Università della Svizzera italiana

Particle Methods **2023**

Student: Filippo Casari

Report for Assignment 2

Introduction

To achieve the goal of the assignment I implemented 2 versions:

- **Python 2D**: In the beginning I used lists of objects and then I switched to dictionaries for better performances. Complexity: $O(n)$ using cell lists.
- **Python 3D**: Same as the previous version but in a 3D environment. Complexity: $O(n)$

I found the best parameters that can lead to similar results for Lotka-Volterra equations. I am going to list those parameters later.

Initial Conditions

The rabbits and the wolves are distributed randomly within the environment as shown below (example). I decided to not set fixed seed for random functions to have every time a different initial condition.

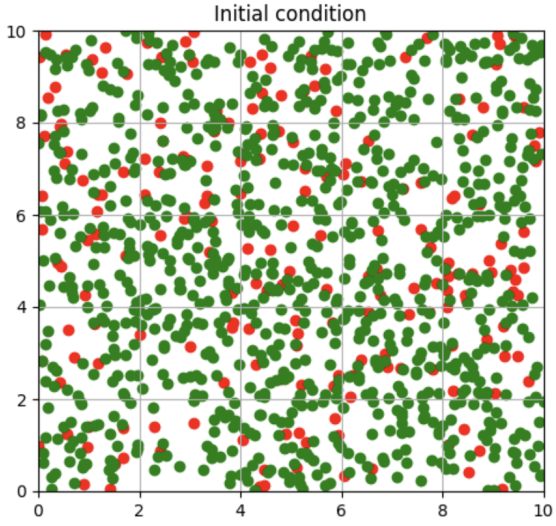

Figure 1: Initial conditions

Implementations

I wrote different json files which are used as input for both main.py and main3D.py scripts. Each of those file conteins the specific params described in the assignment.
4 json files are in the main folder, called:

- paramsA.json
- paramsB.json
- paramsC.json
- best_params.json

Point A

After very few hundred iterations and with the given parameters the wolves die. In contrast what I expected, the wolves death rate maybe is high probably because the rabbit birth rate is not high enough.
By running main.py I draw the following plots:

- **Simulation**
- **Number of rabbits and wolves over iterations**
- **Rabbits and wolves density**

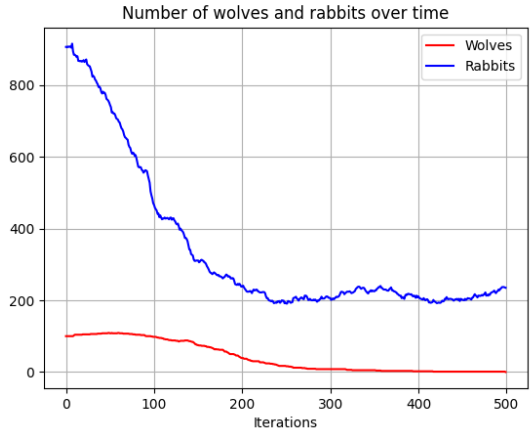

Figure 2: PointA;2D; Number of rabbits and wolves over iterations

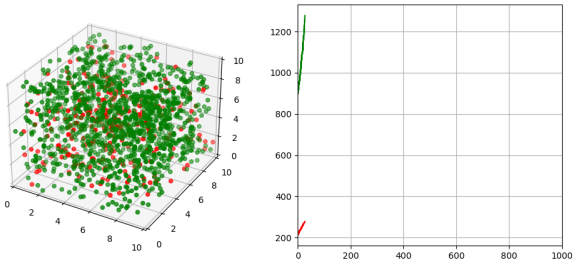

Figure 3: RT visualization

In the 3D plot the number of rabbits grows exponentially as well (8000 rabbits) in the very first iterations.

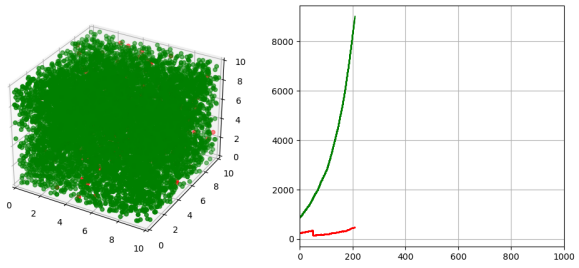

Figure 4: Environment visualization after just 200 iters

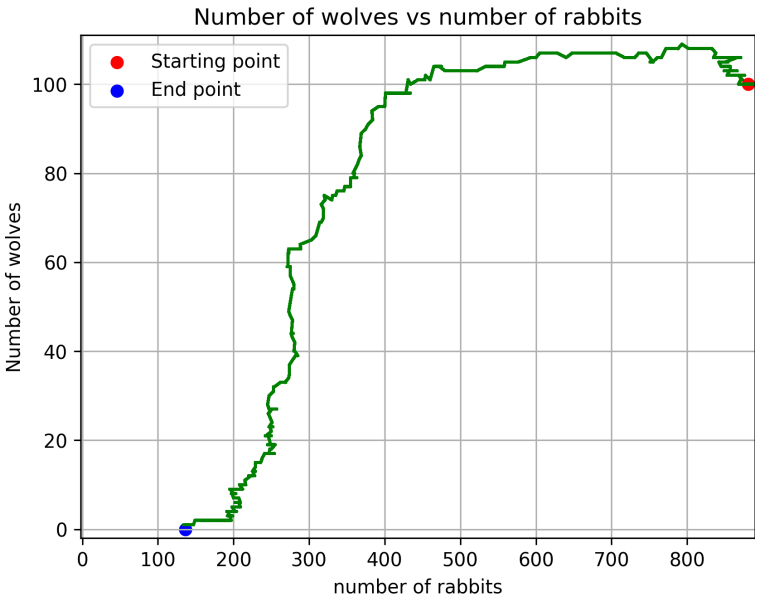

Figure 5: Population

Point B

In this point I set the $t_d^r = 50$. However, the situations is even worse than point A because the rabbits live less and no food is provided to the wolves. The number of rabbits and wolves converges very quickly to zero as shown below.

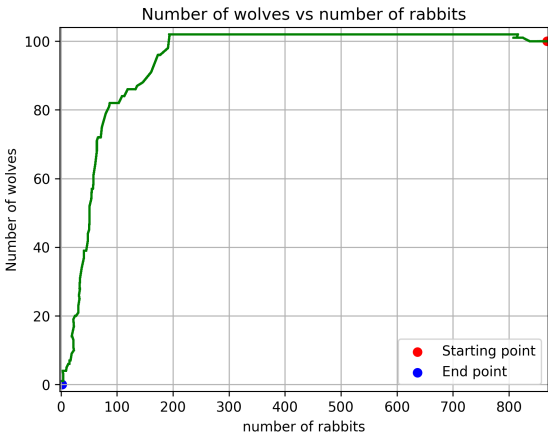

Figure 6: Population

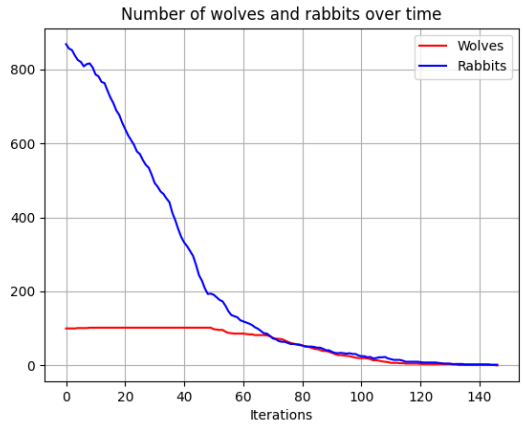

Figure 7: Population over iterations

Point C

In this point I set the $\sigma = 0.05$. Even with this different parameter the wolves cannot survive more than just 350 iterations probably because the rabbit birth rate is not sufficient. Moreover, the rabbits numbers oscillate every 100 iterations.

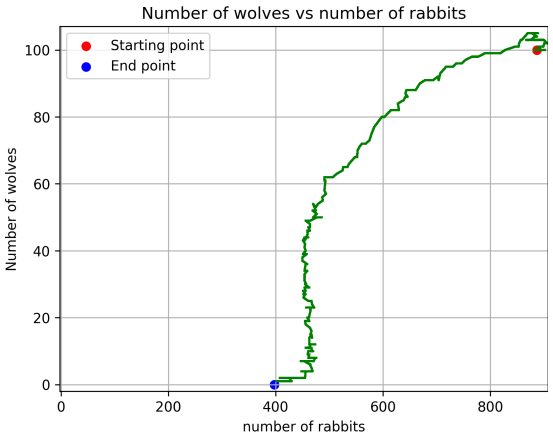

Figure 8: Population

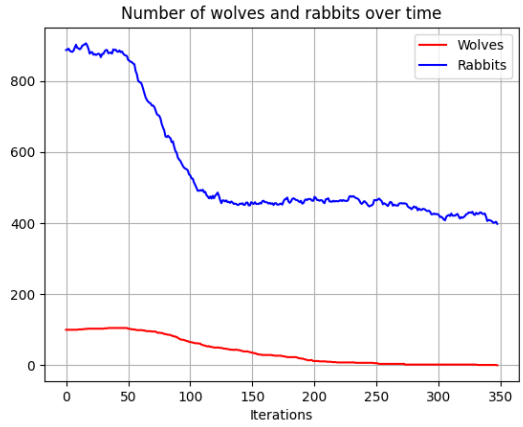

Figure 9: Population over iterations

My Best parameters

After many attempts I found optimal paramters for this model. I will list what I used. Note: this are the best only for 2D dimensions. I still have to find out which params are the best for the 3D space.

- $ITER$: 1000
- N_R": 900
- N_W: 200, I increase the initial number of wolves
- R_C: 0.5
- P_E_W: 0.03, I increase the probability of eating a rabbit
- P_R^W: 0.1, I increase the probability of reproduction rate

- P_R^R: 0.07 I increase the probability of reproduction rate
- T_D^R: 100
- T_D^W: 50
- μ: 0.0
- σ: 0.05, I used σ value of point C

I will show 2 runs of this simulation. One with 1000 and $L = 10$ and the other with 2000 iterations and $L = 8$.

First run, L=10, ITER=1000

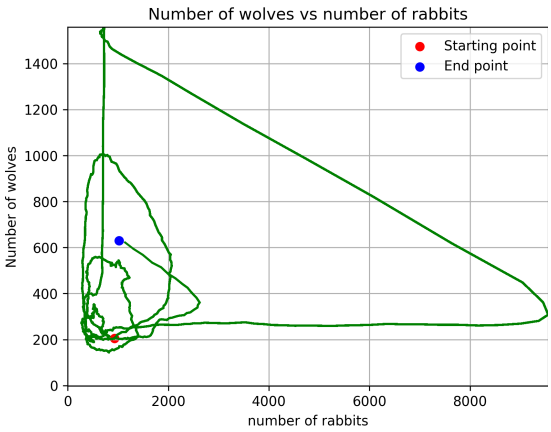

Figure 10: Population

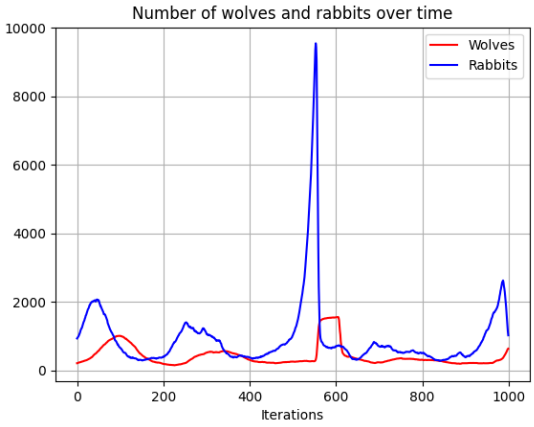

Figure 11: Population over iterations

First run, L=8, ITER=2000

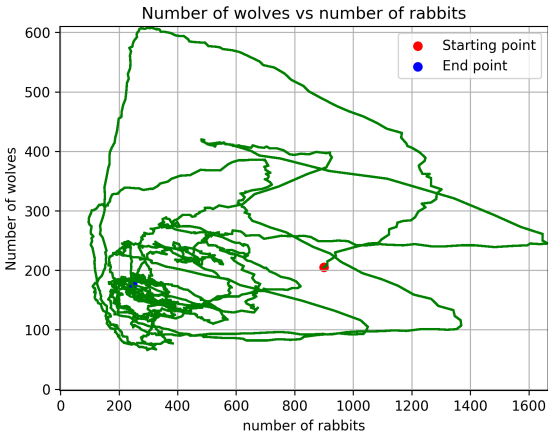

Figure 12: Population

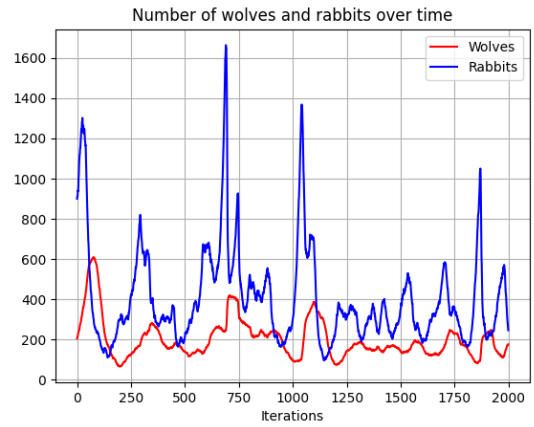

Figure 13: Population over iterations

Conclusion

As one can see these simulations perform much more better than previous points.
The plots reflect similarly the beheviour of Lotka–Volterra equations. The parameters used lead to a system equilibria. This is because:

$$y = \alpha/\beta \ \ and \ x = \gamma/\delta$$

where $y = \#wolves$, $x = \#rabbits$, and α, β, γ, δ are the parameters of the model.