Computing Methods for Physics – 12 May 2022

Your exam material (code files, plots, datafiles, etc.) must be submitted via google classroom by 13:00 as a single `zip` file.

C++ evaluation will be based on: correct syntax, proper return types, proper arguments of functions, data members and class interfaces, comments throughout the code, separation of class implementations and interfaces.

Python evaluation will be based on: correct syntax, avoiding C-style loops, using Python features in general, comments throughout the notebook/scripts, labels, legends and plot styling and clarity in general.

Part 1 – Matrices in C++

Implement a C++ class to represent and manipulate real matrices.

The class attributes must include: `Ncols` and `Nrows`, i.e., the number of columns and rows of the matrix, and `els`, i.e., the two dimensional array of individual matrix elements.

The methods to implement are the following.

- A default constructor and a copy constructor.
- Constructors for: a square matrix with known `Ncols`=`Nrows` and unknown `els`, a rectangular matrix with known `Ncols` and `Nrows` and unknown `els`, a diagonal matrix with known elements on the diagonal, a matrix with known elements and shape.
- A destructor.
- Getters and setters for all attributes.
- Getters and setters for a specific row, column, element.
- Methods to swap two rows, to swap two columns, to swap two elements.
- A method to transpose a matrix.
- A `Print()` method.

Finally, overload the plus (+), minus (-), and times (*) operators appropriately to operate between pairs of instances of the class.

Your submitted material must include a file `app.cpp` that showcases each method you implemented.

Part 2 – Python

The following system of first-order nonlinear differential equations describes the dynamics of a biological system composed of R rabbits and F foxes:

$$\begin{cases} \dfrac{dR}{dt} = \alpha R - \beta R F \\ \dfrac{dF}{dt} = -\gamma F + \delta R F \end{cases},$$

where t represents time and $\alpha, \beta, \gamma, \delta \in \mathbb{R}^{+}$. $\frac{dR}{dt}$ and $\frac{dF}{dt}$ are therefore the instantaneous growth rates of the population of rabbits and foxes, respectively.

Use a Python notebook or Python scripts to complete the following tasks.

1. Read the values of $\alpha, \beta, \gamma, \delta, R(t=0), F(t=0)$ provided by the user and simulate the evolution of the system for a time `T` also provided by the user. Provide the ability to store t, $R(t)$ and $F(t)$ to file with a sampling time `dt` provided by the user.

2. Provide the ability to display the results of a simulation with the following graphs: R and F versus time, $\frac{dR}{dt}$ and $\frac{dF}{dt}$ versus time, $\frac{dR}{dt}$ versus R, $\frac{dF}{dt}$ versus F, and R versus F.

3. Plot the quantity $\mathcal{I} = \delta R - \gamma \ln R + \beta F - \alpha \ln F$ as a function of time for 5 different simulations. You are free to pick the combinations you want for the parameters $\alpha, \beta, \gamma, \delta, R(t=0), F(t=0)$, but keep `T` fix across the simulations.

4. Set $\alpha = 1$, $\beta = 1$, $\gamma = 1$, $\delta = 1$, and $F(t=0) = 2$, and run simulations for $R(t=0)$ values between 2 and 20 with an incremental step of 2. Show the results of your 10 simulations in a single R versus F plot.