} = -\frac{GM_s}{r^2}\hat{r} - \Gamma|\vec{v}|\frac{3\rho_a}{4\rho_m}\left(\frac{4\pi\rho_m}{3m}\right)^{\frac{1}{3}}\vec{v} + \frac{Q}{m}(\vec{E} + \vec{v} \times \vec{B}) \tag{19}$$

Since the grains now lose mass and also reduce in size, the Q/m ratio increases. In order to also change the charge, so that the grain also loses charges as well as mass, we combine Eq. (7), of the grain radius, which varies with the mass at each time step, with Eq. (16) for the capacitance. This way the change in size reduces also the capacitance and therefore the charge, Eq. (15).

To better visualize the interplay between the Lorentz fore and the drag force, we implement a new orthogonal coordinate system, which has its first unit vector parallel to the magnetic field line in each position, and two other unit vectors perpendicular to that. The coordinates of a vector in this coordinate system are calculated as follows:

- First we define the parallel to the field line vector of our basis, by:

$$\hat{b}_{\|} = \frac{\vec{B}}{|\vec{B}|} \tag{20}$$

  where $\vec{B} = (B_x, B_y, B_z)$ is calculated in a position (x, y, z) from Eq. (11).

- The first perpendicular, or eastward $(\hat{b}_{ew})$ unit vector is the perpendicular vector to the plane defined by $\hat{b}_{\|}$ and $\hat{z} = (0, 0, 1)$, divided with its magnitude:

$$\hat{b}_{ew} = \frac{\hat{b}_{\|} \times \hat{z}}{\|\hat{b}_{\|} \times \hat{z}\|} \tag{21}$$

- And the second perpendicular, or poleward $(\hat{b}_{pw})$ unit vector is perpendicular to the first two:

$$\hat{b}_{pw} = \hat{b}_{\|} \times \hat{b}_{ew} \tag{22}$$

- Finally the components of a vector $\vec{A} = (A_x, A_y, A_z)$ in the new coordinate system are:

$$\begin{aligned} A_{\|} &= \vec{A} \cdot \hat{b}_{\|} \\ A_{ew} &= \vec{A} \cdot \hat{b}_{ew} \\ A_{pw} &= \vec{A} \cdot \hat{b}_{pw} \end{aligned} \tag{23}$$

3 Results

In Fig. 4 and Fig. 5 the trajectories of grains moving under the influence of the electromagnetic, gravity, and atmospheric drag forces are shown. The trajectories for negatively charged grains ($U = -1\,\mathrm{V}$), as shown in Fig. 4, are similar to those where the drag force is absent. The smallest grain gets sublimated, but also follows a similar path. The trajectories for positively charged grains ($U = 1\,\mathrm{V}$) are shown in Fig. 5. Including the drag force their trajectories (except for the largest grain) appear quite different, as instead of ending up at low latitudes, they spiral around the planetary axis moving to higher (negative) latitudes. They finally fall into the planet or get sublimated very close to the south pole and compared to the negative ones, are moving for a longer period of time. For example, the $U = 1\,\mathrm{V}$, $R_{gr} = 10\,\mathrm{nm}$ grain with v_{cor} initially, is moving for a little less than 12 hrs, whereas the corresponding negatively charged one only for ~ 2 hrs.

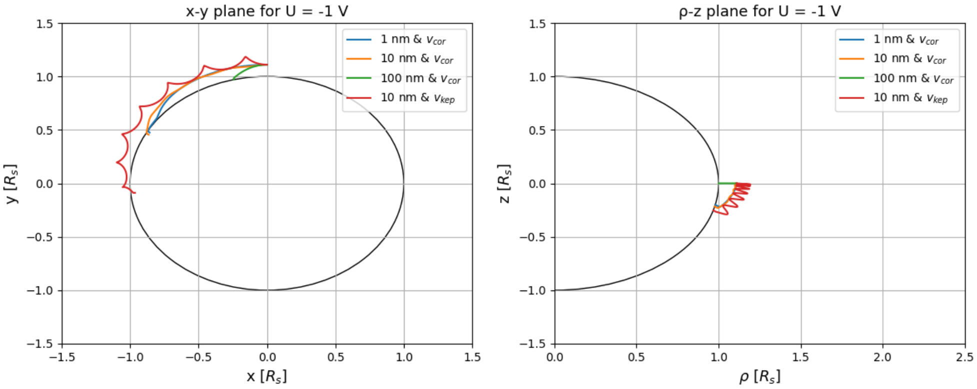

Figure 4: The same as Fig. 2, including also the drag force, for $U = -1\,\mathrm{V}$.

Again, the larger grains seems to not be affected by the magnetic force, since their Q/m ratios are small, therefore are slowed down quickly by the drag force. The smaller than keplerian initial velocity also works towards diverging the grain from a stable orbit to a distance $\sim 1.03\,\mathrm{R_s}$, where the drag force is even stronger.

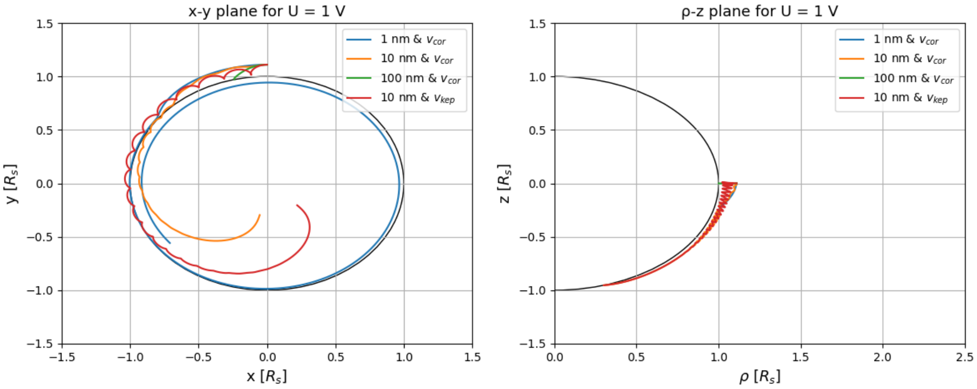

Figure 5: The same as Fig. 2, including also the drag force, for $U = 1\,\mathrm{V}$.

In order to see when each force dominates during the motion, we plot the force components against time, for the $U = 1\,\mathrm{V}$, $R_{gr} = 10\,\mathrm{nm}$ grain with v_{cor} initially. The components are calculated on the magnetic field line coordinate system, as described in section 2.3 and plotted against time in Fig. 6. We also plot the velocity components against time in the same coordinate system, since the velocity of the grain (in both magnitude and direction) affects the magnetic and drag forces. On the bottom panel we plot the distance to the planetary surface (1-bar level) against time, where at $\sim 1.03\,\mathrm{R_s}$ we expect to have the increase of the atmospheric drag force, due to the density profile from Eq. (2).

In the parallel to the field line direction, the magnetic and corotation electric field forces are always zero, as expected since both forces are perpendicular to a plane defined by the magnetic field and the particle velocity or the corotation velocity respectively, as we see in the equation of motion (19). The electric force is also zero in the $\hat{b}_{ew}$ direction, since $\vec{\Omega} \times \vec{r}$ is parallel to it, making $(\vec{\Omega} \times \vec{r}) \times \vec{B}$ zero. As the grain starts in the equatorial-ring plane, the eastward unit vector there corresponds to the

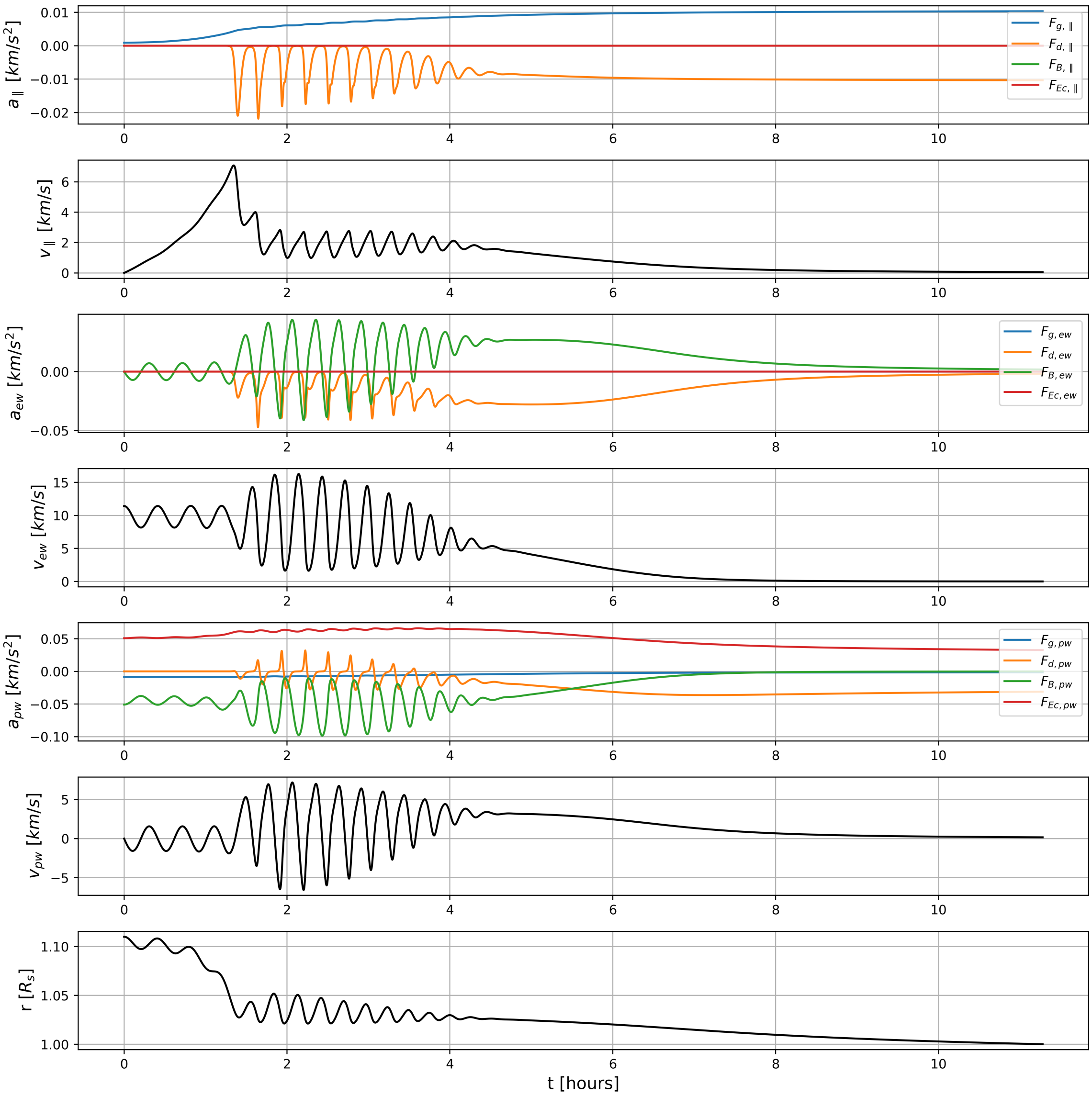

Figure 6: Acceleration components for each force in the magnetic field line coordinate system, as well as the corresponding velocity components of the grain in each direction, over time. Red line is the corotation electric field force, green the magnetic field force, orange the drag force and blue is gravity. The bottom plot shows the distance of the grain from the planetary surface over time. This figure is for a $U = 1\,\mathrm{V}$, $R_{gr} = 10\,\mathrm{nm}$ grain with v_{cor} as its initial velocity.

azimuthal direction, which leads to a v_{ew} initial velocity equal to v_{cor} and no velocities in the other directions. Because of that the F_{Ec} and F_B act on the perpendicular, poleward direction in the beginning. This accelerates the grain in the poleward direction, leading to also a F_B in the eastward direction. The $v_{\|}$ increases under the influence of gravity, which in its turn increases as the particle follows the magnetic field line in smaller radii.

When the distance to the planetary center reaches $\sim 1.03\,\mathrm{R_s}$ for the first time, the drag force starts acting in the direction opposite to the particle's velocity. The grain spends some time below this distance and then goes again higher, since the magnetic

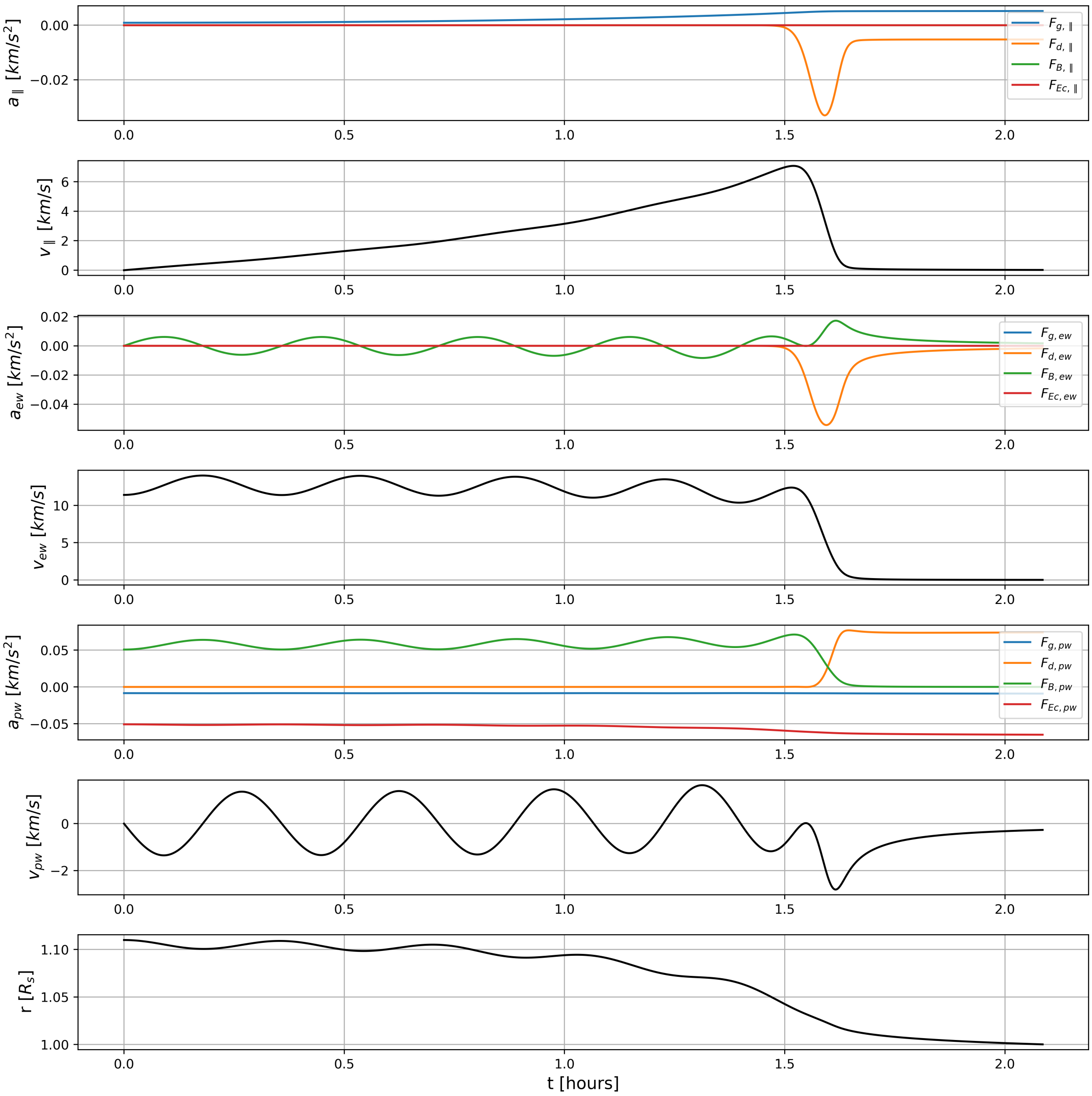

Figure 7: Acceleration components for each force in the magnetic field line coordinate system, as well as the corresponding velocity components of the grain in each direction, over time. Red line is the corotation electric field force, green the magnetic field force, orange the drag force and blue is gravity. The bottom plot shows the distance of the grain from the planetary surface over time. This figure is for a $U = -1\,\mathrm{V}$, $R_{gr} = 10\,\mathrm{nm}$ grain with v_{cor} as its initial velocity.

force in the eastward direction becomes more positive and less negative during its oscillations. This is repeated many times, with each passing through the atmosphere slowing it down more and more, until the velocity in all directions stops oscillating and the forces balance each other.

After this point the grain moves very slowly, with the eastward and poleward velocity components being slightly higher than the parallel to the field line ones. This is the part of the motion where the grain moves eastward and southward, spiraling towards the south pole, but very slowly approaching lower altitudes too. The drag force plays a key role in creating this major difference compared to the case with only the magnetic field, because it balances the gravitational force along $\hat{b}_{\parallel}$. If it was absent, then the gravity would pull it towards the field line direction and therefore inwards much faster, with less of a spiraling motion, as we previously showed in Fig. 3.

Similarly to Fig. 6, we include Fig. 7, for a 10 nm sized grain, starting with corotation velocity in a negative $-1\,\mathrm{V}$ potential. This grain is also reaching the $1.03\,\mathrm{R_s}$ distance at about the same time $\sim$ 1.7 hrs, but instead of being brought up again, the balance between the poleward magnetic and electric force is pointing towards the planet, sinking the particle even deeper in the atmosphere. The $v_{\parallel}$ and v_{ew} velocity components vanish rapidly at the same time. The v_{pw} also decreases, but goes to zero at a later point, thus the grain moves southwards but ends up at less negative latitude than the 1 V one. The motion terminates in a little over 2 hrs, with the grain falling into the planet.

It seems that the interplay between the forces due to the dipole magnetic field (F_B) and due to its rotation (F_{Ec}) in the poleward direction determines the fate of the grain and how fast its motion will be. Since the corotation velocity $v_{cor} = \vec{\Omega} \times \vec{r}$ and the magnetic field $\vec{B}$ both depend on the distance, different initial launch radii might significantly change the results. In combination with the simplistic assumption of a constant grain potential these results form an idealized scenario.

4 Discussion and Conclusion

We studied the motion of charged icy grains launched from the innermost radius of Saturn's D ring. We implemented a magnetic dipole as the planetary magnetic field, which, due to the planet's rotation, also results in a corotation electric force being exerted on the grains. As the grains move in the deeper parts of the molecular hydrogen atmosphere, atmospheric drag starts acting on them and sublimation reduces their mass/size and therefore their charge as well.

Our results show that negatively charged grains' fate is not significantly changed by the drag force and sublimation. The main effect of the atmosphere on them is the slightly higher final latitudes (in the souther hemisphere) and smaller longtitudal length they travel. The positively charged ones on the other hand, end up at more negative latitudes, also in the southern hemisphere, very close to the south pole.

We note that no ejection speed perpendicular to the ring plane is needed to produce the results mentioned, although in reality some vertical perturbation might exist, due to collisions of the dust particles with interplanetary micrometeoroids. These collisions eject water vapors, which can recondense into nanometer sized grains moving away from the ring plane with velocities $\sim (1-2)\,\mathrm{km\,s^{-1}}$, as indicated in Ip et al. 2016 and Ip 1995. This ejection velocity, if large, can potentially change the fate of the grains. Its direction is also important, as the trajectory of a charged grain will vary, depending on which pole it moves towards initially. An estimation of the size of the micrometeoroids moving through the ring plane and their velocities would provide more insight on whether they are sufficient to alter the icy grains' trajectories from the ones we simulated.

The constant charge state of the grains also consists of a non-realistic scenario, as it depends on the environment of the plasma. The grain can acquire some charge stochastically, with a variation in time, due to a change in the plasma environment. Accounting for a varying grain potential (depending on the ambient plasma and radiation environment) could be a feature of an improved model. In any future study, we could also account for the radiation pressure in the equations of motion. The phase of decelerating a neutral grain from keplerian velocity to corotation velocity when it finally gets charged should also be considered.

Appendix A

Below is the table with the values of parameters and constants used, as presented in section 2.

Symbol	Description	Value
M_s	Mass of Saturn	5.68×10^{26} kg
R_s	Equatorial radius of Saturn	60 268 km
Ω	Saturn's rotational frequency	1.707×10^{-4} rad $\cdot$ s^{-1}
r_0	D ring inner radius	1.11 R$_s$
G	Gravitational constant	6.674×10^{-11} m^{3}kg^{-1}s^{-2}
M_{H_2}	Molecular mass of H_2	2.016×10^{-3} kg/mol
ρ_m	Mass density of water ice	917 kg/m^3
N_A	Avogadro's number	6.022×10^{23}
Γ	Free molecular drag coefficient	1
k_B	Boltzmann constant	$1.380\,649 \times 10^{-23}$ J $\cdot$ K^{-1}
Λ	Free molecular heat transfer coefficient	1
σ	Stefan-Boltzmann constant	$5.670\,374 \times 10^{-8}$ Wm^{-2}K^{-4}
T_{env}	D ring temperature	85 K
ϵ	Emissivity of ice	1
$r_{1/2}$	Half-light distance of atmosphere	2340 km
$n_{1/2}$	Half-light density of atmosphere	8.9×10^{13} m^{-3}
H	Scale height of atmosphere	219 km
M_0	Equatorial magnetic moment of Saturn	4.6×10^{18} T $\cdot$ m^3
ϵ_0	Vacuum permittivity	8.54×10^{-12} A^{2}s^{4}kg^{-1}m^{-3}
ϵ_r	Relative permittivity	1

Table 1: Values of parameters and constants used in this study.

Appendix B

In this appendix, the codes used for this project are presented.

```
import numpy as np
import scipy
import matplotlib as mplt
from matplotlib import pyplot as plt
import sympy as sp
from sympy import *
from scipy.optimize import fsolve
from scipy.optimize import bisect
from IPython.display import display, Latex
import mpl_toolkits.mplot3d.axes3d as axes3d
import pickle

"CONSTANTS in SI"
G=6.674E-11 # m^3 kg^-1 s^-2
amu=1.67e-27; ## 1 amu in kg
eps0=8.854e-12;  #vacuum permittivity
Ms=568.32E+24 #Saturn's mass in kg
Rs=60268*1e3 # Saturn's radius in m
Gamma=1#drag force factor
rho_m_mass=917 #grain density (ice) in kg/m^3
M=18.01528e-3 #molecular weight of water, kg/mol
NA=6.022e23 #Avogadro's number
rho_m= NA/M *rho_m_mass #number density of water grain m^-3
M_H2=2.016e-3 #molecular weight of H_2, kg/mol
q_e=1.60217663e-19 #electron charge in C
Mom=4.6e+18  # equatorial magnetic momment of Saturn T*m^3
Om=1.707e-4 #angular frequency of Saturn's rotation
R_synch=(G*Ms/Om**2)**(1/3)
print('Radius of synchronous orbit: ', R_synch/Rs ,' Rs')

"Time"
dt=1
t=np.arange(0,12000,dt)
print('total simulation time: ', t[-1]/(60*60), ' hrs')
N=len(t)


"Initial position, velocity"
R_dring=1.11*Rs
pos_init = np.array([0, R_dring, 0]) #r_0, theta_0, phi_0

v_kep=np.sqrt(G*Ms/np.linalg.norm(pos_init))
v_cor=Om*R_dring

print('keplerian velocity: ', v_kep*1e-3, ' km/s')
print('corotation velocity: ', v_cor*1e-3, ' km/s')
1e-3*(R_dring-Rs)

R_grain0=np.array([1, 10, 100, 10.00001])*1e-9 #ice grain initial radii
    in m
U0=[1, 2, 3, 4]  # assumed magnitude of grain potential in Volts (change
    as you wish!)
if np.sign(U0[0])<0:
    data_title='negative_U'

    if Gamma==0:
        data_title='mag_only'+data_title

elif np.sign(U0[0])>0:
    data_title='positive_U'
    if dt<0.1: data_title=data_title+'_small_dt'
    if Gamma==0:
        data_title='mag_only'+data_title
else:
    data_title='drag_only'
v0=['cor', 'cor', 'cor', 'kep']
#R_NH=(2/3 * G*Ms/(Om-G*Ms/3 * m0* 1/(abs(Q0)*Mom))**2)**(1/3)
m_init=rho_m_mass*4/3* np.pi* R_grain0**3 #inital grain masses in kg
C_init=4*np.pi*eps0*R_grain0 #initial capacitance in C/V, capacitance of
   sphere is C = 4*pi*eps_0*r (assume rel perm      1)

for U in U0:
    Q_init=C_init*U # initial grain charge in SI
    print('For␣U␣=␣', U, '␣Volts␣\n␣R␣=␣', R_grain0*1e9, '␣nm␣\n', '␣Q_0/
       m_0␣=␣', Q_init/m_init * (amu/q_e), '␣e/amu')

particle={}
for U in U0:
    particle[U]={}
    for i, Rg0 in enumerate(R_grain0):
        C0=4*np.pi*eps0*Rg0
        R_NH=(2/3 * G*Ms/(Om-G*Ms/3 * m0* 1/(abs(C0*U)*Mom))**2)**(1/3)
        print(R_NH/(Rs*1e-3))
        particle[U][Rg0]={'m0':rho_m_mass*4/3* np.pi* Rg0**3, 'Q0':C0*U,
           'Q_0/m_0': C0*U/(rho_m_mass*4/3*np.pi*Rg0**3) * (amu/q_e), '
           fate':'moving'}
        if v0[i]=='kep':
            particle[U][Rg0].update({'v0':np.array([-v_kep, 0, 0]), '
               v0_txt':'$v_{kep}$'})
        elif v0[i]=='cor':
            particle[U][Rg0].update({'v0':np.array([-v_cor, 0, 0]), '
               v0_txt':'$v_{cor}$'})

from functions_cart import gravity, rho_a, drag, vel_RHS, temp, mass,
   magnetic

def RK4(dt, pos, pos_der, R_grain, Q, m):

    k1r=vel_RHS(pos, pos_der)

    k1v=gravity(pos, pos_der, G, Ms) + \
        drag(pos, pos_der, R_grain, rho_m, Gamma) + \
        magnetic(pos, pos_der, Q, m, Rs, Mom, Om)[0] + \
        magnetic(pos, pos_der, Q, m, Rs, Mom, Om)[1]

    k2r=vel_RHS(pos+0.5*dt*k1r, pos_der+0.5*dt*k1v)

    k2v=gravity(pos+0.5*dt*k1r, pos_der+0.5*dt*k1v, G, Ms) + \
        drag(pos+0.5*dt*k1r, pos_der+0.5*dt*k1v, R_grain, rho_m, Gamma) +
           \
        magnetic(pos+0.5*dt*k1r, pos_der+0.5*dt*k1v, Q, m, Rs, Mom, Om)
            [0] + \
        magnetic(pos+0.5*dt*k1r, pos_der+0.5*dt*k1v, Q, m, Rs, Mom, Om)
            [1]

    k3r=vel_RHS(pos+0.5*dt*k2r, pos_der+0.5*dt*k2v)

    k3v=gravity(pos+0.5*dt*k2r, pos_der+0.5*dt*k2v, G, Ms) + \
        drag(pos+0.5*dt*k2r, pos_der+0.5*dt*k2v, R_grain, rho_m, Gamma) +
             \
        magnetic(pos+0.5*dt*k2r, pos_der+0.5*dt*k2v, Q, m, Rs, Mom, Om)
            [0] + \
        magnetic(pos+0.5*dt*k2r, pos_der+0.5*dt*k2v, Q, m, Rs, Mom, Om)
            [1]

    k4r=vel_RHS(pos+dt*k3r, pos_der+dt*k3v)

    k4v=gravity(pos+dt*k3r, pos_der+dt*k3v, G, Ms) + \
        drag(pos+dt*k3r, pos_der+dt*k3v, R_grain, rho_m, Gamma) + \
        magnetic(pos+dt*k3r, pos_der+dt*k3v, Q, m, Rs, Mom, Om)[0] + \
        magnetic(pos+dt*k3r, pos_der+dt*k3v, Q, m, Rs, Mom, Om)[1]

    pos_new=pos + dt/6 * (k1r+2*k2r+2*k3r+k4r)
    pos_der_new=pos_der + dt/6 * (k1v+2*k2v+2*k3v+k4v)

    return pos_new, pos_der_new

"SOLVER"
for U in U0:
    print('U␣=␣', U, 'Volts')
    for Rg0 in R_grain0:
        print('Rg0␣=␣', Rg0*1e9, '␣nm␣', end='')
        pos_init = np.array([0, R_dring, 0]) #r_0, theta_0, phi_0
        pos_der_init = particle[U][Rg0]['v0']

        x = []; y = []; z = []
        r = []
        vx=[]; vy=[]; vz=[]; vtot=[];
        state=np.array([[pos_init],[pos_der_init]])
        pos=pos_init; pos_der=pos_der_init
        m0=particle[U][Rg0]['m0']
        Q0=particle[U][Rg0]['Q0']
        a_gr=[]; a_drag=[]; a_mag=[]; a_Ec=[]; gr_mag=[]; drag_mag=[];
            mag_mag=[]; Ec_mag=[]
        T=[]; m=[]; R_grain=[]; Q=[]
        m_bef=m0
        pos=pos_init; pos_der=pos_der_init

        for i in range(1,N):
            x.append(pos[0]); y.append(pos[1]); z.append(pos[2])
            r.append(np.sqrt(x[i-1]**2+y[i-1]**2+z[i-1]**2))
            vx.append(pos_der[0]); vy.append(pos_der[1]); vz.append(
                pos_der[2])
            vtot.append(np.sqrt(vx[i-1]**2+vy[i-1]**2+vz[i-1]**2)*1e-3)

            T_sol=temp(pos, pos_der, vtot[i-1]*1e3)
            T.append(T_sol[0])
        if i==1 or Gamma==0:
            m.append(m0)
        else:
            m.append(mass(T[i-1], m_bef, t[i], dt, rho_m_mass))
        m_bef=m[i-1]

        R_grain.append((3/(4*np.pi) *m[i-1]/rho_m_mass)**(1/3))
        Q.append(4*np.pi*eps0*R_grain[i-1]*U)

        a_gr.append(gravity(pos, pos_der, G, Ms)*1e-3); gr_mag.append
           (np.linalg.norm(a_gr[i-1]))
        a_drag.append(drag(pos, pos_der, R_grain[i-1], rho_m, Gamma)
           *1e-3); drag_mag.append(np.linalg.norm(a_drag[i-1]))
        a_Ec.append(magnetic(pos, pos_der, Q[i-1], m[i-1], Rs, Mom,
           Om)[0]*1e-3); Ec_mag.append(np.linalg.norm(a_Ec[i-1]))
        a_mag.append(magnetic(pos, pos_der, Q[i-1], m[i-1], Rs, Mom,
           Om)[1]*1e-3); mag_mag.append(np.linalg.norm(a_mag[i-1]))

        if (r[i-1]/Rs-1)<=0.0001:
             particle[U][Rg0]['fate']='fell'
            m.pop(); T.pop(); Q.pop()
            x.pop(); y.pop(); z.pop(); r.pop()
            vx.pop(); vy.pop(); vz.pop(); vtot.pop()
            a_gr.pop(); a_drag.pop(); a_Ec.pop(); a_mag.pop()
            gr_mag.pop(); drag_mag.pop(); Ec_mag.pop(); mag_mag.pop()
            break

        if m[i-1]/m0<=0.0001:
             particle[U][Rg0]['fate']='sublimated'
            m.pop(); T.pop(); Q.pop()
            x.pop(); y.pop(); z.pop(); r.pop()
            vx.pop(); vy.pop(); vz.pop(); vtot.pop()
            a_gr.pop(); a_drag.pop(); a_Ec.pop(); a_mag.pop()
            gr_mag.pop(); drag_mag.pop(); Ec_mag.pop(); mag_mag.pop()
            break

        if vtot[i-1]<=0.01:
             particle[U][Rg0]['fate']='slowed'
            m.pop(); T.pop(); Q.pop()
            x.pop(); y.pop(); z.pop(); r.pop()
            vx.pop(); vy.pop(); vz.pop(); vtot.pop()
            a_gr.pop(); a_drag.pop(); a_Ec.pop(); a_mag.pop()
            gr_mag.pop(); drag_mag.pop(); Ec_mag.pop(); mag_mag.pop()
            break

        state = RK4(dt, pos, pos_der, R_grain[i-1], Q[i-1], m[i-1])
        pos=state[0]; pos_der=state[1]

    print('fate:␣', particle[U][Rg0]['fate'], '␣t:␣', t[i-1]/(60*60),
         '␣hours')
    particle[U][Rg0].update({'x':x, 'y':y, 'z':z, 'r':r, 'vx':vx, 'vy
        ':vy, 'vz':vz, 'vtot':vtot, 'T':T, 'm':m, 'Q':Q,
                                'a_gr':a_gr, 'a_drag':a_drag, 'a_Ec':
                                    a_Ec, 'a_mag':a_mag, 'gr_mag':gr_mag
                                    , 'drag_mag':drag_mag, 'mag_mag':
                                    mag_mag,
                                'Ec_mag':Ec_mag, 'dt':dt, 't_final': t[i
                                    -2]})
```

```
U_plot=list(particle.keys())
R_g0_plot=list(particle[U_plot[0]].keys())
particle[U_plot[0]].keys()

for U in U_plot:
    if np.sign(U)<0:
        V_txt='U = m_'+str(int(abs(U)))+' V'
        title='for U = '+str(U)+' V'
        if Gamma==0:
            V_txt='mag_only_'+V_txt
    elif np.sign(U)>0:
        V_txt='U = p_'+str(int(abs(U)))+' V'
        title='for U = '+str(U)+' V'
        if Gamma==0:
            V_txt='mag_only_'+V_txt
    else:
        V_txt='drag_only_'
        title=''
    fig, axs = plt.subplots(1, 2, figsize=(14,5))
    Sat_xy=plt.Circle((0,0),Rs/Rs, color='k',fill=False)
    Sat_xz=plt.Circle((0,0),Rs/Rs, color='k',fill=False)
    axs[0].add_patch(Sat_xy)
    axs[1].add_patch(Sat_xz)
    for Rg0 in R_g0_plot:
        x = particle[U][Rg0]['x']; y = particle[U][Rg0]['y']; z =
            particle[U][Rg0]['z']
        if particle[U][Rg0]['v0_txt']=='$v_{kep}$':
            line='-'
        elif particle[U][Rg0]['v0_txt']=='$v_{cor}$':
            line='-'
        axs[0].plot(np.array(x)/Rs,np.array(y)/Rs,linestyle=line, label=
            str(int(Rg0*1e+9))+' nm & '+ particle[U][Rg0]['v0_txt'])
        axs[1].plot(np.sqrt(np.array(x)**2+np.array(y)**2)/Rs,np.array(z)
            /Rs,linestyle=line, label=str(int(Rg0*1e+9))+' nm & '+
            particle[U][Rg0]['v0_txt'])

    axs[0].set_title('x-y plane '+title, fontsize=13)
    axs[0].set_xlabel('x [$R_s$]', fontsize=13)
    axs[0].set_ylabel('y [$R_s$]', fontsize=13)
    axs[0].set_xlim([-1.5,1.5])
    axs[0].set_ylim([-1.5,1.5])
    axs[0].grid()
    axs[0].legend()

    axs[1].set_title('  -z plane '+title, fontsize=13)
    axs[1].set_xlabel(r'$\rho$ [$R_s$]', fontsize=13)
    axs[1].set_ylabel('z [$R_s$]', fontsize=13)
    axs[1].set_xlim([0,2.5])
    axs[1].set_ylim([-1.5,1.5])
    axs[1].grid()
    axs[1].legend()
    fig.savefig('traj_'+V_txt+'.png', dpi='figure')

def B_coord(Bx, By, Bz):
    B=np.array([Bx, By, Bz])
    Bt=B/np.linalg.norm(B)
    Bp1_non_unit=np.cross(Bt, np.array([0, 0, 1]))
    Bp1=Bp1_non_unit/np.linalg.norm(Bp1_non_unit)
    Bp2=np.cross(Bt, Bp1)
    return Bt, Bp1, Bp2

def transf(Bt, Bp1, Bp2, A):
    return np.dot(A, Bt), np.dot(A, Bp1), np.dot(A, Bp2)

def B_cart(x, y, z, Mom, Rs):
    z_s = 0.04*Rs
    r_s = np.sqrt(x**2+y**2+(z-z_s)**2)
    Bx = 3*Mom*x*(z-z_s)/r_s**5
    By = 3*Mom*y*(z-z_s)/r_s**5
    Bz = Mom*(3*(z-z_s)**2-r_s**2)/r_s**5

    return Bx, By, Bz

U_plot=list(particle.keys())
R_g0_plot=list(particle[U_plot[0]].keys())

for U in U_plot:
    if U==0: break
    if np.sign(U)<0:
        V_txt='U␣=␣m_'+str(int(abs(U)))+'␣V,␣'
        if Gamma==0:
            V_txt='mag_only_'+V_txt
    elif np.sign(U)>0:
        V_txt='U␣=␣p_'+str(int(abs(U)))+'␣V,␣'
        if Gamma==0:
            V_txt='mag_only_'+V_txt
    else:
        V_txt='drag_only_'
    for Rg0 in R_g0_plot:
        le=int(len(particle[U][Rg0]['r']))
        tt=np.linspace(0, particle[U][Rg0]['t_final'], le)/(60*60)
        x=np.array(particle[U][Rg0]['x']); y=np.array(particle[U][Rg0]['y
            ']); z=np.array(particle[U][Rg0]['z']); r=np.array(particle[U
            ][Rg0]['r']);

        fig, axs = plt.subplots(7, 1, figsize=(12,12))
        vx=[]; vy=[]; vz=[]
        gr_x=[]; gr_y=[]; gr_z=[]; dr_x=[]; dr_y=[]; dr_z=[]; mag_x=[];
            mag_y=[]; mag_z=[]; Ec_x=[]; Ec_y=[]; Ec_z=[]
        Bx=[]; By=[]; Bz=[]; B=[]; Bt=[]; Bp1=[]; Bp2=[]

        gr_f1=[]; gr_f2=[]; gr_f3=[]; dr_f1=[]; dr_f2=[]; dr_f3=[];
            mag_f1=[]; mag_f2=[]; mag_f3=[]; Ec_f1=[]; Ec_f2=[]; Ec_f3
            =[];
        v_f1=[]; v_f2=[]; v_f3=[]
        for i in range(le):
            vx.append(particle[U][Rg0]['vx'][i]); vy.append(particle[U][
                Rg0]['vy'][i]); vz.append(particle[U][Rg0]['vz'][i])
            gr_x.append(particle[U][Rg0]['a_gr'][i][0]); gr_y.append(
                particle[U][Rg0]['a_gr'][i][1]); gr_z.append(particle[U][
                Rg0]['a_gr'][i][2]);
            dr_x.append(particle[U][Rg0]['a_drag'][i][0]); dr_y.append(
                particle[U][Rg0]['a_drag'][i][1]); dr_z.append(particle[U
                ][Rg0]['a_drag'][i][2]);
            mag_x.append(particle[U][Rg0]['a_mag'][i][0]); mag_y.append(
                particle[U][Rg0]['a_mag'][i][1]); mag_z.append(particle[U
                ][Rg0]['a_mag'][i][2]);
            Ec_x.append(particle[U][Rg0]['a_Ec'][i][0]); Ec_y.append(
                particle[U][Rg0]['a_Ec'][i][1]); Ec_z.append(particle[U][
                Rg0]['a_Ec'][i][2]);
            Bx.append(B_cart(x[i], y[i], z[i], Mom, Rs)[0]); By.append(
                B_cart(x[i], y[i], z[i], Mom, Rs)[1]); Bz.append(B_cart(x
                [i], y[i], z[i], Mom, Rs)[2])

            Bt, Bp1, Bp2 = B_coord(Bx[i], By[i], Bz[i])

            gr=np.array([gr_x[i], gr_y[i], gr_z[i]]); dr=np.array([dr_x[i
                ], dr_y[i], dr_z[i]]); mag=np.array([mag_x[i], mag_y[i],
                mag_z[i]])
            Ec=np.array([Ec_x[i], Ec_y[i], Ec_z[i]]); v=np.array([vx[i],
                vy[i], vz[i]])
            gr_f1.append(transf(Bt, Bp1, Bp2, gr)[0]); gr_f2.append(
                transf(Bt, Bp1, Bp2, gr)[1]); gr_f3.append(transf(Bt, Bp1
                , Bp2, gr)[2]);
            dr_f1.append(transf(Bt, Bp1, Bp2, dr)[0]); dr_f2.append(
                transf(Bt, Bp1, Bp2, dr)[1]); dr_f3.append(transf(Bt, Bp1
                , Bp2, dr)[2]);
            mag_f1.append(transf(Bt, Bp1, Bp2, mag)[0]);  mag_f2.append(
                transf(Bt, Bp1, Bp2, mag)[1]);  mag_f3.append(transf(Bt,
                Bp1, Bp2, mag)[2]);
            Ec_f1.append(transf(Bt, Bp1, Bp2, Ec)[0]); Ec_f2.append(
                transf(Bt, Bp1, Bp2, Ec)[1]); Ec_f3.append(transf(Bt, Bp1
                , Bp2, Ec)[2]);
            v_f1.append(transf(Bt, Bp1, Bp2, v)[0]); v_f2.append(transf(
                Bt, Bp1, Bp2, v)[1]); v_f3.append(transf(Bt, Bp1, Bp2, v)
                [2]);

        axs[0].plot(tt, gr_f1, label='$F_{g,\parallel}$')
        axs[0].plot(tt, dr_f1, label='$F_{d,\parallel}$')
        axs[0].plot(tt, mag_f1, label='$F_{B,\parallel}$')
        axs[0].plot(tt, Ec_f1, label='$F_{Ec,\parallel}$')
        axs[0].set_ylabel('$a_{\parallel}$␣[$km/s^2$]', fontsize=13)
        axs[0].grid()
        axs[0].legend(loc='upper␣right')

        axs[1].plot(tt, np.array(v_f1)*1e-3, 'k')
        axs[1].set_ylabel('$v_{\parallel}$␣[$km/s$]', fontsize=13)
        axs[1].grid()

        axs[2].plot(tt, gr_f2, label='$F_{g,␣ew}$')
        axs[2].plot(tt, dr_f2, label='$F_{d,␣ew}$')
        axs[2].plot(tt, mag_f2, label='$F_{B,␣ew}$')
        axs[2].plot(tt, Ec_f2, label='$F_{Ec,␣ew}$')
        axs[2].set_ylabel('$a_{ew}$␣[$km/s^2$]', fontsize=13)
        axs[2].grid()
        axs[2].legend(loc='upper␣right')

        axs[3].plot(tt, np.array(v_f2)*1e-3, 'k')
        axs[3].grid()
        axs[3].set_ylabel('$v_{ew}$␣[$km/s$]', fontsize=13)

        axs[4].plot(tt, gr_f3, label='$F_{g,␣pw}$')
```

```
        axs[4].plot(tt, dr_f3, label='$F_{d,␣pw}$')
        axs[4].plot(tt, mag_f3, label='$F_{B,␣pw}$')
        axs[4].plot(tt, Ec_f3, label='$F_{Ec,␣pw}$')
        axs[4].set_ylabel('$a_{pw}$␣[$km/s^2$]', fontsize=13)
        axs[4].grid()
        axs[4].legend(loc='upper␣right')

        axs[5].plot(tt, np.array(v_f3)*1e-3, 'k')
        axs[5].grid()
        axs[5].set_ylabel('$v_{pw}$␣[$km/s$]', fontsize=13)

        axs[6].plot(tt, r/Rs, 'k', label='r')
        axs[6].grid()
        axs[6].set_ylabel('r␣[$R_s$]', fontsize=13)
        axs[6].set_xlabel('t␣[hours]', fontsize=13)
        fig.tight_layout()
        fig.savefig('Field_coordinates_force_components_' +V_txt+str(int(
            Rg0*1e9))+'␣nm␣&'+particle[U][Rg0]['v0_txt'], dpi=500)

for U in U_plot:
    if np.sign(U)<0:
        V_txt='U␣=␣m_'+str(int(abs(U)))+'␣V,␣'
        if Gamma==0:
            V_txt='mag_only_'+V_txt
    elif np.sign(U)>0:
        V_txt='U␣=␣p_'+str(int(abs(U)))+'␣V,␣'
        if Gamma==0:
            V_txt='mag_only_'+V_txt
    else:
        V_txt='drag_only_'

    for Rg0 in R_g0_plot:

        le=int(len(particle[U][Rg0]['r']))
        tt=np.linspace(0, particle[U][Rg0]['t_final'], le)/(60*60)
        gr_mag=particle[U][Rg0]['gr_mag']; drag_mag=particle[U][Rg0]['
            drag_mag']; mag_mag=particle[U][Rg0]['mag_mag']; Ec_mag=
            particle[U][Rg0]['Ec_mag']
        r=particle[U][Rg0]['r']; vtot=particle[U][Rg0]['vtot']; m=
            particle[U][Rg0]['m']; Q=particle[U][Rg0]['Q']; T=particle[U
            ][Rg0]['T']

        fig, axs = plt.subplots(3, 2, figsize=(16,12))
        axs[0,0].plot(tt, gr_mag, label='a_gravity')
        axs[0,0].plot(tt, drag_mag, label='a_drag')
        axs[0,0].plot(tt, mag_mag, label='a_mag')
        axs[0,0].plot(tt, Ec_mag, label='a_Ec')
        axs[0,0].set_title('Forces', fontsize=13)
        axs[0,0].legend()
        axs[0,0].set_ylabel('a␣[km/s^2]', fontsize=13)
        axs[0,0].grid()

        axs[0,1].plot(tt, np.array(m)/particle[U][Rg0]['m0'], 'k')
        axs[0,1].set_title('$m(t)$', fontsize=13)
        axs[0,1].set_ylabel('$m/m_0$', fontsize=13)
        axs[0,1].grid()

        axs[1,0].plot(tt, vtot, 'k')
```

```
axs[1,0].set_ylabel('v␣[km/s]', fontsize=13)
axs[1,0].grid()

axs[1,1].plot(tt, T, 'k')
axs[1,1].set_ylabel('T␣[K]', fontsize=13)
axs[1,1].grid()

axs[2,0].plot(tt, np.array(r)/Rs, 'k')
axs[2,0].set_ylabel('r␣[$R_s$]', fontsize=13)
axs[2,0].plot(tt, np.linspace(1, 1,len(r)))
axs[2,0].set_xlabel('t␣[hours]', fontsize=13)
axs[2,0].grid()

axs[2,1].plot(tt, np.array(Q)/np.array(m) * (amu/q_e))
axs[2,1].set_ylabel('␣Q/m␣[$amu/q_e$]', fontsize=13)
axs[2,1].set_xlabel('t␣[hours]', fontsize=13)
axs[2,1].grid()
fig.tight_layout()
fig.savefig('results_time_'+V_txt+str(int(Rg0*1e9))+'␣nm␣&␣'+
   particle[U][Rg0]['v0_txt'], dpi='figure')
if V_txt=='drag_only_':

    fig, axs = plt.subplots(6, 1, figsize=(16,12))
    #fig.suptitle('Force components: '+V_txt + 'Rg_0 = ' + str
        ("{:.1f}".format(Rg0*1e9)) + ' nm')
    gr_x=[]; gr_y=[]; gr_z=[]; dr_x=[]; dr_y=[]; dr_z=[]; mag_x
        =[]; mag_y=[]; mag_z=[]; Ec_x=[]; Ec_y=[]; Ec_z=[]
    for i in range(le):

        gr_x.append(particle[U][Rg0]['a_gr'][i][0]); gr_y.append(
            particle[U][Rg0]['a_gr'][i][1]); gr_z.append(particle
            [U][Rg0]['a_gr'][i][2]);
        dr_x.append(particle[U][Rg0]['a_drag'][i][0]); dr_y.
            append(particle[U][Rg0]['a_drag'][i][1]); dr_z.append
            (particle[U][Rg0]['a_drag'][i][2]);
        mag_x.append(particle[U][Rg0]['a_mag'][i][0]); mag_y.
            append(particle[U][Rg0]['a_mag'][i][1]); mag_z.append
            (particle[U][Rg0]['a_mag'][i][2]);
        Ec_x.append(particle[U][Rg0]['a_Ec'][i][0]); Ec_y.append(
            particle[U][Rg0]['a_Ec'][i][1]); Ec_z.append(particle
            [U][Rg0]['a_Ec'][i][2]);

    axs[0].plot(tt, gr_x, label='gr_x')
    axs[0].plot(tt, dr_x, label='dr_x')
    axs[0].plot(tt, mag_x, label='mag_x')
    axs[0].plot(tt, Ec_x, label='Ec_x')
    axs[0].set_ylabel('a_x␣[$km/s^2$]', fontsize=13)
    axs[0].set_title('x-components', fontsize=13)
    axs[0].grid()
    axs[0].legend()

    axs[1].plot(tt, np.array(particle[U][Rg0]['vx'])*1e-3)
   # axs[1].set_ylim([-16,16])
    axs[1].set_ylabel('v_x␣[km/s]', fontsize=13)
    axs[1].grid()

    axs[2].plot(tt, gr_y, label='gr_y')
    axs[2].plot(tt, dr_y, label='dr_y')
```

```
            axs[2].plot(tt, mag_y, label='mag_y')
            axs[2].plot(tt, Ec_y, label='Ec_y')
            axs[2].set_ylabel('a_y␣[$km/s^2$]', fontsize=13)
            axs[2].set_title('y-components', fontsize=13)
            axs[2].grid()
            axs[2].legend(loc='upper␣right')

            axs[3].plot(tt, np.array(particle[U][Rg0]['vy'])*1e-3)
            axs[3].grid()
           # axs[3].set_ylim([-16,16])
            axs[3].set_ylabel('v_y␣[$km/s$]', fontsize=13)

            axs[4].plot(tt, gr_z, label='gr_z')
            axs[4].plot(tt, dr_z, label='dr_z')
            axs[4].plot(tt, mag_z, label='mag_z')
            axs[4].plot(tt, Ec_z, label='Ec_z')
            axs[4].set_ylabel('a_z␣[$km/s^2$]', fontsize=13)
            axs[4].set_title('z-components', fontsize=13)
            axs[4].grid()
            axs[4].legend()

            axs[5].plot(tt, np.array(particle[U][Rg0]['vz'])*1e-3)
            axs[5].grid()
            #axs[5].set_ylim([-16,16])
            axs[5].set_ylabel('v_z␣[$km/s$]', fontsize=13)
            fig.tight_layout()
            fig.savefig('force_components_time_' +V_txt+str(int(Rg0*1e9))
               +'␣nm␣&␣'+particle[U][Rg0]['v0_txt'], dpi='figure')


for U in U_plot:
    if np.sign(U)<0:
        V_txt='U␣=␣m_'+str(int(abs(U)))+'␣V,␣'
        if Gamma==0:
            V_txt='mag_only_'+V_txt
    elif np.sign(U)>0:
        V_txt='U␣=␣p_'+str(int(abs(U)))+'␣V,␣'
        if Gamma==0:
            V_txt='mag_only_'+V_txt
    else:
        V_txt='drag_only_'

    if V_txt=='drag_only_':
        fig, axs = plt.subplots(nrows=3, ncols=2, figsize=(14,14))
        axs[2,0].remove()
        axs[2,1].remove()
    else:
        fig, axs = plt.subplots(nrows=3, ncols=2, figsize=(21,14))
        axs[2,1].remove()

    for Rg0 in R_g0_plot:
        if particle[U][Rg0]['v0_txt']=='$v_{kep}$':
            line='-'
        else:
            line='-'
        le=len(particle[U][Rg0]['r'])
        tt=np.arange(0, particle[U][Rg0]['t_final'], particle[U][Rg0]['dt
           '])/(60*60)
        gr_mag=particle[U][Rg0]['gr_mag']; drag_mag=particle[U][Rg0]['
            drag_mag']; mag_mag=particle[U][Rg0]['mag_mag']; Ec_mag=
            particle[U][Rg0]['Ec_mag']
        r=np.array(particle[U][Rg0]['r'])/Rs; vtot=np.array(particle[U][
            Rg0]['vtot']); m=np.array(particle[U][Rg0]['m']); Q=particle[
            U][Rg0]['Q']; T=particle[U][Rg0]['T']
        m0=particle[U][Rg0]['m0']
        Ep_0=1/2*m0*vtot[0]**2 + m0*G*Ms/Rs**2 * (r[0]-r[-1])
        Ep=1/2*m*vtot**2+m*G*Ms/Rs**2 * (r-r[-1])
        if Rg0==R_g0_plot[1]:
            print(m/m0)
        axs[0,0].plot(m/m0, r, linestyle=line, label='$R_0␣=␣$'+str("{:.1
            f}".format(Rg0*1e9))+'␣nm␣&'+particle[U][Rg0]['v0_txt'])
        axs[0,0].set_ylabel('r␣[$R_s$]', fontsize=16)
        axs[0,0].set_xlabel('$m/m_0$', fontsize=16)
        axs[0,0].grid()
        axs[0,0].legend()

        axs[0,1].plot(vtot, r, linestyle=line, label='$R_0␣=␣$'+str("{:.1
            f}".format(Rg0*1e9))+'␣nm␣&'+particle[U][Rg0]['v0_txt'])
        axs[0,1].set_xlabel('v␣[km/s]', fontsize=16)
        axs[0,1].set_ylabel('r␣[$R_s$]', fontsize=16)
        axs[0,1].grid()
        axs[0,1].legend()

        axs[1,0].plot(T, r, linestyle=line, label='$R_0␣=␣$'+str("{:.1f}"
            .format(Rg0*1e9))+'␣nm␣&'+particle[U][Rg0]['v0_txt'])
        axs[1,0].set_ylabel('r␣[$R_s$]', fontsize=16)
        axs[1,0].set_xlabel('T␣[K]', fontsize=16)
        axs[1,0].grid()
        axs[1,0].legend()

        axs[1,1].plot(Ep/Ep_0, r, linestyle=line, label='$R_0␣=␣$'+str("
            {:.1f}".format(Rg0*1e9))+'␣nm␣&'+particle[U][Rg0]['v0_txt'])
        axs[1,1].set_ylabel('r␣[$R_s$]', fontsize=16)
        axs[1,1].set_xlabel('$E/E_0$', fontsize=16)
        axs[1,1].grid()
        axs[1,1].legend()

        if V_txt!='drag_only_':
            axs[2,0].plot((np.array(Q)/np.array(m))/(particle[U][Rg0]['
                Q_0/m_0']), r, linestyle=line, label='$R_0␣=␣$'+str("{:.1
                f}".format(Rg0*1e9))+'␣nm␣&'+particle[U][Rg0]['v0_txt'])
            axs[2,0].set_xlabel('␣($Q/m$)␣/␣($Q_0/m_0$)', fontsize=16)
            axs[2,0].set_ylabel('r␣[$R_s$]', fontsize=16)
            axs[2,0].grid()
            axs[2,0].legend()
    fig.tight_layout()
    fig.savefig('results_r_'+V_txt, dpi='figure', bbox_inches='tight')
```

The code below implements the formulas described in chapter 2.

```
import numpy as np
import scipy
import sympy as sp
from sympy import *
from scipy.optimize import fsolve
from scipy.optimize import bisect
from matplotlib import pyplot as plt
from sympy import symbols, nonlinsolve
from sympy import log, exp, sqrt

def gravity(pos, pos_der, G, Ms):
    x=pos[0]; y=pos[1]; z=pos[2]
    r=np.sqrt(x**2+y**2+z**2)
    a_grav = -G*Ms/r**2

    theta=np.arccos(z/r)
    phi=np.arctan2(y,x)

    return a_grav*np.array([np.sin(theta)*np.cos(phi), np.sin(theta)*np.
        sin(phi), np.cos(theta)])

def rho_a(pos, pos_der, n_half=8.9e13, a1=1/(219*1e3), r_half=2340*1e3,
   Rs=60268*1e3):
    x=pos[0]; y=pos[1]; z=pos[2]
    r=np.sqrt(x**2+y**2+z**2)-Rs
    rho=n_half*np.exp(-a1*(r-r_half))

    return rho

def drag(pos, pos_der, R_grain, rho_m, Gamma, M_H2=2.016e-3, M_ice=18.02e
   -3, NA=6.0221408e+23):
    x=pos[0]; y=pos[1]; z=pos[2]
    vx=pos_der[0]; vy=pos_der[1]; vz=pos_der[2]
    v=np.array([vx, vy, vz])
    A = 3*rho_a(pos, pos_der)*(M_H2/NA)/(4*rho_m*(M_ice/NA)*R_grain)
    v_mag=np.linalg.norm(v)
    a_drag_x=-A*Gamma*v_mag*vx; a_drag_y=-A*Gamma*v_mag*vy; a_drag_z=-A*
        Gamma*v_mag*vz
    #a_drag = -A*Gamma*v_mag**2*v/v_mag
    a_drag=np.array([a_drag_x, a_drag_y, a_drag_z])

    return a_drag

def magnetic(pos, pos_der, Q, m, Rs, Mom, Om):

    x=pos[0]; y=pos[1]; z=pos[2]
    vx=pos_der[0]; vy=pos_der[1]; vz=pos_der[2]

    z_s = 0.04*Rs
    r_s = np.sqrt(x**2+y**2+(z-z_s)**2)

    Bx = 3*Mom*x*(z-z_s)/r_s**5
    By = 3*Mom*y*(z-z_s)/r_s**5
    Bz = Mom*(3*(z-z_s)**2-r_s**2)/r_s**5

    B = np.array([Bx, By, Bz]) #B in cartesian
    v = np.array([vx, vy, vz])

    #phi=np.arctan2(y,x)

    Om_vector=np.array([0,0,Om])
    Om_x_r = np.cross(Om_vector, pos) # xr

    E_c = -Q/m * np.cross(Om_x_r, B) #-Q/m*(  xr  )xB
    VxB = Q/m * np.cross(v, B) #Q/m*VxB

    return E_c, VxB

def vel_RHS(pos, pos_der):

    vel_RHS=[pos_der[0],
             pos_der[1],
             pos_der[2]]

    return np.array(vel_RHS)

def temp(pos, pos_der, v, ep=1, sigma=5.670374e-8, lam=1, mu=18.01528e-3,
    kb=1.380649e-23, Tenv=85, NA=6.0221408e+23, M_H2=2.016e-3):
    rho_atm=rho_a(pos, pos_der)*M_H2/NA
    def func(T, v, rho_atm, ep=1, sigma=5.670374e-8, lam=1, mu=18.01528e
        -3, kb=1.380649e-23, Tenv=85, NA=6.0221408e+23):
        e=np.exp(9.550426-5723.265/T+3.53068*np.log(T)-0.00728332*T)
        L=1000*(2834.1-0.29*T-0.004*T**2)

        return v**3*rho_atm-8*ep*sigma/lam *(T**4-Tenv**4+e*L/(ep*sigma)
            * np.sqrt((mu)/(2*np.pi*kb*NA*T)))

    T_guess=90
    T_sol=fsolve(func, T_guess, args=(v, rho_atm), xtol=1e-5, maxfev
        =5000)

    return T_sol

def mass(T, m0, t, dt, rho_m_mass, mu=18.01528e-3, kb=1.380649e-23, NA
    =6.0221408e+23):
    e=np.exp(9.550426-5723.265/T+3.53068*np.log(T)-0.00728332*T)

    m=(-(4*np.pi)**(1/3)*(3/rho_m_mass)**(2/3)/3 *e*np.sqrt((mu)/(2*np.pi
        *kb*NA*T))*t+m0**(1/3))**3

    #m = m0 - 4*np.pi*(3*m0/(4*np.pi*rho_m_mass))**(2/3)*e*np.sqrt((mu/NA
        )/(2*np.pi*kb*T))*dt

    return m

def temp2(pos, pos_der, v, T_lim, n, ep=1, sigma=5.670374e-8, lam=1, mu
    =18.01528e-3, kb=1.380649e-23, Tenv=85, NA=6.0221408e+23, M_H2=2.016e
    -3):

    rho_atm=rho_a(pos, pos_der)*M_H2/NA

    def f(T, v, rho_atm, ep=1, sigma=5.670374e-8, lam=1, mu=18.01528e-3,
        kb=1.380649e-23, Tenv=85, NA=6.0221408e+23):
        T=np.array(T)
        e=np.exp(9.550426-5723.265/T+3.53068*np.log(T)-0.00728332*T)
        L=1000*(2834.1-0.29*T-0.004*T**2)

        return v**3*rho_atm-8*ep*sigma/lam *(T**4-Tenv**4+e*L/(ep*sigma)
            * np.sqrt((mu)/(2*np.pi*kb*NA*T)))

    T_vec = np.linspace(T_lim[0], T_lim[1], n)
    y = f(T_vec, v, rho_atm)
    roots = []
    for i in range(n-1):
        if y[i]*y[i+1] < 0:
            root = T_vec[i] - (T_vec[i+1] - T_vec[i])/(y[i+1] - y[i])*y[i
                ]
            roots.append(root)

    return roots
```

References

[1] J. E. P. Connerney and J. H. Waite. New model of Saturn's ionosphere with an influx of water from the rings. *Nature*, 312(5990):136–138, November 1984. doi: 10.1038/312136a0.

[2] O. Hamil, T. E. Cravens, N. L. Reedy, and S. Sakai. Fate of ice grains in Saturn's ionosphere. *Journal of Geophysical Research: Space Physics*, 123(2):1429–1440, 2018. doi: https://doi.org/10.1002/2017JA024616.

[3] W.-H. Ip. The exospheric systems of saturn's rings. *Icarus*, 115(2):295–303, 1995. ISSN 0019-1035. doi: https://doi.org/10.1006/icar.1995.1098.

[4] W. H. Ip, C. M. Liu, and K. C. Pan. Transport and electrodynamical coupling of nano-grains ejected from the Saturnian rings and their possible ionospheric signatures. *Icarus*, 276:163–169, September 2016. doi: 10.1016/j.icarus.2016.04.004.

[5] T.T. Koskinen, B.R. Sandel, R.V. Yelle, F.J. Capalbo, G.M. Holsclaw, W.E. McClintock, and S. Edgington. The density and temperature structure near the exobase of Saturn from Cassini UVIS solar occultations. *Icarus*, 226(2):1318–1330, 2013. ISSN 0019-1035. doi: https://doi.org/10.1016/j.icarus.2013.07.037.

[6] D. G. Mitchell, M. E. Perry, D. C. Hamilton, J. H. Westlake, P. Kollmann, H. T. Smith, J. F. Carbary, J. H. Waite, R. Perryman, H.-W. Hsu, J.-E. Wahlund, M. W. Morooka, L. Z. Hadid, A. M. Persoon, and W. S. Kurth. Dust grains fall from Saturn's D-ring into its equatorial upper atmosphere. *Science*, 362(6410): eaat2236, 2018. doi: 10.1126/science.aat2236.

[7] M. W. Morooka, J.-E. Wahlund, L. Z. Hadid, A. I. Eriksson, N. J. T. Edberg, E. Vigren, D. J. Andrews, A. M. Persoon, W. S. Kurth, D. A. Gurnett, W. M. Farrell, J. H. Waite, R. S. Perryman, and M. Perry. Saturn's dusty ionosphere. *Journal of Geophysical Research: Space Physics*, 124(3):1679–1697, 2019. doi: https://doi.org/10.1029/2018JA026154.

[8] T.G. Northrop and Jay Roderick Hill. Inner edge of saturn's b ring. *Journal of Geophysical Research*, 88(A8):6102 – 6108, 1983. doi: 10.1029/JA088iA08p06102. Cited by: 41.

[9] T. Vondrak, J. M. C. Plane, S. Broadley, and D. Janches. A chemical model of meteoric ablation. *Atmospheric Chemistry & Physics*, 8(23):7015–7031, December 2008. doi: 10.5194/acp-8-7015-2008.

[10] S.-Y. Ye, W. S. Kurth, G. B. Hospodarsky, A. M. Persoon, A. H. Sulaiman, D. A. Gurnett, M. Morooka, J.-E. Wahlund, H.-W. Hsu, Z. Sternovsky, X. Wang, M. Horanyi, M. Seiß, and R. Srama. Dust observations by the radio and plasma wave science instrument during cassini's grand finale. *Geophysical Research Letters*, 45 (19):10,101–10,109, 2018. doi: https://doi.org/10.1029/2018GL078059.