Lab 01. Two Competing Species: Coupled Logistic Maps

Jordi-Lluís Figueras

Modelling Complex Systems, Spring 2026

1 Introduction

In this lab we study a simple discrete-time model for two competing populations. If a single species evolves in isolation, a standard model is the logistic map

$$x_{n+1} = r x_n (1 - x_n),$$

where $x_n \in [0, 1]$ denotes the normalized population at generation n and $r > 0$ is the intrinsic growth parameter. The factor $(1 - x_n)$ models limited resources: when the population is small, growth is approximately linear, while for large population density the competition for resources reduces growth.

Now consider two populations, with normalized densities x_n and y_n, living in the same environment and competing for the same finite resources. If the two species were completely independent, one could write the dynamical system

$$x_{n+1} = r_1 x_n (1 - x_n), \qquad y_{n+1} = r_2 y_n (1 - y_n).$$

Competition can be modeled by assuming that each population is affected not only by its own reproductive dynamics, but also by the presence of the other one. In the simplified model used in this lab, the next generation is a convex combination of the two logistic updates:

$$\begin{aligned} x_{n+1} &= (1 - \varepsilon) r_1 x_n (1 - x_n) + \varepsilon r_2 y_n (1 - y_n), \\ y_{n+1} &= (1 - \varepsilon) r_2 y_n (1 - y_n) + \varepsilon r_1 x_n (1 - x_n). \end{aligned}$$

Here $\varepsilon \in [0, 1]$ measures the coupling strength.

For $\varepsilon = 0$, the two populations evolve independently. For $\varepsilon > 0$, each population receives a contribution from the other one. This should be understood as a stylized interaction model rather than as a fully mechanistic competition model: the success of one population modifies the effective update of the other through the coupling term. Even though the model is simple, it already produces fixed points, periodic orbits, and chaotic dynamics.

2 Exercises

Work in a clear and reproducible way. Your report should explain both the mathematics and the numerical experiments. Unless explicitly stated otherwise, all exercises below are part of the lab.

1. Fixed points

Find the fixed points of the coupled system, that is, the points (x^*, y^*) satisfying

$$x^* = (1-\varepsilon)r_1 x^*(1-x^*) + \varepsilon r_2 y^*(1-y^*),$$

$$y^* = (1-\varepsilon)r_2 y^*(1-y^*) + \varepsilon r_1 x^*(1-x^*).$$

In particular:

- identify the trivial fixed point,
- discuss the diagonal case $x^* = y^*$,
- investigate whether asymmetric fixed points may exist when $r_1 \neq r_2$.

When discussing the diagonal case, explain separately what happens for $r_1 = r_2$ and for $r_1 \neq r_2$.

2. Periodic orbits of period 2 and 3

Write a Python program, **periodicPoints.py**, that computes periodic orbits of period 2 and period 3 for the coupled map.

More precisely, a point (x, y) belongs to a period-p orbit if

$$F^p(x, y) = (x, y),$$

with $p = 2$ or $p = 3$, and it is not already a periodic orbit of smaller period.

It is recommended to solve these nonlinear equations numerically by using Newton's method.

In this exercise, treat r_1, r_2, and ε as fixed parameters, and solve for the unknown point (x, y). A practical formulation is to define

$$G_p(x, y) = F^p(x, y) - (x, y),$$

and then solve

$$G_p(x, y) = (0, 0)$$

for $p = 2$ and $p = 3$.

Your program should:

- implement the map and its iterates F^2 and F^3,
- formulate the nonlinear systems corresponding to period-2 and period-3 points,
- use Newton's method, or a standard numerical root solver based on the same idea, to locate solutions,
- use several initial guesses and report which ones converge,
- verify that the computed solutions really have minimal period 2 or 3.

Discuss for which parameter values you are able to detect these periodic orbits, and compare them with what you observe in direct iteration.

Suggested parameter values to start with are

$$(r_1, r_2, \varepsilon) = (3.1, 3.4, 0.3), \qquad (3.1, 3.55, 0.3), \qquad (3.1, 3.8, 0.3).$$

You may then explore other values if needed.

3. Phase portraits

Write a Python program, **phasePortrait.py**, that iterates the coupled map and plots the orbit in the phase plane (x_n, y_n).

Your program should:

- implement one iteration of the coupled map,
- discard a suitable transient,
- plot the long-time orbit in the (x, y) plane,
- produce several phase portraits for different values of r_1, r_2, and ε.

Compare qualitatively different regimes. For example, try to identify cases in which the orbit approaches a fixed point, a periodic orbit, or a more irregular set.

As a starting point, you may test the following parameter values:

$$(r_1, r_2, \varepsilon) = (2.8, 2.9, 0.1), \qquad (3.1, 3.4, 0.3), \qquad (3.85, 3.95, 0.2).$$

4. Bifurcation cascade

Write a Python code, **bifurcationCascade.py**, that reproduces for the coupled system a bifurcation cascade analogous to the one studied in the one-dimensional logistic map.

More precisely, fix $r_1 = 3.1$ and $\varepsilon = 0.3$, vary r_2 in the interval

$$2.8 \leq r_2 \leq 3.97,$$

discard a transient, and plot the long-time values of x_n against r_2.

Your code should:

- sweep a sufficiently fine grid of r_2 values, for example about 800 values,
- keep several iterates after the transient, for example about 200–300 iterates,
- plot the resulting point cloud (r_2, x_n),
- use labels and a title that clearly indicate the chosen parameters.

After producing the figure, comment on what changes as r_2 increases, and try to identify regions with a single branch, a small number of branches, and a broader cloud of values.

3 What to submit

Submit:

- a short report in PDF format,
- the Python files used for the numerical computations and the figures,
- the generated figures, if they are not already embedded in the report.

Your report should include:

- the derivation or discussion of the fixed points,

- the computation of period-2 and period-3 periodic orbits,
- phase portraits for several parameter choices,
- the bifurcation cascade plot,
- a brief interpretation of the observed dynamics,
- in case LLM tools were used, an appendix reporting the most important prompts used for the project.