Introduction:

The Toomre model is used to show the movement and collisions of galaxies. It uses a central particle, called a core, with some gravitating mass which has some n amount of stars orbiting it. These stars have a vanishing gravitating mass meaning that we only pay attention to gravitational force exerted on them by the cores and not other stars. For this project, we are using two cores with some n amount of orbiting stars to simulate a collision between the two to observe how the stars move.

Review of Theory and Numerical Approach:

The Toomre model for this project is being solved by using the second order finite difference approximation. That is, for some continuous time domain 0 ≤ t ≤ t_max, we create a finite difference grid which we can use to derive and solve an approximate second time derivative for the equations in the model. In this case we are taking the second derivative of position (acceleration) that helps us solve the law of gravitation used.

To create our time mesh we take some time step $n_t = 2^l + 1$ where l is the level parameter. We then specify a $\Delta t = \frac{t_{max}}{n_t - 1} = 2^{-l} t_{max}$ giving us a set of times $t^n = (n - 1)\Delta t$ where n=1,2,...,n_t.

Now using this time mesh, we can solve the actual equations used in the Toomre model, which in this case is derived through combining Newton's second law and the law of gravitation.

$$m_i \bar{a}_i = G \sum_{j=1, j\neq i}^{N} \frac{m_i m_j}{r_{ij}^2} \hat{r}_{ij},\ i = 1, 2, \dots, N,\ 0 \le t \le t_{max}$$

This is the combination of the two equations which gives us our equation of motion. Here we are using a_i as our acceleration of every i-th particle, G as our gravitational constant which we set as 1 to have unitless dimensions, and r_ij as the magnitude of the separation vector **r_ij.** The second derivative we are trying to solve then becomes:

$$\frac{d^2 \bar{r}_i}{dt^2} = \sum_j \frac{m_j}{r_{ij}^3} \bar{r}_{ij}$$

The second derivative of position here is just our acceleration in the previous equation. The equation has been divided by m_i on both sides to integrate when the particles could be massless. This second time derivative is being solved in a function called **nbodyaccn.m** that computes the acceleration for every particle in the system.

To solve the actual FDA we need to take the continuous equation we derived and create a discrete one.

By doing so we get:

$$\left[\frac{d^2\bar{r}(t)}{dt^2}\right]_{t=t^n} \approx \frac{r^{n+1}-2r^n+r^{n-1}}{\Delta t^2}$$

We know this second derivative so plugging that in we get:

$$\sum_j \frac{m_j}{(r^n_{ij})^3}(r^n_j - r^n_i) = \frac{r^{n+1}-2r^n+r^{n-1}}{\Delta t^2},\ n + 1 = 3, 4, \dots, n_t$$

To solve for r^{n+1} we just rearrange and get:

$$r^{n+1} = 2r^n - r^{n-1} + \Delta t^2 \sum_j \frac{m_j}{(r^n_{ij})^3}(r^n_j - r^n_i)$$

We must plug in the initial conditions then:

$$r^1_i = r_i(0) = r(0)$$

$$r^2_i = r_i(\Delta t) = r(0) + v(0)\Delta t + \frac{1}{2}\Delta t^2 \sum_j \frac{m_j}{(r^0_{ij})^3}(r^0_j - r^0_i)$$

Both our main equation for solving r^{n+1} and the initial conditions are within our **newtongravity.m** function which calculates the second order finite difference approximation for the Toomre model.

The functions that actually calculate the equations of motion are in **nbodyaccn.m** and **newtongravity.m.** The latter calls the former to actually get the objects to move. This function is then called in **initialize_sim.m** to first create the cores and the stars and then to actually allow the stars to move around the core. Within this function is also the code for the animation and the plots that show the initial galaxy before movement and the trajectories of the stars as well. This function is not generalised and only works for exactly two cores at the moment. To call this function, the user must input tmax, level, num_stars, r0, v0, and m where r0 and v0 are 2 x 3 arrays corresponding to initial position and velocity respectively and m is a 2 x 1 of the masses of the cores.

The **initialize_sim.m** function uses two functions within it, **gen_stars** and **initialize_starmovement** to generate stars around each core and then give the stars an initial velocity to start orbiting around the cores.

Results:

Convergence Test

The first thing tested was the convergence of our finite difference solution. For this, we used two cores with masses 1.0 and 0.5 that were in a circular orbit around each other. We took the x-coordinate then to do a three-level convergence test.

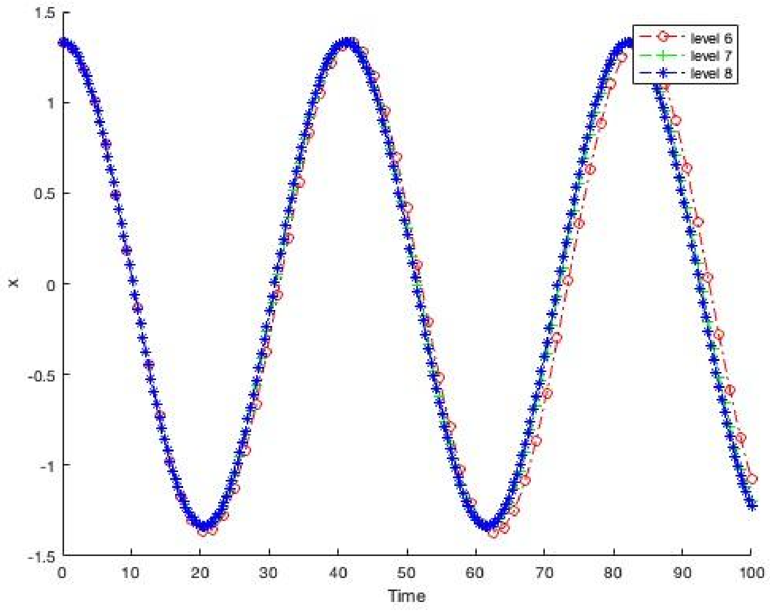

Figure 1: Plot of x-coordinates over time on three different levels.

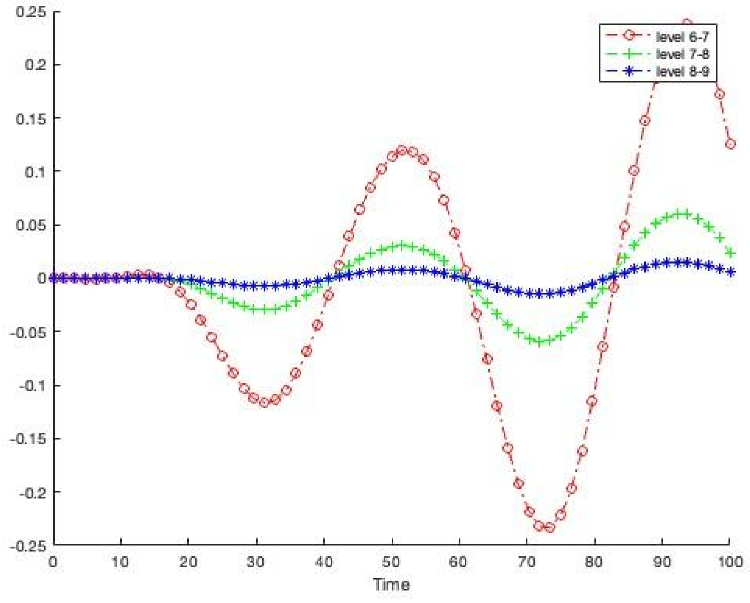

Figure 2: Plot of differences between levels.

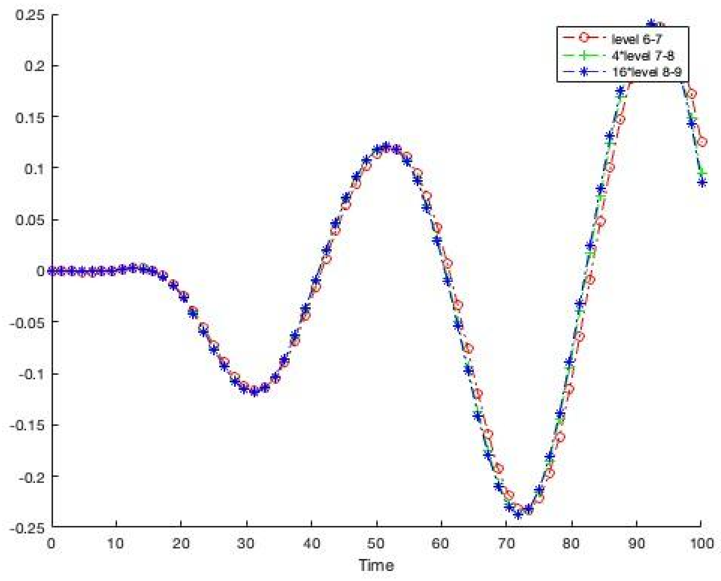

FIgure 3: Plot of differences between levels once scaled.

As we can see from the three figures, the convergence test is showing strong evidence that we have $O(\Delta t^2)$ error in our implementation. Figure 1 shows that the model on three different levels (6, 7, 8) are nearly matching and Figure 3 shows that when properly scaled, the differences are also nearly matching. To do the convergence test, we used a t_max of 100.

Simulation

First off, to test the **newtongravity.m** function, we used a test case of two cores with some arbitrary mass in mutual circular orbit in the x-y plane.

Test:

m1 = 1;
m2 = 0.5;
r = 0.75;

$$r1 = \frac{m_2 r}{m}$$

$$r2 = \frac{m_1 r}{m}$$

$$v1 = \sqrt{\frac{m_2 r_1}{r}}$$

$$v2 = \sqrt{\frac{m_1 r_2}{r}}$$

r_1(0) = (r1, 0, 0)
r_2(0) = (-r2, 0, 0)
v_1(0) = (0, v1, 0)
v_2(0) = (0, -v2, 0)

Using this test case we get a plot that looks like this:

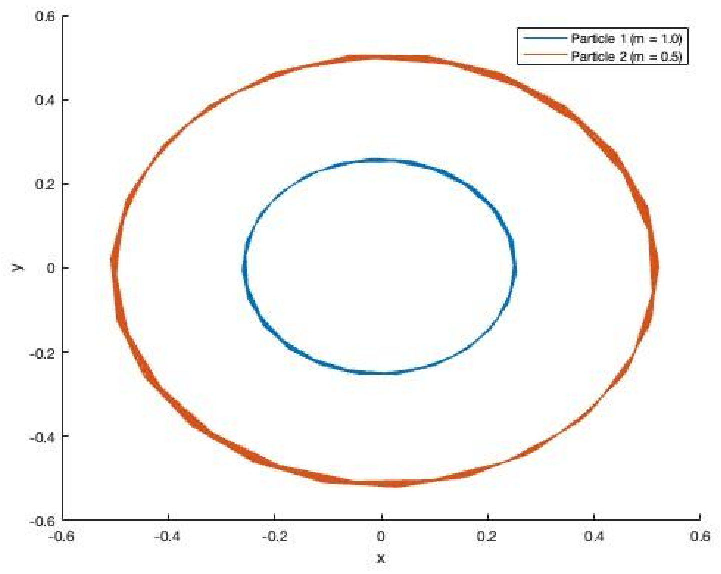

Figure 4: Test case of circular orbit for two cores.

This figure confirms that the two cores are indeed in circular orbit around each other.

Then we tested galaxy dynamics. We first used one galaxy at rest to test if the stars remained in a circular orbit around the core.

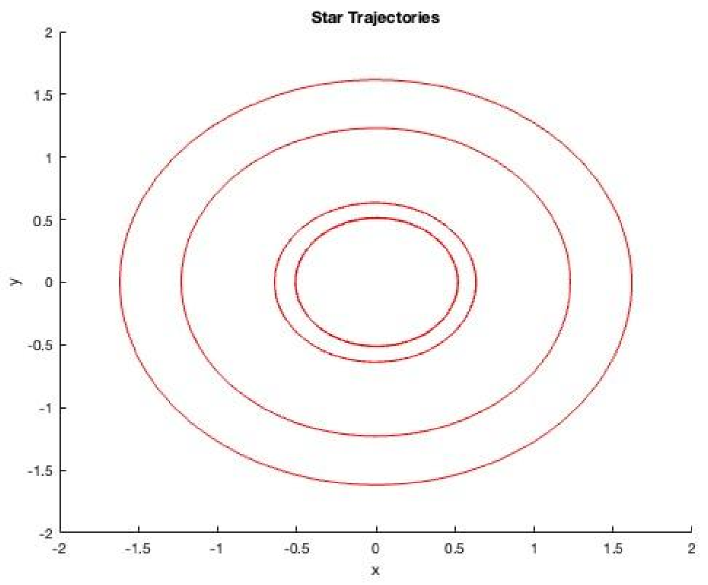

Figure 5: Test case of five stars where one of them is the core.

Here we can see that when we input num_stars = 5, we get four stars in a circular orbit around one core in the xy-plane.

We then tested a moving galaxy.

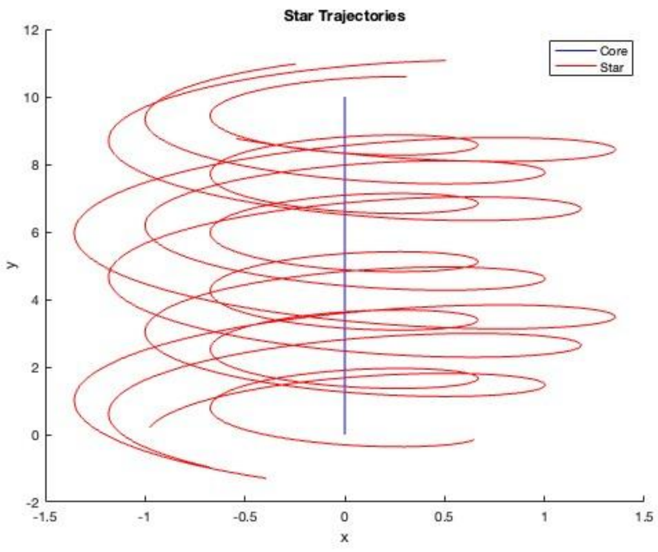

Figure 6: Test case of a moving galaxy.

Here we can see the same initial parameters as in figure 5 except that the core is moving with a velocity of 1 in the positive-y direction.

Finally, we created an animation that showed the collision between the two cores/galaxies. The initial conditions were set such that core 1 would be at rest and core 2 would be to the right of core 1 slowly moving to the left.

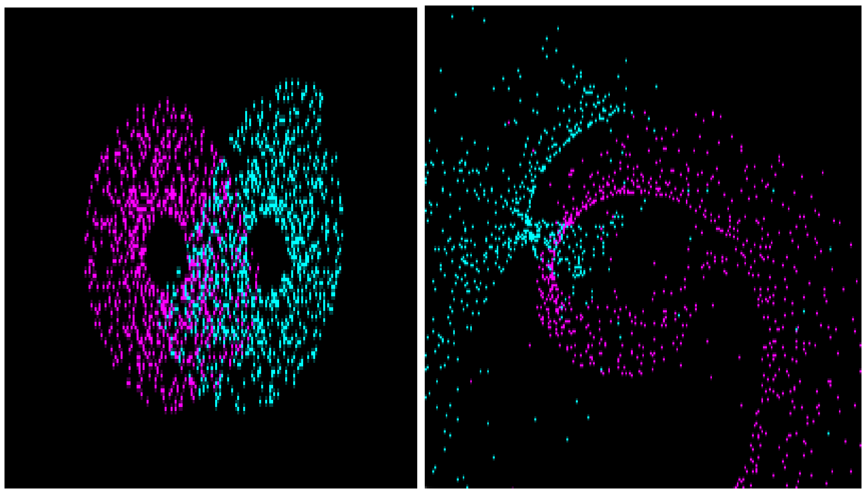

Figure 7: Galaxies right at collision and after collision.

From this figure, we can see that the turquoise galaxy moving to the left collides with the magenta galaxy at rest and soon after results in the annihilation of both galaxies as stars fly out from both.

Discussion/Conclusion:

The big difficulty that consumed the most time was trying to get the galaxy dynamics and collisions to work. For a long time, the stars around the core would immediately disperse and not create the circular motion around the core as expected. It took a lot of debugging to finally realise that the problem was in the **init_starmovement** function in **initialize_sim.m** where the magnitude of rij was simply not being calculated correctly when norm(rij) was used.

For this project, you want to run **runsim.m** to generate the animation, **runsim1.m** to generate the plot with the single moving core, and **newtongravity_conv.m** to run the convergence test.

ChatGPT and other generative AI tools were not used at all for this project.