Math 128A: Homework 7

Due: August 2

1. In this problem, 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` can be used to visualize the double pendulum in

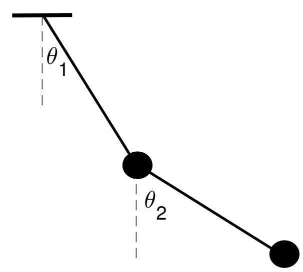

Figure 1: The double pendulum

any configuration. Here is an example of how to use it, to animate the motion $\theta_1(t) = 2\sin(t), \theta_2(t) = \sin(2t)$:

$$\texttt{for } \texttt{t} = 0 : 0.01 : 10, \texttt{pendplot}(2 * \texttt{sin}(\texttt{t}), \texttt{sin}(2 * \texttt{t})); \texttt{ end}$$

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

(a) 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 as

$$\begin{aligned}
\theta_1''(t) &= \frac{-3\sin(\theta_1) - \sin(\theta_1 - 2\theta_2) - 2\sin(\theta_1 - \theta_2)\cdot(\theta_2'^2 + \theta_1'^2\cos(\theta_1 - \theta_2))}{3 - \cos(2\theta_1 - 2\theta_2)} \\
\theta_2''(t) &= \frac{2\sin(\theta_1 - \theta_2)\cdot(2\theta_1'^2 + 2\cos(\theta_1) + \theta_2'^2\cos(\theta_1 - \theta_2))}{3 - \cos(2\theta_1 - 2\theta_2)} \qquad (1)
\end{aligned}$$

Rewrite (1) 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 $\mathbf{z} = (\theta_1, \theta_2, \omega_1, \omega_2)$, and the 1st order system can be written as $\mathbf{z}'(t) = \mathbf{f}(\mathbf{z})$. Write a MATLAB function `fpend.m` of the form

```
function zprime = fpend(~,z)
```

which evaluates $\mathbf{f}(\mathbf{z})$. The tilde in the first argument is because this function is autonomous, i.e., does not depend explicitly on time, but all the numerical solvers we have written evaluate the function at some t anyway.

(b) Use the fourth-order Runge-Kutta method to solve the system $\mathbf{z}'(t) = \mathbf{f}(\mathbf{z})$ from $t = 0$ to $t = 100$ with step-size $h = 0.05$. Use the four initial conditions described in the table below. You can put the command `pendplot(z(1),z(2));` at the end of each time-step to visualize the motion of the pendulum, which might be fun.

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.

(c) Run Case 1 in (b) with the five stepsizes $h = 0.05/2^{(k-1)}$ where $k = 1, 2, 3, 4$, and $h = 0.001$. Compute the value of $\theta_2(100)$ for each step-size. 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.

2. For the following multi-step methods, find the local truncation error in the form $O(h^k)$ and determine if they are convergent.

   (a) $2u_{i+3} = -3u_{i+2} + 6u_{i+1} - u_i + 6hf(t_{i+2}, u_{i+2})$

   (b) $3u_{i+4} = 3u_i + 4h[2f(t_{i+3}, u_{i+3}) - f(t_{i+2}, u_{i+2}) + 2f(t_{i+1}, u_{i+1})]$

   (c) $3u_{i+2} = 4u_{i+1} - u_i + 2hf(t_{i+1}, u_{i+1})$

3. (a) Edit the `AdBa3.m` function so that it's able to solve systems of differential equations.

   (b) Consider the Rabbit-Fox problem

$$\begin{aligned} R'(t) &= 2R(t) - R(t)F(t) \\ F'(t) &= 2R(t)F(t) - 2F(t) \end{aligned}$$

   over $0 \leq t \leq 20$ with $R(0) = 3$, $F(0) = 2$. Solve this problem using the function you wrote in (a) with step-sizes $h = 20/2^k$ for $k = 8, 9, \ldots, 16$ and estimate the order of convergence as in 1(c) using both the $R(t)$ and $F(t)$ values.

4. Consider the identity

$$y(t_{i+2}) = y(t_i) + \int_{t_i}^{t_{i+2}} f(t, y(t))\, dt.$$

   (a) Derive a multi-step method by replacing $f(t, y(t))$ by the interpolating polynomial for the points (t_i, f_i) and (t_{i+1}, f_{i+1}).

(b) Determine whether this method converges and, if it does, find its order of convergence.

5. (a) Write a function

```
function [u,T] = BackEuler(f,df,a,b,N,ya,maxiter,tol)
```

that implements the Backward Euler method for a scalar differential equation $y'(t) = f(t, y)$. Here, `N` is the number of steps in the discretization and `df` refers to the function $f_y(t, y)$. To calculate u_{i+1} at each step, run Newton's method with u_i as the initial guess. Perform at most `maxiter` number of iterations at each step and terminate early if successive values are less than `tol`.

(b) Solve the combustion model equation

$$y'(t) = y^2(1 - y), \quad 0 \leq t \leq 2000, \quad y(0) = 0.9$$

using your code in (a) with `maxiter` $= 20$, `tol` $= 10^{-12}$ and `N` $= 5, 10, 20$. Plot the results on the same axes.