Numerical Analysis

Computer Exercise 2

Our goal here is to solve a stiff, non-linear system of differential equations using a 4th order diagonally implicit Runge-Kutta method. The first step is to write a multi-dimensional solver which implements Newton's method:

$$x^{n+1} = x^n - J(x^n)^{-1} f(x^n), \quad J_{ij} = \frac{\partial f_i}{\partial x_j}.$$

Your program should be a function called my_newton. The function should have as input arguments f, J,x^0 and tol. The last argument is the tolerance below which you stop the iterations by checking the L^2 norm between two consecutive steps.

Q1: Let's first check the Newton solver on the four dimensional nonlinear system of equations

$$x_i = e^{\cos\left(i \sum_{j=1}^{4} x_j\right)}$$

Find the solution of this system equations by using Newtons method by defining appropriately the function f and its Jacobian J. What is a good way to chose a starting guess? Remember to chose a small value for the tolerance parameter.

Q2: Now consider the following stiff system of ODE's:

$$\begin{aligned} y_1' &= -0.04y_1 + y_2 y_3, \\ y_2' &= 400y_1 - 10000y_2 y_3 - 3000y_2^2, \ , 0 \le t \le 3 \\ y_3' &= 0.3y_2^2, \end{aligned} \tag{1}$$

with initial condition $y(0) = (1, 0, 0)^T$. You will solve this using both the Runge-Kutta-Fehlberg (RKF) method and a 4th order L-stable diagonally implicit Runge-Kutta method.

Let me first briefly describe the RKF method which aims in determining the optimal discretization step to be used in the solution of an Initial Value Problem (IVP). As we said in the course, one way to guarantee the accuracy when solving an IVP is to solve it twice using step sizes h and $h/2$ and then compare answers at the mesh points corresponding to the larger step size. This requires a significant amount of computation !!! The Runge-Kutta-Fehlberg method is one way to solve this problem. It has a procedure to determine if the proper step size h is being used. At each step, two different approximations for the solution are computed and compared. If the two answers are in close agreement, the approximation is accepted. If the two answers do not agree to a specified accuracy, the step size is reduced. If the answers agree to more significant digits than required, the step size is increased.

The method is defined as

$$k_i = f\left(w_n + h \sum_{j=1}^{i} a_{ij} k_j\right), \quad i = 1, \ldots, s \tag{2}$$

and

$$w_{n+1} = w_n + h \sum_{j=1}^{s} b_j k_j, \quad \tilde{w}_{n+1} = w_n + h \sum_{j=1}^{s} \tilde{b}_j k_j, \tag{3}$$

where the matrix A and the vectors b and $\tilde{b}$ are defined by

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1/4 & 0 & 0 & 0 & 0 & 0 \\ 3/32 & 9/32 & 0 & 0 & 0 & 0 \\ 1932/2197 & -7200/2197 & 7296/2197 & 0 & 0 & 0 \\ 439/216 & -8 & 3680/513 & -845/4104 & 0 & 0 \\ -8/27 & 2 & -3544/2565 & 1859/4104 & -11/40 & 0 \end{bmatrix}$$

,

$$b = \begin{pmatrix} 25/216 \\ 0 \\ 1408/2565 \\ 2197/4104 \\ -1/5 \\ 0 \end{pmatrix}, \tilde{b} = \begin{pmatrix} 16/135 \\ 0 \\ 6656/12825 \\ 28561/56430 \\ -9/50 \\ 2/55 \end{pmatrix}$$

The provided file rkf.m is an implementation of the Runge-Kutta-Fehlberg method for which $s = 6$.

Change appropriately the file rkf.m so as to implement the DIRK method defined by (2)-(3) with $s = 5$ and

$$A = \begin{bmatrix} 1/4 & 0 & 0 & 0 & 0 \\ 1/2 & 1/4 & 0 & 0 & 0 \\ 17/50 & -1/25 & 1/4 & 0 & 0 \\ 371/1360 & -137/2720 & 15/544 & 1/4 & 0 \\ 25/24 & -49/48 & 125/16 & -85/12 & 1/4 \end{bmatrix} \tag{4}$$

,

$$b = \begin{pmatrix} 25/24 \\ -49/48 \\ 125/16 \\ -85/12 \\ 1/4 \end{pmatrix}, \tilde{b} = \begin{pmatrix} 59/48 \\ -17/96 \\ 225/32 \\ -85/12 \\ 0 \end{pmatrix}$$

The main difference between RKF and this DIRK method is that the diagonal entries of A in (4) are not zero. This means that the equations that determine k_i are of the form

$$k_i = f(z + ha_{ii}k_i)$$

where $z = w_n + h\sum_{j=1}^{i-1} a_{ij}k_j$ is known. Thus if we define

$$g(k) = k - f(z + ha_{ii}k),$$

we can use Newton's method to solve for k. As a starting guess you can take $f(w_n)$.

Q2: Implement the DIRK method in a file called my_dirk.m. Note that the step-size control works the same for DIRK as it does for RKF, so you do not need to modify this part of the code.

Q3: Compare the results obtained by the two methods. How many iterations do you need by each method to provide the results with the same accuracy?

Q4: Determine numerically the order of accuracy of the two methods that are used in the RKF method. To do so, you need to solve a system of equations for which you know the exact solution. The order can be determined by looking at the rate at which the error goes to zero by successively refining h. To do this you have to write another version of the rkf where h is fixed.

You should submit all your files and your report by email by 18/12/2014 at 9pm. Late submissions will not be considered. All your codes and the report should be compressed in one file named EA2_xxx.tgz (or zip) where xxx is your AM. Include enough comments in the code to convince the reader that you understand the algorithms. Also comment and explain the results obtained in your report. Please name your report EA2_xxx.pdf and make directory (folder) that contains all your files. The directory should also named EA2_xxx. Then compress this directory and send it to me.