7 Solver RADAU

7.1 General information

Authors:	E. Hairer and G. Wanner
first version:	April 23, 1998
last update:	January 18, 2002
language:	Fortran 77
availability:	the code RADAU is freely available (in the public domain)
official link:	http://www.unige.ch/~hairer/prog/stiff/radau.f
problem type:	ODEs and DAEs of index less than or equal to 3
IVPtestset files:	solver: radau.f
	driver: radaud.f
	auxiliary files: radaua.f (auxiliary linear algebra routines)

7.2 Numerical method

The code RADAU is based on implicit Runge-Kutta methods (Radau IIa) of orders 5, 9 and 13. These methods are L-stable and were firstly implemented in fixed order mode in the code RADAUP [HW96]. It is written for problems of the form $My' = f(t, y)$ with a possibly singular matrix M. It is therefore also suitable for the solution of differential-algebraic problems.

7.3 Implementation details

All the implementation techniques described for RADAU5 hold here as well. The code has been provided with an order variation strategy. This is based upon the observation that high order methods perform better than low order methods as soon as the convergence of the simplified Newton iteration is sufficiently fast (a measure of the rate of convergence is the so called *contractivity factor*) [HW99].

7.4 How to solve test problems with RADAU

Compiling

```
f90  -o dotest  radaud.f  problem.f  radau.f  radaua.f  report.f,
```

will yield an executable `dotest` that solves the problem, of which the Fortran routines in the format described in Section IV.3 are in the file `problem.f`.

As an example, we perform a test run, in which we solve problem HIRES. Figure I.7.1 shows what one has to do.

References

[HW96] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-algebraic Problems.* Springer-Verlag, second revised edition, 1996.

[HW99] E. Hairer and G. Wanner. Stiff differential equations solved by radau methods. *J. Comput. Appl. Math.*, 111:93–111, 1999.

```
$ f90 -O5 -o dotest radaud.f hires.f radau.f radaua.f report.f
$ ./dotest

  Test Set for IVP Solvers (release 2.3)

  Solving Problem HIRES using RADAU

 User input:

 give relative error tolerance:
1d-4
 give absolute error tolerance:
1d-4
 give initial stepsize:
1d-4

 Numerical solution:

                                              scd
       solution component                -----------------------------     ignore
                                         mixed       abs        rel     mix - abs,rel
 ----------------------------------      -----      -----      -----    -------------
 y(  1) =  0.7485152484440879E-003        4.94       4.94       1.81
 y(  2) =  0.1464912389469645E-003        5.65       5.65       1.81
 y(  3) =  0.6101426280653334E-004        5.67       5.67       1.44
 y(  4) =  0.1196763210067838E-002        4.68       4.68       1.75
 y(  5) =  0.2731889907948499E-002        3.46       3.46       0.84
 y(  6) =  0.7347017643277632E-002        2.96       2.96       0.75
 y(  7) =  0.3074620885907540E-002        3.65       3.65       1.10
 y(  8) =  0.2625379114092413E-002        3.65       3.65       1.10

 used components for  scd                    8          8          8
 scd of Y (maximum norm)                  2.96       2.96       0.75


 using mixed error yields mescd           2.96
 using relative error yields scd                                 0.75


 Integration characteristics:

    number of integration steps           38
    number of accepted steps              31
    number of f evaluations              295
    number of Jacobian evaluations        20
    number of LU decompositions           37

 CPU-time used:                         0.0010 sec
```

FIGURE I.7.1: *Example of performing a test run, in which we solve problem HIRES with RADAU. The experiment was done on an ALPHAserver DS20E, with a 667MHz EV67 processor. We used the Fortran 90 compiler* `f90` *with the optimization flag* `-O5`.