MATH 4093/5093 Homework 9 Due Fri, 5/6/2011

Problem 1. This problem illustrates some issues with the stability of different methods for solving IVPs for ODEs. Some of these issues are discussed in Section 7.9 of Bradie's book, where you can also find references for further reading.

In this problem you will study the IVP

$$y' = -1000\, y - \mathrm{e}^{-t} \ , \qquad y(0) = 0 \ , \tag{1}$$

whose exact solution is

$$y_{\text{exact}}(t) = \frac{1}{999} \left(\mathrm{e}^{-1000\, t} - \mathrm{e}^{-t} \right) \ .$$

This is a tricky equation to solve numerically because it evolves with two markedly different time scales (i.e., times over which the solution changes significantly) – one of them of order 1 (corresponding to the term e^{-t}), the other of order $\frac{1}{1000}$ (corresponding to the term $\mathrm{e}^{-1000\, t}$). Equations that have vastly different time scales are called *stiff equations*.

A numerical method is said to be *absolutely stable* if it reproduces correctly the asymptotic (i.e., long-time) behavior of the exact solution of the ODE. Usually there is an upper bound on the step size h above which the asymptotic behavior of the exact and the approximate solutions do not match. For the second-order Taylor method applied to the IVP (1), if h is larger than $\frac{1}{500}$, the exponential behavior of the numerical solution is incorrect.

(a) Solve the IVP (1) on the interval $t \in [0, 0.025]$ by using the code `taylor2nd.m`, which is an implementation of the second-order Taylor method. Use step sizes h equal to 10^{-5} (which gives a result very close to the exact solution) and $\frac{1}{400} = 0.0025$. Plot the two graphs together with different symbols. Attach a printout of your graph. Discuss what you see in the light of the absolute stability of the second-order Taylor method.

The phenomenon observed in part (b) was not caused by the stiffness to the IVP (1), but by the properties of the numerical method used. Below you will study the problems caused by the stiffness of (1) by using two implicit first-order one-step methods – the *trapezoidal method* and the *backward Euler method*.

The trapezoidal method, implemented in the MATLAB code `ode_trap.m`, is given by

$$\frac{w_{i+1} - w_i}{h} = \frac{1}{2} \left[f(t_{i+1}, w_{i+1}) + f(t_i, w_i) \right] \ . \tag{2}$$

It has a very simple interpretation – the slope $\frac{w_{i+1} - w_i}{h}$ is taken to be equal to the average of the slopes at the "present point" (t_i, w_i) and the "future point" (t_{i+1}, w_{i+1}). Since it is implicit, the code implementing the method has to solve the nonlinear equation (3). To this end, it uses Newton method you have to supply not only the right-hand side $f(t, y)$ of the ODE, but also the first derivative $\frac{\partial f}{\partial y}(t, y)$ of the right-hand side with respect to the second variable (or the Jacobian of the right-hand side if you are solving a system of ODEs).

The backward Euler method, implemented in the MATLAB code `back_euler.m`, is given by

$$\frac{w_{i+1} - w_i}{h} = f(t_{i+1}, w_{i+1}) \ . \tag{3}$$

Its interpretation is transparent – the slope $\frac{w_{i+1}-w_i}{h}$ is taken to be equal to the slope at the "future point" (t_{i+1}, w_{i+1}). Since it is implicit, it is a little bit more difficult to implement than the ordinary Euler method. Its advantage, however, is in its very good stability properties, which make it appropriate for solving stiff equations.

(b) Solve the IVP (1) for $t \in [0, 0.05]$ by using the trapezoidal method with step sizes $h = \frac{1}{10000}$, $\frac{1}{400}$,, $\frac{1}{300}$, and $\frac{1}{200}$ (i.e., with $N = 500$, 20, 15, and 10); the solution with $h = \frac{1}{10000}$ plays the role of the exact solution in your graph. Plot all the four graphs on the same plot, using solid lines, circles connected by dashed lines, stars connected by dotted lines, and plusses connected by dot-dashed lines, respectively (in the MATLAB command `plot`, these symbols correspond to the third argument equal to `'-'`, `'--o'`, `':*'`, and `'-.+'`). Attach a printout of your MATLAB session and of your graph. Discuss what you observe.

(c) Do the same as in part (b), but by using the backward Euler methods. Compare the backward Euler method with the trapezoidal method.

You can check (but you do not have to do this here) that the Taylor methods and the Runge-Kutta methods have the same troubles with the IVP (1) as the trapezoidal method.

Problem 2. In this problem you will study the performance of the finite differences method for solving linear boundary value problems of the form

$$\begin{aligned} &u'' = p(x)u' + q(x)u + r(x) \ , \qquad x \in [a, b] \ , \\ &\alpha_1 u(a) + \alpha_2 u'(a) = \alpha_3 \ , \\ &\beta_1 u(b) + \beta_2 u'(b) = \beta_3 \ , \end{aligned}$$

where p, q, and r are functions and α_j and β_j $(j = 1, 2, 3)$ are constants. The numerical method is implemented in the MATLAB code `linfd.m`; this code and `tridiagonal.m` (which is called from `linfd.m`) can be downloaded from the class web-site; these two codes should be put in the same directory.

(a) The code `linfd.m` uses finite difference method to solve a linear boundary-value problem with Dirichlet, Neumann, or Robin boundary conditions (as explained in Section 8.2 of the book). Run this code to solve the linear Dirichlet BVP

$$\begin{aligned} &y'' = 4xy' + (1 - 4x^2)y + \mathrm{e}^{x^2} \ , \qquad x \in \left[0, \tfrac{\pi}{2}\right] \ , \\ &y(0) = 1 \ , \quad y\left(\tfrac{\pi}{2}\right) = 2\mathrm{e}^{\pi^2/4} \ , \end{aligned}$$

whose exact solution is

$$y_{\text{exact}}(x) = (1 + \sin x)\mathrm{e}^{x^2} \ .$$

Plot the exact and the approximate solution on the same graph for $N = 10$ uniformly-sized subintervals (i.e., taking the argument `n` in the code to be equal to 10); using lines for the exact solution and stars for the approximate solution. Do this separately also for $N = 100$ and $N = 1000$. Attach a printout of your MATLAB session and all graphs.

Here are some hints. Suppose that you want to run the code by dividing the interval $[a, b] = [0, \frac{\pi}{2}]$ into $N = 100$ subintervals of equal length, in which case you have to create a uniform grid of $(N+1)$ points x_j, $j = 0, 1, \ldots, N$. Then you want to compute the values of the exact solution at each of of the exact solution at each of the grid points x_j, $j = 0, 1, \ldots, N$. You can do this with the commands

```
N=100;
ti=linspace(0,pi/2,N+1);
exact=(1+sin(ti)).*exp(ti.^2);
```

Then you run the code `linfd.m` with $N = 100$, and create a vector `wi` of length $N + 1$ with the approximate values of the solution of the boundary value problem, as follows:

```
wi=linfd(@coeffs_linshoot1,0,pi/2,N,[1 0 1],[1 0 2*exp(pi^2/4)]);
```

(here it is assumed that the MATLAB function `coeffs_linshoot1.m` returns the values of the functions $p(x)$, $q(x)$ and $r(x)$).

(b) Solve the problem from part (a) with $N = 10$, 100, 1000 and 10000, and compute the norms $\|\texttt{exact} - \texttt{wi}\|_\infty$ of the absolute errors for each N (you can do this with the command `norm(exact-wi,Inf)`). What seems to be the order of convergence?

(c) Do the same as in part (b), but this time compute the root mean square (r.m.s.) error,

$$(\text{r.m.s. error}) = \sqrt{\frac{1}{N+1}\sum_{i=0}^{N}(w_i - y_i)^2} \ .$$

How does this error seems to decrease with the step size?

Problem 3. This problem is a continuation of Problem 2.

(a) Solve the boundary value problem

$$\begin{aligned} &y'' = 4xy' + (1 - 4x^2)y + \mathrm{e}^{x^2} \ , \qquad x \in \left[0, \tfrac{\pi}{2}\right] \ , \\ &y(0) = 1 \ , \quad \pi y\left(\tfrac{\pi}{2}\right) + y'\left(\tfrac{\pi}{2}\right) = 4\pi \mathrm{e}^{\pi^2/4} \ , \end{aligned}$$

and then the boundary value problem

$$y'' = 4xy' + (1 - 4x^2)y + \mathrm{e}^{x^2} \ , \qquad x \in \left[0, \tfrac{\pi}{2}\right] \ ,$$
$$y(0) = 1 \ , \quad \pi y\left(\tfrac{\pi}{2}\right) - y'\left(\tfrac{\pi}{2}\right) = 0 \ ;$$

in both cases use $N = 10000$. These two boundary value problems have the same solution as the one in Problem 2. For each of the two boundary value problems in this part plot both the exact and the numerical solution.

(b) Do the same as in part (a) but for the boundary value problems

$$y'' = 4xy' + (1 - 4x^2)y + \mathrm{e}^{x^2} \ , \qquad x \in \left[0, \tfrac{\pi}{2}\right] \ ,$$
$$y(0) = 1 \ , \quad \pi y\left(\tfrac{\pi}{2}\right) - y'\left(\tfrac{\pi}{2}\right) = 0.0001 \ ,$$

and

$$y'' = 4xy' + (1 - 4x^2)y + \mathrm{e}^{x^2} \ , \qquad x \in \left[0, \tfrac{\pi}{2}\right] \ ,$$
$$y(0) = 1 \ , \quad \pi y\left(\tfrac{\pi}{2}\right) - y'\left(\tfrac{\pi}{2}\right) = -0.0001 \ .$$

Attach the printouts of your MATLAB session and the four graphs (two for part (a) and two for (b)). What do you observe? What is the moral of your observations?

Problem 4. From the class web-site download the Matlab code `linshoot.m`. In this problem you will study this code and its performance only for the case of Dirichlet boundary value problems. In this problem you will solve the same BVP as in Problem 2, namely

$$y'' = 4xy' + (1 - 4x^2)y + \mathrm{e}^{x^2} \ , \qquad x \in \left[0, \tfrac{\pi}{2}\right] \ ,$$
$$y(0) = 1 \ , \quad y\left(\tfrac{\pi}{2}\right) = 2\mathrm{e}^{\pi^2/4} \ ,$$

but using a different method.

Note that the output of `linshoot.m` is not a vector, but a $2 \times (N + 1)$ matrix, the first column of which contains the approximate values of the solution evaluated at the nodes, and the second column contains the approximate values of the derivatives of the solution evaluated at the nodes. If you call the code like this:

```
wi = linshoot(@coeffs, 0.0, pi/2.0, 100, [1 0 1], [1 0 2*exp(pi^2/4)] );
```

then `wi` will be a 2×101 matrix. If you then want to use only the first row of `wi`, you can type `wi(1,:)`, which will be a vector of dimension 101.

(a) The general solution of the differential equation

$$y'' = 4xy' + (1 - 4x^2)y + \mathrm{e}^{x^2}$$

is $y_{\text{gen}}(t; C_1, C_2) = (1 + C_1 \cos x + C_2 \sin x)\, \mathrm{e}^{x^2}$ (the arbitrary constants are written as arguments). Imposing the "initial" condition $y(0) = 1$ implies that $C_1 = 0$, so that functions satisfying the ODE and the "initial" condition have the form

$$y_{\text{gen}}(t; 0, C_2) = (1 + C_2 \sin x)\, \mathrm{e}^{x^2} \ .$$

Define the function

$$F(\xi) = y_{\text{gen}}(\tfrac{\pi}{2}; 0, z) \ ,$$

and solve an appropriate equation for F in order to find the solution of the BVP. Explain what you do and write your calculations in detail.

(b) Study the behavior of `linshoot.m` on the example of this BVP, with $N = 10$, 100, 1000 and 10000, as in Problem 2(b).

(c) Study the behavior of `linshoot.m` on the example of this BVP, with $N = 10$, 100, 1000 and 10000, as in Problem 2(c).

Problem 5. Find the general solutions of the partial differential equations

(a) $u_x(x, y) = 2x \sin y$;

(b) $u_{xy}(x, y) = 0$;

(c) $u_{xyy}(x, y) = 3x^2\, \mathrm{e}^y$.

Do not forget that the general solution of a partial differential equation of order n for a function of d contains n arbitrary functions, each of which is a function of $(n - 1)$ variables. For example, the general solution of the PDE $y_{xxy}(x, y) = 6x \cos y$ is

$$u(x, y) = x^3 \sin y + x f(y) + g(x) + h(y) \ ,$$

for arbitrary functions f, g, and h of one variable.