Assignment 7.

Latest due date: Sunday, March 10th.

Note: make sure to please **include all figures and requested explanations** in the **PDF report!**

In this assignment you get a chance to experience important concepts such as numerical and theoretical **stability** and **convergence** (with changing step sise h) while implementing the implicit and explicit Euler methods.

Task 1. Program the explicit Euler method in order to approximate the solution of the equation $\dot{y}(t) = \lambda y(t)$ with initial condition $y(0) = 1$ and $\lambda \in \mathbb{R}$.

- Derive on paper the region of stability for this equation when $\lambda = 3$ and include it in your report.
- Check numerically the stability for $\lambda \in [-5, ..., 5]$ and appropriate values for h which will allow you to obtain conclusive stability results.
  Include in your report a table with your results for each set of λ and h you tried.

Task 2. The mathematical pendulum is described by the following second order ODE,

$$\begin{cases} \ddot{\alpha}(t) &= -\frac{g}{l}\sin\alpha(t), \quad \text{where} \quad g = 9.81 \ \text{ and } \ l = 1, \\ \dot{\alpha}(0) &= 0, \\ \alpha(0) &= \pi/2. \end{cases}$$

- Transform this equation into a system of first order ODEs and their initial conditions. Show **all** your work in the report.
- Solve the ODE above with MATLAB's `ode113`[1] solver in the time interval $[0, 10]$ with relative and absolute tolerance $1.e-3$ and $1.e-9$ respectively.
  Plot the solution and include the figure in your report.

Task 3. Write a numerical algorithm for the implicit Euler method with fixed point iteration as a corrector.

- Write in your report how exactly the implicit Euler method looks for this pendulum problem and also describe in detail which is the fixed point problem you will be solving.
- Use your code to solve the pendulum equation over the same time interval as in Task 2 above. Do this with different step sizes h and plot the resulting solutions. Can you explain from your plot, what is meant by *numerical damping*?

[1] Type ode113 in matlab help in order to see an example (at the very end of the help file) on how to do this as well as how to use the command odeset and set the tolerance options.