Debris Model Documentation

This documentation explains the code (and models it uses) for the satellite-debris model. Written by Justin Lawrence (lawren17@student.ubc.ca). Last updated 07/09/2022.

1 JASON N-Cell Model

The models used in this project are all built on the following variant of one produced by JASON [1]. It's recommended that you read this section before continuing, even if you have experience with the JASON model.

1.1 JASON Model

The base model for the evolution of the orbital debris system was taken from a JASON report [1] on the topic, and models the evolution in a $\approx 25km$ orbital band. It has variables

S : number of live satellites
D : number of derelict satellites
N : number of catastrophically lethal debris

(see section 5.1 for details on the types of debris) and constant parameters

λ : satellite launch rate
Δt : mean satellite lifetime
σ : satellite cross-section
v : relative collision speed
δ : ratio of density of non-catastrophic to catastrophic lethal debris
α : fraction of collisions that a live satellite fails to avoid
P : post-mission disposal probability
τ : atmospheric drag lifetime
N_0 : number of catastrophic debris fragments from a collision
V : volume of the spherical shell

Default values for these constants are given in [1], and are generally kept unless otherwise noted. The relative collision speed is calculated based on

the altitude of the band as specified in section 4.3. The continuous model is then given as

$$\dot{S} = \lambda - S/\Delta t - (\delta + \alpha) n \sigma v S \tag{1}$$

$$\dot{D} = (1 - P) S/\Delta t + \delta n \sigma v S - n \sigma v D - D/\tau \tag{2}$$

$$\dot{N} = n \sigma v N_0 (\alpha S + D) + n_0 V/\tau - N/\tau \tag{3}$$

where $n = \frac{N + D/2}{V}$ (the origin of the factor of $1/2$ is unclear, and this factor is not carried forwards to the models in later sections), and $n_0 = n(t = 0)$.

1.2 Continuous N-Cell Extension

For the purposes of this documentation, we refer to an atmospheric layer as a cell. The JASON model only accounts for one atmospheric band. To expand this to n atmospheric bands, we look at the evolution of the vectors

$$\mathbf{x}_i = \begin{pmatrix} S_i \\ D_i \\ N_i \\ C_i \end{pmatrix}$$

where S_i, D_i, N_i are the number of live satellites, derelict satellites, and catastrophic debris in the ith shell. C_i is a count of the total number of collisions that occur in the shell. Shells are numbered from lowest to highest elevation, and there's assumed to be no gap between bands. The derived model is then $\frac{d\mathbf{x}_i}{dt} = \mathbf{f}(\mathbf{x}_i) + \mathbf{g}(\mathbf{x}_{i+1})$, where

$$\mathbf{f}(\mathbf{x}_i) = \begin{pmatrix} \lambda_i - S_i/\Delta t_i - (\delta_i + \alpha_i) n_i \sigma_i v_i S_i \\ (1 - P_i) S_i/\Delta t_i + \delta_i n_i \sigma_i v_i S_i - n_i \sigma_i v_i D_i - D_i/\tau_i \\ n_i \sigma_i v_i N_{0i} (\alpha_i S_i + D_i) - N_i/\tau_i \\ (\delta_i + \alpha_i) n_i \sigma_i v_i S_i + n_i \sigma_i v_i D_i \end{pmatrix} \tag{4}$$

As can be seen in the indices, each shell is allowed to have its own values for the constant parameters (in fact the decay lifetime is determined based on the altitude of the shell, see section 4). This is simply the JASON model less the higher altitude debris moving down, which will be handled in $\mathbf{g}$. For $\mathbf{g}$, there are two cases. If $i = n$ ($\mathbf{x}_{n+1}$ doesn't actually exist, but given the $\mathbf{g}$ here that's not a problem), then we simply have

$$\mathbf{g}(\mathbf{x}_{i+1}) = \begin{pmatrix} 0 \\ 0 \\ n_{0i} V_i/\tau_i \\ 0 \end{pmatrix} \tag{5}$$

essentially, we make the same approximation as the JASON model. If $i \neq n$, we can actually use the number of derelicts and catastrophic debris coming in from the above shell via orbital decay to get

$$\mathbf{g}(\mathbf{x}_{i+1}) = \begin{pmatrix} 0 \\ D_{i+1}/\tau_{i+1} \\ N_{i+1}/\tau_{i+1} \\ 0 \end{pmatrix} \tag{6}$$

2 Minimal Fractional Table Model

The fractional table model is a continuous model built on the N-Cell JASON model, and extending it to include multiple satellite types, explosions, rocket bodies, and other factors. This section assumes you've read the section 1.

2.1 Model Parameters

2.1.1 Satellites

Unlike the previous N-Cell extension, not all satellites in a cell are assumed to have the same properties. In every cell, we keep track of a list of satellite types (the same types being tracked in all cells). For each satellite type in

each shell we keep track of

S : number of live satellites
D : number of derelict satellites
λ : satellite launch rate into that shell
Δt : mean live satellite lifetime
σ : satellite cross-section
m_s : satellite mass
A/m : satellite area-to-mass ratio
α_S : fraction of collisions that a live satellite fails to avoid with another live satellite
α_D : fraction of collisions that a live satellite fails to avoid with a derelict
α_N : fraction of collisions that a live satellite fails to avoid with trackable debris
α_R : fraction of collisions that a live satellite fails to avoid with rocket bodies
P : post-mission disposal probability
τ : atmospheric drag lifetime
C : fit constant for explosions, as defined in section 5.1
E_L : the number of live satellites (out of a population of 100) that explode in 1yr
E_D : the number of derelict satellites (out of a population of 100) that explode in 1yr

S, D are the parameters being tacked over time, while the others are generally constant, user chosen parameters. The exception to this is τ, which is updated periodically based on the current atmospheric conditions (see section 4). If we assume that satellites don't explode, then the value of C is irrelevant. As a bit of clarification, Δt is the mean amount of time it takes for a live-satellite to begin to de-orbit (removing it from the system).

2.1.2 Rocket Bodies

Like for satellites, in each cell we keep track of a list of rocket body types (the same types being tracked in all cells). All rocket bodies are treated like discarded or expended stages, and hence treated like a derelict satellite. For

every rocket body type we keep track of

R : number of rocket bodies
λ : rocket body launch rate
σ : rocket body cross-section
m_{rb} : rocket body mass
A/m : rocket body area-to-mass ratio
τ : atmospheric drag lifetime
C : fit constant for explosions, as defined in section 5.1
E_R : the number of rocket bodies (out of a population of 100) that explode in 1yr

where here R is tracked over time, and the other parameters are user-defined. τ is calculated periodically using the atmospheric model in section 4.

2.1.3 Debris

Unlike the JASON model, the fractional table model attempts to track all lethal debris, both catastrophic and non-catastrophic. To this end, debris is binned by its characteristic length L_C and area-to-mass ratio A/m, resulting in a 2D matrix of debris $\overleftrightarrow{N}$ for each cell. It's assumed that debris with $L_C \geq 10cm$ is "trackable" by satellite or Earth-based systems, and hence can be avoided, while smaller debris cannot. A separate decay lifetime is $\tau_{i,j}$ is calculated for the debris of each bin in each cell, using the average A/m of that bin and the atmospheric model in section 4. Binning is done using a linear logarithmic scale for both L_C and A/m, and average values are also calculated logaritmically. Generally, the range of $\log_{10}(A/m)$ values is taken to be -2 to 1, as this was found to contain effectively all debris output from collisions and explosions. The range of L_C values is taken to be $1mm$ to $1m$. Debris larger than $1m$ is exceedingly unlikely for collisions, and debris smaller than $1mm$ is assumed to be too small to be lethal.

2.1.4 Discrete Events

Finally, each cell also has a list of discrete events that occur in the cell. These can generally change anything about the cell (amount of satellites, rocket bodies, debris, etc.), and either occur with a specific frequency of at specific times. These are meant to represent things like anti-satellite weapons tests.

2.2 Model Description

2.2.1 Rates of Change

Let $S_{i,j}$ be the amount of live satellites in the ith cell of the jth type, and define $D_{i,j}, R_{i,j}$ similarly. The other parameters for satellites and rocket bodies are indexed in a similar manner. Let $\overleftrightarrow{N}_{i,j,k}$ be the amount of debris in the ith cell in the (j,k)th bin. For collisions between satellites/rocket bodies with cross sections σ_j, σ_k, we take the collision cross-section to be

$$\sigma_{j,k} = \sigma_j + \sigma_k + 2\sqrt{\sigma_j \sigma_k} \tag{7}$$

which is the correct result if both objects are roughly spherical. Following the example of the JASON model, the rate of collision between live satellites would be

$$dR_{SSi,j,k} = \frac{\alpha_{Sj}\alpha_{Sk}\sigma_{j,k}v_i S_{i,j}S_{i,k}}{V_i} \tag{8}$$

The rate of collisions between live satellites of type j and derelict satellites of type k would be

$$dR_{SDi,j,k} = \frac{\alpha_{Dj}\sigma_{j,k}v_i S_{i,j}D_{i,k}}{V_i} \tag{9}$$

The rate of collisions between derelict satellites of type j and k would be (as derelicts cannot avoid each other)

$$dR_{DDi,j,k} = \frac{\sigma_{j,k}v_i D_{i,j}D_{i,k}}{V_i} \tag{10}$$

For the purposes of collisions, rocket bodies behave in the same manner as derelict satellites. Hence the rate of collisions between live/derelict satellites of type j and rocket bodies of type k would be

$$dR_{SRi,j,k} = \frac{\alpha_{Rj}\sigma_{j,k}v_i S_{i,j}R_{i,k}}{V_i} \tag{11}$$

$$dR_{DRi,j,k} = \frac{\sigma_{j,k}v_i D_{i,j}R_{i,k}}{V_i} \tag{12}$$

Finally, the rate of collisions between rocket bodies of type j and k would be

$$dR_{RRi,j,k} = \frac{\sigma_{j,k}v_i R_{i,j}R_{i,k}}{V_i} \tag{13}$$

Similarly, we can get that the rate of collisions between satellites (live, derelict) and rocket bodies of type j with debris from bin (k, l) would be

$$dR_{Si,j,k,l} = \begin{cases} \frac{\alpha_{Nj}\sigma_j v_i S_{i,j} \overleftrightarrow{N}_{i,k,l}}{V_i} & L_C \geq 10cm \\ \frac{\sigma_j v_i S_{i,j} \overleftrightarrow{N}_{i,k,l}}{V_i} & L_C < 10cm \end{cases} \tag{14}$$

$$dR_{Di,j,k,l} = \frac{\sigma_j v_i D_{i,j} \overleftrightarrow{N}_{i,k,l}}{V_i} \tag{15}$$

$$dR_{Ri,j,k,l} = \frac{\sigma_j v_i R_{i,j} \overleftrightarrow{N}_{i,k,l}}{V_i} \tag{16}$$

The rate of explosions of satellites (live/derelict) of type j in the ith cell is given by

$$dE_{Si,j} = E_{Lj}\frac{S_{i,j}}{100} \tag{17}$$

$$dE_{Di,j} = E_{Dj}\frac{D_{i,j}}{100} \tag{18}$$

and similarly for rocket bodies

$$dE_{Ri,j} = E_{Rj}\frac{R_{i,j}}{100} \tag{19}$$

Let τ_D represent the decay lifetime of derelict satellites, τ_R rocket bodies, and τ debris. Then following the JASON model the rate at which derelicts/rocket bodies decay out of a shell is given by

$$K_{Di,j} = \frac{D_{i,j}}{\tau_{Di,j}} \tag{20}$$

$$K_{Ri,j} = \frac{R_{i,j}}{\tau_{Ri,j}} \tag{21}$$

Similarly, the rate of decay of debris in the ith cell and (j,k)th bin is

$$K_{Ni,j,k} = \frac{\overleftrightarrow{N}_{i,j,k}}{\tau_{i,j,k}} \tag{22}$$

Let $\mathcal{S}$ be the set of all indices for satellite types, $\mathcal{R}$ be the set of all indices for rocket body types, $\mathcal{N}$ be the set of all indices for debris bins, let ζ be a factor equal to 1 if a collision is catastrophic and zero otherwise (the logic

for this decision can be found in section 5.1), and let ζ' be the opposite of ζ. Then it follows from the NCell JASON model that

$$\frac{dS_{i,j}}{dt} = \lambda_{i,j} - \frac{S_{i,j}}{\Delta t_{i,j}} - 2dR_{SSi,j,j} - \sum_{k\in\mathcal{S},k\neq j} dR_{SSi,j,k} - \sum_{k\in\mathcal{S}} dR_{SDi,j,k} - \sum_{k\in\mathcal{R}} dR_{SRi,j,k} - \sum_{(k,l)\in\mathcal{N}} dR_{Si,j,k,l} - dE_{Si,j} \quad (23)$$

For derelicts we have that

$$\frac{dD_{i,j}}{dt} = (1-P_{i,j})\frac{S_{i,j}}{\Delta t_{i,j}} + K_{Di+1,j} + \sum_{(k,l)\in\mathcal{N}} \zeta'_{j,k,l} dR_{Si,j,k,l} - K_{Di,j} - \sum_{k\in\mathcal{S}} dR_{SDi,k,j} - \sum_{k\in\mathcal{S},k\neq j} dR_{DDi,j,k} - 2dR_{DDi,j,j} - \sum_{k\in\mathcal{R}} dR_{DRi,j,k} - \sum_{(k,l)\in\mathcal{N}} \zeta_{j,k,l} dR_{Di,j,k,l} - dE_{Di,j} \quad (24)$$

where $K_{Di+1,j} = 0$ for the top cell. Similarly, we get for rocket bodies that

$$\frac{dR_{i,j}}{dt} = \lambda_{Ri,j} + K_{Ri+1,j} - K_{Ri,j} - \sum_{k\in\mathcal{S}} dR_{SRi,k,j} - \sum_{k\in\mathcal{S}} dR_{DRi,k,j} - \sum_{k\in\mathcal{R},k\neq j} dR_{RRi,j,k} - 2dR_{RRi,j,j} - \sum_{(k,l)\in\mathcal{N}} \zeta_{j,k,l} dR_{Ri,j,k,l} - dE_{Ri,j} \quad (25)$$

where $K_{Ri+1,j} = 0$ for the top cell, and λ_R is the rate of rocket bodies being launched. Let $N_{i,j,k}()$ be a function which computes the rate of debris being generated in the ith cell and (j,k)th bin by a rate of collisions/explosions, which is generally obtained by multiplying the rate with the correct probability table. For details, see sections 2.2.2-2.2.4. Let $\mathcal{H}$ be the set of all cell

indices. Then we'd have that

$$\frac{d\overleftrightarrow{N}_{i,j,k}}{dt} = K_{Ni+1,j,k} + \sum_{l\in\mathcal{H}} \Bigg(\sum_{m\in\mathcal{S}} \Big(\sum_{n\in\mathcal{S}} N_{i,j,k}(dR_{SDl,m,n}) + \sum_{n=m}^{\max(S)} (N_{i,j,k}(dR_{SSl,m,n}) + N_{i,j,k}(dR_{DDl,m,n})) + \sum_{n\in\mathcal{R}} (N_{i,j,k}(dR_{SRl,m,n}) + N_{i,j,k}(dR_{DRl,m,n})) + \sum_{(n,o)\in\mathcal{N}} (N_{i,j,k}(dR_{Sl,m,n,o}) + N_{i,j,k}(dR_{Dl,m,n,o})) + N_{i,j,k}(dE_{Sl,m}) + N_{i,j,k}(dE_{Dl,m}) \Big) + \sum_{m\in\mathcal{R}} \Big(N_{i,j,k}(dE_{Rl,m}) + \sum_{n=m}^{\max(\mathcal{R})} N_{i,j,k}(dR_{RRl,m,n}) + \sum_{(n,o)\in\mathcal{N}} N_{i,j,k}(dR_{Rl,m,n,o}) \Big) \Bigg) - K_{Ni,j,k} - \sum_{l\in\mathcal{S}} (dR_{Si,l,j,k} + dR_{Di,l,j,k}) - \sum_{l\in\mathcal{R}} dR_{Ri,l,j,k} \quad (26)$$

where indices are sometimes started short (i.e. the $n = m$ lines) to avoid double counting. For the top cell, it's assumed that $K_{Ni+1,j,k} = K_{Ni,j,k}(t = 0)$ or zero, it's up to whoever runs the model.

2.2.2 Calculating the Probability Tables

The probability table exploits something fundamental from the NASA breakup model [2]. Namely, that the random draws for characteristic length, area-to-mass ratio, and ejection velocity of the debris generated by a collision or explosions are only dependent on:

1. Whether the debris is generated by a collision or explosion.
2. Whether it's a satellite or rocket body involved.

Hence, for each cell, we can pre-compute four probability tables representing the probability that a generated piece of debris from that cell (from either a satellite or rocket body in either a collision or explosion) has certain final altitude, characteristic length, and cross section ranges.

For any given initial and final shell, we use the vis-viva equation

$$v^2 = GM\Big(\frac{2}{r} - \frac{1}{a}\Big) \quad (27)$$

with the current orbital radius r, and range of semi-major axes inside the shell (i.e. range of valid a values) to compute the range of final velocities

for debris going into the shell. In the above equation, M is the mass of the Earth. If we call the final velocity v', the pre-collision orbital velocity v_0 (assuming a circular orbit in the centre altitude of the shell), and the ejection velocity Δv, then we get for any particular direction that

$$v'^2 = (v_0 + \sin(\theta)\cos(\phi)(\Delta v))^2 + \sin^2(\theta)\sin^2(\phi)(\Delta v)^2 + \cos^2(\theta)(\Delta v)^2 \quad (28)$$

(see section 5.2.3), where since we've assumed a circular orbit

$$v_0 = \sqrt{\frac{GM}{r}} \quad (29)$$

where G the gravitational constant. Equation (28) can be put into the much more simplified form

$$v'^2 = v_0^2 + 2v_0(\Delta v)\sin(\theta)\cos(\phi) + (\Delta v)^2$$

which has the solutions

$$\Delta v = -v_0 \sin(\theta)\cos(\phi) \pm \sqrt{v_0^2 \sin^2(\theta)\cos^2(\phi) - (v_0^2 - v'^2)} \quad (30)$$

where of course we place the restriction that $\Delta v \geq 0$. We'll come back to this equation in a bit. For now, it suffices to note that for any given direction and v', we can compute the values of Δv for which v' has the desired value. Hence, since we have a PDF for $\nu = \log_{10}(\Delta v)$, we can define a PDF for v'.

The procedure for calculating a given table is then as follows. Let p be a PDF (probability distribution) and P the corresponding CDF (cumulative distribution). Pick an initial shell, and call it the nth shell. Pick a final shell, and call it the ith shell. Let $v_{i,min}, v_{i,max}$ be the range of final velocities for which debris ends up in the ith shell. Then for any given direction (θ, ϕ), we'd clearly have that

$$T[n,i,j,k] = \int_{L_j}^{L_{j+1}} \int_{\chi_k}^{\chi_{k+1}} \int_{v_{i,min}}^{v_{i,max}} p(L)p(\chi|L)p(v'|\chi,\theta,\phi)dv'd\chi dL \quad (31)$$

where T is the table, characteristic length and area-to-mass ratio are integrated from one bin edge to the next, and $\chi = \log_{10}(A/M)$. Actually computing this would be quite slow, so we take the following approximation

$$T[n,i,j,k] = (P(L_{j+1}) - P(L_j))(P(\chi_{j+1}|L^*) - P(\chi_j|L^*))(P(v_{i,max}|L^*,\chi^*) - P(v_{i,min}|L^*,\chi^*)) \quad (32)$$

where L^*, χ^* are the (logarithmically) averaged values of L, χ in the bin respectively. We remove the θ, ϕ dependence by integrating over all directions, using a Monte-Carlo method to with points uniformly chosen over the sphere using a Fibbonacci spiral method.

The final thing to look at is the specifics of the cumulative probability distribution for v'. Taking a look at the equation (30), we can see three possible cases.

Case 1 : The equation has no real, positive solutions. Then the desired final velocity is impossible, so since lowering v' only makes the the larger solution for Δv smaller, the cumulative probability is zero.

Case 2 : The equation as one real positive root (which must be the larger root). Then if the real root is Δv_{max}, the cumulative probability is just $P(\Delta v_{max})$ (the cumulative probability in terms of Δv, as is described in section 5.1).

Case 3 : The equation has two real positive roots. Lowering v' makes the larger solution smaller and the smaller solution larger, so any Δv between the two solutions is valid. Hence, if the smaller solution is Δv_{min}, we return $P(\Delta v_{max}) - P(\Delta v_{min})$.

It's also good to note that occasionally the vis-viva equation will result in a value $v'^2 < 0$, which is clearly impossible. If $v_{min}^2 < 0$ and $v_{max}^2 \geq 0$, we take $v_{min} = 0$. If $v_{max}^2 < 0$ as well, we just say that this shell cannot be reached by debris. Note that since some of the debris will generally escape the system, the table may not be normalized.

2.2.3 Handling Satellite-Rocket Body Collisions

In the previous section, we assumed that a collision involves either a satellite or a rocket body, but not both. This of course begs the question of what to do if a satellite collides with a rocket body. The NASA breakup model doesn't prescribe anything for this, so we had to come up with another method. The method used is to treat the collision as two separate, catastrophic collisions, one between the satellite and a negligible piece of debris and one between the rocket body and a negligible piece of debris. This has no explicit physical or theoretical motivation, and was simply chosen because it seems reasonable.

2.2.4 Using the Probability Table

In order to use a probability table to compute the $N_{i,j,k}()$ function, we use the following procedure

1. Calculate the total amount of debris generated by the event using equation (57) or (60), depending on the event type and using the parameters of the objects involved in the collision.
2. Take this number, and multiply by $T[n, i, j, k]$ using the correct table for that event type.

2.2.5 Model Assumptions

This section lists some important assumptions made by the model:

1. Launched bodies immediately reach their target altitude for orbit.
2. Satellites de-orbit immediately, with a negligible chance of collision.
3. All objects are in a roughly circular orbit.
4. Objects from outside the region of atmosphere analyzed have a negligible impact on the system.
5. Satellites, rocket bodies, and debris are homogeneously spread out in each cell.

2.3 Implementation

2.3.1 Program Structure

The overall structure of the program, written in Python3, consists of 3 main classes.

Event (and derived classes) : Contains all the information about a given discrete event, as well as the function to simulate the event occurring.

Cell : Represents one section of the atmosphere. Contains all relevant information about that layer (volume, width, altitude, etc.), the all information on the satellites and rocket bodies of every type for that layer, and the list of all discrete events occurring in that layer. Also contains a function for calculating the rates of change/collisions/explosions for satellites, rocket

bodies, and debris in the layer.

NCell : Contains a list of Cells representing the entire atmosphere. Contains functions for running the simulation, calculating probability tables, functions to simulate collisions and explosions, and generally manages all of the Cells.

Generally, only NCell and Event ever need to be interacted with by the user. NCell is capable of creating its own Cell instances based on its initializing arguments, but since discrete events are handled on a case-by-case basis Event instances must be given to it by the user.

2.3.2 Implementing Collisions/Explosions Efficiently

To avoid unnecessary repeated calculations, the rate of collisions/explosions of each type is counted up fully for each cell before simulating the collisions. For example, take a collision type between satellite types j, k in the ith cell. These would all have the same average number of debris and debris distribution, so we count up the total amount of collisions of this type (i.e. collisions between live and derelict) satellites of these types, then simply multiply this rate by the corresponding distribution.

2.3.3 Implementing Discrete Events

Discrete events are treated as happening (and effecting the system) instantaneously, and hence return changes in relevant values (such as the amount of debris, or a discrete number of collisions) rather than as rates of change. Events are simulated after a time step is run, and so don't have to be undone for adaptive time steps.

2.3.4 Numerical Methods

There are two numerical solution methods included in NCell. One is just a basic Euler method. The other is a predictor-corrector (PC) method with an adaptive time step, which uses a 2-step Adams-Bashforth method (AB(2) method) along with the trapezoid method. The AB(2) method, with an adaptive time step, for some general differential equation $\mathbf{y}'(t) = \mathbf{f}(t, \mathbf{y})$, is [3]

$$\mathbf{y}_{n+2} = \mathbf{y}_{n+1} + \frac{1}{2} h_{n+1} \left[\left(2 + \frac{h_{n+1}}{h_n} \right) \mathbf{y}'_{n+1} - \frac{h_{n+1}}{h_n} \mathbf{y}'_n \right] \tag{33}$$

where $h_i = t_{i+1} - t_i$. The trapezoid rule is given by [3]

$$\mathbf{y}_{n+2} = \mathbf{y}_{n+1} + \frac{1}{2} h_{n+1}(\mathbf{y}'_{n+2} + \mathbf{y}'_{n+1}) \tag{34}$$

The general form of PC methods used here is also given in [3]. An estimate of the error ϵ of this method is given by

$$\epsilon = -\frac{1}{3} \frac{h_{n+1}}{h_{n+1} + h_n} (\mathbf{y}_{n+2} - \mathbf{y}^{[0]}_{n+2}) \tag{35}$$

where $\mathbf{y}^{[0]}_{n+2}$ is the value of $\mathbf{y}_{n+2}$ predicted by the AB(2) method. If $|\epsilon| \leq \text{tol}$, an arbitrary tolerance, then we accept the step and move on. If not, we retry the step. In either case, we take the new h to be [3]

$$h_{new} = h_{n+1} \left| \frac{\text{tol}}{\epsilon} \right|^{\frac{1}{3}} \tag{36}$$

(note that this equation was technically derived for usage with just a trapezoid method, but it still seems reasonable to use in this case). In the implementation of this method, a tolerance of 1 was found to be reasonable in most cases, although it occasionally results in time steps that are slightly too large at the start.To prevent the time step from getting too large, an arbitrary minimum/maximum time step (adjustable by the user) were added as well.

The AB(2) method needs 2 sets of initial conditions, whereas we only have 1. The second set is generated by the Euler method using the minimum time step. In general, the Euler method is less accurate and stable than the PC method, along with not having an adaptive time step, so we highly recommend that you use the PC method instead.

2.3.5 Saving/Loading Data

All the relevant data of a NCell system can be saved, and a new NCell object can be generated using that data with the save and load methods. Events cannot be saved. See code docstrings for more details.

3 Improved Fractional Table Models

There are two improved versions of the basic fractional table model that we created, and both are outlined here.

3.1 De-orbit/No-Ascent Fractional Table Model

3.1.1 Model Description

The de-orbit variant of the fractional model takes into account the time it takes for satellites to de-orbit and burn up in the atmosphere (removing the second assumption of the basic fractional model). To that end, we add the following two satellite parameters to track

$$\begin{aligned} S_d &: \text{ number of live de-orbiting satellites} \\ \tau_{do} &: \text{ live satellite de-orbiting lifetime} \end{aligned}$$

S_d is tracked over time and τ_{do} is generally a user-defined parameter. We then also need to calculate the rate at which de-orbiting satellites are colliding with other bodies, giving (using the same notation as the previous section)

$$dR_{SS_{di,j,k}} = \frac{\alpha_{Sj}\alpha_{Sk}\sigma_{j,k}v_i S_{i,j}S_{di,k}}{V_i} \tag{37}$$

$$dR_{S_dS_{di,j,k}} = \frac{\alpha_{Sj}\alpha_{Sk}\sigma_{j,k}v_i S_{di,j}S_{di,k}}{V_i} \tag{38}$$

$$dR_{S_dD_{i,j,k}} = \frac{\alpha_{Dj}\sigma_{j,k}v_i S_{di,j}D_{i,k}}{V_i} \tag{39}$$

$$dR_{S_dR_{i,j,k}} = \frac{\alpha_{Rj}\sigma_{j,k}v_i S_{di,j}R_{i,k}}{V_i} \tag{40}$$

$$dR_{S_{di,j,k,l}} = \begin{cases} \frac{\alpha_{Nj}\sigma_j v_i S_{di,j}\overleftrightarrow{N}_{i,k,l}}{V_i} & L_C \geq 10cm \\ \frac{\sigma_j v_i S_{di,j}\overleftrightarrow{N}_{i,k,l}}{V_i} & L_C < 10cm \end{cases} \tag{41}$$

We also need to calculate the rate at which de-orbiting satellites explode, which is given by (we classify de-orbiting satellites as a type of live satellite for this purpose)

$$dE_{S_{di,j}} = E_{Lj}\frac{S_{di,j}}{100} \tag{42}$$

Finally, we account for the fact that de-orbiting satellites exit each cell at a rate of

$$K_{S_{di,j}} = \frac{S_{di,j}}{\tau_{doi,j}} \tag{43}$$

Taking all of this into account, our updated rates of change are then

$$\frac{dS_{i,j}}{dt} = \lambda_{i,j} - \frac{S_{i,j}}{\Delta t_{i,j}} - 2dR_{SSi,j,j} - \sum_{k\in\mathcal{S},k\neq j} dR_{SSi,j,k} - \sum_{k\in\mathcal{S}} dR_{SSdi,j,k} - \sum_{k\in\mathcal{S}} dR_{SDi,j,k} - \sum_{k\in\mathcal{R}} dR_{SRi,j,k} - \sum_{(k,l)\in\mathcal{N}} dR_{Si,j,k,l} - dE_{Si,j} \quad (44)$$

for S and

$$\frac{dS_{di,j}}{dt} = P_{i,j}\frac{S_{i,j}}{\Delta t_{i,j}} + K_{Sdi+1,j} - K_{Sdi,j} - \sum_{k\in\mathcal{S}} dR_{SSdi,k,j} - 2dR_{SdSdi,j,j} - \sum_{k\in\mathcal{S},k\neq j} dR_{SdSdi,j,k} - \sum_{k\in\mathcal{S}} dR_{SdDi,j,k} - \sum_{k\in\mathcal{R}} dR_{SdRi,j,k} - \sum_{(k,l)\in\mathcal{N}} dR_{Sdi,j,k,l} - dE_{Sdi,j} \quad (45)$$

for S_d, where $K_{Sdi+1,j} = 0$ for the top cell. We get that

$$\frac{dD_{i,j}}{dt} = (1-P_{i,j})\frac{S_{i,j}}{\Delta t_{i,j}} + K_{Di+1,j} + \sum_{(k,l)\in\mathcal{N}} \zeta'_{j,k,l}(dR_{Si,j,k,l} + dR_{Sdi,j,k,l}) - K_{Di,j} - \sum_{k\in\mathcal{S}} dR_{SDi,k,j} - \sum_{k\in\mathcal{S}} dR_{SdDi,k,j} - \sum_{k\in\mathcal{S},k\neq j} dR_{DDi,j,k} - 2dR_{DDi,j,j} - \sum_{k\in\mathcal{R}} dR_{DRi,j,k} - \sum_{(k,l)\in\mathcal{N}} \zeta_{j,k,l} dR_{Di,j,k,l} - dE_{Di,j} \quad (46)$$

where again $K_{Di+1,j} = 0$ for the top cell. Similarly, for rocket bodies we get

$$\frac{dR_{i,j}}{dt} = \lambda_{Ri,j} + K_{Ri+1,j} - K_{Ri,j} - \sum_{k\in\mathcal{S}} dR_{SRi,k,j} - \sum_{k\in\mathcal{S}} dR_{SdRi,k,j} - \sum_{k\in\mathcal{S}} dR_{DRi,k,j} - \sum_{k\in\mathcal{R},k\neq j} dR_{RRi,j,k} - 2dR_{RRi,j,j} - \sum_{(k,l)\in\mathcal{N}} \zeta_{j,k,l} dR_{Ri,j,k,l} - dE_{Ri,j} \quad (47)$$

where $K_{Ri+1,j} = 0$ for the top cell, and λ_R is the rate of rocket bodies being launched. Finally, the updated rate of change of debris would be

$$\frac{d\overleftrightarrow{N}_{i,j,k}}{dt} = K_{Ni+1,j,k} + \sum_{l\in\mathcal{H}} \Bigg(\sum_{m\in\mathcal{S}} \Big(\sum_{n\in\mathcal{S}} (N_{i,j,k}(dR_{SS_dl,m,n}) + N_{i,j,k}(dR_{SDl,m,n}) + N_{i,j,k}(dR_{S_dDl,m,n})) + \sum_{n=m}^{\max(S)} (N_{i,j,k}(dR_{SSl,m,n}) + N_{i,j,k}(dR_{S_dS_dl,m,n}) + N_{i,j,k}(dR_{DDl,m,n})) + \sum_{n\in\mathcal{R}} (N_{i,j,k}(dR_{SRl,m,n}) + N_{i,j,k}(dR_{S_dR_l,m,n}) + N_{i,j,k}(dR_{DRl,m,n})) + \sum_{(n,o)\in\mathcal{N}} (N_{i,j,k}(dR_{Sl,m,n,o}) + N_{i,j,k}(dR_{S_dl,m,n,o}) + N_{i,j,k}(dR_{Dl,m,n,o})) + N_{i,j,k}(dE_{Sl,m}) + N_{i,j,k}(dE_{S_dl,m}) + N_{i,j,k}(dE_{Dl,m}) \Big) + \sum_{m\in\mathcal{R}} \Big(N_{i,j,k}(dE_{Rl,m}) + \sum_{n=m}^{\max(\mathcal{R})} N_{i,j,k}(dR_{RRl,m,n}) + \sum_{(n,o)\in\mathcal{N}} N_{i,j,k}(dR_{Rl,m,n,o}) \Big) \Bigg) - K_{Ni,j,k} - \sum_{l\in\mathcal{S}} (dR_{Si,l,j,k} + dR_{S_di,l,j,k} + dR_{Di,l,j,k}) - \sum_{l\in\mathcal{R}} dR_{Ri,l,j,k} \quad (48)$$

3.1.2 Implementation

The implementation of the de-orbiting variant of the fractional table model is just an updated version of the same code. The basic structure and methods are not changed. Implementation remains in Python3.

3.2 Full Fractional Table Model

3.2.1 Model Description

The full fractional model also takes into account the time it takes for satellites to ascend into their target orbits after being launched. To that end, we add the following three satellite parameters to track

a_t : target altitude of the satellite

τ_{up} : amount of time it takes for an ascending satellite to ascend through a band

τ_{fail} : failure lifetime of ascending satellites

Furthermore, λ is no longer associated with a particular shell, as all satellite types are launched into the bottom shell before ascending to their target

altitude. All of these are user-defined parameters. The rate of live satellites rising is given by

$$K_{S_{i,j}} = \begin{cases} \frac{S_{i,j}}{\tau_{up_{i,j}}} & \text{if } a_t \text{ is above cell } i \\ 0 & \text{otherwise} \end{cases} \tag{49}$$

The rate at which live satellites become derelicts/de-orbiting satellites is also changed, and is now given by

$$Q_{i,j} = \begin{cases} \frac{S_{i,j}}{\tau_{fail_{i,j}}} & \text{if } a_t \text{ is above cell } i \\ \frac{S_{i,j}}{\Delta t_i} & \text{otherwise} \end{cases} \tag{50}$$

Taking all of this into account, our updated rates of change are then

$$\begin{aligned} \frac{dS_{i,j}}{dt} = K_{Si-1,j} - K_{Si,j} - Q_{i,j} - 2dR_{SSi,j,j} - \sum_{k\in\mathcal{S},k\neq j} dR_{SSi,j,k} - \sum_{k\in\mathcal{S}} dR_{SS_{di,j,k}} \\ - \sum_{k\in\mathcal{S}} dR_{SDi,j,k} - \sum_{k\in\mathcal{R}} dR_{SRi,j,k} - \sum_{(k,l)\in\mathcal{N}} dR_{Si,j,k,l} - dE_{Si,j} \end{aligned} \tag{51}$$

for S, where $K_{S-1,j} = \lambda_j$ and

$$\begin{aligned} \frac{dS_{di,j}}{dt} = P_j Q_{i,j} + K_{S_{di+1,j}} - K_{S_{di,j}} - \sum_{k\in\mathcal{S}} dR_{SS_{di,k,j}} - 2dR_{S_dS_{di,j,j}} - \sum_{k\in\mathcal{S},k\neq j} dR_{S_dS_{di,j,k}} \\ - \sum_{k\in\mathcal{S}} dR_{S_dD_{i,j,k}} - \sum_{k\in\mathcal{R}} dR_{S_dR_{i,j,k}} - \sum_{(k,l)\in\mathcal{N}} dR_{S_{di,j,k,l}} - dE_{S_{di,j}} \end{aligned} \tag{52}$$

for S_d, where $K_{S_{di+1,j}} = 0$ for the top cell. We get that

$$\begin{aligned} \frac{dD_{i,j}}{dt} = (1-P_j)Q_{i,j} + K_{Di+1,j} + \sum_{(k,l)\in\mathcal{N}} \zeta'_{j,k,l}(dR_{Si,j,k,l} + dR_{S_{di,j,k,l}}) - K_{Di,j} - \sum_{k\in\mathcal{S}} dR_{SDi,k,j} \\ - \sum_{k\in\mathcal{S}} dR_{S_dD_{i,k,j}} - \sum_{k\in\mathcal{S},k\neq j} dR_{DDi,j,k} - 2dR_{DDi,j,j} - \sum_{k\in\mathcal{R}} dR_{DRi,j,k} - \sum_{(k,l)\in\mathcal{N}} dR_{Di,j,k,l} - dE_{Di,j} \end{aligned} \tag{53}$$

where again $K_{Di+1,j} = 0$ for the top cell. The resulting rate equations for rocket bodies and debris are unchanged.

3.2.2 Implementation

Again, the implementation of the full variant of the fractional table model is just an updated version of the same code. The basic structure and methods are not changed. Implementation remains in Python3.

4 Atmospheric Model

The model used to calculate atmospheric drag lifetimes is contained in AtmosphericDevayModels.py, and consists of three main parts.

4.1 Atmospheric Density

Before the atmospheric density, the current solar radiation flux is calculated using a linear interpolation of the JB2008 solar flux model as compiled by CIRA-2012 (ADD) in altitude log-space. This leads to three separate cases.

Case 1: The solar flux is $\leq 65 \times 10^{-22} W/m^2$. In this case, simply take a linear interpolation (in log-space) of the JB2008 total air density model for low solar activity [4].

Case 2: The solar flux (SF) is such that $65 \times 10^{-22} W/m^2 < SF \leq 140 \times 10^{-22} W/m^2$. Then the air density is calculated via a linear interpolation of the JB2008 model for both low and medium flux. These two values are then linearly interpolated using the flux, with a flux of 140 returning only the medium flux density.

Case 3: $140 \times 10^{-22} W/m^2 < SF \leq 250 \times 10^{-22} W/m^2$. This is the same as the previous case, but using the JB2008 model for medium and long-term high flux.

Case 4: $SF > 250 \times 10^{-22} W/m^2$. This is the same as case 1, but using the air density model for long-term high solar activity.

4.2 Atmospheric Drag Lifetime

For this calculation, we assume that all objects are on a perfectly circular orbit around Earth. The drag lifetime characterizes the amount of time an object spends before its orbit degrades into a lower shell, and is taken to be the amount of time for the orbit to degrade from the top of the shell to the bottom. This value is taken as the process is roughly an exponential decay, so almost all of this time is taken up decaying near the top of the shell (i.e. this is a worst-case value). The rate of change of the separation a between an object in circular orbit about the Earth and the centre of the Earth is given by

$$\dot{a} = -C_D \rho_{atm}(a,t) \frac{A}{m} \sqrt{GMa} \tag{54}$$

where C_D is the drag coefficient (assumed to be 2.2 by default), $\rho_{atm}(a,t)$ is the density at the given altitude and time, A/m is the area-to-mass ratio of the object in orbit, and M is the mass of Earth. For a Cell with top altitude h_2 and bottom altitude h_1, we look for a t_f such that

$$h_1 - h_2 = \int_0^{t_f} \dot{a} dt \tag{55}$$

where $a(0) = h_2$. This is done numerically using a simple PC method, with the predictor being Euler method and corrector the trapezoid method. A basic adaptive time step is used, taken as

$$dt = -2\frac{a - R_e}{\dot{a}_n + \dot{a}_{n+1}}k \tag{56}$$

where R_e is the radius of the Earth, and k is an arbitrary factor taken to be 1/100 by default (this value was found to be good using some test calculations).

4.3 Collision Velocities

For collisions between objects in a shell, we assume that the orbital planes have uniformly distributed normal vectors. Then the probability of a collision angle $\theta \in [0, \pi]$ is given by $\frac{\sin\theta}{2}$. Assuming that the velocity of the target object is along the x-axis, the relative velocity would be given by (where v_{orb} is the orbital velocity in the shell)

$$v^2 = (v_{orb} - v_{orb}\cos\theta)^2 + (v_{orb}\sin\theta)^2$$

So the average velocity is then

$$\langle v \rangle = v_{orb}\sqrt{2}\int_0^{\pi} \sqrt{1 - cos\theta}\frac{\sin\theta}{2} d\theta$$

which evaluates to

$$\langle v \rangle = \frac{4}{3}v_{orb} \tag{57}$$

4.4 Updating Drag Lifetimes

In the simulations, drag lifetimes are updated with a particular set frequency since the atmospheric density is time-dependent. Generally, since CIRA-2012 tabulates monthly values [4], the lifetime is updated once per month. Furthermore, the initial stage of the solar cycle can effect the result (i.e. we

really take $\rho(a, t + t_0)$ where t_0 is some offset). We start at the arbitrary first month of the solar cycle as tabulated by [4] by default. For phenomena taking place over a time period longer than one cycle (12 years), the initial time in the cycle is likely not that important.

5 Debris Generation Model

Unless otherwise stated, assume all quantities in this section are in standard SI units.

5.1 NASA Breakup Model

This will be a more broad overview of the model, more specifics are outlined in the numerical implementation and in the following sources [2,5]. We characterize debris by two parameters: its characteristic length (the cube root of the volume) L_C, and it's area-to-mass ration A/M. The number of pieces of debris with characteristic length of at least L_c generated by a collision is then given by [2]

$$N(L_C) = 0.1M^{0.75}L_C^{-1.71} \tag{58}$$

where M is a parameter dependent on the type of collision, given by [2]

$$M = \begin{cases} m_t + m_d & \frac{m_d v^2}{2m_t} \geq 40[J/g] \\ m_d v & \frac{m_d v^2}{2m_t} < 40[J/g] \end{cases} \tag{59}$$

where m_t, m_d are the mass of the target and debris in kg, and v is the relative speed of the satellite and debris in the collision in km/s. As a quick notation note, we define that a collision is

Catastrophically Lethal if $\frac{m_d v^2}{2m_s} \geq 40[J/g]$
Non-Catastrophically Lethal if $\frac{m_d v^2}{2m_s} < 40[J/g]$
Non-Lethal if the debris is too small to have a discernable effect on the satellite.

Catastrophically lethal collisions are taken to completely destroy the target, while non-catastrophically lethal collisions are taken to disable a satellite (or have little effect on a derelict/rocket body).

Using this, we can see that for collisions

$$P(L_{min} \leq L \leq L_C) = \frac{N(L_{min}) - N(L_C)}{N(L_{min}) - N(L_{max})}$$

which equivalently is

$$P(L_{min} \leq L \leq L_C) = \frac{L_{min}^{-1.71} - L_C^{-1.71}}{L_{min}^{-1.71} - L_{max}^{-1.71}} \tag{60}$$

For explosions, we instead have [2]

$$N(L_C) = 6CL_C^{-1.6} \tag{61}$$

the C parameter is generally defined as a function of altitude in [5], to fit historical data. Since this is more a product of the types of satellites/rockets launched to each altitude band, and hence not particulairly predictive, we take C as a free parameter. It's range is $[0.1, 1]$ [5], with a value of 1 for rocket upper stages. Either way, we can get that for explosions

$$P(L_{min} \leq L \leq L_C) = \frac{L_{min}^{-1.6} - L_C^{-1.6}}{L_{min}^{-1.6} - L_{max}^{-1.6}} \tag{62}$$

Once an L_C has been selected, we define the probability distribution for the A/M ratio as (for satellites, assuming that $L_C > 11cm$) [2]

$$D_{A/M}(\lambda_c, \chi) = \alpha^{S/C}(\lambda_c) N(\mu_1^{S/C}(\lambda_c), \sigma_1^{S/C}(\lambda_c), \chi) + (1 - \alpha^{S/C}(\lambda_c)) N(\mu_2^{S/C}(\lambda_c), \sigma_2^{S/C}(\lambda_c), \chi) \tag{63}$$

where $\lambda_c = \log_{10}(L_C)$, $\chi = \log_{10}(A/M)$, and $N(\mu, \sigma, \chi)$ is the value of a normal distribution with standard deviation σ and mean μ at χ. The functions for α, μ, and σ factors are given in [2], and are labeled R/B for rocket bodies). If we instead assume that $L_C < 8cm$, we get that for both cases [2]

$$D_{A/M}(\lambda_c, \chi) = N(\mu^{SOC}(\lambda_c), \sigma^{SOC}(\lambda_c), \chi) \tag{64}$$

These integrate to the cumulative distributions

$$P(\chi_{min} \leq \chi* \leq \chi) = C\left(\alpha^{S/C} \Big[\operatorname{erf}\left(\frac{\chi^* - \mu_1^{S/C}}{\sqrt{2}\sigma_1^{S/C}} \right) \Big] \Big|_{\chi^* = \chi_{min}}^{\chi^* = \chi} + (1 - \alpha^{S/C}) \Big[\operatorname{erf}\left(\frac{\chi^* - \mu_2^{S/C}}{\sqrt{2}\sigma_2^{S/C}} \right) \Big] \Big|_{\chi^* = \chi_{min}}^{\chi^* = \chi} \right) \tag{65}$$

and

$$P(\chi_{min} \leq \chi* \leq \chi) = C \Big[\operatorname{erf}\left(\frac{\chi^* - \mu^{SOC}}{\sqrt{2}\sigma^{SOC}} \right) \Big] \Big|_{\chi^* = \chi_{min}}^{\chi^* = \chi} \tag{66}$$

respectively, where C is a normalization factor. For $L_C \in [8cm, 11cm]$, we take an interpolation of the two distributions, linear in log-space. Once we've chosen a χ, we get a distribution for the ejection velocity Δv of the debris of [2]

$$D_{\Delta v}(\chi, \nu) = \begin{cases} N(0.9\chi + 2.9, 0.4, \nu) & \text{Satellites} \\ N(0.2\chi + 1.85, 0.4, \nu) & \text{Rocket Bodies} \end{cases} \tag{67}$$

where $\nu = \log_{10}(\Delta v)$. This integrates to the cumulative distribution

$$P(\nu_{min} \leq \nu* \leq \nu) = C\Big[\operatorname{erf}\Big(\frac{\nu^* - (0.9\chi + 2.9)}{0.4\sqrt{2}}\Big)\Big]\Big|_{\nu^*=\nu_{min}}^{\nu^*=\nu} \tag{68}$$

for satellites and

$$P(\nu_{min} \leq \nu* \leq \nu) = C\Big[\operatorname{erf}\Big(\frac{\nu^* - (0.2\chi + 1.85)}{0.4\sqrt{2}}\Big)\Big]\Big|_{\nu^*=\nu_{min}}^{\nu^*=\nu} \tag{69}$$

for rocket bodies, where C is a normalization factor.

As a quick note, given the characteristic length and A/M of a piece of debris, we estimate its surface area by [2]

$$A = \begin{cases} 0.540424 L_C^2 & L_C < 0.00167m \\ 0.556945 L_C^{2.0047077} & L_C \geq 0.00167m \end{cases} \tag{70}$$

which allows us to estimate that $m_d = \frac{A}{A/M}$.

6 Known Problems

There is one main assumptions of the model which may be of questionable accuracy, namely that bands are homogeneous. Satellite clusters could instead be launched in a very tight altitude band or tight orbital inclination band, violating this assumption. Furthermore, when the number of objects is extremely low, treating its value as continuous becomes less and less accurate. This is a common issue when there's satellite types with very few individual objects associated to them, and for debris values. Getting enough bins to accurately model the debris (usually a 10x10 grid is used) often results in a large number of debris bins with values $< 10^{-6}$ or worse. As long as a rectangular table of debris bins is used there's no way around this.

Tight altitude bands can be compensated for with tight altitude binning, but this comes at the cost of lower object counts (and hence a less accurate continuity assumption) and higher computation times. No way is currently known of accounting for the inclination of orbits without a complete overhaul of the model, and this would likely require switching to an approach that tracks discrete objects. The low debris value issue could be fixed by ignoring the requisite debris bins, but the ignored bins cannot change during the simulation and care would have to be taken when selecting which bins to ignore.

7 Works Cited

[1] https://www.nsf.gov/news/special_reports/jasonreportconstellations/JSR-20-2H_The_Impacts_of_Large_Constellations_of_Satellites_508.pdf

[2] https://www.sciencedirect.com/science/article/pii/S0273117701004239

[3] Numerical Methods for Ordinary Differential Equation: Initial Value Problems by D.F. Griffiths and D.J. Higham (2010)

[4] https://spacewx.com/wp-content/uploads/2021/03/chapters_1_3.pdf

[5] https://link.springer.com/book/10.1007/3-540-37674-7