

Section 7.3 : Competing Species

Chapter 7 : Systems of First Order Linear Equations

Math 2552 Differential Equations

Section 7.3

Topics

We will cover these topics in this section.

1. Competing species

Objectives

The objectives for this section are the same as those for 7.2. This section applies what we learned in 7.2 to specific cases.

Population Modelling

Recall the **logistic equation**:

$$\frac{dx}{dt} = x(\epsilon - \sigma x) \quad (1)$$

where

- ϵ is a growth rate,
- $\frac{\sigma}{\epsilon}$ is a saturation level,
- the quantity $\epsilon - \sigma x$ represents an environmental capacity for a species (eg - food supply),
- $x(t)$ represents the population of **one species** at time t .

We want to extend these concepts to **two species**.

Two Species

Now suppose that we have two species that do not interact directly with other, but they do compete for the same food supply.

We could use two separate **logistic equations**:

$$\frac{dx}{dt} = x(\epsilon_1 - \sigma_1 x), \quad \frac{dy}{dt} = y(\epsilon_2 - \sigma_2 y) \quad (2)$$

But the two species rely on the same, limited food source. They impact each others' environmental capacity.

We can instead use

$$\frac{dx}{dt} = x(\epsilon_1 - \sigma_1 x - \alpha_1 y), \quad \frac{dy}{dt} = y(\epsilon_2 - \sigma_2 y - \alpha_2 x) \quad (3)$$

We are interested in cases where $\epsilon_1, \sigma_1, \alpha_1, \epsilon_2, \sigma_2, \alpha_2$ are nonnegative.

Examples (as time permits)

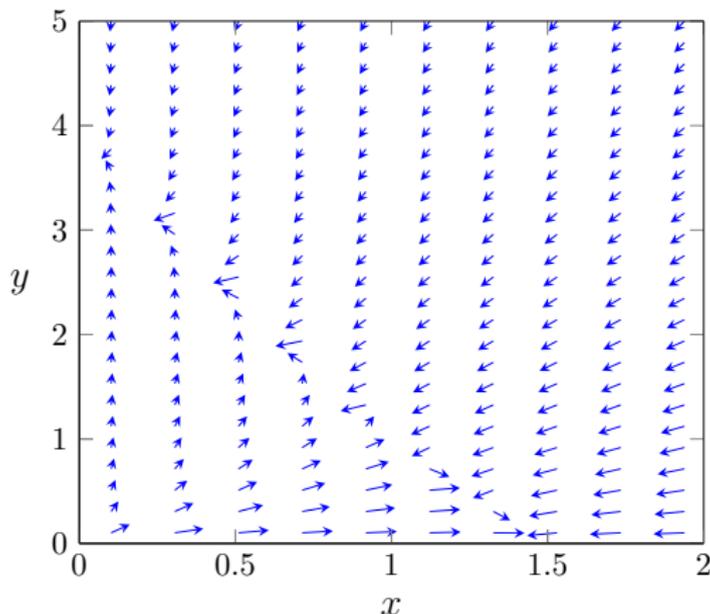
For each system below a) identify all critical points, b) construct the linear system for each critical point, and c) classify the critical points of the linear system according to stability (stable, unstable, asymptotically stable).

$$1) \quad \frac{dx}{dt} = x(1.5 - x - 0.5y), \quad \frac{dy}{dt} = y(2 - 0.5y - 1.5x)$$

$$2) \quad \frac{dx}{dt} = x(1.5 - x/2 - y), \quad \frac{dy}{dt} = y(3/4 - y - 0.125x)$$

Slope Field

A phase portrait of the first system in the previous example is below.



Alternatively, we can obtain the portrait using wolframalpha:

```
streamplot[{x(1.5 - x - y/2), y * (2 - y/2 - 1.5x)}, {x, 0, 2}, {y, 0, 5}]
```