[0] REBOUND: Intergalactic Interactions
By Peter Micciche 17 May, 2025

[1] The REBOUND package is an N-body simulator. It's capable of computing (or "integrating") the motions of objects in space under the influence of gravity. Those objects can be a variety of interstellar bodies; in this project, I make them stars.

[2] I teamed up with Ben Ackerman on this project, and initially, we wanted to use pNbody, another N-body simulation package, to model the creation of a black hole. We ran into issues with the installation, however, and searched for other packages. I liked the capabilities of REBOUND and decided to model the interaction of two galaxies using the package. After meeting with Prof. Teuben and learning more about what these simulations can look like/do, I became even more interested in working with REBOUND.

[3,4] REBOUND was developed in 2011 by researchers at the University of Toronto as well as other contributors. It is still maintained by Hanno Rein, et al. as can be seen at this link. There are plenty of N-body simulation packages, but REBOUND is a popular, easy-to-install and use, and open-source package, making it a great choice for this project. As an open-source code, anyone wanting to contribute is welcome. One of the contributors to REBOUND, Dan Tamayo, created REBOUNDx, which incorporates additional physics into the package.

I am running version 4.4.8 in this project.

[5,6] Installation was extremely easy. I ran a simple "pip install rebound" command, and it was fully installed.

[7] The source code is all available within the GitHub repository github.com/hannorein/rebound/

[8] The code is used in other packages. REBOUNDx (for REBOUND eXtra) uses the code but builds in additional physical effects to add to simulations. Posidonius is a code that models tidal planetary interactions based on Mercury-T. It uses an integrator (WHFast) from REBOUND to compute positions and velocities.

[9] REBOUND can be run in C from the command line or in a Python script and Jupyter notebook, but the GitHub repository also offers a plethora of example code to demonstrate the uses without having to install and run the code. There is no web interface, however.

[10-15] REBOUND is coded in C and Python, but no C coding is required to run it in JupyterLab, as I did.

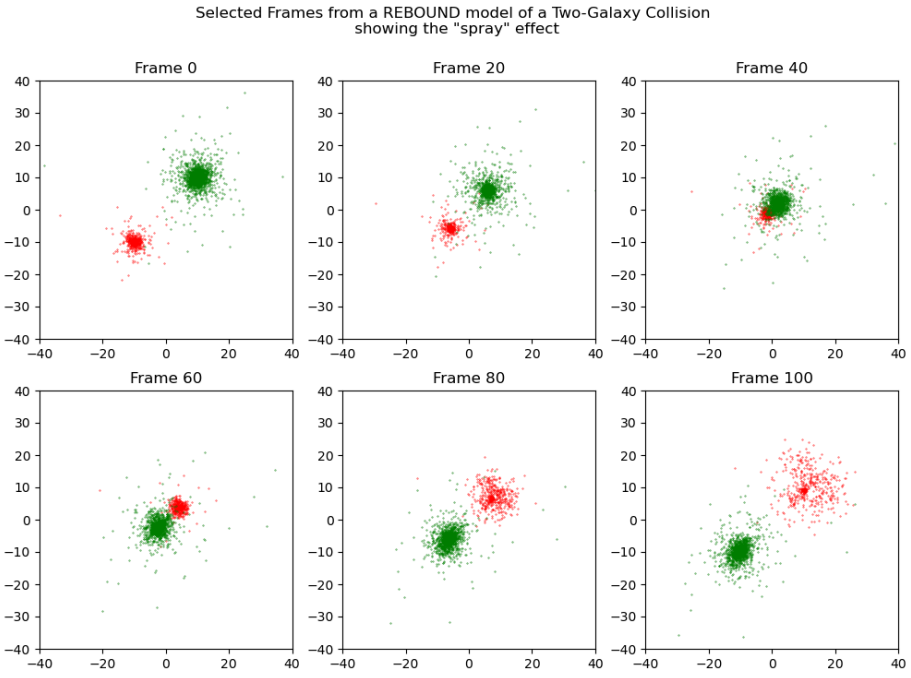
The accompanying notebook demonstrates the creation of two plummer spheres, one ¼ the mass of the other, and moving towards each other at a net velocity of 2sqrt(2). I created a function to create those plummer spheres using an RNG based on some examples I found online. To generate the simulation itself, I simply had to enter a few lines of code, listed below:

```python
sim = rebound.Simulation();
G = 1
sim.G = G
sim.integrator = "leapfrog"
sim.dt = 0.15
sim.softening = .05
sim.collision = 'direct';
sim.collision_resolve = 'merge';
```

For inputs, I then had to enter the parameters for each galaxy I generated, and generate them, adding them to the simulation with the sim.add() command. I then set a time array from 0 to 40 units and 101 steps, and used the sim.integrate() command to integrate and get the output of the xzy positions and velocities of each individual particle at time t in the time array. I then captured those outputs into a positions array to be referenced later for plotting.

In a generalised way, the output from sim.integrate(t) is the 3d position, 3d velocity, and mass at that time t.

REBOUND includes a plotting tool (sim.widget()), but I chose to do my plots directly in matplotlib. In my notebook, there is also an animated simulation of the time evolution of the galaxy interaction, which was generated by repeatedly clearing the plot and replotting with a slight delay.



This figure shows six equally spaced time frames in the simulation. A very cool aspect of the collision can be seen by the way both galaxies “compress” as they approach each other, then expand again once they have passed. In the smaller galaxy’s case, this “shredding” leaves its center noticeably smaller than before the collision. The cause of this is the increased potential experienced by all stars in both galaxies when the two approach each other. The galaxies feel this potential increase, compress towards their centers, then expand outward again once that potential suddenly drops as the galaxies drift farther apart. This is a trustworthy result since, in larger simulations with higher N values - for example, those done in NEMO and showed to me by Prof. Teuben - one can see the smaller galaxy eject a conical spray of stars as it emerges from the collision, and we are able to see something similar in this very crude simulation.

[16-17] The compression and expansion behavior explained above in section [10-12] is a good confidence indicator that my simulation was accurate, but on top of that, REBOUND has many tests available in its GitHub repository for a variety of integrators and features.

[18] In the GitHub repository, there is a file called requirements.txt, which lists the two required packages - numpy and matplotlib. Looking through the source code, most modules require an import of math, so I would consider that another main package requirement.

[19] REBOUND provided a ton of documentation on all the features it contains and how to use them. As an open-source package, the developers are very helpful and want to make it as easy to use as possible. They also link to some short YouTube tutorials on basic features of REBOUND.

[20] REBOUND makes citations very simple. If you enter sim.cite(), it outputs all the information needed to cite the setup of REBOUND you are using.

[21]

REBOUND Website: https://rebound.readthedocs.io/en/latest/
REBOUND ASCL: https://ascl.net/1110.016
REBOUNDx: https://reboundx.readthedocs.io/en/latest/effects.html
Posidonius: https://github.com/marblestation/posidonius
The Discovery and Characterization of Earth-Crossing Asteroid 2024 YR
The dynamical orbital stability of the proposed 2+1+1 hierarchical eclipsing binary systems

[22] The Discovery and Characterization of Earth-Crossing Asteroid 2024 YR_4 analyses a recently discovered asteroid and attempts to derive its origin based on orbital data. REBOUND was used to map orbits.

The dynamical orbital stability of the proposed 2+1+1 hierarchical eclipsing binary systems uses REBOUND to analyse the stability of eclipsing binary systems. Using various integrators and

tools in REBOUND, such as the MEGNO chaos indicator, it was found that the proposed orbital models can exhibit high instability.

[23] At least at the level I was using it, REBOUND required only very basic knowledge of Python. The generation of inputs and plotting the results were the most challenging parts, as all the integration is taken care of behind the scenes.

[24] REBOUND was entirely new to me, and I collaborated with Ben Ackerman on this project.