Homework 8 (25 points)

Numerical integration of ODEs

March 3, 2026

1 HW8 contents

This file comes zipped with:

1. Simple C++ source code (ode.cpp) for numerical integration of the equation of harmonically excited nonlinearized damped pendulum,

$$\ddot{\theta} + \gamma\dot{\theta} + \sin\theta = A\cos\omega t,$$

which has been transformed into the following system of two first-order equations:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = A\cos\omega t - \gamma x_2 - \sin x_1 \end{cases}$$

by the substitution $x_1 := \theta$ and $x_2 := \dot{\theta}$.

All the options (damping γ, excitation parameters A and ω, integration interval, time step, initial conditions) are hard-coded, so you have to recompile after each modification. The Euler method is already implemented by the function

```
void printEuler(T x1, T x2, T t0, int n, T gamma, T a, T omega);
```

The arguments are explained in the source code. The data type used in computations (**double** or **float**, aliased by T) can be changed by (un)commenting the corresponding line (no 7 or 8).

The program prints the results on the screen. To store the results in a text file named filename.txt, redirect the output: go to the command line, switch to the folder with the executable ode.exe and issue the command

```
ode >filename.txt
```

2. A simple script plot.gnu for generating plots using the gnuplot (`http://www.gnuplot.info/`), which is a very nice plotting package (you can also use any other plotting software, if you wish).

3. An example data set computed and plotted for a *linear pendulum* with the Euler method and **float** datatype at time steps 10^{-2} s, 10^{-3} s, 10^{-4} s and 10^{-5} s. A corresponding plot can be generated and shown on the screen with gnuplot by:

   (a) running gnuplot (wgnuplot.exe) (the font can be changed to a more readable by right-clicking and selecting Choose font ...),

   (b) going to the folder, where the script file plot.gnu is located by e.g.

   cd 'D:\PNO\HW5'

   (note the single quotation marks). The current folder can be checked by issuing withing the gnuplot the command pwd.

   (c) editing the script plot.gnu, if necessary,

   (d) issuing in gnuplot the command **load** 'plot.gnu'.

   The script for generating plots should be self-evident. To generate a PNG image file, instead of showing it on the screen, issue in gnuplot the following commands:

```
set term png
set output 'filename.png'
load 'plot.txt'
unset output
```

   To revert to the screen display use

```
set term windows
```

2 Your tasks

1. (10 points) Fill the body of the function printRK2(), which should be similar to printEuler(), but implement the 2nd order Runge-Kutta method instead of the Euler method.

2. (6) points total) Compute four numerical solutions (for the nonlinearized pendulum) in the time interval $[0, 50]$ s using the Euler method with **float** datatype and with the time step sizes 10^{-2} s, 10^{-3} s, 10^{-4} s and 10^{-5} s. Plot the computed solutions.

   (a) (4 points) Which solution seems to be the (more or less) exact solution? Why the solutions computed at the shortest and the longest time step sizes differ so much from the other two solutions?

   (b) (2 points) Recompute and replot the solutions using the same time interval, time step sizes and

      i. Euler method and **double** datatype,

      ii. 2nd order Runge-Kutta method and **float** datatype,

iii. 2^{nd} order Runge-Kutta method and **double** datatype.

Comment on the accuracy of the results.

3. (4 points) 2^{nd} order Runge-Kutta method combined with **double** data type seems — as expected — to be the most reliable from the tested methods, at least in the used time interval of $[0, 50]$ s. How far (how many seconds) can you more or less trust this solution at step size 10^{-2} s: $\ldots [0, 75]$ s? $\ldots [0, 100]$ s? $\ldots$ even more? How can you know it?

4. (5 points) Rework the code for the Euler method to tackle the one variable equation $\dot{x} = -\frac{1}{x}$. Start from $x(0) = 1$ and compute numerical solutions in the interval $[0, 2]$ at different step sizes. What happens and why? What is the exact (analytical) solution?

E-mail your answers, plots and the reworked source code to `ljank@ippt.pan.pl`.