PHYS 410

Nicholas Pun

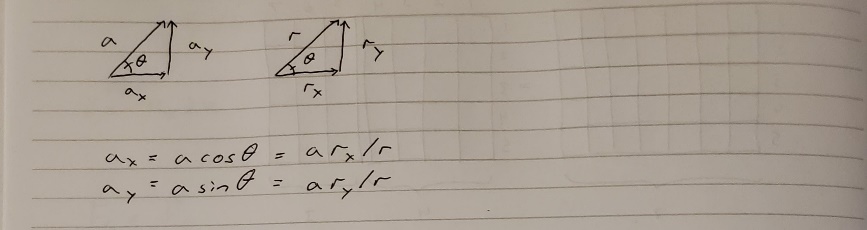
27872167

October 27, 2020

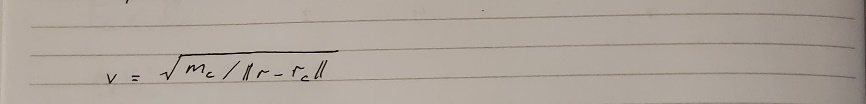
Project 01

Introduction

I begin developing my model for galaxy collisions by first attempting the simple case of 1 particle in circular orbit about 1 stationary core. Here, I calculate that since the acceleration vector must be parallel to the position vector, the following equations must hold:

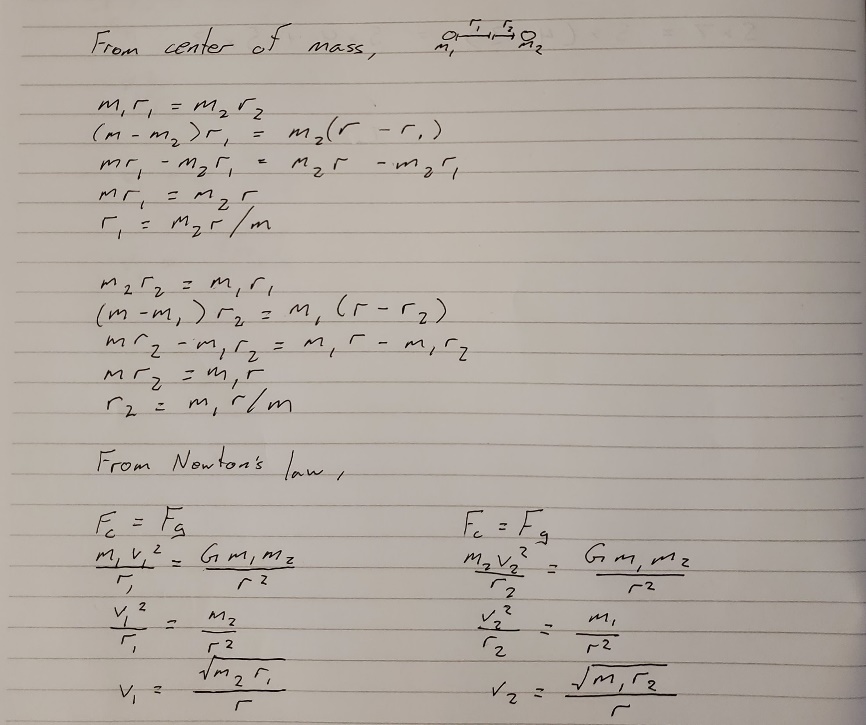


The velocity for circular orbit is as follows:



where mc is the mass of the core, rc is the position vector of the core, and r is the position vector of the star.

The acceleration magnitude can be calculated from the following equation:



Initial development of physics engine

In nbody0.avi a single star orbits a stationary core but not in a circular orbit. This is because there is some lower limit to the radius of the orbit for which my model can calculate circular orbits. By either decreasing the mass or increasing the radius of the orbit, I can achieve circular orbits.

In nbody1.avi I do the latter.

In nbody2.avi I then add a second core. This requires recoding the acceleration calculation for the star in the iterative part of the code such that it is accelerated by both cores. For n cores, this calculation would be repeated n times, once for each core.

In nbody3.avi I confirm the star is affected by movement of the core by giving the core some initial velocity. This also develops the velocity calculation for the cores.

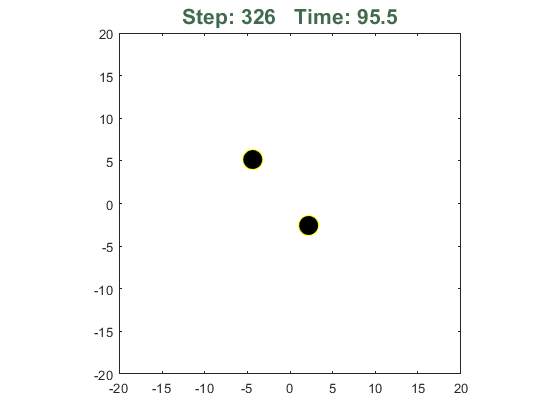
In nbody4.avi I develop the acceleration calculation for cores by reusing the acceleration calculation for stars. However, in this model the core is not attracted to stars, only other cores. The video confirms the cores are attracted.

In nbody5.avi I confirm that all the basic physics is working with 1 star and 2 cores.

Convergence testing

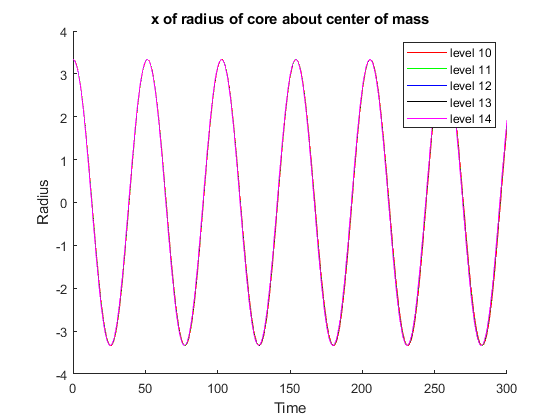
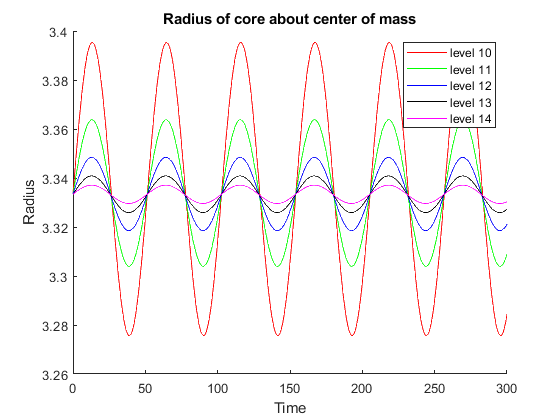
I now must test that my model is accurate by checking if the solution it provides converges. By decreasing the increment in time for each iteration of the physics engine I should achieve more accurate results. I decrease the time step by increasing the number of steps there are between time = 0 to time = tmax. A level determines the number of steps is 2^level+1.

In order to achieve meaningful results, I model a binary system where two cores of distinct masses are in mutual circular orbits. I then track the magnitude of one of the radius vectors of one of the cores. nbodyconv.avi shows the simulation for this configuration at level 10. I use masses 1 and 0.5 with a distance of 10. The following is a screenshot of it:

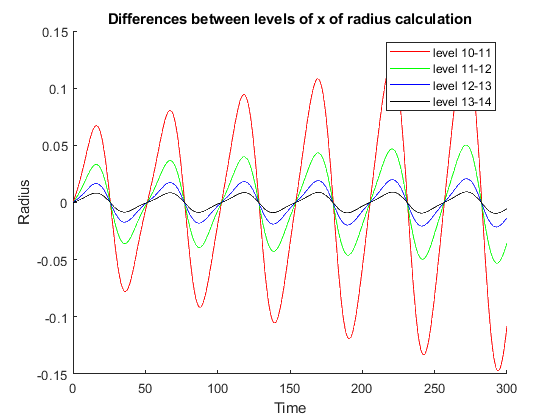
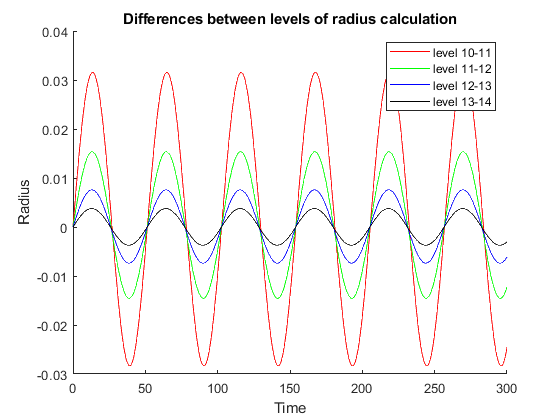


For two cores of mass m1 and m2, where m = m1 + m2, separated by a distance r, their distances r1 and r2, where r= r1 + r2, and velocities v1 and v2 can be determined through the following calculations:

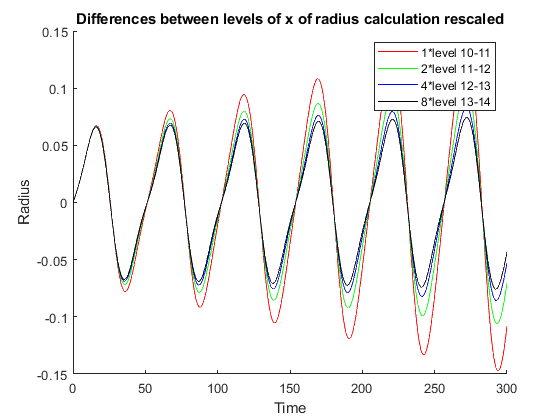
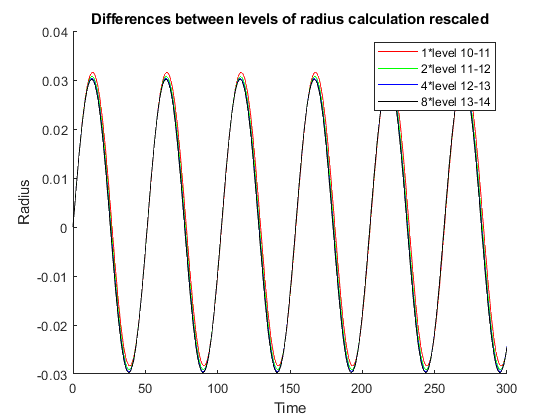
The following is a plot of the radius of the orbit of one of the masses over time for five different levels. As expected, it is sinusoidal.



I then plot the differences between the levels. As evident in the graph, the differences decrease as we go up in levels.



I then scale each difference. The scaling factor increases like O(deltat^2) as the level increases. This is evident in the graph since the plots are nearly coincident after scaling. This suggests that the model has O(deltat^2) error.



The second of each of the previous graphs are the same plots but just the x value of the r vectors. By the same logic, it also shows that the error goes like O(deltat^2) since the rescaled graph shows nearly coincidental plots. Although, it is less apparent since it seems differences are not constant with time. The convergence test using the magnitude of the r vector as shown in the left-hand plots is a better representation of the error.

Galactic collisions

In nbody6.avi I develop the code for multiple star systems. This is just replacing the single calculation of the single star during each iteration do a loop over all stars. This video models a stationary core with many stars around it to ensure circular orbits.

In nbody7.avi I check that the stars remain in circular orbit about a moving core. At first this was problematic as the stars seemed to be lagging behind. I then realized that I should be adding the initial velocity of the core to the initial velocity of the stars. After fixing that in my code, the stars now correctly remain in circular orbits about a moving core.

In nbody8.avi I model two galaxies colliding. The galaxies move slow enough such that stars from each galaxy end up in the other. Some stars fly away from the galaxies. This collision causes spiral-like galaxies with arms. I used the following settings:

tmax = 4000;

mc1 = 1000; mc2 = 1000;

ns1 = 1000; ns2 = 1000;

dir1 = 1; dir2 = 1;

rc10 = [-500, -500, 0];

rc20 = [500, 500, 0];

vc10 = [1, 0, 0];

vc20 = [-1, 0, 0];

[~,~] = nbody(8, tmax, mc1, mc2, ns1, ns2, dir1, dir2, ...

rc10, rc20, vc10, vc20, 0, 0);

In nbody9.avi the galaxies move a speed such that their collisions do not actually cause any stars to leave their initial galaxy. Instead, the gravity of the other galaxy just warps the orbits of the other galaxy’s stars slightly. This collision causes spiral-like galaxies with small arms. I used the following settings:

tmax = 4000;

mc1 = 1000; mc2 = 1000;

ns1 = 1000; ns2 = 1000;

dir1 = 1; dir2 = 1;

rc10 = [-500, -500, 0];

rc20 = [500, 500, 0];

vc10 = [1.5, 0, 0];

vc20 = [-1.5, 0, 0];

[~,~] = nbody(8, tmax, mc1, mc2, ns1, ns2, dir1, dir2, ...

rc10, rc20, vc10, vc20, 0, 0);

In nbody10.avi I model what I believe to be the closest representation to the example avi. I used the following settings:

tmax = 4000;

mc1 = 1000; mc2 = 1000;

ns1 = 1000; ns2 = 1000;

dir1 = 1; dir2 = 1;

rc10 = [-500, -500, 0];

rc20 = [500, 500, 0];

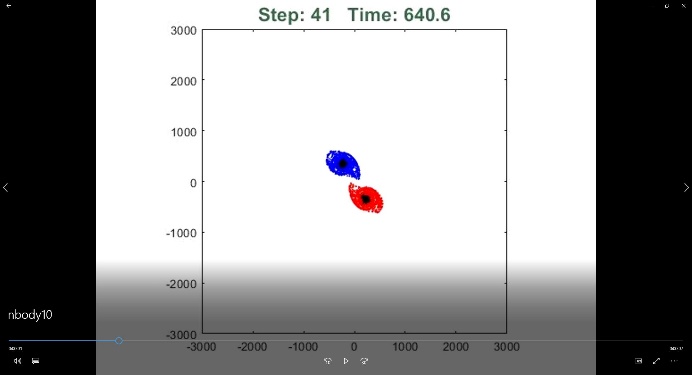
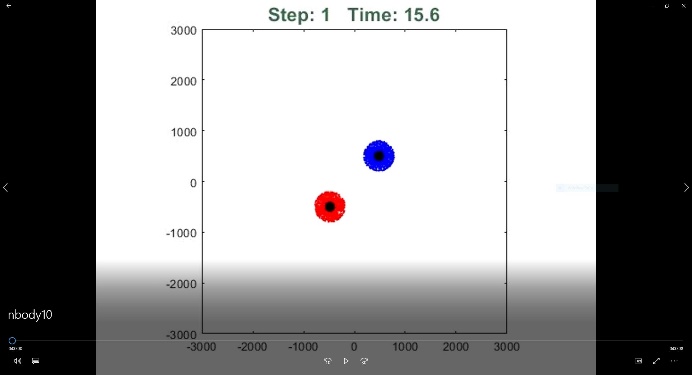
vc10 = [1.05, 0, 0];

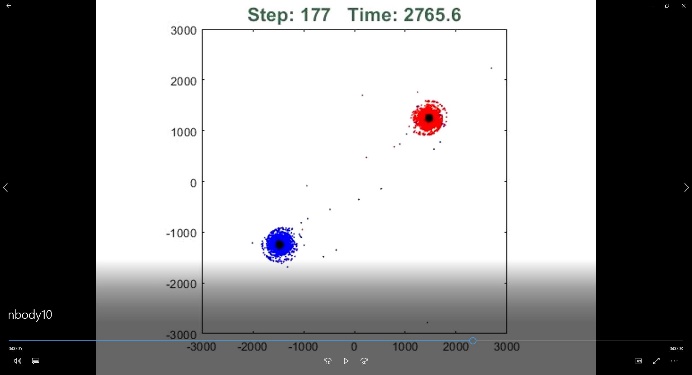
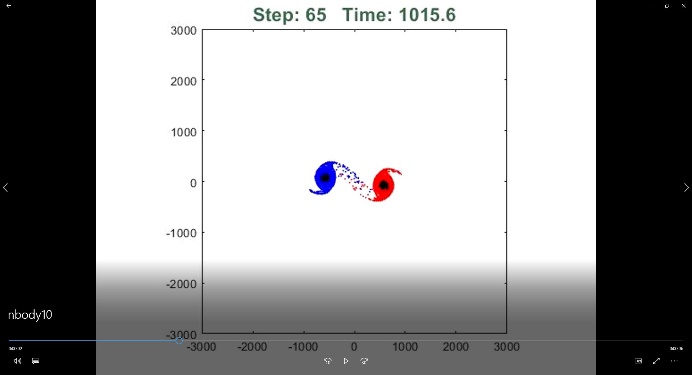
vc20 = [-1.05, 0, 0];

[~,~] = nbody(8, tmax, mc1, mc2, ns1, ns2, dir1, dir2, ...

rc10, rc20, vc10, vc20, 0, 0);

The following snapshots illustrate nbody10.avi:





The galaxies initially start circular. As they approach each other their gravities pull stars from the other galaxy towards them. The gravities of the cores also change the trajectory of the other core. After the galaxies pass each other, some of the stars that were between them are left behind. Some stars join the other galaxy and assume an orbit about its core. The galaxies now have spiral-like arms.

Theory

The Finite Difference Approximation used in my code works by converting continuous equations into discrete ones that can be evaluated at discrete time steps. The closer together the time steps are, the more accurate the overall model is. The acceleration on a given star or core can be approximated using:



where r is the position vector. The acceleration can also be calculated using:



which is simply the gravitational equation where G = 1. Equating these two equations results in:



which I use during each time step to calculate the next position of each star and core after isolating rin+1. This isolation is non-trivial. This does however mean that I must first perform the calculations for n = 1 and n = 2 before being able to perform any iterations. Fortunately, this gives me the opportunity to insert the initial positions at n = 1 and the next position due to the initial velocities at n = 2. n = 2 is also the step where I calculate the velocity required for circular orbits. Here, I have included a variable dir for the direction that I desire the stars to orbit the cores. dir = 1 means they orbit counter clockwise and dir = -1 means they orbit clockwise.

Overall, the code starts by setting the initial conditions for the position vectors. Then the second iteration of the position vectors are calculated using the initial velocities and the velocity required for circular orbits. The code then iterates through each time step where each iteration calculates the next position vectors based on the previous two position vectors of each star and core.