Project assignment - MA2501

Spring 2010

Exercise 1

We consider the initial-value problem

$$\frac{\mathrm{d}x}{\mathrm{d}t} = f(t,x) = \frac{2x}{t+1} - \frac{x^2}{(t+1)^3}, \quad t \in [0,T], \quad x(0) = 1. \tag{1}$$

We will use three numerical methods to solve this problem:

1) Forward Euler

$$x_{n+1} = x_n + hf(t_n, x_n)$$

2) Modified Euler

$$x_{n+1} = x_n + hf\left(t_n + \frac{h}{2}, x_n + \frac{1}{2}hf(t_n, x_n)\right)$$

3) Explicit Runge-Kutta

$$\begin{aligned} k_1 &= hf(t_n, x_n) \\ k_2 &= hf\left(t_n + \frac{h}{2}, x_n + \frac{k_1}{2}\right) \\ k_3 &= hf\left(t_n + \frac{h}{2}, x_n + \frac{k_2}{2}\right) \\ k_4 &= hf(t_n + h, x_n + k_3) \\ x_{n+1} &= x_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{aligned}$$

a) Verify that (1) has exact solution $x(t) = \frac{(t+1)^2}{1+\ln(t+1)}$.

b) We know that Forward Euler is a first order method and that the given Runge-Kutta method is of order 4. Show analytically the order of modified Euler.

Implement the three methods in Matlab. You may for instance do this by writing three functions (one for each method), which take as input T and the number of steps, N. The stepsize is then given by $h = \frac{T}{N}$.

c) Choose a T and compute the error for the three methods with different choices of N. (Start for instance with $h = 0.5$ and repeatedly divide h by 2.) Plot the error as a function of the stepsize, h, and verify that the implementation behaves as expected for each method. (**Hint:** Use `loglog` in Matlab.)

d) Choose $T = 8$, and measure the error for the three methods with $h = 0.5$. Based on these values, estimate analytically the number of iterations necessary for each of the methods in order to get an error of 10^{-7}. How many function-evaluations does this involve for each method?

e) In Matlab you can start a clock with the command `tic` and stop with the command `toc`. Compare the elapsed time for the three methods when using the result from **d)** (this means use a stepsize such that all methods have an error about 10^{-7}). Also, compare the elapsed time when all methods use the number of steps. How does this compare to the result in **d)** if we assume that function-evaluations are dominating?

Exercise 2

We want to solve the problem

$$x'' - 2(1 - x^2)x' + x = 0, \qquad x(0) = 2,\, x'(0) = 0. \tag{2}$$

a) Rewrite this problem as a system of 1.order ordinary differential equations. What is the initial-value for this system?

To solve this system we will use the following numerical method:

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \frac{h}{2}(\mathbf{f}(t_n, \mathbf{x}_n) + \mathbf{f}(t_{n+1}, \mathbf{x}_{n+1})). \tag{3}$$

The error of this method is $\mathcal{O}(h^2)$, and this is an example of an *implicit* method since $\mathbf{f}$ is evaluated in $\mathbf{x}_{n+1}$ at every timestep. However, $\mathbf{x}_{n+1}$ is the vector that we want to compute, and this means that we need to solve a (in this case) non-linear system-of-equations at each time-step. We know that Newton's method for systems is a good alternative for solving such problems.

b) What is the non-linear system that needs to be solved at a given time-step? What initial guess will you use for solving this problem?

We do not have a known exact solution for this problem. However, what we can do is to use one of Matlab's ODE-solvers, use this to solve the problem with high accuracy, and use this solution as the exact solution. This can be done in the following way:

```
options = odeset('RelTol',1e-10,'AbsTol',[1e-10 1e-10]);
[t,xe] = ode45(@f,[t_s t_e],[x1_0; x2_0],options);
```

Here, `options` indicates the accuracy we want the solution to have. `f` is the function describing the right-hand side of the system. This should be on the form:

```
function xd = f(t,x)
xd = zeros(2,1);
xd(1) = ...
xd(2) = ...
```

`@f` means a *function handle* for the function `f`. `t_s` and `t_e` is the initial and end-time, `x1_0` and `x2_0` are initialvalues for the system in **a)**. `ode45` returns `t` which is a column vector (which we do not know the length of) which contains values in $[\texttt{t\_s}, \texttt{t\_e}]$ where the solution is approximated. `xe` contains two column vectors (each with the same length as `t`) with approximated values for the corresponding times in `t`. The final element, which corresponds to the solution at time `t_e`, can be found from `xe(numel(xe(:,1)),1)`.

c) Implement the method (3) and solve the system (2) in the interval $[0, 20]$. Show that the implementation shows the expected error-behaviour.