VU Modeling and Simulation

Cellular Automaton vs. Difference Equation
Case Study: Predator Prey System

Autoren

Christoph Hämmerle

066926 - 01226577

Philipp Ganiu

033526 - 01051198

Jennifer Lumetzberger

066453 - 01206148

Supervised by

Jennifer Straub

Abstract

Using two modelling concepts the dynamics of predator and prey population in a forest is modelled. The objective is to see the correlation between the two species.

Contents

1 Introduction

A forester wants to get information about the maximum number and the dynamics of the population of deer in its forest. In order to observe the correlation between the number of red-foxes and deer, a predator-prey approach is modelled. Different modelling concepts are compared: a cellular automata model and a formulation given by a damped version of the classic ODE model by Lotka and Volterra.

2 Model

2.1 Differential Equation Model

First of all a difference equation model is derived (The derivation is very similar to the derivation of the classic differential-equation model). The following is observed (e.g. by external datasets or by opinion of the forester):

- During each month in the absence of a natural predator the population of deer increases by a certain rate $\alpha > 0$ which includes natural birth and death rates. As the forest has a certain resource capacity M the growth is limited leading to $\alpha(M-x)$ (respectively $\alpha(M-x-y)$ if there is a fox-species too)
- During each month in the absence of any deer the population of foxes decreases by a certain rate $\gamma > 0$ as the population is starving.
- In case of cohabitation of the two species we could observe that the number of deer each month is reduced by xyβ with a positive β animals.
- Benefting from fresh meat, the decreasing rate γ is balanced by a positive rate δx. In case $\delta x - \gamma > 0$ there is an increase of the number of foxes. Otherwise the number decreases.

Putting all those observations together leads to the following dynamics:

$$x_0(t) = x(t)\alpha \times (M - x(t) - y(t)) - x(t)y(t)\beta \tag{1}$$

$$y_0'(t) = y(t)(\delta x(t)\gamma) \tag{2}$$

2.2 Cellular Automata Model

Given the same information a cellular-automata is derived.

- We assume our forest to be approximately rectangular and apply a rectangular grid.
- At the beginning of the simulation each cell of the grid is either placed a prey, a predator or left empty. We furthermore sloppily say that a cell is either "prey", "predator" or "empty".
- The whole matrix is updated simultaneously each month by the following rules:
  - In case cell i,j is empty, we chose one neighboured cell at random and check if the cell is prey. In this case, the neighboured cell reproduces and cell i,j becomes prey as well with probability p_1.
  - In case cell i,j is prey, we chose one neighboured cell at random and check if the cell is predator. In this case, the neighboured cell eats up the prey and reproduces. Therefore cell i,j becomes predator as well with probability p_2.
  - In case cell i,j is predator, the predator unfortunately dies with a certain probability. Therefore cell i,j becomes empty with probability p_3.

By a neighboured cell we understand any cell which is not father away as k cells from the original cell, whereas k is a natural number.

3 Task description

Task 1
Implement the damped differential equation model and perform some trial runs. Try several self chosen parameters $0 < \alpha, \beta, \gamma, \delta$ for the simulation and fix $x_0 = 4000$ and $y_0 = 1000$ as initial populations. Notice that $M \gg x_0 + y_0$ and fix a self chosen natural number for it.

Task 2
Implement the cellular automaton (CA) model and perform some trial runs. Try some self chosen parameters $0 < p_1, p_2, p_3 < 1$ and fix neighbourhood parameter k = 1 at the first place. Use the same initial conditions for the CA as for the difference model and initially mark 4000 cells as prey and 1000 cells as predator - note that you need to use a rectangle with more than 4000 + 1000 cells (furthermore denoted as N) in order to perform simulations.

Task 3
Try to find out how to present the results of the CA in animated form and how you can aggregate the results of the CA in order to achieve comparable curves to the differential equation.

Task 4
Fix all other parameters and experiment with different rectangles with same number of cells. Experiment whether the shape of the rectangle influences the results.

Task 5
Try to find parameter-settings $(\alpha, \beta, \gamma, \delta, M)$ and (p_1, p_2, p_3, N, k) so that the corresponding resulting curves of both models look alike.

Task 6
Summarize your project work as a protocol.

4 Methods

The predator-prey approach is modelled with Matlab.

5 Implementation

5.1 Task 1

For the first task, the damped differential equation model is implemented in Matlab.
As initial values for the deer (x) and fox (y) population, we take $x_0 = 4000$ and $y_0 = 1000$, as given in the task. For the resource capacity of the wood M, we consider $M \gg x_0 + y_0$ and fix it with $M = 10000$.
The parameters α, β, γ and δ are also fixed with:

$$\alpha = 0.001$$
$$\beta = 0.003$$
$$\gamma = 2$$
$$\delta = 0.001$$

First of all, a matrix is created, which stores the month and the corresponding amount of deer and foxes, for every month from 0 (starting point) to the desired maximum time point:

month	**0**	1	2	...	m
deer	**4000**	?	?	...	?
foxes	**1000**	?	?	...	?

Table 1: Matrix scheme with information on amount of deer/foxes for each month

Via the given difference equations (1) and (2) the population of each time point can be calculated. After each iteration, the data is stored into the matrix.

5.2 Task 2

For implementing the cellular automata (CA) model, some self chosen parameters $0 < p_1, p_2, p_3 < 1$ are used. The neighbourhood parameter k is fixed with $k = 1$ at the first place. As with the difference model, the inital values are the same: $x_0 = 4000$ and $y_0 = 1000$.
So, the initial parameter setting is as following:

$$rows = 70$$
$$col = 200$$
$$p1 = 0.01$$
$$p2 = 0.8$$
$$p3 = 0.01$$

Now the first step is to set up a matrix with the 4000 deers and 1000 foxes randomly distributed in our rectangle (forest). The rectangle is built up by the number of rows and columns. Therefore in this case we have 14000 (70x200, universally valid: N) cells, that's why we create a row-vector *initvalues* with N entries. With the function **randperm** we pick 5000 $(x_0 + y_0)$ cells with a random permutation between 0 and N, so each value

appears only once. These random numbers we use as indices of *initvalues* and assign the first 4000 (x_0) cells to 1 (which stands for deer). The next 1000 (y_0) random numbers we assign to 2 (which means foxes). Since we initialized *initvalues* with zeros, the step of creating the initial population is done.
In order to create the rectangle, we take our *initvalues* vector and fill a new matrix *population* with its values, by using the **reshape** function.

For creating the states for each timestep, a for-loop is used, regarding each cell and applying the rules of the CA described at the beginning. (See Listing 1)

```
for j=1:col % go through all cells

                %if cell is empty{
                 if ca(i,j) == 0

                     neighbour = getRandomNeighbour(ca,k,i,j);

                    %check if neighbour is prey
                     if neighbour == 1

                         %if rand <= p1 empty cell is prey
                          if rand()<= p1
                              tmp(i,j) = 1;
                          end
                     end

                %if cell is prey
                 elseif ca(i,j) == 1
                  neighbour = getRandomNeighbour(ca,k,i,j);

                    %check if neighbour is predator
                     if neighbour == 2

                         %if rand <= p2 empty cell is predator
                          if rand()<= p2
                              tmp(i,j) = 2;
                          end
                     end

```

```
30                %if cell is predator
31                elseif ca(i,j) == 2
32
33                    %if rand <= p3 predator is dying
34                    if rand() <= p3
35                        tmp(i,j) = 0;
36                    end
37                end
38
39            end
```

Listing 1: Implementation of the CA rules applied

After each iteration we sum up all our foxes, deer and empty spaces, to have results which can be compared later with the values of the differential equation model.

5.3 Task 3

For achieving comparable results with the CA as with the differential equation, after each loop iteration, all cells belonging to the fox/deer/no population are summed up and stored in a matrix.

For animating the CA, a plot is created with the x-axis representing the length of our rectangle and the y-axis illustrating the width of the rectangle. Using the Matlab function **spy**, each cell of the rectangle/matrix can be marked in a different color. (See Listing 2)

```
1     hold on;
2         spy(ca==0, 'w'); % want to color my cells
3         spy(ca==2,'r');
4         spy(ca==1,'g');
5         %axis([]);
6         legend('empty','foxes','deer');
7         xlabel(['Months: ' num2str(m)])
8         hold off;
9         %drawnow
10        %Delaying animation
11        pause(0.001);
```

Listing 2: Implementation of CA animation

When updating the plot after each loop iteration (each time step), we have created a simple animation presenting the population development of our species.

5.4 Task 4

In Task 4 all parameters are fixed, except of the shape of the rectangle (number of cells stays the same). The results are shown in the results section.

5.5 Task 5

In Task 5 the parameters $\alpha, \beta, \gamma, \delta, M$ as well as p_1, p_2, p_3, N, k shall be changed so that the corresponding resulting curves of both models look alike. The results are shown in the results section.

5.6 Task 6

This task is obviously done.

6 Results

6.1 Task 1

When establishing the difference equation model with the parameters fixed in the execution section of Task1, we gain the following result:

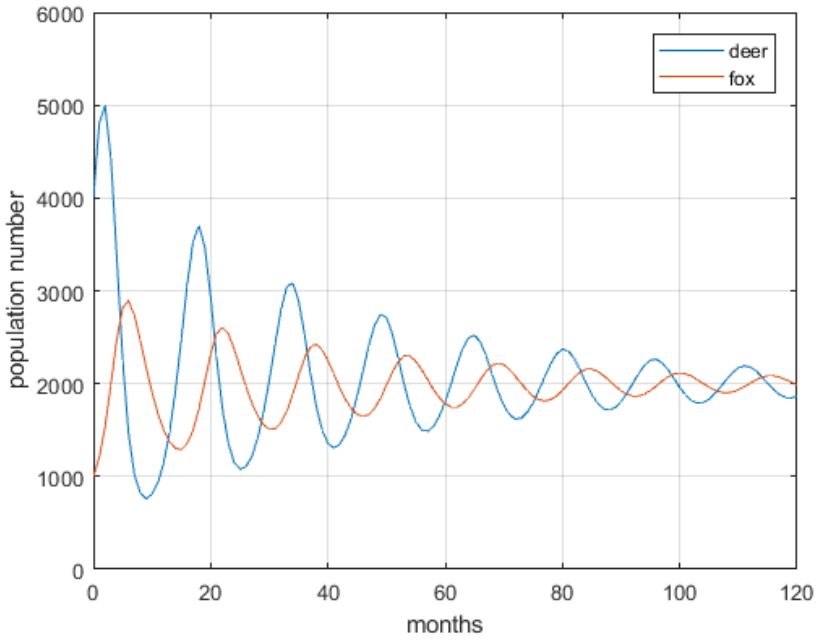

Figure 1: Population development of foxes and deer over 10 years

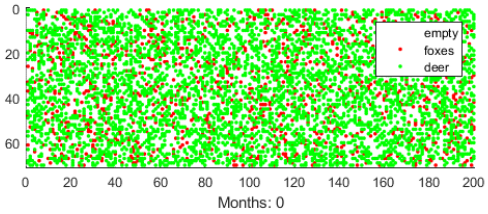

Figure 2: CA at initialization time

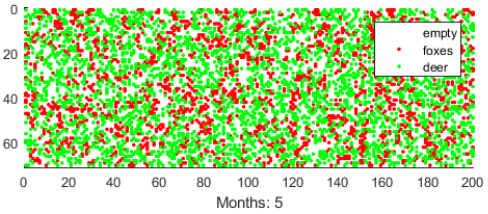

Figure 3: CA after 5 months

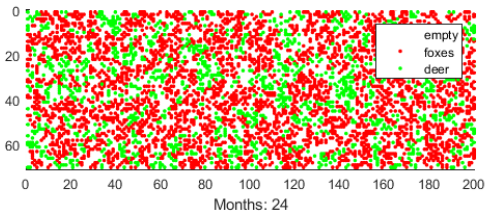

Figure 4: CA after 2 years

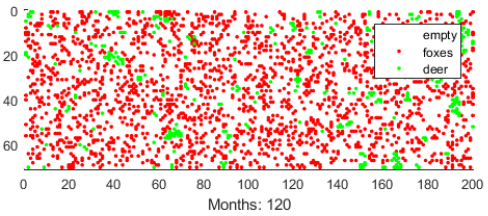

Figure 5: CA after 10 years

At the beginning, the deer population is 4 times the fox population. Due to the high amount of food for the foxes, the fox population rises, which leads to a fall in the deer population. After several years, an equilibrium is found, when each species comprises about 2000 individuals. In table 2 the population numbers of certain time points can be seen.

month	**0**	1	5	24	60	80	100	120
deer	**4000**	4800	2258	1154	1806	2370	1959	1864
foxes	**1000**	1200	2817	1375	1783	1927	2116	1984

Table 2: Difference Equation Model: Population development at certain time points

6.2 Task 2 and 3

In Task 2 and 3, the cellular automata model was created. We can compare the rectangle in the initial state with the results after 10 years. We also look at two other time steps, namely at $m_1 = 5$ months and $m_2 = 24$ months.

As can be seen in Figure 2 - 5, the population of deer decreases drastically with time, whereas the population of foxes increases until a certain time and then decreases again. In table3 some values of the population development can be seen.

month	**0**	1	4	24	60	80	100	120
deer	**4000**	3821	3082	1190	540	440	405	403
foxes	**1000**	1191	1952	3529	3146	2707	2275	1882

Table 3: Cellular Automata results at certain time points

The whole course of events can be seen in Figure 6.

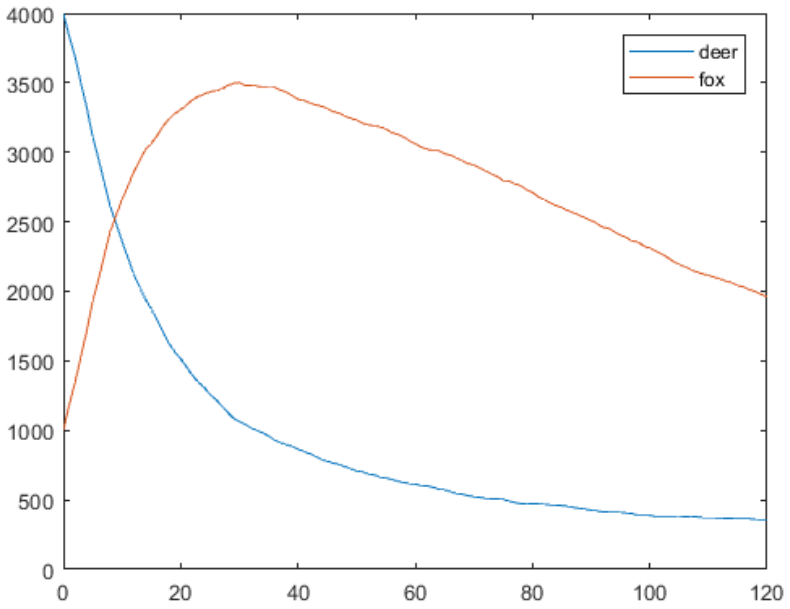

Figure 6: Population behaviour over 10 years with CA model

6.3 Task 4

The number of cells is fixed with N = 10,000. We are only changing the shape of the rectangle and observe the results. When we do not change the shape of the rectangle dramatically, the behaviour of the model also does not change a lot. Due to the randomly created matrix and the probability influences, each run of the code is different and cannot be repeated. To test extreme cases we set the rectangle in a way in which we set the ratio to (1 x 10,000) and compared it to a squared shape with a ratio of (100 x 100), the results can be seen in Figures 7 and 8.

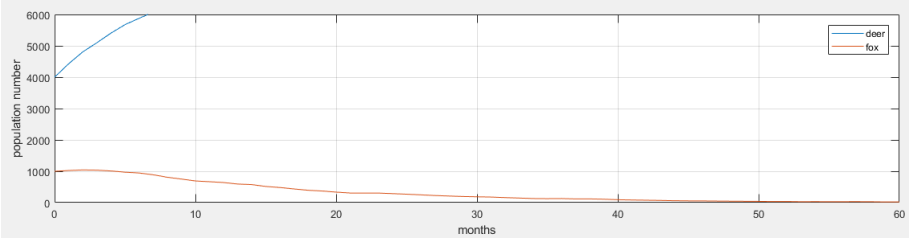

Figure 7: CA model with 1 x 10,000 ratio

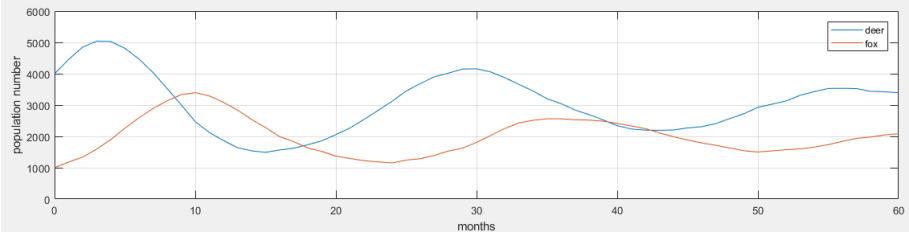

Figure 8: CA model with 100 x 100 ratio

As we can observe the results differ significantly. This is due to the fact that foxes are less likely to find deer in the 1 x 10,000 ratio forest because they only have two neighbours at maximum. Therefore they slowly die out since they have no food and deer reproduce and grow in population in this model.

6.4 Task 5

Now the parameters $\alpha, \beta, \gamma, \delta, M$ as well as p_1, p_2, p_3, N, k are changed so that the corresponding resulting curves of both models look alike.
The parameters were changed to:

$$
\begin{aligned}
p1 &= 0.44 \\
p2 &= 0.99 \\
p3 &= 0.25 \\
N &= 10,000 \\
k &= 2 \\
\alpha &= 0.001 \\
\beta &= 0.003 \\
\gamma &= 2 \\
\delta &= 0.001
\end{aligned}
$$

The resulting plot can be seen in Figure 9.

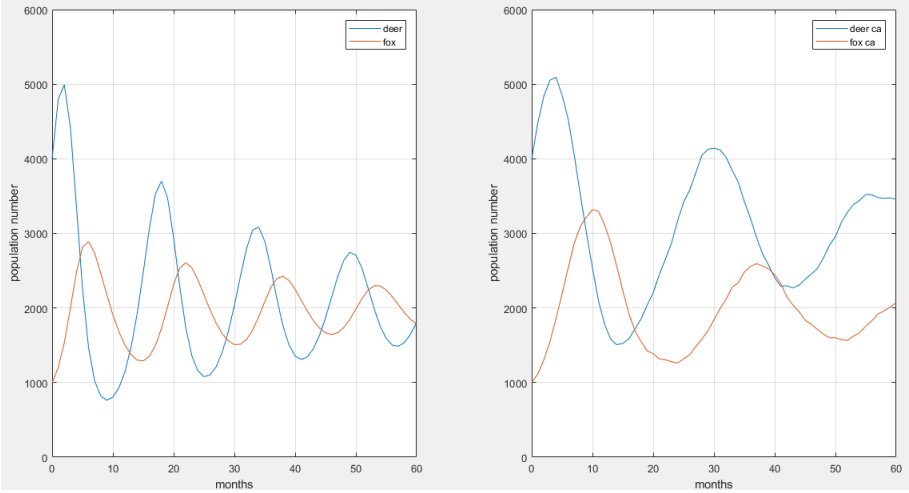

Figure 9: DE(left) and CA model(right)

7 Conclusion

In conclusion it can be seen that the predator thrives when there is plenty of prey and the prey thrives when there is no predators around that can eat them. In a system with 4 times as many prey as predators and reasonable assumptions for prey growth/reduction and predator reduction and fresh meat increase the differential equation model shows an oscillation in limit cycles meaning the population of both species decrease over time, while there is a short term oscillation. When modelling the system in a cellular automaton the results can vary a lot based on assumptions of probabilities of prey reproduction, predator reproduction and predator deaths. Because prey has no way of dying on its own in our model it can thrive in the forest very quickly if predators are not reproducing quickly enough or dying too quickly. Therefore setting the probability of predator reproduction very high and the probability of predator death rather low was essential to getting a model similar to the differential equation model. Another important factor was the distance of neighbours the predators could reach to find prey. We only looked at immediate neighbours in our initial model and had to increase this number to help the predators reproduce to get a result more similar to the differential equation. While this helped with the results of the CA model, increasing the distance of neighbours also meant a higher computer load, which made it difficult to analyse the models due to long run-times.

Even with optimizing all parameters the CA model didn't produce satisfactory results and can be improved. During our presentation of the project it was suggested that we improve the model by implementing an additional feature that allows prey to "run away" by changing its cell when it sees a predator. This would reflect reality more precisely.

Since the model is not deterministic but rather follows a stochastic distribution results vary on every test run. To solve this problem one could run a number of test runs and calculate average values for the results. This would also increase the accuracy of the model results.