NTNU

Norwegian University of Science and Technology

TMA4125 Matematikk 4N

Numerical methods for ordinary differential equations — Runge–Kutta methods and adaptive stepping

Ronny Bergmann and Douglas R. Q. Pacheco

Department of Mathematical Sciences, NTNU.

March 1, 2022

Introduction

Remember the ODEs we are trying to solve: $\boldsymbol{y}'(t) = \boldsymbol{f}(t, y)$

Introduction

Remember the ODEs we are trying to solve: $\boldsymbol{y}'(t) = \boldsymbol{f}(t, y)$

Euler's method:

$\boldsymbol{k}_1 = \boldsymbol{f}(t_n, \boldsymbol{y}_n)$

$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\boldsymbol{k}_1 \quad \rightarrow \quad \mathcal{O}(h)$ convergence

Introduction

Remember the ODEs we are trying to solve: $\boldsymbol{y}'(t) = \boldsymbol{f}(t, y)$

Euler's method:

$\boldsymbol{k}_1 = \boldsymbol{f}(t_n, \boldsymbol{y}_n)$

$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\boldsymbol{k}_1 \quad \rightarrow \quad \mathcal{O}(h)$ convergence

Heun's method:

$\boldsymbol{k}_1 = \boldsymbol{f}(t_n, \boldsymbol{y}_n)$

$\boldsymbol{k}_2 = \boldsymbol{f}(t_n + h, \boldsymbol{y}_n + h\boldsymbol{k}_1)$

$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\,[0.5\boldsymbol{k}_1 + 0.5\boldsymbol{k}_2] \quad \rightarrow \quad \mathcal{O}(h^2)$ convergence

Introduction

Remember the ODEs we are trying to solve: $\boldsymbol{y}'(t) = \boldsymbol{f}(t, y)$

Euler's method:

$$\boldsymbol{k}_1 = \boldsymbol{f}(t_n, \boldsymbol{y}_n)$$
$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\boldsymbol{k}_1 \quad \rightarrow \quad \mathcal{O}(h) \text{ convergence}$$

Heun's method:

$$\boldsymbol{k}_1 = \boldsymbol{f}(t_n, \boldsymbol{y}_n)$$
$$\boldsymbol{k}_2 = \boldsymbol{f}(t_n + h, \boldsymbol{y}_n + h\boldsymbol{k}_1)$$
$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\left[0.5\boldsymbol{k}_1 + 0.5\boldsymbol{k}_2\right] \quad \rightarrow \quad \mathcal{O}(h^2) \text{ convergence}$$

Questions:

- Can we generalise this?
- Can we do better than $\mathcal{O}(h^2)$?

Runge–Kutta methods

RK methods are one-step methods following the general scheme

$$\boldsymbol{k}_i = \boldsymbol{f}\Big(t_n + c_i h,\, \boldsymbol{y}_n + h \sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j\Big), \quad i = 1, \dots, s \quad (s \text{ stages})$$

Runge–Kutta methods

RK methods are one-step methods following the general scheme

$$\boldsymbol{k}_i = \boldsymbol{f}\Big(t_n + c_i h,\ \boldsymbol{y}_n + h\sum_{j=1}^{s} a_{ij}\boldsymbol{k}_j\Big), \quad i = 1, \dots, s \quad (s \text{ stages})$$

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\sum_{i=1}^{s} b_i \boldsymbol{k}_i$$

Runge–Kutta methods

RK methods are one-step methods following the general scheme

$$\boldsymbol{k}_i = \boldsymbol{f}\Big(t_n + c_i h,\ \boldsymbol{y}_n + h\sum_{j=1}^{s} a_{ij}\boldsymbol{k}_j\Big), \quad i = 1, \dots, s \quad (s \text{ stages})$$

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\sum_{i=1}^{s} b_i \boldsymbol{k}_i$$

The coefficients are all real numbers, presented in the Butcher tableau:

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

Runge–Kutta methods

$$\boldsymbol{k}_i = \boldsymbol{f}\Big(t_n + c_i h,\ \boldsymbol{y}_n + h\sum_{j=1}^{s} a_{ij}\boldsymbol{k}_j\Big)$$

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\sum_{i=1}^{s} b_i\boldsymbol{k}_i$$

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

Euler's method:

$$\boldsymbol{k}_1 = \boldsymbol{f}(t_n, \boldsymbol{y}_n)$$

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\boldsymbol{k}_1$$

Runge–Kutta methods

$$\boldsymbol{k}_i = \boldsymbol{f}\Big(t_n + c_i h,\ \boldsymbol{y}_n + h\sum_{j=1}^{s} a_{ij}\boldsymbol{k}_j\Big)$$

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\sum_{i=1}^{s} b_i \boldsymbol{k}_i$$

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

(????) method:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

Runge–Kutta methods

$$\boldsymbol{k}_i = \boldsymbol{f}\left(t_n + c_i h,\ \boldsymbol{y}_n + h\sum_{j=1}^{s} a_{ij}\boldsymbol{k}_j\right)$$

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\sum_{i=1}^{s} b_i\boldsymbol{k}_i$$

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

Heun (3rd-order):

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 \\ 2/3 & 0 & 2/3 & 0 \\ \hline & 1/4 & 0 & 3/4 \end{array}$$

NTNU

Consistency

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

Consistency

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

Theorem

An RK method is order-p consistent if, and only if all the conditions up to p in the table are satisfied

Consistency

p	Conditions
1	$\sum_{i=1}^{s} b_i = 1$
2	$\sum_{i=1}^{s} b_i c_i = \frac{1}{2}$
3	$\sum_{i=1}^{s} b_i {c_i}^2 = \frac{1}{3}$
	$\sum_{i=1}^{s} \sum_{j=1}^{s} b_i a_{ij} c_j = \frac{1}{6}$
4	$\sum_{i=1}^{s} b_i {c_i}^3 = \frac{1}{4}$
	$\sum_{i=1}^{s} \sum_{j=1}^{s} b_i c_i a_{ij} c_j = \frac{1}{8}$
	$\sum_{i=1}^{s} \sum_{j=1}^{s} b_i a_{ij} {c_j}^2 = \frac{1}{12}$
	$\sum_{i=1}^{s} \sum_{j=1}^{s} \sum_{k=1}^{s} b_i a_{ij} a_{jk} c_k = \frac{1}{24}$

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

Theorem

An RK method is order-p consistent if, and only if all the conditions up to p in the table are satisfied

Consistency

Heun's method:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

p	Conditions
1	$\sum_{i=1}^{s} b_i = 1$
2	$\sum_{i=1}^{s} b_i c_i = \frac{1}{2}$
3	$\sum_{i=1}^{s} b_i {c_i}^2 = \frac{1}{3}$ $\sum_{i=1}^{s} \sum_{j=1}^{s} b_i a_{ij} c_j = \frac{1}{6}$
4	$\sum_{i=1}^{s} b_i {c_i}^3 = \frac{1}{4}$ $\sum_{i=1}^{s} \sum_{j=1}^{s} b_i c_i a_{ij} c_j = \frac{1}{8}$ $\sum_{i=1}^{s} \sum_{j=1}^{s} b_i a_{ij} {c_j}^2 = \frac{1}{12}$ $\sum_{i=1}^{s} \sum_{j=1}^{s} \sum_{k=1}^{s} b_i a_{ij} a_{jk} c_k = \frac{1}{24}$

Error estimation

One-step methods: $\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\boldsymbol{\Phi}(t_n, \boldsymbol{y}_n, h)$

Question: for a method $\boldsymbol{\Phi}$, how can we estimate the error $\epsilon_n = |\boldsymbol{y}_n - \boldsymbol{y}(t_n)|$, if we don't know the exact solution $\boldsymbol{y}(t)$?

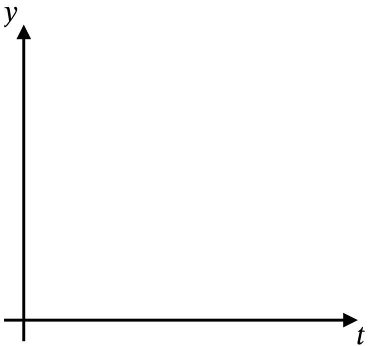
Error estimation

One-step methods: $\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\boldsymbol{\Phi}(t_n, \boldsymbol{y}_n, h)$

Question: for a method $\boldsymbol{\Phi}$, how can we estimate the error $\epsilon_n = |\boldsymbol{y}_n - \boldsymbol{y}(t_n)|$, if we don't know the exact solution $\boldsymbol{y}(t)$?

Idea

Why not use a more accurate method $\hat{\boldsymbol{\Phi}}$ to compute $\hat{\epsilon}_n = |\boldsymbol{y}_n - \hat{\boldsymbol{y}}_n|$?



Error estimation

If $\hat{\epsilon}_{n+1} > \text{tol}$, we might want to reduce the time-step size h and recompute the step... but how can we estimate a good h_{new} ?

Error estimation

If $\hat{\epsilon}_{n+1} > \text{tol}$, we might want to reduce the time-step size h and recompute the step... but how can we estimate a good h_{new} ?

- Since $\hat{\mathbf{\Phi}}$ is convergent of order $p+1$, we can write

$$\hat{\boldsymbol{y}}_{n+1} - \boldsymbol{y}(t_n + h) = Ah^{p+1}$$

Error estimation

If $\hat{\epsilon}_{n+1} > \text{tol}$, we might want to reduce the time-step size h and recompute the step... but how can we estimate a good h_{new} ?

- Since $\hat{\mathbf{\Phi}}$ is convergent of order $p+1$, we can write
  $\hat{\mathbf{y}}_{n+1} - \mathbf{y}(t_n + h) = Ah^{p+1}$
- Since $\mathbf{\Phi}$ is consistent of order p, we can write
  $\frac{\mathbf{y}(t_n+h)-\mathbf{y}(t_n)}{h} - \mathbf{\Phi}(t_n, \mathbf{y}(t_n), h) = Bh^p$, that is,

Error estimation

If $\hat{\epsilon}_{n+1} > \text{tol}$, we might want to reduce the time-step size h and recompute the step... but how can we estimate a good h_{new} ?

- Since $\hat{\mathbf{\Phi}}$ is convergent of order $p+1$, we can write

  $\hat{\mathbf{y}}_{n+1} - \mathbf{y}(t_n + h) = Ah^{p+1}$

- Since $\mathbf{\Phi}$ is consistent of order p, we can write

  $\frac{\mathbf{y}(t_n+h)-\mathbf{y}(t_n)}{h} - \mathbf{\Phi}(t_n, \mathbf{y}(t_n), h) = Bh^p$, that is,

  $\mathbf{y}(t_n + h) - [\mathbf{y}(t_n) + h\mathbf{\Phi}(t_n, \mathbf{y}(t_n), h)] = Bh^{p+1}$

NTNU

Error estimation

If $\hat{\epsilon}_{n+1} > \text{tol}$, we might want to reduce the time-step size h and recompute the step... but how can we estimate a good h_{new} ?

- Since $\hat{\mathbf{\Phi}}$ is convergent of order $p+1$, we can write

  $$\hat{\boldsymbol{y}}_{n+1} - \boldsymbol{y}(t_n + h) = Ah^{p+1}$$

- Since $\mathbf{\Phi}$ is consistent of order p, we can write

  $\frac{\boldsymbol{y}(t_n+h)-\boldsymbol{y}(t_n)}{h} - \mathbf{\Phi}(t_n, \boldsymbol{y}(t_n), h) = Bh^p$, that is,

  $$\boldsymbol{y}(t_n + h) - [\boldsymbol{y}(t_n) + h\mathbf{\Phi}(t_n, \boldsymbol{y}(t_n), h)] = Bh^{p+1}$$

Hence, we get

$$\left|\hat{\boldsymbol{y}}_{n+1} - \boldsymbol{y}_{n+1}\right| \approx |A + B|h^{p+1}$$

Error estimation

If $\hat{\epsilon}_{n+1} > \text{tol}$, we might want to reduce the time-step size h and recompute the step... but how can we estimate a good h_{new} ?

- Since $\hat{\mathbf{\Phi}}$ is convergent of order $p+1$, we can write

  $$\hat{\boldsymbol{y}}_{n+1} - \boldsymbol{y}(t_n + h) = Ah^{p+1}$$

- Since $\mathbf{\Phi}$ is consistent of order p, we can write

  $\frac{\boldsymbol{y}(t_n+h)-\boldsymbol{y}(t_n)}{h} - \mathbf{\Phi}(t_n, \boldsymbol{y}(t_n), h) = Bh^p$, that is,

  $$\boldsymbol{y}(t_n + h) - [\boldsymbol{y}(t_n) + h\mathbf{\Phi}(t_n, \boldsymbol{y}(t_n), h)] = Bh^{p+1}$$

Hence, we get

$$\hat{\epsilon}_{n+1} = \left|\hat{\boldsymbol{y}}_{n+1} - \boldsymbol{y}_{n+1}\right| \approx |A + B| h^{p+1}$$

Time-step control

$$\hat{\epsilon}_{n+1} \approx Ch^{p+1}$$

$$\hat{\epsilon}_{new} \approx Ch_{new}^{p+1} \quad \Rightarrow$$

Time-step control

$$\hat{\epsilon}_{n+1} \approx Ch^{p+1}$$

$$\hat{\epsilon}_{new} \approx Ch_{new}^{p+1} \quad \Rightarrow$$

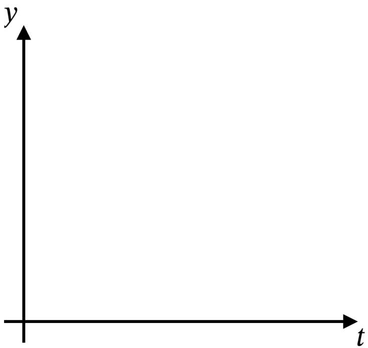
$$\hat{\epsilon}_{new} < tol \quad \Leftrightarrow \quad h_{new} < \left[\left(\frac{\text{tol}}{\hat{\epsilon}_{n+1}} \right)^{\frac{1}{p+1}} \right] h$$

Time-step control

$$\hat{\epsilon}_{n+1} \approx C h^{p+1}$$

$$\hat{\epsilon}_{new} \approx C h_{new}^{p+1} \quad \Rightarrow$$

$$\hat{\epsilon}_{new} < tol \quad \Leftrightarrow \quad h_{new} < \left[\left(\frac{\text{tol}}{\hat{\epsilon}_{n+1}} \right)^{\frac{1}{p+1}} \right] h$$



Time-step control: Heun + Euler

$$y' = 2 + 2^{-t} - \sin(0.25\pi y), \quad y(0) = 2, \quad h = 0.5, \quad \text{tol} = 0.01$$

Time-step control: Heun + Euler

$$y' = 2 + 2^{-t} - \sin(0.25\pi y), \quad y(0) = 2, \quad h = 0.5, \quad \text{tol} = 0.01$$

First step:

- Evaluate $k_1 = f(t_n, y_n) = 2$
- Compute Euler step: $y_{n+1}^{\text{Euler}} = y_n + hk_1 = 3$
- Evaluate $k_2 = f(t_n + h, y_n + hk_1) = 2$
- Compute $y_{n+1} = y_n + h\left[\frac{k_1}{2} + \frac{k_2}{2}\right] = 3$

Time-step control: Heun + Euler

$$y' = 2 + 2^{-t} - \sin(0.25\pi y), \quad y(0) = 2, \quad h = 0.5, \quad \text{tol} = 0.01$$

First step:

- Evaluate $k_1 = f(t_n, y_n) = 2$
- Compute Euler step: $y_{n+1}^{\text{Euler}} = y_n + hk_1 = 3$
- Evaluate $k_2 = f(t_n + h, y_n + hk_1) = 2$
- Compute $y_{n+1} = y_n + h\left[\frac{k_1}{2} + \frac{k_2}{2}\right] = 3$

Error estimate: $\hat{\epsilon}_{n+1} = |3 - 3| = 0 < \text{tol}$

Time-step control: Heun + Euler

$$y' = 2 + 2^{-t} - \sin(0.25\pi y), \quad y(0) = 2, \quad h = 0.5, \quad \text{tol} = 0.01$$

First step:

- Evaluate $k_1 = f(t_n, y_n) = 2$
- Compute Euler step: $y_{n+1}^{\text{Euler}} = y_n + hk_1 = 3$
- Evaluate $k_2 = f(t_n + h, y_n + hk_1) = 2$
- Compute $y_{n+1} = y_n + h\left[\frac{k_1}{2} + \frac{k_2}{2}\right] = 3$

Error estimate: $\hat{\epsilon}_{n+1} = |3 - 3| = 0 < \text{tol}$

$\Rightarrow$ No need to recompute step!

$\Rightarrow$ We can keep h as it is,
or even increase it a bit...

Time-step control: Heun + Euler

$$y' = 2 + 2^{-t} - \sin(0.25\pi y), \quad y_1 = 3, \quad h = 0.5, \quad \text{tol} = 0.01$$

Second step:

- Evaluate $k_1 = f(t_n, y_n) = 2$
- Compute Euler step: $y_{n+1}^{\text{Euler}} = y_n + hk_1 = 4$
- Evaluate $k_2 = f(t_n + h, y_n + hk_1) = 2.5$
- Compute $y_{n+1} = y_n + h\left[\frac{k_1}{2} + \frac{k_2}{2}\right] = 4.125$

Time-step control: Heun + Euler

$$y' = 2 + 2^{-t} - \sin\left(0.25\pi y\right), \quad y_1 = 3, \quad h = 0.5, \quad \text{tol} = 0.01$$

Second step:

- Evaluate $k_1 = f(t_n, y_n) = 2$
- Compute Euler step: $y_{n+1}^{\text{Euler}} = y_n + hk_1 = 4$
- Evaluate $k_2 = f(t_n + h, y_n + hk_1) = 2.5$
- Compute $y_{n+1} = y_n + h\left[\frac{k_1}{2} + \frac{k_2}{2}\right] = 4.125$

Error estimate: $\hat{\epsilon}_{n+1} = |4.125 - 4| = 0.125 > \text{tol}$

Time-step control: Heun + Euler

$$y' = 2 + 2^{-t} - \sin(0.25\pi y), \quad y_1 = 3, \quad h = 0.5, \quad \text{tol} = 0.01$$

Second step:

- Evaluate $k_1 = f(t_n, y_n) = 2$
- Compute Euler step: $y_{n+1}^{\text{Euler}} = y_n + hk_1 = 4$
- Evaluate $k_2 = f(t_n + h, y_n + hk_1) = 2.5$
- Compute $y_{n+1} = y_n + h\left[\frac{k_1}{2} + \frac{k_2}{2}\right] = 4.125$

Error estimate: $\hat{\epsilon}_{n+1} = |4.125 - 4| = 0.125 > \text{tol}$

$\Rightarrow$ Reduce h and recompute the step:

$$h_{new} < \left[\left(\frac{\text{tol}}{\hat{\epsilon}_{n+1}}\right)^{\frac{1}{p+1}}\right] h = \left(\frac{0.01}{0.125}\right)^{\frac{1}{2}} \times 0.5 \approx 0.14$$

Time-step control: embedded RK

We can combine a higher-order method and a lower-order method as:

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

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h \sum_{i=1}^{s} b_i \boldsymbol{k}_i \,, \quad \mathcal{O}(h^{p+1})$$

$$\boldsymbol{y}^*_{n+1} = \boldsymbol{y}_n + h \sum_{i=1}^{s} b_i^* \boldsymbol{k}_i \,, \quad \mathcal{O}(h^{p})$$

NTNU

Time-step control: embedded RK

We can combine a higher-order method and a lower-order method as:

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

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h \sum_{i=1}^{s} b_i \boldsymbol{k}_i\,, \quad \mathcal{O}(h^{p+1})$$

$$\boldsymbol{y}^*_{n+1} = \boldsymbol{y}_n + h \sum_{i=1}^{s} b_i^* \boldsymbol{k}_i\,, \quad \mathcal{O}(h^{p})$$

Therefore, we get simply $\hat{\epsilon}_{n+1} = \left|\boldsymbol{y}_{n+1} - \boldsymbol{y}^*_{n+1}\right| = h\left|\sum_{i=1}^{s}(b_i - b_i^*)\boldsymbol{k}_i\right|$,

Time-step control: embedded RK

We can combine a higher-order method and a lower-order method as:

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

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h \sum_{i=1}^{s} b_i \boldsymbol{k}_i \,, \quad \mathcal{O}(h^{p+1})$$

$$\boldsymbol{y}^*_{n+1} = \boldsymbol{y}_n + h \sum_{i=1}^{s} b^*_i \boldsymbol{k}_i \,, \quad \mathcal{O}(h^{p})$$

Therefore, we get simply $\hat{\epsilon}_{n+1} = \left|\boldsymbol{y}_{n+1} - \boldsymbol{y}^*_{n+1}\right| = h \left|\sum_{i=1}^{s} (b_i - b_i^*)\boldsymbol{k}_i\right|$,

so we can easily compute

$$h_{new} < \left[\left(\frac{\text{tol}}{\hat{\epsilon}_{n+1}}\right)^{\frac{1}{p+1}}\right] h$$