Differential Equations - Assignment

MME - Bioinformatics

Benjamin Ramberger

0. Naming the files

Please name your .py files in the following way and don't use special characters (like ö,ß,ñ,á):

assignment_assignment#_exercise#_lastname.py

e.g.: assignment _2_1_fuesschen.py for assignment 2, exercise 1 of student Füßchen

Submit one file for each exercise!

1. IVP Solver

Write a single python module that:

A) can solve the general initial value problem (IVP):

$$\boldsymbol{y}' = \boldsymbol{f}(t, \boldsymbol{y})\ ;\ \boldsymbol{y}(t_0) = \boldsymbol{y}_0$$

using the explicit 4-th order Runge-Kutta method (RK4). (Note: **y**, $\mathbf{y_0}$ and **f** are vectors!)

B) solves the following concrete IVP in the interval [0,10] with your RK4 implementation:

$$y'' = -4y - y'$$

$$y(0) = 1;\ y'(0) = -1$$

and compares the result with the RK45 solution obtained from `scipy.integrate.solve_ivp`.[1]

The module should contain the following:

- A function `RK4_step(t,y,dt,f)` that performs a single RK4 integration step and returns the next time and (approximate) function values (t_next,y_next).

  $$RK4\_step\ (t_n, y_n, dt, f) \rightarrow\ return\ (t_{n+1}, y_{n+1})$$

  The input arguments are:
  - `t`: current time (scalar)
  - `y`: current function values (vector)
  - `dt`: time step (scalar)
  - `f`: function to calculate the derivative (function).

  Note: it should work for an arbitrary function

  $$f: \mathbb{R} \times \mathbb{R}^n \rightarrow \mathbb{R}^n; (t, \boldsymbol{y}) \rightarrow \boldsymbol{f}(t, \boldsymbol{y})$$
- A function `RK4integrator(y0,f,N,t0,dt)` that performs:
  - `N` integration steps
  - of length `dt`
  - starting at initial conditions `(t0,y0)`

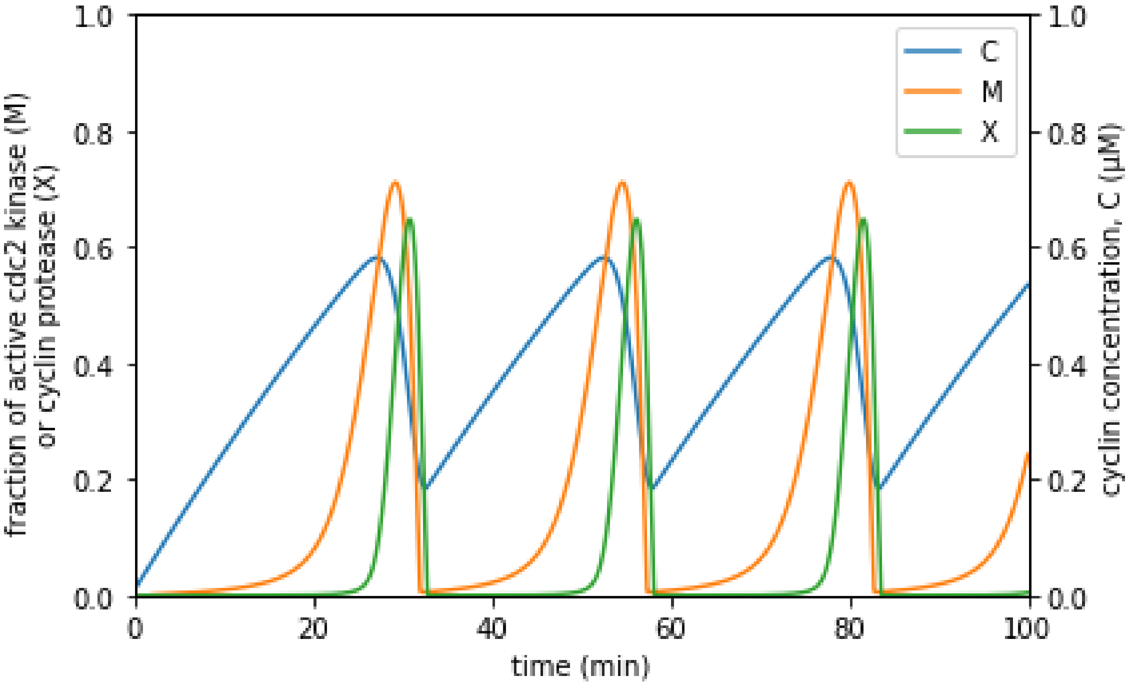
  by calling the `RK4_step` function N-times. It should return (**t**,**Y**):
  - `t`: an array containing the time grid
  - `Y`: an array containing the numerical solution to the general IVP.

  $$Y[n,:] = \boldsymbol{y}_n$$
- A function `test_integration`() for task B. It should plot both solutions in a single figure.

[1] RK45 (Dormand-Prince [2], [3]) is the default method of `scipy.integrate.solve_ivp`.
https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.solve_ivp.html

2. Minimal Cascade Model

Write a python module that reproduces figure 3 of reference [1]. The output could look like this:



[1] A. Goldbeter, “A minimal cascade model for the mitotic oscillator involving cyclin and cdc2 kinase.,” *Proc. Natl. Acad. Sci.*, vol. 88, no. 20, pp. 9107–9111, Oct. 1991, doi: 10.1073/pnas.88.20.9107.

[2] J. R. Dormand and P. J. Prince, “A family of embedded Runge-Kutta formulae,” *J. Comput. Appl. Math.*, vol. 6, no. 1, pp. 19–26, Mar. 1980, doi: 10.1016/0771-050X(80)90013-3.

[3] L. F. Shampine, “Some Practical Runge-Kutta Formulas,” *Math. Comput.*, vol. 46, no. 173, p. 135, Jan. 1986, doi: 10.2307/2008219.