Chapter6 Numerical method for ordinary differential equation

Contents

§ 6 Introduction

- Equation consists of the functions and their one or more derivatives is called differential equation. The problem of solving differential equations is often encountered in engineering.

- Differential equation:

Ordinary differential equation(ODE) has one variable

Partial differential equation(PDE) has two or more variables.

§ 6 Introduction

- Consider the first order ODE with initial value problem(IVP) :

$$\begin{cases} \dfrac{dy}{dx} = f(x, y) \quad x \in [a, b] \\ y(x_0) = y_0 \end{cases}$$

formula 1

- Solution:

the function y=y(x) that satisfies the formula 1

§ 6 Introduction

- For most problems, it is very difficult to find an exact solution, so an approximate solution is required.
- It is very hard to analytically express some solution function, thus the numerical solution of the function is needed.
- Numerical solution is generally only required to obtain approximate values on several points or simple approximate expressions of solution (accuracy meets the requirement).

§ 6 Introduction

- Fixed solution problems (2+1 types) :

1. boundary value problem(BVP);
2. IVP;
3. IVP+BVP

- Fixed solution problem refers to the constraint (given the value of variable of function on some point)
- 1. boundary value problem(BVP)

constraint is given on the boundary of the equation:

$$\begin{cases} y'' = f(x, y, y') \\ y(a) = \alpha,\ y(b) = \beta \end{cases}$$

find y=?

The BVP can often transformed into IVP to solve

§ 6 Introduction

- 2. Initial value problem (IVP)

constraint is given on the initial value of the equation:

$$\frac{dy}{dx} = f(x, y), \quad y(a) = y_0,$$
$$x \in [a, b].$$

find y(x)=?

- 3. Initial and boundary value problem(IVP+BVP):

$$u_t - u_{xx} = f, \quad u(\pm 1, t) = 0, u(x,0) = u_0(x).$$

find u(x,t)=?

§ 6 Introduction

common method for IVP

Single-step method:

Use the information of the previous step y_i to find the y_{i+1}, such as:

Euler method, Runge-Kutta method

Multistep method:

Use the information of the previous step y_k, $y_{k+1},\ldots,y_i$ to find the y_{i+1}, such as:

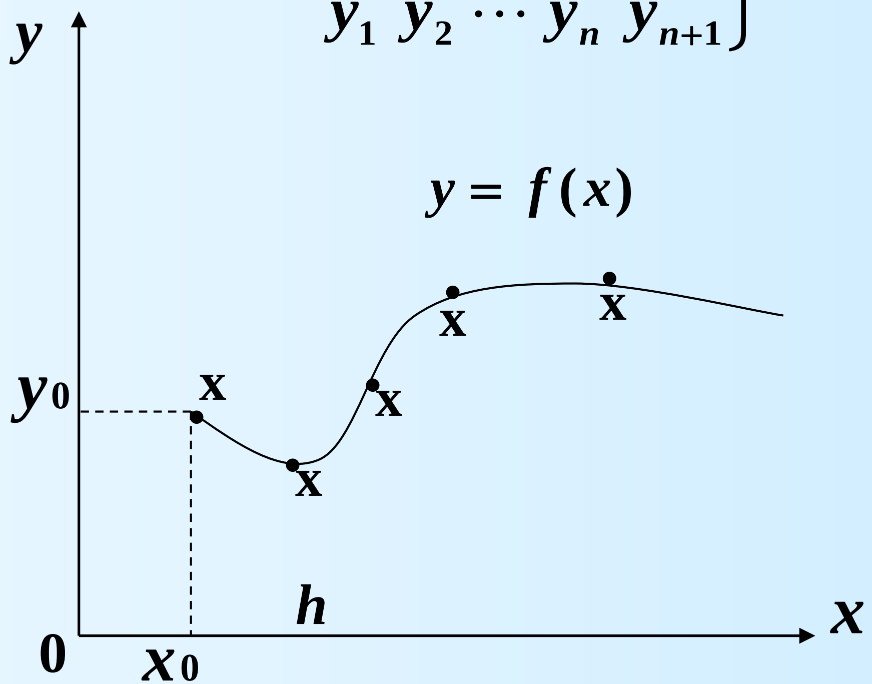
Modified Euler method, Adam method

§ 6 Introduction

Find the solution of formula 1 is to find the value (approximation) of function y(x) on the discrete points:

$$x_0 < x_1 < x_2 < \cdots < x_n < x_{n+1}$$

$$\left.\begin{array}{r} y(x_1)\ y(x_2)\ y(x_n)\ y(x_{n+1}) \\ y_1\ \ y_2\ \cdots\ y_n\ \ y_{n+1} \end{array}\right\} y_n \approx y(x_n)$$



§ 6 Introduction

- IVP：
$$\frac{dy}{dx} = f(x, y), \quad y(a) = y_0, \quad x \in [a, b].$$
find y(x)=?

- Autonomous system：
$$\frac{dz}{dt} = f(z), \quad z(0) = z_0, \quad t \in [0, T].$$
find z(t)=?

- Question: 1. Does the exact solution exist?

  2. How to calculate the numerical solution?

§ 6 Introduction

- Question 1: Does the exact solution exist ?
- Theorem : if function f(x,y) is smooth and continuous, and satisfies the Lipschitz condition, thus the solution of the IVP exists and is unique.

Lipschitz condition :

A function $f(t, y)$ is **Lipschitz continuous** in the variable y on the rectangle $S = [a, b] \times [\alpha, \beta]$ if there exists a constant L (called the **Lipschitz constant**) satisfying

$$|f(t, y_1) - f(t, y_2)| \leq L|y_1 - y_2|$$

for each $(t, y_1), (t, y_2)$ in S. ❒

§ 6 Introduction

- Question 2: How to find the numerical solution?

- IVP: $$\frac{dy}{dx} = f(x, y), \quad y(a) = y_0, \quad x \in [a, b].$$ find y(x)=?

- Base on numerical differentiation

  Euler method

- Base on numerical integration

§ 6.1 Euler method

Set the node： $x_i = a + ih \ \ (i = 0,1,2\cdots,n)$ where： $h = \dfrac{b-a}{n}$

method 1: Taylor expansion

$$y(x_{i+1}) = y(x_i) + y'(x_i)(x_{i+1} - x_i) + \frac{y''(\xi_i)}{2!}(x_{i+1} - x_i)^2$$

$$= y(x_i) + hf(x_i, y(x_i)) + \frac{h^2}{2} y''(\xi_i) \quad \text{or}$$

negligible

$$y_{i+1} = y_i + hf(x_i, y_i) \qquad (i = 0,1,2\cdots,n-1) \quad \text{formula 2}$$

formula 2 is called Euler explicit scheme and can be solved cyclically

§ 6.1 Euler method

- **Explicit Euler method**

derivative approximation ➡ $y'(x_0) \approx \dfrac{y(x_1) - y(x_0)}{h}$

$$y(x_1) \approx y(x_0) + hy'(x_0) = y_0 + h\,f(x_0, y_0) \overset{\text{denote}}{=\!=} y_1$$

$$\boxed{y_{i+1} = y_i + h\,f(x_i, y_i) \quad (i = 0, \ldots, n-1)}$$

§ 6.1 Euler method

➢ **/* Implicit Euler method */**

derivative approximation ➡ $y'(x_1) \approx \dfrac{y(x_1) - y(x_0)}{h}$

➡ $y(x_1) \approx y_0 + h\, f(x_1, y(x_1))$

$$y_{i+1} = y_i + h\, f(x_{i+1}, y_{i+1}) \quad (i = 0, \ldots, n-1)$$

As the unknown y_{i+1} occurs on both side of the equation, it can't be solve directly, thus called implicit Euler method.

§ 6.1 Euler method

- **/* Implicit Euler method */**

derivative approximation → $y'(x_1) \approx \dfrac{y(x_1) - y(x_0)}{h}$

→ $y(x_1) \approx y_0 + h\, f\,(x_1, y(x_1))$

$$y_{i+1} = y_i + h\, f\,(x_{i+1}, y_{i+1}) \quad (i = 0, \dots, n-1)$$

$$y^{m+1} = y_i + hf(x_{i+1}, y^m), m = 0,1,2,\dots, M-1.$$

$$y_{i+1} = y^M$$

Iterative method (usually use explicit scheme to get the initial value)

§ 6.1 Euler method

- **Implicit midpoint formula**

derivative approximation →

$$y'\left(\frac{x_1+x_0}{2}\right) \approx \frac{y(x_1)-y(x_0)}{h}$$

$$y\left(\frac{x_1+x_0}{2}\right) \approx \frac{y(x_1)+y(x_0)}{2}$$

→

$$y(x_1) \approx y(x_0)+hf\left(\frac{x_1+x_0}{2}, y\left(\frac{x_1+x_0}{2}\right)\right)$$

$$\approx y(x_0)+hf\left(\frac{x_1+x_0}{2}, \frac{y(x_1)+y(x_0)}{2}\right)$$

$$\boxed{y_{i+1} \approx y_i+hf\left(\frac{x_{i+1}+x_i}{2}, \frac{y_{i+1}+y_i}{2}\right)}$$

§ 6.1 Euler method

Example: solve the IVP by Euler method

$$\begin{cases} y' = \dfrac{1}{1+x^2} - 2y^2, \ \ 0 \le x \le 2 \\ y(0) = 0 \end{cases}$$

the exact solution of the equation is : $y(x) = x/(1+x^2)$。

Solve: the Euler method gives

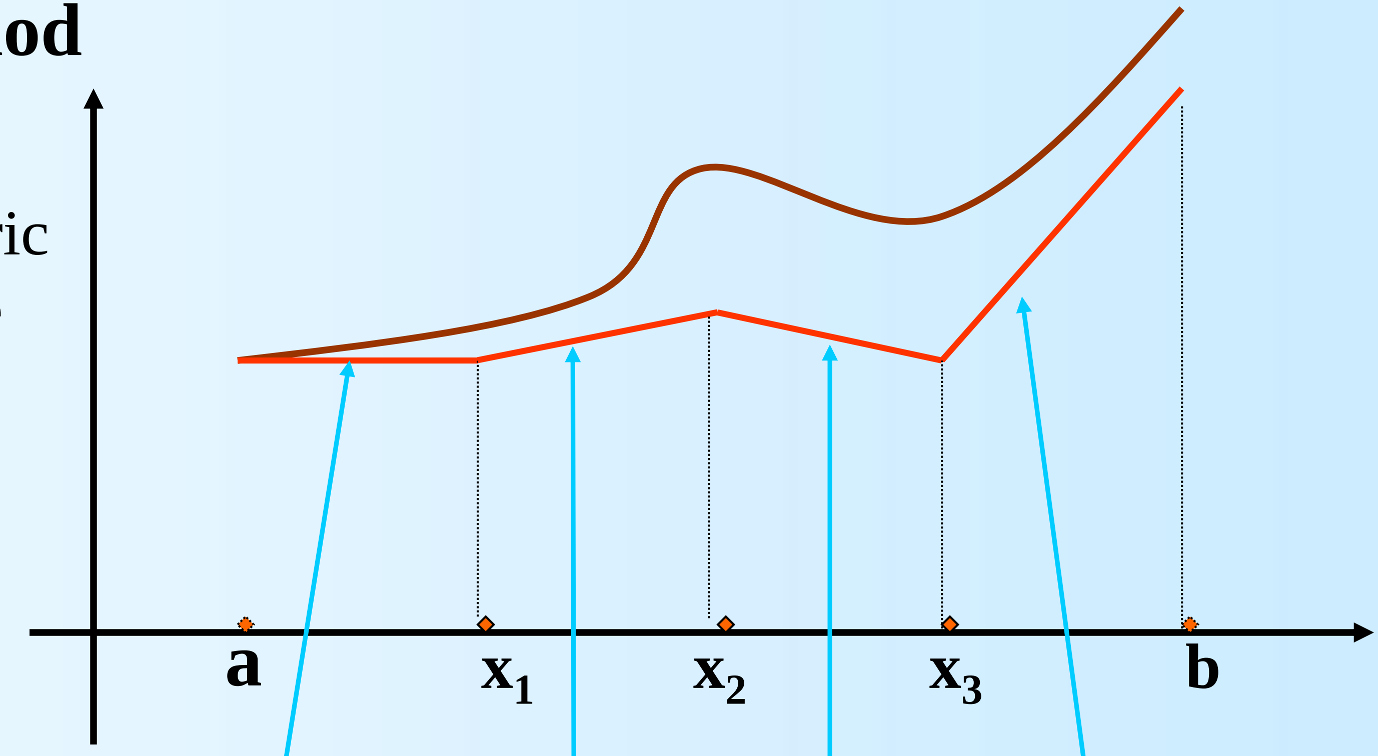
$$\begin{cases} y_{i+1} = y_i + h(\dfrac{1}{1+x_i^2} - 2y_i^2) \\ y_0 = 0 \quad ,i = 0,1,2\cdots \end{cases}$$

take the step size h =0.2, 0.1,0.05, the computation results are

h	x_i	y_i	$y(x_i)$	$y(x_i)-y_i$
h=0.2	0.00	0.00000	0.00000	0.00000
	0.40	0.37631	0.34483	-0.03148
	0.80	0.54228	0.48780	-0.05448
	1.20	0.52709	0.49180	-0.03529
	1.60	0.46632	0.44944	-0.01689
	2.00	0.40682	0.40000	-0.00682
h=0.1	0.00	0.00000	0.00000	0.00000
	0.40	0.36085	0.34483	-0.01603
	0.80	0.51371	0.48780	-0.02590
	1.20	0.50961	0.49180	-0.01781
	1.60	0.45872	0.44944	-0.00928
	2.00	0.40419	0.40000	-0.00419
h=0.05	0.00	0.00000	0.00000	0.00000
	0.40	0.35287	0.34483	-0.00804
	0.80	0.50049	0.48780	-0.01268
	1.20	0.50073	0.49180	-0.00892
	1.60	0.45425	0.44944	-0.00481
	2.00	0.40227	0.40000	-0.00227

Euler method

Geometric structure of Euler method



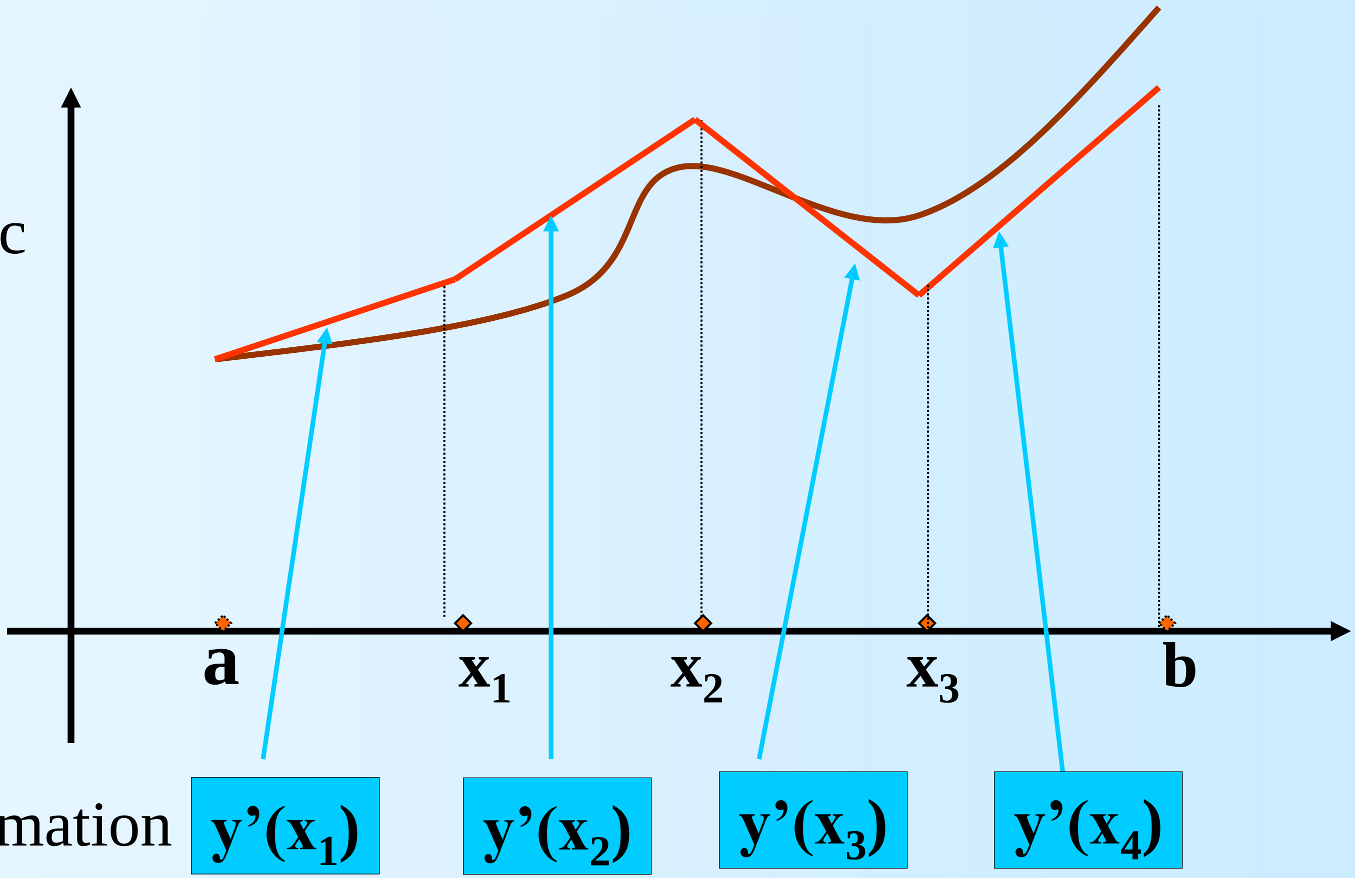
slope approximation $y'(a)$ $y'(x_1)$ $y'(x_2)$ $y'(x_3)$

Explicit Euler method

$$y_{i+1} = y_i + hf(x_i, y_i)$$

$$(i = 0,1,2\cdots,n-1)$$

Geometric structure of Euler method



slope approximation

Implicit Euler method

$$y_{i+1} = y_i + hf(x_{i+1}, y_{i+1})$$

$$(i = 0,1,2\cdots, n-1)$$

§ 6.1 Euler method

- Question 2: How to find the numerical solution?

- IVP: $$\frac{dy}{dx} = f(x,y), \quad y(a) = y_0,$$ $$x \in [a,b].$$ find y(x)=?

- Base on numerical differentiation

- Base on numerical integration

§ 6.1 Euler method

- Numerical integration form:

Integrate the equation $\frac{dy}{dx} = f(x, y)$ on the $[x_i, x_{i+1}]$

$$\int_{x_i}^{x_{i+1}} y' dx = \int_{x_i}^{x_{i+1}} f(x, y(x)) dx$$

$$y(x_{i+1}) = y(x_i) + \int_{x_i}^{x_{i+1}} f(x, y(x)) dx \quad \text{formula 3}$$

use the trapezoid formula to calculate the integral term, then

$$y(x_{i+1}) = y(x_i) + \frac{h}{2}(f(x_i, y_i) + f(x_{i+1}, y_{i+1}))$$

$$y_{i+1} = y_i + \frac{h}{2}(f(x_i, y_i) + f(x_{i+1}, y_{i+1}))$$

supplement:

trapezoidal formula

$$\int_a^b f(x)dx \approx \frac{b-a}{2}[f(a)+f(b)]$$

left rectangle formula

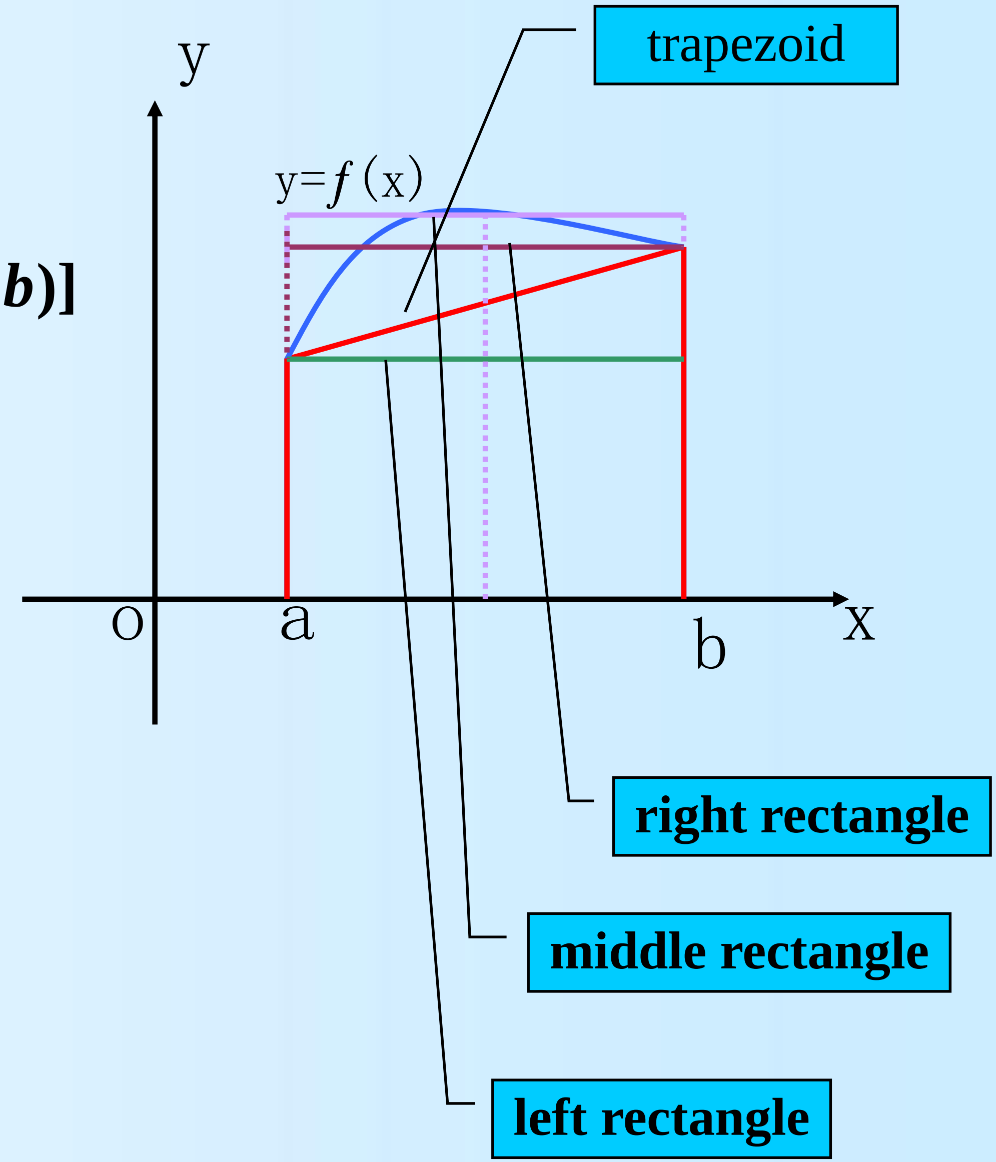
$$\int_a^b f(x)dx \approx (b-a)f(a)$$

right rectangle formula

$$\int_a^b f(x)dx \approx (b-a)f(b)$$

middle rectangle formula

$$\int_a^b f(x)dx \approx (b-a)f\left(\frac{a+b}{2}\right)$$



§ 6.1 Euler method

/* modified Euler's method */ (predictor-corrector method)

***Step 1*: use explicit Euler formula as the predictor**
to calculate $\bar{y}_{i+1} = y_i + h\, f(x_i, y_i)$

***Step 2*: take** $\bar{y}_{i+1}$ **into the implicit Euler formula as the corrector, thus**

$$y_{i+1} = y_i + \frac{h}{2}[\, f(x_i, y_i) + f(x_{i+1}, \bar{y}_{i+1})]$$

$$y_{i+1} = y_i + \frac{h}{2}\left[f(x_i, y_i) + f\big(x_{i+1}, y_i + h\, f(x_i, y_i)\big)\right] \quad (i = 0, \ldots, n-1)$$

§ 6.1 Euler method

modified Euler's method can be also written as

$$
\begin{cases}
y_{i+1} = y_i + \dfrac{h}{2}(K_1 + K_2) \\
K_1 = f(x_i, y_i) \\
K_2 = f(x_i + h, y_i + hK_1) \\
y_0 = \alpha \quad , i = 0,1,2\cdots, n-1
\end{cases}
$$

Example:

Numerically solve the IVP
$$\begin{cases} y' = y - \dfrac{2x}{y}, \quad 0 \le x \le 1 \\ y(0) = 1 \end{cases}$$

take the step size h=0.1 and the exact solution is $y(x) = (1+2x)^{1/2}$

explicit Euler method
$$\begin{cases} y_{i+1} = 1.1y_i - 0.2x_i / y_i \\ y_0 = 1 \quad, i = 0,1,2\cdots,9 \end{cases}$$

modified Euler method
$$\begin{cases} y_{i+1} = y_i + 0.05(K_1 + K_2) \\ K_1 = y_i - 2x_i / y_i \\ K_2 = y_i + 0.1K_1 - \dfrac{2(x_i + 0.1)}{y_i + 0.1K_1} \\ y_0 = 1 \quad, i = 0,1,2\cdots,9 \end{cases}$$

Computation result:

i	x_i	Euler method y_i	Modified Euler y_i	precise $y(x_i)$
0	0	1	1	1
1	0.1	1.1	1.095909	1.095445
2	0.2	1.191818	1.184096	1.183216
3	0.3	1.277438	1.266201	1.264991
4	0.4	1.358213	1.343360	1.341641
5	0.5	1.435133	1.416402	1.414214
6	0.6	1.508966	1.485956	1.483240
7	0.7	1.580338	1.552515	1.549193
8	0.8	1.649783	1.616476	1.612452
9	0.9	1.717779	1.678168	1.673320
10	1	1.784770	1.737869	1.732051

Convergence and stability

- **Truncation error:**

local truncation error global truncation error

$$\begin{cases} y(x_{i+1}) - y_{i+1} = O(h^{p+1}) \\ y(x_i) = y_i, \end{cases} \Rightarrow y(x_n) - y_n = O(h^p)$$

Convergence and stability

- Convergence:

If the method (algorithm) has $y_n \to y(x_n)$ when $h \to 0$ (n $\to \infty$) for any fixed $x_n = x_0 + ih$, then the method (algorithm) is called convergence.

Convergence and stability

- Theorem:

If the single-step method $y_{n+1}=y_n+h\varphi(x_n,y_n,h)$ has p-th order accuracy (local truncation error is $O(h^{p+1})$) and $f(x_n,y_n,h)$ is Lipschitz continuous in y,

Lipschitz condition: $$\left|\varphi(x,y,h)-\varphi(x,\bar{y},h)\right| \le L_{\varphi}\left|y-\bar{y}\right|$$

Initial value y_0 is exact, thus the method is convergence with the global truncation error $O(h^p)$

Local truncation error

- Explicit and implicit Euler method: $O(h^2)$
- Midpoint formula: $O(h^3)$
- Trapezoid formula: $O(h^3)$
- Modified Euler method :$O(h^3)$

Example： $\begin{cases} y'(x) = -30y(x) \\ y(0) = 1 \end{cases}$ **in [0, 0.5],**

Solution of explicit Euler、implicit Euler and modified Euler

node x_i	explicit	implicit	modified	exact $y = e^{-30x}$
0.0	1.0000	1.0000	1.0000	1.0000
0.1	–2.0000	2.5000×10^{-1}	2.5000	4.9787×10^{-2}
0.2	4.0000	6.2500×10^{-2}	6.2500	2.4788×10^{-3}
0.3	–8.0000	1.5625×10^{-2}	1.5626×10^{1}	1.2341×10^{-4}
0.4	1.6000×10^{1}	3.9063×10^{-3}	3.9063×10^{1}	6.1442×10^{-6}
0.5	-3.2000×10^{1}	9.7656×10^{-4}	9.7656×10^{1}	3.0590×10^{-7}

What is wrong ??!

Convergence and stability

- Stability : $|\delta_i| < |\delta_0|, i = 1,2,3 \ldots$

- /* test equation */ $\boxed{y' = \lambda y}$
- for *Euler* method

$$y_{i+1} = y_i + h(\lambda y_i) = (1 + \lambda h) y_i = \cdots = (1 + \lambda h)^{i+1} y_0$$

thus $y_{i+1} + \delta_{i+1} = (1 + \lambda h)^{i+1}(y_0 + \delta_0)$

$$\delta_{i+1} = (1 + \lambda h)^{i+1} \delta_0$$

obvioulsy， *Euler* method $\Leftrightarrow |1 + \lambda h| \le 1$， $h \le -\frac{1}{\lambda}$

- The implicit scheme has good stability

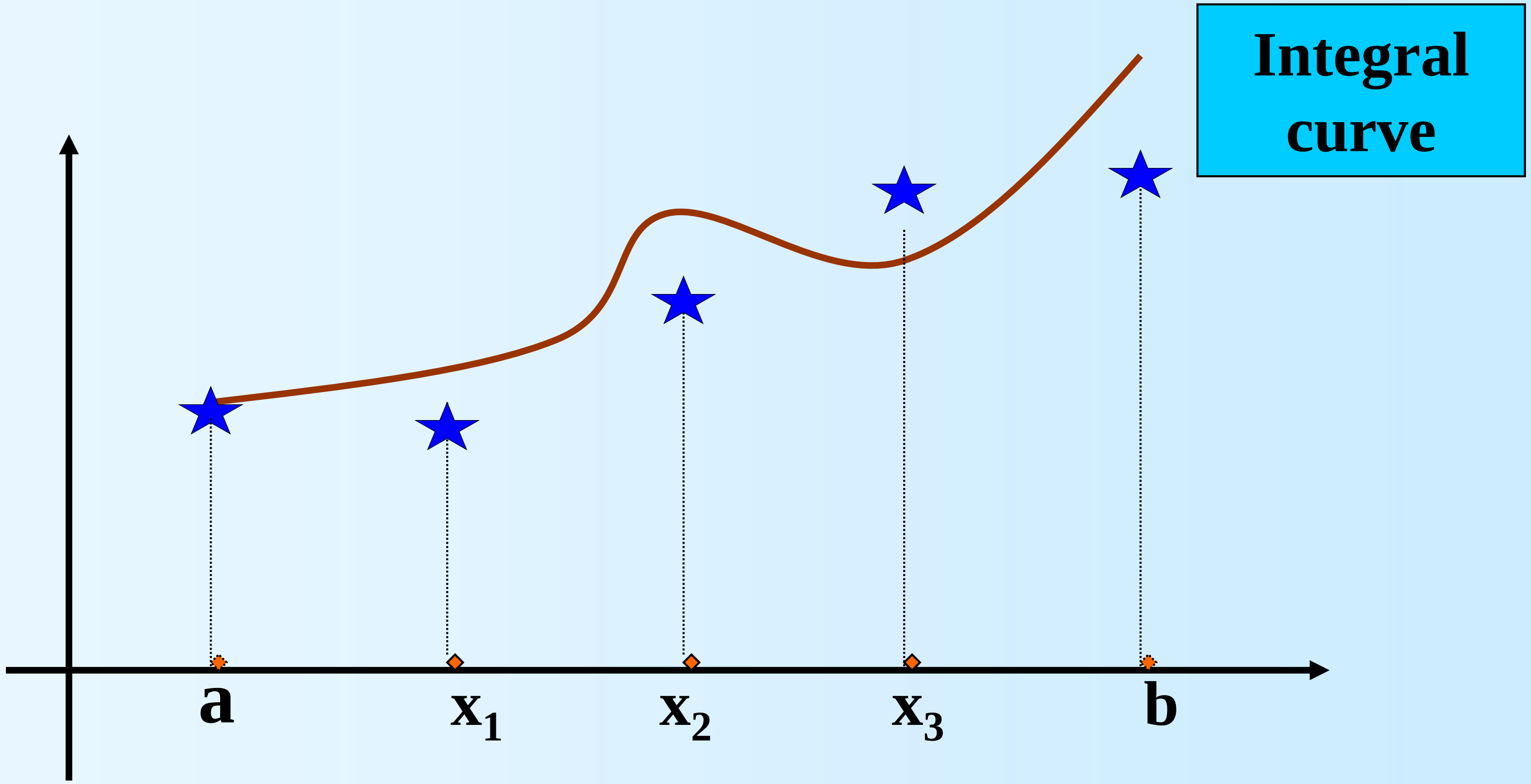
Contents

- §6.2 Runge-Kutta method

Idea of numerical method : uniformly divided $[a,b]$ into n parts with n+1 nodes

$$a = x_0 < x_1 < \cdots < x_{n-1} < x_n = b$$

Calculate the approximation y_k of $y(x_k)$ $(k>0)$



§ 6.2 Runge-Kutta method

- 2^{nd} order Runge-Kutta scheme

Single-step method : use the value on the previous node y_i to obtain the value on the next step y_{i+1} .

Aim： establish a one-step recursive scheme with high numerical precision

The idea is to start from(x_i , y_i) , follow a certain straight line to get (x_{i+1} , y_{i+1}) . The highest accuracy of Euler method and modified Euler method is 2^{nd} order .

§ 6.2 Runge-Kutta method

The modified Euler method

$$\begin{cases} \bar{y}_k = y_{k-1} + hf(x_{k-1}, y_{k-1}) \\ y_k = y_{k-1} + \dfrac{h}{2}[f(x_{k-1}, y_{k-1}) + f(x_k, \bar{y}_k)] \end{cases}$$

change it into another form:

$$\begin{cases} y_{i+1} = y_i + h\left[\dfrac{1}{2}K_1 + \dfrac{1}{2}K_2\right] \\ K_1 = f(x_i, y_i) \\ K_2 = f(x_i + h, y_i + hK_1) \\ y_0 = y(x_0) \end{cases}$$

Always take the average of K_1 and K_2?

one fixed step size h?

Extend the modified Euler formula :

$$\begin{cases} y_{i+1} = y_i + h[\lambda_1 K_1 + \lambda_2 K_2] \\ K_1 = f(x_i, y_i) \\ K_2 = f(x_i + ph, y_i + phK_1) \end{cases}$$

Find the coefficients λ_1、λ_2、p, which enables the formula with 2nd order precision, i.e., under the condition $y_i = y(x_i)$

$$R_i = y(x_{i+1}) - y_{i+1} = O(h^3)$$

***Step 1*: take the Taylor expansion of K_2 at (x_i, y_i)**

$$K_2 = f(x_i + ph, y_i + phK_1)$$

$$= f(x_i, y_i) + phf_x(x_i, y_i) + phK_1 f_y(x_i, y_i) + O(h^2)$$

$$= y'(x_i) + phy''(x_i) + O(h^2)$$

$$\boxed{y''(x) = \frac{d}{dx} f(x, y) = f_x(x, y) + f_y(x, y)\frac{dy}{dx} = f_x(x, y) + f_y(x, y) f(x, y)}$$

§ 6.2 Runge-Kutta method

Step 2: **put K_2 into the formula, then**

$$y_{i+1} = y_i + h\left\{\lambda_1 y'(x_i) + \lambda_2[y'(x_i) + phy''(x_i) + O(h^2)]\right\}$$
$$= y_i + (\lambda_1 + \lambda_2)hy'(x_i) + \lambda_2 ph^2 y''(x_i) + O(h^3)$$

Step 3: **compare the Taylor expansion of y_{i+1} at x_i with that of $y(x_{i+1})$**

$$y_{i+1} = y_i + (\lambda_1 + \lambda_2)hy'(x_i) + \lambda_2 ph^2 y''(x_i) + O(h^3)$$

$$y(x_{i+1}) = y(x_i) + hy'(x_i) + \frac{h^2}{2} y''(x_i) + O(h^3)$$

§ 6.2 Runge-Kutta method

To satisfy: $R_i = y(x_{i+1}) - y_{i+1} = O(h^3)$**, thus we have**

$$\lambda_1 + \lambda_2 = 1, \quad \lambda_2 p = \frac{1}{2}$$

3 unknown number, 2 equation

There are infinite solutions, the formula satisfies that condition is called 2nd order Runge-Kutta scheme.

when $p = 1, \ \lambda_1 = \lambda_2 = \dfrac{1}{2}$ is the modified Euler method

§ 6.2 Runge-Kutta method

- Higher order Runge-Kutta scheme

Question: how to obtain higher order accuracy?

$$\begin{cases} y_{i+1} = y_i + h[\lambda_1 K_1 + \lambda_2 K_2 + \ldots + \lambda_m K_m] \\ K_1 = f(x_i, y_i) \\ K_2 = f(x_i + \alpha_2 h, y_i + \beta_{21} h K_1) \\ K_3 = f(x_i + \alpha_3 h, y_i + \beta_{31} h K_1 + \beta_{32} h K_2) \\ \quad \ldots\ldots \\ K_m = f(x_i + \alpha_m h, y + \beta_{m1} h K_1 + \beta_{m2} h K_2 + \ldots + \beta_{m\,m-1} h K_{m-1}) \end{cases}$$

where λ_i **(** $i = 1, \ldots, m$ **),** α_i **(** $i = 2, \ldots, m$ **)** ***and*** β_{ij} **(** $i = 2, \ldots, m;\ j = 1, \ldots, i-1$ **) are undetermined coefficient.**

§ 6.2 Runge-Kutta method

- 2nd order Runge-Kutta scheme (RK2)

$$\begin{cases} y_{i+1} = y_i + \frac{h}{2}(K_1 + K_2) \\ K_1 = f(x_i, y_i) \\ K_2 = f(x_i + h, y_i + hK_1) \\ y_0 = \alpha \quad , i = 0,1,2\cdots, n-1 \end{cases}$$

§ 6.2 Runge-Kutta method

- 3rd order Runge-Kutta scheme (RK3)

$$\begin{cases} y_{i+1} = y_i + \frac{h}{6}(K_1 + 4K_2 + K_3) \\ K_1 = f(x_i, y_i) \\ K_2 = f(x_i + \frac{h}{2}, y_i + \frac{h}{2}K_1) \\ K_3 = f(x_i + h, y_i + h(2K_2 - K_1)) \\ y_0 = y(x_0) \end{cases}$$

§ 6.2 Runge-Kutta method

- 4th order classical Runge-Kutta scheme (RK4)

$$\begin{cases} y_{i+1} = y_i + \dfrac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ K_1 = f(x_i, y_i) \\ K_2 = f(x_i + \dfrac{h}{2}, y_i + \dfrac{h}{2}K_1) \\ K_3 = f(x_i + \dfrac{h}{2}, y_i + \dfrac{h}{2}K_2) \\ K_4 = f(x_i + h, y_i + hK_3) \\ y_0 = y(x_0) \end{cases}$$

§ 6.2 Runge-Kutta method

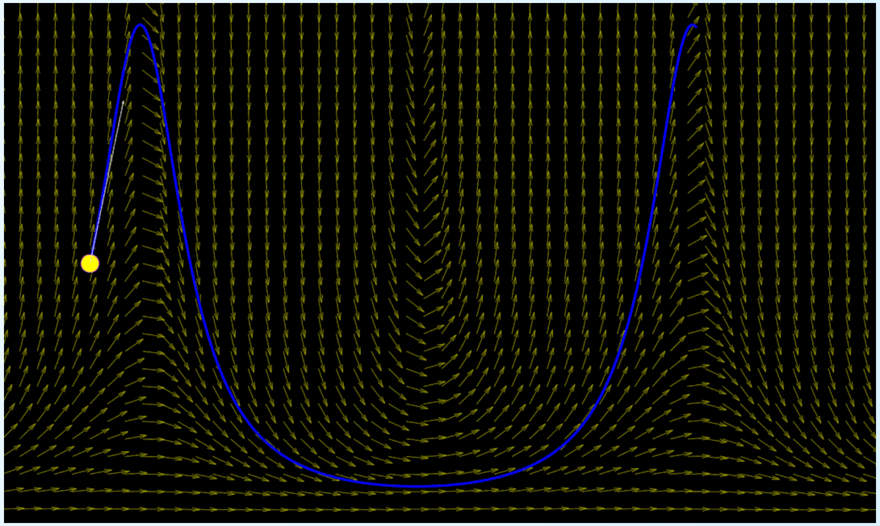
Note:

☞ **Main computation of Runge-Kutta is on the calculation of K_i , i.e. calculation of f . *Butcher* gives the relation of computation step and accuracy in 1965 :**

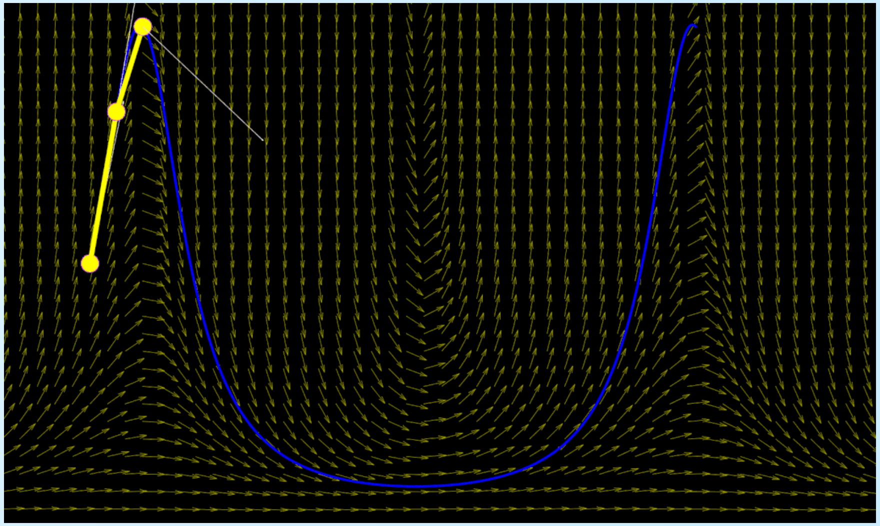
number of K_i to compute	2	3	4	5	6	7	$n \geq 8$
Highest accuracy	$O(h^2)$	$O(h^3)$	$O(h^4)$	$O(h^4)$	$O(h^5)$	$O(h^6)$	$O(h^{n-2})$

☞ **Runge-Kutta method is based on the Taylor expansion, so the accuracy is depends on the smoothness.**

Example：Euler method & RK4



Euler method



4th-order Runge-Kutta

Example： use higher order *R*-*K* method to calculate the IVP

$$\begin{cases} y' = y^2 & 0 \le x \le 0.5 \\ y(0) = 1 \end{cases}$$

take $h = 0.1.$

Solution : (1) use the 3rd *R*-*K* scheme

when n=1, $K_1 = y_0^2 = 1$

$$K_2 = (y_0 + \frac{0.1}{2} K_1)^2 = 1.1025$$

$$K_3 = (y_0 + 0.1(2K_2 - K_1))^2 = 1.2555$$

$$y_1 = y_0 + \frac{0.1}{6}(K_1 + 4K_2 + K_3) = 1.1111$$

The calculation result is:

i	x_i	k_1	k_2	k_3	y_i
1.0000	0.1000	1.0000	1.1025	1.2555	1.1111
2.0000	0.2000	1.2345	1.3755	1.5945	1.2499
3.0000	0.3000	1.5624	1.7637	2.0922	1.4284
4.0000	0.4000	2.0404	2.3423	2.8658	1.6664
5.0000	0.5000	2.7768	3.2587	4.1634	1.9993

(2) use the 4th order *R*-*K* method

when ***n***=1, $$K_1 = y_0^2 = 1$$

$$K_2 = (y_0 + \frac{0.1}{2}K_1)^2 = 1.1025$$

$$K_3 = (y_0 + \frac{0.1}{2}K_2)^2 = 1.1133$$

$$K_4 = (y_0 + 0.1K_3)^2 = 1.2351$$

$$y_1 = y_0 + \frac{0.1}{6}(K_1 + 2K_2 + 2K_3 + K_4) = 1.1111$$

The calculation result is :

i	x_i	k_1	k_2	k_3	k_4	y_i
1.0000	0.1000	1.0000	1.1025	1.1133	1.2351	1.1111
2.0000	0.2000	1.2346	1.3756	1.3921	1.5633	1.2500
3.0000	0.3000	1.5625	1.7639	1.7908	2.0423	1.4286
4.0000	0.4000	2.0408	2.3428	2.3892	2.7805	1.6667
5.0000	0.5000	2.7777	3.2600	3.3476	4.0057	2.0000

Local truncation error

- 2^{nd} Runge-Kutta method: $O(h^3)$
- 3^{rd} Runge-Kutta method : $O(h^4)$
- 4^{th} Runge-Kutta method : $O(h^5)$

§ 6.3 Linear multistep method

- 1. Adams explicit formula
- 2. Adams implicit formula

§ 6.3 Linear multistep method

- 1. Linear multistep method
  - Make use of the known y_n, y_{n-1}, … , and $f(x_n,y_n)$, $f(x_{n-1},y_{n-1})$, … , to develop the differential formula with high accuracy and small computation cost to calculate y_{n+1}
  - Use the linear combination of y and y' at several nodes to approximate $y(x_{i+1})$。

§ 6.3 Linear multistep method

- 1. Linear multistep method
  - The general form can be written as

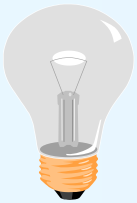
$$f_j = f(x_j, y_j)$$

$$y_{i+1} = \alpha_0 y_i + \alpha_1 y_{i-1} + \ldots + \alpha_k y_{i-k} + h(\beta_{-1} f_{i+1} + \beta_0 f_i + \beta_1 f_{i-1} + \ldots + \beta_k f_{i-k})$$

when $\beta_{-1} \neq 0$, implicit formula; $\beta_{-1} = 0$ explicit formula.

§ 6.3 Linear multistep method

- **Based on numerical integration**



Integrate $y' = f(x, y)$ **in the interval** $[x_i, x_{i+1}]$**, thus**

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

Approximate the integral $I_k \approx \int_{x_i}^{x_{i+1}} f(x, y(x))dx$**, and make use of** $y_{i+1} = y_i + I_k$ **to approximate** $y(x_{i+1})$ **.**

The choice of different approximation to I_k**, the different calculation formulas are obtained.**

§ 6.3 Linear multistep method

- /* Adams explicit formula */

Use the integrand function $f_i, f_{i-1}, \ldots, f_{i-k}$ **on** ***k*+1** **nodes to construct the k-th order Newton interpolation** $N_k(x_i + th)$, $t \in [0, 1]$

$$\int_{x_i}^{x_{i+1}} f(x, y(x))dx = \int_0^1 N_k(x_i + th)h\,dt + \int_0^1 R_k(x_i + th)h\,dt$$

truncation term

$$\Rightarrow \quad y_{i+1} = y_i + h\int_0^1 N_k(x_i + th)dt$$

/* explicit formula*/

local truncation error： $R_i = y(x_{i+1}) - y_{i+1} = h\int_0^1 R_k(x_i + th)dt$

§ 6.3 Linear multistep method

noted: generally $R_i = B_k h^{k+2} y^{(k+2)}(\xi_i)$, where the **coefficients** of $f_i, \ldots, f_{i-k}$ in the calculation of B_k and y_{i+1} can be found the the table

k	f_i	f_{i-1}	f_{i-2}	f_{i-3}	$\cdots$	B_k
0	1					$\frac{1}{2}$
1	$\frac{3}{2}$	$-\frac{1}{2}$				$\frac{5}{12}$
2	$\frac{23}{12}$	$-\frac{16}{12}$	$\frac{5}{12}$			$\frac{3}{8}$
3	$\frac{55}{24}$	$-\frac{59}{24}$	$\frac{37}{24}$	$-\frac{9}{24}$		$\frac{251}{720}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

§ 6.3 Linear multistep method

The fourth order Adams explicit formula with k = 3 is commonly used

$$y_{i+1} = y_i + \frac{h}{24}(55 f_i - 59 f_{i-1} + 37 f_{i-2} - 9 f_{i-3})$$

§ 6.3 Linear multistep method

- /* Adams implicit formulae */

Use the integrand function f_{i+1}, f_i, ..., f_{i-k+1} on k+1 nodes to construct k-th order Newton forward interpolation polynomial. Similarly, a series implicit formula can be obtained, and $R_i = \tilde{B}_k h^{k+2} y^{(k+2)}(\eta_i)$

where the coefficients of $\tilde{B}_k$ and f_{i+1}, f_i, …, f_{i-k+1} can be found in table

§ 6.3 Linear multistep method

k	f_{i+1}	f_i	f_{i-1}	f_{i-2}	$\cdots$	$\tilde{B}_k$
0	1					$-\frac{1}{2}$
1	$\frac{1}{2}$	$\frac{1}{2}$				$-\frac{1}{12}$
2	$\frac{5}{12}$	$\frac{8}{12}$	$-\frac{1}{12}$			$-\frac{1}{24}$
3	$\frac{9}{24}$	$\frac{19}{24}$	$-\frac{5}{24}$	$\frac{1}{24}$		$-\frac{19}{720}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Less than B_k

The fourth order Adams implicit formula with k = 3 is commonly used

$$y_{i+1} = y_i + \frac{h}{24}(9 f_{i+1} + 19 f_i - 5 f_{i-1} + f_{i-2})$$

more stable

- **/* Adams predictor-corrector system */**

***Step 1*: use *Runge-Kutta* method to calculate the previous *k* initial values**

***Step 2*: use *Adams* explicit formula to calculate the value of predictors ;**

***Step 3*: use the same order *Adams* implicit formula to calculate the value of corrector.**

noted： The accuracy of the formulas used in the three steps must be the same. The classical Runge Kutta method is usually used in combine with the 4th order Adams formula.

■ /* Adams predictor-corrector system */

truncation error of 4th order
***Adams* explicit formula**

$$y(x_{i+1}) - y_{i+1} \equiv \frac{251}{720} h^5 y^{(5)}(\xi_i)$$

truncation error of 4th order
***Adams* implicit formula**

$$y(x_{i+1}) - y_{i+1} = -\frac{19}{720} h^5 y^{(5)}(\eta_i)$$

when *h* is small enough,
approximately have $\xi_i \approx \eta_i$, thus:

$$\frac{y(x_{i+1}) - \bar{y}_{i+1}}{y(x_{i+1}) - y_{i+1}} \approx -\frac{251}{19}$$

$$\Rightarrow \left.\begin{aligned} y(x_{i+1}) &\approx \bar{y}_{i+1} + \frac{251}{270}(y_{i+1} - \bar{y}_{i+1}) \\ y(x_{i+1}) &\approx y_{i+1} - \frac{19}{270}(y_{i+1} - \bar{y}_{i+1}) \end{aligned}\right\}$$

/* extrapolation */