Continuous modelling by ODEs

Numerical methods

Computational Biology

Adérito Araújo (alma@mat.uc.pt)
June 27, 2024

ODE solvers: open the black-box

- Initial value problem: find $x(t)$ such that

$$\begin{cases} \dfrac{dx}{dt} = f(t,x), \\ x(0) = x_0. \end{cases}$$

- Euler method: find $x_n \approx x(t_n)$ on $t_n = n\Delta t$, $n = 0, 1, 2, \ldots$, such that

$$\frac{x_{n+1} - x_n}{\Delta t} = f(t_n, x_n),$$

which is equivalent to

$$x_{n+1} = x_n + \Delta t f(t_n, x_n), \qquad n = 0, 1, 2, \ldots.$$

Note: For autonomous equations (f do not depend explicitly on t)

$$x_{n+1} = x_n + \Delta t f(x_n), \qquad n = 0, 1, 2, \ldots.$$

Euler method

Exercise 2.19: Consider the initial value problem

$$\begin{cases} \frac{dx}{dt} &= t + x \\ x(0) &= 1 \end{cases}.$$

Find an approximate solution $x_n \approx x(t_n)$ on $t_n = n0.5$, $n = 0, 1, 2$.

Solution:

- For $n = 0$: $t_0 = 0$ and

$$x(0) = x_0 = 1.$$

- For $n = 1$: $t_1 = t_0 + \Delta t = 0.5$ and

$$x(t_1) \simeq x_1 = x_0 + \Delta t f(t_0, x_0) = 1 + 0.5 \times 1 = 1.5.$$

- For $n = 2$: $t_2 = t_1 + \Delta t = 1$ and

$$x(t_2) \simeq x_2 = x_1 + \Delta f(t_1, x_1) = 1.5 + 0.5 \times 2 = 2.5.$$

Euler method: geometric interpretation

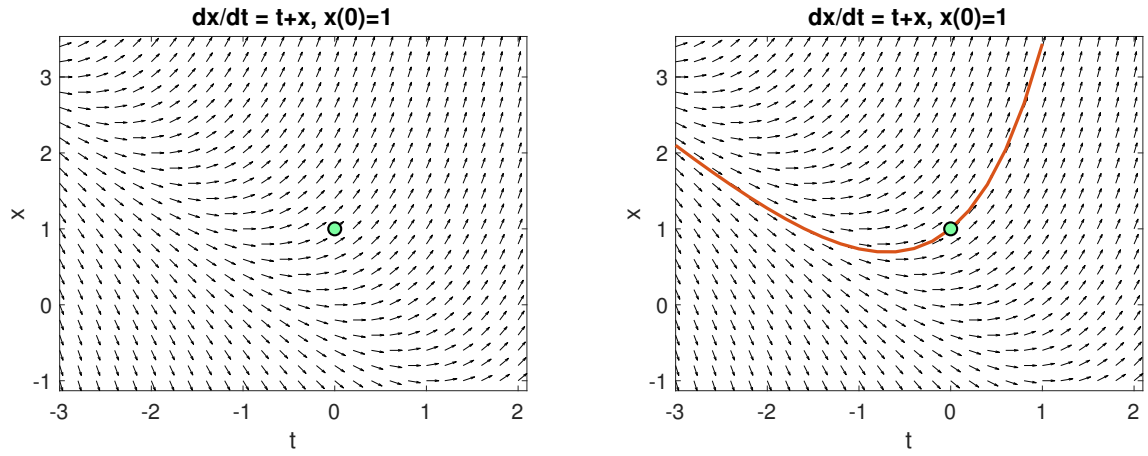


Figure: Left: initial condition; Right: exact solution.

Euler method: geometric interpretation

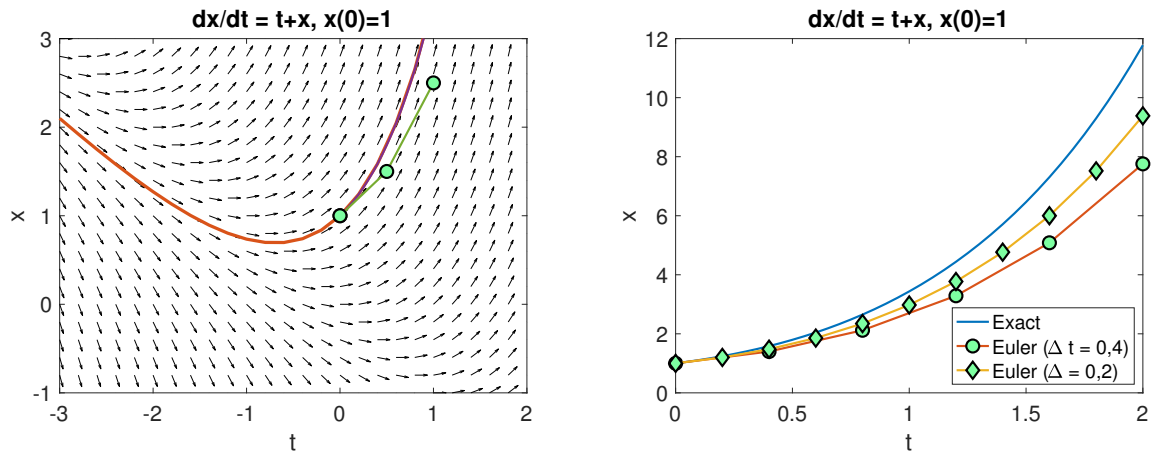


Figure: Left: exact solution and Euler solution ($\Delta t = 0.5$); Right: Euler solution ($\Delta t = 0.2, 0.4$).

Euler method: Matlab code

```
% Matlab code to solve an IVP with Euler method
f = @(t,x) t + x;          % function for the ODE
tint = [0 2]; dt = 0.5; % time interval and step size
x0   = 1;                  % initial condition
[t, x] = euler(f, tint, x0, dt);
plot(t, x)                 % plot the solution
```

```
function [t,x] = euler(f,tint,x0,dt)
% Euler method for
%   dx/dt = f(t,x), x(t0) = x0
%   on tint = [t0, tfinal]
t0 = tint(1); tfinal = tint(2);
t = t0:dt:tfinal;
x = zeros(size(t)); x(1) = x0;
for n = 1:length(t)-1
    x(n+1) = x(n) + dt*f(t(n),x(n));
end
end
```

Runge-Kutta methods

RK4: the four-stage, fourth-order RK method

$$k_1 = f(t_n, x_n), \quad k_2 = f(t_n + \frac{\Delta t}{2}, x_n + \frac{\Delta t}{2} k_1),$$
$$k_3 = f(t_n + \frac{\Delta t}{2}, x_n + \frac{\Delta t}{2} k_2), \quad k_4 = f(t_n + \Delta t, x_n + \Delta t k_3),$$
$$x_{n+1} = x_n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4), \quad n = 0, 1, \ldots, N-1.$$

RK4 for $dx/dt = f(t, x)$, $x(t_0) = x_0$
Fix $T > 0$. Choose $\Delta t > 0$. Obtain N.
For $n = 0, 1, \ldots, N-1$

1. $k_1 = f(t_n, x_n)$, $k_2 = f(t_n + \Delta t/2, x_n + \Delta t k_1/2)$,
   $k_3 = f(t_n + \Delta t/2, x_n + \Delta t k_2/2)$, $k_4 = f(t_n + \Delta t, x_n + \Delta t k_3)$.
2. $x_{n+1} = x_n + \Delta t\,(k_1 + 2k_2 + 2k_3 + k_4)/6$.
3. $t_{n+1} = t_n + \Delta t$.

Maltab solvers ode23 and ode45

Solver	Method	Characteristics
ode45	six-stage, fifth-order, RK method	ode45 does more work per step than ode45, but can take much larger steps. For differential equations with smooth solutions, ode45 is often more accurate than ode23
ode23	three-stage, third-order, RK method	ode23 can be more efficient than ode45 at problems with crude tolerances, or in the presence of moderate stiffness

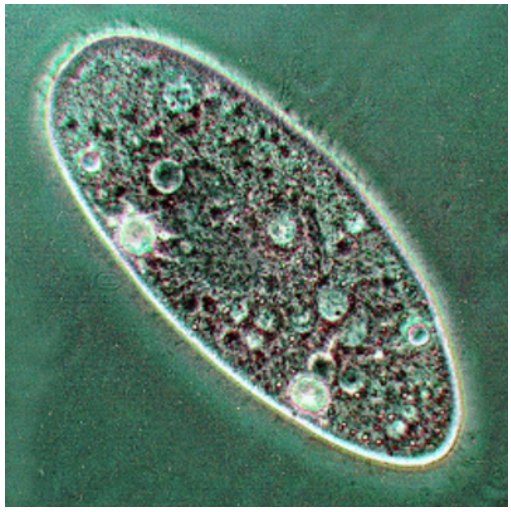
https://blogs.mathworks.com/cleve/2014/05/26/ordinary-differential-equation-solvers-ode23-and-ode45/

Mathematical modelling flowchart

1. Collect the data
2. Derive the equations based on knowledge and assumptions
3. Solve equations numerically using initial guesses for parameters
4. Obtain values of some parameters from literature or previous studies
5. Use data to estimate the remaining model parameters
6. Test model fit and predictive ability
7. Are mode predictions satisfactory? If YES GOTO 8; if NO
   - 7.1 Revise model structure based on new knowledge and GOTO 2 or collect new/more/better data and GOTO 1
   - 7.2 Devise and conduct more experiments and GOTO 5
8. Use the model

Cell competition: G.F. Gause (1932) experiment

In two containers containing the same growth medium, populations of *Paramecium caudatum* and *Paramecium aurelia* are grown. Also, in a larger container, the two populations are mixed and grown together, competing for the same resources. The populations are measured once a day.



1. Develop a model of the competition, and fit it to the given data.
2. What does your model predict about the long-term viability of the populations (will both populations survive, or will one population become extinct)?

1. Collect the data

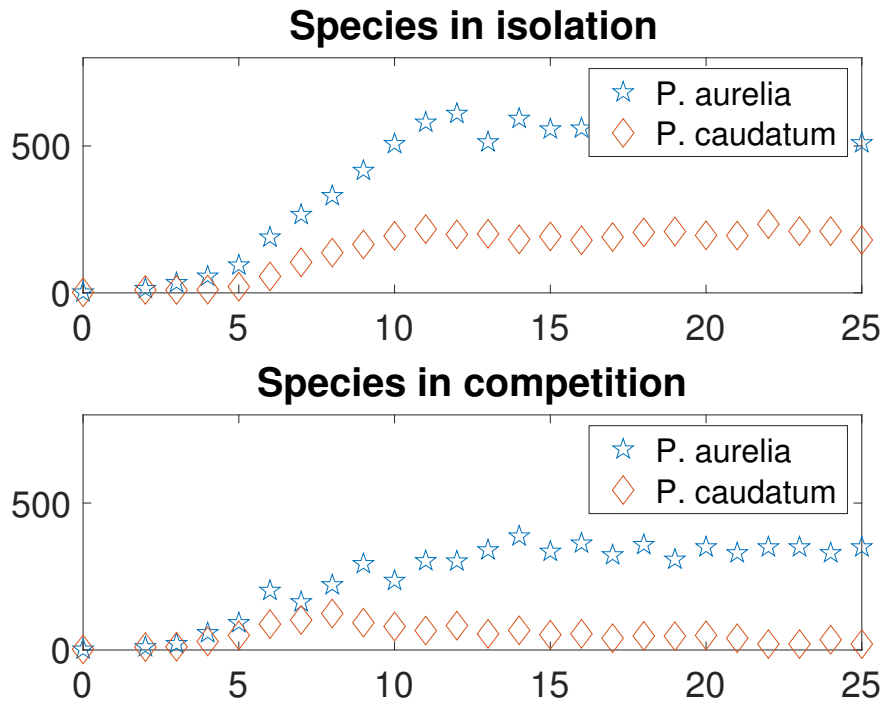
Day	Mean density (individuals per $cm^3/2$)	
	Paramecium aurelia	*Paramecium caudatum*
0	2 (2)	2 (2)
1	-	-
2	14 (10)	10 (10)
3	34 (21)	10 (11)
4	56 (58)	11 (29)
5	94 (92)	21 (50)
6	189 (202)	56 (88)
7	266 (163)	104 (102)
8	330 (221)	137 (124)
9	416 (293)	165 (93)
10	507 (236)	194 (80)
11	580(303)	217(66)
12	610(302)	199 (83)
13	513 (340)	201 (55)
14	593 (387)	182 (67)
15	557 (335)	192 (52)
16	560 (363)	179 (55)
17	522 (323)	190 (40)
18	565 (358)	206 (48)
19	517 (308)	209 (47)
20	500 (350)	196 (50)
21	585 (330)	195 (40)
22	500 (350)	234 (20)
23	495 (350)	210 (20)
24	525 (330)	210 (35)
25	510 (350)	180 (20)

Table: Data collected by G.F. Gause (1932): in isolation (in competition).

Read data from an excel file

```
% read data
I = readmatrix('gause.xlsx','Sheet','isolation');
C = readmatrix('gause.xlsx','Sheet','competition');
% define lines and columns (optional)
lines = 1:25;
day = 1; aurelia = 2; caudatum = 3;
% define vectors with the collected data
t = I(lines,day);
x1i = I(lines,aurelia);
x2i = I(lines,caudatum);
x1c = C(lines,aurelia);
x2c = C(lines,caudatum);
% plot the data
subplot(211), plot(t,x1i,'p',t,x2i,'d')
legend('aurelia','caudatum'), title('isolation')
subplot(212), plot(t,x1c,'p',t,x2c,'d')
legend('aurelia','caudatum'), title('competition')
```

Plot the collected data



2. Derive the model

- Species x_1 and x_2 in isolation

$$\frac{dx_i}{dt} = r_i x_i \left(1 - \frac{x_i}{K_i}\right), \qquad i = 1, 2,$$

where $x_i(t)$ is the mean density (in individuals per 0.5 cm^3) at time t (in days), r_i is the instantaneous rate of increase (births/deaths), and K_i is the carrying capacity per 0.5 cm^3.

- Species x_1 and x_2 in competition (see HW#3)

$$\begin{cases} \dfrac{dx_1}{dt} = r_1 x_1 \left(1 - \dfrac{x_1}{K_1} - \alpha \dfrac{x_2}{K_1}\right) \\ \dfrac{dx_2}{dt} = r_2 x_2 \left(1 - \dfrac{x_2}{K_2} - \beta \dfrac{x_1}{K_2}\right) \end{cases},$$

where α and β are competition coefficients.

3. Solve numerically: isolation (phase 1)

```
% initial guesses:  rate and carrying capacity
r10 = 1;   K10 = 540; p10 = [r10,K10];
r20 = 0.3; K20 = 200; p20 = [r20,K20];
% solve numerically
tfit  = linspace(t(1),t(end),100); % 100 points
x1fit = flogistic(p10,tfit);
x2fit = flogistic(p20,tfit);
plot(tfit,x1fit,tfit,x2fit)
```

```
function x = flogistic(p,t)
% solve the logistic model
r = p(1); K = p(2);
logistic = @(t,x) r*x.*(1-x/K);
x0 = 2; % initial condition
[t,x] = ode45(logistic,t,x0);
end
```

Solution for the initial parameters: isolation

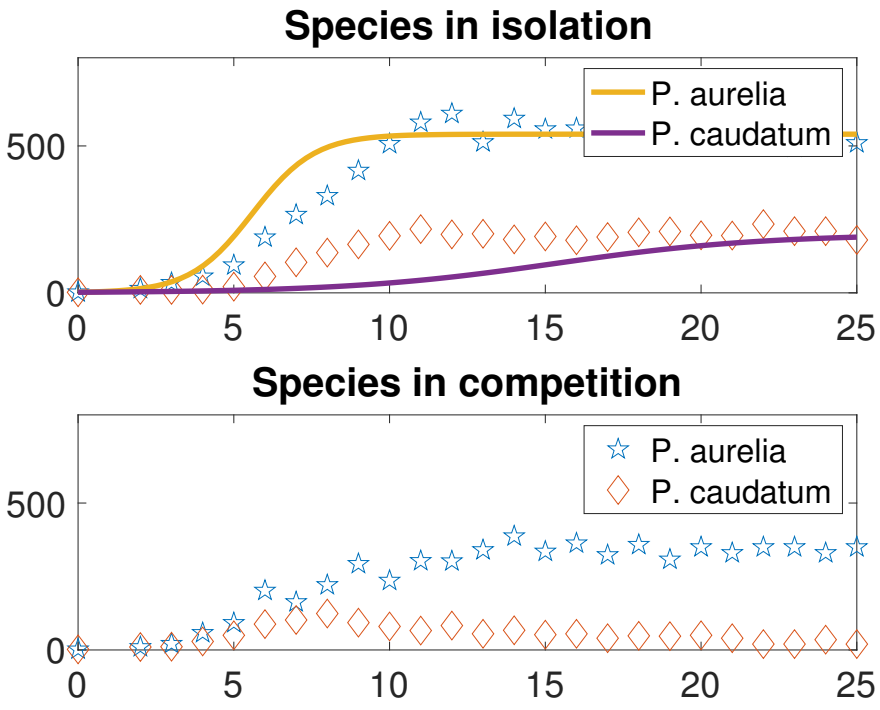


Figure: $r_1 = 1$, $K_1 = 540$, $r_2 = 0.3$, $K_2 = 200$.

4. Parameter optimization: least squares method (phase 1)

- System of ODEs

$$\frac{dx}{dt} = f(x;p), \qquad x(0) = x_0,$$

where $p = (p_1, \ldots, p_n)$ a set on unknown parameters.

- Goal: obtain p such that $x(t;p)$ (that depends on p) fits the experimental data

$$\{(t_0, x_0), (t_1, x_1), \ldots, (t_m, x_m)\},$$

where x_i is the collected data for t_i, $i = 1, ..., m$ ($m \gg n$).

- Least squares method: minimize the total quadratic error

$$E(p) = \sum_{i=0}^{m} (x_i - x(t_i;p))^2.$$

- Matlab code for the least squares method

```
p1 = lsqcurvefit(@flogistic,p10,t,x1i);
p2 = lsqcurvefit(@flogistic,p20,t,x2i);
```

Solution for the optimized parameters: isolation

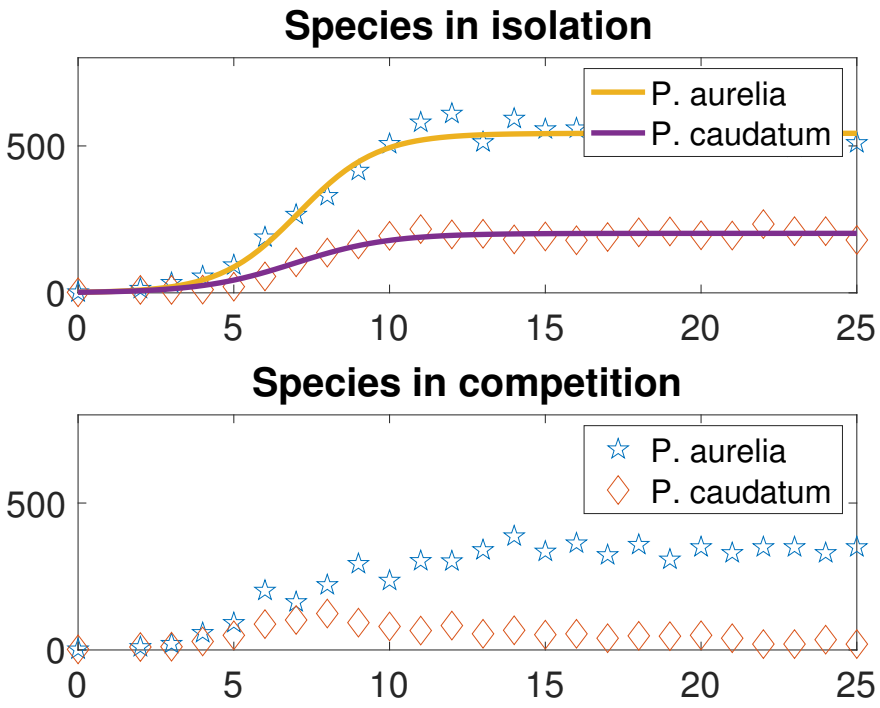


Figure: $r_1 = 0.79$, $K_1 = 542.94$, $r_2 = 0.66$, $K_2 = 202.50$.

5. The competition model (phase 2)

```
% competition coefficients
alpha0 = 1; beta0 = 1; pcomp0 = [alpha0,beta0];
% parameter optimization
xc = [x1c x2c];     % both species together
pcomp = lsqcurvefit(@fcompetition,pcomp0,t,xc);
xcfit = fcompetition(pcomp,tfit); plot(tfit,xcfit);

function x = fcompetition(pcomp,t)
% solve the competition model
x0 = [2; 2]; [t,x] = ode45(@competition,t,x0);
  function dxdt = competition(t,x)
  global p1 p2
  alpha = pcomp(1); beta = pcomp(2);
  r1 = p1(1); K1 = p1(2); r2 = p2(1); K2 = p2(2);
  dx1dt = r1*x(1).*(1-x(1)/K1-alpha*x(2)/K1);
  dx2dt = r2*x(2).*(1-x(2)/K2-beta*x(1)/K2);
  dxdt = [dx1dt;dx2dt];
  end
end
```

Solution for the optimized parameters: competition

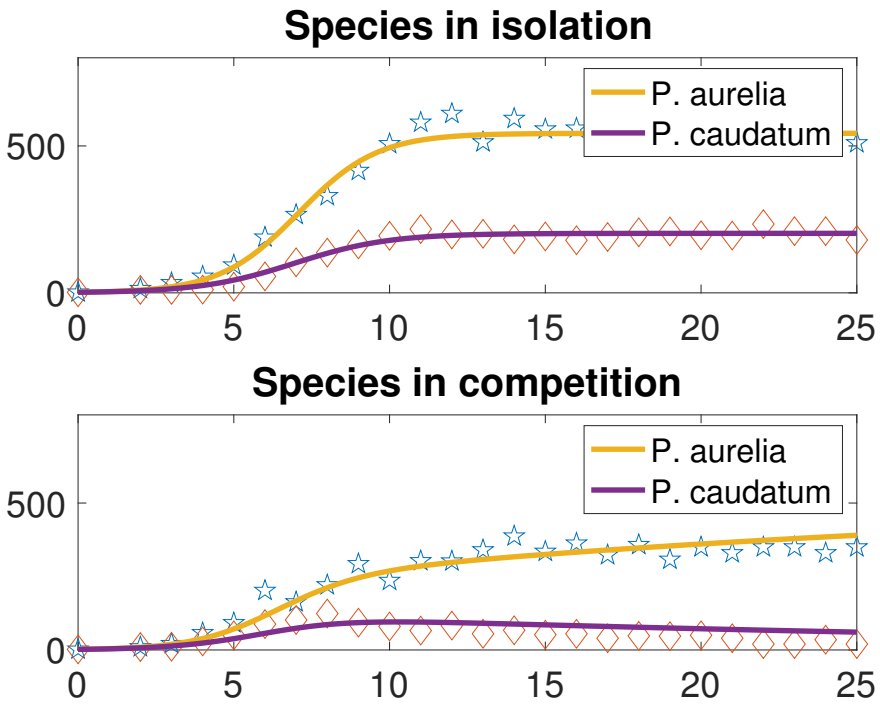


Figure: $\alpha = 2.36$, $\beta = 0.39$.

5. Conclusion

- Agreement between the data and the model is good.
- It appears that *P. caudatum* is heading towards either extinction or a small steady-state population. Consequently, *P. aurelia* would grow towards its carrying capacity in isolation or close to it.
- Is the coexistence possible? Following HW#3

$$a = \alpha \frac{K_2}{K_1} = 2.36 \frac{202.50}{542.94} = 0.88 < 1$$

$$b = \beta \frac{K_1}{K_2} = 0.39 \frac{542.94}{202.50} = 1.05 > 1,$$

and so the answer is (maybe) no.

Note: In book [1, chapter 10] (see Lecture 1), the authors obtained a value of $\beta = 0.36$ and so both $a < 1$ and $b < 1$, which corresponds to a coexistence of both species.

Case study

Modelling circadian rhythms

Computational Biology

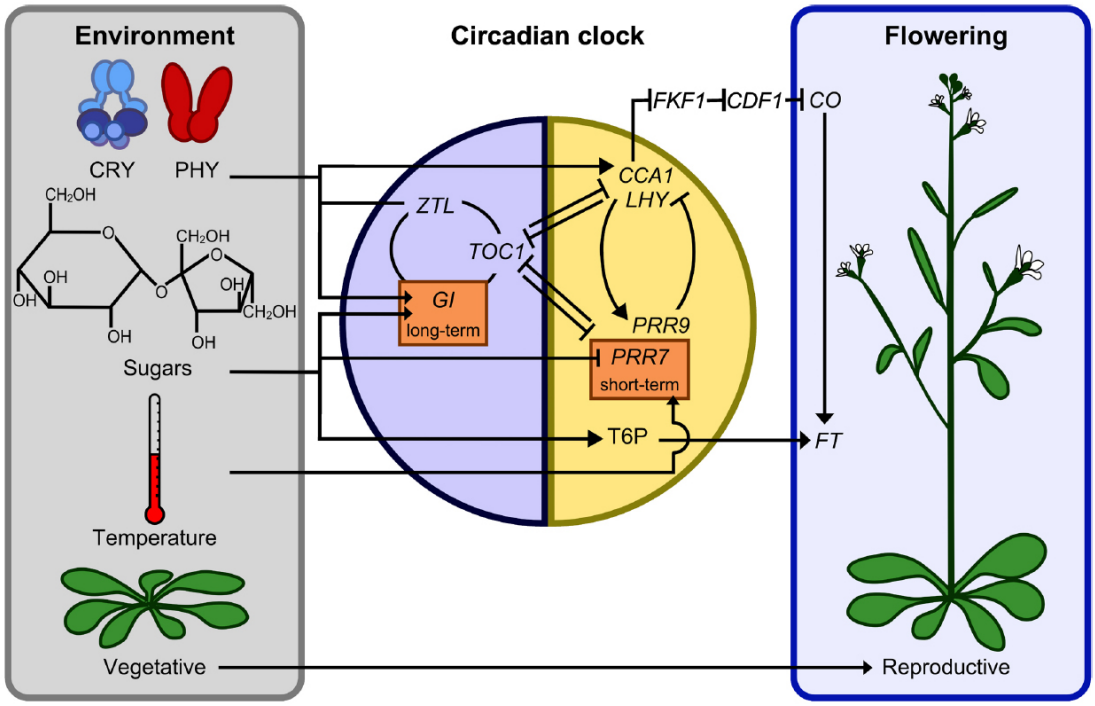
Adérito Araújo (alma@mat.uc.pt)
June 27, 2024

Arabidopsis thaliana

The plants cannot escape the external environment conditions since they are immobile organisms.

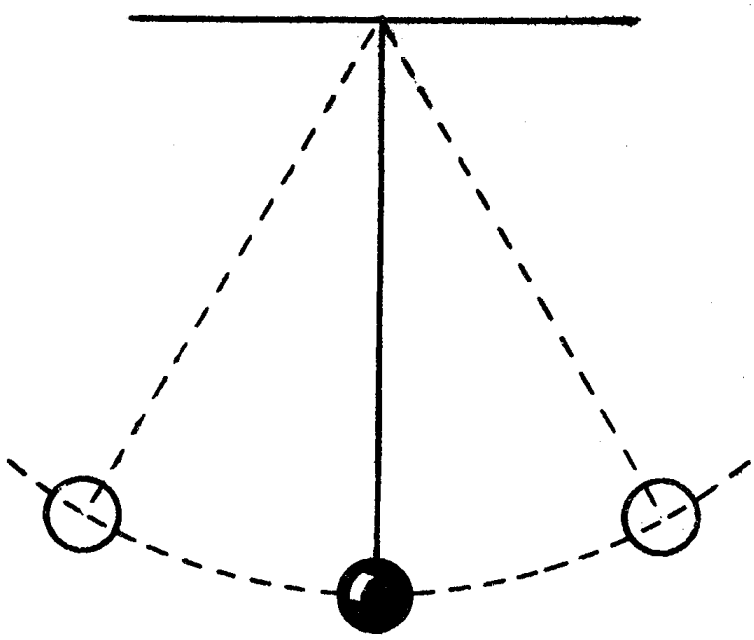
Arabidopsis thaliana: small plant; relative short life cycle; produces many seeds; its genome in known.

Basic model of a clock: *Arabidopsis thaliana*



Biological clocks

An harmonic oscillator is a system that executes a periodic behavior.



Harmonic-like oscillators

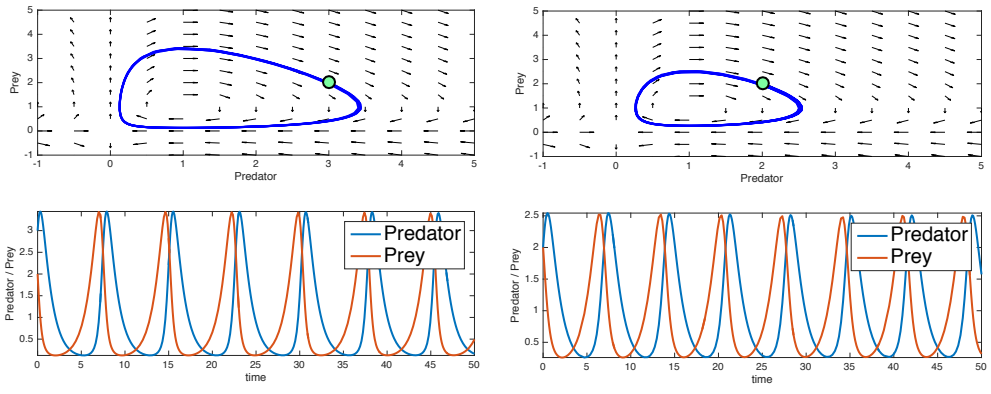


Figure: The Lotka-Volterra model behaves like a harmonic oscillator: changing the initial number of preys/predators changes the amplitude of the oscillations.

Biological oscillators

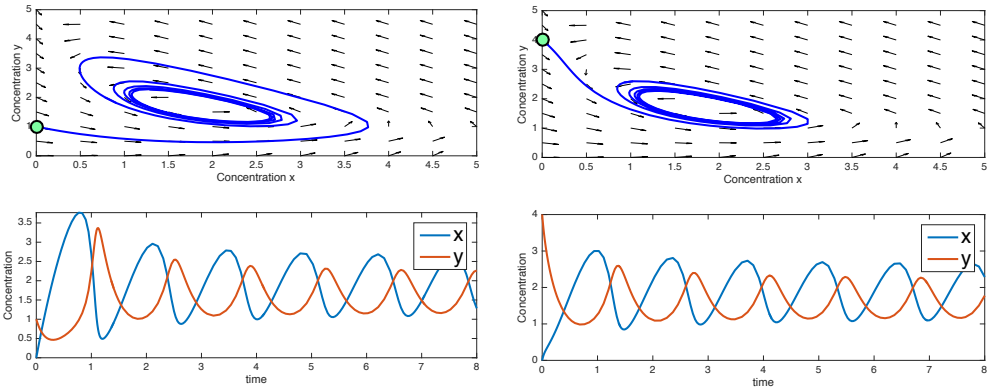


Figure: Biological oscillators tend to have not only a characteristic period, but also a characteristic amplitude. If a perturbation is exerted on such a system, they will automatically come back to their normal behavior (limit cycle).

How to build a limit-cycle oscillator?

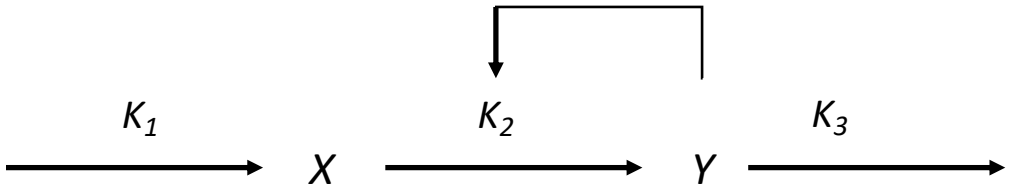
$$\xrightarrow{K_1} \quad X \quad \xrightarrow{K_2} \quad Y \quad \xrightarrow{K_3}$$

Law of Mass Action

$$\begin{cases} \frac{dX}{dt} = k_1 - k_2 X \\ \\ \frac{dY}{dt} = k_2 X - k_3 Y \end{cases}$$

How to build a limit-cycle oscillator?

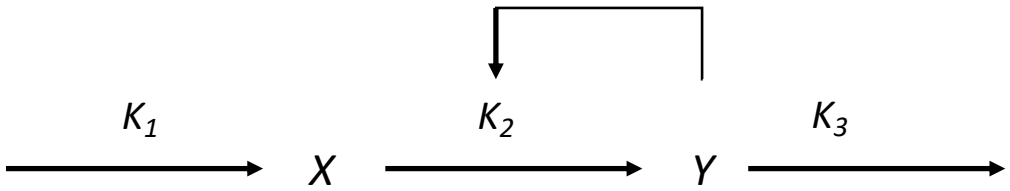
Sustained limit cycle behaviours are generated from two necessary ingredients: feedback loops and nonlinearity.



$$\begin{cases} \frac{dX}{dt} = k_1 - k_2 X Y \\ \frac{dY}{dt} = k_2 X Y - k_3 Y \end{cases}$$

How to build a limit-cycle oscillator?

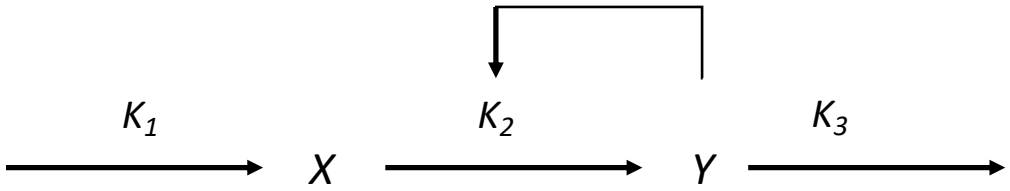
Sustained limit cycle behaviours are generated from two necessary ingredients: feedback loops and nonlinearity.



$$\begin{cases} \frac{dX}{dt} = k_1 - k_2 X f(Y) \\ \frac{dY}{dt} = k_2 X f(Y) - k_3 Y \end{cases}$$

How to build a limit-cycle oscillator?

Sustained limit cycle behaviours are generated from two necessary ingredients: feedback loops and nonlinearity.



$$\begin{cases} \frac{dX}{dt} = k_1 - k_2 X(1+Y)^n \\ \frac{dY}{dt} = k_2 X(1+Y)^n - k_3 Y \end{cases}$$

How to build a limit-cycle oscillator?

$$\begin{cases} \frac{dX}{dt} = k_1 - k_2 X(1+Y)^n \\ \frac{dY}{dt} = k_2 X(1+Y)^n - k_3 Y \end{cases}$$

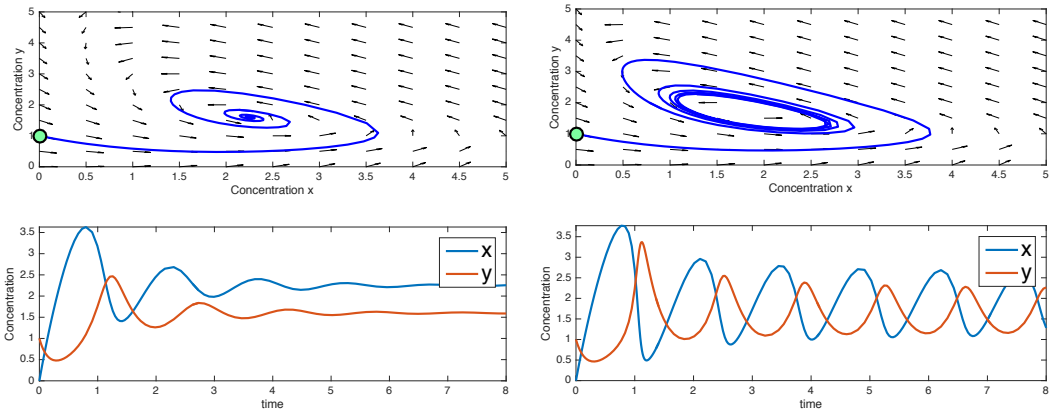


Figure: Left: $n = 2$ (damped oscillations). Right: $n = 2.5$ (limit cycle)

Gene regulatory network

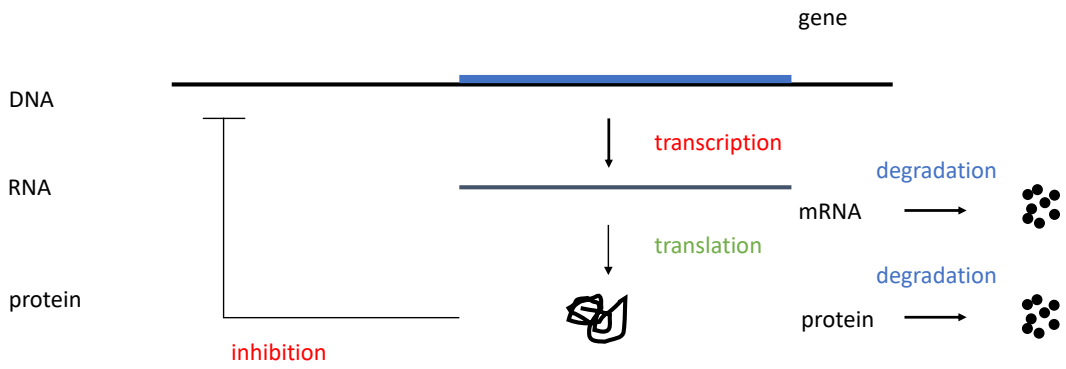
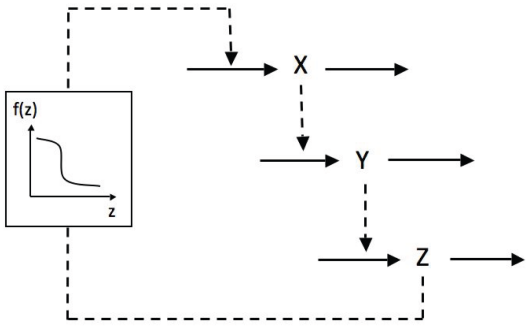


Figure: Transcription of the gene results in the formation of mRNA molecules, which can then be translated by ribosomes to produce proteins. These production processes are balanced by degradation of mRNA and protein molecules.

Goodwin model (1968)

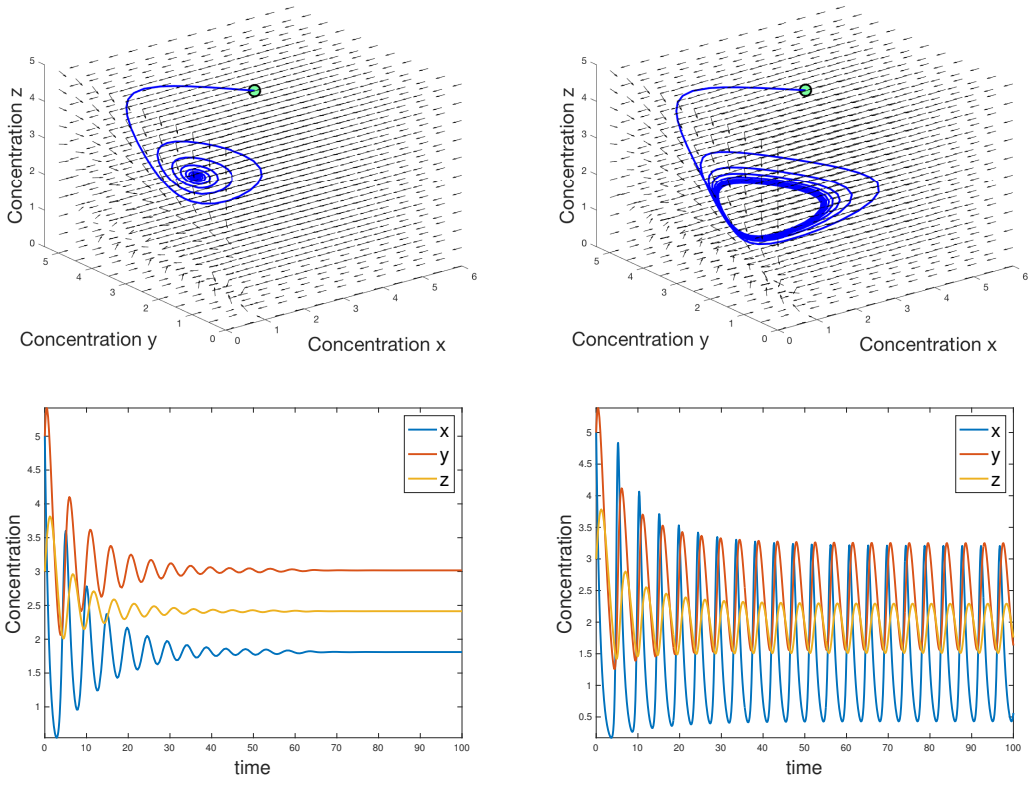


$$\begin{cases} \frac{dX}{dt} = k_1 f(Z) - k_2 X \\ \frac{dY}{dt} = k_3 X - k_4 Y \\ \frac{dZ}{dt} = k_5 Y - k_6 Z \end{cases}$$

$$f(Z) = \frac{1}{1+Z^n} \qquad \text{(Hill function)}$$

Figure: Goodwin model and Hill function.

Goodwin model (1968)



Homework #8: Arabidopsis thaliana (simplified model)

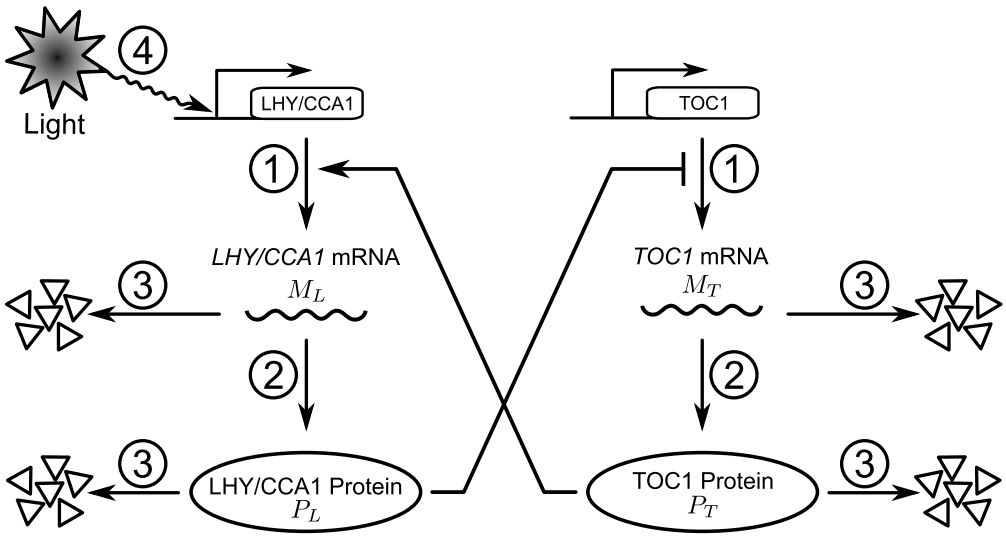


Figure: Schematic representation of our simple two-gene negative feedback loop model. Numbers indicate biochemical reactions: (1) transcription; (2) translation; (3) degradation; (4) light input.

Homework #8: Arabidopsis thaliana (simplified model)

Consider:

- the transcription factors TOC and lump together $LHY/CCA1$.
- only the mRNA concentration (M_L and M_T) and protein concentration (P_L and P_T).

The temporal evolution of the dynamical variables is given by:

$$\frac{dM_L}{dt} = L(t) + v_1 \frac{P_T^2}{a^2 + P_T^2} - \frac{d_1 M_L}{k_1 + M_L}$$
$$\frac{dP_L}{dt} = p_1 M_L - \frac{d_2 P_L}{k_2 + P_L}$$
$$\frac{dM_T}{dt} = v_2 \frac{b^2}{b^2 + P_L^2} - \frac{d_3 M_T}{k_3 + M_T}$$
$$\frac{dP_T}{dt} = p_2 M_T - \frac{d_4 P_T}{k_4 + P_T}$$

Color code: transcription; translation; degradation; light input.

Initial conditions: $M_L(0) = 0.1$, $P_L(0) = 0.5$, $M_T(0) = 0.1$, $P_T(0) = 0.1$.

Homework #8: Arabidopsis thaliana (simplified model)

1. Considering $v_1 = 0.3$, $a = 0.5$, $d_1 = 0.4$, $k_1 = 1$, $p_1 = 0.5$, $d_2 = 0.6$, $k_2 = 0.5$, $v_2 = 0.6$, $b = 0.1$, $d_3 = 0.6$, $k_3 = 1$, $p_2 = 0.3$, $d_4 = 0.3$, $k_4 = 1$, simulate the time evolution of the four dynamical variables, as well as the limit-cycle oscillations plotted in the $M_L - P_L$ and $M_T - P_T$ phase spaces.
   Note: for the function $L(t)$ use
   ```
   amp = 0.5; php = 0.5; per = 24;
   tm = mod(t,per); tmtest = per*(1-php)-tm;
   F = heaviside(tmtest); L =
   amp*(tmtest>0).*exp(-tm);
   ```
   where `per` is the photoperiod and `php` the percentage of light during a day period.
2. Change the parameters of light (`php` and/or `amp`) and analyse the behaviour of the dynamical system.
3. Simulate what happens when a mutation occurs in TOC or $LHY/CCA1$ (as you wish) that affect the transcription (v_1 or v_2) or the translation (p_1 or p_2).
4. Take a look on: http://www.ebi.ac.uk/biomodels/.