PREY-PREDATOR MODEL

Table of Contents

Identify the Model

Modeling population growth is important as it can give insight as to whether a population will drastically increase, decrease, or stay the same in the future. This would then allow for appropriate action to intervene if necessary. For this project, a population growth model will be created for foxes and rabbits, expanding on part 1. This time, the model will include interactivity between the rabbits and foxes, in which rabbits will act as the prey and foxes as the predator. It will be based off of the Lotka-Volterra model. Additionally, the model will include seasonal migration. The nature and limitations of the model are that the model does not consider any other prey or predators for either the rabbits or foxes. In the context of this project, it is assumed that foxes and rabbits are the only species on the island. It is also difficult to measure birth rate, death rate, and the interaction between both species. Predicting migration rates is not an easy task as well. The model also does not take into account a maximum population size for their species, assuming that there will always be enough space on the island to support any population size. Specific variables to identify in the model are growth rates (based on the prey growing without a predator and predator growing with enough prey to eat), the interactivity between both species (death rates due to the prey being preyed on and the predator not having enough food), initial populations in both locations of migration, and the rates of migration.

Make Assumptions

Assumptions of the scenario

- The birth and death rates always remain constant for both populations, other than what is caused by the other species
- The model is a closed system
- No part of the population will migrate out of this system and not return
- No creatures will migrate into the system
- There will be no change in environment that has an effect on the birth and death rate, other than the change in seasons
- Both species are unable to withstand the harsh weather and must migrate when the time comes, or they will die.

Lotka-Volterra model

- The model does not consider other predators or prey, only the two species and how they interact with each other
- Limitations of the Lotka-Volterra model result in our model never reaching an equilibrium. While population cycles do occur in nature, it is a disadvantage of this model that it does not have the possibility of an equilibrium state.
- there is no maximum population, so the populations can theoretically grow indefinitely

Assumptions in the implementation

- While the implementation allows for giving any input, we use as example:

Note: The Lotka-Volterra model does not include migration or multiple locations, that is added in the implementation.

$$\frac{dx}{dt} = (a - by)x$$

$$\frac{dy}{dt} = (-m + nx)y$$

Prey:
a (growth constant) = 0.03
b (interaction constant) = 0.0001
initial population in location 1 = 1500
initial population in location 2 = 0
migration rate = 0.15
Predator:
m (growth constant) = 0.03
n (interaction constant) = 5e-5 = 0.00005
initial population in location 1 = 1000
initial population in location 2 = 0
migration rate = 0.15

Solve the Model

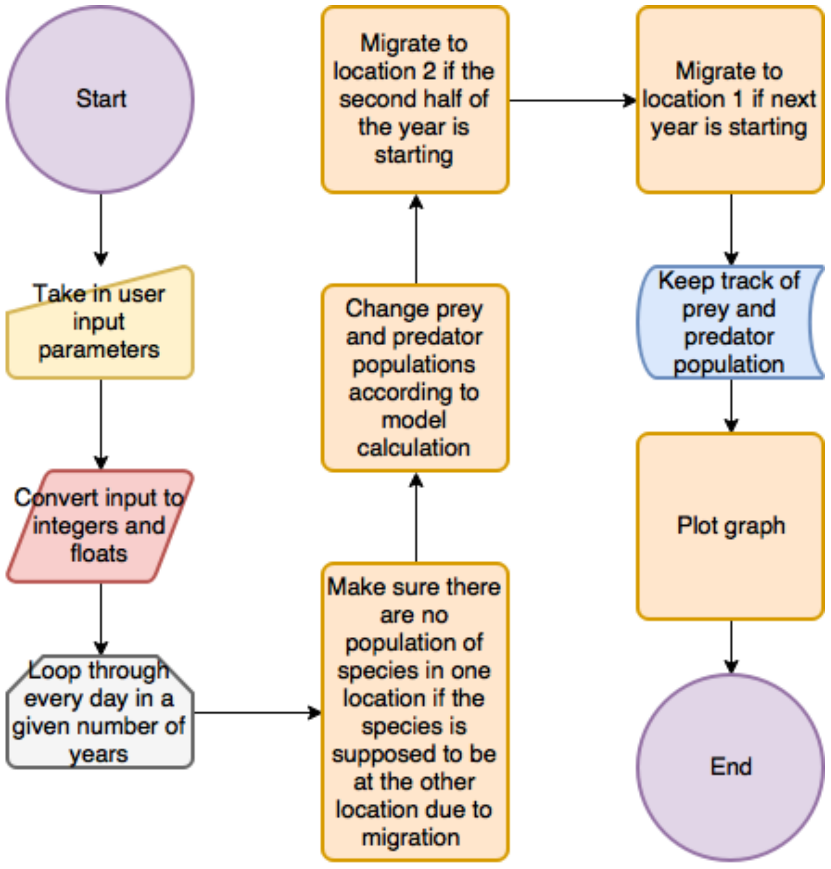
The model mainly adapts from the classic Lotka-Volterra model. The rate of change of both prey and predator populations is depicted by the equation below.

$$\frac{dx}{dt} = (a - by)x$$

$$\frac{dy}{dt} = (-m + nx)y$$

The variable x is the population of the prey and y is the population of the predator. Variable a is the rate of change of prey when they are not preyed on. Variable b is the chance of the prey being killed by the predator. Variable m is the rate of death of the predator if there is no prey exists. Variable n is the rate of growth of the predator that depends on the amount of the prey. However, the topic material article provided some insights on how to create a two-patch model with species migration. The model, therefore, has two locations where predators and prey migrate to due to weather change, thus the variables that represent the populations of prey and predator x and y becomes x1, x2, y1, y2, which stand for the populations in two different locations. Prey and predators only interact with each other if they are at the same location. Due to weather change, the prey would first stay at location 1 for five months. Then in the sixth month, they would migrate to location 2, with a daily migration rate represented by variable mx. After staying at location 2 for five months, the prey would then migrate back to location 1 at the twelfth month with the same migration rate. The predator is tougher, therefore migrates a month later than the prey. The rate of predator migrate is represented by variable my. Therefore, each year is a cycle of migration, that both species would first migrate to a second location in the middle of the year, and at the end of both species would migrate back. Both species take the same amount of time to migrate and spend the same amount of time at each location. Furthermore, if after the migration month, there are animals still left at the initial location, they would die because of the severe weather. Therefore, when one species should be at one location, it is assumed that there is no population of that species at the other location. The program would have a GUI provided for the user to interact with. The user can either choose the default data to see the result or type in their own data. After the user enters the data, the program then would able to plot the graph of populations of two species for a given number of years.

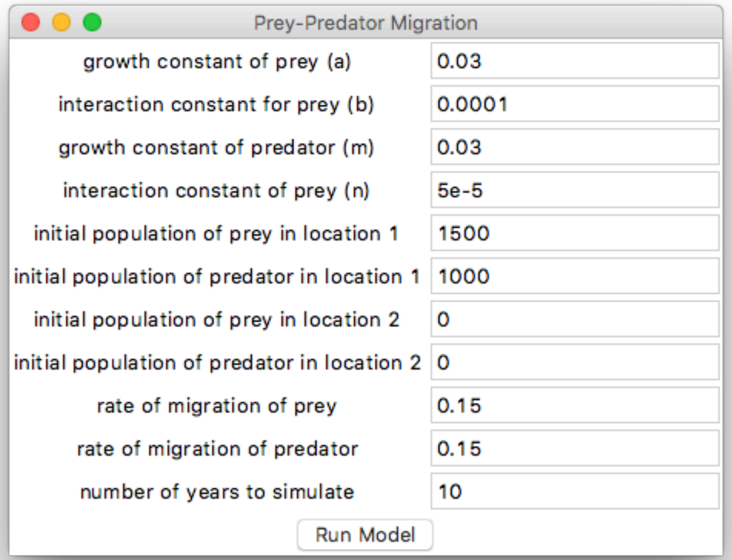
Flowchart


Start
Take in user input parameters
Convert input to integers and floats
Loop through every day in a given number of years
Make sure there are no population of species in one location if the species is supposed to be at the other location due to migration
Change prey and predator populations according to model calculation
Migrate to location 2 if the second half of the year is starting
Migrate to location 1 if next year is starting
Keep track of prey and predator population
Plot graph
End

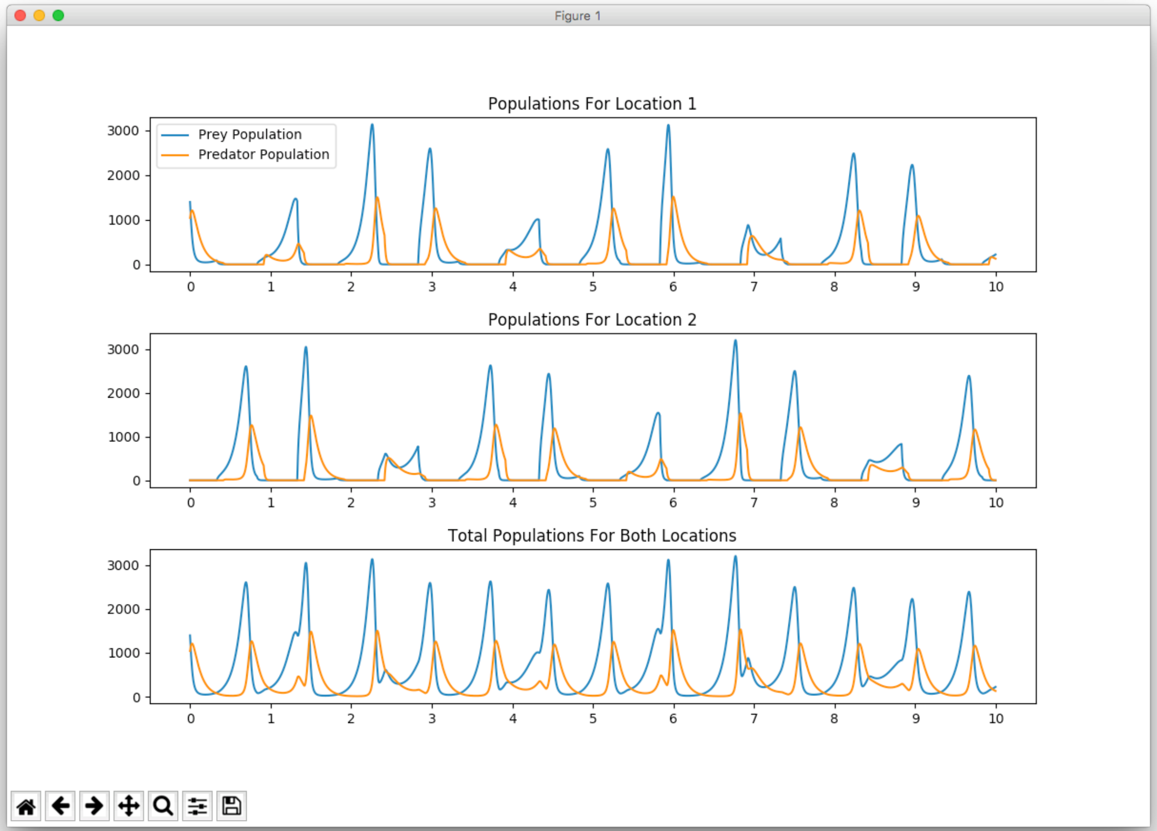
Code

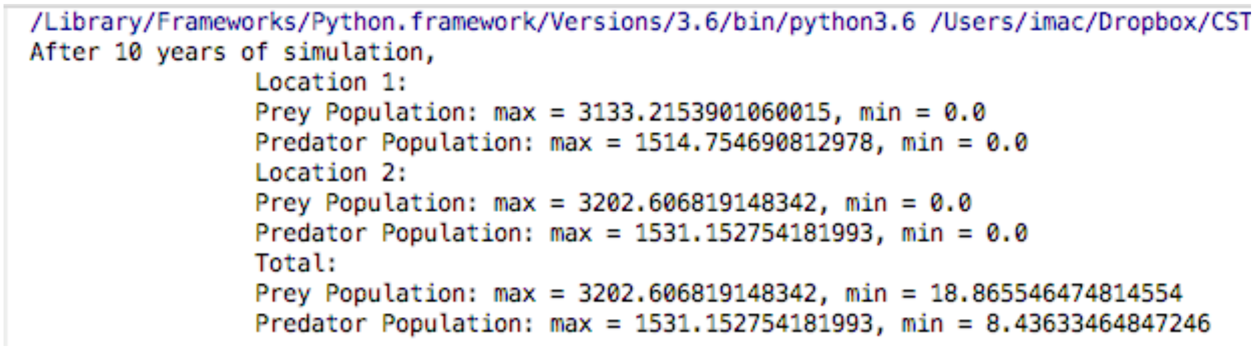
```
import numpy as np
import tkinter as tk
import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt
#################################### Set Up GUI ##################################
gui = tk.Tk() # setup tkinter
gui.title("Prey-Predator Migration")

# setup grid

# create labels
tk.Label(gui, text="growth constant of prey (a)").grid(row=0, column=0)
tk.Label(gui, text="interaction constant for prey (b)").grid(row=1, column=0)
tk.Label(gui, text="growth constant of predator (m)").grid(row=2, column=0)
tk.Label(gui, text="interaction constant of prey (n)").grid(row=3, column=0)
tk.Label(gui, text="initial population of prey in location 1").grid(row=4, column=0)
tk.Label(gui, text="initial population of predator in location 1").grid(row=5,
column=0)
tk.Label(gui, text="initial population of prey in location 2").grid(row=6, column=0)
tk.Label(gui, text="initial population of predator in location 2").grid(row=7,
column=0)
tk.Label(gui, text="rate of migration of prey").grid(row=8, column=0)
tk.Label(gui, text="rate of migration of predator").grid(row=9, column=0)
tk.Label(gui, text="number of years to simulate").grid(row=10, column=0)

# create text boxes
tk_a = tk.Entry(gui)
tk_a.grid(row=0, column=1)
tk_a.insert(-1, 0.03)
tk_b = tk.Entry(gui)
tk_b.grid(row=1, column=1)
tk_b.insert(-1, 0.0001)
tk_m = tk.Entry(gui)
tk_m.grid(row=2, column=1)
tk_m.insert(-1, 0.03)
tk_n = tk.Entry(gui)
tk_n.grid(row=3, column=1)
tk_n.insert(-1, 0.00005)
tk_x1 = tk.Entry(gui)
tk_x1.grid(row=4, column=1)
tk_x1.insert(-1, 1500)
tk_y1 = tk.Entry(gui)
tk_y1.grid(row=5, column=1)
tk_y1.insert(-1, 1000)
tk_x2 = tk.Entry(gui)
tk_x2.grid(row=6, column=1)
tk_x2.insert(-1, 0)
tk_y2 = tk.Entry(gui)
tk_y2.grid(row=7, column=1)
tk_y2.insert(-1, 0)
tk_mx = tk.Entry(gui)
tk_mx.grid(row=8, column=1)
tk_mx.insert(-1, 0.15)
tk_my = tk.Entry(gui)
tk_my.grid(row=9, column=1)
tk_my.insert(-1, 0.15)
tk_years = tk.Entry(gui)
tk_years.grid(row=10, column=1)
tk_years.insert(-1, 10)


def prey_rate(x, y, a, b):  # setup model for prey
    return (a - b * y) * x

def predator_rate(x, y, m, n):  # setup model for predator
    return (-m + n * x) * y

def model():
    a = float(tk_a.get()) # the rate of growth of prey without predator
    b = float(tk_b.get())# the rate of death of prey because of being preyed on
    m = float(tk_m.get()) # the rate of death of predator without food
    n = float(tk_n.get()) # the rate of growth of predator went eating prey
    x1 = int(tk_x1.get())# the initial population of the the prey in location 1
    y1 = int(tk_y1.get()) # the initial population of the predator in location 1
    x2 = int(tk_x2.get()) # the initial population of the prey in location 2
    y2 = int(tk_y2.get()) # the initial population of the predator in location 2

    mx = float(tk_mx.get()) # rate of the migration of prey during migrating season
    my = float(tk_my.get()) # rate of the migration of predator when poplation is
declining

    years = int(tk_years.get()) # number of years to run the simulation

    # initialize arrays keeping track to populations
    prey_populations_1 = []
    predator_populations_1 = []
    prey_populations_2 = []
    predator_populations_2 = []

    for year in range(years):   # loop through each year
        for month in range(12): # loop through each month
            if month in (5, 6, 7, 8, 9):    # when prey should be at location 2,
                #  all prey in location 1 should die because of severe weather
                x1 = 0
            elif month in (11, 0, 1, 2, 3): # when prey should be at location 1,
                #  all prey in location 2 should die because of severe weather
                x2 = 0
            if month in (6, 7, 8, 9, 10): # when predator should be at location 2,
                #  all predator in location 1 should die because of severe weather
                y1 = 0
            elif month in (0, 1, 2, 3, 4): # when predator should be at location 1,
                #  all predator in location 2 should die because of severe weather
                y2 = 0

            for day in range(30):   # loop through each day
                # population of both species grow according to the model
                x1 += prey_rate(x1, y1, a, b)
                x2 += prey_rate(x2, y2, a, b)
                y1 += predator_rate(x1, y1, m, n)
                y2 += predator_rate(x2, y2, m, n)

                # during designated migration month, populate migrate according
                # to migration rate every day to another location
                if month == 4:
                    migration = x1 * mx
                    x1 -= migration
                    x2 += migration
                if month == 10:
                    migration = x2 * mx
                    x2 -= migration
                    x1 += migration

                # the predator migrate one month after the prey do
                if month == 5:
                    migration = y1 * my
                    y1 -= migration
                    y2 += migration
                if month == 11:
                    migration =y2 * mx
                    y2 -= migration
                    y1 += migration

                # keep track of population over the years
                prey_populations_1.append(x1)
                prey_populations_2.append(x2)
                predator_populations_1.append(y1)
                predator_populations_2.append(y2)

    # calculate total populations for two species
    prey_populations_total = [x + y for x, y in zip(prey_populations_1,
prey_populations_2)]
    predator_populations_total = [x + y for x, y in zip(predator_populations_1,
predator_populations_2)]

    print(f"""After {str(years)} years of simulation,
                Location 1:
                Prey Population: max = {max(prey_populations_1)}, min =
{min(prey_populations_1)}
                Predator Population: max = {max(predator_populations_1)}, min =
{min(predator_populations_2)}
                Location 2:
                Prey Population: max = {max(prey_populations_2)}, min =
{min(prey_populations_2)}
                Predator Population: max = {max(predator_populations_2)}, min =
{min(predator_populations_2)}
                Total:
                Prey Population: max = {max(prey_populations_total)}, min =
{min(prey_populations_total)}
                Predator Population: max = {max(predator_populations_total)}, min =
{min(predator_populations_total)}
                """)

    n = np.arange(0, years, 1/360)

    # plot population graph for location 1
    plt.figure(1, figsize=(12, 8))
    plt.subplot(311)
    plt1 = plt.plot(n, prey_populations_1, label = "Prey Population")
    plt2 = plt.plot(n, predator_populations_1, label = "Predator Population")
    plt.xticks(np.arange(0, years+1))
    plt.title("Populations For Location 1")
    plt.legend()

    # plot population graph for location 2
    plt.subplot(312)
    plt.plot(n, prey_populations_2, n, predator_populations_2)
    plt.xticks(np.arange(0, years+1))
    plt.title("Populations For Location 2")

    # plot total population graph
    plt.subplot(313)
    plt.plot(n, prey_populations_total, n, predator_populations_total)
    plt.title("Total Populations For Both Locations")
    plt.xticks(np.arange(0, years+1))

    plt.subplots_adjust(hspace=0.4)
    plt.show()



tk.Button(gui, text="Run Model", command=model).grid(row=11, column=0, columnspan=2)

tk.mainloop()
```

Results



growth constant of prey (a)	0.03
interaction constant for prey (b)	0.0001
growth constant of predator (m)	0.03
interaction constant of prey (n)	5e-5
initial population of prey in location 1	1500
initial population of predator in location 1	1000
initial population of prey in location 2	0
initial population of predator in location 2	0
rate of migration of prey	0.15
rate of migration of predator	0.15
number of years to simulate	10




```
/Library/Frameworks/Python.framework/Versions/3.6/bin/python3.6 /Users/imac/Dropbox/CST
After 10 years of simulation,
                Location 1:
                Prey Population: max = 3133.2153901060015, min = 0.0
                Predator Population: max = 1514.754690812978, min = 0.0
                Location 2:
                Prey Population: max = 3202.606819148342, min = 0.0
                Predator Population: max = 1531.152754181993, min = 0.0
                Total:
                Prey Population: max = 3202.606819148342, min = 18.865546474814554
                Predator Population: max = 1531.152754181993, min = 8.43633464847246
```

Verify the Model

By utilizing the Lotka-Volterra model, there were several input-based parameters that could be used in modifying the annual population trends for the rabbits and the foxes. After thorough analysis and interpretation of the implementation of the model, it can be concluded that the model does address the problem. The trends that are produced as a result of feasible input (i.e. not allowing initial predator population to greatly exceed initial prey population, disproportionate growth constants, etc.) show clear delays in population growth and decay. That is, as the rabbit population increases, the fox population will also increase shortly after. As the fox population increases, the rabbit population will decrease and the fox population will decrease as a result. Additionally, the populations will migrate between locations on a seasonal basis, which accounts for the population fluctuations when viewing both locations. It also makes sense because it is rare to observe instances where the fox population exceeds the rabbit population while the rabbit population is still increasing rapidly. In cases like this, it would likely mean that there are more predators than prey, and could contribute to the extinction of both species. In cases where there is a good balance of population fluctuations as demonstrated by our model, both species can thrive as long as no other outside variables (e.g. hunting, different predators/prey) are introduced. Overall, the Lotka-Volterra model that was implemented in the model addresses the problem and makes common sense.

Maintain the Model

One improvement to the model is that additional predators and prey can be added for both rabbits and foxes because it is unlikely that foxes will not have any predators and feed only on rabbits. It is also unlikely that foxes are rabbits' only predator and that rabbits never have to worry about their food supply. Also, instead of assuming that each month has 30 days, it would be more realistic to actually use the correct number of days for each month and take into account leap year. This is because the further in the time the model tries to predict population sizes, the more inaccurate the sizes become for the actual date. Another improvement is that the model could consider a degree of internal competition among the prey for the limited resources (protection, water, food, etc.) and degree of competition among the predators for the finite amount of prey. This is because larger groups will have a higher chance of conflict when fighting for what they need to survive.

To improve the implementation, graphs would be automatically updated as values are into the GUI so the user would not have to click “Run Model” to see the affect. Also, the GUI could make it clear the range of appropriate values to enter for each variable so the user has an idea of how big each input should be. These values could then be checked by the program when entered so the program will only run the model under appropriate conditions. Finally, allowing the user to select a specific point on one of the graphs to see the (x,y) data would help to get a more accurate reading the exact value could be difficult.

References

AL-DARABSAH, I. i., XLANHUA TANG2, t., & YUAN YUAN1, 2. y. (2016). A PREY-PREDATOR MODEL WITH MIGRATIONS AND DELAYS. Discrete & Continuous Dynamical Systems - Series B, 21(3), 787-761.