MA 512 **Homework 7** **March 16, 2017**

1. a. (5 points) Show that the midpoint method $y_{n+1} = y_{n-1} + 2hf(t_n, y_n)$ is absolutely stable for $0 < h < 1$ on the undamped-oscillator ODE, written in system form as

$$y' \equiv \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}' = f(t, y) \equiv \begin{pmatrix} y_2 \\ -y_1 \end{pmatrix}. \qquad (\star)$$

Note: This is a theoretical exercise, not a numerical one. Suggestion: Find the system Jacobian f_y and apply results developed in class.

b. (10 points) Apply the midpoint method to approximately solve $(\star)$ over $[0, 100]$ with initial condition $y(0) = (1, 0)^T$. To begin, the method needs a solution value at $t_1 = h$ as well as at $t_0 = 0$. Use the forward-Euler method to obtain this. Use step-sizes $h = .1$, $.01$, and $.001$. Print out and hand in a table showing the step-sizes in the first column and the maximum errors in the second column. (The exact solution is $y(t) = (\cos(t), -\sin(t))^T$.)
Note: The method is second-order. You should see this reflected in the maximum errors as h is reduced.

2. (5 points) Download and run the M-file `absStabRegionBDFdemo.m`, which uses the boundary-locus method to plot the boundaries of the absolute-stability regions for the BDF methods of orders 1-6. Note: There is a `pause` statement following the plotting of each region, so you'll have to repeatedly hit `return` to see the regions for all six methods. Print out and hand in the figure that shows all six regions.

Remark: As you'll see, the absolute-stability regions of the methods of orders 3-6 exclude increasingly large parts of the left half plane. For methods of orders greater than six, the regions exclude parts of the negative real axis. Consequently, on many problems, the methods will not be absolutely stable for some step sizes. Because of this potential instability, the BDF methods of orders greater than six are not used.

3. (10 points) Consider one more time the general damped-oscillator IVP

$$y' \equiv \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}' = f(t, y) \equiv \begin{pmatrix} y_2 \\ -y_1 - Dy_2 \end{pmatrix}, \quad y(0) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Use MATLAB's `ode45`, `ode113`, and `ode15s` to solve this IVP from $t = 0$ to $t = 20$ with $D = .1,\ 1,\ 10,\ 100$. As in Homework 6, use `odeset` to set `RelTol` = `AbsTol`= 10^{-8}. For `ode113` and `ode15s`, also set the `Jacobian` option to the system Jacobian f_y. Also, for all three methods, set `Stats` to `on` in order to see the run statistics in each case. For `ode15s`, you may either use the default numerical-differentiation formula (NDF) methods or select the BDF methods by setting `BDF` to `on`. (Better yet, use both to see how the NDF and BDF methods compare on this problem.) For each value of D, create a table that shows the number of successful and unsuccessful steps and function evaluations for each method. Hand in your tables along with a brief commentary on the numbers that you see.

4. (10 points) This exercise introduces an ODE that models a very interesting chemical reaction and also provides an example of a problem that stiff methods solve with ease but on which non-stiff methods must labor at great length. The problem is the *Oregonator* problem. This was first developed at the University of Oregon; the term is a combination of *Oregon* and *oscillator*. This has been described as "the simplest realistic model of the chemical dynamics of the oscillatory ... Belousov-Zhabotinsky (BZ) reaction." (See `http://www.scholarpedia.org/article/Oregonator`.) The BZ reaction is a remarkable chemical reaction in which the components combine, separate, and re-combine in a periodic oscillation. (Imagine that!) You can find a number of movies of the BZ reaction on YouTube.

The Oregonator ODE is the following

$$\begin{aligned} y_1' &= 77.27[y_2 + y_1(1 - 8.375 \times 10^{-6} y_1 - y_2)] \\ y_2' &= \tfrac{1}{77.27}[y_3 - (1 + y_1)y_2] \\ y_3' &= 0.161(y_1 - y_3) \end{aligned}$$

Apply `ode15s` to approximately solve this ODE for $0 \le t \le 290$ and with initial conditions $y_1(0) = 1$, $y_2(0) = y_3(0) = 1.8$. Use an options structure that sets `Stats` to `on`. Plot the logs of the three solution components in 3D using `plot3`. Print out and hand in your plot and also the run statistics.

Remark: This isn't required, but you might like to see how `ode113` *performs on this problem. By all means try it, but you might want to go off and do something else while the code runs.*