Chapter 5 Solutions of Ordinary Differential Equations

Baodong LIU
baodong@sdu.edu.cn

本章主要讨论微分方程问题的数值计算方法:

- 常微分方程初值问题
- 常微分方程组的初值问题
- 常微分方程边值问题

Initial-Value Problem of Ordinary Differential Equation

- **Initial-value problem of Ordinary Differential Equation**

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y), \quad a < t \leq b. \tag{1}$$

- Subject to an **initial condition**:

$$y(a) = \alpha. \tag{2}$$

Initial-Value Problem of Ordinary Differential Equation

- Initial-value problem of Ordinary Differential Equation

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y), \;\; a < t \leq b. \tag{1}$$

- Subject to an **initial condition**:

$$y(a) = \alpha. \tag{2}$$

- **System of first order differential equations:**

$$\begin{cases} \frac{\mathrm{d}y_1}{\mathrm{d}t} = f_1(t, y_1, y_2, \cdots, y_n), \\ \frac{\mathrm{d}y_2}{\mathrm{d}t} = f_2(t, y_1, y_2, \cdots, y_n), \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ \frac{\mathrm{d}y_n}{\mathrm{d}t} = f_n(t, y_1, y_2, \cdots, y_n), \end{cases}$$

  for $a < t \leq b$

- Subject to the initial conditions

$$y_1(a) = \alpha_1, y_2(a) = \alpha_2, \cdots, y_n(a) = \alpha_n.$$

System of first order differential equations:

- System of first order differential equations:

$$\begin{cases} \frac{\mathrm{d}y_1}{\mathrm{d}t} = f_1(t, y_1, y_2, \cdots, y_n), \\ \frac{\mathrm{d}y_2}{\mathrm{d}t} = f_2(t, y_1, y_2, \cdots, y_n), \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ \frac{\mathrm{d}y_n}{\mathrm{d}t} = f_n(t, y_1, y_2, \cdots, y_n), \end{cases}$$

for $a < t \leq b$

- Subject to the initial conditions

$$y_1(a) = \alpha_1, y_2(a) = \alpha_2, \cdots, y_n(a) = \alpha_n.$$

Example:

种群动力学中非线性系统（Lotka-Volterra）：

$$\begin{cases} x' = f(t,x,y) = x - xy - \frac{1}{10}x^2, & 0 < t \le 30; \\ y' = g(t,x,y) = xy - y - \frac{1}{20}y^2, & 0 < t \le 30. \end{cases}$$

满足初始条件：

$$x(0) = 2, y(0) = 1$$

高阶常微分方程初值问题

- **The nth order initial value problem of ordinary differential equation:**

$$y^{(n)} = f(t, y', y'', \cdots, y^{(n-1)}),$$

for $a < t \le b$

- Subject to the initial conditions

$$y(a) = \alpha_1, y'(a) = \alpha_2, \cdots, y^{(n-1)}(a) = \alpha_n.$$

- **The nth order initial value problem of ordinary differential equation:**

$$y^{(n)} = f(t, y', y'', \cdots, y^{(n-1)}),$$

for $a < t \leq b$

- Subject to the initial conditions

$$y(a) = \alpha_1, y'(a) = \alpha_2, \cdots, y^{(n-1)}(a) = \alpha_n.$$

DEFINITION 5.1

- A function $f(t, y)$ is said to satisfy a **Lipschitz Condition** in the variable y on a set $D \subset \Re^2$, if a constant $L > 0$ exists with the property that

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

whenever $(t, y_1), (t, y_2) \in D$.
- The constant L is called a **Lipschitz Constant** for f.

DEFINITION 5.2 (Convex Set)

A set $D \subset \Re^2$ is said to be convex if whenever

$$(t_1, y_1), (t_2, y_2) \in D,$$

the point

$$((1-\lambda)t_1 + \lambda t_2, (1-\lambda)y_1 + \lambda y_2)$$

also belongs to D for each λ in $[0, 1]$.

THEOREM 5.3

- Suppose $f(t, y)$ is defined on a convex set $D \subset \Re^2$.
- If a constant L exists with

$$\left|\frac{\partial f}{\partial y}(t, y)\right| \leq L, \quad \text{for all} \qquad (t, y) \in D.$$

- Then f satisfies a Lipschitz condition on D in the variable y with Lipschitz constant L.

THEOREM 5.4

- Suppose that

$$D = \{(t, y) | a \leq t \leq b, -\infty < y < \infty\}$$

  and that $f(t, y)$ is continuous on D.
- If f satisfies a Lipschitz condition on D in the variable y, then the initial-value problem

$$\begin{cases} y'(t) = f(t, y), & a < t \leq b \\ y(a) = \alpha. \end{cases}$$

  has a unique solution $y(t)$ for $a \leq t \leq b$.

DEFINITION 5.5

The initial problem

$$\begin{cases} \frac{dy}{dt} = f(t, y), a < t \leq b \\ y(a) = \alpha \end{cases} \tag{3}$$

is said to be **well-posed problem(适定或良态问题)** if

1. A unique solution $y(t)$, to the problem exists;
2. For any $\epsilon > 0$, there exists a positive constant $k(\epsilon)$ with the property that, whenever $|\epsilon_0| < \epsilon$ and $\delta(t)$ is continuous with $|\delta(t)| < \epsilon$ on $[a, b]$, a unique solution, $z(t)$ to the problem

$$\begin{cases} \frac{\mathrm{d}z}{\mathrm{d}t} = f(t, z) + \delta(t), \quad a < t \leq b, \\ z(a) = \alpha + \epsilon_0, \end{cases} \tag{4}$$

exists with $|z(t) - y(t)| < k(\epsilon)\epsilon, \ a \leq t \leq b$.

- **NOTE:** Problem specified by Eq. (4) is called a **perturbed problem** associated with the **original problem** Eq.(3).

THEOREM 9.6

Suppose that

$$D = \{(t, y) | a \leq t \leq b, -\infty < y < \infty\}.$$

If $f(t, y)$ is **continuous** and satisfies a **Lipschitz condition** on D in the variable y on the set D, then the initial-value problem

$$\begin{cases} \frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y), a < t \leq b \\ y(a) = \alpha \end{cases}$$

is well-posed.

- To solve a well-posed initial-value problem

$$\begin{cases} y'(t) = f(t, y), & a < t \leq b, \\ y(a) = \alpha. \end{cases}$$

数值计算格式设计步骤

- **Step 1**: 求解区间网格化(离散化):
  Partition $[a, b]$ into N subintervals with $N+1$ mesh points
  $$t_0 = a < t_1 < \cdots < t_N = b$$
  with equal **step size**:
  $$h = \frac{(b-a)}{N}.$$
  where $t_j = a + jh, \; j = 0, 1, 2, \cdots, N.$

- **Step 2: 连续问题离散化** (即把求连续解$y(t)$ 的问题$\rightarrow$ 求$y(t_j), j = 0, 1, 2, \cdots, N$):
  - If $y(t_j)$ is known, by **Taylor** formula, we have:

$$y(t_{j+1}) = y(t_j) + (t_{j+1} - t_j)y'(t_j) + \frac{(t_{j+1} - t_j)^2}{2} y''(\xi_j)$$

    where ξ_j lies in $[t_j, t_{j+1}]$.
  - Since $y'(t) = f(t, y)$, and $h = t_{j+1} - t_j$, implies that

$$y(t_{j+1}) = y(t_j) + hf(t_j, y(t_j)) + \frac{h^2}{2} y''(\xi_j)$$

- **Step 3: 数值计算格式:**
  - Let $y(t_j)$ be the exact solution,
  - $y_j \approx y(t_j)$ be the value of numerical approximation of $y(t_j)$ at mesh point t_j for each $j = 0, 1, 2, \cdots, N$.
  - Deleting the remainder term(余项—截断误差项) $\frac{h^2}{2} y''(\xi_j)$, we have **Euler's Method**:

$$\begin{cases} y_0 = \alpha, \\ y_{j+1} = y_j + hf(t_j, y_j), \quad j = 0, 1, \cdots, N-1. \end{cases} \tag{5}$$

- **Step 4:** 编程计算

Notes:

- Equation (5) is called the **Forward Difference Equation** associated with **Euler's method**.
- 利用Taylor 展开式设计微分方程的数值计算格式是一种常见的方法，其特点是可以在设计算法格式的同时，可以同步得到其截断误差估计（局部误差）.
- **差商近似方法也是理解Euler 格式的好的方法，即:**

$$\frac{y(t_{j+1})-y(t_j)}{t_{j+1}-t_j}=\frac{y(t_j+h)-y(t_j)}{h}\approx y'(t_j)=f(t_j,y(t_j))$$

- Euler 格式的几何意义：求过点$(t_j,y(t_j))$ 以$ff(t_j,y(t_j))$为斜率的切线，求切线在$t=t_{j+1}$ 点的值作为$y(t_{j+1})$的近似解.

ALGORITHM 9.1 Euler's Method

INPUT: endpoints a, b; integer N, and initial condition α.

OUTPUT: approximation w to y at $N+1$ points of t.

STEP 1 set $h = (b-a)/N; t = a; w = \alpha;$

- OUTPUT t, w.

STEP 2 for $i = 1, 2, \cdots, N$, do STEP 3,4.

STEP 3 set $w = w + h * f(t, w)$ (compute w_i)

- $t = a + ih$ (compute t_i)

STEP 4 OUTPUT (t, w).

STEP 5 STOP.

欧拉格式的误差估计理论

设$y = y(t)$ 是微分方程初值问题的解析解，$\{y_j\}_{j=0}^{N}$ 是Euler 格式得到的数值解

- 全局误差

$$e_j = y(t_j) - y_j, \quad j = 0, 1, 2, \cdots, N$$

- 局部截断误差的定义，文献中的叙述有差异，一类是给出数值解的逼近局部误差，一类定义为微分方程的逼近误差，本文采用后者，即

$$y'(t_j) \approx \frac{y(t_{j+1}) - y(t_j)}{h} = f(t_j, y(t_j)) + \frac{h}{2} y''(\xi_j)$$

由此，定义局部截断误差为(假定$y_j = y(t_j)$):

$$\tau_{j+1}(h) \quad = \quad \frac{y_{j+1} - y_j}{h} - f(t_j, y_j), \quad j = 0, 1, 2, \cdots, N-1$$

欧拉格式的误差估计理论

LEMMA 9.7

For all $x \geq -1$ and any positive m, we have

$$0 \leq (1+x)^m \leq e^{mx}.$$

- **Proof:** Using the Taylor'S formula for e^x on $x_0 = 0$, and $n = 1$, gives

$$e^x = 1 + x + \frac{1}{2}x^2 e^{\xi}.$$

where ξ is between x and zero.
Since $x \geq -1$, thus,

$$0 \leq 1 + x \leq 1 + x + \frac{1}{2}x^2 e^{\xi} = e^x$$

so

$$0 \leq (1+x)^m \leq e^{mx}. \blacksquare\blacksquare\blacksquare$$

LEMMA 9.8

If s and t are positive real number, $\{a_j\}_{j=0}^{k}$ is a sequence satisfying $a_0 \geq -t/s$,and

$$a_{j+1} \leq (1+s)a_j + t, \quad j = 0, 1, \cdots, k,$$

then

$$a_{j+1} \leq e^{(j+1)s}\left(a_0 + \frac{t}{s}\right) - \frac{t}{s}.$$

Proof: From known condition

$$a_{j+1} \leq (1+s)a_j + t, \quad j = 0, 1, \cdots, k,$$

we can drive that

$$\begin{aligned} a_{j+1} &\leq (1+s)a_j + t \\ &\leq (1+s)[(1+s)a_{j-1} + t] + t \\ &\leq (1+s)\{(1+s)[(1+s)a_{j-2} + t] + t\} + t \\ &\vdots \\ &\leq (1+s)^{j+1}a_0 \\ &\quad +[1 + (1+s) + (1+s)^2 + \cdots + (1+s)^j]t. \end{aligned}$$

Since

$$1+(1+s)+(1+s)^2+\cdots+(1+s)^j=\frac{1-(1+s)^{j+1}}{1-(1+s)}$$

$$a_{j+1} \quad \leq \quad (1+s)^{j+1}\left(a_0+\frac{t}{s}\right)-\frac{t}{s}.$$

and by the LEMMA 9.7, we have

$$a_{j+1}\leq e^{(j+1)s}\left(a_0+\frac{t}{s}\right)-\frac{t}{s}.\blacksquare\blacksquare\blacksquare.$$

THEOREM 9.9

- Suppose f is continuous and satisfies a Lipschitz Condition with Constant L on

$$D = \{(t, y) | a \leq t \leq b, -\infty < y < \infty\},$$

and that a constant M exists with the property that

$$|y''(t)| \leq M, \quad \text{for all} \quad t \in [a, b].$$

- Let $y(t)$ be the unique solution to the initial problem (1), and $y_j, j = 0, 1, \cdots, N$ be the approximations generated by Euler's method for some positive number N.
- Then

$$|y(t_j) - y_j| \leq \frac{hM}{2L}[e^{L(t_j - a)} - 1], \; j = 0, 1, 2, \cdots, N. \quad (6)$$

Proof.

- When $j = 0$, since $y_0 = y(t_0) = \alpha.$, the result is true.
- When $j = 1, 2, \cdots, N$, by the Taylor's theorem, gives

$$y(t_{j+1}) = y(t_j) + hf(t_j, y(t_j)) + \frac{h^2}{2} y''(\xi_j)$$

  and from the Euler difference equation,

$$y_{j+1} = y_j + hf(t_j, y_j)$$

- Thus

$$\begin{aligned} & y(t_{j+1}) - y_{j+1} \\ = \; & y(t_j) - y_j + h[f(t_j, y(t_j)) - f(t_j, y_j)] + \frac{h^2}{2} y''(\xi_j) \end{aligned}$$

- $$\begin{aligned} & |y(t_{j+1}) - y_{j+1}| \\ \leq\ & |y(t_j) - y_j| + h|f(t_j, y(t_j)) - f(t_j, y_j)| + \frac{h^2}{2}|y''(\xi_j)| \end{aligned}$$
- Since f is continuous and satisfies a Lipchitz in y with constant L, and $|y''(t)| \leq M$, then we have
$$|y(t_{j+1}) - y_{j+1}| \leq (1 + hL)|y(t_j) - y_j| + \frac{Mh^2}{2}$$
- Let
$$a_j = |y(t_j) - y_j|,\ j = 0, 1, 2, \cdots, N$$
and $s = hL$ and $t = Mh^2/2$

- By the Lemma 9.8, we can see that

$$
\begin{aligned}
|y(t_{j+1}) - y_{j+1}| \quad \leq \quad & e^{(j+1)hL}\left(|y(t_0) - y_0| + \frac{Mh^2}{2hL}\right) \\
& -\frac{Mh^2}{2hL}
\end{aligned}
$$

- Since $|y(t_0) - y_0| = 0$ and

$$(j+1)h = t_{j+1} - t_0 = t_{j+1} - a,$$

we have

$$|y(t_{j+1}) - y_{j+1}| \leq \frac{Mh}{2L}[e^{(t_{j+1}-a)L} - 1],$$

for each $j = 0, 1, \cdots, N$.$\blacksquare\blacksquare\blacksquare$

Analysis of Roundoff Error

To consider the roundoff error(舍入误差) for each approximation, we use instead an equation of the form

$$\begin{aligned} u_0 &= a + \delta_0, \\ u_{j+1} &= u_j + hf(t_j, u_j) + \delta_{j+1} \end{aligned}$$

for each $j = 0, 1, \cdots, N-1$, where δ_j denotes the roundoff error associated with u_j.

THEOREM 9.10

Let $y(t)$ be the solution of initial-value problem

$$\begin{cases} y'(t) = f(t, y), \quad a \le t \le b, \\ y(a) = \alpha. \end{cases}$$

and $u_0, u_1, \cdots, u_N$ be the approximations obtained by using

$$\begin{cases} u_0 = a + \delta_0, \\ u_{j+1} = u_j + hf(t_j, u_j) + \delta_{j+1}. \end{cases}$$

If $|\delta_j| < \delta$ for each $j = 0, 1, \cdots, N$ and the hypotheses of Theorem 9.9 holds for original problem, then

$$|y(t_j) - u_j| \le \frac{1}{L}\left(\frac{hM}{2} + \frac{\delta}{h}\right)[e^{L(t_j - a)} - 1] + |\delta_0| e^{L(t_j - a)},$$

for each $j = 0, 1, \cdots, N$.

Remarks:

- To compare the error form of Theorem 9.10 with Theorem 9.9, we can find that the **error bound** of Theorem 9.10 is no longer linear in h.
- In fact, since
$$\lim_{h\to 0}\left(\frac{hM}{2}+\frac{\delta}{h}\right)=\infty,$$
thus the error will be expected to become large for sufficient small value of h.

Remarks:

- Let
$$E(h) = \left(\frac{hM}{2} + \frac{\delta}{h}\right)$$
then
$$E'(h) = \left(\frac{M}{2} - \frac{\delta}{h^2}\right).$$
- If $h < \sqrt{2\delta/M}$ then $E'(h) < 0$, and $E(h)$is decreasing.
- If $h > \sqrt{2\delta/M}$ then $E'(h) > 0$, and $E(h)$is increasing.
- So the minimal value of $E(h)$ occurs when
$$h = \sqrt{\frac{2\delta}{M}}.$$

Local Truncation Error

DEFINITION 9.11 (Local Truncation Error)

The difference method

$$\begin{aligned} y_0 &= \alpha \\ y_{i+1} &= y_i + h\phi(t_i, y_i), i = 0, 1, 2, \cdots, N-1, \end{aligned}$$

has **local truncation error** given by

$$\tau_{i+1}(h) = \frac{y_{i+1} - y_i}{h} - \phi(t_i, y_i),$$

for each $i = 0, 1, 2, \cdots, N-1$.

The local truncation error analysis for Euler's Method

- **To Consider Initial Value Problem of Ordinary Differential Equation**

$$\begin{cases} y'(t) = f(t, y), \quad a \leq t \leq b \\ y(a) = \alpha. \end{cases}$$

- By the Euler's method, we know

$$\begin{aligned} y_0 &= \alpha \\ y_{i+1} &= y_i + hf(t_i, y_i), i = 0, 1, 2, \cdots, N-1, \end{aligned}$$

- By the Taylor's theorem, gives

$$y(t_{i+1}) = y(t_i) + hf(t_i, y(t_i)) + \frac{h^2}{2} y''(\xi_i)$$

- If $y_i = y(t_i)$ denotes the exact value of the solution at t_i, $i = 1, 2, \cdots, N$.
- So the local truncation error for Euler's Method is

$$\tau_{i+1}(h) = \frac{y_{i+1} - y_i}{h} - f(t_i, y_i) = \frac{h}{2} y''(\xi_j),$$

- Further, if $|y''(\xi_i)| \leq M$ where $M > 0$ is a positive constant, thus, this implies

$$|\tau_{i+1}(h)| \leq \frac{h}{2}M.$$

- That is the **local truncation error for Euler's method** is $O(h)$.
- To improve the order of local truncation error, to get higher-order error, we construct Taylor Method of Order n

9.3 Higher-Order Taylor Methods — 高阶泰勒级数法

- Suppose that the solution to the initial value problem of ordinary differential equation

$$\begin{cases} y'(t) = f(t, y), & a \le t \le b \\ y(a) = \alpha. \end{cases}$$

has $(n+1)$ continuous derivatives.

- By the nth Taylor Polynomial about t_i, we obtain

$$\begin{aligned} y(t_{i+1}) &= y(t_i) + hy'(t_i) + \frac{h^2}{2}y''(t_i) + \cdots \\ &\quad + \frac{h^n}{n!}y^{(n)}(t_i) + \frac{h^{n+1}}{(n+1)!}y^{(n+1)}(\xi_i) \qquad (7) \end{aligned}$$

for ξ_i in (t_i, t_{i+1})).

- Since for the solution $y(t)$ satisfies

$$y'(t) = f(t, y)$$

thus we can get

$$\begin{aligned} y''(t) &= f'(t, y(t)), \\ y^{(3)}(t) &= f''(t, y(t)), \\ &\vdots \\ y^{(k)}(t) &= f^{(k-1)}(t, y(t)). \end{aligned}$$

- Substituting these results into Eq. (7), gives

$$\begin{aligned} y(t_{i+1}) &= y(t_i) + hf(t_i, y(t_i)) + \frac{h^2}{2} f'(t_i, y(t_i)) + \cdots \\ &+ \frac{h^n}{n!} f^{(n-1)}(t_i, y(t_i)) \\ &+ \frac{h^{n+1}}{(n+1)!} f^{(n)}(\xi_i, y(\xi_i)) \qquad (8) \end{aligned}$$

- Based on the Eq. (8), we can construct the **Taylor Method of Order n:**

$$\begin{cases} y_0 = \alpha \\ y_{i+1} = y_i + hT^{(n)}(t_i, y_i), \quad i = 0, 1, 2, \cdots, N-1, \end{cases}$$

where

$$\begin{aligned} T^{(n)}(t_i, y_i) &= f(t_i, y_i) + \frac{h}{2} f'(t_i, y_i) + \cdots + \\ &\quad + \frac{h^{(n-1)}}{n!} f^{(n-1)}(t_i, y_i). \end{aligned}$$

- **Note that** the Euler's method is Taylor's method of order one.

- If $y_i = y(t_i)$ is the exact value of the solution at $t_i, i = 0, 1, 2, \cdots, n$
- Thus the local truncation error of the Taylor Method or Order n is

$$\begin{aligned} \tau_{i+1}(h) &= \frac{y_{i+1} - y_i}{h} - T^{(n)}(t_i, y_i) \\ &= \frac{h^n}{(n+1)!} f^{(n)}(\xi_i, y(\xi_i)), \;\; i = 0, 1, , 2, \cdots, N-1 \end{aligned}$$

- Further if $y \in C^{n+1}[a, b]$, this implies that

$$y^{(n+1)}(t) = f^{(n)}(t, y(t))$$

is bounded on $[a, b]$ and that

$$\tau_i = O(h^n), \;\; i = 1, 2, \cdots, N.$$

THEOREM 9.12(Taylor Theorem—多元泰勒展开式)

- Suppose that $f(t, y)$ and its partial derivatives of order less than or equal to $n+1$ are continuous on

$$D = \{(t, y) | a \leq t \leq b, c \leq y \leq d\},$$

  and let $(t_0, y_0) \in D$.

- For every $(t, y) \in D$, there exists ξ between t and t_0 and μ between y and y_0 with

$$f(t, y) = P_n(t, y) + R_n(t, y),$$

where

$$
\begin{aligned}
& P_n(t,y) \\
= \quad & f(t_0,y_0) + \left[(t-t_0)\frac{\partial f}{\partial t}(t_0,y_0) + (y-y_0)\frac{\partial f}{\partial y}(t_0,y_0)\right] \\
& + \left[\frac{(t-t_0)^2}{2}\frac{\partial^2 f}{\partial t^2}(t_0,y_0) + (t-t_0)(y-y_0)\frac{\partial^2 f}{\partial t \partial y}(t_0,y_0) \right. \\
& \left. +(y-y_0)^2\frac{\partial^2 f}{\partial y^2}(t_0,y_0)\right] + \cdots + \\
& + \left[\frac{1}{n!}\sum_{j=0}^{n}\left(\begin{array}{c} n \\ j \end{array}\right)(t-t_0)^{(n-j)}(y-y_0)^j\frac{\partial^n f}{\partial t^{n-j}\partial y^j}(t_0,y_0)\right].
\end{aligned}
$$

and

$$
\begin{aligned}
& R_n(t,y) \\
= \quad & \frac{1}{(n+1)!}\sum_{j=0}^{n+1}\left(\begin{array}{c} n+1 \\ j \end{array}\right)(t-t_0)^{(n+1-j)}(y-y_0)^j\frac{\partial^{n+1} f}{\partial t^{n+1-j}\partial y^j}(\xi,\mu)
\end{aligned}
$$

Note that

- $P_n(t, y)$ is called the n**th Taylor polynomial in two variables** for the function f about (t_0, y_0)
- $R_n(t, y)$ is the **remainder term** associated with $P_n(t, y)$.

Runge-Kutta Method

- The Runge-Kutta Method is new method with higher-order error, without repeated computing the higher-order derivatives of function $f(t, y)$, its general form is

$$y_{n+1} = y_n + \sum_{i=1}^{N} c_i K_i$$

where

$$\begin{aligned} K_1 &= hf(t_n, y_n) \\ K_i &= hf(t_n + \alpha_i h, y_n + \sum_{j=1}^{i-1} b_{ij} K_j), i = 2, 3, \cdots, N \end{aligned}$$

Case $N = 2$

- For the case of $N = 2$, the Runge-Kutta method is

$$y_{n+1} = y_n + c_1 K_1 + c_2 K_2$$

where

$$\begin{aligned} K_1 &= hf(t_n, y_n) \\ K_2 &= hf(t_n + \alpha_2 h, y_n + b_{21} K_1), \end{aligned}$$

- **Question?** How to evaluate the coefficients $c_1, c_2, \alpha_2, b_{21}$?

- Using the Taylor's Theorem in two variables t and y, gives

$$\begin{aligned}K_2 &= h\left[f(t_n, y_n) + (\alpha_2 h \frac{\partial f}{\partial t}(t_n, y_n) + b_{21} h f(t_n, y_n)\frac{\partial f}{\partial y}(t_n, y_n))\right] \\ &+ hR_1(t_n, y_n).\end{aligned}$$

where

$$R_1(t_n, y_n) = \frac{\alpha_2^2 h^2}{2}\frac{\partial^2 f}{\partial t^2} + \alpha_2 b_{21} h^2 f \cdot \frac{\partial^2 f}{\partial t \partial y} + b_{21}^2 h^2 f^2 \cdot \frac{\partial^2 f}{\partial y^2}|_{(\xi,\mu)}$$

- Substituting K_1, K_2 into $y_{n+1} = y_n + c_1 K_1 + c_2 K_2$, gives

$$\begin{aligned}y_{n+1} &= y_n + h(c_1 + c_2) f(t_n, y_n) + c_2 \alpha_2 h^2 \frac{\partial f}{\partial t}(t_n, y_n) \\ &+ c_2 b_{21} h^2 f(t_n, y_n)\frac{\partial f}{\partial y}(t_n, y_n)) \\ &+ c_2 h^3 \left[\frac{\alpha_2^2}{2}\frac{\partial^2 f}{\partial t^2} + \alpha_2 b_{21} f \cdot \frac{\partial^2 f}{\partial t \partial y} + b_{21}^2 f^2 \cdot \frac{\partial^2 f}{\partial y^2}\right]\Bigg|_{(\xi,\mu)}\end{aligned}$$

- Reconsidering the Taylor method of Order 2

$$y_{n+1} = y_n + hT^{(2)}(t_n, y_n),$$

where

$$\begin{aligned} T^{(2)}(t_n, y_n) &= f(t_n, y_n) + \frac{h}{2} f'(t_n, y_n) \\ &= f(t_n, y_n) + \frac{h}{2}\frac{\partial f}{\partial t}(t_n, y_n) \\ &\quad + \frac{h}{2}\frac{\partial f}{\partial y}(t_n, y_n) \cdot f(t_n, y_n) \end{aligned}$$

- **Note:**

$$f'(t,y) = \frac{\mathrm{d}f}{\mathrm{d}t}(t,y) = \frac{\partial f}{\partial t}(t,y) + \frac{\partial f}{\partial y}(t,y) \cdot y'(t) = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial y} \cdot f.$$

- Thus

$$
\begin{aligned}
y_{n+1} &= y_n + hT^{(2)}(t_n, y_n) \\
&= y_n + hf(t_n, y_n) + \frac{h^2}{2}\frac{\partial f}{\partial t}(t_n, y_n) \\
&\quad + \frac{h^2}{2}\frac{\partial f}{\partial y}(t_n, y_n) \cdot f(t_n, y_n)
\end{aligned}
$$

- Comparing this equation with previous Runge-kutta method of the case $N = 2$

$$
\begin{aligned}
y_{n+1} &= y_n + h(c_1 + c_2)f(t_n, y_n) \\
&\quad + c_2\alpha_2 h^2 \frac{\partial f}{\partial t}(t_n, y_n) + c_2 b_{21} h^2 f(t_n, y_n)\frac{\partial f}{\partial y}(t_n, y_n)) \\
&\quad + c_2 h^3 \left[\frac{\alpha_2^2}{2}\frac{\partial^2 f}{\partial t^2} + \alpha_2 b_{21} f \cdot \frac{\partial^2 f}{\partial t \partial y} + b_{21}^2 f^2 \cdot \frac{\partial^2 f}{\partial y^2}\right]\Bigg|_{(\xi,\mu)}
\end{aligned}
$$

- Let

$$\begin{aligned} c_1 + c_2 &= 1, \\ c_2\alpha_2 &= 1/2, \\ c_2 b_{21} &= 1/2 \end{aligned}$$

- Then the two equations have the same local truncation error of order $O(h^2)$, if we assume that all the second-order derivatives of f are bounded.
- Since the parameters $c_1, c_2, \alpha_2, b_{21}$ are determined not uniquely, then we can some specific Runge-Kutta Methods of Order 2.

- **Midpoint Method(中点格式):**
—-$c_1 = 0, c_2 = 1, \alpha_2 = 1/2, b_{21} = 1/2$

$$\begin{aligned} y_0 &= \alpha \\ y_{i+1} &= y_i + hf(t_i + \frac{h}{2}, y_i + \frac{h}{2} f(t_i, y_i)), \end{aligned}$$

for each $i = 1, 2, \cdots, N-1$.

- **Modified Euler Method(修正的欧拉格式，即预估校正格式):** —$c_1 = 1/2, c_2 = 1/2, \alpha_2 = 1, b_{21} = 1$

$$\begin{aligned} y_0 &= \alpha \\ y_{i+1} &= y_i + \frac{h}{2}[f(t_i, y_i) + f(t_{i+1}, y_i + hf(t_i, y_i))], \end{aligned}$$

for each $i = 1, 2, \cdots, N-1$.

- **Heun's Method:**—$c_1 = \frac{1}{4}, c_2 = \frac{3}{4}, \alpha_2 = \frac{2}{3}, b_{21} = \frac{2}{3}$

$$\begin{aligned} y_0 &= \alpha \\ y_{i+1} &= y_i + \frac{h}{4}\left[f(t_i, y_i) + 3f\left(t_i + \frac{2}{3}h, y_i + \frac{2}{3}hf(t_i, y_i)\right)\right], \end{aligned}$$

for each $i = 1, 2, \cdots, N-1$.

Runge-Kutta method of order four

Using same idea, we can derive the **Runge-Kutta method of Order Four**

$$\begin{cases} y_0 = \alpha, \\ y_{i+1} = y_i + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4) \end{cases}$$

where

$$\begin{aligned} K_1 &= hf(t_i, y_i), \\ K_2 &= hf(t_i + \frac{h}{2}, y_i + \frac{1}{2}K_1), \\ K_3 &= hf(t_i + \frac{h}{2}, y_i + \frac{1}{2}K_2), \\ K_4 &= hf(t_{i+1}, y_i + K_3), \end{aligned}$$

for each $i = 1, 2, \cdots, N-1$.

ALGORITHM Runge-Kutta (Order Four)

To approximate the solution of the initial-value problem

$$y' = f(t, y), a \leq t \leq b, y(a) = \alpha,$$

at

$$t_0 = a < t_1, < t_2 < \cdots < t_N = b$$

totally $(N+1)$ equally spaced numbers in the interval $[a, b]$:

INPUT endpoints a, b; integer N; initial condition α.

OUTPUT approximation w to y at the $(N+1)$ values of t.

Step 1 Set

$$h = (b-a)/N; t = a; y = \alpha;$$

OUTPUT (t, y)(output initial values at $t_0 = a$).

Step 2 For $i = 1, 2, \cdots, N$,do Steps 3-5.

Step 3 Set

$$
\begin{aligned}
K_1 &= hf(t, y), \\
K_2 &= hf(t + \frac{h}{2}, y + \frac{1}{2}K_1), \\
K_3 &= hf(t + \frac{h}{2}, y + \frac{1}{2}K_2), \\
K_4 &= hf(t + h, y + K_3),
\end{aligned}
$$

Step 4 Set

$$
\begin{aligned}
y &= y + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4), \\
&\quad (\text{Compute} y_i) \\
t &= a + ih. (\text{Compute} t_i)
\end{aligned}
$$

Step 5 OUTPUT (t, y).

Step 6 STOP.

9.5 Error Control and the Runge-Kutta-Fehlberg Method

- An ideal difference-equation method

$$y_{i+1} = y_i + h_i \phi(t_i, y_i, h_i), i = 0, 1, \cdots, N-1,$$

for approximating the solution, $y(t)$, to the initial-value problem

$$y'(t) = f(t, y), a \le t \le b, y(a) = \alpha$$

would have the property that: Given a tolerance $\varepsilon > 0$, the **minimal number of mesh points** would be used to ensure that the **global error**, $|y(t_i) - y_i|$, would not exceed ε for any $i = 0, 1, \cdots, N$.

Runge-Kutta-Fehlberg Method

- To illustrate the technique, **suppose that we have two approximation techniques.**
- The first is an nth-order method obtained from an nth-order Taylor method of the form

$$y(t_{i+1}) = y(t_i) + h\phi(t_i, y(t_i), h) + O(h^{n+1}),$$

producing approximations

$$\begin{aligned} y_0 &= \alpha, \\ y_{i+1} &= y_i + h\phi(t_i, y_i, h), \text{for} \quad i > 0 \end{aligned}$$

with local truncation error $\tau_{i+1}(h) = O(h^n)$.

- The second method is similar but of higher order.
- For example, let us suppose it comes from an $(n+1)$st-order Taylor method of the form

$$y(t_{i+1}) = y(t_i) + h\tilde{\phi}(t_i, y(t_i), h) + O(h^{n+2}),$$

producing approximations

$$\begin{aligned} \tilde{y}_0 &= \alpha, \\ \tilde{y}_{i+1} &= \tilde{y}_i + h\tilde{\phi}(t_i, \tilde{y}_i, h), \text{for} \quad i > 0 \end{aligned}$$

with local truncation error $\tilde{\tau}_{i+1}(h) = O(h^{n+1})$.
- We first make the assumption that

$$y_i \approx y(t_i) \approx \tilde{y}_i$$

and choose a fixed step size h to generate the approximations y_{i+1} and $\tilde{y}_{i+1}$ to $y(t_{i+1})$.

- Then

$$\begin{aligned}\tau_{i+1}(h) &= \frac{y(t_{i+1}) - y(t_i)}{h} - \phi(t_i, y(t_i), h)\\ &= \frac{y(t_{i+1}) - y_i}{h} - \phi(t_i, y_i, h)\\ &= \frac{y(t_{i+1}) - [y_i + h\phi(t_i, y_i, h)]}{h}\\ &= \frac{y(t_{i+1}) - y_{i+1}}{h}\end{aligned}$$

- In a similar manner

$$\tilde{\tau}_{i+1}(h) = \frac{y(t_{i+1}) - \tilde{y}_{i+1}}{h}.$$

- As a consequence,

$$\begin{aligned}\tau_{i+1}(h) &= \frac{y(t_{i+1}) - y_{i+1}}{h} = \frac{(y(t_{i+1}) - \tilde{y}_{i+1}) + (\tilde{y}_{i+1} - y_{i+1})}{h}\\ &= \tilde{\tau}_{i+1}(h) + \frac{\tilde{y}_{i+1} - y_{i+1}}{h}\end{aligned}$$

- Since $\tau_{i+1}(h)$ is $O(h^n)$ and $\tilde{\tau}_{i+1}(h)$ is $O(h^{n+1})$.
- So the significant portion of $\tau_{i+1}(h)$ must come from

$$\frac{\tilde{y}_{i+1} - y_{i+1}}{h}.$$

- This gives us an easily computed approximation for the local truncation error of the $O(h^n)$ method:

$$\tau_{i+1}(h) \approx \frac{\tilde{y}_{i+1} - y_{i+1}}{h}$$

- The object, however, is not simply to estimate the local truncation error but to **adjust the step size** to keep it within a specified bound.
- To do this, we now assume that since $\tau_{i+1}(h)$ is $O(h^n)$, a number K, independent of h, exists with

$$\tau_{i+1}(h) \approx Kh^n.$$

- Then the local truncation error produced by applying the nth-order method with a **new step size** qh can be estimated using the original approximations y_{i+1} and $\tilde{y}_{i+1}$:

- $$\begin{aligned}\tau_{i+1}(qh) &\approx K(qh)^n = q^n(Kh^n) \approx q^n \tau_{i+1}(h)\\ &\approx \frac{q^n}{h}(\tilde{y}_{i+1} - y_{i+1}).\end{aligned}$$
- To bound $\tau_{i+1}(qh)$ by ε, we choose q so that
$$\frac{q^n}{h}|\tilde{y}_{i+1} - y_{i+1}| \approx |\tau_{i+1}(qh)| \leq \varepsilon,$$
- that is, so that
$$q \leq \left(\frac{\varepsilon h}{|\tilde{y}_{i+1} - y_{i+1}|}\right)^{1/n}.$$
- 备注：使用固定均匀网格简单易行，但无法控制计算精度。本节所介绍的方法给出了一种根据计算精度要求自动调整网格步长（在原剖分基础上压缩或放大网格步长）的方法.

Runge- Kutta-Fehlberg method

- 以下方法为Runge- Kutta-Fehlberg 方法：
- Using a Runge-Kutta method with local truncation error of **order five**:

$$\tilde{y}_{i+1} = y_i + \frac{16}{135}K_1 + \frac{6656}{12825}K_3 + \frac{28561}{56430}K_4 - \frac{9}{50}K_5 + \frac{2}{55}K_6,$$

to estimate the local error in a Runge- Kutta method of **order four** given by

$$y_{i+1} = y_i + \frac{25}{216}K_1 + \frac{1408}{2565}K_3 + \frac{2197}{4104}K_4 - \frac{1}{5}K_5,$$

where

$$
\begin{aligned}
K_1 &= hf(t_i, y_i), \\
K_2 &= hf\left(t_i + \frac{h}{4}, y_i + \frac{1}{4}K_1\right), \\
K_3 &= hf\left(t_i + \frac{3h}{8}, y_i + \frac{3}{32}K_1 + \frac{9}{32}K_2\right), \\
K_4 &= hf\left(t_i + \frac{12h}{13}, y_i + \frac{1932}{2197}K_1 - \frac{7200}{2197}K_2 + \frac{7296}{2197}K_3\right), \\
K_5 &= hf\left(t_i + h, y_i + \frac{439}{216}K_1 - 8K_2 + \frac{3680}{513}K_3 - \frac{845}{4104}K_4\right), \\
K_6 &= hf\left(t_i + \frac{h}{2}, y_i - \frac{8}{27}K_1 + 2K_2 - \frac{3544}{2565}K_3 + \frac{1859}{4104}K_4 - \frac{11}{40}K_5\right).
\end{aligned}
$$

Runge-Kutta-Fehlberg ALGORITHM

To approximate the solution of the initial-value problem

$$y' = f(t, y), a \leq t \leq b, y(a) = \alpha,$$

with local truncation error within a given tolerance:

INPUT endpoints a, b; initial condition α; tolerance TOL; maximum step size $hmax$; minimum step size $hmin$.

OUTPUT t, w, h where w approximates $y(t)$ and the step size h was used, or a message that the minimum step size was exceeded.

Step 1 Set $t = a; w = \alpha; h = hmax; FLAG = 1;$
OUTPUT (t, w).

Step 2 While $(FLAG = 1)$ do Steps 3-11.

Step 3 Set

$$
\begin{aligned}
K_1 &= hf(t, w); \\
K_2 &= hf(t + \frac{h}{4}, w + \frac{1}{4}K_1); \\
K_3 &= hf(t + \frac{3h}{8}, w + \frac{3}{32}K_1 + \frac{9}{32}K_2), \\
K_4 &= hf(t + \frac{12h}{13}, w + \frac{1932}{2197}K_1 - \frac{7200}{2197}K_2 + \frac{7296}{2197}K_3), \\
K_5 &= hf(t + h, w + \frac{439}{216}K_1 - 8K_2 + \frac{3680}{513}K_3 - \frac{845}{4104}K_4), \\
K_6 &= hf(t + \frac{h}{2}, w - \frac{8}{27}K_1 + 2K_2 - \frac{3544}{2565}K_3 + \frac{1859}{4104}K_4 - \frac{11}{40}K_5).
\end{aligned}
$$

Step 4 Set

$$R = \frac{1}{h}\left|\frac{1}{360}K_1 - \frac{128}{4275}K_3 - \frac{2197}{75240}K_4 + \frac{1}{50}K_5 + \frac{2}{55}K_6\right|.$$

(Note: $R = \frac{1}{h}|\tilde{y}_{i+1} - y_{i+1}|$.)

Step 5 If $R \leq TOL$ then do Steps 6 and 7.

- **Step 6** Set $t = t + h$; (Approximation accepted.)

$$w = w + \frac{25}{216}K_1 + \frac{1408}{2565}K_3 + \frac{2197}{4104}K_4 - \frac{1}{5}K_5.$$

- **Step 7** OUTPUT (t, w, h).

Step 8 Set $\delta = 0.84(TOL/R)^{1/4}$.

Step 9
- If $\delta \leq 0.1$ then set $h = 0.1h$
- else if $\delta \geq 4$ then set $h = 4h$
- else set $h = \delta h$. (Calculate new h.)

Step 10 If $h > hmax$, then set $h = hmax$.

Step 11
- If $t \geq b$ then set $FLAG = 0$
- else if $t + h > b$ then set $h = b - t$
- else if $h < hmin$ then set $FLAG = 0$;
- OUTPUT ('minimum h exceeded').
  (Procedure completed unsuccessfully.)

Step 12 (The procedure is complete.) STOP.

1. 一阶常微分方程组的数值计算方法

只要把 y 和 f 理解为向量, 则前面所研究的各种算法即可推广应用到一阶常微分方程组的情形. 以如下最简单的常微分方程组为例

$$\begin{cases} y' = f(x, y, z), \\ z' = g(x, y, z), \\ y(x_0) = y_0, z(x_0) = z_0. \end{cases}$$

预估-校正格式

令$x_n = x_0 + nh, n = 1, 2, \cdots$, 以$y_n, z_n$表示节点$x_n$ 上的近似解, 则其基于梯形公式的预估-校正格式具有形式:

$$
\begin{aligned}
\text{预估:}\bar{y}_{n+1} &= y_n + hf(x_n, y_n, z_n)\\
\bar{z}_{n+1} &= z_n + +hg(x_n, y_n, z_n)\\
\text{校正:}\bar{y}_{n+1} &= y_n + \frac{h}{2}[f(x_n, y_n, z_n) + f(x_n, \bar{y}_{n+1}, \bar{z}_{n+1})]\\
\bar{z}_{n+1} &= z_n + \frac{h}{2}[g(x_n, y_n, z_n) + g(x_n, \bar{y}_{n+1}, \bar{z}_{n+1})]
\end{aligned}
$$

相应的四阶龙格一库塔格式为

$$\begin{cases} y_{n+1} = y_n + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ z_{n+1} = z_n + \frac{h}{6}(L_1 + 2L_2 + 2L_3 + L_4) \\ K_1 = f(x_n, y_n, z_n) \\ L_1 = g(x_n, y_n, z_n) \\ K_2 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2}K_1, z_n + \frac{h}{2}L_1) \\ L_2 = g(x_n + \frac{h}{2}, y_n + \frac{h}{2}K_1, z_n + \frac{h}{2}L_1) \\ K_3 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2}K_2, z_n + \frac{h}{2}L_2) \\ L_3 = g(x_n + \frac{h}{2}, y_n + \frac{h}{2}K_2, z_n + \frac{h}{2}L_2) \\ K_4 = f(x_{n+1}, y_n + hK_3, z_n + hL_3) \\ L_4 = g(x_{n+1}, y_n + hK_3, z_n + hL_3) \end{cases}$$

四、高阶常微分方程的数值计算方法

高阶微分方程的初值问题, 原则上总可以归结为一阶方程组来求解.

- 以二阶常微分方程为例：

$$\begin{cases} y''(x) = f(x, y, y'), x_0 \le x \le x_n \\ y(x_0) = y_0, y'(x_0) = z_0 \end{cases}$$

- 令 $z = y'$, 化为一阶方程组求解:

$$\begin{cases} y'(x) = z, , x_0 \le x \le x_n \\ z'(x) = f(x, y, z), \\ y(x_0) = y_0, z(x_0) = z_0 \end{cases}$$

则问题即转化为一阶常微分方程组的问题.

9.7 Multistep Methods

Review on Previous Sections:

- Euler's Methods
- Higher-Order Taylor's Method
- Runge-Kutta's Method
- Definitions of Local Truncation Error, Global Error
- Error Control

几点注释

- 欧拉方法、泰勒方法、中点格式、欧拉修正格式、龙格库塔格式（二阶、四阶）方法等均为一步显式格式(单步法)，即计算下一个节点t_{j+1} 的函数值$y(t_{j+1})$时，只利用了当前节点t_j 的信息$y(t_j)$.
- 这些格式只需要一个初始条件$y(t_0) = \alpha$.
- 格式的构造基于Taylor 展开式(一元或二元)进行构造完成的.
- 从局部截断误差估计的角度看，欧拉格式仅为$O(h)$ 属于低精度格式, 实际应用中，采用四阶龙格库塔格式是较常见的方法.
- 从系统舍入误差和局部截断误差的累计分析结果来看，网格剖分步长并不是越小精度越高.

几点注释：其他格式构造方法：

- 差商方法:

$$\frac{y(t_{j+1}) - y(t_j)}{t_{j+1} - t_j} = \frac{y(t_j + h) - y(t_j)}{h} \approx \frac{\mathrm{d}y}{\mathrm{d}t}\Big|_{t_j}$$

- 积分方法: 把微分方程改写为

$$\mathrm{d}y = f(t, y)\mathrm{d}t$$

两端积分得(利用曲边梯形的左矩形近似计算公式，等价于对$f(t, y)$ 用分段0 次多项式插值近似逼近):

$$\begin{aligned} y(t_{j+1}) - y(t_j) &= \int_{t_j}^{t_{j+1}} \mathrm{d}y = \int_{t_j}^{t_{j+1}} f(t, y)\mathrm{d}t \\ &\approx hf(t_j, y(t_j)) \end{aligned}$$

几点注释:关于积分构造方法

- **欧拉隐式格式：** 利用曲边梯形的右矩形近似计算公式，等价于对$f(t,y)$ 用分段0 次多项式插值近似逼近):

$$
\begin{aligned}
y(t_{j+1})-y(t_j) &= \int_{t_j}^{t_{j+1}} \mathrm{d}y = \int_{t_j}^{t_{j+1}} f(t,y)\mathrm{d}t \\
&\approx hf(t_{j+1}, y(t_{j+1})). \\
y_{j+1} &= y_j + hf(t_j, y_j)
\end{aligned}
$$

- **梯形隐式格式：** 利用曲边梯形的梯形近似计算公式，等价于对$f(t,y)$ 用分段线性多项式插值近似逼近):

$$
\begin{aligned}
y(t_{j+1})-y(t_j) &= \int_{t_j}^{t_{j+1}} \mathrm{d}y = \int_{t_j}^{t_{j+1}} f(t,y)\mathrm{d}t \\
&\approx \frac{h}{2}[f(t_j, y(t_j)) + f(t_{j+1}, y(t_{j+1}))]. \\
y_{j+1} &= y_j + \frac{h}{2}[f(t_j, y_j) + f(t_{j+1}, y_{j+1}]
\end{aligned}
$$

几点注释

- 隐式格式(可以视为不动点格式)不能直接求解，需要用迭代方法求解.
- 利用欧拉显式格式作预估，用梯形隐式格式作修正，即用二阶修正格式（Runge-Kutta 二阶格式）：

$$
\begin{aligned}
y_{j+1}^{(0)} &= y_j + hf(t_j, y_j) \\
y_{j+1}^{(1)} &= y_j + \frac{h}{2}[f(t_j, y_j) + f(t_j, y_{j+1}^{(0)})] \\
y_0 &= \alpha, j = 0, 1, \cdots, N-1
\end{aligned}
$$

或者等价地写成:

$$
\begin{aligned}
y_{j+1} &= y_j + \frac{h}{2}[f(t_j, y_j) + f(t_j, y_j + hf(t_j, y_j))] \\
y_0 &= \alpha, j = 0, 1, \cdots, N-1
\end{aligned}
$$

Multistep Method

Definition 9.13: For the initial-value problem

$$\begin{cases} y' = f(t, y), \quad a \le t \le b \\ y(a) = \alpha. \end{cases} \tag{9}$$

To find the approximation y_{i+1} at the mesh point t_{i+1}, according to the following equation($m > 1$):

$$\begin{aligned} y_{i+1} \quad = \quad & a_{m-1}y_i + a_{m-2}y_{i-1} + a_{m-3}y_{i-2} + \cdots + a_0 y_{i-(m-1)} \\ & +h[b_m f(t_{i+1}, y_{i+1}) + b_{m-1}f(t_i, y_i) + \cdots + b_0 f(t_{i-(m-1)}, y_{i-(m-1)})] \\ & i = m-1, m, \cdots, N-1 \end{aligned} \tag{10}$$

where $h = (b-a)/N$, the $a_0, a_1, \cdots, a_{m-1}$ and $b_0, b_1, \cdots, b_m$ are constants, and the starting values

$$y_0 = \alpha, y_1 = \alpha_1, y_2 = \alpha_2, \cdots, y_{m-1} = \alpha_{m-1}$$

are specified.

Notes:

- When $b_m = 0$, the method is called **explicit–显式格式**, or **open**, since Eq. (10) gives y_{i+1} explicitly in terms of previously determined values.
- When $b_m \neq 0$, the method is called **implicit–隐式格式**, or **closed**, since y_{i+1} occurs on both sides of Eq. (10) and is specified only implicitly

Example:

Fourth-order Adams-Bashforth method:

$$\begin{aligned} y_0 &= \alpha, \;\; y_1 = \alpha_1, \;\; y_2 = \alpha_2, \;\; y_3 = \alpha_3 \\ y_{i+1} &= y_i + \frac{h}{24}[55f(t_i, y_i) - 59f(t_{i-1}, y_{i-1}) \\ &\quad +37f(t_{i-2}, y_{i-2}) - 9f(t_{i-3}, y_{i-3})] \end{aligned} \tag{11}$$

for each $i = 3, 4, \cdots, N-1$, define an **explicit four-step method**.

Fourth-order Adams-Moulton method:

$$
\begin{aligned}
y_0 &= \alpha,\ \ y_1 = \alpha_1,\ \ y_2 = \alpha_2, \\
y_{i+1} &= y_i + \frac{h}{24}[9f(t_{i+1}, y_{i+1}) + 19f(t_i, y_i) \\
&\quad -5f(t_{i-1}, y_{i-1}) + f(t_{i-2}, y_{i-2})]
\end{aligned} \tag{12}
$$

for each $i = 2, 3, \cdots, N-1$, define an **implicit three-step method**

Idea of difference formula design

- For the initial- value problem (9), if integrated over the interval $[t_i, t_{i+1}]$, has the property that

$$y(t_{i+1}) - y(t_i) = \int_{t_i}^{t_{i+1}} y'(t)\mathrm{d}t = \int_{t_i}^{t_{i+1}} f(t, y(t))\mathrm{d}t$$

- Consequently

$$y(t_{i+1}) = y(t_i) + \int_{t_i}^{t_{i+1}} f(t, y(t))\mathrm{d}t \qquad (13)$$

- Using an interpolating polynomial $P(t)$ instead of $f(t, y(t))$, which is determined by

$$(t_0, y_0), (t_1, y_1), \cdots, (t_i, y_i).$$

- Assume that $y(t_i) \approx y_i$, Eq. (13) becomes

$$y(t_{i+1}) \approx y_i + \int_{t_i}^{t_{i+1}} P(t)\mathrm{d}t \qquad (14)$$

Newton backward-difference formula

- To derive an Adams-Bashforth **explicit** m-step technique, we form the backward-difference polynomial $P_{m-1}(t)$ through

$$(t_i, f(t_i, y(t_i))), (t_{i-1}, f(t_{i-1}, y(t_{i-1}))), \cdots,$$

$$(t_{i-(m-1)}, f(t_{i-(m-1)}, y(t_{i-(m-1)}))).$$

- Since $P_{m-1}(t)$ is an interpolatory polynomial of degree $m-1$, some number ξ_i in $(t_{i-(m-1)}, t_i)$ exists with

$$f(t, y(t)) = P_{m-1}(t) + \frac{f^{(m)}(\xi_i, y(\xi_i))}{m!}(t-t_i)(t-t_{i-1})\cdots(t-t_{i-(m-1)}).$$

- As a consequence,

$$\int_{t_i}^{t_{i+1}} f(t, y(t))dt = h\Big[f(t_i, y(t_i)) + \frac{1}{2}\nabla f(t_i, y(t_i))$$

$$+\frac{5}{12}\nabla^2 f(t_i, y(t_i)) + \frac{3}{8}\nabla^3 f(t_i, y(t_i)) + \cdots\Big]. \qquad (15)$$

Since

$$y(t_{i+1}) - y(t_i) = \int_{t_i}^{t_{i+1}} f(t, y(t))\mathrm{d}t,$$

It can be written as

$$\begin{aligned} y(t_{i+1}) &= y(t_i) + h\Big[f(t_i, y(t_i)) + \frac{1}{2}\nabla f(t_i, y(t_i)) \\ &\quad + \frac{5}{12}\nabla^2 f(t_i, y(t_i)) + \cdots\Big] \\ &\quad + h^{m+1} f^{(m)}(\mu_i, y(\mu_i))(-1)^m \int_0^1 \begin{pmatrix} -s \\ m \end{pmatrix} \mathrm{d}s \end{aligned} \tag{16}$$

three-step Adams-Bashforth method

Consider Eq. (16) with $m = 3$:

$$
\begin{aligned}
y(t_{i+1}) &\approx y(t_i) + h\Big[f(t_i, y(t_i)) + \frac{1}{2}\nabla f(t_i, y(t_i)) \\
&\quad + \frac{5}{12}\nabla^2 f(t_i, y(t_i))\Big] \\
&= y(t_i) + h\Big\{f(t_i, y(t_i)) + \frac{1}{2}[f(t_i, y(t_i)) \\
&\quad - f(t_{i-1}, y(t_{i-1}))] \\
&\quad + \frac{5}{12}[f(t_i, y(t_i)) - 2f(t_{i-1}, y(t_{i-1})) \\
&\quad + f(t_{i-2}, y(t_{i-2}))]\Big\} \\
&= y(t_i) + \frac{h}{12}[23f(t_i, y(t_i)) \\
&\quad - 16f(t_{i-1}, y(t_{i-1})) + 5f(t_{i-2}, y(t_{i-2}))].
\end{aligned}
$$

Three-step Adams-Bashforth method

$$\begin{cases} y_0 = \alpha, y_1 = \alpha_1, y_2 = \alpha_2, \\ y_{i+1} = y_i + \frac{h}{12}[23f(t_i, y_i) - 16f(t_{i-1}, y_{i-1}) + 5f(t_{i-2}, y_{i-2})] \end{cases}$$

for $i = 2, 3, \cdots, N-1$.

Error analysis

Definition 9.14

- If $y(t)$ is the solution to the initial-value problem $y' = f(t, y), a \leq t \leq b, y(a) = \alpha$, and

$$\begin{aligned} y_{i+1} &= a_{m-1}y_i + a_{m-2}y_{i-1} + \cdots + a_0 y_{i+1-m} \\ &\quad + h[b_m f(t_{i+1}, y_{i+1}) + b_{m-1} f(t_i, y_i) + \cdots \\ &\quad + b_0 f(t_{i+1-m}, y_{i+1-m})] \end{aligned}$$

  is the $(i+1)$st step in a multistep method

- the local truncation error at this step is

$$\begin{aligned} \tau_{i+1}(h) &= \frac{y(t_{i+1}) - a_{m-1}y(t_i) + \cdots + a_0 y(t_{i+1-m})}{h} \\ &\quad - [b_m f(t_{i+1}, y(t_{i+1})) + b_{m-1} f(t_i, y(t_i)) + \cdots \\ &\quad + b_0 f(t_{i+1-m}, y(t_{i+1-m}))] \end{aligned} \tag{17}$$

  for each $i = m-1, m, \cdots, N-1$.

The local truncation error for the three-step Adams-Bashforth method:

- Consider the form of the error term in Eq.(16) with $m = 3$, we have

$$h^4 f^{(3)}(\mu_i, y(\mu_i))(-1)^3 \int_0^1 \left(\begin{array}{c} -s \\ 3 \end{array} \right) \mathrm{d}s. = \frac{3h^4}{8} y^{(4)}(\mu_i),$$

  for some $\mu_i \in (t_{i-2}, t_{i+1})$.

- Using the fact that $f^{(3)}(\mu_i, y(\mu_i)) = y^{(4)}(\mu_i)$ and the difference equation derived in the three-step Adams-Bashforth method, we have

$$\begin{aligned}
\tau_{i+1}(h) &= \frac{y(t_{i+1}) - y(t_i)}{h} \\
&\quad -\frac{1}{12}[23f(t_i, y(t_i)) - 16f(t_{i-1}, y(t_{i-1})) + 5f(t_{i-2}, y(t_{i-2}))] \\
&= \frac{1}{h}\left[\frac{3h^4}{8} f^{(3)}(\mu_i, y(\mu_i))\right] \\
&= \frac{3h^3}{8} y^{(4)}(\mu_i), \text{for some} \mu_i \in (t_{i-2}, t_{i+1}).
\end{aligned}$$

Some examples of the explicit multi-step methods

Adams-Bashforth Two-Step Method:

- **Adams-Bashforth Two-Step Method:**

$$\begin{cases} y_0 = \alpha, y_1 = \alpha_1, \\ y_{i+1} = y_i + \frac{h}{2}[3f(t_i, y_i) - f(t_{i-1}, y_{i-1}))] \end{cases} \tag{18}$$

  for $i = 1, 2, \cdots, N-1$.

- The local truncation error is

$$\tau_{i+1}(h) = \frac{5}{12} h^2 y'''(\mu_i),$$

  for some $\mu_i \in (t_{i-1}, t_{i+1})$.

Adams-Bashforth Three-Step Method:

- **Adams-Bashforth Three-Step Method:**

$$\begin{cases} y_0 = \alpha, y_1 = \alpha_1, y_2 = \alpha_2, \\ y_{i+1} = y_i + \frac{h}{12}[23f(t_i, y_i) - 16f(t_{i-1}, y_{i-1}) + 5f(t_{i-2}, y_{i-2})] \end{cases} \tag{19}$$

  for $i = 2, 3, \cdots, N-1$.

- The local truncation error is

$$\tau_{i+1}(h) = \frac{3}{8} y^{(4)}(\mu_i) h^3.$$

  for some $\mu_i \in (t_{i-2}, t_{i+1})$.

Adams-Bashforth Four-Step Method:

- **Adams-Bashforth Four-Step Method:**

$$\left\{\begin{array}{l} y_0 = \alpha, y_1 = \alpha_1, y_2 = \alpha_2, y_3 = \alpha_3, \\ y_{i+1} = y_i + \frac{h}{24}[55f(t_i, y_i) - 59f(t_{i-1}, y_{i-1})) \\ \qquad\qquad +37f(t_{i-2}, y_{i-2})) - 9f(t_{i-3}, y_{i-3}))] \end{array}\right. \tag{20}$$

  for $i = 3, 4, \cdots, N-1$.

- The local truncation error is

$$\tau_{i+1}(h) = \frac{251}{720} y^{(5)}(\mu_i) h^4$$

  for some $\mu_i \in (t_{i-3}, t_{i+1})$.

Adams-Bashforth Five-Step Method:

- **Adams-Bashforth Five-Step Method:**

$$
\begin{aligned}
y_0 &= \alpha, y_1 = \alpha_1, y_2 = \alpha_2, y_3 = \alpha_3, y_4 = \alpha_4 \\
y_{i+1} &= y_i + \frac{h}{720}[1901f(t_i, y_i) - 2774f(t_{i-1}, y_{i-1}) \\
&\quad +2616f(t_{i-2}, y_{i-2}) - 1274f(t_{i-3}, y_{i-3}) \\
&\quad +251f(t_{i-4}, y_{i-4})] \qquad (21)
\end{aligned}
$$

for $i = 4, 5, \cdots, N-1$.

- The local truncation error is

$$\tau_{i+1}(h) = \frac{95}{288} y^{(6)}(\mu_i) h^5,$$

for some $\mu_i \in (t_{i-4}, t_{i+1})$.

implicit methods

- Implicit methods are derived by using

$$(t_{i+1}, f(t_{i+1}, y(t_{i+1})))$$

as an additional interpolation node in the approximation of the integral

$$\int_{t_i}^{t_{i+1}} f(t, y(t))\mathrm{d}t.$$

- Some of the more common **implicit methods** are as follows.

Some examples of common implicit methods:

Adams-Multon Two-Step Method:

- **Adams-Multon Two-Step Method:**

$$\begin{cases} y_0 = \alpha, y_1 = \alpha_1, \\ y_{i+1} = y_i + \frac{h}{12}[5f(t_{i+1}, y_{i+1}) + 8f(t_i, y_i)) - f(t_{i-1}, y(t_{i-1}))] \end{cases} \tag{22}$$

  for $i = 1, 2, \cdots, N-1$.

- The local truncation error is

$$\tau_{i+1}(h) = -\frac{1}{24} y^{(4)}(\mu_i) h^3$$

  for some $\mu_i \in (t_{i-1}, t_{i+1})$.

Adams-Moulton Three-Step Method:

- **Adams-Moulton Three-Step Method:**

$$\begin{cases} y_0 = \alpha, y_1 = \alpha_1, y_2 = \alpha_2, \\ y_{i+1} = y_i + \frac{h}{24}[9f(t_{i+1}, y_{i+1}) + 19f(t_i, y_i) \\ \qquad -5f(t_{i-1}, y_{i-1}) + f(t_{i-2}, y_{i-2})] \end{cases} \tag{23}$$

  for $i = 2, 3, \cdots, N-1$.

- The local truncation error is

$$\tau_{i+1}(h) = -\frac{19}{720} y^{(5)}(\mu_i) h^4$$

  for some $\mu_i \in (t_{i-2}, t_{i+1})$.

Adams-Moulton Four-Step Method:

- **Adams-Moulton Four-Step Method:**

$$\begin{aligned} y_0 &= \alpha, y_1 = \alpha_1, y_2 = \alpha_2, y_3 = \alpha_3, \\ y_{i+1} &= y_i + \frac{h}{720}[251f(t_{i+1}, y_{i+1})) + 646f(t_i, y_i) \\ &\quad -264f(t_{i-1}, y_{i-1})) + 106f(t_{i-2}, y_{i-2})) \\ &\quad -19f(t_{i-3}, y_{i-3}))] \end{aligned} \tag{24}$$

  for $i = 3, 4, \cdots, N-1$.

- The local truncation error is

$$\tau_{i+1}(h) = -\frac{3}{160} y^{(6)}(\mu_i) h^5,$$

  for some $\mu_i \in (t_{i-3}, t_{i+1})$.

Predictor-Corrector Method

- The combination of an explicit and implicit technique is called a **predictor-corrector method**.
- The explicit method predicts an approximation, and the implicit method corrects this prediction.

Example

- Consider the following fourth-order method for solving an initial-value problem.
- The first step is to calculate the starting values y_0, y_1, y_2, and y_3 using Runge-Kutta method of order four.
- The next step is to calculate an approximation, $y_4^{(0)}$, to $y(t_4)$ using the four-step Adams- Bashforth method as **predictor**:

$$\begin{aligned} y_4^{(0)} &= y_3 + \frac{h}{24}[55f(t_3, y_3) - 59f(t_2, y_2)) + \\ &\quad +37f(t_1, y_1)) - 9f(t_0, y_0))] \end{aligned}$$

- This approximation is improved by inserting $y_4^{(0)}$ in the right side of the three-step Adams- Moulton method and using that method as a **corrector**:

$$\begin{aligned} y_{i+1}^{(1)} &= y_i + \frac{h}{24}[9f(t_{i+1}, y_{i+1}^{(0)}) + 19f(t_i, y_i) \\ &\quad -5f(t_{i-1}, y_{i-1}) + f(t_{i-2}, y_{i-2})], i = 3, 4, \cdots \end{aligned}$$

ALGORITHM: Adams Fourth-Order Predictor-Corrector

- using Runge-Kutta method of order four to estimate starting values y_0, y_1, y_2, and y_3
- using the four-step Adams- Bashforth method as **predictor**:

$$\begin{aligned} y_4^{(0)} &= y_3 + \frac{h}{24}[55f(t_3, y_3) - 59f(t_2, y_2)) + \\ &\quad +37f(t_1, y_1)) - 9f(t_0, y_0))] \end{aligned}$$

- Using three-step Adams- Moulton method as a **corrector**:

$$\begin{aligned} y_{i+1}^{(1)} &= y_i + \frac{h}{24}[9f(t_{i+1}, y_{i+1}^{(0)}) + 19f(t_i, y_i) \\ &\quad -5f(t_{i-1}, y_{i-1}) + f(t_{i-2}, y_{i-2})], i = 3, 4, \cdots \end{aligned}$$

INPUT: endpoints a, b; integer N; initial condition α.

OUTPUT approximation y to $y(t)$ at the $(N+1)$ values of t.

Step 1 Set $h = (b-a)/N$; $t_0 = a$; $y_0 = \alpha$; OUTPUT (t_0, y_0).

Step 2 For $i = 1, 2, 3$, do Steps 3-5. (Compute starting values using Runge-Kutta method.)

Step 3 Set

$$\begin{aligned} K_1 &= hf(t_{i-1}, y_{i-1}); \\ K_2 &= hf(t_{i-1} + h/2, y_i + K_1/2); \\ K_3 &= hf(t_{i-1} + h/2, y_i + K_2/2) \\ K_4 &= hf(t_{i-1} + h, y_i + K_3). \end{aligned}$$

Step 4 Set

$$y_i = y_{i-1} + (K_1 + 2K_2 + 2K_3 + K_4)/6;$$

$$t_i = a + ih.$$

Step 5 OUTPUT (t_i, y_i).

Step 6 For $i = 4, \cdots, N$ do Steps 7-10.

Step 7 Set $t = a + ih$;

$$\begin{aligned} y &= y_3 + h[55f(t_3, y_3) - 59f(t_2, y_2) \\ &\quad +37f(t_1, y_1) - 9f(t_0, y_0)]/24; (\text{Predict} y_i) \\ y &= y_3 + h[9f(t, y) + 19f(t_3, y_3) \\ &\quad -5f(t_2, y_2) + f(t_1, y_1)]/24; (\text{Correct} y_i) \end{aligned}$$

Step 8 OUTPUT (t, y).

Step 9 For $j = 0, 1, 2$
set $t_j = t_{j+1}$; (Prepare for next iteration.)
$y_j = y_{j+1}$.

Step 10 Set $t_3 = t$;
$y_3 = y$.

Step 11 STOP.

Milne's method

- Other multistep methods can be derived using integration of interpolating polynomials over intervals of the form $[t_j, t_{i+1}]$, for $j \le i-1$, to obtain an approximation to $y(t_{i+1})$.

$$y(t_{i+1}) - y(t_j) = \int_{t_j}^{t_{i+1}} \mathrm{d}y = \int_{t_j}^{t_{i+1}} f(t, y)\mathrm{d}t, \quad j \le i-1.$$

- When an interpolating polynomial is integrated over $[t_{i-3}, t_{i+1}]$, the result is an explicit technique known as **Milne's** method(**Predictor**)

$$y_{i+1} = y_{i-3} + \frac{4h}{3}[2f(t_i, y_i) - f(t_{i-1}, y_{i-1}) + 2f(t_{i-2}, y_{i-2})],$$

which has local truncation error $\frac{14}{45}h^4 y^{(5)}(\xi_i)$, for some $\xi_i \in (t_{i-3}, t_{i+1})$.

- This method is occasionally used as a **predictor** for the **implicit Simpson's method**,

$$y_{i+1} = y_{i-1} + \frac{h}{3}[f(t_{i+1}, y_{i+1}) + 4f(t_i, y_i) + f(t_{i-1}, y_{i-1})],$$

  which has local truncation error $-\frac{1}{90}h^4 y^{(5)}(\xi_i)$, for some $\xi_i \in (t_{i-1}, t_{i+1})$, and is obtained by integrating an interpolating polynomial over $[t_{i-1}, t_{i+1}]$.
- The local truncation error involved with a predictor-corrector method of the Milne- Simpson type is generally smaller than that of the Adams-Bashforth-Moulton method.
- The technique has limited use because of **problems of stability**, which do not occur with the Adams procedure.

Reviews on Error Control Methods

1. The Runge- Kutta- Fehlberg method is used for error control because at each step it provides, at little additional cost, two approximations that can be compared and related to the local error.
2. Predictor-corrector techniques always generate two approximations at each step, so they are natural candidates for error-control adaptation

- To demonstrate the error-control procedure, we will construct a variable step-size.
- Predictor-corrector method using the **four-step Adams-Bashforth method as predictor** and **the three-step Adams-Moulton method as corrector.**

- **Adams-Bashforth four-step method** comes from the relation

$$\begin{aligned} y(t_{i+1}) &= y(t_i) + \frac{h}{24}[55f(t_i, y(t_i)) \\ &\quad -59f(t_{i-1}, y(t_{i-1})) + 37f(t_{i-2}, y(t_{i-2})) \\ &\quad -9f(t_{i-3}, y(t_{i-3}))] + \frac{251}{720} y^{(5)}(\hat{\mu}_i) h^5, \end{aligned}$$

  for some $\hat{\mu}_i \in (t_{i-3}, t_{i+1})$.
- The assumption that the approximations $y_0, y_1, y_2, \cdots, y_i$ are all exact implies that the Adams- Bashforth truncation error is

$$\frac{y(t_{i+1}) - y_{i+1}^{(0)}}{h} = \frac{251}{720} y^{(5)}(\hat{\mu}_i) h^4. \tag{25}$$

- A similar analysis of the **Adams-Moulton three-step method**, which comes'from

$$\begin{aligned} y(t_{i+1}) &= y(t_i) + \frac{h}{24}[9f(t_{i+1}, y(t_{i+1})) + 19f(t_i, y(t_i)) \\ &\quad -5f(t_{i-1}, y(t_{i-1})) + f(t_{i-2}, y(t_{i-2}))] \\ &\quad -\frac{19}{720} y^{(5)}(\tilde{\mu}_i) h^4, \end{aligned}$$

- for some $\tilde{\mu}_i \in (t_{i-2}, t_{i+1})$ leads to the local truncation error

$$\frac{y(t_{i+1}) - y_{i+1}}{h} = -\frac{19}{720} y^{(5)}(\tilde{\mu}_i) h^4. \quad (26)$$

- To proceed further, we must make the assumption that for small values of h,

$$y^{(5)}(\hat{\mu}_i) = y^{(5)}(\tilde{\mu}_i).$$

- The effectiveness of the error-control technique depends directly on this assumption.
- If we subtract Eq. (26) from Eq. (25), we have

$$\begin{aligned}
\frac{y_{i+1} - y_{i+1}^{(0)}}{h} &= -\frac{h^4}{720}[251y^{(5)}(\hat{\mu}_i) + 19y^{(5)}(\tilde{\mu}_i)] \\
&\approx \frac{3}{8}h^4 y^{(5)}(\tilde{\mu}_i),
\end{aligned}$$

so

$$y^{(5)}(\tilde{\mu}_i) \approx \frac{8}{3h^5}(y_{i+1} - y_{i+1}^{(0)}) \tag{27}$$

- Using this result to eliminate the term involving $y^{(5)}(\tilde{\mu}_i)h^4$ from (26), gives the approximation to the error

$$
\begin{aligned}
|\tau_{i+1}(h)| &= \frac{|y(t_{i+1}) - y_{i+1}|}{h} \\
&\approx \frac{19h^4}{720} \cdot \frac{8}{3h^5} |y_{i+1} - y_{i+1}^{(0)}| \\
&= \frac{19|y_{i+1} - y_{i+1}^{(0)}|}{270h}.
\end{aligned}
$$

- Suppose we now reconsider (26) with a new step size qh generating new approximations $\hat{y}_{i+1}^{(0)}$ and $\hat{y}_{i+1}$.
- The object is to choose q so that the local truncation error given in (26) is bounded by a prescribed tolerance ε.

- If we assume that the value $y^{(5)}(\mu)$ in (26) associated with qh is also approximated using (27), then

$$\begin{aligned}\frac{|y(t_i+qh)-\hat{y}_{i+1}|}{qh} &= \frac{19}{720}|y^{(5)}(\mu)|q^4h^4 \\ &\approx \frac{19}{720}\left[\frac{8}{3h^5}|y_{i+1}-y_{i+1}^{(0)}|\right]q^4h^4,\end{aligned}$$

- we need to choose q so that

$$\frac{|y(t_i+qh)-\hat{y}_{i+1}|}{qh} \approx \frac{19}{720}\frac{|y_{i+1}-y_{i+1}^{(0)}|}{h}q^4 < \varepsilon,$$

- That is, we choose q so that

$$q < \left(\frac{270}{19}\frac{h\varepsilon}{|y_{i+1}-y_{i+1}^{(0)}|}\right)^{1/4} \approx 2\left(\frac{h\varepsilon}{|y_{i+1}-y_{i+1}^{(0)}|}\right)^{1/4}.$$

- A number of approximation assumptions have been made in this development
- In practice q is chosen conservatively, usually as

$$q = 1.5\left(\frac{h\varepsilon}{|y_{i+1} - y_{i+1}^{(0)}|}\right)^{1/4}.$$

- A change in step size for a multistep method is more costly in terms of function evaluations than for a one-step method since new, equally-spaced starting values must be computed
- As a consequence, it is common practice to ignore the step-size change whenever the local truncation error is between $\varepsilon/10$ and ε, that is, when

$$\begin{aligned}\frac{\varepsilon}{10} \quad &< \quad |\tau_{i+1}(h)| = \frac{|y(t_{i+1}) - y_{i+1}|}{h} \\ &\approx \quad \frac{19|y_{i+1} - y_{i+1}^{(0)}|}{270h} < \varepsilon.\end{aligned}$$

Adams Variable Step-Size Predictor-Corrector

To approximate the solution of the initial-value problem

$$y' = f(t, y), a \leq t \leq b, y(a) = \alpha$$

with local truncation error within a given tolerance :

INPUT endpoints a, b; initial condition a; tolerance TOL; maximum step size $hmax$; minimum step size $hmin$.

OUTPUT i, t_i, y_i, h where at the ith step w_i approximates $y(t_i)$ and the step size h was used, or a message that the minimum step size was exceeded.

The steps of this algorithm are omitted.

Definition 9.17

A one-step difference-equation method with **local truncation error** $\tau_i(h)$ at the ith step is said to be **consistent**(相容性) with the differential equation it approximates, if

$$\lim_{h \to 0} \max_{1 \le i \le N} |\tau_i(h)| = 0.$$

Note:

1. this definition is a $local$ definition , since , for each of the values $\tau_i(h)$, we are assuming that the approximation y_{i-1} and the exact solution $y(t_{i-1})$ are the same.
2. A more realistic means of analyzing the effects of making h small is to determine the **global** effect of the method.
3. This is the maximum error of the method over the entire range of the approximation, assuming only that the method gives the exact result at the initial value.

Definition 9.18

A **one step difference equation** method is said to be **convergent** with respect to the differential equation it approximates if

$$\lim_{h \to 0} \max_{1 \le i \le N} |w_i - y(t_i)| = 0$$

where $y_i = y(t_i)$ denotes the exact value of the solution of the differential equation and w_i is the approximation obtained form the difference method at the ith step.

EXAMPLE: Consider the Euler's method.

- By the hypotheses of Theorem 5.9, we have

$$\max_{1\le i\le N} |w_i - y(t_i)| \le \frac{Mh}{2L}|e^{L(b-a)} - 1|.$$

  so the Euler's method is convergent with respect to a differential equation satisfying the conditions of this theorem , and the rate of convergence is $O(h)$.
- A one-step method is consistent precisely when the difference equation for the method approaches the differential equation when the step size goes to zero; that is , the local truncation error approaches zero as the step size approaches zero.
- The definition of convergence has a similar connotation.
- A method is **convergent** precisely when the solution to the difference equation approaches the solution to the differential equation as the step size goes to zero.

Stability

Definition 9.19

- A numerical method for initial value problems of Ordinary Differential Equations is **stable**, if in the sense that small changes or perturbations in the initial conditions, produce correspondingly small changes in the subsequent approximation.
- That is, a stable method is the one whose results depend continuously on the initial data.

THEOREM 9.20

- Suppose the initial-value problem

$$y' = f(t, y), a \leq t \leq b, y(a) = \alpha$$

  is approximated by a **one-step difference method** in the form

$$w_o = \alpha$$

$$w_{i+1} = w_i + h\phi(t_i, w_i, h)$$

- Suppose also that a number $h_0 > 0$ exists and that $\phi(t, w, h)$ is **continuous** and satisfies a **Lipschitz condition** in the variable w with Lipschitz constant L on the set

$$D = (t, w, h)|a \leq t \leq b, -\infty < w < \infty, 0 \leq h \leq h_0$$

- Then
  1. The method is **stable**;
  2. the difference method is **convergent** if and only if it is consistent–that is ,if and only if

$$\phi(t, y, 0) = f(t, y), \texttt{for all } a \leq t \leq b$$

  3. If a function τ exists, and for each $i = 1, 2, \cdots, N$, the local truncation error$\tau_i(h)$ satisfies $|\tau_i(h)| \leq \tau(h)$ whenever $0 \leq h \leq h_0$ then

$$|y(t_i) - w_i| \leq \frac{\tau(h)}{L} d^{L(t_i - a)}$$

- **As an example,** we prove the **Modified Euler's method**, given by

$$\begin{aligned} w_0 &= \alpha \\ w_{i+1} &= w_i + \frac{h}{2}[f(t_i, w_i) + f(t_{i+1}, w_i + hf(t_i, w_i))] \end{aligned}$$

for $i = 0, 1, 2, \cdots, N-1$, satisfies the hypothesis of THEOREM 9.20.