Soluzione di equazioni differenziali

Matteo Duranti

matteo.duranti@pg.infn.it

(cfr. W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery - Numerical Recipes, The Art of Scientific Computing

[http://www.aip.de/groups/soe/local/numres/bookcpdf/c16-1.pdf (legale?!)]
[http://www.aip.de/groups/soe/local/numres/bookcpdf/c16-2.pdf (legale?!)]
https://it.wikipedia.org/wiki/Metodo_di_Eulero
https://en.wikipedia.org/wiki/Euler_method

https://en.wikipedia.org/wiki/Midpoint_method

https://en.wikipedia.org/wiki/Runge-Kutta_methods
https://en.wikipedia.org/wiki/List_of_Runge-Kutta_methods)

Sistemi di risoluzione di ODE

Vogliamo risolvere dei sistemi di equazioni differenziali ordinarie (*ODE*)

$$F\left(x, y(x), y'(x), \ldots, y^{(n)}(x)\right) = 0$$

(ci limitiamo a quelle che è possibile scrivere in *forma normale*)

$$y^{(n)}(x) = f\left(x, y(x), y'(x), \ldots, y^{(n-1)}(x)\right)$$

una volta fornito un valore iniziale

$$y(x_0) = y_0, y'(x_0) = y1, \ldots, y^{(n-1)}(x_0) = y_{n-1}$$

(*Problema di Cauchy*)

Sistemi di risoluzione di ODE

Vogliamo risolvere dei sistemi di equazioni differenziali ordinarie (*ODE*)

$$F\left(x, y(x), y'(x), \ldots, y^{(n)}(x)\right) = 0$$

"risolvere":

-NON significa trovare una forma analitica per *y(x)*;

-ci basta essere in grado di saper calcolare *y(x+h)* a partire da *F* e da $x_0, y_0, y'(x_0)$, etc…

$$y(x+h) = G(x_0, y(x_0), y'(x_0), \ldots, y^{(n-1)}(x_0), F)$$

Sistemi di risoluzione di ODE

Limitiamoci anche ad equazioni del <u>primo ordine</u>

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

Per un'equazione differenziale di <u>ordine N</u>

$$y^{(n)}(x) = f\left(x, y(x), y'(x), \ldots, y^{(n-1)}(x)\right)$$

possiamo introdurre le variabili ausiliarie

$$z_1(x) = y(x), z_2(x) = y'(x), \ldots, z_N(x) = y^{(N-1)}(x)$$

$$\mathbf{z}'(x) = \begin{pmatrix} z_1'(x) \\ \vdots \\ z_{N-1}'(x) \\ z_N'(x) \end{pmatrix} = \begin{pmatrix} y'(x) \\ \vdots \\ y^{(N-1)}(x) \\ y^{(N)}(x) \end{pmatrix} = \begin{pmatrix} z_2(x) \\ \vdots \\ z_N(x) \\ f(x, z_1(x), \ldots, z_N(x)) \end{pmatrix}$$

che è un <u>sistema</u> di equazioni del <u>primo ordine</u>

Sistemi di risoluzione di ODE

Limitiamoci anche ad equazioni del <u>primo ordine</u>

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

che poi è il caso che tipicamente incontriamo "integrando" il moto e la dinamica

$$ma = qvB \quad \rightarrow \qquad v' = f(t, v(t)) = \frac{qB}{m}v$$

$$x = x_0 + vt \;\rightarrow \qquad x' = f(t, x(t)) = \frac{1}{t}\left(x - x_0\right)$$

Metodo di Eulero

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

Sostituendo la derivata con il rapporto incrementale

$$y'(x) \approx \frac{y(x+h) - y(x)}{h} \approx f\left(x, y(x)\right)$$

$$\rightarrow \quad y(x+h) = y(x) + h \cdot f\left(x, y(x)\right)$$

scritto comunemente come

$$y_{n+1} = y_n + h \cdot f\left(x_n, y_n\right)$$

cioè y_n è un'approssimazione valida per la

soluzione dell'*ODE*, $\approx y(x_n)$, a $x_n = x_0 + nh$

Metodo di Eulero - esempio

$$y'(x) = f(x, y(x)) \boxed{= y(x)} \qquad y(x_0) = y_0 \boxed{= 1}$$

e si vuole "risolvere" per $y(4)$.

(la soluzione "analitica" è banale: $y(x) = e^x$)

$$y_{n+1} = y_n + h \cdot f(x_n, y_n)$$

che, con *h=1*

$$f(x_0, y_0) = f(0,1) = 1 \qquad h \cdot f(y_0) = 1 \cdot 1 = 1$$

$$\rightarrow \quad y_1 = y_0 + hf(y_0) = 1 + 1 \cdot 1 = 2$$

$$y_2 = y_1 + hf(y_1) = 2 + 1 \cdot 2 = 4$$

$$y_3 = y_2 + hf(y_2) = 4 + 1 \cdot 4 = 8$$

$$y_4 = y_3 + hf(y_3) = 8 + 1 \cdot 8 = 16$$

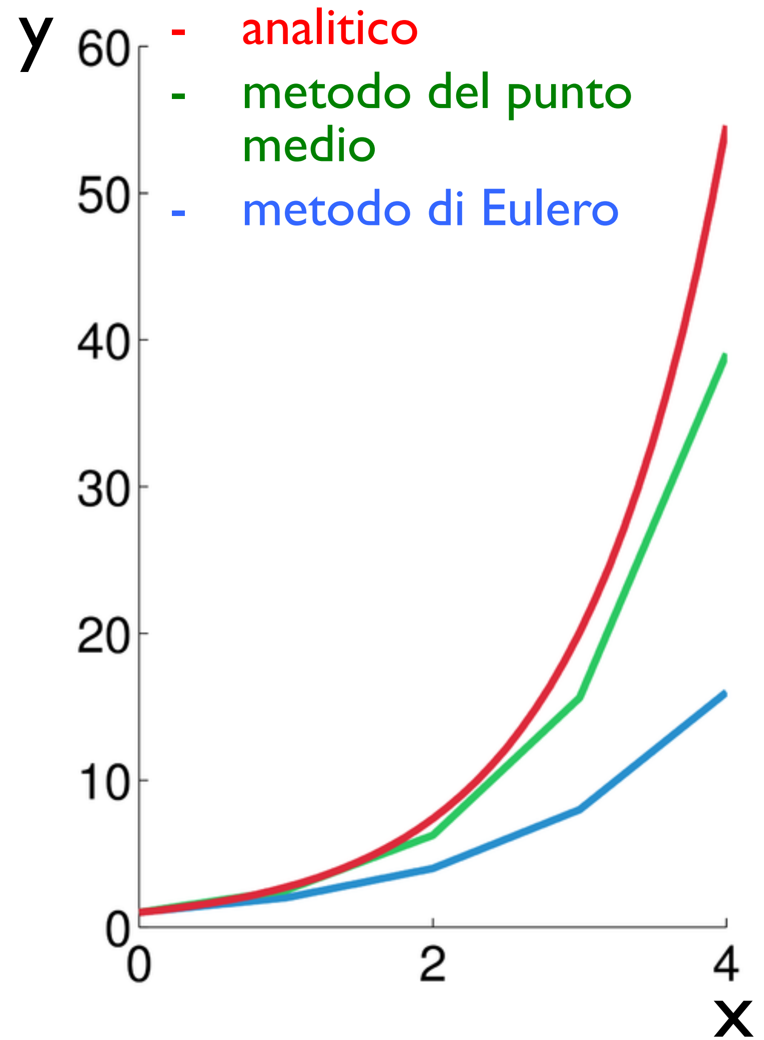
Metodo di Eulero

$$y'(x) = f(x, y(x)) = y(x) \qquad y(x_0) = y_0 = 1$$

→

n	e^n	y_n	Δ
1	~2.718	2	~0.7
2	~7.389	4	~3
3	~20.085	8	~12
4	~54.598	16	~40

n	$e^n - e^{n-1}$	$y_n - y_{n-1}$	Δ
1	~1.7	1	~0.7
2	~4.7	2	~2.7
3	~12.7	4	~8.7
4	~34.5	8	~26.5



Metodo di Eulero – Local Truncation Error

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

L'errore "locale" (nel singolo step), Local Truncation Error (LTE), lo possiamo stimare confrontando la soluzione numerica in uno step

$$y_1 = \cancel{y(x_0)} + h \cdot \cancel{f\left(x_0, y(x_0)\right)}$$

con la soluzione esatta (sviluppo di Taylor)

$$y(x_0 + h) = \cancel{y(x_0)} + \cancel{hy'(x_0)} + \frac{1}{2}h^2 y''(x_0) + O(h^3)$$

$$\rightarrow \quad \mathrm{LTE} = y(x_0 + h) - y_1 = \frac{1}{2}h^2 y''(x_0) + O(h^3)$$

Per bassi *h*, l'errore è proporzionale a h^2

Metodo di Eulero – Local Truncation Error

$$y'(x) = f(x, y(x)) = y(x) \qquad y(x_0) = y_0 = 1$$

$$\rightarrow \mathrm{LTE} = y(x_0 + h) - y_1 = \frac{1}{2}h^2 y''(x_0) + O(h^3)$$

Per bassi *h*, l'errore è proporzionale a h^2

n	$e^n - e^{n-1}$	$y_n - y_{n-1}$	Δ
1	~1.7	1	~0.7
2	~4.7	2	~2.7
3	~12.7	4	~8.7
4	~34.5	8	~26.5

$$\rightarrow \frac{1}{2}h^2 y''(x_0) = \frac{1}{2}1^2 e^0 = 0.5$$

dove abbiamo trascurato $O(h^3)$

Se ci fossimo fermati a $O(h^4)$

avremmo avuto un termine $\dfrac{1}{3!}h^3 y'''(x_0) = \dfrac{1}{6}1^3 e^0 = 0.1\bar{6}$

che già da solo (senza i termini ancora successivi) ci porta già a ~0.7

Metodo di Eulero – Local Truncation Error

$$y'(x) = f\left(x, y(x)\right) = y(x) \qquad y(x_0) = y_0 = 1$$

$$\rightarrow \ \mathrm{LTE} = y(x_0 + h) - y_1 = \frac{1}{2}h^2 y''(x_0) + O(h^3)$$

Per bassi *h*, l'errore è proporzionale a h^2

n	$e^n - e^{n-1}$	$y_n - y_{n-1}$	Δ
1	~1.7	1	~0.7
2	~4.7	2	~2.7
3	~12.7	4	~8.7
4	~34.5	8	~26.5

$$\rightarrow \frac{1}{2}h^2 y''(x_0) + O(h^3) + \cdots$$

~0.5 ~0.16 ~0.05

→ ~ 0.7 ≃ η*0.5 = 1.4*0.5

→ cioè l'errore totale (<u>in questo caso</u>) è $\frac{1}{2}h^2 y''(x_0) + 40\%$

Metodo di Eulero – Local Truncation Error

$$y'(x) = f\left(x, y(x)\right) = y(x) \qquad y(x_0) = y_0 = 1$$

$$\rightarrow \ \mathrm{LTE} = y(x_0 + h) - y_1 = \frac{1}{2} h^2 y''(x_0) + O(h^3)$$

Per bassi *h*, l'errore è proporzionale a h^2

n	e^n-e^{n-1}	y_n-y_{n-1}	Δ
1	~1.7	1	~0.7
2	~4.7	2	~2.7
3	~12.7	4	~8.7
4	~34.5	8	~26.5

→ OK: il primo termine "torna"!

E i termini successivi, y_{n+1}?

"Soffrono" di tre errori:

a) l'incremento, $hy'(x_n)$, è un'approssimazione (come per il I° termine)
b) si incrementa a partire da un valore, y_n, già "approssimato";
c) per la derivata utilizziamo l'equazione da risolvere ($y'=f(x,y)$ vs. $(e^x)'=e^x$), che è funzione di y_n (cfr. punto b) invece che di x_n (esatto)

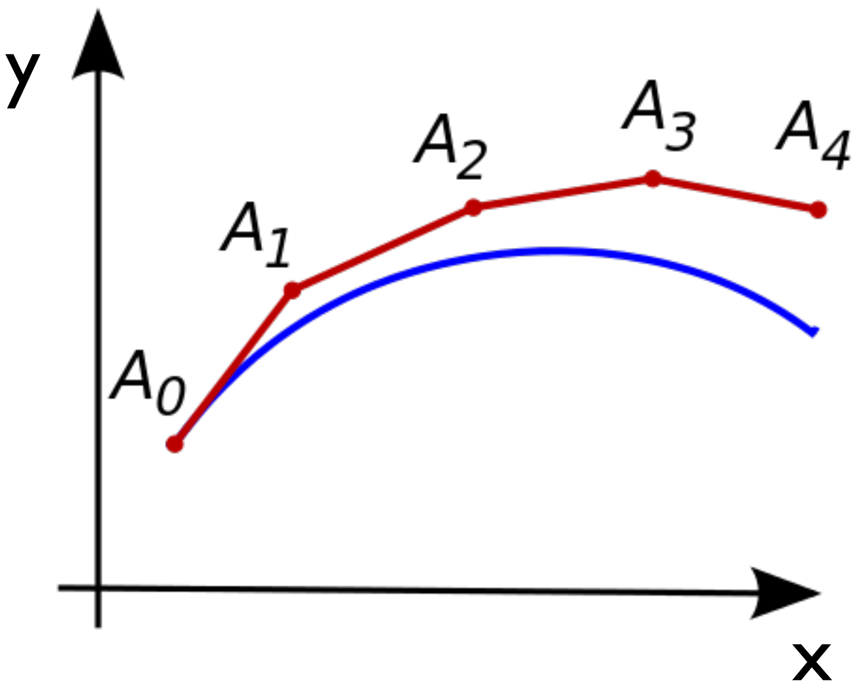
Metodo di Eulero – Local Truncation Error

$$y'(x) = f\left(x, y(x)\right) = y(x) \qquad y(x_0) = y_0 = 1$$

$$\rightarrow \; \mathrm{LTE} = y(x_0 + h) - y_1 = \frac{1}{2}h^2 y''(x_0) + O(h^3)$$

Per bassi *h*, l'errore è proporzionale a h^2

n	e^n-e^{n-1}	y_n-y_{n-1}	Δ
1	~1.7	1	~0.7
2	~4.7	2	~2.7
3	~12.7	4	~8.7
4	~34.5	8	~26.5



Cioè, in poche, parole, i successivi step "soffrono" anche degli errori fatti negli step precedenti

Metodo di Eulero – Local Truncation Error

$$y'(x) = f\left(x, y(x)\right) = y(x) \qquad y(x_0) = y_0 = 1$$

$$\rightarrow \; \mathrm{LTE} = y(x_0 + h) - y_1 = \frac{1}{2} h^2 y''(x_0) + O(h^3)$$

Per bassi *h*, l'errore è proporzionale a h^2

n	e^n-e^{n-1}	y_n-y_{n-1}	Δ
1	~1.7	1	~0.7
2	~4.7	2	~2.7
3	~12.7	4	~8.7
4	~34.5	8	~26.5

Andiamo, quindi, a guardare l'errore commesso nei termini successivi:

-solamente sull'incremento (cioè "rimuovendo" il punto b), *hf(x,y)*

-calcolando la derivata "vera" (cioè "rimuovendo" il punto c)

$$\cancel{h \cdot f\left(x_n, y(x_n)\right) = h \cdot y'(x_n) = h \cdot y_n}$$

$$h \cdot f\left(x_n, y(x_n)\right) = h \cdot y'(x_n) = h \cdot e^{x_n}$$

Metodo di Eulero – Local Truncation Error

$$y'(x) = f\left(x, y(x)\right) = y(x) \qquad y(x_0) = y_0 = 1$$

$$\rightarrow \ \mathrm{LTE} = y(x_0 + h) - y_1 = \frac{1}{2}h^2 y''(x_0) + O(h^3)$$

Per bassi *h*, l'errore è proporzionale a h^2

n	e^n-e^{n-1}	y_n-y_{n-1}	Δ
1	~1.7	1	~0.7
2	~4.7	2	~2.7
3	~12.7	4	~8.7
4	~34.5	8	~26.5

n	e^n-e^{n-1}	h e^{n-1}	Δ
1	~1.7	1	~0.7
2	~4.7	~2.7	~2
3	~12.7	~7.4	~5.3
4	~34.5	~20.1	~14.4

Metodo di Eulero – Local Truncation Error

$$y'(x) = f(x, y(x)) = y(x) \qquad y(x_0) = y_0 = 1$$

$$\rightarrow \ \mathrm{LTE} = y(x_0 + h) - y_1 = \frac{1}{2}h^2 y''(x_0) + O(h^3)$$

Per bassi *h*, l'errore è proporzionale a h^2

n	e^n-e^{n-1}	h e^{n-1}	Δ
1	~1.7	1	~0.7
2	~4.7	~2.7	~2
3	~12.7	~7.4	~5.3
4	~34.5	~20.1	~14.4

½ h^2 y''	1.4*½ h^2 y''	
0.5*1	0.7	→ OK
~0.5*2.7	~1.9	→ OK
~0.5*7.4	~5.2	→ OK
~0.5*20.1	~14.2	→ OK

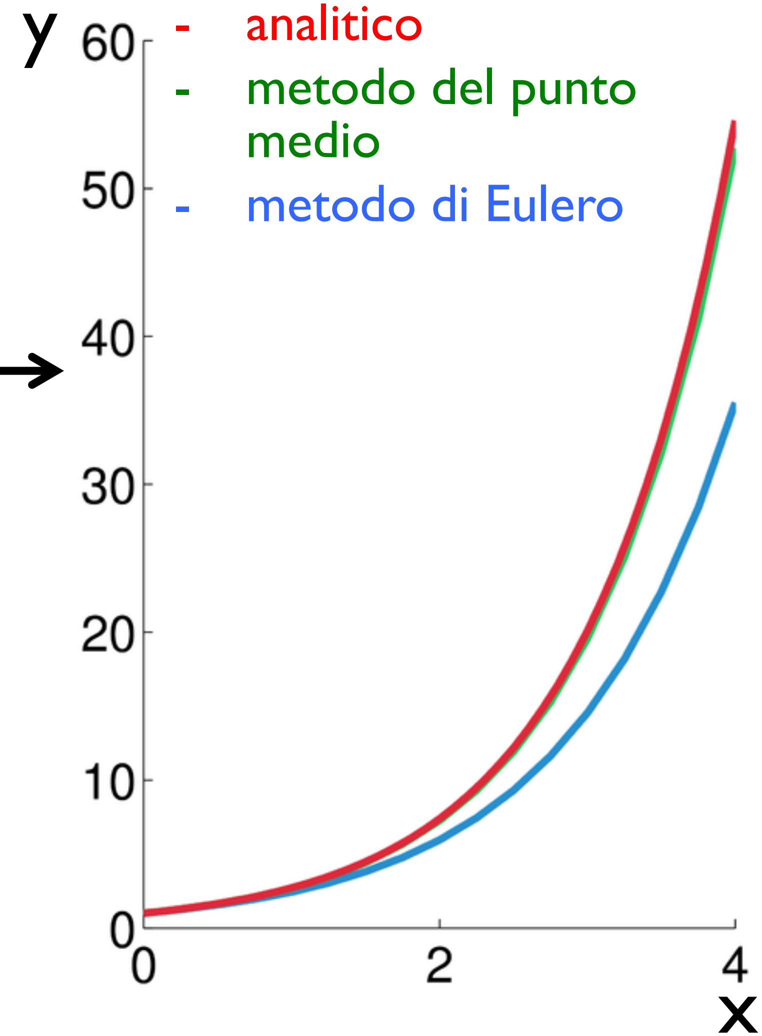
L'errore che si è commesso in questo esempio "torna" con la formula ricavata (<u>il "+40%" è valido solo in questo esempio specifico</u>)

Metodo di Eulero

$$y'(x) = f(x, y(x)) = y(x) \qquad y(x_0) = y_0 = 1$$

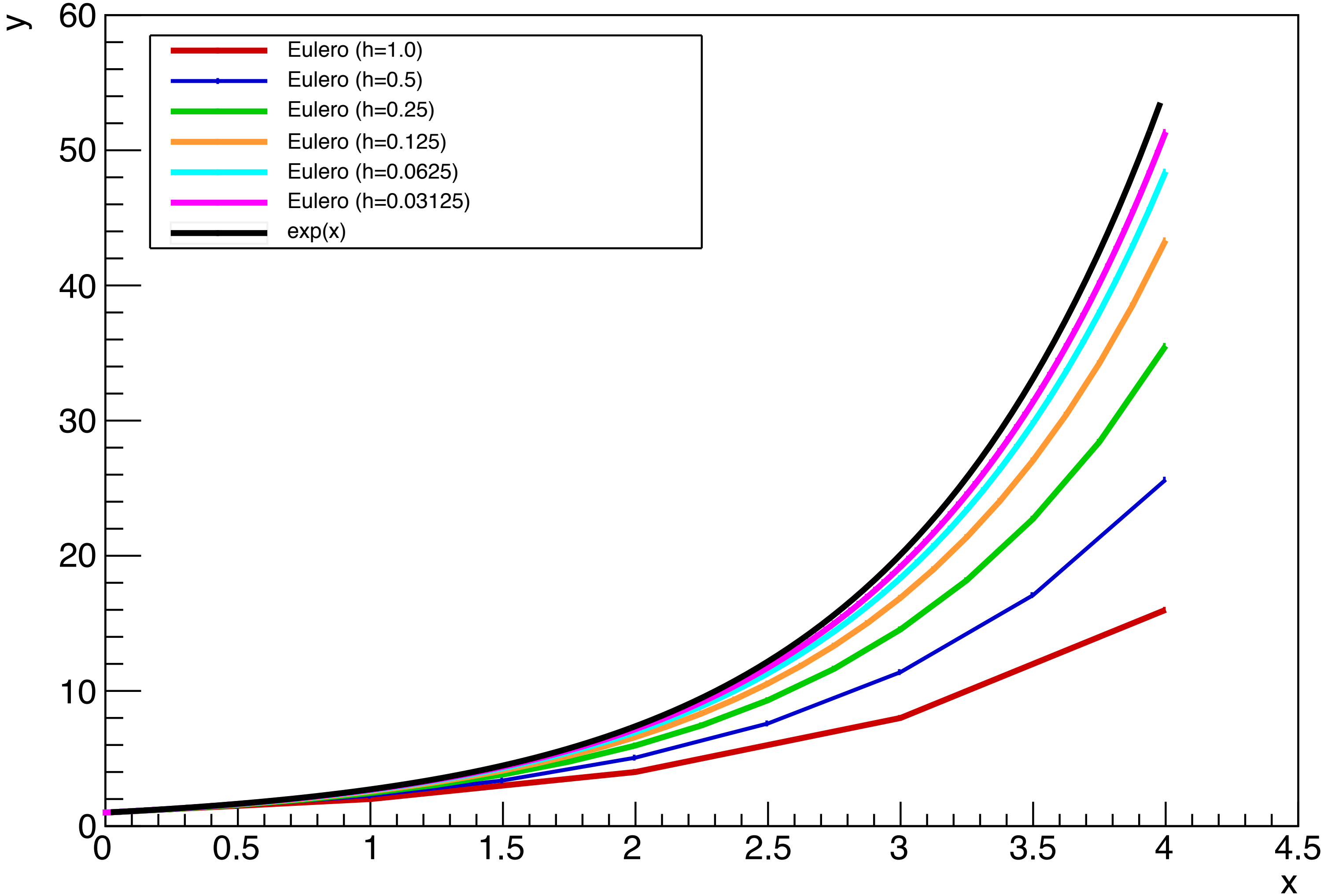
riducendo il passo,
l'accuratezza migliora

h	e^4	Eulero	Δ
1	~54.598	16	~40
0.25	~54.598	~35.53	~19
0.1	~54.598	~45.26	~9
0.05	~54.598	~49.56	~5
0.025	~54.598	~51.98	~2.6
0.0125	~54.598	~53.26	~1.3



Metodo di Eulero

$$y'(x) = f\left(x, y(x)\right) = y(x) \qquad y(x_0) = y_0 = 1$$



Metodo di Eulero – Global Truncation Error

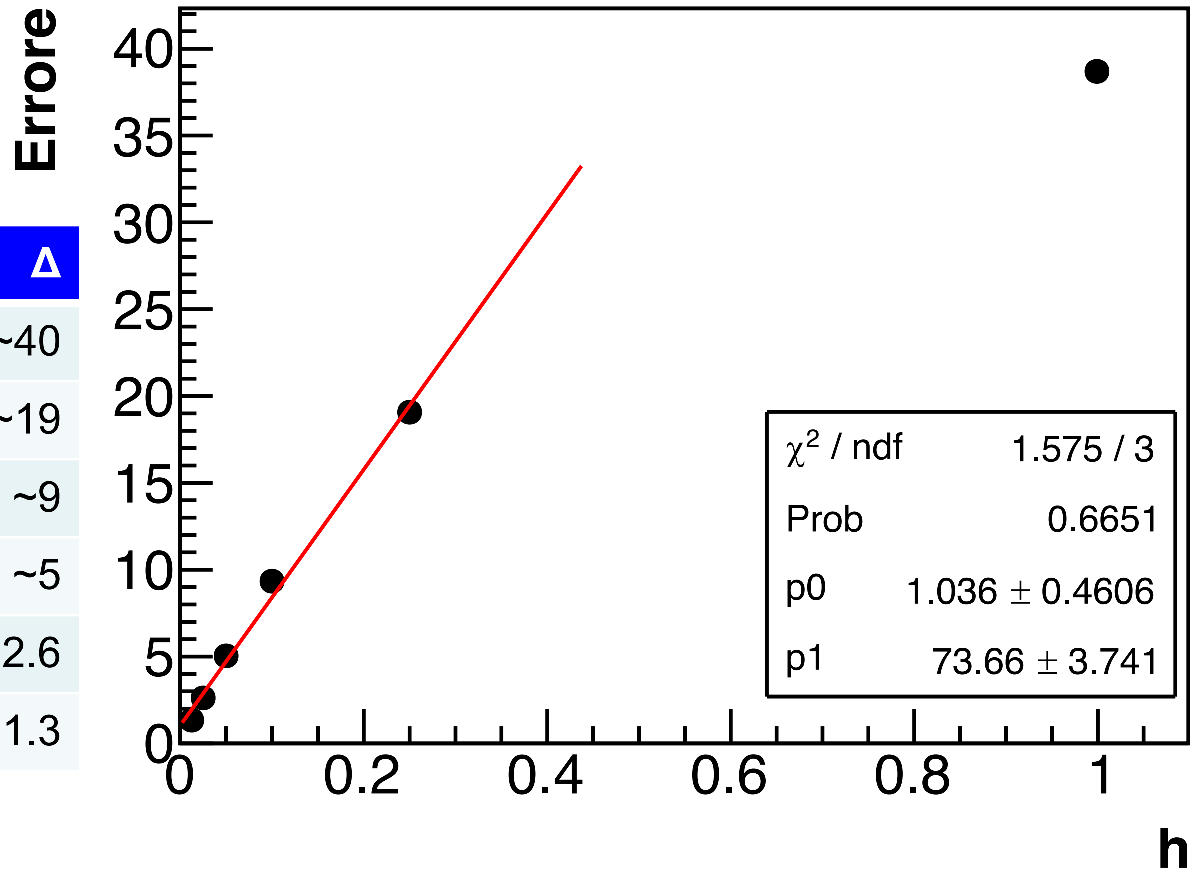
$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

- il numero di step è: $(x - x_0)/h$ che è proporzionale a h^{-1}
- l'errore in ogni step è proporzionale a h^2

→ ci si può convincere facilmente che l'errore totale è <u>proporzionale a</u> *h*

Metodo di Eulero – Global Truncation Error

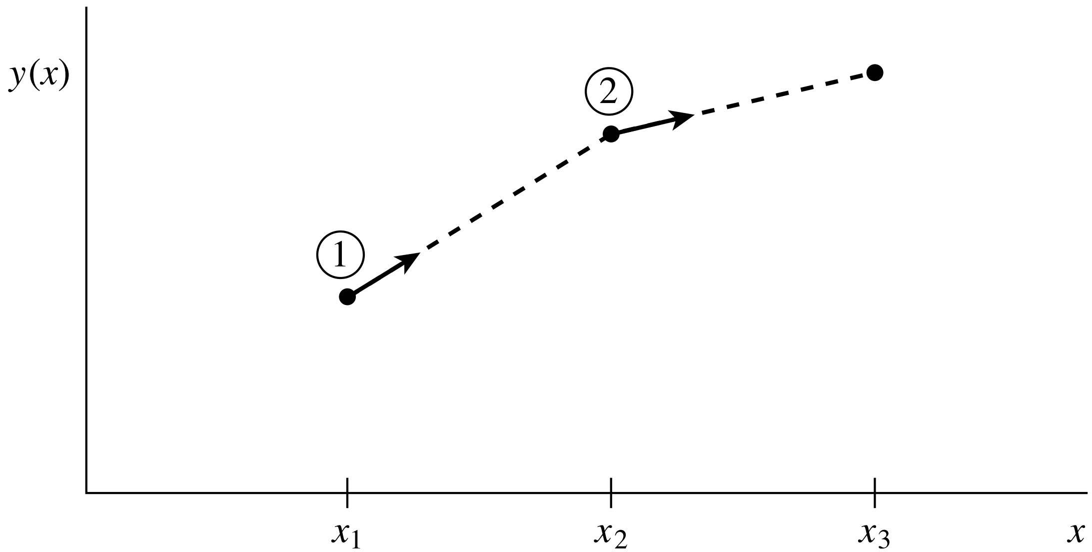
h	e^4	Eulero	Δ
1	~54.598	16	~40
0.25	~54.598	~35.53	~19
0.1	~54.598	~45.26	~9
0.05	~54.598	~49.56	~5
0.025	~54.598	~51.98	~2.6
0.0125	~54.598	~53.26	~1.3



OK: almeno per bassi *h*, l'errore è proporzionale a *h*

Metodo di Eulero

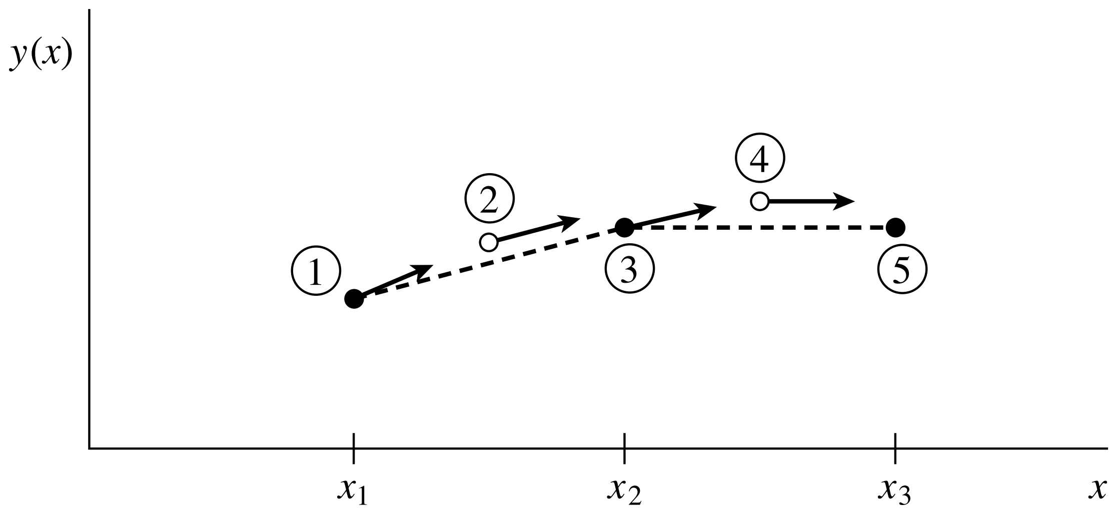
Un “punto” debole del metodo di Eulero è quello di non essere simmetrico rispetto allo step:



viene utilizzata la “direzione” (i.e la derivata), solamente nel punto iniziale dello step

Metodo del punto medio

Nel metodo del punto medio si utilizza anche l'informazione (della derivata) in un punto centrale allo step



Un metodo è quello di utilizzare la derivata nel punto iniziale (*1* e *3*), per stimare la *y* nel punto medio (*2* e *4*) e usare questa per valutare *y'(x+h/2)*, che viene utilizzata come derivata per arrivare all'inizio dello step successivo (*3* e *5*)

Metodo del punto medio

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

Per poter calcolare

$y'(x_n+h/2)$

dobbiamo di nuovo utilizzare l'equazione

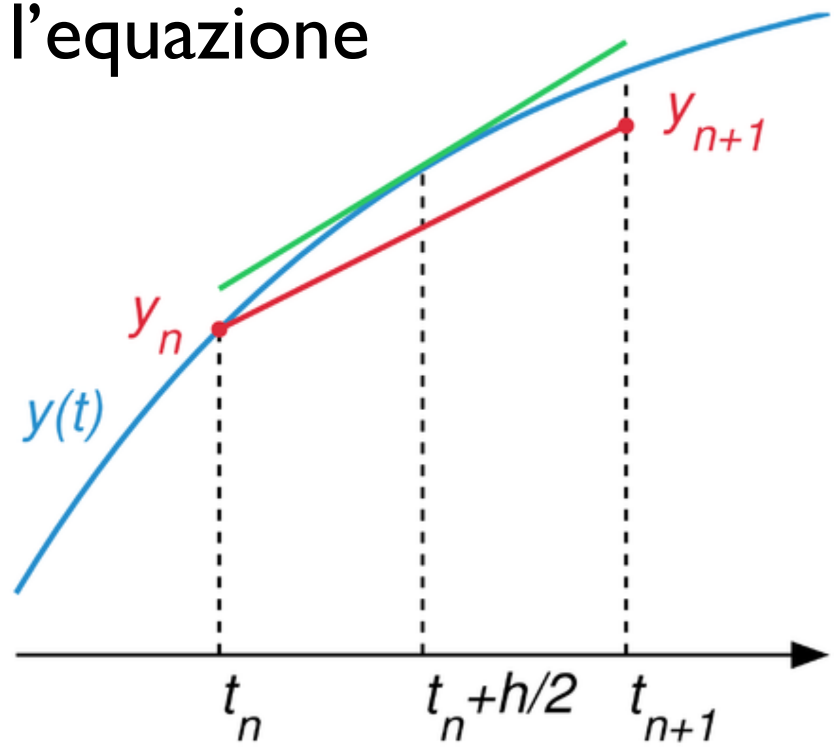
da risolvere

$y' = f(x, y(x))$

cioè ci serve di sapere

$y(x_n+h/2)$

Come?



Metodo del punto medio

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

Possiamo fare due scelte:

- calcolare la $y(x_0+h/2)$ usando la derivata nel punto iniziale (i.e. il metodo di Eulero):

$$y_{n+1} = y_n + hf\left(x_n + \frac{h}{2}, y_n + \frac{h}{2} f(x_n, y_n)\right)$$

- calcolare il punto medio su y, sul quale valutare la derivata, che implica usare una formula "ricorsiva":

$$y_{n+1} = y_n + hf\left(x_n + \frac{h}{2}, \frac{1}{2}(y_n + y_{n+1})\right)$$

Metodo del punto medio

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

Possiamo fare due scelte:

- metodo <u>esplicito</u> del punto medio (o di Eulero modificato):

$$y_{n+1} = y_n + hf\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}f(x_n, y_n)\right)$$

- metodo <u>implicito</u> del punto medio:

$$y_{n+1} = y_n + hf\left(x_n + \frac{h}{2}, \frac{1}{2}(y_n + y_{n+1})\right)$$

Metodo del punto medio

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

Si dimostra che, a causa della simmetria del metodo, rispetto a *x*, tutti i termini di grado pari in *h* (i.e. h^2), dell'errore locale, si cancellano e quindi l'errore locale del metodo è di

$$O(h^3)$$

quindi, per $h \to 0$, l'errore diminuisce più rapidamente che nel metodo di Eulero

Metodo del punto medio - esempio

$$y'(x) = f\left(x, y(x)\right) \boxed{= y(x)} \qquad y(x_0) = y_0 \boxed{= 1}$$

- metodo <u>esplicito</u> del punto medio:

$$y_{n+1} = y_n + hf\left(x_n + \frac{h}{2}, y_n + \frac{h}{2} f(x_n, y_n)\right)$$

$$y_{n+1} = y_n + h\left(y_n + \frac{h}{2} f(x_n, y_n)\right) = y_n + hy_n + \frac{h^2}{2} y_n$$

- metodo <u>implicito</u> del punto medio:

$$y_{n+1} = y_n + hf\left(x_n + \frac{h}{2}, \frac{1}{2}(y_n + y_{n+1})\right)$$

$$y_{n+1} = y_n + h\left(\frac{1}{2}(y_n + y_{n+1})\right) = y_n + \frac{h}{2} y_n + \frac{h}{2} y_{n+1}$$

$$y_{n+1} = \frac{1 + \frac{h}{2}}{1 - \frac{h}{2}} y_n$$

Metodo del punto medio

$$y'(x) = f\,(x, y(x)) = y(x) \qquad y(x_0) = y_0 = 1$$

- metodo esplicito del punto medio, *h*=*1*:

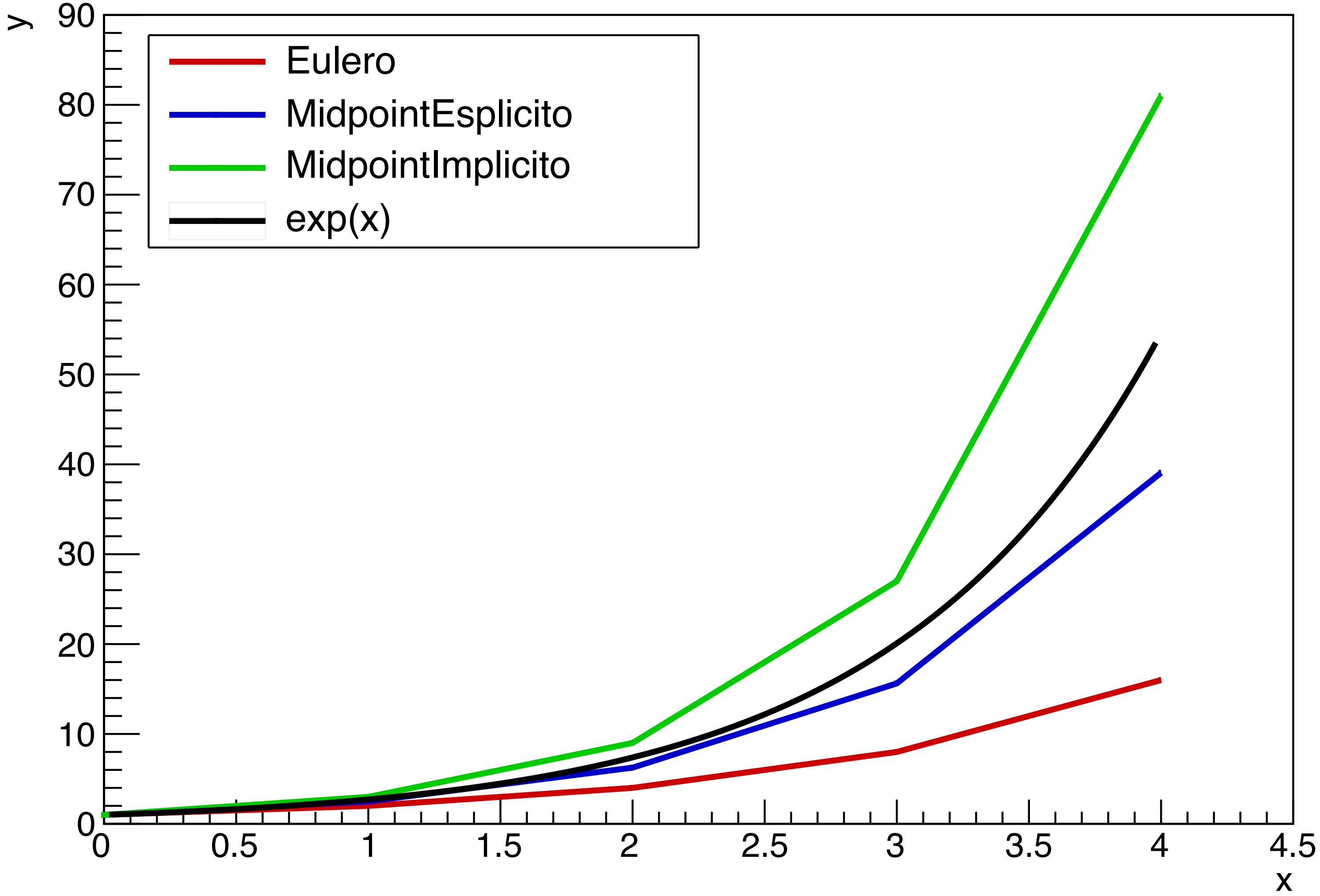
$$y_{n+1} = 2.5\ y_n$$

- metodo implicito del punto medio, *h*=*1*:

$$y_{n+1} = 3\ y_n$$

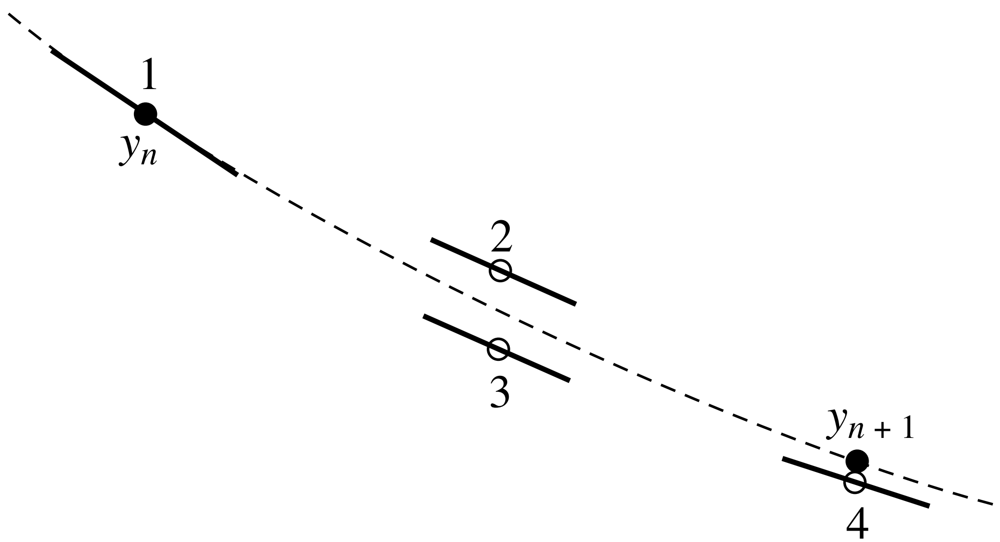
Metodo del punto medio

$$y'(x) = f\left(x, y(x)\right) = y(x) \qquad y(x_0) = y_0 = 1$$



Metodo Runge-Kutta (RK4)

Nel metodo Runge-Kutta (in realtà: Runge-Kutta del quarto ordine, “RK4”)



Si utilizza la derivata nel punto iniziale, due volte la derivata nel punto medio (ciascuna stimata con una diversa approssimazione e pesata il doppio) e la derivata nel punto finale

Metodo Runge-Kutta del 4° ordine (RK4)

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

La funzione alla fine dello step (*n+1*)-esimo è data, al solito, da quella *n*-esima, incrementata di *h* volte la media pesata (con somma dei pesi =6) di *4* diverse valutazioni della derivata *y'=f(x,y)*

$$y_{n+1} = y_n + \frac{h}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

Con:

$$k_1 = f(x_n, y_n)$$

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

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

$$k_4 = f\left(x_n + h, y_n + hk_3\right)$$

Metodo Runge-Kutta del 4° ordine (RK4)

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + \frac{h}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

- $k_1 = f(x_n, y_n)$ ← è la derivata utilizzata nel metodo di Eulero
- $k_2 = f\left(x_n + \frac{h}{2}, y_n + h\frac{k_1}{2}\right)$ ← è la derivata utilizzata nel metodo del punto medio esplicito
- $k_3 = f\left(x_n + \frac{h}{2}, y_n + h\frac{k_2}{2}\right)$ ← è la derivata calcolata nel punto medio utilizzando il metodo del punto medio per stimarla
- $k_4 = f\left(x_n + h, y_n + hk_3\right)$ ← è la derivata nel punto di arrivo utilizzando la seconda stima della derivata nel punto medio

Metodo Runge-Kutta del 4° ordine (RK4)

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + \frac{h}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

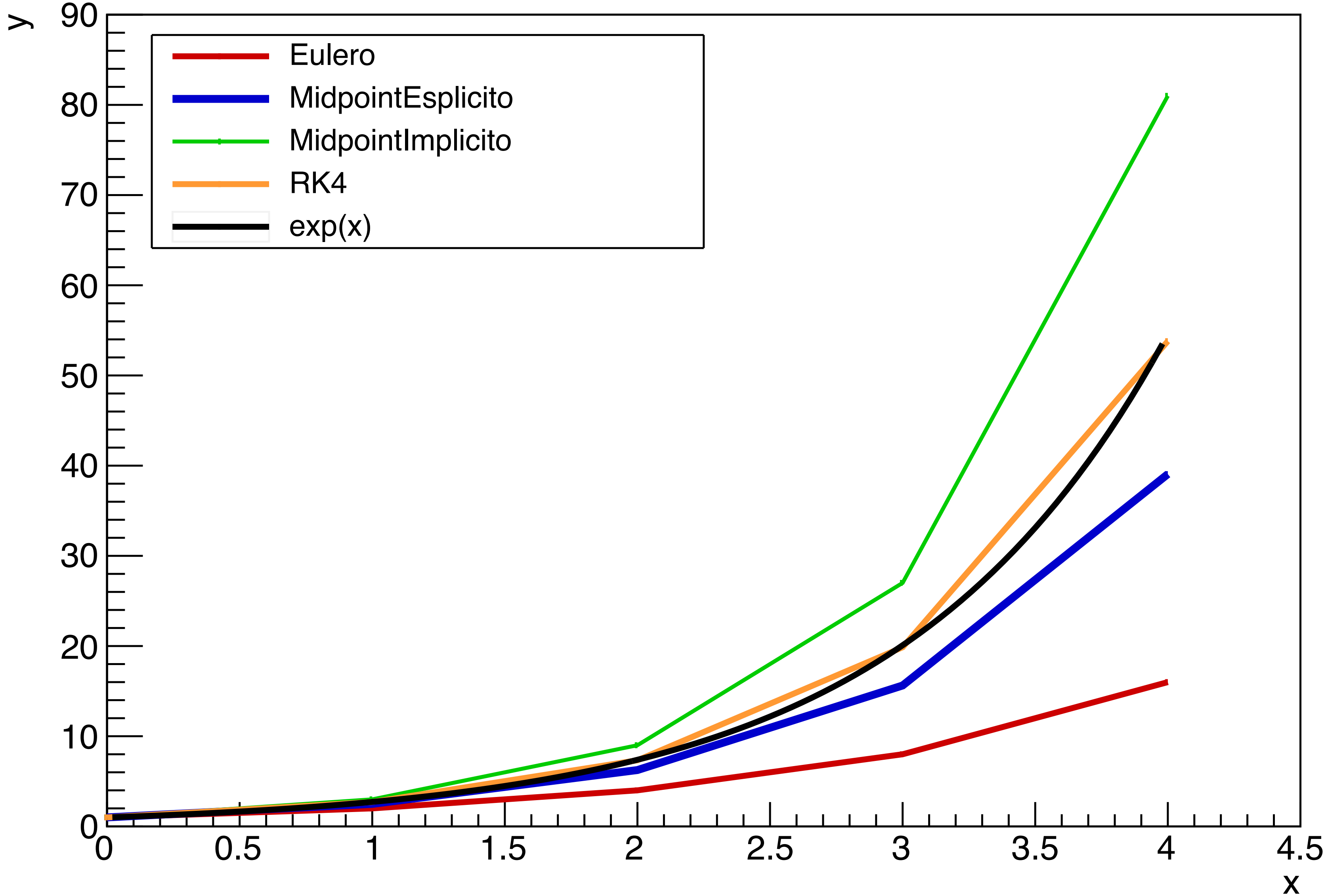
- il metodo è esplicito
- la media pesata è fatta dando maggior peso (doppio) ai due termini con la derivata nel punto medio
- se la *f(x,y)* è indipendente da *y* (*f(x,y)=f(x)*), cioè il problema si riconduce ad un semplice integrale, il metodo coincide con la *formula di Simpson* (cfr.: *A=1/3*, *B=4/3*, C=1/3)
- l'errore <u>locale</u> è di ordine

$$O(h^5)$$

- l'errore <u>globale</u>, al solito, è una potenza di *h* peggiore dell'errore locale (ecco perchè Runge-Kutta del *quarto* ordine):

$$O(h^4)$$

Metodo Runge-Kutta del 4° ordine (RK4)



Metodi Runge-Kutta espliciti

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

Il RK4 è solamente uno di una famiglia di metodi Runge-Kutta espliciti, generalizzabili come:

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i k_i$$

Con:

$$k_1 = f(x_n, y_n)$$

$$k_2 = f(x_n + c_2 h, y_n + h(a_{21} k_1))$$

$$k_3 = f(x_n + c_3 h, y_n + h(a_{31} k_1 + a_{32} k_2))$$

$$\vdots$$

$$k_s = f(x_n + c_s h, y_n + h(a_{s1} k_1 + a_{s2} k_2 + \cdots + a_{s,s-1} k_{s-1}))$$

Metodi Runge-Kutta espliciti

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i$$

$$k_1 = f(x_n, y_n)$$

$$k_i = f\left(x_n + c_i h, y_n + h\sum_{j=1}^{i-1} a_{ij}k_j\right)$$

Il valori della matrice *[a_{ij}]*, dei pesi b_i e dei nodi c_i, sono spesso "tabulati" sotto forma di *Butcher tableau* →

$$\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & & \ddots & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s
\end{array}$$

Metodi Runge-Kutta espliciti

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i$$

$$k_1 = f(x_n, y_n)$$

$$k_i = f\left(x_n + c_i h, y_n + h\sum_{j=1}^{i-1} a_{ij} k_j\right)$$

Il metodo è consistente (*) se:

- $\sum_{i=1}^{s} b_i = 1$
- $\sum_{j=1}^{i-1} a_{ij} = c_i$ per $i = 2, \ldots, s.$

$$\longrightarrow \quad \begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & & \ddots & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s
\end{array}$$

(la derivata numerica approssima quella analitica)*

Metodi Runge-Kutta espliciti

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i$$

$$k_1 = f(x_n, y_n)$$

$$k_i = f\left(x_n + c_i h, y_n + h\sum_{j=1}^{i-1} a_{ij}k_j\right)$$

$$\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & & \ddots & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s
\end{array}
\longrightarrow
\begin{array}{c|cccc}
0 & & & & \\
1/2 & 1/2 & & & \\
1/2 & 0 & 1/2 & & \\
1 & 0 & 0 & 1 & \\
\hline
 & 1/6 & 1/3 & 1/3 & 1/6
\end{array}$$

RK4

Metodi Runge-Kutta espliciti

$$y'(x) = f\left(x, y(x)\right) \qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i$$

$$k_1 = f(x_n, y_n)$$

$$k_i = f\left(x_n + c_i h, y_n + h\sum_{j=1}^{i-1} a_{ij} k_j\right)$$

$$\begin{array}{c|ccccc} 0 & & & & & \\ c_2 & a_{21} & & & & \\ c_3 & a_{31} & a_{32} & & & \\ \vdots & \vdots & & \ddots & & \\ c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\ \hline & b_1 & b_2 & \cdots & b_{s-1} & b_s \end{array} \longrightarrow$$

Eulero

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

Metodi Runge-Kutta espliciti

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i$$

$$k_1 = f(x_n, y_n)$$

$$k_i = f\left(x_n + c_i h, y_n + h\sum_{j=1}^{i-1} a_{ij} k_j\right)$$

$$\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & & \ddots & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s
\end{array}
\longrightarrow$$

Punto medio esplicito

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

Metodi Runge-Kutta

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i k_i$$

$$k_1 = f(x_n, y_n)$$

$$k_i = f\left(x_n + c_i h, y_n + h \sum_{j=1}^{i-1} a_{ij} k_j\right)$$

Il *Butcher tableau* è utilizzabile anche per i metodi non espliciti: per quelli espliciti però, la matrice ha la proprietà di essere "triangolare inferiore"

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

Metodi Runge-Kutta espliciti

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

Un metodo di ordine *N*, quindi, in generale, darà un errore <u>locale</u>

$$LTE \sim \sum_{n=N+1}^{\infty} \frac{1}{n!} h^n \frac{\mathrm{d^n y}}{\mathrm{dx^n}}(x_0)$$

che sapremmo calcolare *a priori* solo se conoscessimo *y(x)* (e la sapessimo derivare…), e un errore <u>globale</u>

$$GTE \propto h^N$$

per calcolare il quale dovremmo calcolare il LTE in ogni *nodo* (l'x_0 nella formula dell'LTE)

→ non è "banale" definire lo step (i.e. *h*) in base all'accuratezza richiesta

Metodi Runge-Kutta adattivi

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

Un modo per ovviare al problema è quello di stimare l'errore durante la "propagazione" e utilizzare questa stima per adattare la step size (i.e. *h*) in base all'accuratezza richiesta

Tipicamente, in un metodo di ordine *N*, con LET proporzionale a *N+1*, e step size *h*, si utilizza <u>anche</u> un secondo metodo, o di ordine più alto o con step size ridotta, e la stima dell'accuratezza viene fornita dalla *discrepanza* fra i due risultati

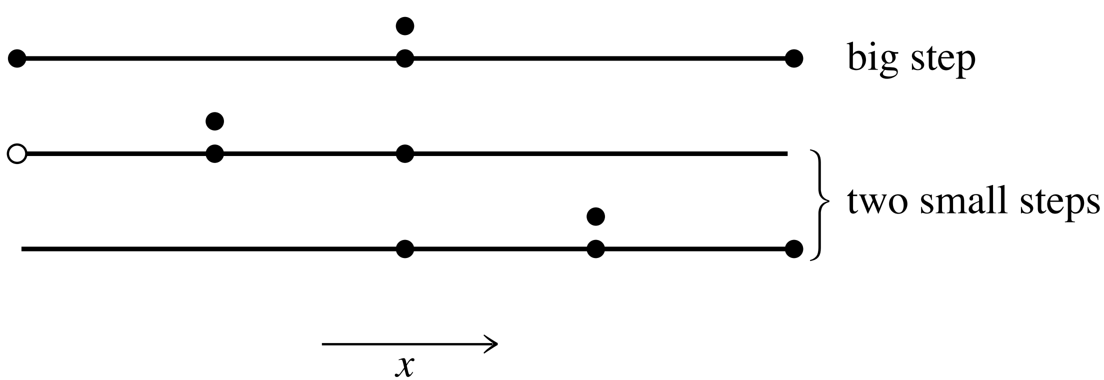
RK4 + step doubling

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

La tecnica più "immediata" è quella della duplicazione dello step (i.e. step doubling), Ogni step viene fatto <u>due</u> volte:

- una volta "normalmente", RK4 con step size 2*h*;

- una volta "in due metà", 2 volte RK4 con step size *h*;



Quanto ci costa "più" computazionalmente? Cioè, quante volte in più dobbiamo valutare *f(x,y)*?

RK4 + step doubling

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

Quanto ci costa "più" computazionalmente? Cioè, quante volte in più dobbiamo valutare *f(x,y)*?

Ogni step ci "costa" 4 valutazioni di *f(x,y)*

- 4 per lo step con step size *2h*;

- 4 per ciascuno (i.e. x2) degli step con step size *h*;

→ 12 valutazioni

In realtà la valutazione iniziale di *f(x,y)*, dello step *2h* e del primo step *h*, è la stessa

→ 11 valutazioni

Questo è da confrontare con le 8 valutazioni complessive dei due step *h* (l'accuratezza guadagnata deve essere <u>almeno</u> quella con step size dimezzata!)

→ 11 rispetto a 8 valutazioni

RK4 + step doubling

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

Come la stimiamo l'accuratezza, però?

Troviamo, di nuovo, una soluzione in serie di Taylor, per i due step:

$$y(x+2h) = y_{2h} + (2h)^5\,\phi + O(h^6)$$
$$y(x+2h) = y_h + 2\,(h)^5\,\phi + O(h^6)$$

dove:

- y_{2h} è la soluzione approssimata, usando uno step *2h*;
- y_h è la soluzione approssimata, usando due step *h*;
- ϕ è di ordine di grandezza (1/5!)*$y^{(5)}$(x);
- la seconda forma ha 2 volte h^5 perchè vengono fatti 2 step;

RK4 + step doubling

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

La differenza fra le due stime

$$\Delta \equiv y_h - y_{2h}$$

è la grandezza da “tenere d’occhio” (i.e. da “aggiustare” in base all’accuratezza richiesta), agendo su *h*.

Quindi in un metodo *adattivo* uno va a “giocare” sullo step size *h*, in modo da portare Δ all’accuratezza richiesta

RK4 + step doubling

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

$$\Delta \equiv y_h - y_{2h}$$

Adesso potremmo pensare di risolvere il sistema (moltiplicando la seconda per 16 e sottrandoci la prima), ovviamente ignorando i termini *O(h*^{6}*)*

$$y(x+2h) = y_{2h} + (2h)^5\,\phi + O(h^6)$$
$$y(x+2h) = y_h + 2\,(h)^5\,\phi + O(h^6)$$

ottenendo così una stima di *y* al <u>quint'ordine</u>:

$$y(x+2h) = y_h + \frac{\Delta}{15} + O(h^6)$$

Questa <u>è</u> un metodo al <u>quint'ordine</u> (i.e. errore *locale* di ordine *h*^{6}), ma di cui <u>non</u> abbiamo controllo sull'errore

RKn + RK_{n-1}

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

L'altro metodo consiste:

- nell'utilizzare un metodo di ordine *n* insieme ad uno di ordine *n-1*;

- utilizzare due metodi che condividano i termini k_i (cioè abbiano gli stessi nodi e le stesse valutazioni di *f(x,y)*), per motivi anche puramente computazionali

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i k_i$$

$$y^*_{n+1} = y_n + h \sum_{i=1}^{s} b^*_i k_i$$

Extended Butcher tableau

$$\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & & \ddots & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s \\
 & b^*_1 & b^*_2 & \cdots & b^*_{s-1} & b^*_s
\end{array}$$

RKn + RK$_{n-1}$

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i$$

$$y^*_{n+1} = y_n + h\sum_{i=1}^{s} b^*_i k_i$$

Il valore di *s* indica quanti “stadi” ha un metodo e, *tipicamente* (ma non è una regola!):

- un metodo con *s* stadi è di ordine *s*
- ogni stadio “non utilizzato” (i.e. $b_i=0$) fa scendere l’ordine di 1 (questo, ad esempio NON è vero per il metodo del midpoint)

Extended Butcher tableau

$$\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & & \ddots & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s \\
 & b^*_1 & b^*_2 & \cdots & b^*_{s-1} & b^*_s
\end{array}$$

RKn + RK$_{n-1}$

$$y'(x) = f\left(x, y(x)\right) \qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i \qquad y^*_{n+1} = y_n + h\sum_{i=1}^{s} b^*_i k_i$$

In questo caso la stima dell'accuratezza è data da:

$$\Delta \equiv y_{n+1} - y^*_{n+1} = h\sum_{i=1}^{s}(b_i - b^*_i)k_i,$$

che è la grandezza da "monitorare" (agendo su *h*) per ottenere l'accuratezza desiderata

Runge-Kutta-Fehlberg

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i \qquad y^*_{n+1} = y_n + h\sum_{i=1}^{s} b^*_i k_i$$

La tecnica è stata sviluppata da Fehlberg che ha identificato un set di parametri (tabulati in un *Butcher tableau*) per avere un metodo con ordine 4 e uno con ordine 5

0						
1/4	1/4					
3/8	3/32	9/32				
12/13	1932/2197	–7200/2197	7296/2197			
1	439/216	–8	3680/513	-845/4104		
1/2	–8/27	2	–3544/2565	1859/4104	–11/40	
ordine 5	16/135	$\boxed{0}$	6656/12825	28561/56430	–9/50	2/55
ordine 4	25/216	$\boxed{0}$	1408/2565	2197/4104	–1/5	$\boxed{0}$

RKn + RK_{n-1} di ordine più basso

$$y'(x) = f\left(x, y(x)\right) \qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i k_i \qquad y^*_{n+1} = y_n + h \sum_{i=1}^{s} b^*_i k_i$$

Il più semplice combina il metodo di Eulero (ordine 1) con il metodo di Heun (ordine 2, come il metodo del punto medio, ma utilizza derivata nel punto iniziale e derivata nel punto finale)

$$\begin{array}{lc|cc} & 0 & & \\ & 1 & 1 & \\ \cline{2-4} \text{ordine 2} \longleftarrow & & 1/2 & 1/2 \\ \text{ordine 1} \longleftarrow & & 1 & \boxed{0} \end{array}$$

RKn + RK$_{n-1}$ Cash-Carp

$$y'(x) = f\left(x, y(x)\right) \qquad\qquad y(x_0) = y_0$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i \qquad y^*_{n+1} = y_n + h\sum_{i=1}^{s} b^*_i k_i$$

Oppure, sempre 4° e 5° ordine, il metodo Cash-Carp:

	0						
	1/5	1/5					
	3/10	3/40	9/40				
	3/5	3/10	−9/10	6/5			
	1	−11/54	5/2	−70/27	35/27		
	7/8	1631/55296	175/512	575/13824	44275/110592	253/4096	
ordine 4		37/378	$\boxed{0}$	250/621	125/594	$\boxed{0}$	512/1771
ordine 5		2825/27648	$\boxed{0}$	18575/48384	13525/55296	277/14336	1/4