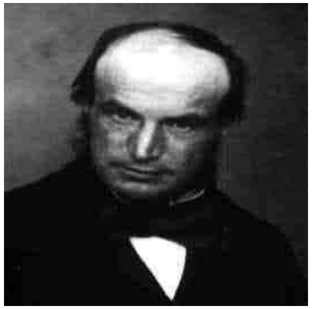
Ordinary Differential Equations – IVP III

Numerical Methods for ODE

- **One-step Methods**
  - Euler's Method
  - Analysis of the one-step methods
  - Runge-Kutta Methods
- **Multi-step Methods**
  - Adams-Bashforth
  - Adams-Moulton
  - Predictor-Corrector
- **Systems of ODE**
- **Stability**
- **Stiff Problems**

Leonhard Euler (1707-1783),

Martin Kutta
Carl David Runge (1856-1927)



J.C. Adams
(1819-1882)

Runge-Kutta Methods (1905)

- The Runge-Kutta methods are famous due to their efficiency; are included in almost all ODE software packages.
- They are one-step methods, like Euler's methods, but they are more accurate (order) $p \geq 2$
- However, the number of function evaluations for each step increases.

Runge-Kutta Methods

Generalized form of one-step methods:

$$u_{n+1} = u_n + h\Phi(x_n, u_n, h; f)$$

$$RK: \qquad \Phi = b_1 K_1 + \ldots + b_m K_m$$

Key Idea: compute the integral

$$y(x_{n+1}) - y(x_n) = \int_{x_n}^{x_{n+1}} f(x, y(x))dx$$

Consider a partition of $[x_n, x_{n+1}]$:

$$x_n \le x_n + hc_1 \le x_n + hc_2 \le .. \le x_n + hc_m \le x_{n+1}$$

we approximate the integral with a quadrature formula on **m** stages (nodes):

Runge-Kutta Methods m stages

$$\int_{x_n}^{x_{n+1}} f(x, y(x))dx \approx h\sum_{r=1}^{m} b_r \underbrace{f(x_n + hc_r, \boxed{y(x_n + hc_r)})}_{K_r}$$

THE VALUE OF y(x) IN NODES IS NOT KNOWN x_n+hc_r , we approximate it with v_r .

Let $v_1 = y(x_n) = u_n$. At each node, compute an approximation K_r evaluating f(t,y) as a linear combination of the previous estimates

$$y(x_n + hc_r) \approx v_r \quad r = 1,2,..,m$$

$$v_r = u_n + h\sum_{s=1}^{r-1} a_{rs} f(x_n + hc_s, v_s)$$

Runge-Kutta Methods m stages

General formula of RK methods:

$$u_{n+1} = u_n + h\sum_{r=1}^{m} b_r K_r$$

$$K_r = f\left(x_n + c_r h, u_n + h\sum_{s=1}^{r-1} a_{rs} K_s \right) \qquad r = 1, 2, \ldots, m$$

$\{a_{rs}\}, \{c_r\}, \{b_r\}$ They characterize the method

m = number of stages

Runge-Kutta Methods m stages Explicit and Implicit

$$u_{n+1} = u_n + h\sum_{r=1}^{m} b_r K_r$$

r-1

$$K_r = f\left(x_n + c_r h, u_n + h\sum_{s=1}^{m} a_{rs}K_s\right) \qquad r = 1,2,\ldots,m$$

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \cdot & a_{1s} \\ c_2 & a_{21} & a_{22} & \cdot & a_{2s} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ c_s & a_{s1} & a_{s2} & \cdot & a_{ss} \\ \hline & b_1 & b_2 & \cdot & b_s \end{array}$$

Butcher's matrix

Explicit RK: If $a_{ij} = 0$ for $j \geq i, \quad i = 1, 2, \ldots, m$
then each Kr can be explicitly calculated in function of the only r-1 coefficients K_1, K_2, ..., K_{r-1} already previously calculated.
Otherwise it is implicit and the Ki calculation generally requires the solution of a non-linear system.

Runge-Kutta Methods m stages

Butcher's matrix

$$
\begin{array}{c|ccccc}
\boldsymbol{c_1} & \boldsymbol{a_{11}} & \boldsymbol{a_{12}} & \cdot & \boldsymbol{a_{1s}} \\
\boldsymbol{c_2} & \boldsymbol{a_{21}} & \boldsymbol{a_{22}} & \cdot & \boldsymbol{a_{2s}} \\
\cdot & \cdot & \cdot & \cdot & \cdot \\
\boldsymbol{c_s} & \boldsymbol{a_{s1}} & \boldsymbol{a_{s2}} & \cdot & \boldsymbol{a_{ss}} \\
\hline
 & \boldsymbol{b_1} & \boldsymbol{b_2} & \cdot & \boldsymbol{b_s}
\end{array}
$$

Condition:

$$\boxed{c_r = \sum_{s=1}^{m} a_{rs} \qquad r = 1,2,..,m}$$

$$\Phi(x_n, u_{n,} f_n; h) = \sum_r b_r K_r$$

The method is certainly accurate if: $\sum_{r=1}^{m} b_r = 1$

RK Methods: coefficient calculation

The accuracy order of a method is the exponent of the power of ***h*** of the ELT.

The unknown RK coefficients are calculated by imposing the desired order s in the local truncation error, i.e.

Imposing that s terms in the Taylor series expansion of the exact solution $y(x_{n+1})$ in a neighborhood of x_n coincide with those of the approximate solution u_{n+1}.

Second order Runge-Kutta (m=2)

General formula for a second order method:

$$\boxed{\begin{aligned} k_1 &= f\left(x_n, y_n\right) \\ k_2 &= f\left(x_n + c_2 h, y_n + a_{21} k_1\right) \\ u_{n+1} &= u_n + h(b_1 k_1 + b_2 k_2) \end{aligned}}$$

$\{a_{rs}\}, \{c_r\}, \{b_r\}$ They characterize the method

m = 2 , number of stages (nodes)

$$\begin{array}{c|cc} c_2 & a_{21} & \\ \hline & b_1 & b_2 \end{array}$$

Second order Runge-Kutta (m=2): Heun's method

If we use the coefficients:

$$\begin{array}{c|cc} 1 & 1 & \\ \hline & \tfrac{1}{2} & \tfrac{1}{2} \end{array}$$

we get the RK scheme:

$$\boxed{\begin{aligned} k_1 &= f\left(x_i, u_i\right) \\ k_2 &= f\left(x_i + h, u_i + hk_1\right) \\ u_{i+1} &= u_i + \frac{1}{2}h\left[k_1 + k_2\right] \end{aligned}}$$

Heun's method

Second order Runge-Kutta (m=2): example

If we use the coefficients:

$$\begin{array}{c|cc} 2/3 & 2/3 & \\ \hline & 1/4 & 3/4 \end{array}$$

we get the RK scheme:

$$\boxed{\begin{aligned} k_1 &= f\left(x_i, u_i\right) \\ k_2 &= f\left(x_i + \frac{2}{3}h, u_i + \frac{2}{3}k_1\right) \\ u_{i+1} &= u_i + \frac{1}{4}hk_1 + \frac{3}{4}hk_2 \end{aligned}}$$

Fourth order Runge-Kutta (m=4): example

The most popular RK methods are fourth order.
The following is the most commonly used form:

$$u_{n+1} = u_n + \frac{h}{6}\left[k_1 + 2k_2 + 2k_3 + k_4\right]$$

$$k_1 = f\left(x_n, u_n\right)$$

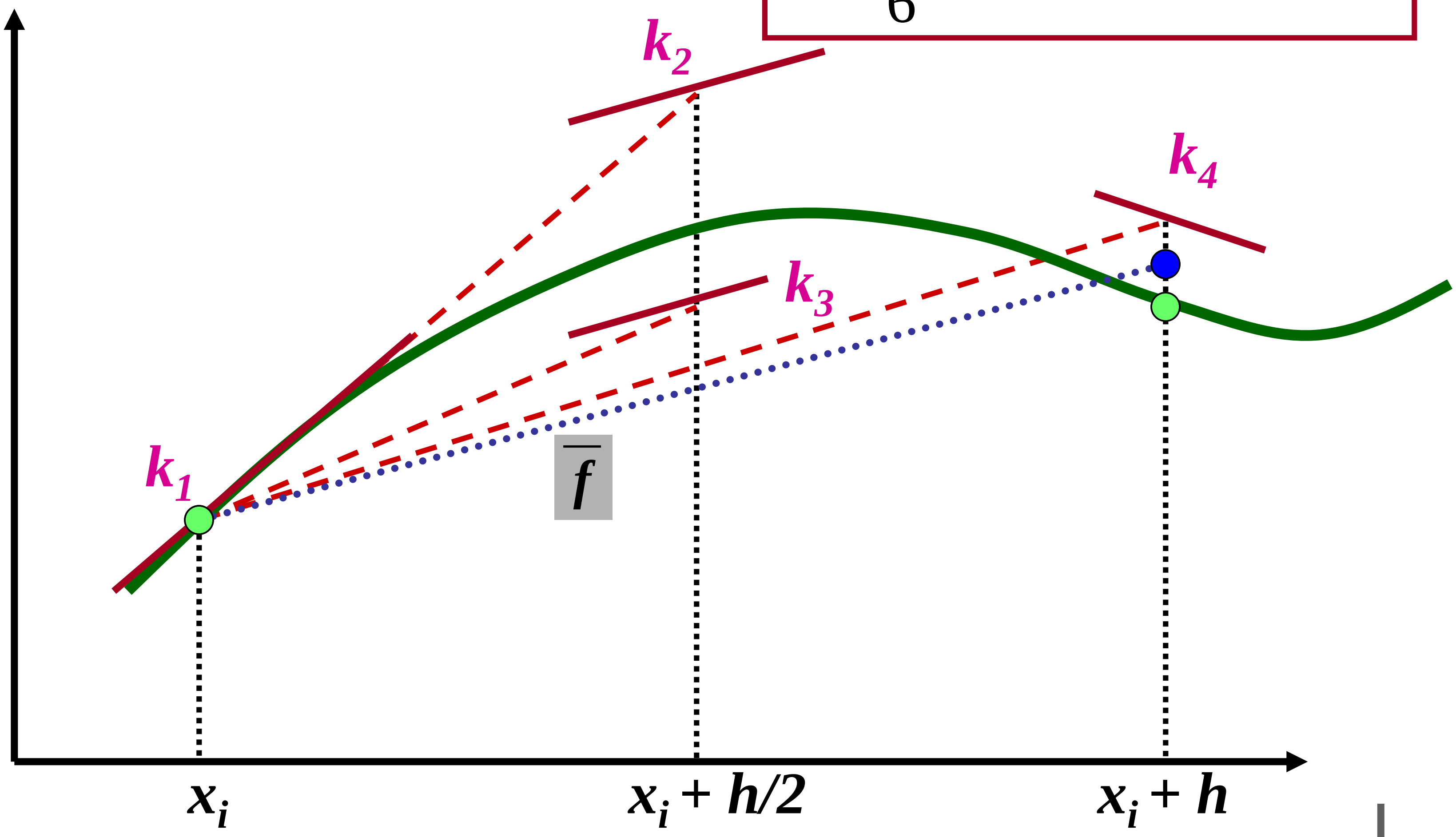
$$k_2 = f\left(x_n + \frac{1}{2}h, u_n + \frac{1}{2}k_1\right)$$

$$k_3 = f\left(x_n + \frac{1}{2}h, u_n + \frac{1}{2}k_2\right)$$

$$k_4 = f\left(x_n + h, u_n + k_3\right)$$

Fourth order Runge-Kutta (m=4)

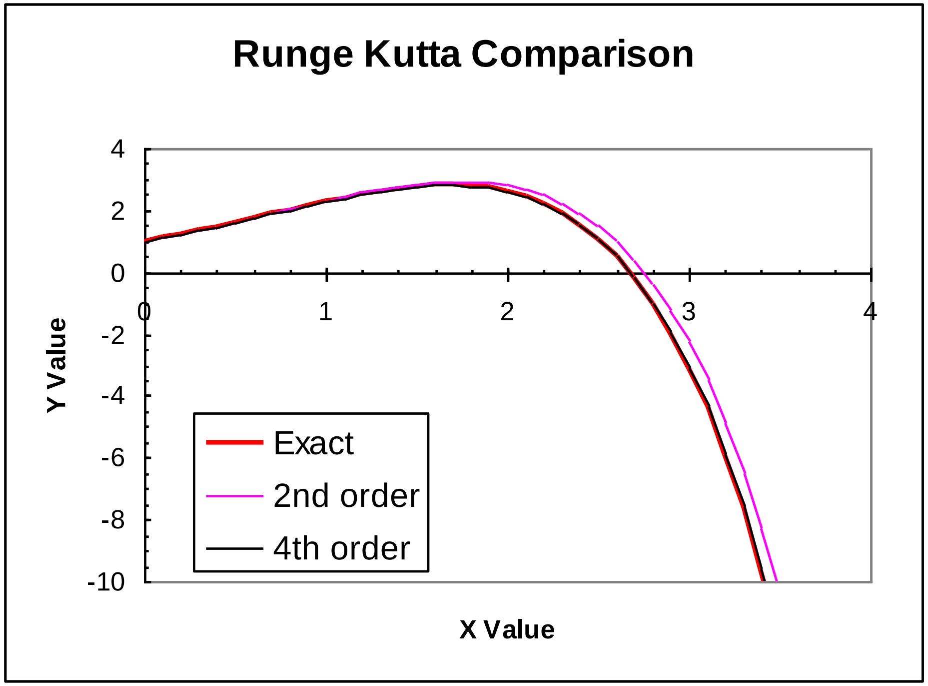
$$\overline{f} = \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

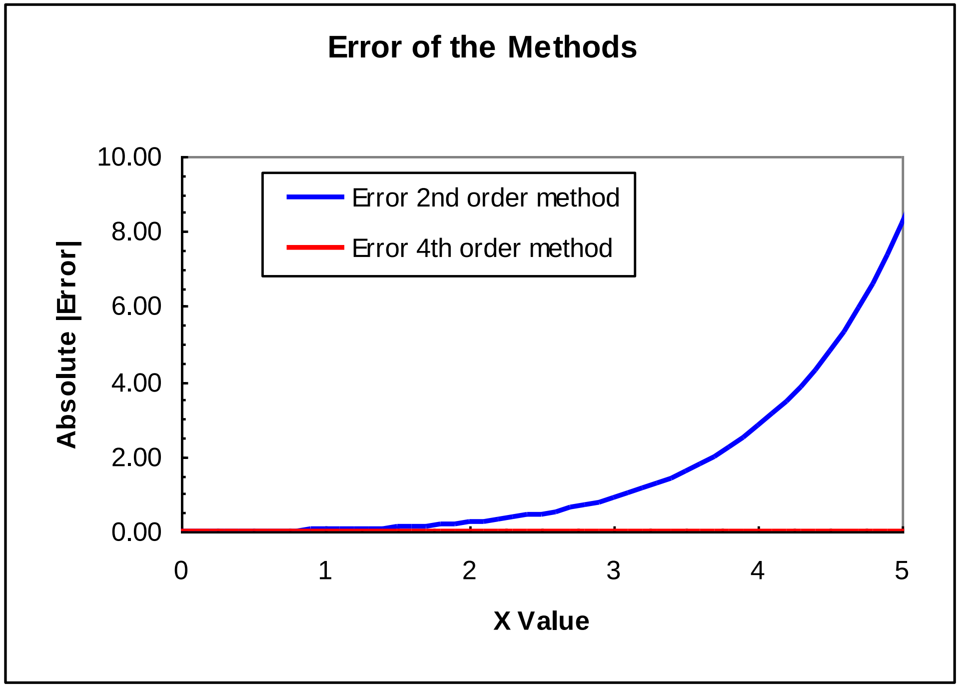


RK Methods

Comparison of Runge-Kutta methods

of the 2nd order and 4th order.





Runge-Kutta Methods

The relationship between the number of evaluations of the function f (**m**) and the order of the local truncation error (**p**) is given by the following table:

RK Explicit

Order p	1	2	3	4	5	6	7	8
Stages m	1	2	3	4	6	7	9	11

$Order_{max}$ =m for m=1,2,3,4
$Order_{max}$ =m-1 for m=5,6,7
$Order_{max}$ =m-2 for m=8,9
$Order_{max}$ =m-3 for m≥10

Proposition: for p ≥ 8 there is no explicit method of order p with m = p + 2 stages

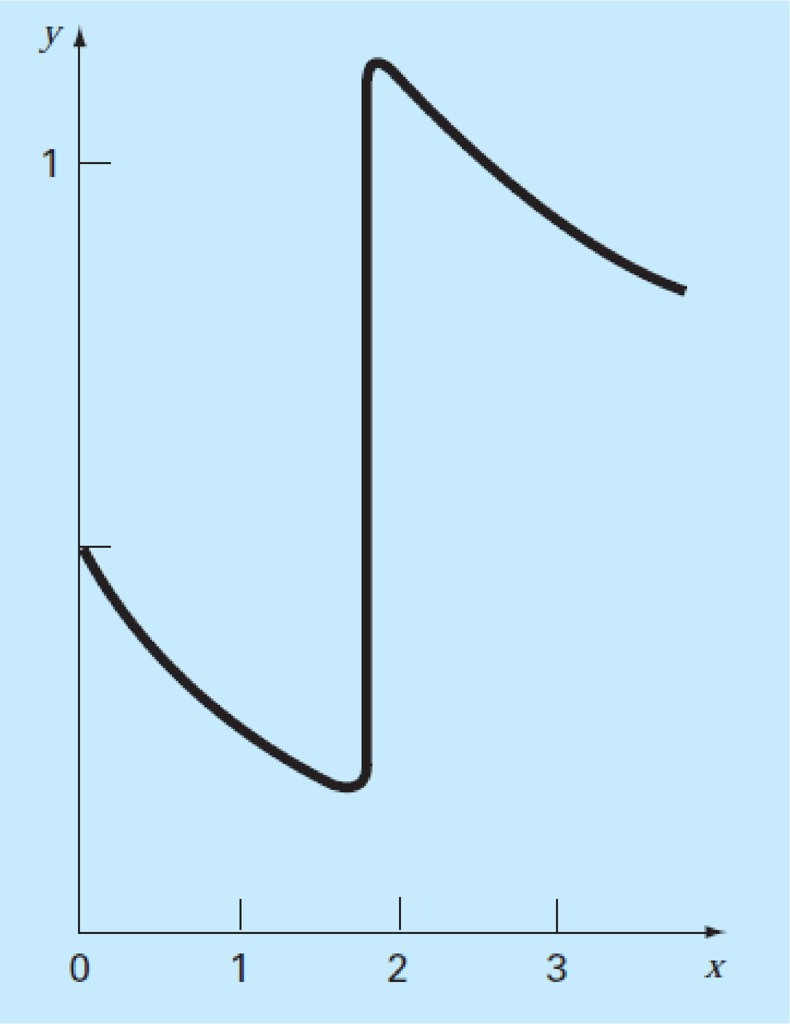
RK Implicit methods with m stages, then maximum order is 2m

Adaptive step methods

$$h_n = x_{n+1} - x_n$$

h big	low cost but large discretization error
h small	significant calculation effort but greater accuracy

Determine the largest increment of the step h in such a way that the discretization error, after having carried out a step with such an increment, remains still below a certain tolerance.



An example of a solution of an ODE that exhibits an abrupt change. Automatic step-size adjustment has great advantages for such cases.

-> *adaptive step-size control methods.*

Adaptive step-size methods

Being one step, the Runge-Kutta methods are well suited to changing the integration step h, as long as you have an efficient estimator of the local error committed to the single step.

Strategies for estimating the local truncation error

1.using the same Runge-Kutta method with two different steps (typically 2h and h);

2.using two different order Runge-Kutta methods at the same step, but with the same number m of stages.

Estimate of the local truncation error

2. Strategy with different orders

Simultaneously use two RK methods at m and m * nodes, of order p and p + 1 respectively, which have the same set of values Ki, i = 1, .., m.

The following ELT estimate is assumed:

$$\boxed{\tau_n \cong \frac{u_{n+1} - \widehat{u}_{n+1}}{h}}$$

u_{n+1} : ***order p*** method with m stages

$\widehat{u}_{n+1}$: ***order p+1*** method with m* stages

Proof

$$u_{n+1} = u_n + h_n \Phi(x_n, u_n, f_n; h)$$ Order p, local truncation error τ_n

$$\hat{u}_{n+1} = \hat{u}_n + h_n \hat{\Phi}(x_n, \hat{u}_n, f_n; h)$$ Order p+1, local truncation error $\hat{\tau}_n$

Assuming:

$$u_n = \hat{u}_n = y(x_n) \longleftarrow$$ Analytical solution at x_n

O(h^p)

$$\tau_n = \frac{1}{h}\left[y(x_{n+1}) - u_{n+1}\right] = \frac{1}{h}\left[y(x_{n+1}) - \hat{u}_{n+1}\right] + \frac{1}{h}\left[\hat{u}_{n+1} - u_{n+1}\right] =$$

$$= \hat{\tau}_n + \frac{1}{h}\left[\hat{u}_{n+1} - u_{n+1}\right]$$

O(h^{p+1})

Adaptive step-size Algorithm

Algorithm to adjust the step size.
In general, the strategy is to increase the step size if the error is too small and decrease it if the error is too large.

Given a tolerance ε
At each step:

Compute $\hat{u}_{n+1}, u_{n+1}$

Estimate the LTE $\boldsymbol{\tau_n \approx \frac{1}{h}\left[\hat{u}_{n+1} - u_{n+1}\right]}$

If $\boxed{|\tau_n| > \varepsilon}$

then u_{n+1} is not accepted, decrease step h, recalculate,

otherwise, u_{n+1} is accepted and continues

RKF45 Method: Runge-Kutta Fehlberg 4° order

	0	0	0	0	0	0	0	
	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0	0	
	$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$	0	0	0	0	
	$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$	0	0	0	
	1	$\frac{439}{216}$	-8	$\frac{3680}{513}$	$-\frac{845}{4104}$	0	0	
	$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3544}{2565}$	$\frac{1859}{4104}$	$-\frac{11}{40}$	0	
RK 4		$\frac{25}{216}$	0	$\frac{1408}{2565}$	$\frac{2197}{4104}$	$-\frac{1}{5}$	0	**b**
RK 5		$\frac{16}{135}$	0	$\frac{6656}{12825}$	$\frac{28561}{56430}$	$-\frac{9}{50}$	$\frac{2}{55}$	**b1**
		$\frac{1}{360}$	0	$-\frac{128}{4275}$	$-\frac{2197}{75240}$	$\frac{1}{50}$	$\frac{2}{55}$	**E=b-b1**

4th order Runge-Kutta coupled with the 5th order RK, same number of stages.

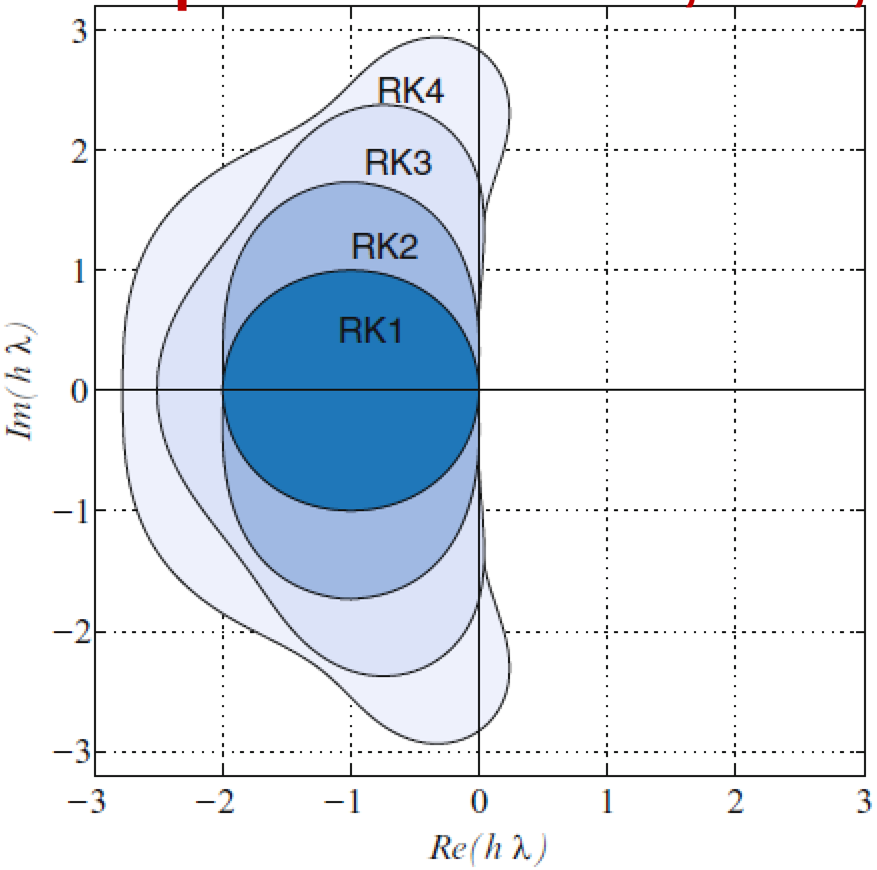
RKF45 : routine with adaptive step-size with automatic ELT control.

Remark: the existence of points of singularity in the solution are detected by the presence of excessively small step-sizes.

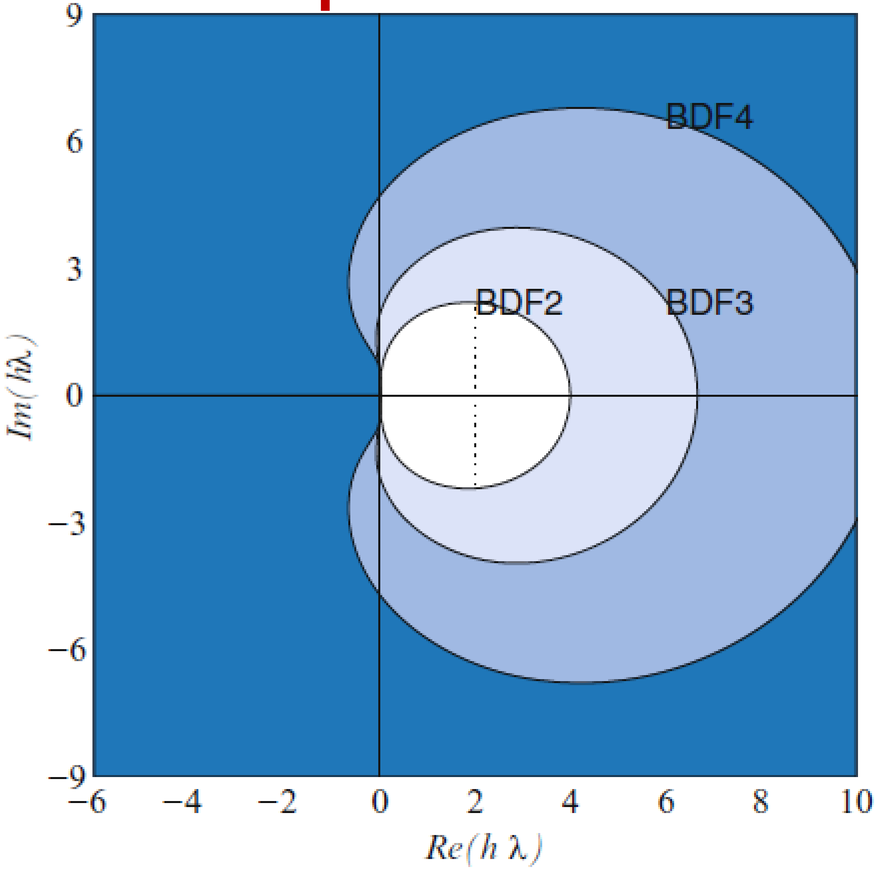
RK23 2nd order Runge-Kutta coupled with the 3rd order RK, same number of stages.

Regions of absolute stability

Explicit RK r nodes, r = 1, .., 4



Implicit RK r nodes



Explicit RK methods are generally unsuitable for the solution of ODE stiff because their region of absolute stability is small; in particular, it is bounded. They can never be A-stable. A-stable Runge-Kutta method is necessarily implicit.

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Serena Morigi

Dipartimento di Matematica

serena.morigi@unibo.it

http://www.dm.unibo.it/~morigi