Control of a model of competition between two animal species

Gianmario Alfonso Varchetta

30/11/2018

- Italian student of automatic engineering

- Erasmus plus student

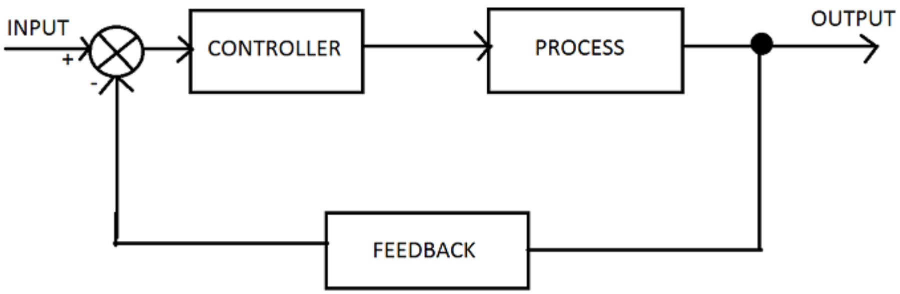
- Erasmus period: 3 months (November-January)

- Goal: Final Thesis work

Erasmus+

Control Strategy

$$\begin{cases} \dot{x} = f(x, u) \\ y = h(x) \end{cases}$$



$x(t) = state\ variables$
$u = control\ input$
$y(x) = output$

- Control of the system using feedback control:

Aim: regulate the system output at a fixed value

The Model

$$\frac{dx_1}{dt} = x_1(a - x_1 - bx_2)$$
$$\frac{dx_2}{dt} = x_2(c - x_2 - x_1)$$
$$x, y \geq 0$$

$\boldsymbol{x_1}$(t) = population of species1 (i.e. rabbits)
$\boldsymbol{x_2}$(t) = population of species 2 (i.e. Sheep)

- Competition for the same food supply and the amount available is limited.
- Each species would grow to its carrying capacity in the absence of the other.
- When both species coexist they start fighting for food.

Equilibria

- $x_A = (0;0)$
- $x_B = (0;2)$
- $x_C = (3;0)$
- $x_D = (1;1)$

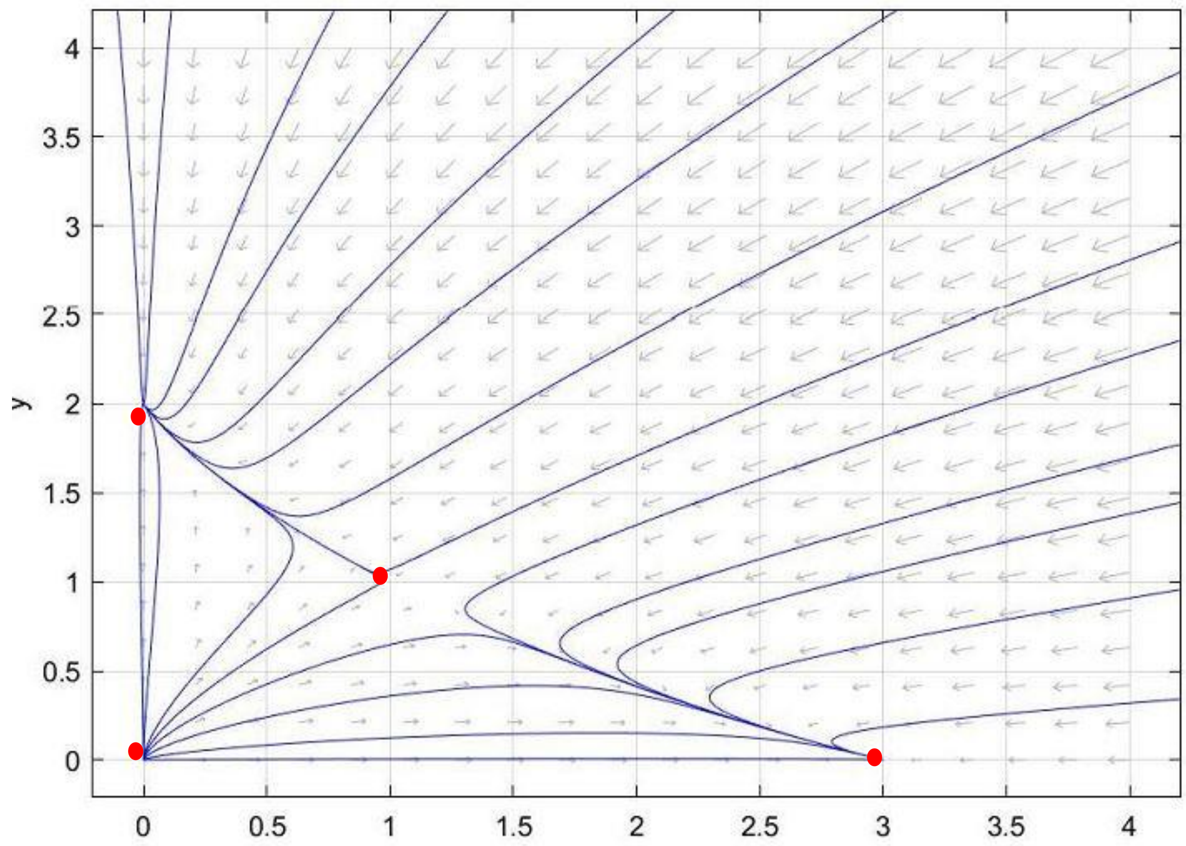
$$J = \begin{bmatrix} 3 - 2x_1 - 2x_2 & -2x_1 \\ -x_2 & 2 - x_1 - 2x_2 \end{bmatrix}$$

x_A *is a repeller node.*
x_B *is a stable node.*

x_C *is a stable node.*
x_D *is a saddle.*

$* \ a = 3 \quad b = 2 \quad c = 2$

Phase portrait



- *Principle of competitive exclusion:* two species competing for the same limited food typically cannot coexist
- Basins and their boundaries partition the phase plane into regions of different long-term behavior

How to control

Dependence on:

- ➢Food availability
- ➢Species rates

- o Control on food
- o Control on species rates
- o culling and fertility control

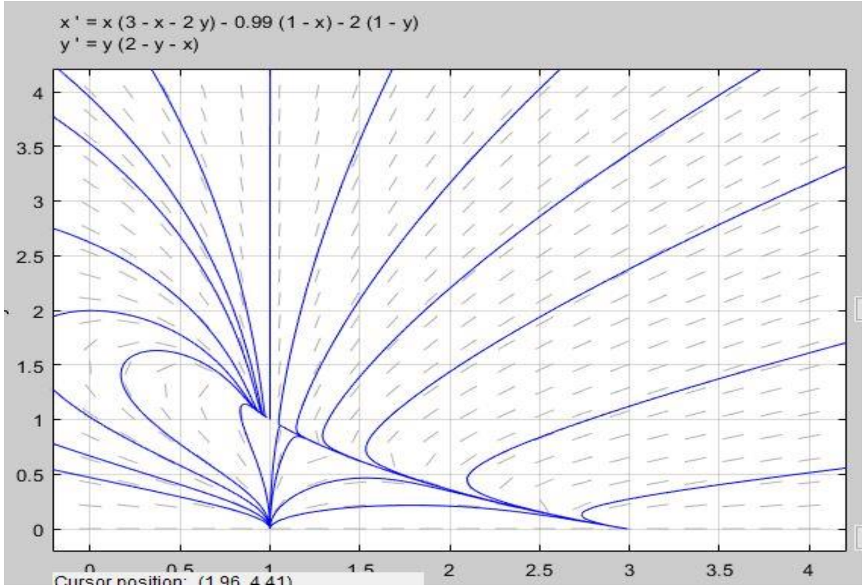
First results

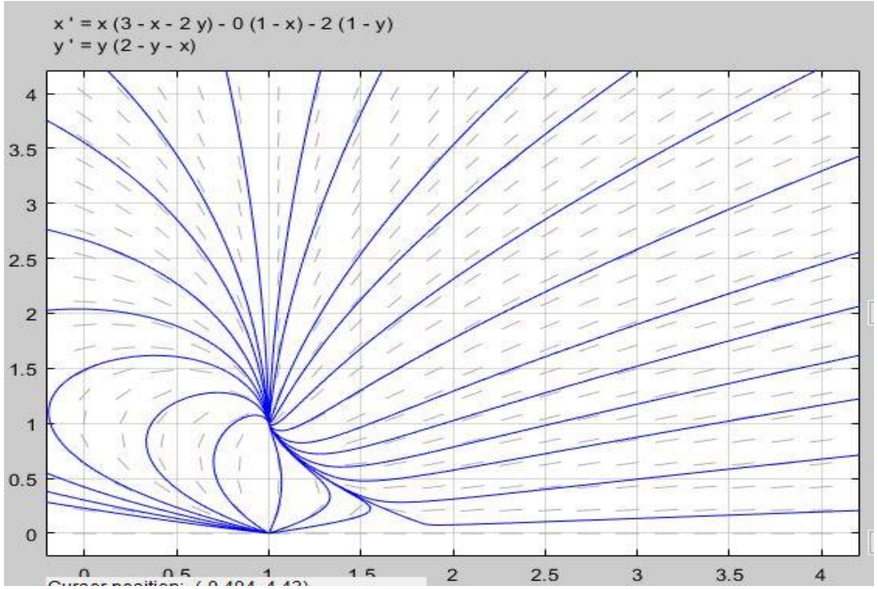
Aim: to stabilize the saddle in (1;1)

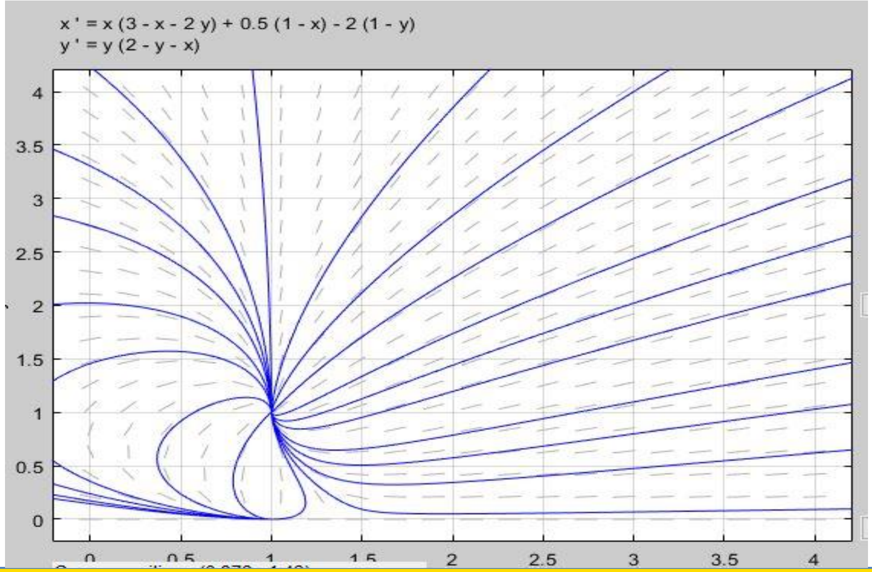
- Proportional error controller:

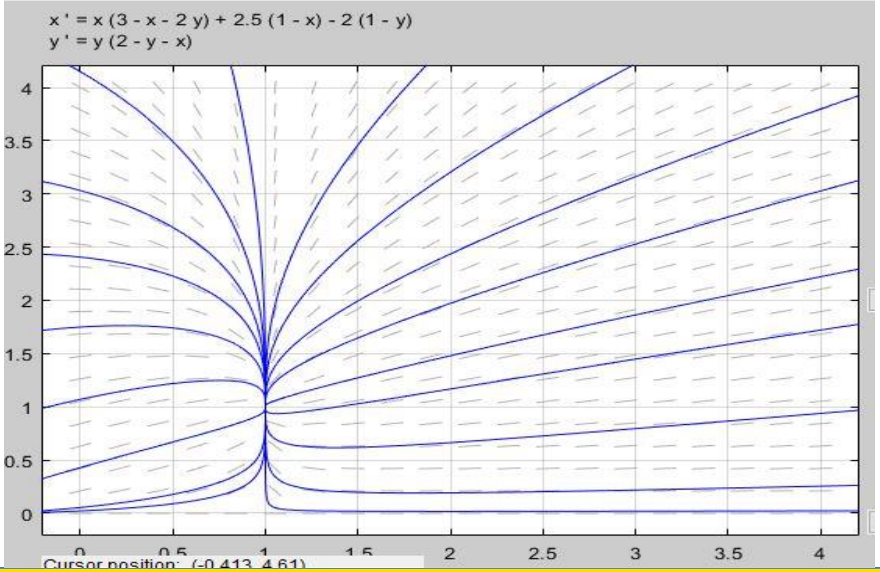
$$u = K_1(x_{1ref} - x_1) + K_2(x_{2ref} - x_2)$$

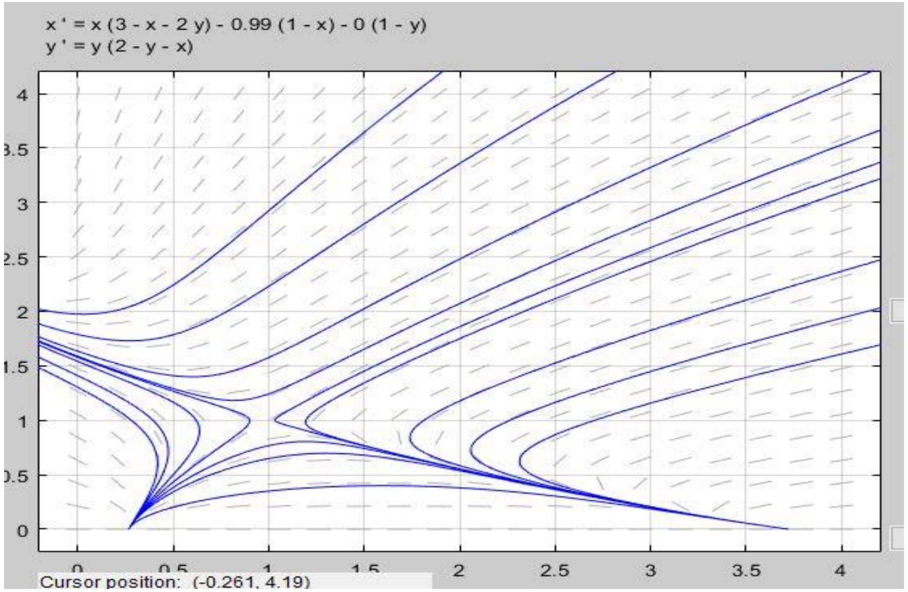
Varying K_1 and K_2, the dynamics of the closed loop system change

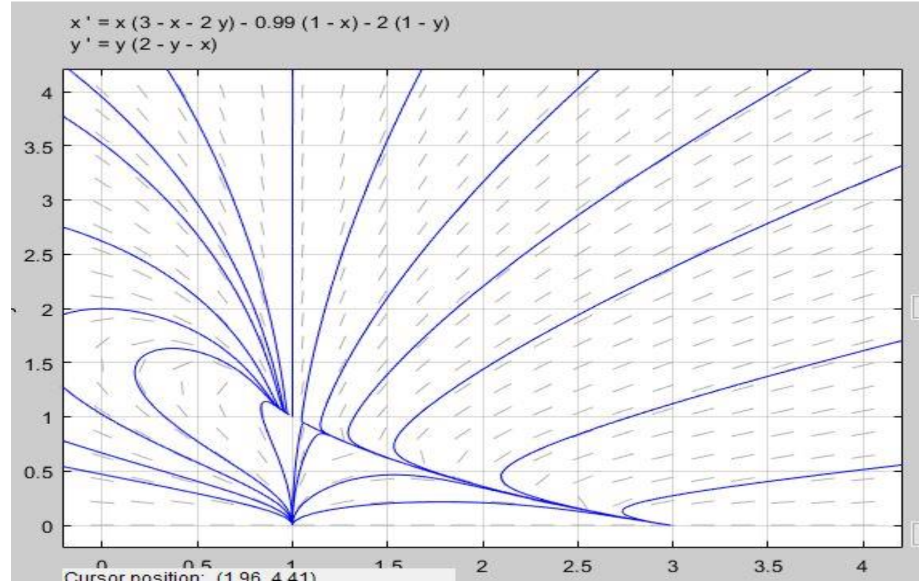

x ' = x (3 - x - 2 y) - 0.99 (1 - x) - 2 (1 - y)
y ' = y (2 - y - x)
Cursor position: (1.96, 4.41)

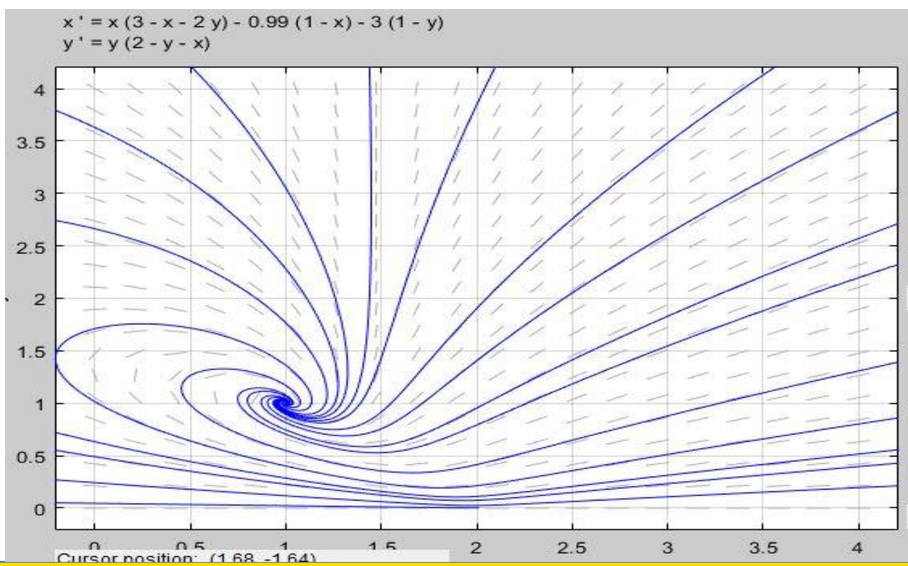

x ' = x (3 - x - 2 y) - 0 (1 - x) - 2 (1 - y)
y ' = y (2 - y - x)

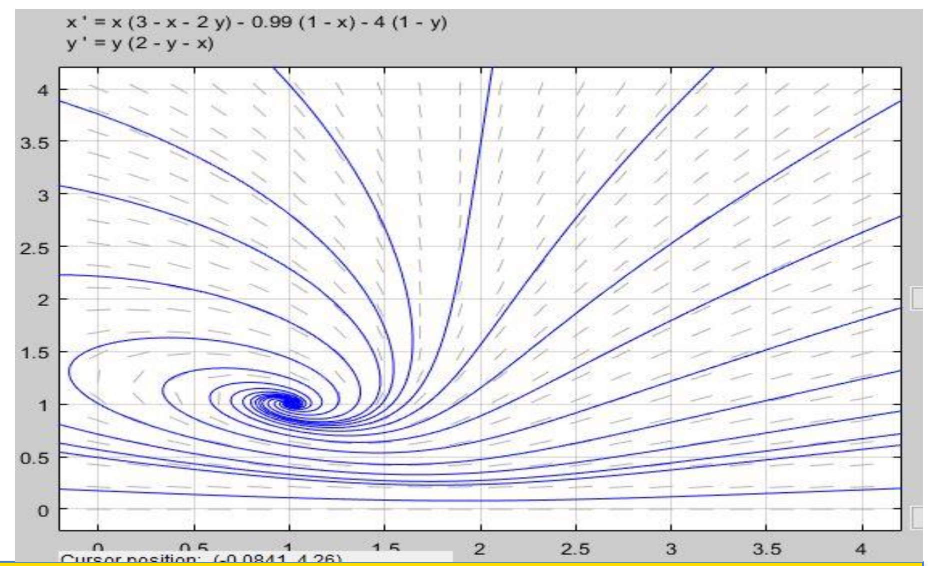

x ' = x (3 - x - 2 y) + 0.5 (1 - x) - 2 (1 - y)
y ' = y (2 - y - x)


x ' = x (3 - x - 2 y) + 2.5 (1 - x) - 2 (1 - y)
y ' = y (2 - y - x)
Cursor position: (-0.413, 4.61)


x ' = x (3 - x - 2 y) - 0.99 (1 - x) - 0 (1 - y)
y ' = y (2 - y - x)
Cursor position: (-0.261, 4.19)


x ' = x (3 - x - 2 y) - 0.99 (1 - x) - 2 (1 - y)
y ' = y (2 - y - x)


x ' = x (3 - x - 2 y) - 0.99 (1 - x) - 3 (1 - y)
y ' = y (2 - y - x)


x ' = x (3 - x - 2 y) - 0.99 (1 - x) - 4 (1 - y)
y ' = y (2 - y - x)

Future steps

- Study the Jacobian Matrix and its eigenvalues depending on K
- Try to get bigger or smaller the basins of attraction of the stable nodes
- Implement different kind of control and compare the results(i.e. Adding an integral part of error)