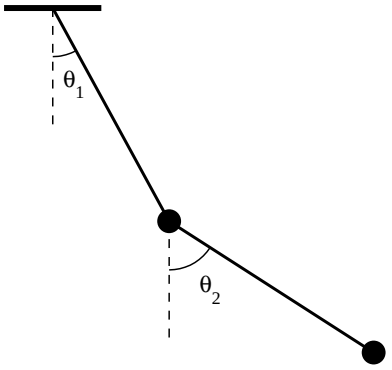
UCB Math 128A, Spring 2020: Programming Assignment 3

Due April 15

In this assignment, we will study the evolution of a double pendulum. The state of the configuration at any time t is given by the angles $\theta_1(t)$ and $\theta_2(t)$, see figure below.



The MATLAB utility `pendplot.m` on the course webpage can be used to visualize the double pendulum in any configuration. Here is an example of how to use it, to animate the motion $\theta_1(t) = 2\sin t$, $\theta_2(t) = \sin(2t)$:

```
for t = 0:0.01:10, pendplot(2*sin(t), sin(2*t)); end
```

This motion was of course just made up and does not correspond to a true, physical motion. Now we will solve for the actual evolution of the pendulum for various initial conditions!

1. Assuming that the lengths of the bars are 1, the masses at the end of the bars are 1, and that the constant of gravity is 1, the equations of motion for the double pendulum can be written:

$$\theta_1'' = \frac{-3\sin\theta_1 - \sin(\theta_1 - 2\theta_2) - 2\sin(\theta_1 - \theta_2)\cdot(\theta_2'^2 + \theta_1'^2\cos(\theta_1 - \theta_2))}{3 - \cos(2\theta_1 - 2\theta_2)} \tag{1}$$

$$\theta_2'' = \frac{2\sin(\theta_1 - \theta_2)(2\theta_1'^2 + 2\cos\theta_1 + \theta_2'^2\cos(\theta_1 - \theta_2))}{3 - \cos(2\theta_1 - 2\theta_2)} \tag{2}$$

   Rewrite (1),(2) into a system of 1st order equations by introducing the angular velocities $\omega_1 = \theta_1'$ and $\omega_2 = \theta_2'$. The current state of the pendulum can then be described by the vector $\boldsymbol{y} = (\theta_1, \theta_2, \omega_1, \omega_2)$, and the 1st order system can be written as $\dot{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{y})$. Write a MATLAB function `fpend.m` of the form

```
function ydot = fpend(y)
```

   which evaluates $\boldsymbol{f}(\boldsymbol{y})$.

   Your report should contain the 1st order system of equations and your MATLAB function `fpend.m`.

Turn page ⟶

2. Implement a fourth-order Runge-Kutta method that integrates the system $\dot{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{y})$ from $t = 0$ to $t = 100$ with stepsize $h = 0.05$. Use the four initial conditions described in the table below. You can put the command `pendplot(y(1),y(2));` at the end of each step to visualize the motion of the pendulum, which might be fun and might help you debug your code. You do not need to include these plots in your report.

Case	$\theta_1(0)$	$\theta_2(0)$	$\omega_1(0)$	$\omega_2(0)$
1	1	1	0	0
2	π	0	0	10^{-10}
3	2	2	0	0
4	2	$2 + 10^{-3}$	0	0

   For each case, plot the function $\theta_2(t)$ versus time. Cases 3 and 4 demonstrate the *initial value sensitivity* of the system, namely, that a small perturbation can lead to drastically different solutions.

   Your report should contain the MATLAB commands required to produce the results, and the four plots of $\theta_2(t)$.

3. Run Case 1 in problem 2 with the five stepsizes $h = 0.05/2^{(k-1)}$, $k = 1, 2, 3, 4$, and $h = 0.001$. Compute the value of $\theta_2(t = 100)$ for each stepsize. Consider the last result the exact solution, and plot the four errors as a function of h in a loglog-plot. Estimate the order of convergence from the slope.

   Your report should contain the MATLAB commands used, the five values of $\theta_2(t = 100)$, the loglog-plot, and the estimated slope.