Modeling Population Dynamics: Logistic Growth, Lotka-Volterra, and PINNs

1st Sri Kalyan Reddy Akiti
dept. Data Science, college of science
Texas A and M university
Corpus Christi, United States
sakiti@islander.tamucc.edu

I. Introduction

Understanding how populations grow, interact, and change over time is important in fields like biology, ecology, and resource management. This paper focuses on using mathematical tools and modern computational methods to study these changes and predict behaviors in real-world systems.

We explore three key areas:

- **Logistic Growth with Harvesting:** This model examines how a fish population grows when there are natural limits like food or space, while also considering the impact of constant fishing. We use a numerical approach to simulate how the population changes over time and what happens when harvesting becomes too high.
- **Predator-Prey Dynamics:** This section looks at the relationship between predators (like foxes) and their prey (like rabbits). By using a mathematical model, we can see how the populations of both species rise and fall in cycles, depending on factors like reproduction and predation.
- **Solving Equations with Neural Networks:** Here, we use a modern technique called Physics-Informed Neural Networks (PINNs) to solve a simple equation that describes how things change over time. This method uses the power of neural networks to find accurate solutions without needing traditional methods.

The goal of this report is to show how these models can help us better understand population changes and make smarter decisions for managing resources and ecosystems. The results provide insights into both traditional mathematical approaches and new computational methods.

II. Related Work

Understanding population dynamics has been a key area of study for many years, as it helps explain how species grow, interact, and adapt over time. Over the years, several models and techniques have been developed to analyze and predict these changes.

One commonly used model is the **logistic growth model**, which explains how populations grow when they are limited by factors like food, space, or other resources. Extensions of this model include the impact of human activities, such as harvesting through fishing or hunting, which can significantly affect population levels.

The **Lotka-Volterra model** is another major development, focusing on predator-prey relationships. This model helps us understand how predators and their prey depend on each other, leading to natural cycles of population growth and decline. It has been widely used to study ecosystems and the balance between species.

In recent years, modern techniques such as **Physics-Informed Neural Networks (PINNs)** have emerged to solve mathematical problems related to population dynamics. These methods use neural networks to find solutions for equations, especially when data is limited or traditional methods are too complicated.

This report builds on these established methods and combines traditional models with newer computational techniques. By doing so, it provides insights into population changes and explores innovative approaches for solving related problems in ecology and mathematics.

III. Experiments

A. Experiment 1: Logistic Growth with Harvesting

Objective:
To study how a fish population grows over time under natural limitations, such as food and space, while also incorporating the effects of constant harvesting. The focus is on understanding how different harvesting rates impact the population and identifying when overharvesting leads to extinction.

Methodology:

1) **Logistic Growth Model:**
The model is based on the equation:

$$\frac{dN}{dt} = r \cdot N \cdot \left(1 - \frac{N}{K}\right) - H$$

Where:

- N: Fish population.
- $r = 0.5$: Maximum growth rate.
- $K = 2 \times 10^6$: Carrying capacity (maximum population the environment can sustain).
- H: Harvesting rate (tons per year).

2) **Forward Euler Method:**
The equation was solved numerically using the Forward Euler Method with a time step ($h = 0.1$) over a simulation period of 10 years.

3) **Harvesting Rates:**
Four constant harvesting rates were tested: $H = 2 \times 10^4$, $H = 5 \times 10^4$, $H = 1 \times 10^5$, and $H = 2 \times 10^5$. The simulation stops harvesting when the population reaches zero, representing extinction.

Code:

```
In [12]: import matplotlib.pyplot
import numpy

harvest_rates = [2e4, 5e4, 1e5, 2e5] # tons / year

# This is used to keep track of the data that we want to plot.
data = []

def logistic_growth():
    maximum_growth_rate = 0.5 # 1 / year
    carrying_capacity = 2e6 # tons

    end_time = 10. # years
    h = 0.1 # years
    num_steps = int(end_time / h)
    times = h * numpy.array(range(num_steps + 1))

    fish = numpy.zeros(num_steps + 1) # tons
    fish[0] = 2e5

    for harvest_rate in harvest_rates:
        is_extinct = False

        for step in range(num_steps):
            if is_extinct:
                fish_next_step = 0.
            else:
                fish_next_step = fish[step] + h*(maximum_growth_rate*(1.-(fish[step]/carrying_capacity))*fish[step] -
                if fish_next_step <= 0.:
                    is_extinct = True
                    fish_next_step = 0.
            fish[step+1] = fish_next_step

        data.append(([time for time in times], [f for f in fish],
            str(harvest_rate)))

    return fish

fish = logistic_growth()


def plot_me():
    fish_plots = []
    for (times, fish, rate_label) in data:
        fish_plots.append(matplotlib.pyplot.plot(times, fish, label=rate_label))
    matplotlib.pyplot.legend([str(rate) for rate in harvest_rates], loc='upper right')
    axes = matplotlib.pyplot.gca()
    axes.set_xlabel('Time in years')
    axes.set_ylabel('Amount of fish in tons')
    matplotlib.pyplot.show()

plot_me()
```

Fig. 1. Code for Logistic Growth with Harvesting Simulation

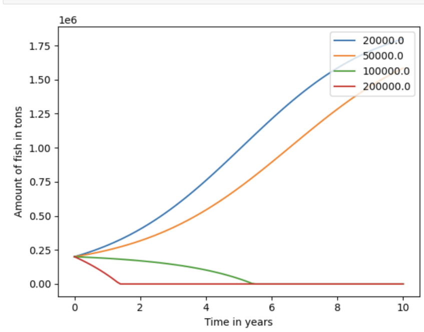

Fig. 2. Code for Visualization of Results

Visualization:
Several graphs were created to better understand the population dynamics under different harvesting rates:

- **Final Population vs. Harvest Rate:**
  A bar chart was created to illustrate how the final population size changes with increasing harvesting rates. The graph highlights an inverse relationship, where higher harvesting rates result in lower population sizes, and extinction occurs at unsustainable rates.

```
In [14]: # Harvest Rate vs. Final Population
final_populations = [fish[-1] for _, fish, _ in data]

matplotlib.pyplot.figure()
matplotlib.pyplot.bar([str(rate) for rate in harvest_rates], final_populations, color='skyblue')
matplotlib.pyplot.xlabel('Harvest Rate (tons/year)')
matplotlib.pyplot.ylabel('Final Fish Population (tons)')
matplotlib.pyplot.title('Final Population vs. Harvest Rate')
matplotlib.pyplot.grid(axis='y')
matplotlib.pyplot.show()
```

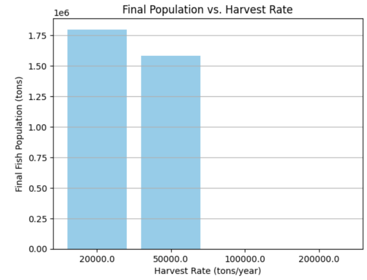

Fig. 3. Final Population vs. Harvest Rate

- **Logarithmic Fish Population Dynamics:**
  A line graph with a logarithmic scale was used to visualize population changes over time for each harvesting rate. This graph emphasizes the rapid decline in population near extinction and provides a clearer view of smaller population sizes.

```
In [15]: # Logarithmic Fish Population Dynamics
matplotlib.pyplot.figure()
for times, fish, rate_label in data:
    matplotlib.pyplot.plot(times, fish, label=f'Harvest Rate: {rate_label}')
matplotlib.pyplot.yscale('log')
matplotlib.pyplot.xlabel('Time in years')
matplotlib.pyplot.ylabel('Fish Population (log scale)')
matplotlib.pyplot.legend(loc='lower left')
matplotlib.pyplot.title('Log-Scaled Population Dynamics')
matplotlib.pyplot.grid()
matplotlib.pyplot.show()
```

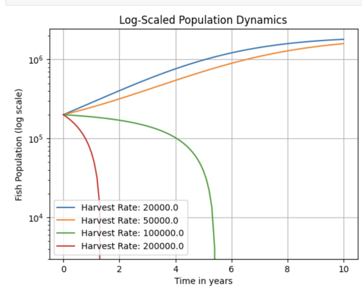

Fig. 4. Logarithmic Fish Population Dynamics

- **Phase Space-like Plot: Harvest Rate vs. Fish Population:**
  A phase-space-like plot was generated to observe how the fish population evolves over time for each harvesting rate. This plot shows the relationship between harvesting rate and population dynamics, offering insights into the system's behavior at different rates.

```
In [19]: import matplotlib.pyplot as plt

# Reusing data from the logistic growth problem
# Assuming `data` contains results for logistic growth as defined in the original code
# Extract a phase space-like visualization using fish population and harvest rates

# Phase space-like plot: Population vs. Harvest Rate
plt.figure()

# Assuming the last recorded population corresponds to the state to plot against harvest rate
for _, fish, rate_label in data:
    plt.plot([float(rate_label)] * len(fish), fish, 'g-', alpha=0.7, label=f'Rate {rate_label}')

plt.xlabel("Harvest Rate (tons/year)")
plt.ylabel("Fish Population (tons)")
plt.title("Phase Space-like Plot: Harvest Rate vs. Fish Population")
plt.grid(True)
plt.legend()
plt.show()
```

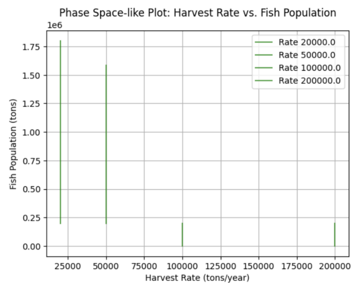

Fig. 5. Phase Space-like Plot: Harvest Rate vs. Fish Population

Conclusion: This experiment demonstrates the importance of maintaining sustainable harvesting rates to preserve fish populations and prevent overexploitation. The results highlight that when harvesting is balanced with the natural growth rate of the population, long-term stability can be achieved. However, higher harvesting rates lead to rapid population decline and eventual extinction, emphasizing the need for careful resource management. By understanding the relationship between harvesting and population dynamics, effective strategies can be developed to ensure ecological and economic sustainability.

B. Experiment 2: Predator-Prey Dynamics (Lotka-Volterra Model)

Objective:
To analyze the interaction between predator and prey species in an ecosystem using the Lotka-Volterra model. The aim is to understand how the populations of predators and their prey depend on each other, resulting in natural cycles of growth and decline.

Methodology:

1) **Lotka-Volterra Equations:**
The predator-prey interaction is modeled using the following equations:

$$\frac{du}{dt} = a \cdot u - b \cdot u \cdot v, \quad \frac{dv}{dt} = -c \cdot v + d \cdot b \cdot u \cdot v$$

Where:

- u: Population of prey (e.g., rabbits).
- v: Population of predators (e.g., foxes).
- $a = 1$: Natural growth rate of prey in the absence of predators.
- $b = 0.1$: Predation rate (rate at which predators consume prey).
- $c = 1.5$: Natural death rate of predators in the absence of prey.
- $d = 0.75$: Rate at which predation contributes to predator reproduction.

2) **Numerical Solution:**
The equations were solved numerically using an ODE solver over a simulation period of 15 days. Initial population sizes for prey ($u_0 = 10$) and predators ($v_0 = 5$) were used.
3) **Code:**

```
In [20]: from numpy import *
from scipy import integrate
import pylab as p

# Define parameters defining the behavior of the population
# Case: fox is predator, rabbit is prey
a = 1.      # a is the natural growing rate of rabbits, when there's no fox
b = 0.1     # b is the natural dying rate of rabbits, due to predation
c = 1.5     # c is the natural dying rate of fox, when there's no rabbit
d = 0.75    # d is the factor describing how many caught rabbits let create a new fox

def dX_dt(X, t=0):
    """Return the growth rate of predator and prey population"""
    return array([a*X[0] - b*X[1]*X[0],
                  -c*X[1] + d*b*X[1]*X[0]])

t = linspace(0, 15, 1000)   # time until 1000 days, with 15 days interval
X0 = array([10, 5])         # initial population (rabbit, fox)
X, infodict = integrate.odeint(dX_dt, X0, t, full_output=True)
infodict['message']         # integration successful

rabbits, foxes = X.T
f1 = p.figure()
p.plot(t, rabbits, 'r-', label='Rabbits')
p.plot(t, foxes, 'b-', label='Foxes')
p.grid()
p.legend(loc='best')
p.xlabel('time')
p.ylabel('population')
p.title('Evolution of foxes and rabbits population')
f1.savefig('rabbits_foxes.png')
```

Fig. 6. Code for Lotka-Volterra Model Implementation (Part 1)

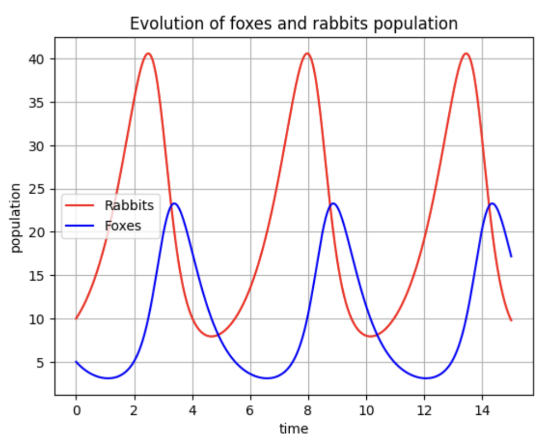

Fig. 7. Code for Lotka-Volterra Model Implementation (Part 2)

4) **Visualization:**
Several graphs were generated to analyze predator-prey dynamics:

- **Population Dynamics Over Time:**
A time-series plot shows the oscillatory behavior of prey and predator populations over the simulation period.

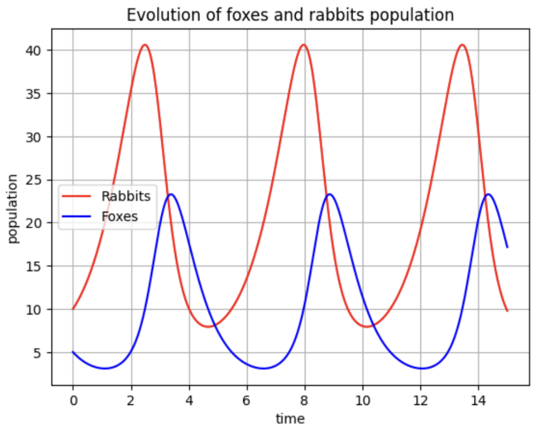

Fig. 8. Population Dynamics of Rabbits and Foxes Over Time

- **Phase-Space Plot:**
A phase-space plot visualizes the predator-prey relationship by plotting the predator population against the prey population, showing their cyclic interactions.

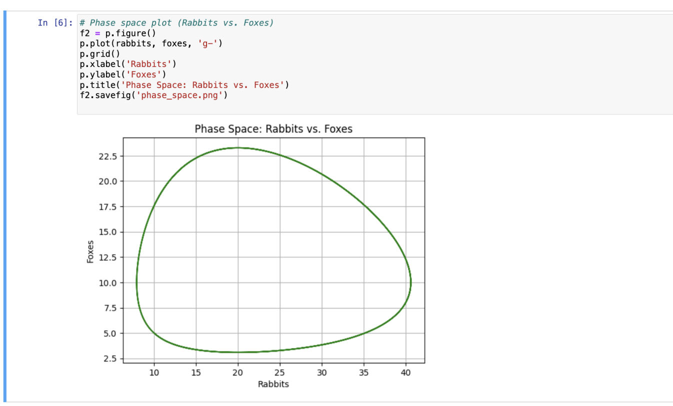

Fig. 9. Phase-Space Plot: Rabbits vs. Foxes

- **Predator-to-Prey Ratio Over Time:**
A plot of the predator-to-prey ratio reveals the balance and fluctuations between the two populations, emphasizing their interdependence.

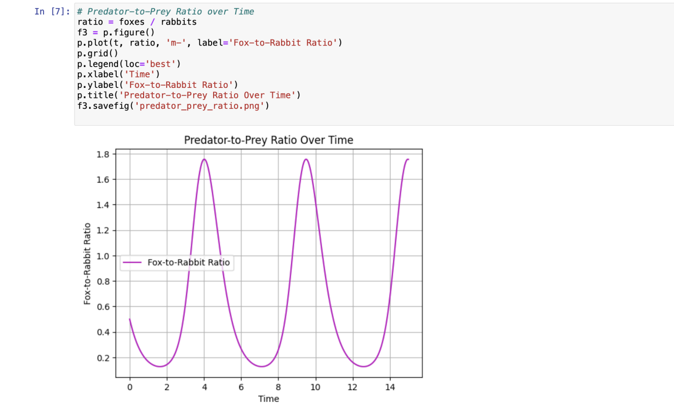

Fig. 10. Predator-to-Prey Ratio Over Time

Conclusion:
The Lotka-Volterra model effectively captures the dynamics of predator-prey interactions, showcasing their natural cycles of growth and decline. The prey population grows when predators are fewer, but declines as predators increase due to predation. Conversely, predator populations rise when prey is abundant and fall when prey becomes scarce. The phase-space plot highlights the cyclic relationship between the two populations, reflecting their interdependence and balance in the ecosystem. Additionally, the predator-to-prey ratio plot emphasizes the dynamic equilibrium and periodic fluctuations, underscoring the importance of ecological balance in sustaining both species.

C. Experiment 3: Solving Differential Equations Using Physics-Informed Neural Networks (PINNs)

Objective:
To solve the differential equation $y' + 2xy = 0$ using a Physics-Informed Neural Network (PINN). The goal is to demonstrate the capability of neural networks in solving differential equations by enforcing physical laws through loss minimization.

Methodology:

1) **Neural Network Architecture:**
The PINN was designed with two hidden layers, each containing 20 neurons and using the `tanh` activation function. The output layer directly predicts $y(t)$ without any activation function.
2) **ODE Residual:**
The residual of the differential equation was defined as:
$$R(t) = \frac{dy}{dt} + 2 \cdot t \cdot y$$
where y is the predicted solution of the PINN and $\frac{dy}{dt}$ is computed using TensorFlow's automatic differentiation.
3) **Training Data:**
Time points $t \in [0, 2]$ were sampled into 100 evenly spaced values. The initial condition $y(0) = 1$ was incorporated into the loss function.
4) **Loss Function:**
The total loss comprised:

- Residual loss: Ensuring that the PINN satisfies the differential equation $y' + 2xy = 0$.
- Initial condition loss: Enforcing $y(0) = 1$.

5) **Training Process:**
The model was trained using the Adam optimizer with a learning rate of 0.001 for 10,000 epochs. Loss values were monitored during training to ensure convergence.

Results and Discussion:
The trained PINN successfully predicted the solution of the differential equation, closely matching the analytical solution $y = e^{-x^2}$. The plot below compares the predicted solution with the analytical one, demonstrating the accuracy of the PINN. The loss values steadily decreased during training, indicating successful minimization of the residual and initial condition losses.

```
Epoch 0, Loss: 1.0578736066818237
Epoch 1000, Loss: 0.0004737475246656686
Epoch 2000, Loss: 0.00013604428386315703
Epoch 3000, Loss: 7.690447819186375e-05
Epoch 4000, Loss: 7.671386265428737e-05
Epoch 5000, Loss: 2.849387783498969e-05
Epoch 6000, Loss: 1.3016540833632462e-05
Epoch 7000, Loss: 5.1472916311468e-06
Epoch 8000, Loss: 3.1430661238118773e-06
Epoch 9000, Loss: 2.673888729987084e-06
```

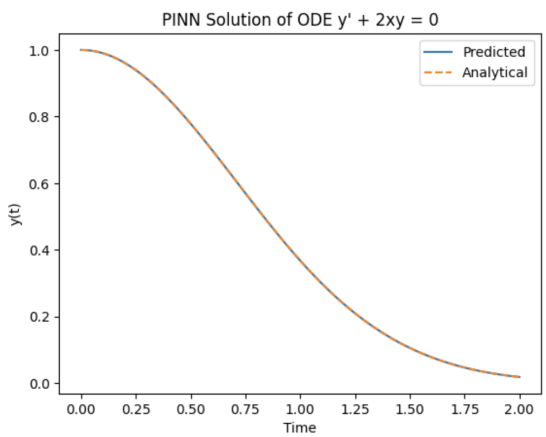

Fig. 11. PINN Solution of ODE $y' + 2xy = 0$. The predicted solution (solid line) aligns closely with the analytical solution (dashed line).

Conclusion:
This experiment highlights the ability of Physics-Informed Neural Networks (PINNs) to solve differential equations with high accuracy. By combining neural networks with physical constraints, PINNs provide a flexible and efficient approach for solving ODEs, especially in scenarios where traditional numerical methods might be limited.

IV. OVER ALL CONCLUSION

This report explored the use of mathematical modeling and modern computational techniques to understand and solve complex population dynamics and differential equations. Three experiments were conducted, each demonstrating unique insights and approaches to addressing ecological and mathematical challenges.

In the first experiment, the logistic growth model with harvesting illustrated the delicate balance required for sustainable resource management. It emphasized that overharvesting leads to rapid population decline and extinction, whereas controlled harvesting allows populations to stabilize near their natural carrying capacity.

The second experiment used the Lotka-Volterra model to analyze predator-prey dynamics. The oscillatory behavior of predators and prey revealed the interdependence between species, where population cycles are driven by predation and resource availability. This experiment highlighted the importance of ecological balance and the cyclic nature of interactions within ecosystems.

The final experiment employed a Physics-Informed Neural Network (PINN) to solve the differential equation $y' + 2xy = 0$. The PINN successfully predicted the solution with high accuracy, closely matching the analytical solution. This demonstrated the capability of PINNs to solve complex mathematical problems by enforcing physical laws, showcasing their flexibility and potential as an alternative to traditional numerical methods.

Overall, this report demonstrated the power of combining traditional models with modern computational tools to address real-world problems. From resource management to solving differential equations, these approaches provide valuable insights and pave the way for future advancements in ecological modeling, mathematics, and computational sciences.