Lecture 33 (hrs. 66,67) - December 10, 2025, 11:30 - 13:30 A13

(4.30) Remark (choice of h(k) in Runge-Kutta methods).

In line with the intent to eliminate the burden of determining and realizing the functions $G_j(t,x)$, the choice of h(k) in RK methods usually occurs as follows.

Let: RK be the Runge-Kutta method, of order p for $h \to 0$, chosen for the computation of x(k+1) and RK' another Runge-Kutta method, of order p' = p+1. Then:

- CHOICE of h(k). Given $E > 0$ and $\lambda > 0$, for each k we choose τ small, and we compute:

  (1) XX = one step using RK starting from (x(k),t(k)), of length τ
  (2) XX' = one step using RK' starting from (x(k),t(k)), of length τ

  set:

$$d(k) = \max \{ \lambda , \| XX - XX' \| \}$$

  and then:

$$h(k) = \min \{ \sqrt[p+1]{\frac{E}{d(k)}}\ \tau \ , tf - t(k) \}$$

This selection procedure is explained by considering that:

(a) The RK method has order p for $h \to 0$ so, once set $C = \frac{s^{(p+1)}(0)}{(p+1)!}$:

$$\textit{we estimate } s(h) \textit{ with } C\, h^{p+1}$$

(b) Since:

$$XX - y(t(k) + \tau) = C\, \tau^{p+1} + z(\tau)\, \tau^{p+1} \ , \quad \text{where } z(\tau) \to 0 \text{ as } \tau \to 0 \quad \text{(the order of RK is p)}$$

$$XX' - y(t(k) + \tau) = C'\, \tau^{p+2} + w(\tau)\tau^{p+2} \ , \quad \text{where } w(\tau) \to 0 \text{ as } \tau \to 0 \quad \text{(the order of RK' is p+1)}$$

then:

$$\frac{XX - XX'}{\tau^{p+1}} = C + [\ z(\tau) - (C' + w(\tau))\ \tau\ ] \to C \quad \text{as } \tau \to 0$$

and:

$$\textit{using a small value of } \tau \textit{ , we estimate } C \textit{ with } \frac{XX - XX'}{\tau^{p+1}}$$

(c) Overall:

$$\textit{using a small value of } \tau \textit{ , we estimate } s(h) \textit{ with } \frac{XX - XX'}{\tau^{p+1}}\ h^{p+1}$$

hence:

$$\left\| \frac{XX - XX'}{\tau^{p+1}}\ h^{p+1} \right\| = E \quad \Leftrightarrow \quad h = \sqrt[p+1]{\frac{E}{\| XX - XX' \|}}\ \tau$$

(4.31) *Scilab* realization (RK12_pv).

As an example of implementation, let us consider the RK method which uses explicit Euler, of order 1 for h → 0, to calculate x(k+1) and which chooses h(k) by combining it with the method of Remark (4.23), Heun's method of order 2 for h → 0. The result is a method of order 1 for h → 0 and therefore convergent of order 1/2 for E → 0.

```
01  function [T, X, PASSO] = RK12_pv(x0, t0, tf, F, E, LAMBDA, HMIN, TAU)
02  //
03  // Numerically integrates, on [t0,tf], the Cauchy Problem
04  // in R(n):
05  //
06  // x' = F(t,x)
07  // x(t0) = x0
08  //
09  // using explicit Euler method (order one RK) - with variable
10  // step - combined, to choose the step, with the RK method
11  // of order two defined by c(2) = 1, a(21) = 1 e b(1) = b(2) = 1/2.
12  //
13  // x0: initial condition (column of n elements)
14  // t0: initial time (real number)
15  // tf: final time (real number)
16  // F: function which defines the differential equation; F(t,x) should
17  //    be a column of n real numbers
18  // E: maximum value of the estimate of the local error (real number)
19  // LAMBDA: real number that sets the maximum value of the step
20  //         (OPTIONAL - predefined value: 1d-5)
21  // HMIN: minimal allowed value of the step
22  //       (OPTIONAL - predefined value: (tf - t0) / 1d6)
23  // TAU: value of the step to compute the estimates used
24  //      in the choice of h(k) (OPTIONAL - predefined value: (tf - t0) / 1d3)
25  //
26  // T = [t(0),...,t(N)]: row whose elements are the integration instants
27  // X = [x(0),...,x(N)]: matrix n x (N+1) whose columns are the approximations
28  // PASSO = [h(0),...,h(N-1)]: row whose elements are the integration step
29  //
30  // Values for the optional input parameters
31  //
32  if ~exists('LAMBDA','l') then LAMBDA = 1d-5; end;
33  if ~exists('HMIN','l') then HMIN = (tf - t0) / 1d6; end;
34  if ~exists('TAU','l') then TAU = (tf - t0) / 1d3; end;
35  //
36  // Initialization of the output variables
37  //
38  T(1,1) = t0;
39  X(:,1) = x0;
40  PASSO = [];
41  //
42  // main loop
43  //
44  while (T(1,$) < tf), // halt the construction if tf has been reached
45    //
```

```
46    // choice of the step
47    //
48    // XX1 = X(:,$) + F(T(1,$),X(:,$)) * TAU;
49    ST1 = F(T(1,$),X(:,$));
50    ST2 = F( T(1,$) + TAU,X(:,$) + ST1 * TAU );
51    // XX2 = X(:,$) + ( (ST1 + ST2)/2 ) * TAU;
52    //
53    //     XX1 - XX2 = (ST1 - ST2)/2 * TAU
54    //
55    d = max(LAMBDA, norm( ((ST1 - ST2)/2) * TAU ));
56    PASSO(1,$+1) = min(sqrt(E/d) * TAU, tf - T(1,$));
57    //
58    // computation of the approximation and of the new integration instant
59    //
60    X(:,$+1) = X(:,$) + F(T(1,$),X(:,$)) * PASSO(1,$);
61    T(1,$+1) = T(1,$) + PASSO(1,$);
62    //
63    // halt the construction if the computed step is too small
64    // and tf has not been reached
65    //
66    if (PASSO(1,$) < HMIN) & (T(1,$) < tf) then break; end;
67    //
68  end;
69  //
70  // Verify if the integration reached tf
71  //
72  if T(1,$) < tf then
73     printf("\n\nIntegrazione interrotta a T = %3.2e", T(1,$));
74  end;
75  //
76  endfunction
77  //
78  // Example to assign values to the optional parameters:
79  //
80  //    [T,X,PASSO] = RK12_pv(x0,t0,tf,F,G,E,HMIN = y);
81  //
82  //          => LAMBDA = predefined value, HMIN = y, TAU =  predefined value
83  //
```

Note that:

- When choosing the step, the difference XX1 - XX2 can be determined *without calculating* XX1 and XX2 (lines 48-55). In fact, it turns out:

$$\text{XX1 - XX2} = \frac{\text{ST1 - ST2}}{2}\ \text{TAU}$$

- *The same value of* τ was used for the choice of the step *at each iteration*.

The file containing the procedure, together with an example of application to the pendulum equation (the same as Example (4.14) of Lecture 30), can be found on the course web page, section "altro materiale didattico".