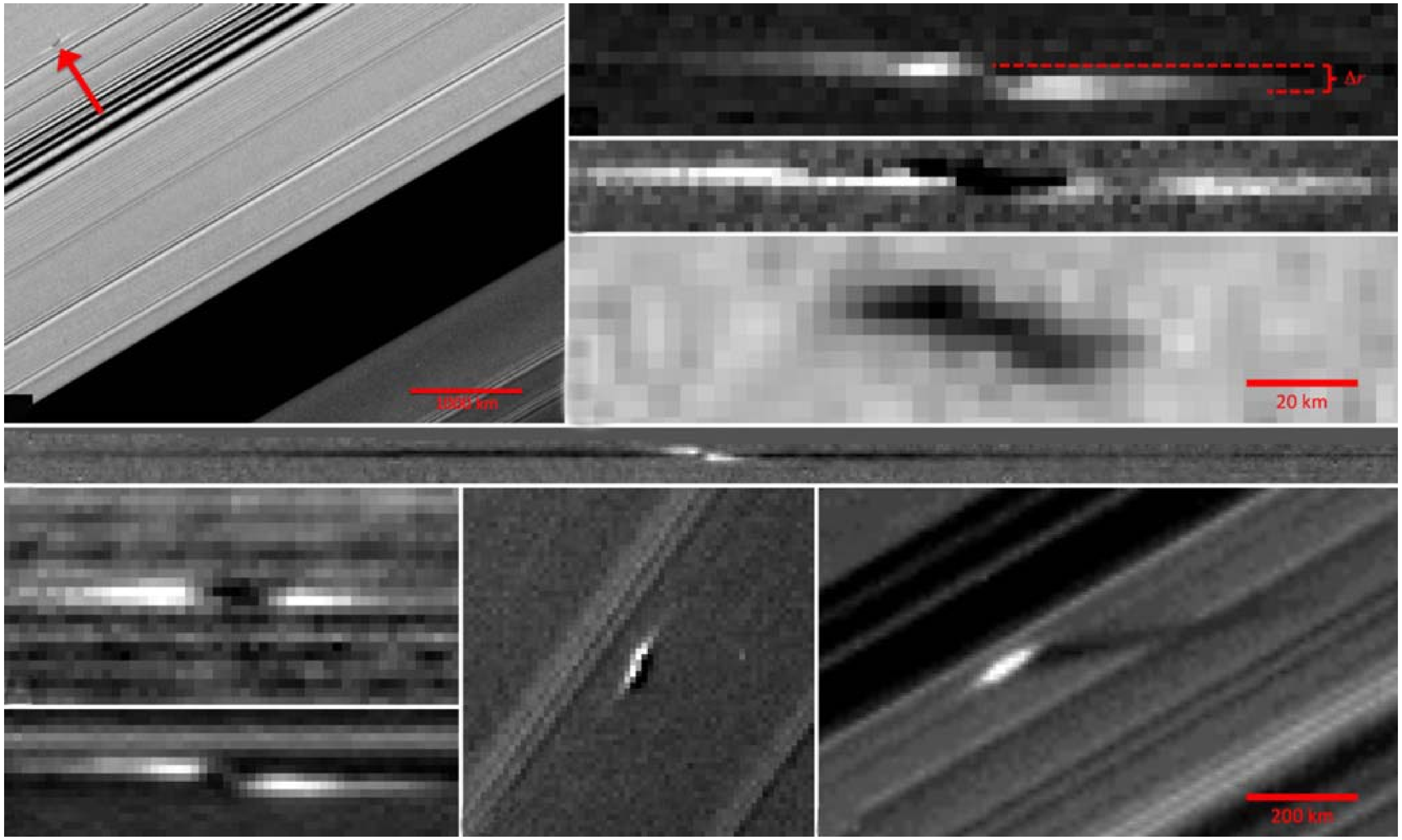
ISIMA

INTERNATIONAL SUMMER INSTITUTE FOR MODELING IN ASTROPHYSICS

Planetary Dynamics in Collisional Particle Disks

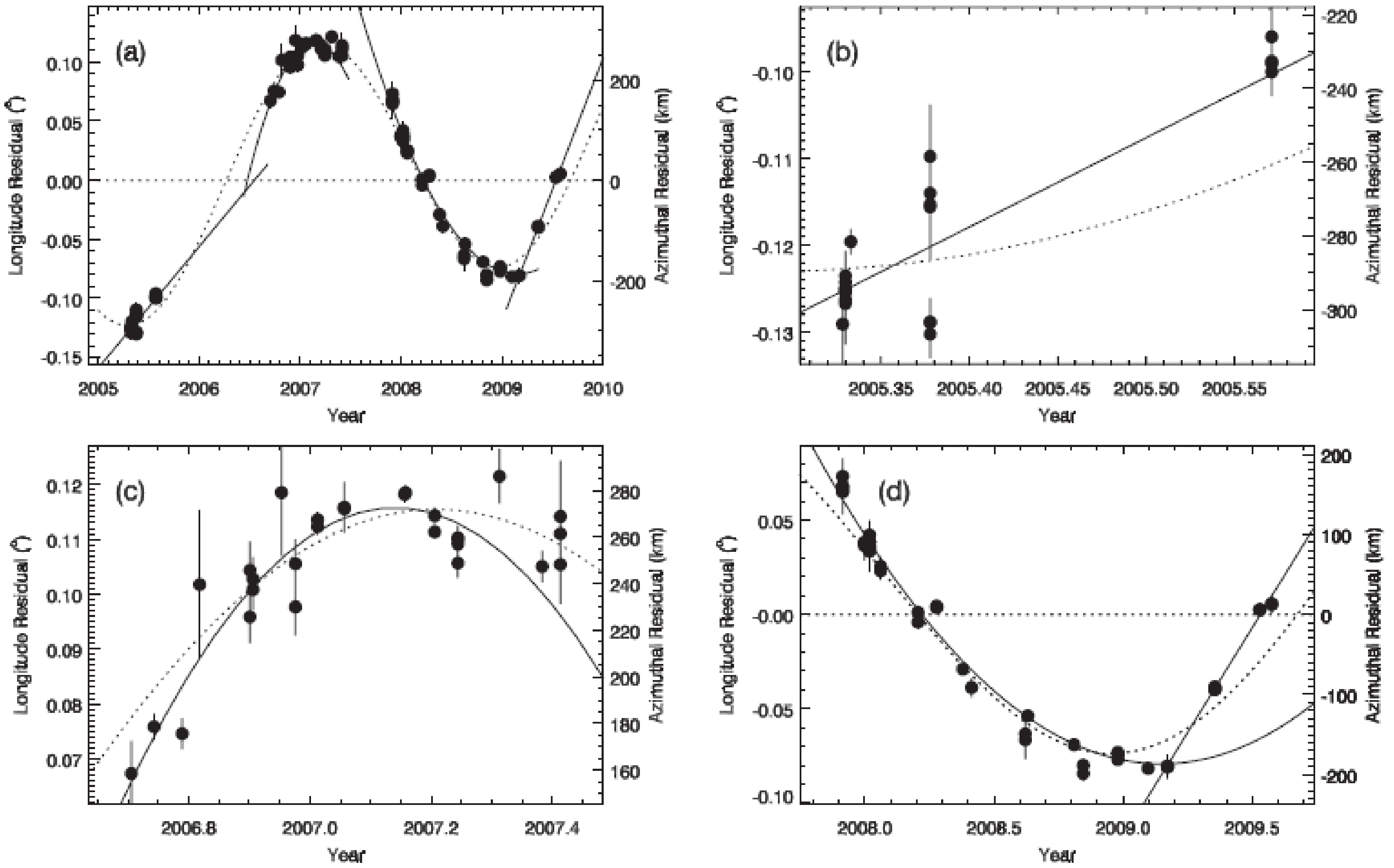
Supervisor: Eugene Chiang, U.C.Berkeley;
Yoram Lithwick, Northwestern University
Student: Zhao Sun, Purple Mountain Observatory

Motivation



Tiscareno et al. (2010, ApJL)

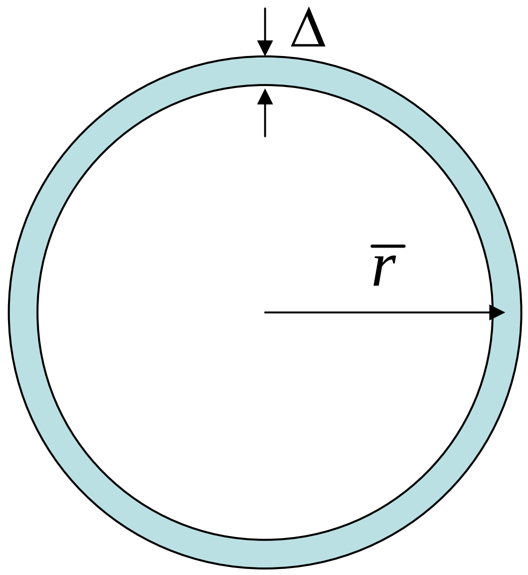
Motivation



Tiscareno et al. (2010, ApJL)

The model

- A subroutine simulates collisions between test particles in a disk with vertical optical depth $\tau \sim N_{\mathrm{tp}} \frac{s^2}{\bar{r}\Delta} < 1$.
- In collisional particle disks, collisions tend to isotropize the velocity distribution.
- The collision time is $t_{\mathrm{col}} \sim 1 / (n_v s^2 u)$, where u is the one-dimensional random speed and n_v is the volumetric number density, which is related to τ via $\tau \sim n_v s^2 u t_{\mathrm{orb}}$.
- Therefore $t_{\mathrm{col}} \sim \frac{t_{\mathrm{orb}}}{\tau}$.



Lithwick & Chiang (2007, ApJ)

The collision algorithm

- We capture this behavior with two-dimensional simulations in which all bodies have zero inclination.
- For every time step dt, a two-dimensional square grid is built, with each grid element having dimensions $s_{\mathrm{grid}} \times s_{\mathrm{grid}}$.
- If two test particles fall in the same grid cell,
- if their relative speed is negative (i.e., if they are approaching each other),
- then they collide with each other with probability $P_{\mathrm{col}} = dt/t_{\mathrm{orb}} \ll 1$.
- A random number generator is used to determine whether or not they actually collide.

Lithwick & Chiang (2007, ApJ)

The collision algorithm

- First, consider a simpler algorithm that also yields the correct collision time.
- In this simpler algorithm, one waits for a time interval of t_{orb} (instead of dt) before finding which particles fall in the same grid cell.
- Then two particles that do fall in the same grid cell, and have converging velocities, collide with probability $P_{\mathrm{col}} = 1$.
- Since the probability that a given particle lies in a cell occupied by a second particle is τ, the collision time is t_{orb}/τ.
- Turning now to the algorithm that we actually use, since we apply this algorithm every time interval dt (and not t_{orb}), we must correspondingly reduce the probability of a collision by dt/t_{orb} in order to maintain the collision time at the value given by $t_{\mathrm{col}} \sim \frac{t_{\mathrm{orb}}}{\tau}$.

Lithwick & Chiang (2007, ApJ)

The collision algorithm

- In a naive bruteforce algorithm, one must restrict $dt \ll s/u$ in order to ensure that any two particles that fall within a distance s of each other collide.
- This restriction on dt can be very cumbersome when $u \gg s/t_{\mathrm{orb}}$, as it will be whenever planets stir up the eccentricities.
- We avoid the Courant condition by treating the vertical dimension statistically: when two particles fall within the same two-dimensional grid cell, they need only collide a small fraction of the time because their vertical positions will, in general, differ.
- With our algorithm, we may choose dt to be as large as is allowed by the integrator, which is typically a significant fraction of the orbital time.

Lithwick & Chiang (2007, ApJ)

The collision algorithm

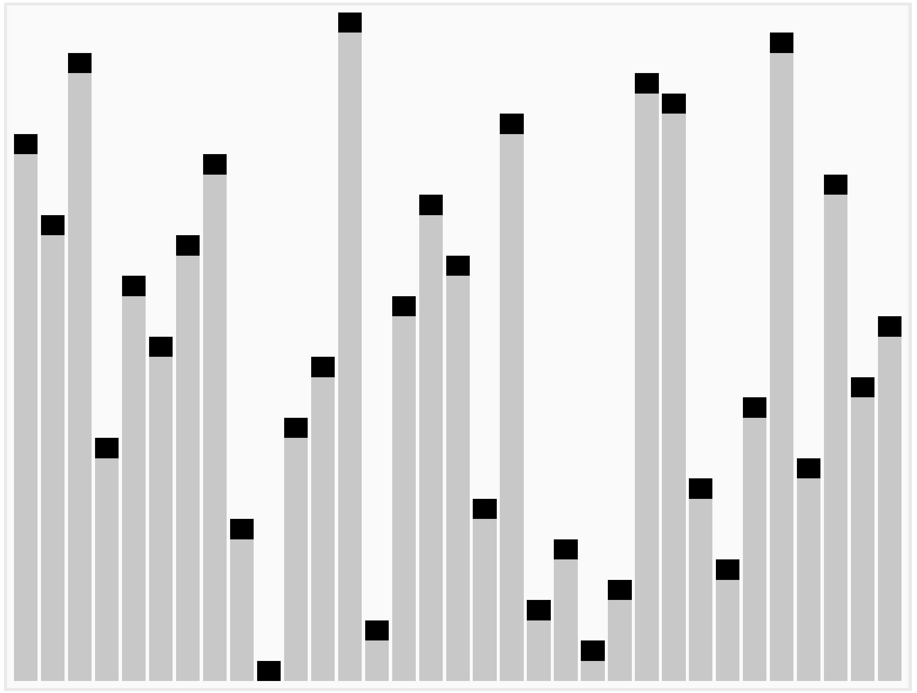
- If two particles have been selected for a collision, then their velocities are updated as though the bodies were frictionless spheres whose surfaces touch (e.g., Trulsen 1971): the component of the relative velocity vector that lies parallel to the axis connecting the two particles is reversed in sign (from a converging velocity to a diverging one) and multiplied by the coefficient of restitution , i.e., in obvious notation, $u'_{\mathrm{rel},||} = -\varepsilon u_{\mathrm{rel},||}$.
- Note that a collision between two particles separated by distance d changes the velocities of the particles as though each were a smooth sphere with radius $d/2$. Since d changes from collision to collision, the particles' sizes are effectively changing; they are only approximately s_{grid}.

The collision algorithm

- how the code finds which pairs of particles lie in the same grid cell:
- To find colliding pairs, the code first determines in which grid cell each particle lies. A grid cell is labeled by two integers representing its location along the x- and y-axes.
- Second, the code sorts the grid cells that contain test particles with the heapsort algorithm (Press et al. 1992).
- The sorted occupied grid cells are then checked to see if the same grid cell is repeated for two different particles.
- The step that takes the most time in the entire collision algorithm is the heapsort, which requires $\sim N_{\mathrm{tp}} \ln N_{\mathrm{tp}}$ operations.

The collision algorithm

- Heapsort begins by building a heap out of the data set, and then removing the largest item and placing it at the end of the partially sorted array. After removing the largest item, it reconstructs the heap, removes the largest remaining item, and places it in the next open position from the end of the partially sorted array. This is repeated until there are no items left in the heap and the sorted array is full.

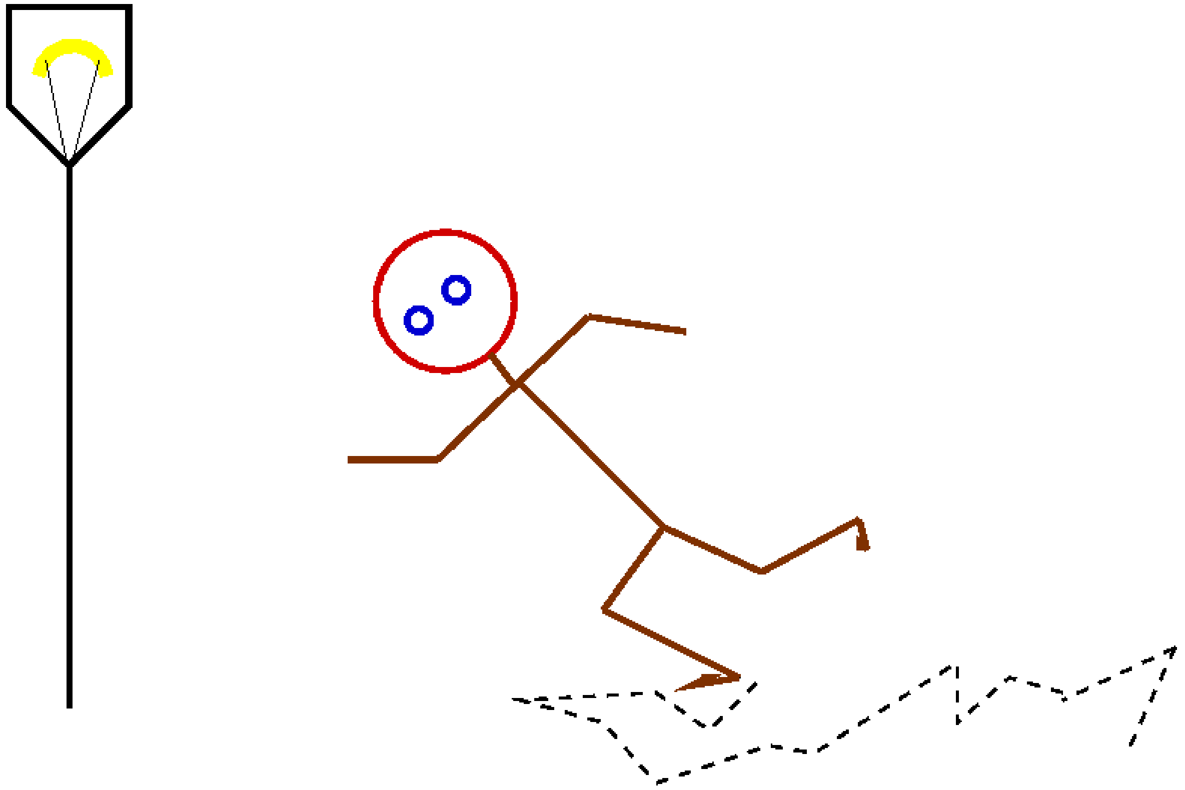


The collision algorithm

6 5 3 1 8 7 2 4

Density evolution

- A ring diffuses in the time that it takes a particle to random walk across its width.

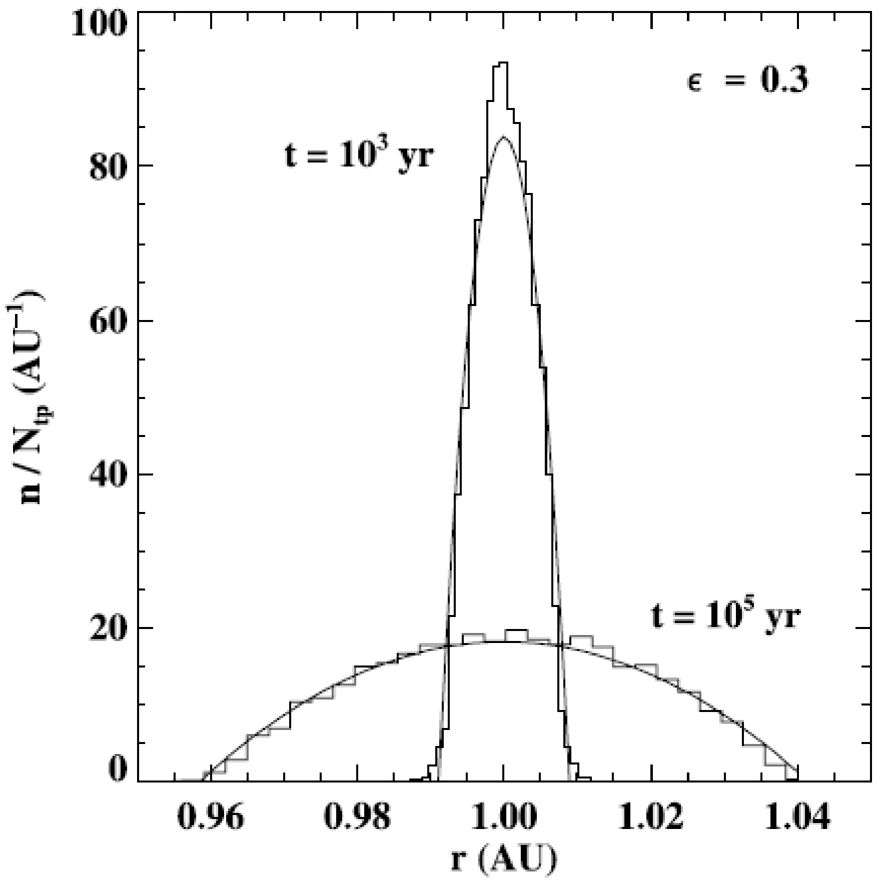


Density evolution

- This random walk has a step-size equal to the epicyclic excursion of a particle ($\sim re_{\mathrm{rms}} \sim s_{\mathrm{grid}}$) and a time per random step of t_{col}. Thus, to diffuse the width of the ring takes a time
$$t_{diff} \sim t_{col} \left(\frac{\Delta}{s_{grid}} \right)^2 \gg t_{col}$$
- Since $t_{col} \propto 1/n \propto \Delta$,where $n \equiv dN_{tp}/dr$, a ring expands as

$$\Delta \propto t^{1/3}$$

Simulations of narrow circular rings



Lithwick & Chiang (2007, ApJ)

The symplectic integrator

$$H = \underbrace{\frac{1}{2}p^2 + \Phi_{\text{Kepler}}(x)}_{\text{Kepler}} + \underbrace{\Phi_{\text{Other}}(x)}_{\text{Kick}}$$

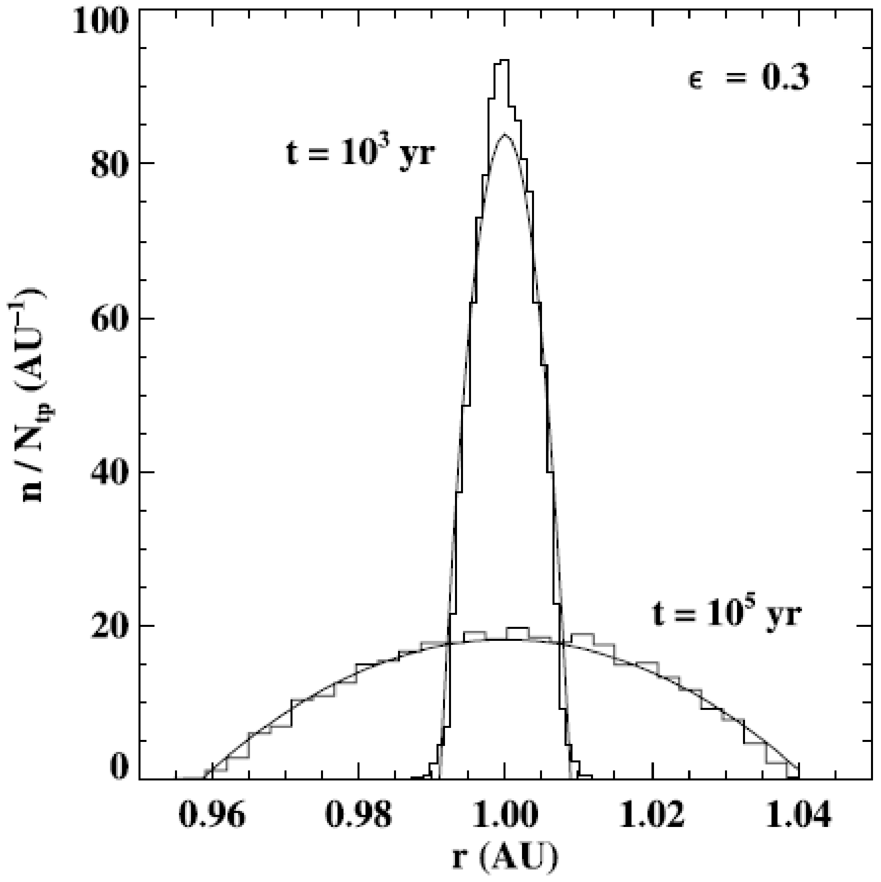
$$\begin{tikzcd}
\text{1/2 Kick} \arrow[r] & \text{Kepler} \arrow[r] & \text{1/2 Kick} \arrow[ll, bend left]
\end{tikzcd}$$

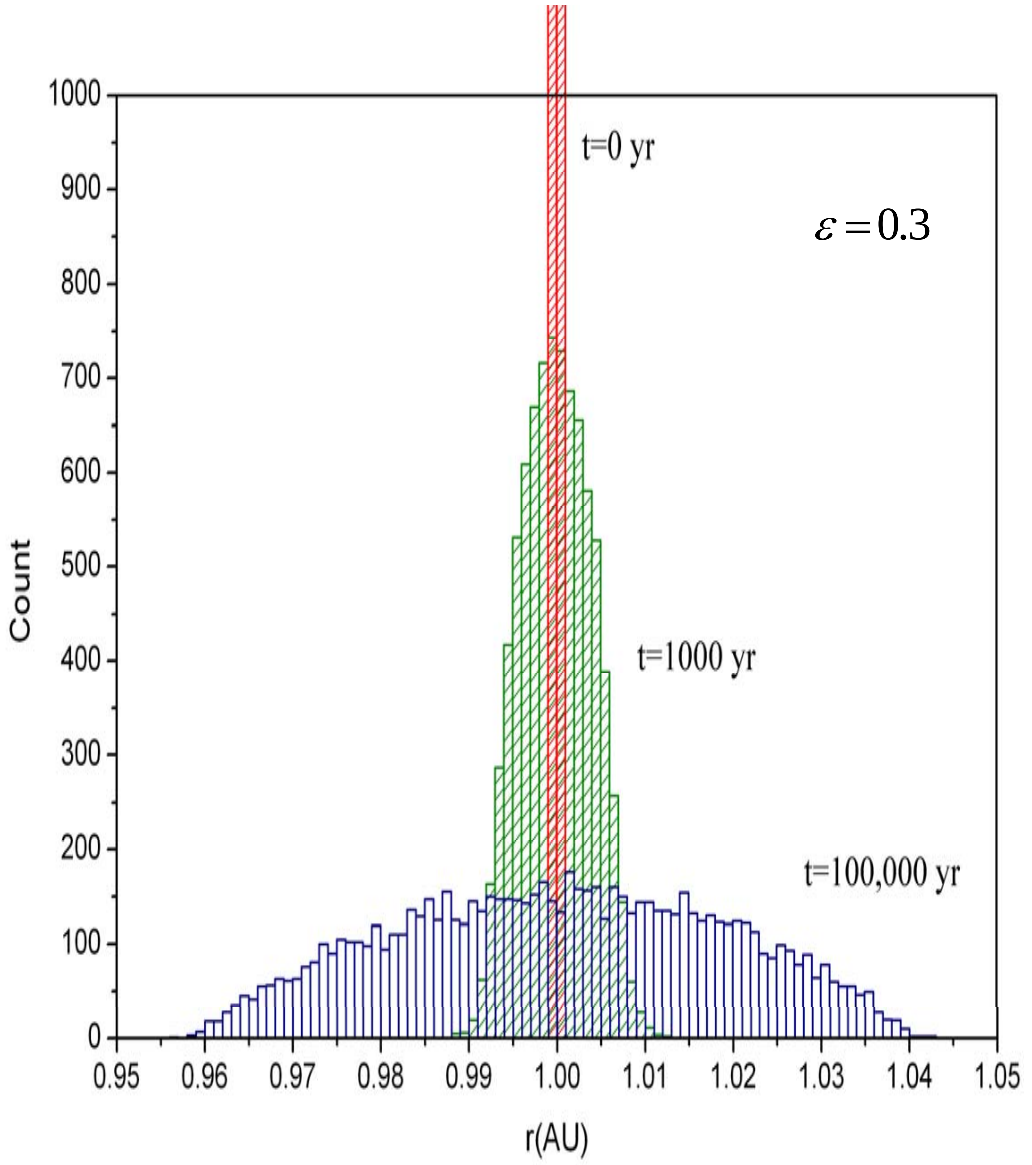
The hybrid symplectic integrator

- The integration scheme for the second-order hybrid integrator is as follows.
- (i) The coordinates remain fixed. Each body receives an acceleration owing to the other bodies (but not the Sun), weighted by a factor K, lasting for $dt/2$.
- (ii) The momenta remain fixed, and each body shifts position by an amount .$dt\sum_i p_i / 2m_\odot$
- (iii) Bodies not in an encounter move on a Keplerian orbit about the Sun for dt. For bodies in an encounter, the Kepler terms, and the close encounter terms weighted by $(1\text{-}K)$, are integrated numerically for dt.
- (iv) As step (ii).
- (v) As step (i).

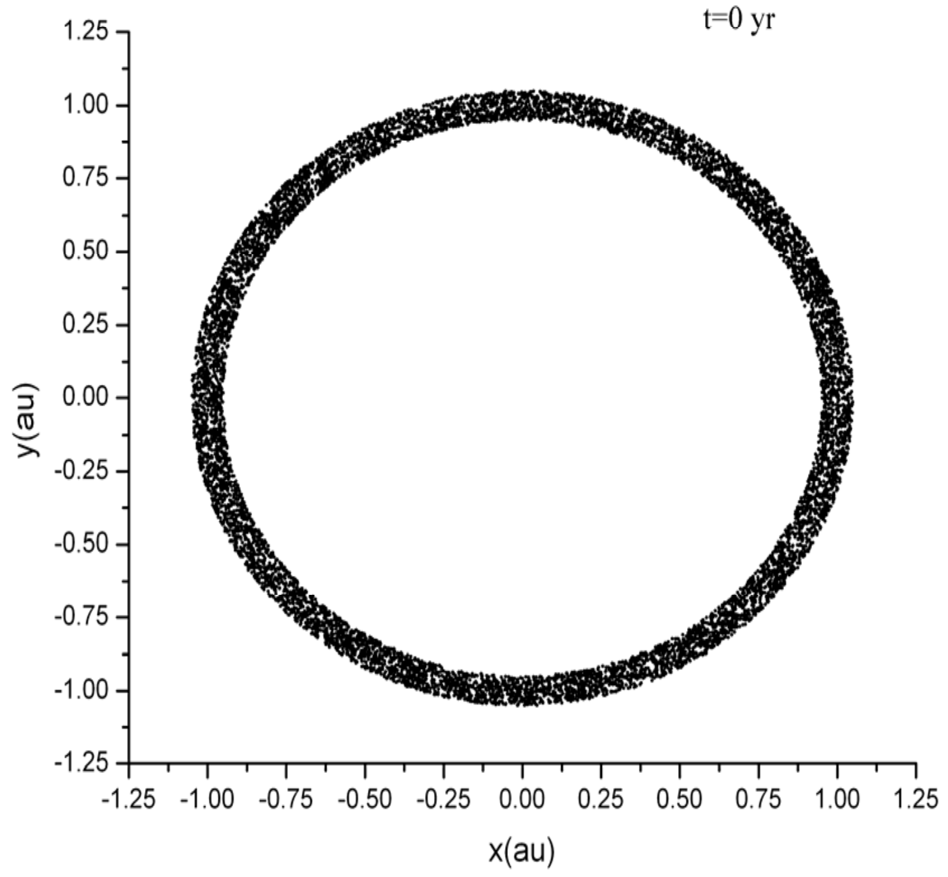
Test of the code

Lithwick & Chiang (2007, ApJ)





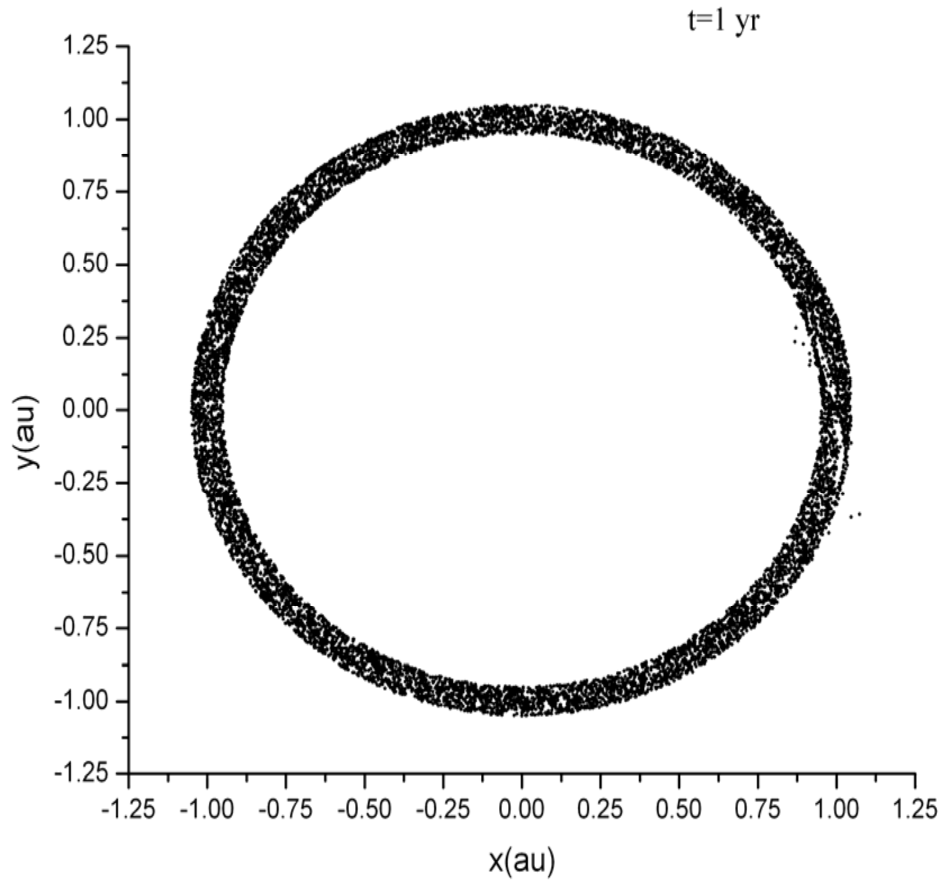
Early results



Massless ring with a moon incide. $m_{moon} = 3\times 10^{-6} m_{\odot}$

Ring width is 0.1AU.

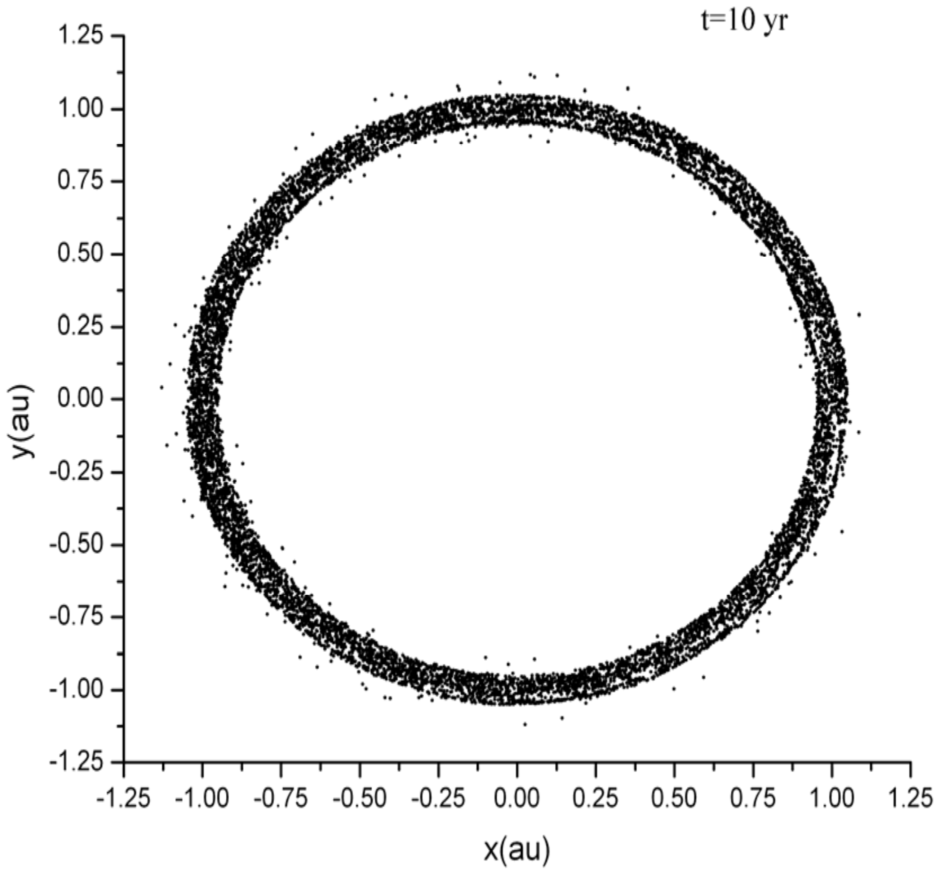
Early results



Massless ring with a moon incide. $m_{moon} = 3\times10^{-6} m_{\odot}$

Ring width is 0.1AU.

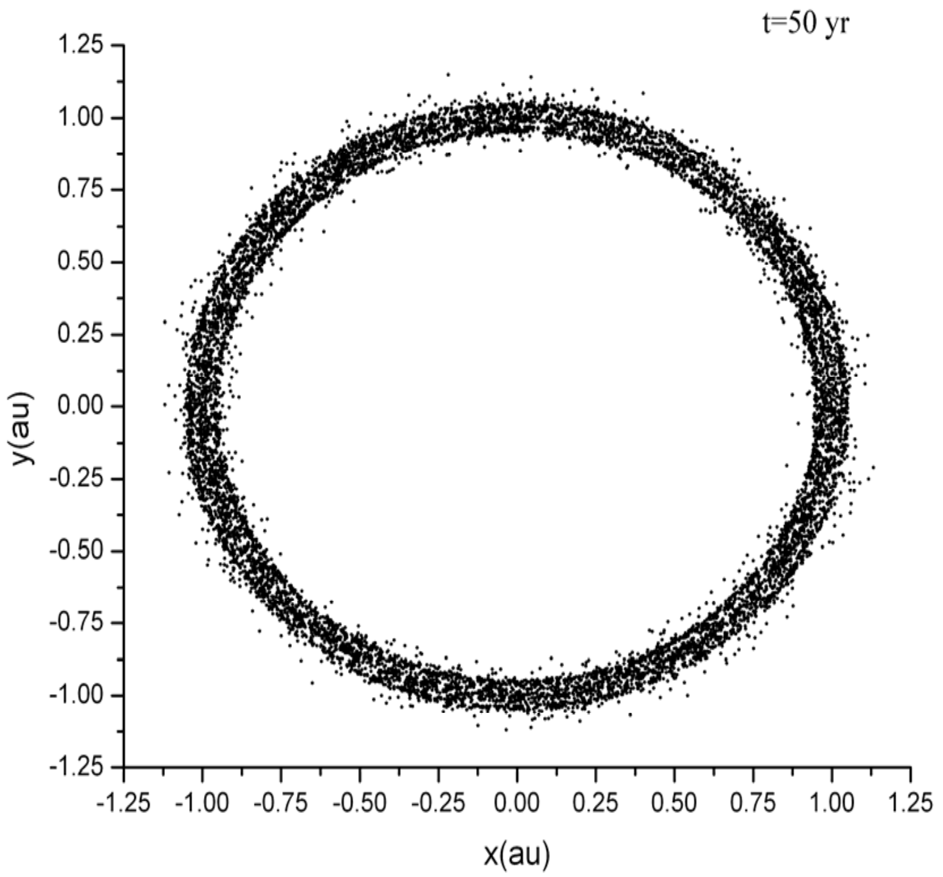
Early results



Massless ring with a moon incide. $m_{moon} = 3\times10^{-6} m_{\odot}$

Ring width is 0.1AU.

Early results



Massless ring with a moon incide. $m_{moon} = 3 \times 10^{-6} m_{\odot}$

Ring width is 0.1AU.

Applications

- How does Type I/II migration rates vary with planet eccentricity?
- How do gap widths and lengths vary with planet mass?
- For planets that do not open gaps or open only partial gaps, what torques are exerted by co-orbital disk material on the planet?
- ……

Future plans

- Optimize the code (and maybe parallelize it).
- Find out the best combinations of the free parameters, $\bar{r}, \Delta, s_{grid}, \mu$.
- Apply it on the propellers in Saturn's rings and compare with the observations.
- ……

Thank you!