HW3 Report

Kangqi Yu 121090735

November 14, 2023

1 Introduction

In this report, I investiage the population of giant pandas in Wuyipeng, Wolong Natural Reserve, China. I try to utilize simulation to find the difference of different initial population and the effect of human intervention on the survival of giant pandas. I find that younger initial population and positive human intervention helps pandas a lot based on the simulation result. At the end of this report, I give some suggestions to improve the survival of giant pandas and improvement ideas to the simulation model.

2 Data

2.1 Data Source

Our simulations utilize some insights from Wei, Feng, and Hu's 1997 study, "Population Viability Analysis Computer Model of Giant Panda Population in Wuyipeng, Wolong Natural Reserve, China", from "Bears: Their Biology and Management". This work gives us some crucial demographic and ecological inforamtion about the population of giant pandas in Wolong Natural Reserve, China. We will use this information to construct our model and simulate the population of giant pandas in Wuyipeng, Wolong Natural Reserve, China.

2.2 Data Description

Our simulation model is based on the following information from the work of Wei, Feng, and Hu (1997):

- Reproduction is a very important parameter in the Vortex model. Normally, the breeding age of pandas in the wild is 7 years for females and 8 years for males. Sex ratios are 1:1 and maximum reproductive age is 20 years. Females produce at best 1 litter/year with a maximum of 2 offspring. The reproductive rate of giant pandas is low. Each year 37.5% of females produce no offspring, 58.3% of females produce 1 offspring, and 4.2% of females produce 2 offspring.

- Age-specific mortality (%) of wild giant pandas is shown below.

Table 1: Age-specific mortality (%) of wild giant pandas.

Age	Male	Female
0-1	40.00	40.00
1-2	9.67	9.67
2-3	3.14	3.14
3-4	1.52	1.52
4-5	1.55	1.55
5-6	1.57	1.57
6-7	1.60	1.60
7-8 (Females ≥ 7)	3.45	13.33
≥ 8	14.16	

- We can use the availability of bamboo biomass to estimate carrying capacity for the giant panda. During 1981-82, the total bamboo biomass was measured of $2.9 * 10^7$ culms, with 18 pandas consuming approximately $8.39 * 10^6$ culms. At that time, there is a carrying capacity of 62 pandas. By 1986, the bamboo biomass had declined to $1.7 * 10^7$ culms, with 15 pandas consuming $5.15 * 10^6$ culms, thus reducing the carrying capacity to 50 pandas. This represents an annual decrease in bamboo biomass of 4.21% from 1981 to 1986.

- Because the bamboo flowering cycle is about 60 years (Qin 1990), we estimated that catastrophes influenced panda's survival and reproduction 1.67% (1/60).

3 Model

3.1 Captured Features

Besides the features mentioned in the data description, we also consider the following features:

3.1.1 Initial Population

I use four types of inital population to simulate:

1. Juvenile population: From 0 to 4 years, there are 5 pandas respectively.
2. Middle-aged population: From 7 to 11 years, there are 5 pandas respectively.
3. Senile population: From 13 to 17 years, there are 5 pandas respectively.
4. Uniform population: From 0 to 24 years, there are 1 panda respectively.

3.1.2 Human Intervention

We consider the following human intervention:

1. Maintaining food: We hold the capacity of food constantly.
2. Help: We provide medical help to pandas. The mortality of pandas is reduced by 50%.

3.2 Simulation Rules

3.2.1 Mortality

In the simulation of one year, we first simulate the mortality of pandas. If we have the help from humans, the mortality will be reduced half than normal.

3.2.2 Capacity

Then, we assume normally there is a capacity decline rate of 4.21% per year. If we maintain the capacity of food, the capacity will be constant. I also considered the flowering of bamboo, that is there are 1/60 chance for capacity to be reduced quarter. When the population of pandas exceeds the capacity, the newborn pandas will die firstly, and then elder.

3.2.3 Reproduction

We pay attention on the sex ratio of pandas and the capacity when the simulation go to reproduction stage.

4 Empirical results

4.1 Juvenile population as Initial Population

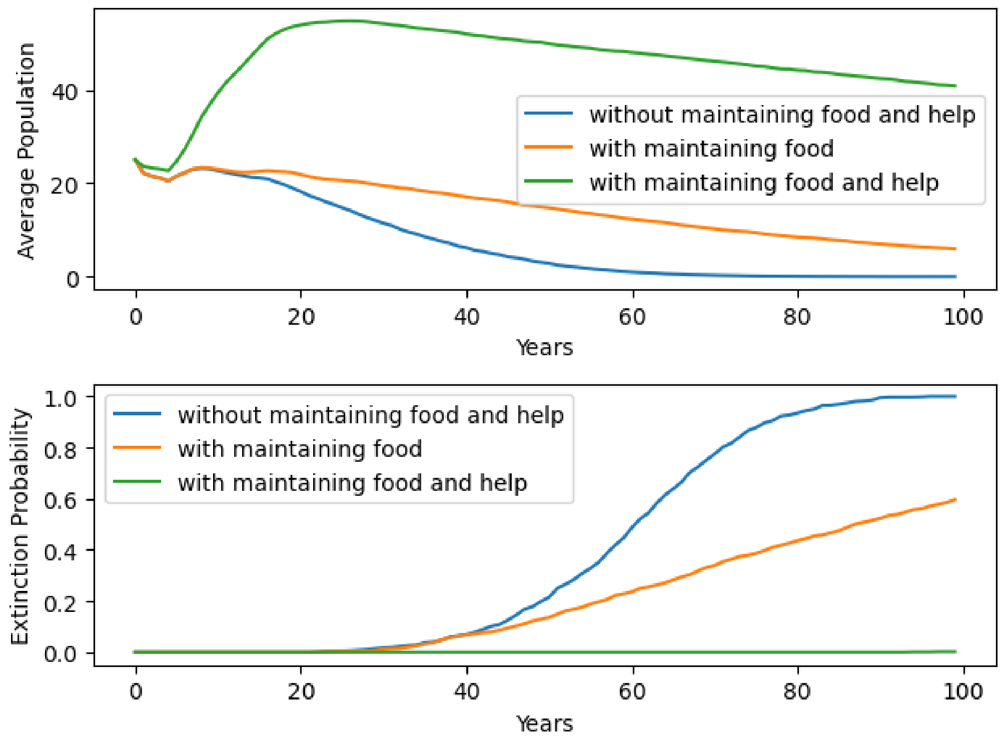


Figure 1: Juvenile population

As shown in the above figure, we can see the porbability of extinction is about 0, 0.6 and 1.0 for the population with maintaining food and help; with maintaining food; without them. For the population trend, the first two populations are decaying, and the last one is decaying a little, growing and decaying.

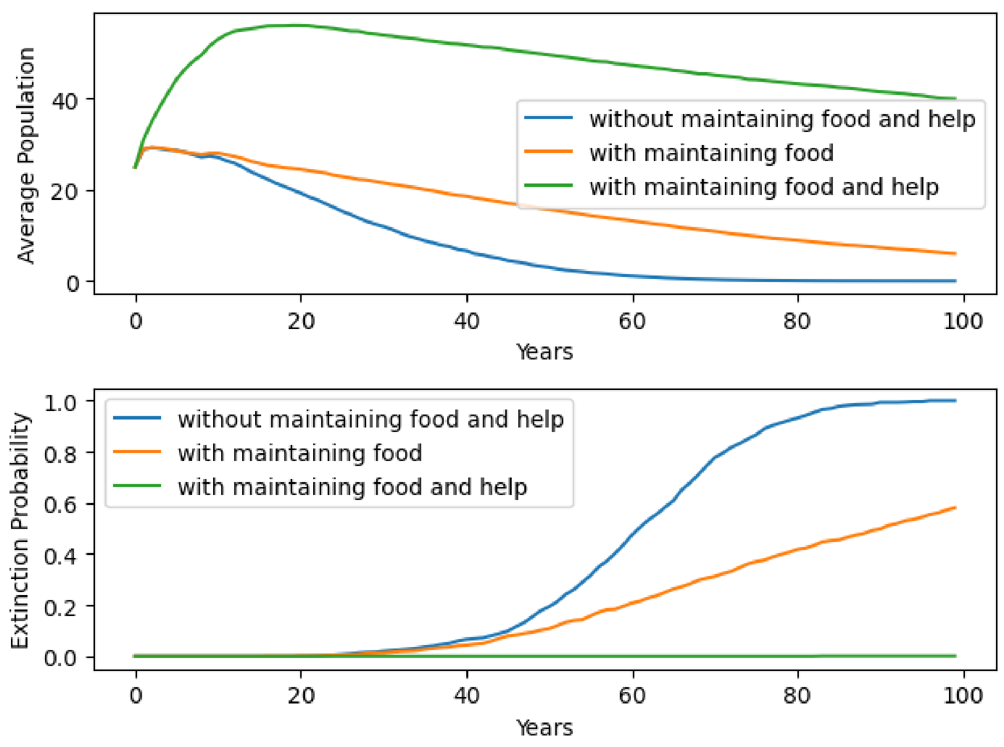


Figure 2: Middle-aged population

4.2 Middle-aged population as Initial Population

As shown in the above figure, we can see the porbability of extinction is about 0, 0.6 and 1.0 for the population with maintaining food and help; with maintaining food; without them. For the population trend, the first two populations are decaying, and the last one is growing firstly and decaying.

4.3 Senile population as Initial Population

As shown in the above figure, we can see the porbability of extinction is about 0, 0.8 and 1.0 for the population with maintaining food and help; with maintaining food; without them. For the population trend, the first two populations are decaying, and the last one is fluctuating firstly and decaying.

4.4 Uniform population as Initial Population

As shown in the above figure, we can see the porbability of extinction is about 0, 0.7 and 1.0 for the population with maintaining food and help; with maintaining food; without them. For the population trend, the first two populations are decaying, and the last one is growing firstly and decaying.

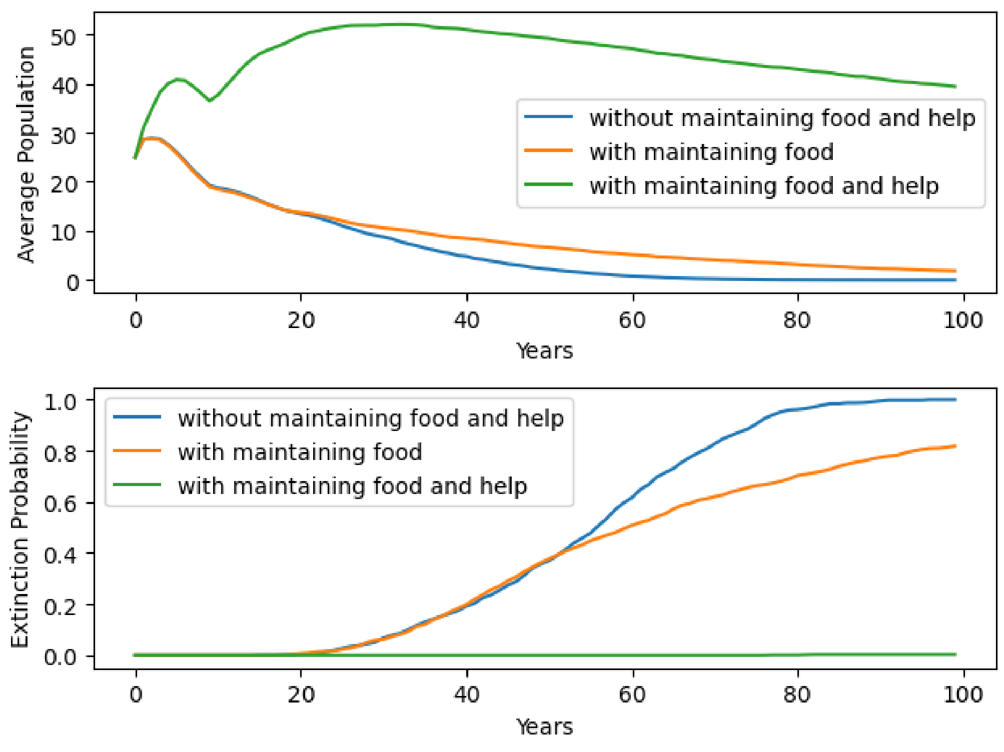


Figure 3: Senile population

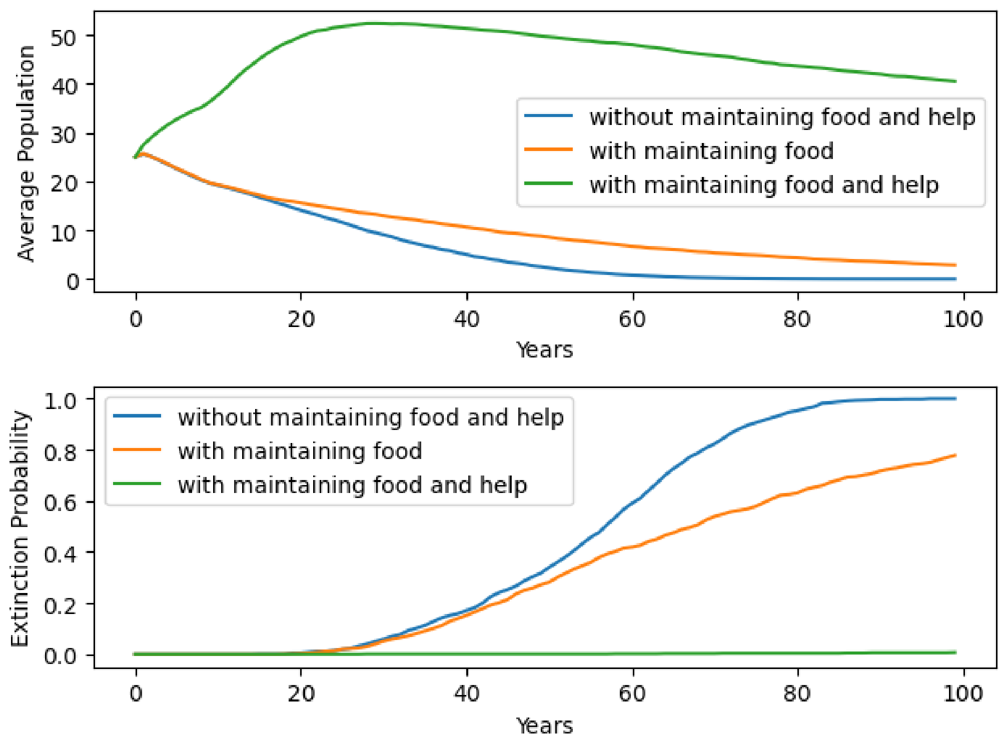


Figure 4: Uniform population

The difference between the four types of initial population is that the number of peaks in the average population subplot and the location of peaks in the figure. The mid-aged population has the earliest peak and the senile population has two peaks.

5 Discussions

5.1 Suggestions

1. We can provide medical help to pandas to reduce the mortality.
2. We can maintain the capacity of food to reduce the probability of extinction.
3. We can provide more food to pandas to increase the carrying capacity.
4. We can pay attention on the flowering of bamboo and take measures to reduce the impact of it.
5. We can set a suitable inital population to increase the probability of survival.

5.2 Limitations and Improvement Ideas

1. The model is too simple to simulate the real situation (the time of death and birth is just a point).
2. The inital population may not be accurate.
3. Some assumption are not accurate enough.

To improve this model, we can consider the following aspects:

1. We can consider the time of death and birth is some periods, that is split a year more finely.
2. We can use the real data of inital population.
3. We can use more accurate assumption, such dynamic mortality and capacity decline rate.

MAT3300_HW3_Code_121090735

November 14, 2023

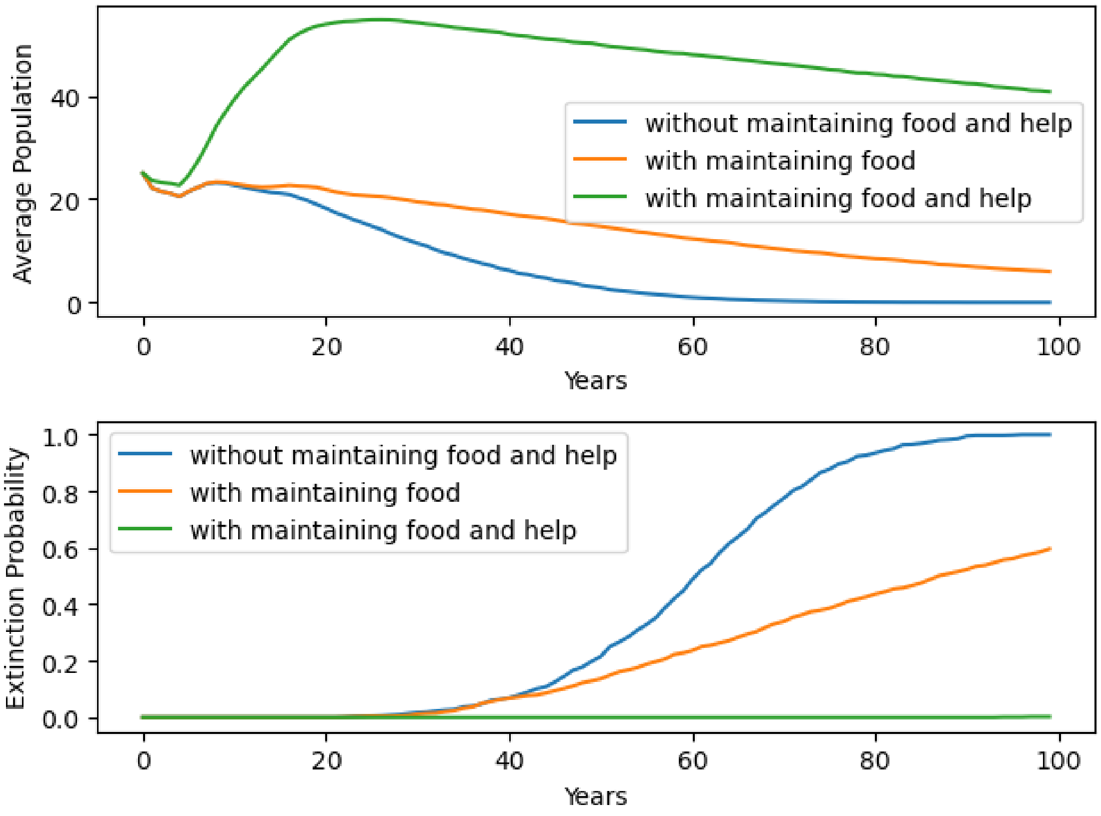
[ ]:
```
import random
import matplotlib.pyplot as plt
```

[ ]:
```
# Parameters
MAX_REPRODUCTIVE_AGE = 20
BREEDING_AGE_F = 7
BREEDING_AGE_M = 8
CARRYING_CAPACITY_INIT = 62
CARRYING_CAPACITY_DECLINE_RATE = 0.0421
YEARS_SIMULATED = 100
INITIAL_POPULATION = 25
SIMULATION_RUNS = 1000
BAMBOO_FLOWERING_YEARS = 60
```

[ ]:
```
# Mortality rates
MORTALITY_RATE_F = [40.00, 9.67, 3.14, 1.52, 1.55, 1.57, 1.60, 13.33, 13.33]
MORTALITY_RATE_M = [40.00, 9.67, 3.14, 1.52, 1.55, 1.57, 1.60, 3.45, 14.16]

# Reproduction rates
NO_OFFSPRING = 0.375
ONE_OFFSPRING = 0.583
TWO_OFFSPRING = 0.042
```

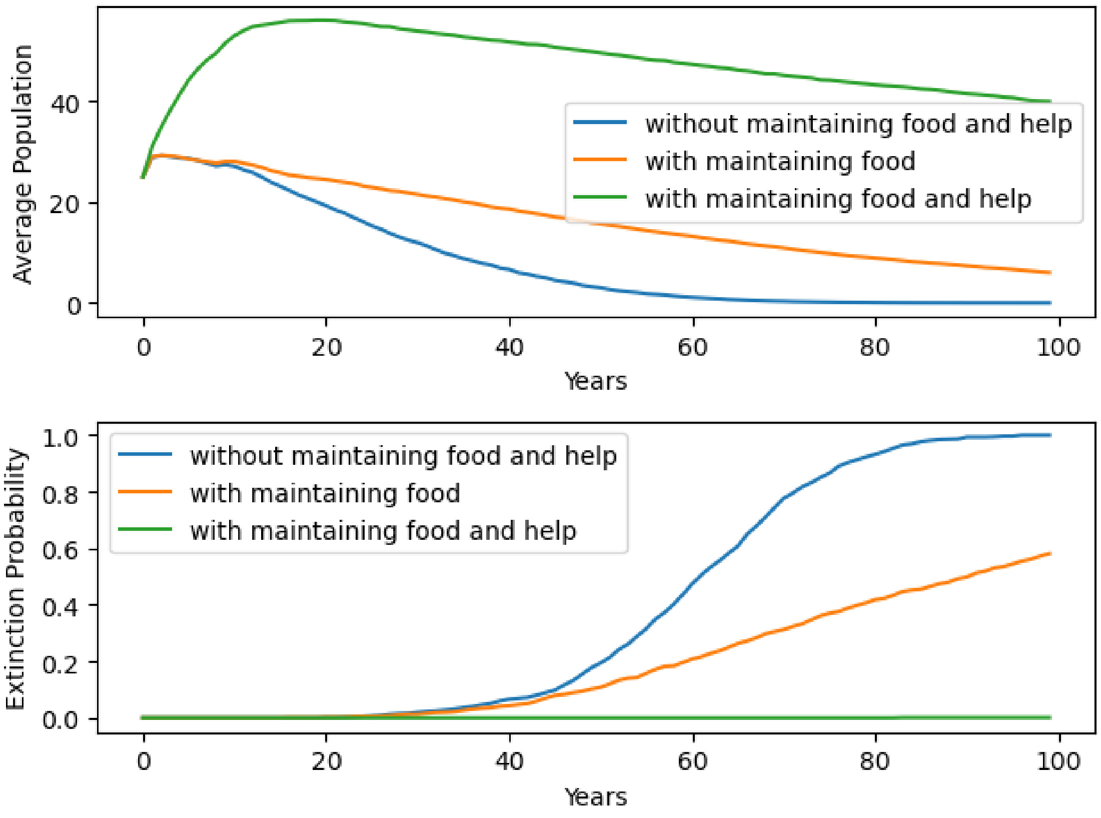
[ ]:
```
class Pandas():
    def __init__(self, age_structure, maintaining_food=False, help = False):
        self.age_structure = age_structure
        self.maintaining_food = maintaining_food
        self.help = help

    def initialize_population(self):
        self.population = []
        self.carrying_capacity = CARRYING_CAPACITY_INIT
        for age, count in self.age_structure.items():
            for _ in range(count):
                sex = "m" if random.random() < 0.5 else "f"
                self.population.append({"age": age, "sex": sex})

    def simulate_year(self):
        new_population = []
        for panda in self.population:
            # check whether panda survives in this year
            mortality_rate = MORTALITY_RATE_M[panda["age"] if panda["age"] <= 8␣
↪else 8] if panda["sex"] == "m" else MORTALITY_RATE_F[panda["age"] if␣
↪panda["age"] <= 8 else 8]
            if self.help:
                mortality_rate *= 0.5
            if random.random() < mortality_rate/100:
                continue
            else:
                panda["age"] += 1
                new_population.append(panda)

        # check carrying capacity
        if self.maintaining_food:
            self.carrying_capacity = self.carrying_capacity
        else:
            self.carrying_capacity = self.carrying_capacity * (1 -␣
↪CARRYING_CAPACITY_DECLINE_RATE)
        if random.random() < 1/BAMBOO_FLOWERING_YEARS:
            self.carrying_capacity = self.carrying_capacity * 0.75 # quarter of␣
↪bamboo flowering

        # if the capacity is not enough, the death order is newborns, then␣
↪others from oldest to youngest.
        if (len(new_population) > self.carrying_capacity):
            new_population = sorted(new_population, key=lambda x: x["age"])
            new_population = new_population[:int(self.carrying_capacity)]
            self.population = new_population

        # check whether panda reproduces in this year
        else:
            # check whether there is any male panda
            new_population = sorted(new_population, key=lambda x: x["sex"])
            reproduction_f_number = 0
            reproduction_m_number = 0
            for i in new_population:
                if i["sex"] == "m":
                    if (i["age"] - 1 >= BREEDING_AGE_M) and (i["age"] - 1 <␣
↪MAX_REPRODUCTIVE_AGE):
                        reproduction_m_number += 1
                else:
                    if (i["age"] - 1 >= BREEDING_AGE_F) and (i["age"] - 1 <␣
↪MAX_REPRODUCTIVE_AGE):
                        reproduction_f_number += 1
            if reproduction_f_number == 0 or reproduction_m_number == 0:
                self.population = new_population
            else:
                for _ in range(reproduction_f_number):
                    if random.random() < NO_OFFSPRING:
                        continue
                    elif random.random() < ONE_OFFSPRING + NO_OFFSPRING:
                        sex = "m" if random.random() < 0.5 else "f"
                        new_population.append({"age": 0, "sex": sex})
                    else:
                        for i in range(2):
                            sex = "m" if random.random() < 0.5 else "f"
                            new_population.append({"age": 0, "sex": sex})
                    if (len(new_population) >= self.carrying_capacity):
                        break
                self.population = new_population

    def simulate(self, years):
        record = []
        self.initialize_population()
        record.append(len(self.population))
        for _ in range(years):
            self.simulate_year()
            if len(self.population) == 0:
                break
            else:
                record.append(len(self.population))
        record = record + [0] * (years - len(record))
        return record

    def multiple_run(self):
        records = []
        for _ in range(SIMULATION_RUNS):
            records.append(self.simulate(YEARS_SIMULATED))
        return records

def plot_result(records, label):
    plt.subplot(2, 1, 1)
    plt.plot(range(YEARS_SIMULATED), [sum([i[j] for i in records]) /␣
 ↪SIMULATION_RUNS for j in range(YEARS_SIMULATED)], label=label)
    plt.xlabel("Years")
    plt.ylabel("Average Population")
    plt.legend()
    plt.subplot(2, 1, 2)
    plt.plot(range(YEARS_SIMULATED), [sum([i[j] == 0 for i in records]) /␣
 ↪SIMULATION_RUNS for j in range(YEARS_SIMULATED)], label=label)
    plt.xlabel("Years")
    plt.ylabel("Extinction Probability")
    plt.legend()
```

[ ]:
```
population1 = Pandas({0: 5, 1: 5, 2: 5, 3: 5, 4: 5})
records1 = population1.multiple_run()
population2 = Pandas({0: 5, 1: 5, 2: 5, 3: 5, 4: 5}, maintaining_food=True)
records2 = population2.multiple_run()
population3 = Pandas({0: 5, 1: 5, 2: 5, 3: 5, 4: 5}, maintaining_food=True,␣
  ↪help=True)
records3 = population3.multiple_run()
plot_result(records1, label="without maintaining food and help")
plot_result(records2, label="with maintaining food")
plot_result(records3, label="with maintaining food and help")
plt.tight_layout()
plt.show()
```

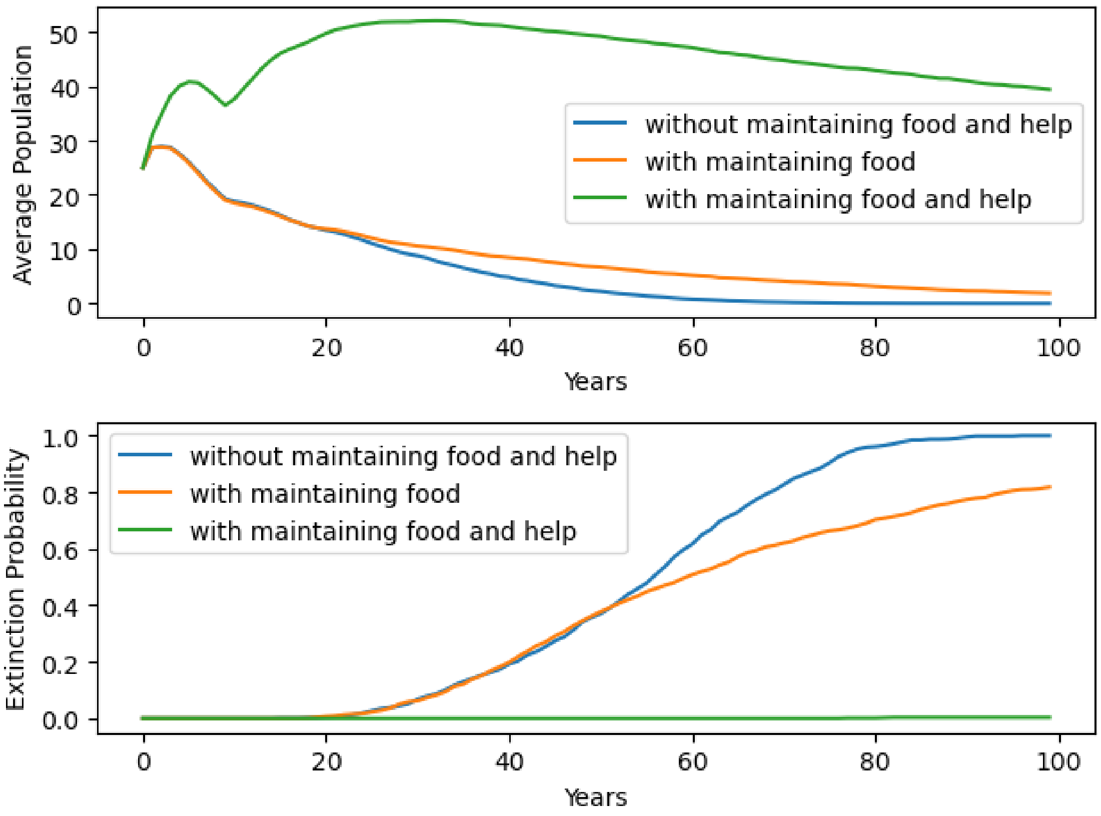


[ ]:
```
population4 = Pandas({7: 5, 8: 5, 9: 5, 10: 5, 11: 5})
records4 = population4.multiple_run()
population5 = Pandas({7: 5, 8: 5, 9: 5, 10: 5, 11: 5}, maintaining_food=True)
records5 = population5.multiple_run()
```

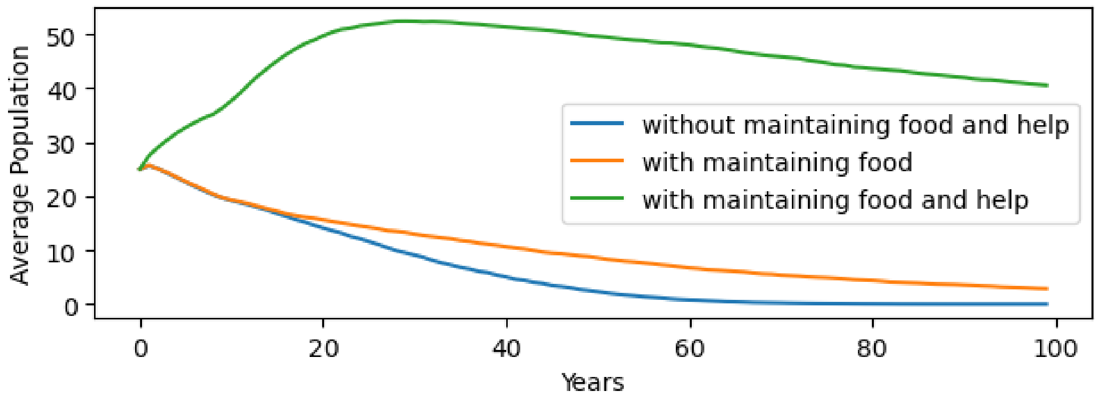
```
population6 = Pandas({7: 5, 8: 5, 9: 5, 10: 5, 11: 5}, maintaining_food=True,␣
  ↪help=True)
records6 = population6.multiple_run()
plot_result(records4, label="without maintaining food and help")
plot_result(records5, label="with maintaining food")
plot_result(records6, label="with maintaining food and help")
plt.tight_layout()
plt.show()
```

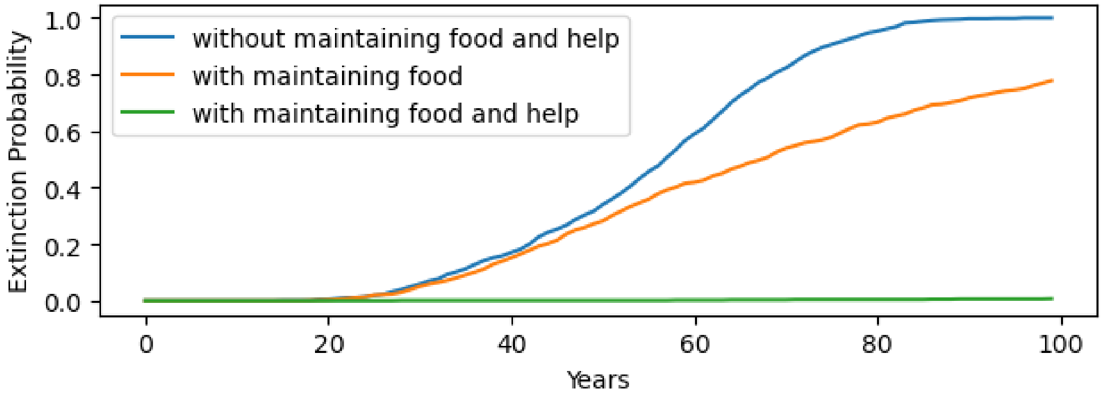


[ ]:
```
population7 = Pandas({13: 5, 14: 5, 15: 5, 16: 5, 17: 5})
records7 = population7.multiple_run()
population8 = Pandas({13: 5, 14: 5, 15: 5, 16: 5, 17: 5}, maintaining_food=True)
records8 = population8.multiple_run()
population9 = Pandas({13: 5, 14: 5, 15: 5, 16: 5, 17: 5},␣
  ↪maintaining_food=True, help=True)
records9 = population9.multiple_run()
plot_result(records7, label="without maintaining food and help")
plot_result(records8, label="with maintaining food")
plot_result(records9, label="with maintaining food and help")
plt.tight_layout()
plt.show()
```



[ ]:
```
population10 = Pandas({0: 1, 1: 1, 2: 1, 3: 1, 4: 1, 5: 1, 6: 1, 7: 1, 8: 1, 9:␣
  ↪1, 10: 1, 11: 1, 12:1, 13: 1, 14: 1, 15: 1, 16: 1, 17: 1, 18:1, 19: 1, 20:␣
  ↪1, 21: 1, 22: 1, 23: 1, 24: 1})
records10 = population10.multiple_run()
population11 = Pandas({0: 1, 1: 1, 2: 1, 3: 1, 4: 1, 5: 1, 6: 1, 7: 1, 8: 1, 9:␣
  ↪1, 10: 1, 11: 1, 12:1, 13: 1, 14: 1, 15: 1, 16: 1, 17: 1, 18:1, 19: 1, 20:␣
  ↪1, 21: 1, 22: 1, 23: 1, 24: 1}, maintaining_food=True)
records11 = population11.multiple_run()
population12 = Pandas({0: 1, 1: 1, 2: 1, 3: 1, 4: 1, 5: 1, 6: 1, 7: 1, 8: 1, 9:␣
  ↪1, 10: 1, 11: 1, 12:1, 13: 1, 14: 1, 15: 1, 16: 1, 17: 1, 18:1, 19: 1, 20:␣
  ↪1, 21: 1, 22: 1, 23: 1, 24: 1}, maintaining_food=True, help=True)
records12 = population12.multiple_run()
plot_result(records10, label="without maintaining food and help")
plot_result(records11, label="with maintaining food")
plot_result(records12, label="with maintaining food and help")
plt.tight_layout()
plt.show()
```


Average Population
Years
without maintaining food and help
with maintaining food
with maintaining food and help


Extinction Probability
Years
without maintaining food and help
with maintaining food
with maintaining food and help