COM3001 Assignment

Aryan Golbaghi
Department of Computer Science, University of Sheffield
Sheffield, England
agolbaghi1@sheffield.ac.uk

April 10, 2025

Chapter 1

Question 1: Dynamical systems

Chapter 2

Question 2: Prey-Predator System

2.1 Simulate the prey-predator model

The result of the prey-predator simulation can be found at Figures (2.10, 2.11), here is a description of the simulation:

In the simulation, we can see an oscillatory behavior of the populations, which is a characteristic feature of predator-prey dynamics.

- Grass (black curve) grows, fueling an increase in Rabbits (orange).
- As rabbits grow numbers, Foxes (blue) find abundant prey, so fox numbers grow.
- Increasing the number of rabbits will cause to decrease the amount of grass (this is seen in Figure 2.11 too).
- Once foxes grow in numbers and the grass amount decreases because of the population of rabbits, the population of rabbits will decrease.
- As the number of rabbits decreases, the foxes will have less food, leading to a decrease in their population.
- as there is less rabbits, the grass will grow again, and the cycle will repeat.

The intreating findings I can see from this simulation is even though the agent based model includes randomized movement and local interactions, the overall population still exhibits an oscillatory up and down cycles, similar to what we see in standard predator prey models (Lotka Volterra curves which mentioned in the lectures).

2.2 Differences between Agent Based Modeling vs Equation Based Modeling

In a classical equation based (ODE) predator prey model, such as Lotka Volterra equations:

$$\begin{cases} \dfrac{dx}{dt} = x(\alpha - \beta y) \\ \dfrac{dy}{dt} = y(\delta x - \gamma) \end{cases} \qquad \begin{array}{ll} \alpha > 0, & \text{Reproduction prey} \\ \beta > 0, & \text{Predation} \\ \gamma > 0, & \text{Extinction predator} \\ \delta > 0, & \text{Reproduction predator} \end{array}$$

populations are treated as continuous, homogeneous averages. By contrast, agent based models (ABMs) track individual rabbits and foxes. I have summarized the key differences between these two models in the following (see this table 2.3):

- **Spatial Heterogeneity**
  - **ABM:** Rabbits and foxes move around, so local depletion of grass or clustering of rabbits affects local outcomes. Some regions can be overgrazed or overhunted while others remain safe havens, and these local effects alter the overall population dynamics.

  - **ODE:** Assumes well mixed populations with no spatial structure. Grass, rabbits, and foxes interact uniformly in a single compartment.

- **Discrete and Stochastic Interactions**
  - **ABM:** Each eating or breeding event is discrete. Foxes may or may not successfully catch a rabbit (random probability), and rabbits may or may not find enough grass. This random element leads to fluctuations around the average trend.
  - **ODE:** Uses deterministic rate equations; there is no randomness in who gets eaten or how far you can move.

- **Individual Traits and Thresholds**
  - **ABM:** Each agent has individual properties such as an internal food store, an age, and breeding frequency. Once a rabbit's internal food is depleted, it dies—even if the average rabbit might have enough food.
  - **ODE:** Tracks the average rate of consumption and reproduction. There is no notion of an individual's internal energy or local shortage.

- **Emergent Behavior and Local Extinction**
  - **ABM:** Because movement is localized, pockets of rabbits or foxes can go extinct locally while others flourish elsewhere. Over time, random events can create drastically different patterns across multiple simulations.
  - **ODE:** Typically yields a single smooth trajectory for each initial condition—no chance of a local "pocket" dying off on its own.

- **Complexity vs. Analytical Tractability**
  - **ABM:** More realistic in capturing how individuals actually move, eat, and breed, but more complex and computationally intensive. Hard to get a neat "closed form" solution.
  - **ODE:** Mathematically elegant with known analysis techniques (e.g., stability analysis, limit cycles). Faster to run and simpler to interpret but less nuanced spatially and individually.

2.2.1 Why Do Results Differ Across Runs?

- **Random Initial Conditions**
  - Each run places rabbits and foxes at random locations in the grid.
  - Grass regeneration also occurs in randomly chosen cells.

- **Probabilistic Movement and Interactions**
  - Rabbits choose random directions if no grass is found in sight.
  - Foxes move randomly and only sometimes succeed in catching a rabbit (based on a probability that depends on the distance).
  - Each new rabbit or fox gets a random starting age, This affects how soon they die or reproduce.

2.2.2 Simulating the ABM Model to Monitor the Change of Key Variables

Here I have considered the following key variables, recorded across 100 simulation runs of the ABM. This information helps us evaluate the sustainability of an environment by revealing population dynamics and resource availability over time. Analyzing these trends can support better ecological understanding and inform strategies to prevent long-term or permanent environmental damage (please see Table 2.3 for the summary statistics of these variables and Figure 2.12 for their distributions).

- **Final Grass:** The total amount of grass available in the environment at the end of the full run of each simulation.
- **Final Rabbits:** The number of rabbits alive at the end of the full run of each simulation. This indicates if the rabbit population sustains itself or collapses over time.
- **Final Foxes:** The number of foxes alive at the end of the full run of each simulation. This helps determine whether the predator population can persist or dies out due to lack of prey.
- **Peak Grass:** The maximum amount of grass observed at any point during a simulation run.
- **Peak Rabbits:** The highest number of rabbits observed during a simulation run. This reflects the peak of reproduction and food availability.
- **Peak Foxes:** The maximum number of foxes observed during a simulation run. It shows the extent the predator population grows, in response to abundant prey.
- **Min Grass:** The lowest amount of grass recorded during the simulation. This indicates how depleted the resource can become due to overpopulation of rabbits.
- **Min Rabbits:** The minimum number of rabbits observed at any point. A value of zero suggests a population collapse or local extinction event.
- **Min Foxes:** The lowest number of foxes observed during a simulation. A minimum of zero suggests predator extinction, due to insufficient rabbit populations.

In Table 2.3, we observe that the system generally maintains ecological stability over the course of the simulations. The minimum amount of grass never reaches zero, which is a critical factor in ensuring the survival of the rabbit population, the rabbit population maintains a positive final count across all runs, showing that they are capable of reproducing and sustaining themselves.

However, due to the stochastic nature of agent-based models, we observe that in some simulation runs (6 out of 100), the fox population drops to zero. Despite this, the presence of surviving foxes in most simulations suggests the system can support predator-prey dynamics.

2.3 Parameter Sensitivity Analysis

In the previous exercise, we observed that the fox population went extinct in 6 out of 100 simulation runs. In this section, I aim to identify the conditions under which the fox population can be sustained more consistently. To do this, I selected a key parameter to vary and recorded the system's behavior in response.

The primary cause of fox extinction appears to be a shortage of rabbits, which in turn is often caused by a lack of grass, the main food source for rabbits. Therefore, I decided to vary the `growrate` parameter of the environment, which controls how many new grass units are added to the grid in each iteration. By increasing the grow rate, I aim to assess whether more abundant grass leads to larger rabbit populations, and subsequently, to better survival conditions for foxes.

List of Figures

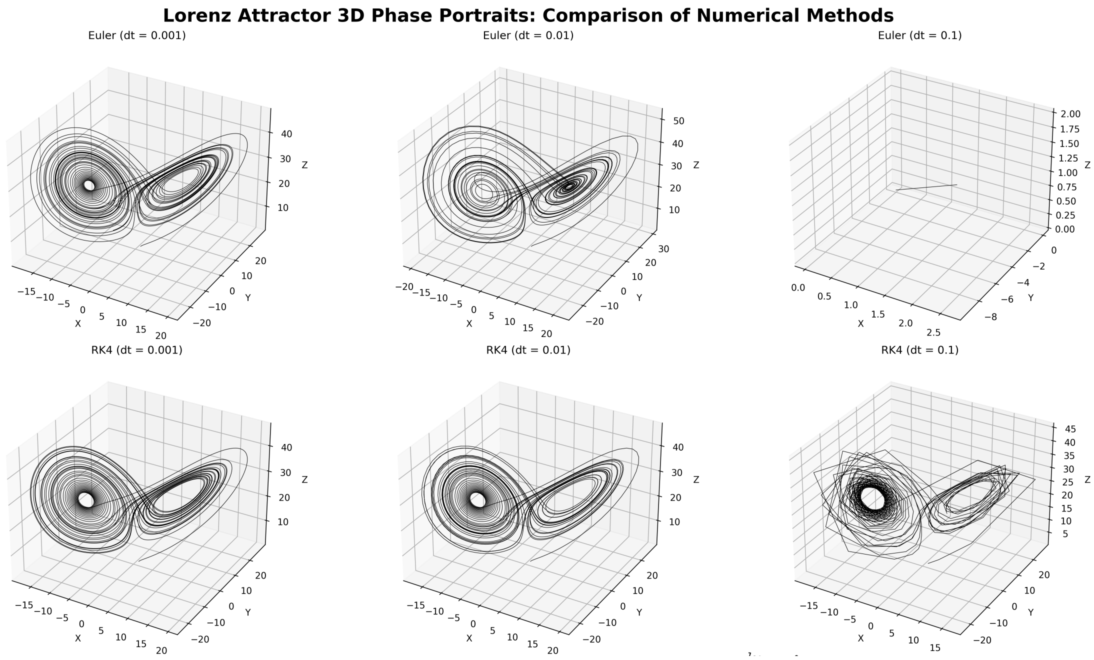

Figure 2.1: Lorenz attractor with initial state (1.0,1.0,1.0) and time step (0.001, 0.01, 0.1), and total simulation time of 40 seconds, For $\Delta t > 0.01$, the solution is not stable enough to reliably reproduce the attractor shape[7]

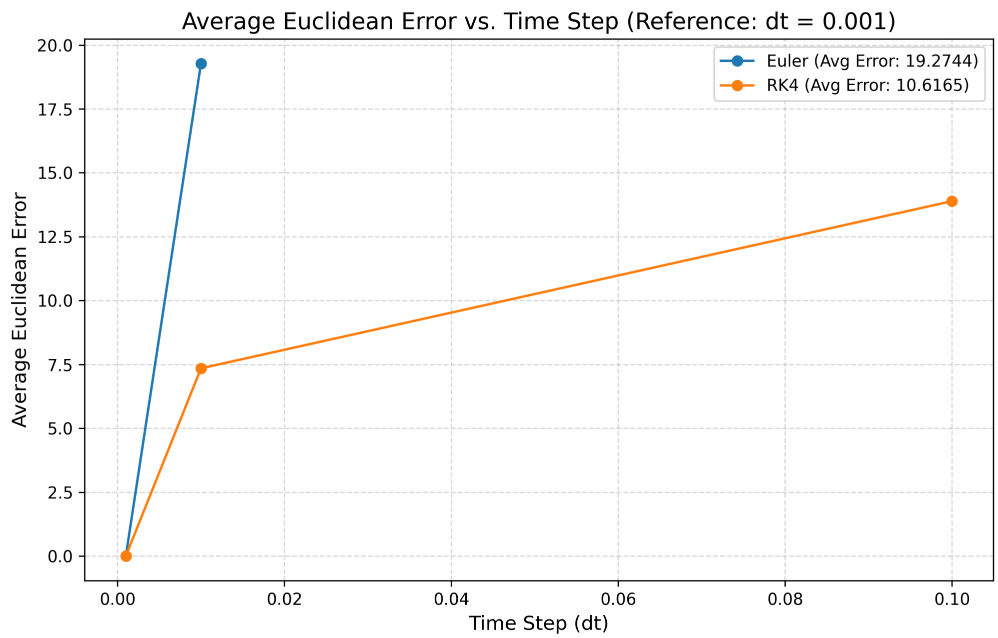

Figure 2.2: Lorenz attractor error analysis with initial state (1.0,1.0,1.0) and time step (0.001, 0.01, 0.1), and total simulation time of 40 seconds, For $\Delta t > 0.01$, the solution is not stable enough to reliably reproduce the attractor shape[7]

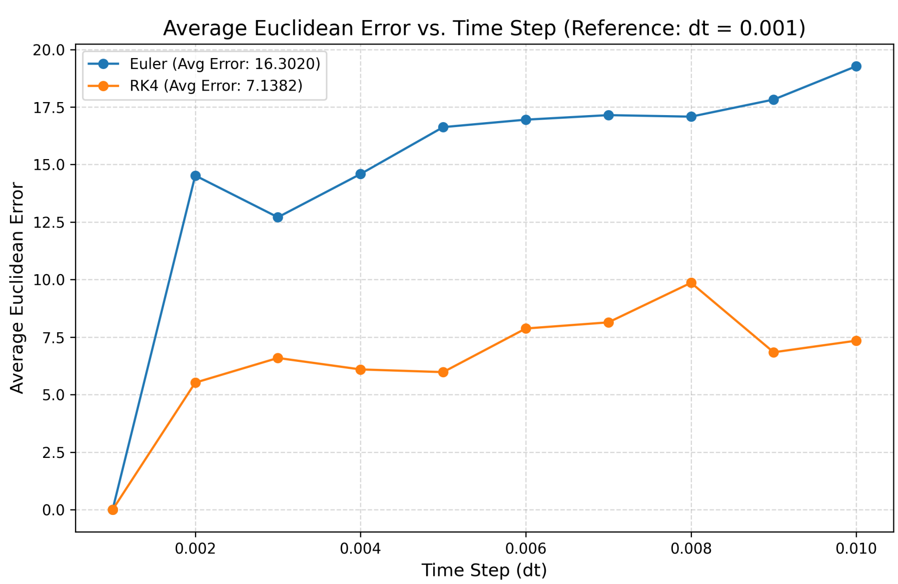

Figure 2.3: Lorenz attractor error analysis with initial state $(1.0, 1.0, 1.0)$, 1000 different time steps between $(0.0001 and 0.001)$, and a total simulation time of 40 seconds. this experiment is to compare the mean error of the two methods[8].

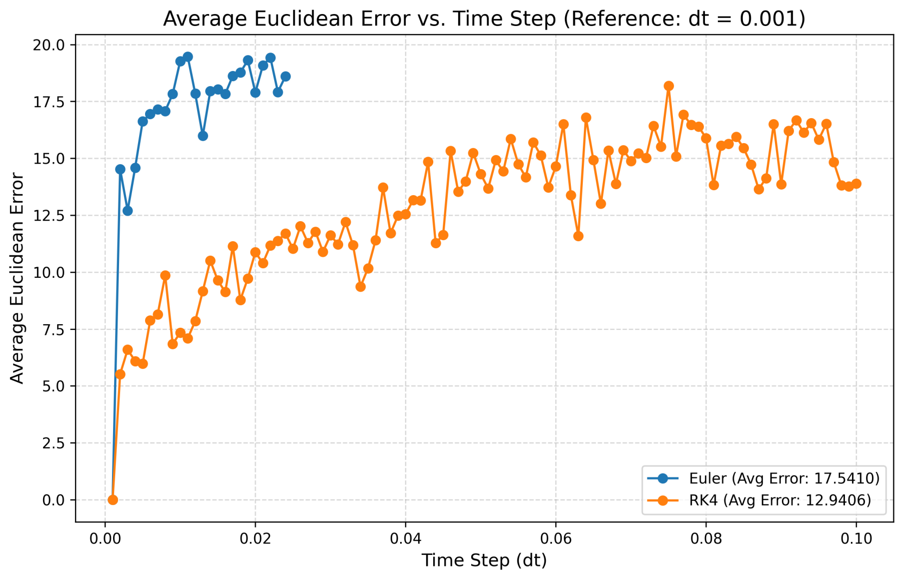

Figure 2.4: Lorenz attractor error analysis with initial state $(1.0, 1.0, 1.0)$, using 1000 different time steps between 10^{-3} and 10^{-1}, and a total simulation time of 40 seconds. For $\Delta t > 0.025$, the Euler solution is not stable enough to reliably reproduce the attractor shape. [4].

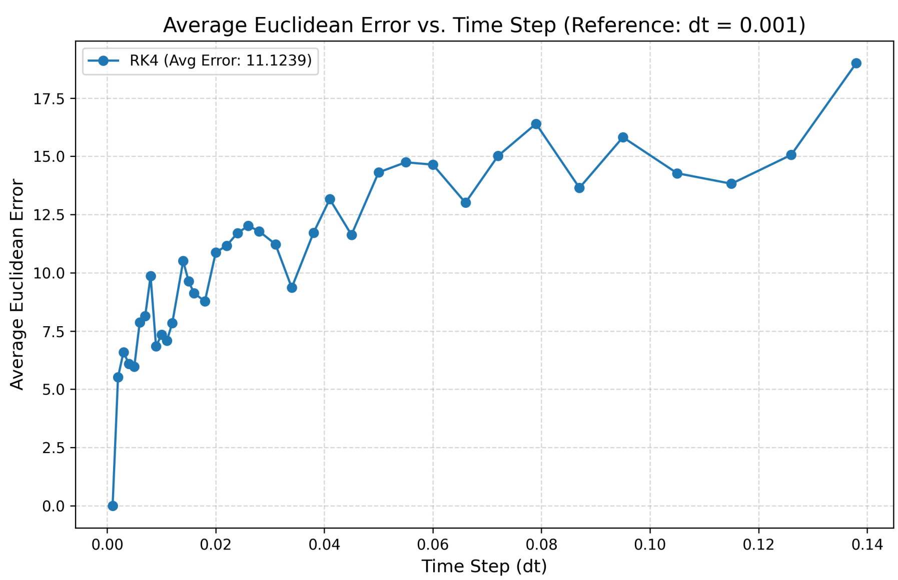

Figure 2.5: Lorenz attractor error analysis with initial state $(1.0, 1.0, 1.0)$, using 100 different time steps between 10^{-3} and 10^{1}, and a total simulation time of 40 seconds. For $\Delta t < 0.14$, the RK4 solution is stable enough to reproduce the attractor shape. [3].

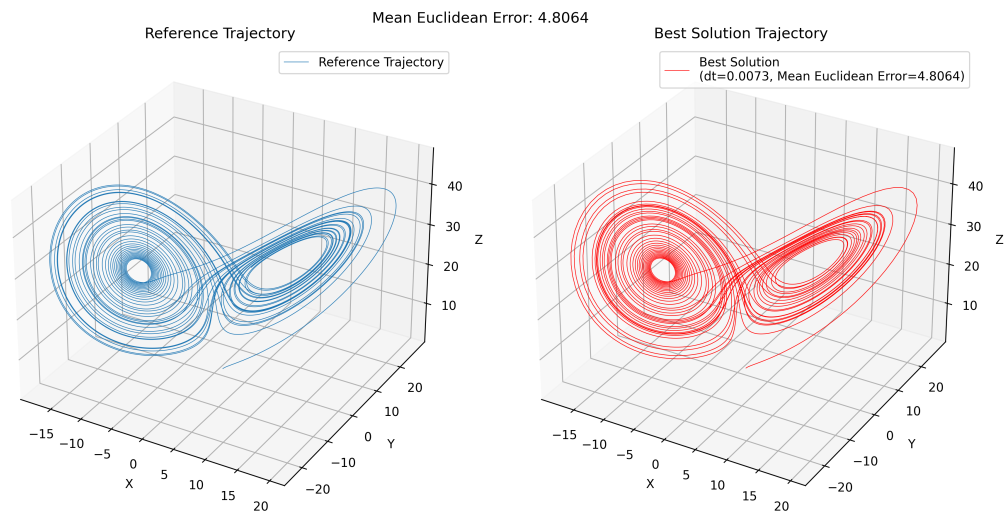

Figure 2.6: Comparison between reference system with $\Delta t = 0.001$ and a less computationally expensive system with step size of $\Delta t = 0.0073$, for this I used Optuna [1] with 50 trials. the code is at [6] and the image [9].

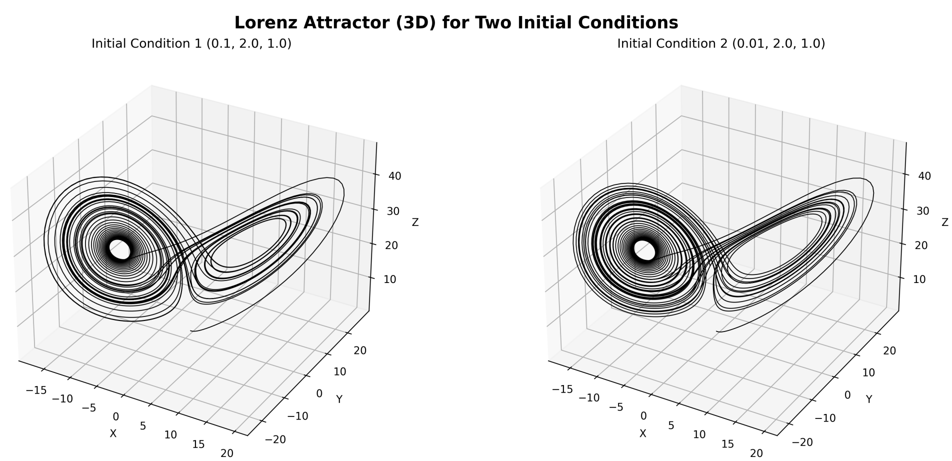

Figure 2.7: Comparison between two similar initial conditions of (0.01, 2.0, 1.0) and (0.1, 2.0, 1.0) and the time steps of $\Delta t = 0.0073$ to show the chaotic behavior of the system. the code is at [12] and the image [13].

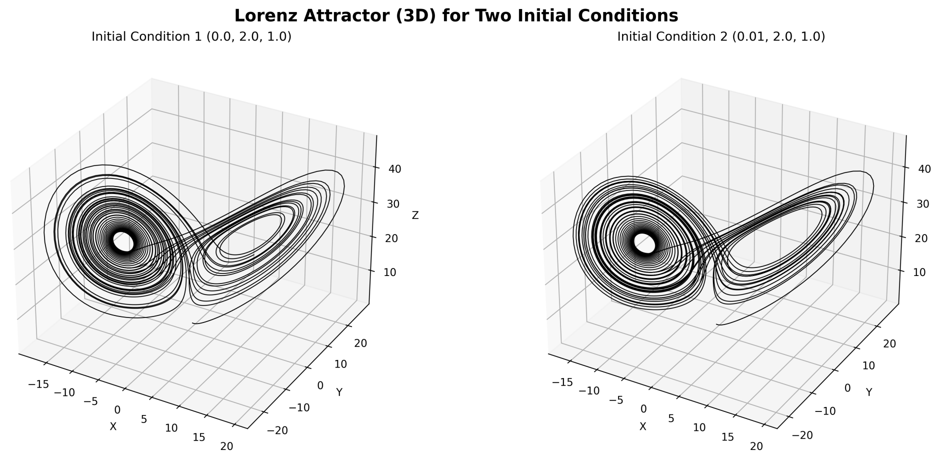

Figure 2.8: Comparison between two similar initial conditions of (0.0, 2.0, 1.0) and (0.1, 2.0, 1.0) and the time steps of $\Delta t = 0.0073$ to show the chaotic behavior of the system. the code is at [12] and the image [10].

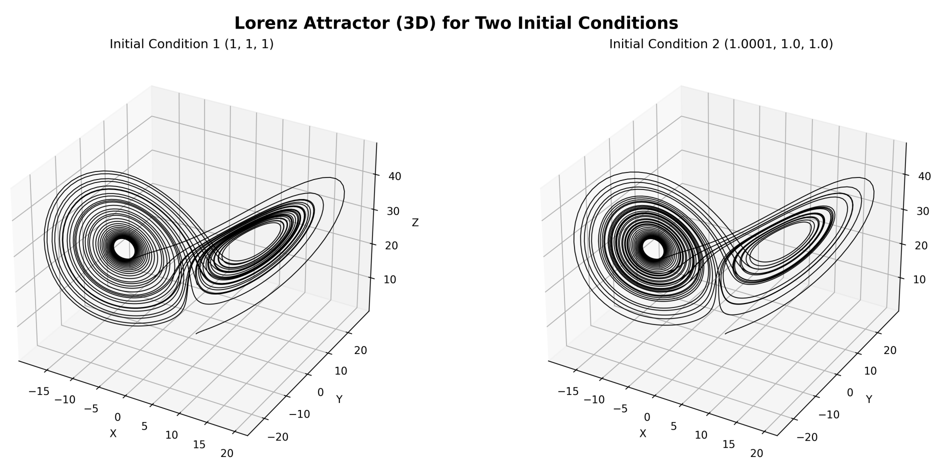

Figure 2.9: Comparison between two similar initial conditions of (1.0, 1.0, 1.0) and (1.0001, 1.0, 1.0) and the time steps of$\Delta t = 0.0073$ to show the chaotic behavior of the system. the code is at [12] and the image [11].

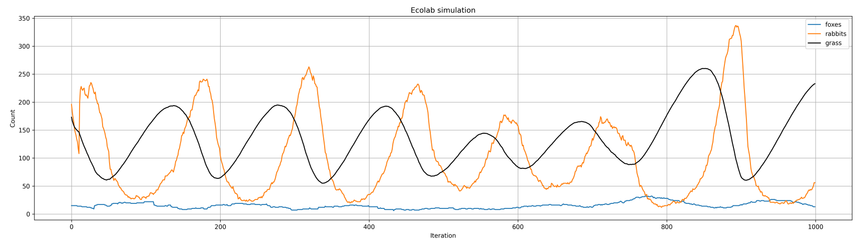

Figure 2.10: Simulation of a prey-predator system with the following initial settings: `Environment(shape=[60,60], growrate=60, maxgrass=50, startgrass=1)`, `Nrabbits = 200`, `Nfoxes = 15`. Rabbits are initialized with `speed=1`, `vision=5`, `breedfreq=10`, `breedfood=10`, and `maxage=40`. Foxes are initialized with `speed=3`, `vision=7`, `breedfreq=30`, `breedfood=20`, and `maxage=80`. See [12] for code and [11] for the source image.

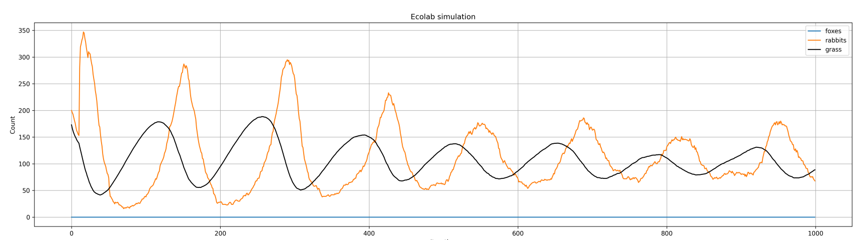

Figure 2.11: Simulation of a prey-predator system with the following initial settings: `Environment(shape=[60,60], growrate=60, maxgrass=50, startgrass=1)`, `Nrabbits = 200`, `Nfoxes = 15`. Rabbits are initialized with `speed=1`, `vision=5`, `breedfreq=10`, `breedfood=10`, and `maxage=40`. And ero foxes. See [12] for code and [11] for the source image.

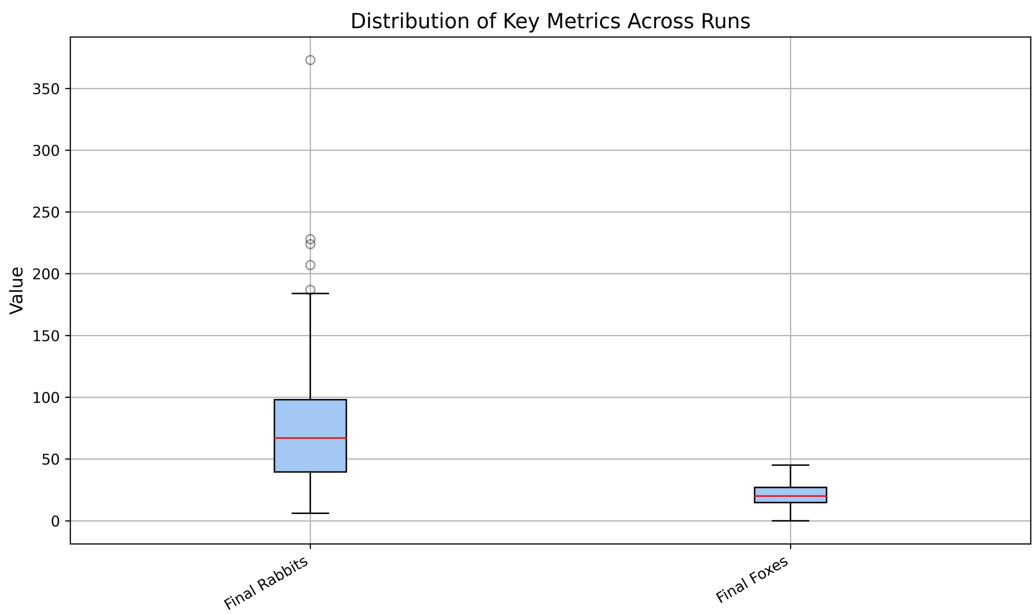

Figure 2.12: Descriptive statistics of key ecological variables recorded across 100 independent simulation runs of the Agent-Based Model (ABM), with each run spanning up to 1000 iterations. For more information see Table 2.3.

List of Tables

Method	**Global Error**	**Pros**	**Cons**
Explicit Euler	$O(\Delta t)$	Simple; low computational cost per step; easy to implement	Poor stability for larger Δt in chaotic regimes; error accumulates linearly
RK4	$O(\Delta t^4)$	Higher accuracy and stability; better error control over fixed step sizes	More computationally expensive per step; increased complexity compared to Euler
Adaptive Methods (e.g., RK45)	Varies with adaptive step-size	Dynamically adjusts Δt based on error; accurate and efficient in varying dynamics	More complex; computational overhead for step adjustment

Table 2.1: Comparison of Integration Methods

Feature	Agent-Based Modeling (ABM)	Equation-Based Modeling (EBM)
Modeling Units	Represents individual agents, such as each rabbit or fox, capturing their unique behaviors and interactions.	Models populations as continuous variables, focusing on aggregate properties rather than individual entities.
Stochasticity	Incorporates a high degree of randomness in agents' movements, births, deaths and so on.	Typically deterministic.
Spatiality	Explicitly models space, allowing agents to move and interact within an environment (in our case, a 2D environment).	Generally lacks spatial modeling.
Emergence	Enables emergent patterns from simple interaction rules among agents.	Patterns are outcomes of governing equations; less emphasis on emergence.
Complex Interactions	Models diverse and complex behaviors at the individual level.	Simplifies interactions by averaging behaviors across the population.
Realism	More biologically realistic, but computationally expensive.	Abstract, analytically tractable, and computationally less expensive, but may lack individual-level detail.

Table 2.2: Comparison of Agent-Based Modeling (ABM) and Equation-Based Modeling (EBM)

Variable	**Mean**	**Std Dev**	**Min**	**Max**
Peak Rabbits	317.00	44.60	249	433
Min Rabbits	13.57	5.48	3	27
Final Rabbits	88.03	68.33	12	257
Peak Foxes	34.07	7.32	18	51
Min Foxes	4.33	2.29	0	10
Final Foxes	18.60	9.70	0	36
Peak Grass	5055.33	702.21	4307	7490
Min Grass	1122.37	141.13	775	1324
Final Grass	2646.97	887.27	1614	4444

Table 2.3: Descriptive statistics of key ecological variables recorded across 100 independent simulation runs of the Agent-Based Model (ABM), with each run spanning up to 1000 iterations. The variables include the peak, minimum, and final values for the population of rabbits and foxes, as well as the amount of grass in the environment. These metrics provide insight into the dynamics of species interactions and resource availability, helping to evaluate ecosystem sustainability and identify scenarios that may lead to population collapse or resource depletion. See Figure 2.12 for the distribution of these metrics.

Bibliography

[1] Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. Optuna: A next-generation hyperparameter optimization framework. https://optuna.org/, 2019. Accessed: 2025-04-07.

[2] Armaos, V., Argiriou, A. A., and Kioutsioukis, I. Quantum computing and atmospheric dynamics: Exploring the lorenz system, 2024.

[3] Golbaghi, A. error$_c omparison_r k4_s table_0.13. https : //github.com/youngaryan/COM3001_A SSIGN/blob/main/1.1.4/images/error_c omparison_r k4_s table_0.13.$ $2025-04-01.$

[4] Golbaghi, A. Lorenz attractor error 1000 time steps. https://github.com/youngaryan/COM3001$_A SSIGN/blob/main/1.1.4/images/error_c omparison1000.png, 202$ $2025-04-01.$

[5] Golbaghi, A. Lorenz attractor visualization. https://github.com/youngaryan/COM3001$_A SSIGN/blob/main$ $2025-04-01.$

[6] Golbaghi, A. Lorenz attractor visualization. https://github.com/youngaryan/COM3001$_A SSIGN/blob/main$ $2025-04-01.$

[7] Golbaghi, A. Lorenz attractor visualization. https://github.com/youngaryan/COM3001$_A SSIGN/blob/main$ $2025-04-01.$

[8] Golbaghi, A. Lorenz attractor visualization error 3 steps. https://github.com/youngaryan/COM3001$_A SSIGN/blob/main/1.1.4/images/error_c omparison.png, 2025.Ac$ $2025-04-01.$

[9] Golbaghi, A. optunacomparisonimage.13. https://github.com/youngaryan/COM3001$_A SSIGN/blob/main/1$ $2025-04-01.$

[10] Golbaghi, A. youngaryantwo$_i nitial_c onditions_3 d_2 separateicode.13. https : //github.com/youngaryan/COM3001_A SSIGN/blob/main/1.1.4/images/two_i nitial_c onditions_3 d_s eparate_2.pn$ $2025-04-01.$

[11] Golbaghi, A. youngaryantwo$_i nitial_c onditions_3 d_3 separateicode.13. https : //github.com/youngaryan/COM3001_A SSIGN/blob/main/1.1.4/images/two_i nitial_c onditions_3 d_s eparate_3.pn$ $2025-04-01.$

[12] GOLBAGHI, A. youngaryantwo$_{i}initial_{c}onditions_{3}d_{s}eparateicode.13.\quad https\ :$ $//github.com/youngaryan/COM3001_{A}SSIGN/blob/main/1.2.1/two_{d}iff_{i}nitial_{c}ond.py, 2025. Accessed :$ $2025-04-01$.

[13] GOLBAGHI, A. youngaryantwo$_{i}initial_{c}onditions_{3}d_{s}eparateimg.\quad https\ :$ $//github.com/youngaryan/COM3001_{A}SSIGN/blob/main/1.1.4/images/two_{i}nitial_{c}onditions_{3}d_{s}eparate.png$ $2025-04-01$.

[14] KASHYAP, S. S., DANDEKAR, R. A., DANDEKAR, R., AND PANAT, S. Modeling chaotic lorenz ode system using scientific machine learning. https://arxiv.org/abs/2410.06452v1, 2024. arXiv:2410.06452v1 [cs.LG], 09 Oct 2024.

[15] LORENZ, E. N. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences 20*, 2 (Mar. 1963), 130–148.

[16] PATHAK, J., LU, Z., HUNT, B. R., GIRVAN, M., AND OTT, E. Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science 27*, 12 (Dec. 2017).

[17] SOCIETY, A. P. Lorenz and the butterfly effect, 2003. Accessed: 2025-04-01.