MATLAB Assignment 9

When working on this assignment you might want to take a look at MATLAB code that was developed by students who took ME451 in previous years. The students back then did not come up with identical solutions. Take a look at their solutions and develop your own.
2010: http://sbel.wisc.edu/Courses/ME451/2010/SimEngine2D/index.htm
2011: http://sbel.wisc.edu/Courses/ME451/2011/SimEngine2D/index.htm

Turning in your assignment: place all your files in a directory called "lastName_Matlab_09", zip that directory, and upload the resulting file "lastName_Matlab_09.zip" in the appropriate Dropbox Folder at Learn@UW.

Problem 1. Consider the following Initial Value Problem:

$$\begin{cases} \dot{y} & = & \sin(y) \\ y(0) & = & 1.0 \end{cases}$$

to be solved on the interval $t \in [0, 10]$.

1. Solve the problem using Matlab's `ode45` function with a relative tolerance of 10^{-10} and produce the solution on a regular time grid with step-size $h = 0.01$.

   Hints

   - To specify the integration tolerance, use the Matlab `odeset` function to create an OPTIONS structure which is passed to `ode45`.
   - Use `ode45`'s ability to produce solution at specific times by specifying its 2nd argument as `(0:0.01:10)`.
   - The call to `ode45` should therefore be something like:

   ```
   [t,y] = ode45(myfun, (0:0.01:10), y0, odeset('Reltol',1e-10));
   ```

2. Use Forward Euler (FE) to solve the IVP using a step-size $h = 0.01$.

   Compare the solution obtained with FE to the one obtained with `ode45`. More precisely, generate and **upload to the class forum** a plot that displays the difference between the two solutions as a function of time.

3. Use Backward Euler (BE) to solve the IVP using a step-size $h = 0.01$.

   Compare the solution obtained with BE to the one obtained with `ode45`. More precisely, generate and **upload to the class forum** a plot that displays the difference between the two solutions as a function of time.

   Remark: Recall that BE is an implicit integration formula and you'll have solve a nonlinear system at each time-step.

4. **(Bonus)** Use the Runge-Kutta RK4 method to solve the IVP using a step-size $h = 0.01$.

   Compare the solution obtained with RK4 to the one obtained with `ode45`. More precisely, generate and **upload to the class forum** a plot that displays the difference between the two solutions as a function of time.

   Note: See Problem 2 below for details on RK4.

5. **(Bonus)** Use the 2nd order BDF formula (see lecture slides) to solve the IVP using a step-size $h = 0.01$.

   Compare the solution obtained with BDF2 to the one obtained with `ode45`. More precisely, generate and **upload to the class forum** a plot that displays the difference between the two solutions as a function of time.

   Notes:

   - The BDF2 formula is:
   $$y_{n+2} - \frac{4}{3}y_{n+1} + \frac{1}{3}y_n = \frac{2}{3}hf(t_{n+2}, y_{n+2})$$
   Attention: BDF2 is an implicit integratin formula!
   - In order to produce the approximate solution y_{n+2} at time t_{n+2}, you need to approximations y_{n+1} and y_n which are assumed to have already been obtained. Therefore, you can only apply the BDF method to get y_2, y_3, y_4, etc. In other words, you cannot use BDF2 to obtain the approximate solution y_1 at $t_1 = h$ because you do not have y_{-1}. This being the case, use the RK4 method to get y_1, and then start using BDF for the rest of the problem. This is one small drawback of BDF, it needs to be "primed" to start.

Problem 2. **(Bonus)** The goal of this problem is to perform an "order analysis". Basically, what you want to do is to verify that the order of accuracy of Forward Euler is 1 and that the order of accuracy of the Runge-Kutta RK4 method provided below is 4.

In this exercise, use the following IVP:

$$\begin{cases} \dot{y} &= -0.1y\sin(y) \\ y(0) &= 0.1 \end{cases}$$

and consider the approximate solution (using either FE or RK4) at the final time $t = 1.5$.

For the Forward Euler, start with a step size $h = 0.1$, then use $h = 0.01$, $h = 0.001$, and $h = 0.0001$ and calculate the corresponding approximate solutions $y_{FE}(1.5)$.

In the analysis for RK4, since this is a more accurate formula, you can use larger step-sizes, so please use $h = 0.25$, $h = 0.3$, $h = 0.5$, and $h = 0.75$.

Generate and **upload to the forum** the two resulting *convergence plots*. These are log-log plots with the value of the step-size on the x-axis and the value of the numerical error on the y-axis.

Notes:

- How do you obtain the numerical error? In theory, you should use $|y_{exact}(1.5) - y_{approx}(1.5)|$ to get the error at the end of the interval, where y_{approx} is the approximate solution obtained using one of the two integration formulas and a certain step-size. However, you don't have $y_{exact}(1.5)$. To get a good estimate for this value, one method is to also use a numerical solution but calculated with very high accuracy. For this, you you will have to use Matlab's `ode45` with a very tight tolerance ('RelTol' $= 10^{-10}$) so that you get something that comes close to $y_{exact}(1.5)$. We will call this solution (obtained with the tight tolerance) the "reference" solution and denote it $\hat{y}_{exact}(1.5)$. Then, you can estimate the numerical error for a particular integration formula (FE or RK4) using a particular step-size h by calculating $|\hat{y}_{exact}(1.5) - y_{approx}(1.5)|$.

- Recall that you should use a **log-log** plot (in Matlab, instead of using the plot function, simply use the function loglog). If everything works well, you should see a straight line with slope 1.0 for Forward Euler, and slope 4.0 for Runge-Kutta. These are the orders of the two methods: 1 and 4, respectively.

- Finally, the Runge-Kutta RK4 approximation y_{n+1} of the actual solution $y(t_{n+1})$ is computed as:

$$\begin{aligned} y_{n+1} &= y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\ t_{n+1} &= t_n + h \end{aligned}$$

  where h is the integration step-size and k_1 through k_4 are obtained as

$$\begin{aligned} k_1 &= f(t_n, y_n) \\ k_2 &= f(t_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}hk_1) \\ k_3 &= f(t_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}hk_2) \\ k_4 &= f(t_n + h, y_n + hk_3) \end{aligned}$$

Problem 3. Consider the double pendulum in Fig.6.5.2 (page 233). Notice that a horizontal force is specified on the second body. Assume that this concentrated force always stays horizontal, and its magnitude is always 5 N. Although not shown, assume that a positive torque $n = 2.5\ N \cdot m$ is applied to body 1. Additionally, take $l_1 = l_2 = 1$, and $m_1 = m_2 = 2$ (all units are SI). The mass moment of inertia should be computed based on information in Table 6.1.1 (page 208). The bodies are moving in a gravitational field with $g = -9.81 m/s^2$. Use for this problem Cartesian generalized coordinates in conjunction with centroidal reference frames as shown in the textbook.

Use the doublePend.acf and doublePend.adm files that you generated for your previous MATLAB assignment to carry out Dynamics Analysis of this mechanism using your simEngine2D. To this end, you will have to use the Newmark integration formula discussed in class and solve the resulting nonlinear algebraic system obtained at each time step t_n upon discretization of the constrained equations of motion. For this problem,

1. Generate and **upload to the forum** a plot of the time evolution of the x coordinate of body 2.

2. Generate and **upload to the forum** a plot of the reaction force induced on body 2 by the presence of the revolute joint connecting bodies 1 and 2. When computing this force, take the point P in conjunction with which you report the reaction force to be the point where the joint is located on body 2.

3. Just like above, generate and **upload to the forum** a plot of the reaction torque induced on body 2 by the presence of the revolute joint connecting bodies 1 and 2. When computing this reaction torque, take the point P in conjunction with which you report the reaction torque to be the point where the joint is located on body 2.