Amelia Doetsch, William Hanley, Alex Kaddis

PHY 6860—Project 2A

April 20, 2019

I. Cluster growth with the DLA Model

In this problem, the goal is to create a cluster on a 2-dimensional lattice using the DLA model. In this case, the cluster should have a radius of 100. This is to say that the cluster should stop growing once a particle is added onto the cluster at a position 100 units away from the initial particle. The starting radius and escape radius are also taken to be 100 for simplicity.

First, an initial particle is placed at the center of the lattice. Then, a new particle is released at a distance of 100 from the center particle. It then performs a random walk. If the particle lands on a point in the lattice that is next to an already occupied lattice point, then it is added onto the cluster. If the particle escapes (i.e. its distance from the center exceeds 100), then the random walk is terminated and that particle is disregarded.

This process of releasing particles and performing random walks is repeated until the desired radius is achieved.

In order to plot the mass of the cluster as a function of its radius, each time a particle is added onto the cluster, the mass is incremented by one. Then, the radius of the cluster can be easily determined. The radius at each step and the mass at each step can be stored in two separate arrays and plotted.

A few sample clusters are shown below.

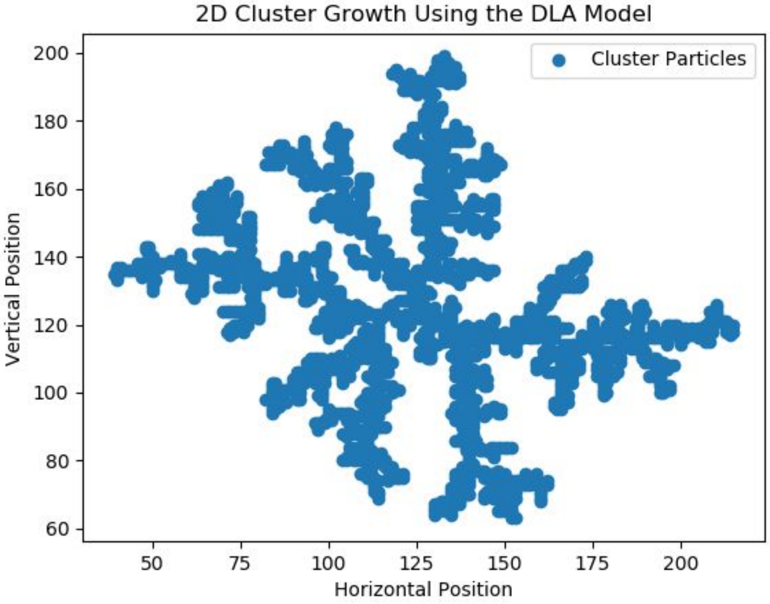

Figure 1: Cluster 1

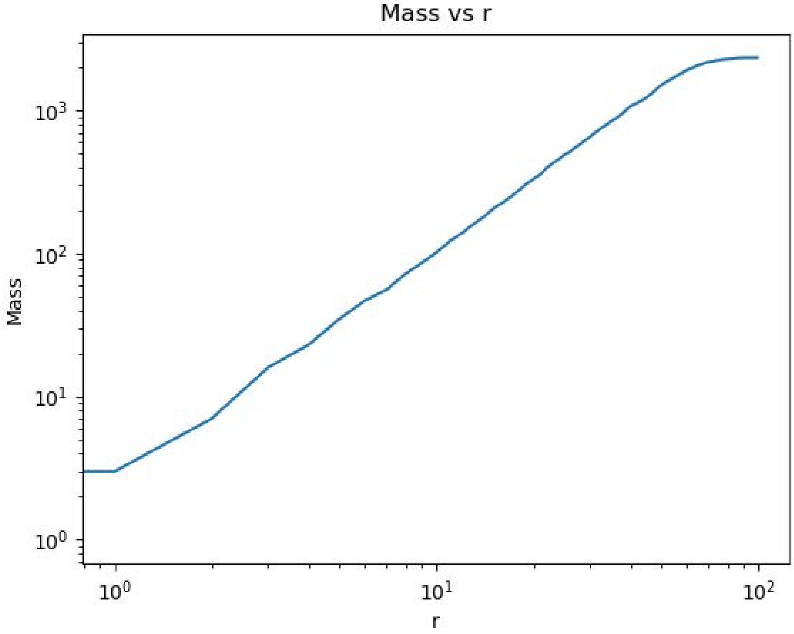

Figure 2: Cluster 1, mass vs. radius

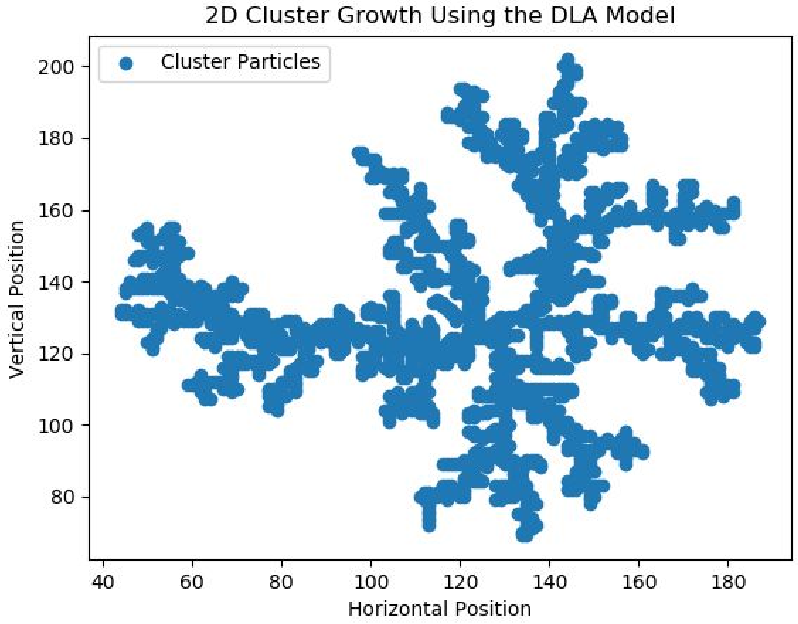

Figure 3: Cluster 2

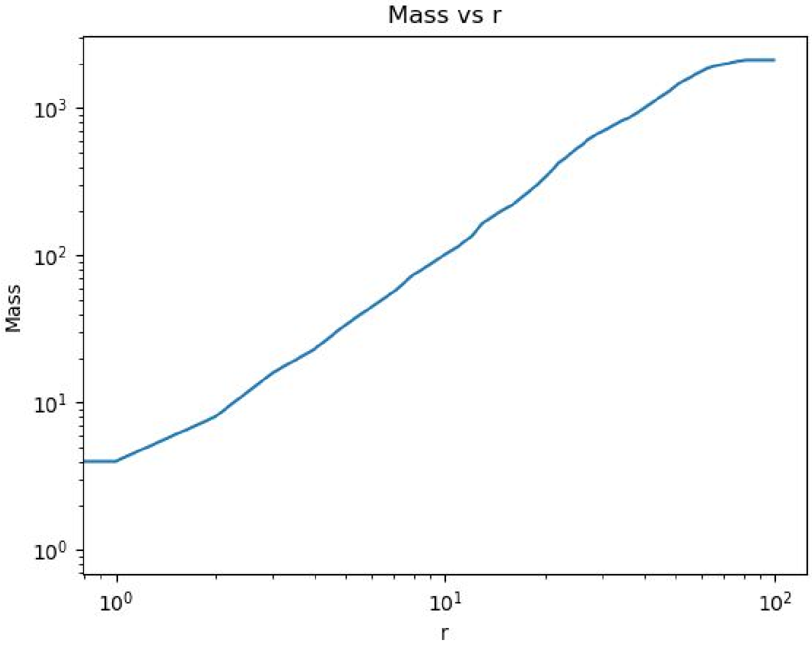

Figure 4: Cluster 2, mass vs. radius

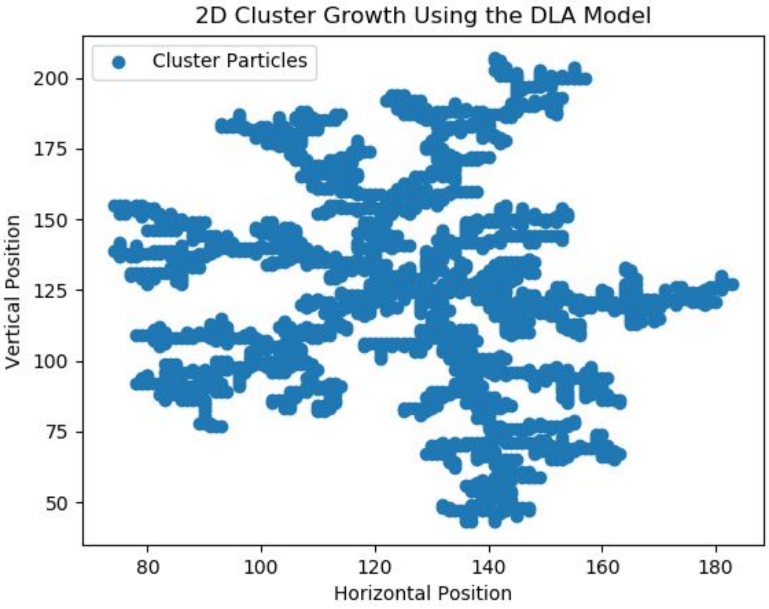

Figure 5: Cluster 3

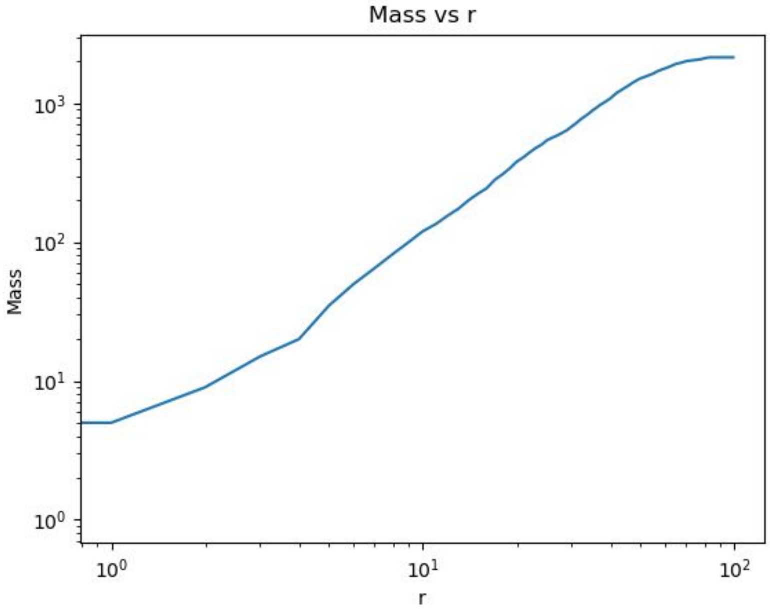

Figure 6: Cluster 3, mass vs. radius

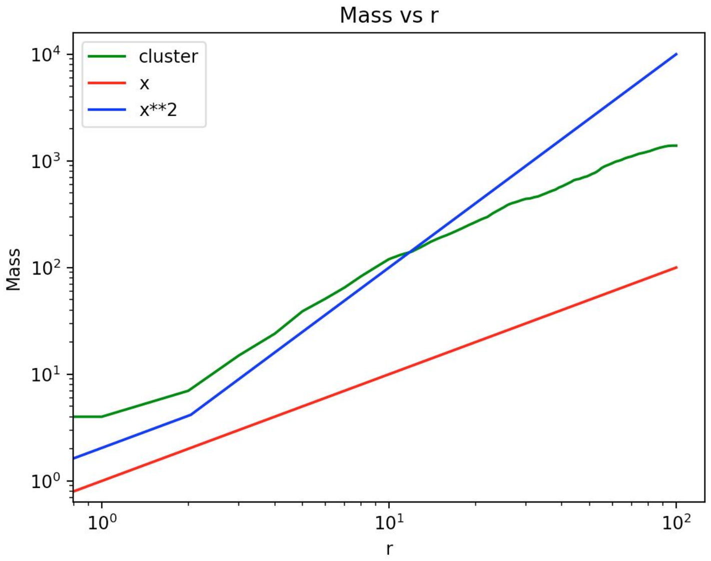

Figure 7: Comparison of dimensionality

II. Predator-Prey Model

In this problem, the goal is to create a predator-prey system like the one discussed in class. In this method we create 2D lattices for the positions and movements of both the predators and prey; and an additional 2D lattice to track the last time the predators ate. All parameters: grid size, starting populations, breeding ages and starvation ages are all adjustable variables at the beginning of the code.

The position lattices are randomly seeded with an adjustable starting population using a function created to seed the two dimensional lattices. Two functions were created to check for available positions for movement and for available prey for the predators. These functions also randomly choose from the all available positions for these movements. The prey on each movement can move and breed (if breeding age is reached) and once they move it is logged in their movement lattice. The predators can likewise move and breed. However, their movement is based first on the location of available prey, to which they will always move to eat. If no nearby prey is available the predator moves randomly to an open adjacent position. In both the eating and non-eating instances the predator breeds at their breeding age. At each time-step the last time the predator has eaten is also tracked in a separate lattice. If the predator reaches the starvation age it is deleted from all the position and starvation lattice.

The population of both the predator and prey are tracked at each time step and plotted as a function of time. In addition a contour plot of the positions of both the predators and prey is created every 50 time steps.

We were able to tune the parameters sufficiently to where every other instance or so we get a stable population for both the predator and prey that is consistent with the results described in the lecture. The position snapshot is help when running the code to quickly determine if it is a successful run or not and whether or not it would be best to re-run the code.

The position snapshots and populations vs. time graphs from a successful run are shown below.

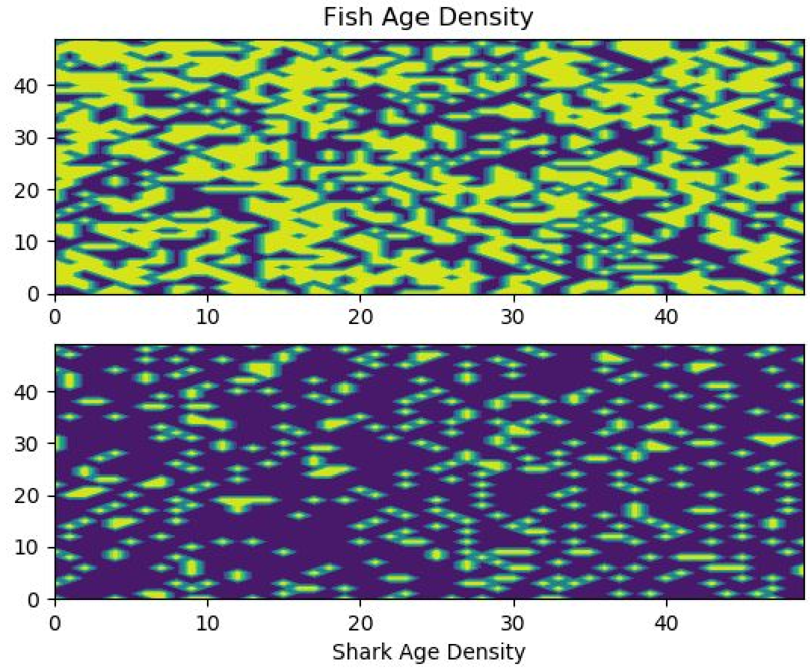

Figure 7: t = 0

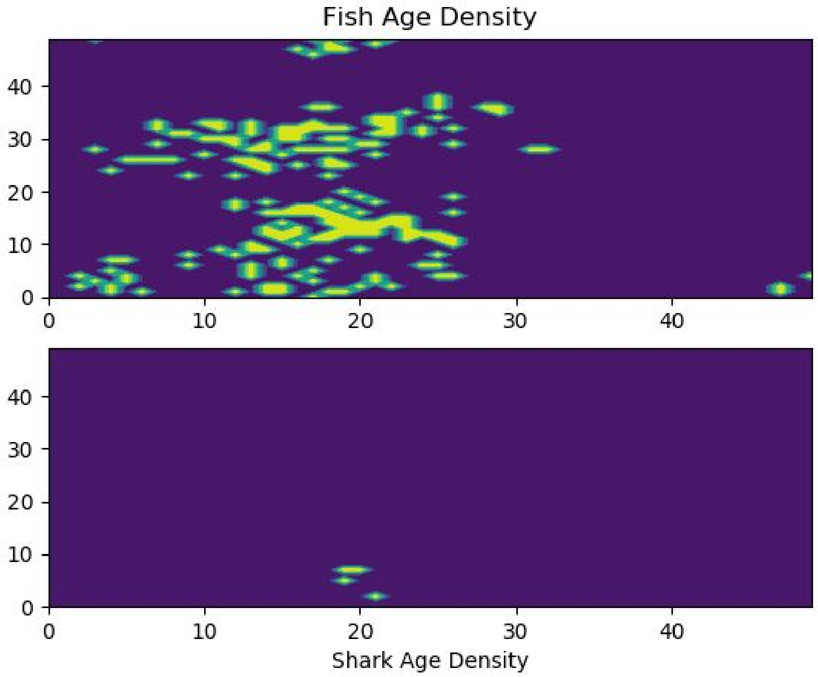

Figure 8: t = 50

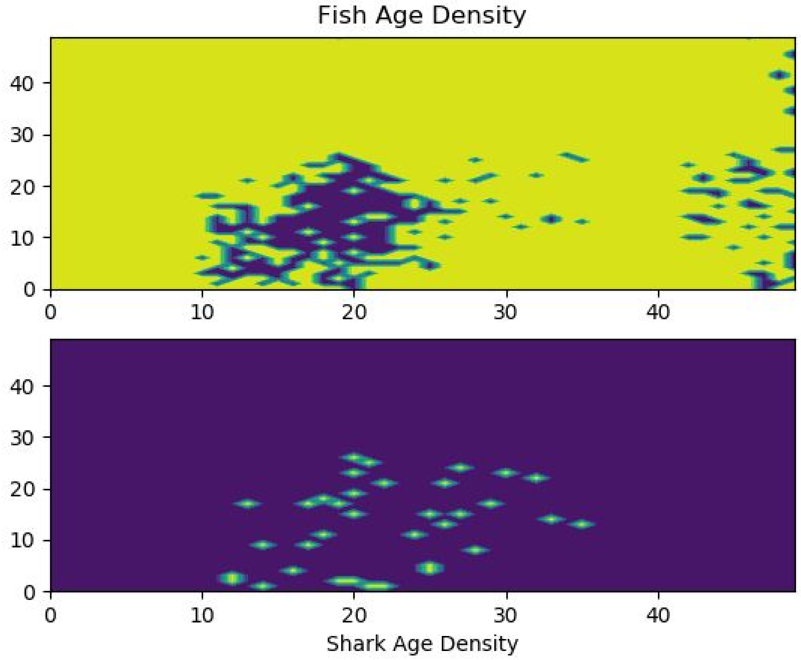

Figure 9: t = 100

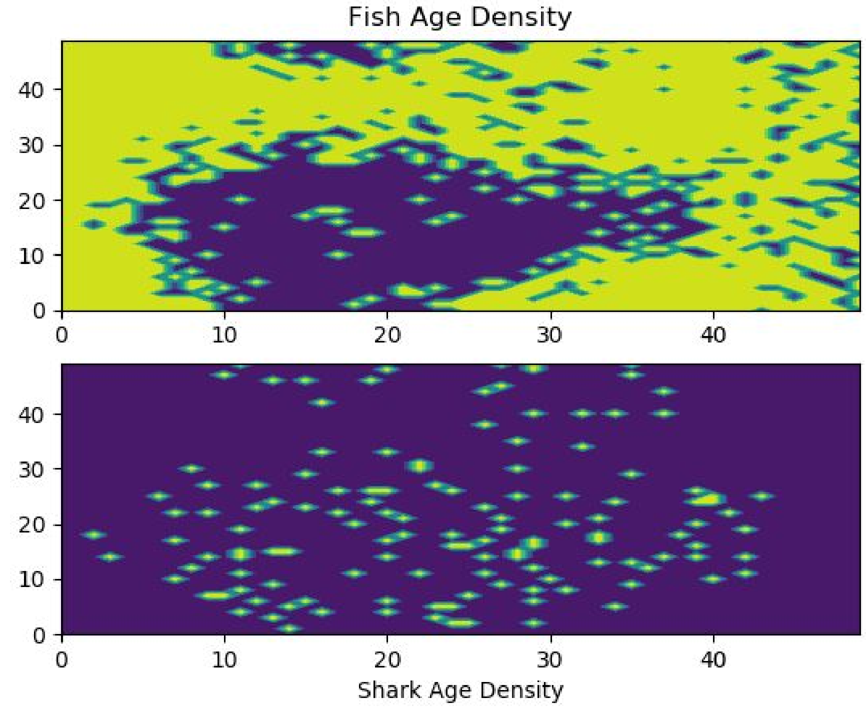

Figure 10: t = 150

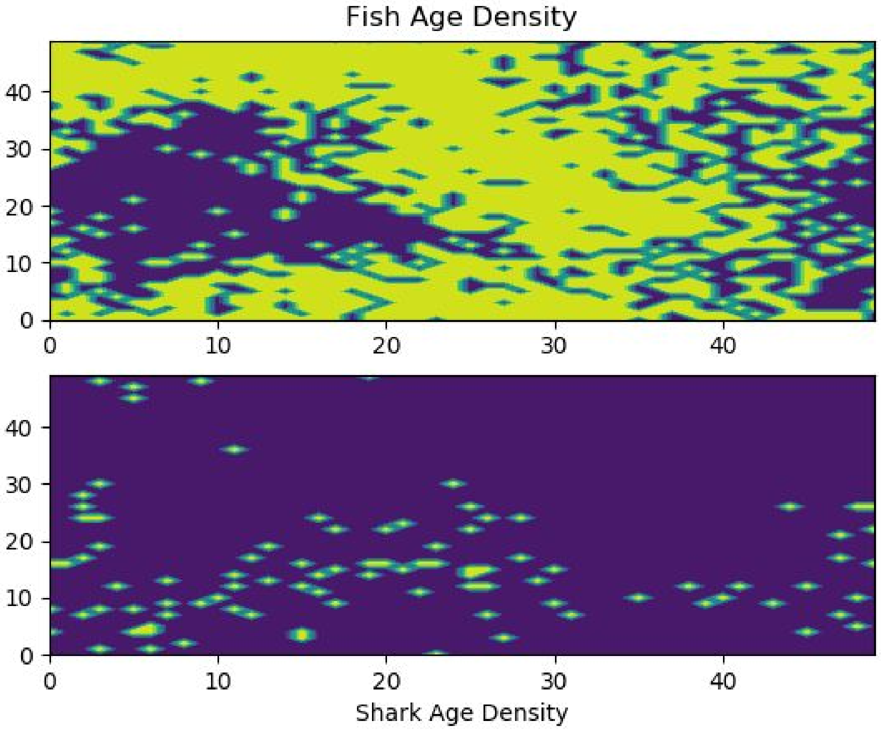

Figure 11: t = 250

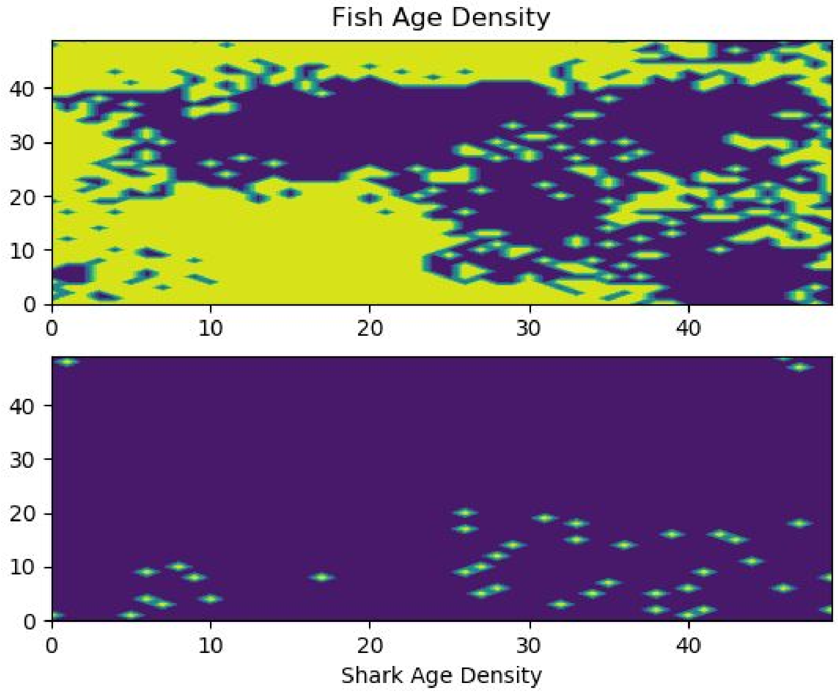

Figure 12: t = 350

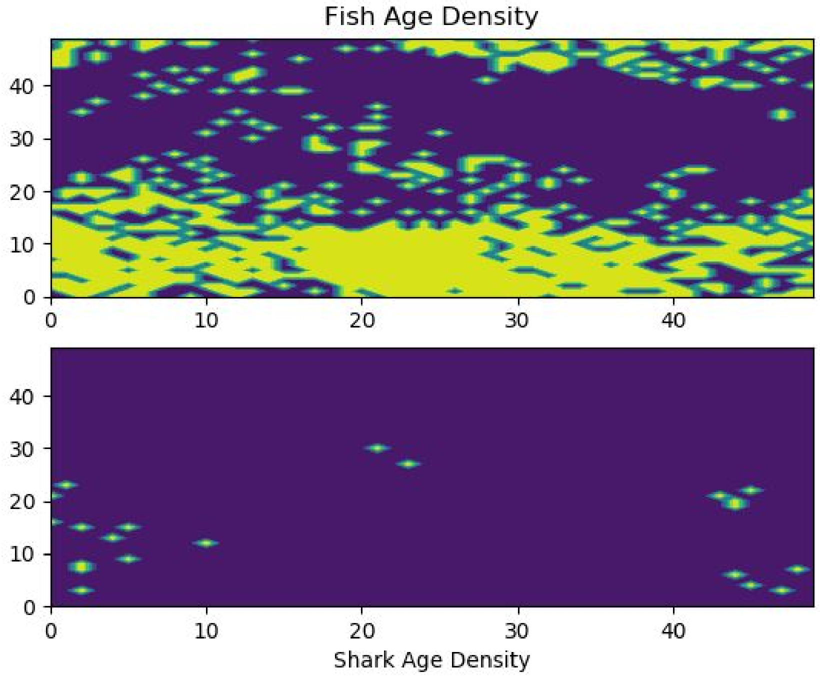

Figure 13: t = 450

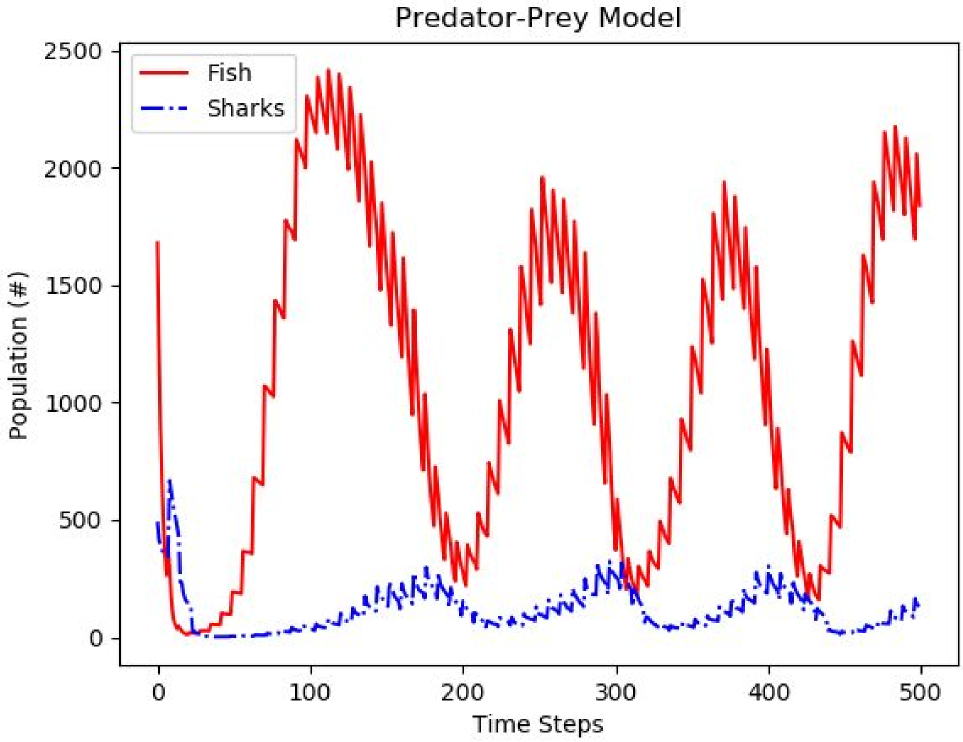

Figure 14: Population vs. Time