常微分方程数值解法

内容提要

1、引言
2、欧拉法、梯形法和改进欧拉法
3、龙格一库塔法
4、多步法
5、Adam法
6、Gear法
7、数值稳定性

对于一个常微分方程：*自变量、未知函数、未知函数的导数*

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

通常会有无穷个解。如：

$$\frac{dy}{dx} = \cos(x) \quad \Rightarrow \quad y = \sin(x) + a$$

因此，我们要加入一个限定条件。通常会在端点处给出，如下面的初值问题：

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

常微分方程数值解—问题的提出

本课程我们仅仅学习常微分方程的数值解法。所研究的常微分方程的形式为：

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

如果用如下形式表示：

$$\dot{x} = f(x, t) \qquad x(t_0) = x_0$$

- $x(t)$是随时间而变化的状态变量，依赖于初值x_0，而这种微分方程的求解问题称为常微分方程的初值问题。

数值解的基本做法

对式(1)进行数值求解的过程，就是根据x_0时刻的初始值y_0，依次计算x_1时刻$y(x_1)$的近似值y_1，x_2时刻$y(x_2)$的近似值y_2…。

其中相邻时间的间隔被称为步长，通常在整个积分区域$x\in(x_0,x_N)$，步长$h_{n+1}=x_{n+1}-x_n$都被取定值。

基本的算法就是从x_n时已知的y_n、y_{n-1}…和$f(x_n,y_n)$、$f(x_{n-1},y_{n-1})$…推出x_{n+1}时的值y_{n+1}。

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

微分方程数值算法的选择准则

任何实用的数值算法都必须满足以下的标准：

1.数值计算的精确度

2.数值计算的稳定性

3.数值计算的效率

- 数值计算的精确度是指每一步数值计算的误差都是有界的。其中整体误差＝｜$y(x_n)-y_n$｜
- 数值计算的稳定性是指每一步数值计算产生的误差不至于影响到以后的计算。
- 数值计算的效率则与计算量和步长大小有关。

第2 节
欧拉法、梯形法和改进欧拉法

函数的泰勒级数展开（1）

用表示式（I）的精确解，将在x=x_n点泰勒展开，并计算级数在x=x_{n+1}时的值，可得下式：

$$
\begin{aligned}
& y(x_{n+1}) \\
& = y(x_n) + y'(x_n)(x_{n+1} - x_n) \\
& + \frac{1}{2} y''(x_n)(x_{n+1} - x_n)^2 + \cdots \\
& + \frac{1}{p!} y^{(p)}(x_n)(x_{n+1} - x_n)^p + h.o.t.
\end{aligned}
$$

展开式中更高次项

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

$$\sum_{n=0}^{\infty} \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n$$

函数 f 在点 $x = x_0$ 处的泰勒展开式
注意：此时未知函数为 y

函数的泰勒级数展开（2）

如果时间步长h=x_{n+1}-x_n,则

$$y(x_{n+1}) = y(x_n) + hy'(x_n) + \frac{h^2}{2}y''(x_n) + \ldots + \frac{h^p}{p!}y^{(p)}(x_n) + h.o.t.$$

由式（1）可知, $y'(x) = f(x, y)$ 所以

$$y(x_{n+1}) - \underbrace{h.o.t.}_{\text{展开式中更高次项}} = y(x_n) + hf(x_n, y(x_n)) + \frac{h^2}{2}f'(x_n, y(x_n)) + \ldots + \frac{h^p}{p!}f^{(p-1)}(x_n, y(x_n)) \quad （2）$$

如果高次项非常小，则可用由式（2）等式右边计算出来的y_{n+1}来 $y(x_{n+1})$作为的近似值。

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

函数的泰勒级数展开（3）

通常，泰勒级数法可以表示为

$$y_{n+1} = y_n + hT_p(y_n) \qquad (3)$$

式中

$$T_p(y_n) = f(x_n, y(x_n)) + \frac{h}{2} f'(x_n, y(x_n)) + \ldots + \frac{h^{p-1}}{p!} f^{(p-1)}(x_n, y(x_n))$$

其中**整数p称为阶**。对于较大的p，用泰勒级数法可以非常精确，但计算效率却不高。

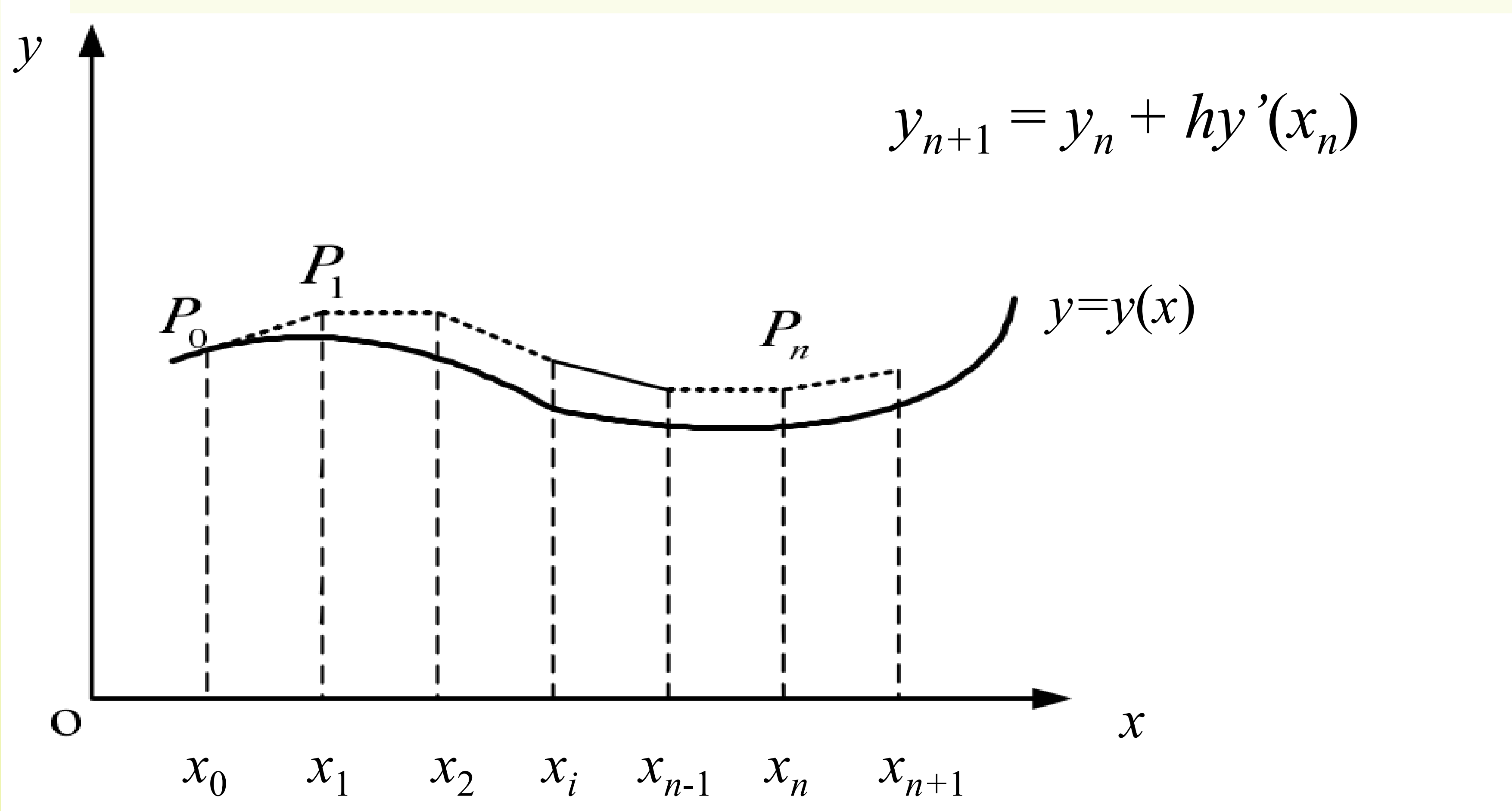
欧拉法

当p=1时，泰勒级数法变为：

$$y_{n+1} = y_n + hf(x_n, y_n) \qquad (4)$$

式(4)称为**欧拉法**。

欧拉法的几何意义



欧拉法的数值积分推导

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

根据数值积分的左矩形公式，有

$$\int_{x_n}^{x_{n+1}} y'(x)dx \approx (x_{n+1} - x_n)y'(x_n) = hy'_n$$

因此，有

$$\boxed{y(x_{n+1}) \approx y(x_n) + hy'_n = y(x_n) + h \cdot f(x_n, y_n)}$$

欧拉法的数值微分推导？

欧拉法的误差与精度

$y(x_{n+1})$在点(x_n,y_n)处的泰勒展开式

$$y(x_{n+1}) = y(x_n) + hy'(x_n) + \frac{h^2}{2} y''(\xi) \quad x_n \le \xi \le x_{n+1}$$

利用欧拉法求得的近似值 y_{n+1}

$$y_{n+1} = y_n + hy'(x_n)$$

假定 y_n 没有误差，即 $y_n = y(x_n)$

则误差 $$R = y(x_{n+1}) - y_{n+1} = \frac{h^2}{2} y''(\xi) = O(h^2)$$

如果误差为$O(h^{p+1})$，则此种算法的精度为 p 阶。

所以欧拉法的精度为一阶。

用Euler方法求解问题

取h=0.1

$$\begin{cases} \dfrac{\mathrm{d}y}{\mathrm{d}x} = -y + x + 1 \quad 0 \le x \le 0.5 \\ y(0) = 1 \end{cases}$$

用Euler方法求解问题

取h=0.1

$$\begin{cases} \dfrac{\mathrm{d}y}{\mathrm{d}x} = -y + x + 1 \quad 0 \le x \le 0.5 \\ y(0) = 1 \end{cases}$$

解 设$f(x,y)=-y+x+1$，$x_0=0, y_0=1, x_n=x_0+nh=0.1n$（$n=0,1,\ldots,5$）

Euler格式为 $y_{n+1} = y_n + hf(x_n, y_n) = y_n + 0.1(-y_n + x_n + 1)$

由$y_0=1$出发，按上面公式的计算结果如表所示

x_n	y_n
0	1.000 000
0.1	1.000 000
0.2	1.010 000
0.3	1.029 000
0.4	1.056 100
0.5	1.090 490

$$
\begin{aligned}
y_{n+1} &= y_n + 0.1(-y_n + x_n + 1) \\
&= 0.9y_n + 0.1x_n + 0.1
\end{aligned}
$$

欧拉法算例（I）

试用欧拉法计算下列初值问题。

$$y'(x)=1-\frac{2xy}{1+x^2},\ y(0)=0,\ 0\le x\le 2$$

取步长h＝0.5，并与精确值 $y(x)=\dfrac{x(3+x^2)}{3(1+x^2)}$ 比较。

解：由欧拉法得：

有：

$$y_{n+1} = y_n + h(1 - \frac{2x_n y_n}{1 + x_n^{\ 2}})$$

$$y_0 = 0, n = 0, 1, 2, 3$$

n	x_n=nh=0.5n	y_n	$y(x_n)$精确值
0	0	0	0
1	0.5	0.500000	0.433333
2	1.0	0.800000	0.666667
3	1.5	0.900000	0.807692
4	2.0	0.984615	0.933333

后退欧拉法

如果计算y_{n+1}时，所取的斜率不是x_n点上的导数$f(x_n,y_n)$，而是x_{n+1}点上的导数$f(x_{n+1}, y_{n+1})$，就得到后退欧拉法

$$y_{n+1} = y_n + hf(x_{n+1}, y_{n+1}) \qquad (5)$$

以后会说明，后退欧拉法比欧拉法具有好得多的数值稳定性。

欧拉法 $$y_{n+1} = y_n + hf(x_n, y_n) \qquad (4)$$

后退欧拉法的数值积分推导

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

根据数值积分的右矩形公式，有

$$\int_{x_n}^{x_{n+1}} y'(x)dx \approx (x_{n+1} - x_n)y'(x_{n+1}) = hy'(x_{n+1})$$

因此，有

$$y(x_{n+1}) \approx y(x_n) + hy'(x_{n+1}) = y(x_n) + h \cdot f(x_{n+1,} y_{n+1})$$

后退欧拉法的误差与精度

误差 $R = y(x_{n+1}) - y_{n+1} = -\frac{h^2}{2} y''(\xi) = O(h^2)$

所以后退欧拉法的精度为一阶。

欧拉法和后退欧拉法的局部截断误差在数值上是相等的，但方向相反。

后退欧拉法算例

试用后退欧拉法计算下列初值问题

$$y'(x)=1-\frac{2xy}{1+x^2},\ y(0)=0,\ 0\le x\le 2$$

取步长h＝0.5，并与精确值 $y(x)=\frac{x(3+x^2)}{3(1+x^2)}$ 比较。

$$y_{n+1} = y_n + h(1 - \frac{2x_{n+1} y_{n+1}}{1+x_{n+1}^2})$$

$$y_{n+1} = \frac{y_n + h}{1 + \frac{2hx_{n+1}}{1+x_{n+1}^2}} \qquad y_0 = 0, n = 0,1,2,3,4$$

n	x_n=nh=0.5n	y_n	$y(x_n)$精确值
0	0	0	0
1	0.5	0.357142	0.433333
2	1.0	0.571428	0.666667
3	1.5	0.733082	0.807692
4	2.0	0.880773	0.933333

梯形法

如果计算y_{n+1}时，所取的斜率不是x_n点上的导数$f(x_n,y_n)$，而是x_n点上的导数$f(x_n,y_n)$和x_{n+1}点上的导数$f(x_{n+1}, y_{n+1})$的**平均值**，就得到梯形法

$$y_{n+1} = y_n + h\frac{y'(x_n) + y'(x_{n+1})}{2} = y_n + h\frac{f(x_n, y_n) + f(x_{n+1}, y_{n+1})}{2}$$

梯形法

梯形法的数值积分推导

根据数值积分的梯形公式，有 $y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} y'(x)dx$

$$\int_{x_n}^{x_{n+1}} y'(x)dx \approx (x_{n+1} - x_n)\frac{y'(x_n) + y'(x_{n+1})}{2}$$

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

因此

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

梯形法算例

试用梯形法计算下列初值问题。

$$y'(x)=1-\frac{2xy}{1+x^2},y(0)=0,0\le x\le 2$$

取步长h＝0.5，并与精确值 $y(x)=\frac{x(3+x^2)}{3(1+x^2)}$ 比较。

$$y_{n+1} = y_n + h\frac{f(x_n, y_n) + f(x_{n+1}, y_{n+1})}{2} \qquad y'(x) = 1 - \frac{2xy}{1+x^2}$$

解：由梯形法得

$$y_{n+1} = \frac{y_n + h - \dfrac{hx_n y_n}{1+x_n^2}}{1 + \dfrac{hx_{n+1}}{1+x_{n+1}^2}} \qquad y_0 = 0, n = 0,1,2,3,4$$

n	x_n=nh=0.5n	y_n	$y(x_n)$精确值
0	0	0	0
1	0.5	0.416667	0.433333
2	1.0	0.666667	0.666667
3	1.5	0.812500	0.807692
4	2.0	0.937500	0.933333

改进欧拉公式

在梯形公式

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

的右端中包含有未知的y_{n+1}，这类数值方法称为**隐式方法**，一般情况下不能直接求解上述方程，而需要采用**迭代**的方法来求解。

改进欧拉公式（2）

一种简单的做法是先用欧拉法计算出y_{n+1}的近似值，然后将这个近似值再代入到梯形公式中，即采用如下的格式：

$$y_{n+1}^{0}=y_n+h\cdot f(x_{n,}y_n)$$

$$y_{n+1}=y_n+\frac{h}{2}[f(x_n,y_n)+f(x_{n+1},y_{n+1}^{0})]$$

这种格式就称为**改进的欧拉公式**，也叫**预报—校正格式**。

改进的欧拉公式算例（I）

试用改进的欧拉公式计算下列初值问题。

$$y'(x)=1-\frac{2xy}{1+x^2},\ y(0)=0,\ 0\le x\le 2$$

取步长h＝0.5，并与精确值 $y(x)=\dfrac{x(3+x^2)}{3(1+x^2)}$ 比较。

解：

$$y_{n+1}^{0} = y_n + h \cdot f(x_n, y_n)$$

$$y_{n+1} = y_n + \frac{h}{2}[f(x_n, y_n) + f(x_{n+1}, y_{n+1}^{0})]$$

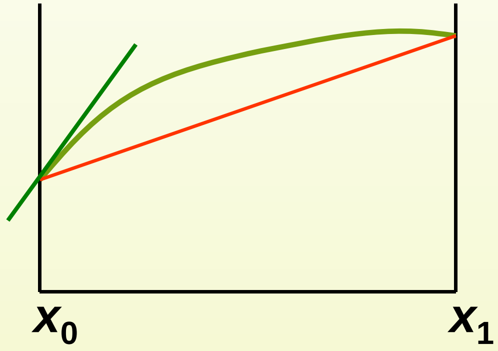
n	x_n=nh=0.5n	y_n	$y(x_n)$精确值
0	0	0	0
1	0.5	0.400000	0.433333
2	1.0	0.635000	0.666667
3	1.5	0.787596	0.807692
4	2.0	0.921025	0.933333

		欧拉	后退欧拉	梯形	改进欧拉	
n	$x_n=nh=0.5n$	y_n	y_n	y_n	y_n	$y(x_n)$精确值
0	0	0	0	0	0	0
1	0.5	0.500000	0.357142	0.416667	0.400000	0.433333
2	1.0	0.800000	0.571428	0.666667	0.635000	0.666667
3	1.5	0.900000	0.733082	0.812500	0.787596	0.807692
4	2.0	0.984615	0.880773	0.937500	0.921025	0.933333

§1 欧拉方法 /* Euler's Method */

➢ 欧拉公式：

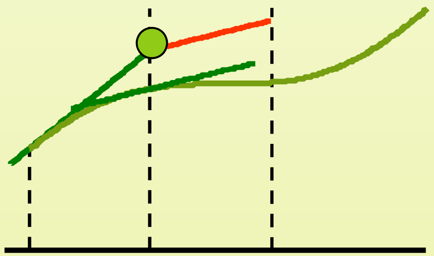
向前差商近似导数 ➡



$$y'(x_0) = \frac{y(x_1) - y(x_0)}{h}$$

$$y(x_1) \approx y(x_0) + hy'(x_0) = y_0 + hf(x_0, y_0)$$

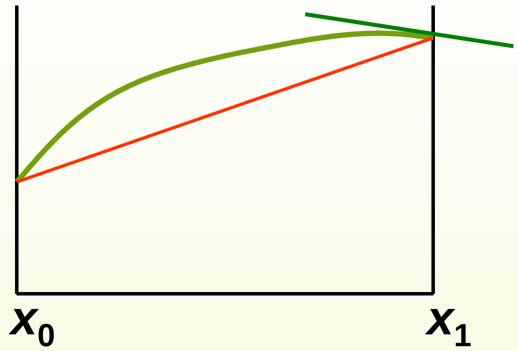
$$\overset{\text{记为}}{=\!=\!=} \; y_1$$



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

- 欧拉公式的改进：

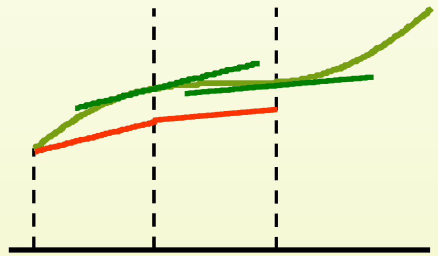
隐式欧拉法 /* implicit Euler method */



向后差商近似导数 → $y'(x_1) = \dfrac{y(x_1) - y(x_0)}{h}$

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

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



一般先用显式计算一个初值，再迭代求解。→ 计算量大！

☞ 隐式欧拉法的局部截断误差：

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

即隐式欧拉公式具有 1 阶精度。

梯形公式 /* trapezoid formula */ — 显、隐式两种算法的平均

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

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

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

注：局部截断误差 $R_i = y(x_{i+1}) - y_{i+1} = O(h^3)$

即梯形公式具有2 阶精度，比欧拉方法有了进步。但注意到该公式是隐式公式，计算时不得不用到迭代法，其迭代收敛性与欧拉公式相似。

改进欧拉法 /* modified Euler's method */

预报公式

Step 1: 先用显式欧拉公式作预测，算出 $\bar{y}_{i+1} = y_i + h\,f(x_i, y_i)$

Step 2: 再将 $\bar{y}_{i+1}$ 代入隐式梯形公式的右边作校正，得到

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

校正公式

注：此法亦称为预测-校正法 /* predictor-corrector method */。可以证明该算法具有 2 阶精度，同时可以看到它是个单步递推格式，比隐式公式的迭代求解过程简单。后面将看到，它的稳定性高于显式欧拉法。

方 法		
显式欧拉	简单	精度低
隐式欧拉	稳定性最好	精度低，计算量大
梯形公式	精度提高	计算量大
中点公式	精度提高，显式	多一个初值， 可能影响精度

第3 节
龙格—库塔法

龙格—库塔法的基本思想（I）

考察差商 $\dfrac{y(x_{n+1})-y(x_n)}{h}$

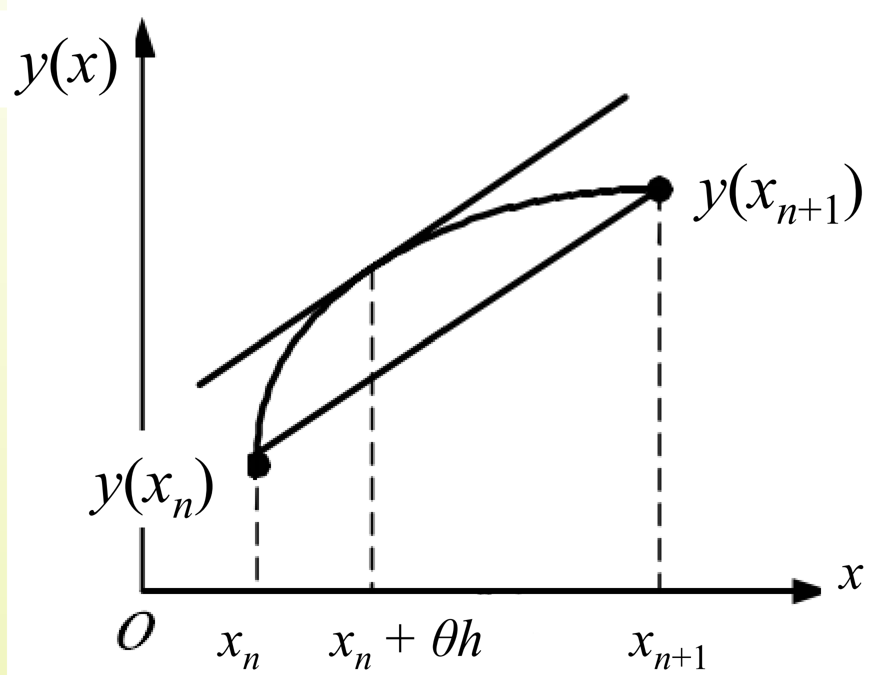
由微分中值定理，可得

$$\frac{y(x_{n+1})-y(x_n)}{h}=y'(x_n+\theta h)$$

$(0<\theta<1)$

因此微分方程

的数值解为 $y'(x)=f(x,y)$



$$y(x_{n+1})=y(x_n)+h\cdot f[(x_n+\theta h),y(x_n+\theta h)]$$

龙格—库塔法的基本思想（2）

这里的$f[(x_n+\theta h),y(x_n+\theta h)]$称为区间（$x_n, x_{n+1}$)上的平均斜率，记作$k^*$,即

$$k^*=f[(x_n+\theta h),y(x_n+\theta h)]$$

因此只要对平均斜率 k^* 提供一种算法，便相应地得到一种微分方程数值计算公式。

用这种观点很容易看出欧拉公式、后退欧拉公式、改进欧拉公式的特点。

$$y(x_{n+1}) = y(x_n) + h \cdot k^*$$

- 欧拉公式

$$k^* \approx f(x_n, y_n) \longrightarrow y_{n+1} = y_n + hf(x_n, y_n)$$

- 后退欧拉公式

$$k^* \approx f(x_{n+1}, y_{n+1}) \longrightarrow y_{n+1} = y_n + hf(x_{n+1}, y_{n+1})$$

- 梯形公式

$$k^* \approx \frac{f(x_n, y_n) + f(x_{n+1}, y_{n+1})}{2}$$

$$\longrightarrow y_{n+1} = y_n + h\frac{f(x_n, y_n) + f(x_{n+1}, y_{n+1})}{2}$$

龙格—库塔法的基本思想（3）

改进欧拉公式利用了(x_n, x_{n+1})两个点的斜率值

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

$$k_2 = f(x_{n+1}, y_n + hk_1)$$

取算术平均作为平均斜率 k^* 的近似值

$$k^* \approx \frac{k_1 + k_2}{2}$$

其中k_2是通过已知信息 y_n 利用欧拉公式预报的。

建立高精度的单步递推格式。

单步递推法的基本思想是从 (x_n, y_n) 点出发，以某一斜率沿直线达到 (x_{n+1}, y_{n+1}) 点。欧拉法及其各种变形所能达到的最高精度为2阶。

考察改进的欧拉法，可以将其改写为：

$$
\begin{cases}
k_1 = f(x_n, y_n) \\
k_2 = f(x_n + h, y_n + hk_1) \\
y_{n+1} = y_n + h\left[\dfrac{k_1}{2} + \dfrac{k_2}{2}\right]
\end{cases}
$$

步长一定是一个 h 吗？

斜率
一定取 k_1 k_2
的平均值吗？

二阶龙格—库塔公式的推导（1）

$$\begin{cases} k_1 = f(x_n, y_n) \\ k_2 = f(x_n + h, y_n + hk_1) \\ y_{n+1} = y_n + h\left[\dfrac{k_1}{2} + \dfrac{k_2}{2}\right] \end{cases}$$

首先推广改进欧拉公式。随意考察区间 (x_n, x_{n+1})内一点

$$x_{n+p} = x_n + ph \quad (0 < p \le 1)$$

我们希望用x_n, x_{n+p}两个点的斜率值 k_1 和 k_2 加权平均得到平均斜率 k^*，即令

$$y_{n+1} = y_n + h[(1-\lambda)k_1 + \lambda k_2]$$

这里的λ为待定常数。 $(0 \le \lambda \le 1)$

$$\begin{aligned} k_1 &= f(x_n, y_n) \\ k_2 &= f(x_n + ph, y_n + phk_1) \\ y_{n+1} &= y_n + h[(1-\lambda)k_1 + \lambda k_2] \end{aligned}$$

二阶龙格－库塔公式的推导（3）

分别将 k_1 和 k_2 Taylor展开，可得

$$k_1 = y'(x_n)$$

$$k_2 = y'(x_n) + phy''(x_n) + O(h^2)$$

$$y_{n+1} = y(x_n) + hy'(x_n) + \lambda ph^2 y''(x_n) + O(h^3)$$

$y(x)$在 $x = x_{n+1}$ 处二阶Taylor展开式为

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

比较系数可以发现，要使得两者具有同样的局部截断误差，需要满足

$$\lambda p = \frac{1}{2}$$

称满足这一条件的所有数值计算格式为二阶龙格－库塔公式

二阶龙格—库塔公式的推导（4）

当取p=1,λ=1/2时，所得的公式即为改进欧拉公式。

$$k_1 = f(x_n, y_n)$$
$$k_2 = f(x_n + ph, y_n + phk_1)$$
$$y_{n+1} = y_n + h[(1-\lambda)k_1 + \lambda k_2]$$

$$k_1 = f(x_n, y_n)$$
$$k_2 = f(x_n + h, y_n + hk_1)$$
$$y_{n+1} = y_n + h[\frac{k_1 + k_2}{2}]$$

二阶龙格—库塔公式的推导（4）

当取p=1/2,λ=1时，所得的公式称为变型的欧拉公式或中点格式

$$k_1 = f(x_n, y_n)$$
$$k_2 = f(x_n + ph, y_n + phk_1)$$
$$y_{n+1} = y_n + h[(1-\lambda)k_1 + \lambda k_2]$$

$$k_1 = f(x_n, y_n)$$
$$k_2 = f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1)$$
$$y_{n+1} = y_n + hk_2$$

含义：用一般欧拉公式预测$[x_n, x_{n+1}]$区间的中点的斜率，并利用此预测值计算 y_{n+1}

欧拉公式

$$y_{n+1} = y_n + hf(x_n, y_n)$$

后退欧拉公式

$$y_{n+1} = y_n + hf(x_{n+1}, y_{n+1})$$

每一步仅计算 $f(x,y)$ 一次，精度为一阶。

梯形公式

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

改进Euler公式可改写成

$$y_{n+1} = y_n + h[\frac{f(x_n, y_n) + f(x_{n+1}, y_n + hk_1)}{2}]$$

每一步计算 $f(x,y)$ 二次，精度为二阶。

上述公式在形式上共同点:

- 都是用$f(x,y)$在某些点上值的线性组合得出$y(x_{n+1})$的近似值y_{n+1}
- 增加计算的次数$f(x,y)$的次数,可提高精度。

可考虑用函数$f(x,y)$在若干点上的函数值的线性组合来构造近似公式。

要求近似公式在(x_n,y_n)处的*Taylor*展开式与解$y(x)$在x_n处的*Taylor*展开式的前面几项重合，从而使近似公式达到所需要的阶数。

或者说，在$[x_n,x_{n+1}]$这一步内多计算几个点的斜率值，然后将其进行加权平均作为平均斜率，则可构造出更高精度的计算格式，这就是**龙格—库塔（*Runge-Kutta*）法的基本思想**。

Runge-Kutta 方法是一种高精度的单步法, 简称*R-K*法

二阶龙格—库塔法

在[x_n,x_{n+1}]区间内取两个点。

$$k_1 = f(x_n, y_n)$$
$$k_2 = f(x_n + ph, y_n + phk_1)$$
$$y_{n+1} = y_n + h[(1-\lambda)k_1 + \lambda k_2]$$

满足如下条件

$$\lambda p = \frac{1}{2}$$

存在无穷多个解。每一步计算 $f(x,y)$ 二次，精度为二阶。

所有满足上式的格式统称为**2阶龙格 － 库塔格式**。

若要获得更高阶得数值方法，就必须增加计算函数值的次数。

三阶龙格—库塔法

在$[x_n,x_{n+1}]$区间内取三个点。

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

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

$$k_3 = f(x_n + h, y_n - hk_1 + 2hk_2)$$

$$y_{n+1} = y_n + h[\frac{1}{6}k_1 + \frac{4}{6}k_2 + \frac{1}{6}k_3]$$

常用的三阶龙格-库塔法。

每一步计算$f(x,y)$三次，精度为三阶。

四阶(经典)龙格—库塔法

在$[x_n,x_{n+1}]$区间内取四个点。

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

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

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

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

$$y_{n+1} = y_n + h[\frac{1}{6}k_1 + \frac{2}{6}k_2 + \frac{2}{6}k_3 + \frac{1}{6}k_4]$$

经典的四阶龙格-库塔法。

每一步计算$f(x,y)$四次，精度为四阶。

注：

☞ 龙格-库塔法的主要运算在于计算 k_i 的值，即计算 f 的值。Butcher 于1965年给出了计算量与可达到的最高精度阶数的关系：

每步须算k_i 的个数	2	3	4	5	6	7	$n \geq 8$
可达到的最高精度	$O(h^2)$	$O(h^3)$	$O(h^4)$	$O(h^4)$	$O(h^5)$	$O(h^6)$	$O(h^{n-2})$

☞ 由于龙格-库塔法的导出基于泰勒展开，故精度主要受解函数的光滑性影响。对于光滑性不太好的解，最好采用低阶算法而将步长h 取小。

例1. 使用三阶和四阶R-K方法计算初值问题

$$\begin{cases} y' = y^2 & 0 \le x \le 0.5 \\ y(0) = 1 & \text{取} h = 0.1. \end{cases}$$

$$k_1 = f(x_n, y_n)$$
$$k_2 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1)$$
$$k_3 = f(x_n + h, y_n - hk_1 + 2hk_2)$$
$$y_{n+1} = y_n + h[\frac{1}{6}k_1 + \frac{4}{6}k_2 + \frac{1}{6}k_3]$$

$$k_1 = f(x_n, y_n)$$
$$k_2 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1)$$
$$k_3 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_2)$$
$$k_4 = f(x_n + h, y_n + hk_3)$$
$$y_{n+1} = y_n + h[\frac{1}{6}k_1 + \frac{2}{6}k_2 + \frac{2}{6}k_3 + \frac{1}{6}k_4]$$

$$k_1 = f(x_n, y_n)$$
$$k_2 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1)$$
$$k_3 = f(x_n + h, y_n - hk_1 + 2hk_2)$$
$$y_{n+1} = y_n + h[\frac{1}{6}k_1 + \frac{4}{6}k_2 + \frac{1}{6}k_3]$$

例1. 使用三阶和四阶R-K方法计算初值问题

$$\begin{cases} y' = y^2 & 0 \le x \le 0.5 \\ y(0) = 1 & \end{cases} \quad 取 h = 0.1.$$

解: (1) 使用三阶R-K方法

$$k_1 = y_0^2 = 1$$
$$k_2 = (y_0 + \frac{1}{2}hk_1)^2 = 1.103$$
$$k_3 = (y_0 - hk_1 + 2hk_2)^2 = 1.256$$
$$y_1 = y_0 + \frac{h}{6}(k_1 + 4k_2 + k_3) = 1.111$$

其余结果如下：

i	x_i	k_1	k_2	k_3	y_i
1.000	0.100	1.000	1.103	1.256	1.111
2.000	0.200	1.235	1.376	1.595	1.250
3.000	0.300	1.562	1.764	2.092	1.428
4.000	0.400	2.040	2.342	2.866	1.666
5.000	0.500	2.777	3.259	4.163	1.999

(2) 如果使用四阶*R-K*方法

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

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

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

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

$$y_{n+1} = y_n + h[\frac{1}{6}k_1 + \frac{2}{6}k_2 + \frac{2}{6}k_3 + \frac{1}{6}k_4]$$

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

$$k_1 = y_0^2 = 1$$

$$k_2 = (y_0 + \frac{1}{2}hk_1)^2 = 1.103$$

$$k_3 = (y_0 + \frac{1}{2}hk_2)^2 = 1.113$$

$$k_4 = (y_0 + hk_3)^2 = 1.235$$

$$y_1 = y_0 + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) = 1.1111$$

其余结果如下:

i	x_i	k_1	k_2	k_3	k_4	y_i
1.000	0.100	1.000	1.103	1.113	1.235	1.111
2.000	0.200	1.235	1.376	1.392	1.563	1.250
3.000	0.300	1.562	1.764	1.791	2.042	1.429
4.000	0.400	2.040	2.342	2.389	2.781	1.667
5.000	0.500	2.777	3.259	3.348	4.006	2.000

四阶龙格－库塔公式算例

试用四阶龙格－库塔公式计算下列初值问题。

$$y'(x) = 1 - \frac{2xy}{1+x^2}, y(0) = 0, 0 \le x \le 2$$

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

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

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

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

$$y_{n+1} = y_n + h[\frac{1}{6}k_1 + \frac{2}{6}k_2 + \frac{2}{6}k_3 + \frac{1}{6}k_4]$$

取步长h＝1，并与精确值

$$y(x) = \frac{x(3+x^2)}{3(1+x^2)}$$

比较。

解：由四阶龙格一库塔公式得：

n	$x_n=nh$	y_n	$y(x_n)$精确值
0	0	0	0
1	1.0	0.666667	0.666667
2	2.0	0.933333	0.933333

第5 节

多步法

基本思路（I）

多步法是指 y_{n+1} 近似地用$y_n, y_{n-1}\ldots$和$f(x_{n+1}, y_{n+1}), f(x_n, y_n), f(x_{n-1}, y_{n-1})\ldots$来表达，而不像单步法只用到前一步长的数据。

多步法的通式为

$$
\begin{aligned}
y_{n+1} &= a_0 y_n + a_1 y_{n-1} + \ldots + a_p y_{n-p} \\
&\quad + h[b_{-1} f(x_{n+1}, y_{n+1}) + b_0 f(x_n, y_n) + b_1 f(x_{n-1}, y_{n-1}) + \ldots + b_p f(x_{n-p}, y_{n-p})] \\
&= \sum_{i=0}^{p} a_i y_{n-i} + h \sum_{i=-1}^{p} b_i f(x_{n-i}, y_{n-i})
\end{aligned}
$$

在x_n点作泰勒展开

$$y_{n+1}=\left(\sum_{i=0}^{p}a_i\right)y_n+\sum_{j=1}^{m}\left[\frac{h^j}{j!}\left(\sum_{i=0}^{p}a_i(-i)^j+j\sum_{i=-1}^{p}b_i(-i)^{j-1}\right)y_n{}^{(j)}\right]+\ldots$$

$$y(x_{n+1})=y(x_n)+hy'(x_n)+\frac{h^2}{2!}y''(x_n)+\ldots+\frac{h^m}{m!}y^m(x_n)+\ldots$$

比较两式，对应系数相等的话，则有

$$\sum_{i=0}^{p}a_i=1$$

$$\sum_{i=0}^{p}a_i(-i)^j+j\sum_{i=-1}^{p}b_i(-i)^{j-1}=1\quad j=1,2,\ldots,m$$

第6 节
Adam法

基本思路（I）

上面曾讨论过多步法的通用表达式为：

$$y_{n+1} = \sum_{i=0}^{p} a_i y_{n-i} + h \sum_{i=-1}^{p} b_i f(x_{n-i}, y_{n-i})$$

且满足

$$\sum_{i=0}^{p} a_i = 1$$

$$\sum_{i=0}^{p} a_i (-i)^j + j \sum_{i=-1}^{p} b_i (-i)^{j-1} = 1 \quad j = 1, 2, \ldots, m \qquad (9)$$

$$\sum_{i=0}^{p} a_i(-i)^j + j\sum_{i=-1}^{p} b_i(-i)^{j-1} = 1 \quad j = 1, 2, \ldots, m$$

显式Adam方法（I）

Adam方法是令系数a_1=a_2=…=a_p=0，得a_0=1，则多步法的公式变为：

$$y_{n+1} = y_n + h\sum_{i=-1}^{p} b_i f(x_{n-i}, y_{n-i})$$

可选择显式法还是隐式法。

显式Adam方法，又称“Adam's-Bashforth”法，是令b_{-1}＝0，由式（9）得

$$\sum_{i=0}^{p}(-i)^{(j-1)} b_i = \frac{1}{j}(j = 1, 2, \ldots, m) \qquad (10)$$

$p+1$个未知数,m个方程，所以m应大于等于$p+1$

显式Adam方法（2）

$$\sum_{i=0}^{p}(-i)^{(j-1)}b_i=\frac{1}{j}(j=1,2,\ldots,m)$$

式（10）也可写成矩阵形式：

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 0 & -1 & -2 & \cdots & -p \\ 0 & 1 & 4 & \cdots & (-p)^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & (-1)^p & (-2)^p & \cdots & (-p)^p \end{pmatrix}\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ b_p \end{pmatrix}=\begin{pmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \\ \vdots \\ \frac{1}{p+1} \end{pmatrix}$$

选择所希望的阶数，即可通过式（11）计算系数b_i。

$$\sum_{i=0}^{p}(-i)^{(j-1)}b_i = \frac{1}{j}(j=1,2,\ldots,m) \tag{10}$$

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 0 & -1 & -2 & \cdots & -p \\ 0 & 1 & 4 & \cdots & (-p)^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & (-1)^p & (-2)^p & \cdots & (-p)^p \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ b_p \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \\ \vdots \\ \frac{1}{p+1} \end{pmatrix}$$

$$= \begin{pmatrix} (0)^0 & (-1)^0 & (-2)^0 & \cdots & (-p)^0 \\ (0)^1 & (-1)^1 & (-2)^1 & \cdots & (-p)^1 \\ (0)^2 & (-1)^2 & (-2)^2 & \cdots & (-p)^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (0)^p & (-1)^p & (-2)^p & \cdots & (-p)^p \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ b_p \end{pmatrix} \tag{11}$$

显式Adam方法一例子（I）

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 0 & -1 & -2 & \cdots & -p \\ 0 & 1 & 4 & \cdots & (-p)^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & (-1)^p & (-2)^p & \cdots & (-p)^p \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ b_p \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \\ \vdots \\ \frac{1}{p+1} \end{pmatrix}$$

试导出三阶Adam's-Bashforth法

解：令p=2，有

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 1 & 4 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \end{pmatrix}$$

得到

$$b_0 = \frac{23}{12}, b_1 = -\frac{16}{12}, b_2 = \frac{5}{12}$$

从而，得到三阶Adam’s-Bashforth法为：

$$y_{n+1} = y_n + \frac{1}{12} h[23 f(x_n, y_n) - 16 f(x_{n-1}, y_{n-1}) + 5 f(x_{n-2}, y_{n-2})]$$

$$\sum_{i=0}^{p} a_i(-i)^j + j\sum_{i=-1}^{p} b_i(-i)^{j-1} = 1 \quad j = 1, 2, \ldots, m$$

隐式Adam方法（I）

当令$b_{-1}\neq 0$，Adam方法称为隐式Adam方法，又称"Adam's-moulton"法。此时：

$$y_{n+1} = y_n + h\sum_{i=-1}^{p} b_i f(x_{n-i}, y_{n-i})$$

由式（9）得：

$$\sum_{i=-1}^{p} (-i)^{(j-1)} b_i = \frac{1}{j}(j = 1, 2, \ldots, m)$$

$p+2$个未知数,m个方程，所以m应大于等于$p+2$

隐式Adam方法（2）

$$\sum_{i=-1}^{p}(-i)^{(j-1)}b_i = \frac{1}{j}(j=1,2,\ldots,m)$$

写成矩阵形式为：

$$\begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & 0 & -1 & -2 & \cdots & -p \\ 1 & 0 & 1 & 4 & \cdots & (-p)^2 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & (-1)^{p+1} & (-2)^{p+1} & \cdots & (-p)^{p+1} \end{pmatrix}\begin{pmatrix} b_{-1} \\ b_0 \\ b_1 \\ \vdots \\ b_p \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \\ \vdots \\ \frac{1}{p+2} \end{pmatrix} \quad (14)$$

$$\sum_{i=-1}^{p}(-i)^{(j-1)}b_i=\frac{1}{j}(j=1,2,\ldots,m)$$

$$\begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & 0 & -1 & -2 & \cdots & -p \\ 1 & 0 & 1 & 4 & \cdots & (-p)^2 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & (-1)^{p+1} & (-2)^{p+1} & \cdots & (-p)^{p+1} \end{pmatrix} \begin{pmatrix} b_{-1} \\ b_0 \\ b_1 \\ \vdots \\ b_p \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \\ \vdots \\ \frac{1}{p+2} \end{pmatrix}$$

$$= \begin{pmatrix} (1)^0 & (0)^0 & (-1)^0 & (-2)^0 & \cdots & (-p)^0 \\ (1)^1 & (0)^1 & (-1)^1 & (-2)^1 & \cdots & (-p)^1 \\ (1)^2 & (0)^2 & (-1)^2 & (-2)^2 & \cdots & (-p)^2 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ (1)^{p+1} & (0)^{p+1} & (-1)^{p+1} & (-2)^{p+1} & \cdots & (-p)^{p+1} \end{pmatrix} \begin{pmatrix} b_{-1} \\ b_0 \\ b_1 \\ \vdots \\ b_p \end{pmatrix}$$

隐式Adam方法—例子（I）

试推导三阶Adam's-moulton法

解：令p=1，有

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & -1 \\ 1 & 0 & 1 \end{pmatrix}\begin{pmatrix} b_{-1} \\ b_0 \\ b_1 \end{pmatrix}=\begin{pmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \end{pmatrix}$$

计算得到

$$b_{-1}=\frac{5}{12}, b_0=\frac{8}{12}, b_1=-\frac{1}{12}$$

隐式Adam方法一例子（2）

从而，三阶Adam's-moulton法为：

$$x_{n+1}=x_n+\frac{1}{12}h[5f(x_{n+1},t_{n+1})+8f(x_n,t_n)-f(x_{n-1},t_{n-1})]$$

注意在算法的执行过程中需要采用迭代解法，因为是隐式的。

避免迭代的一般性做法是采用预测一校正方法，即用显式Adam做预测，用隐式Adam方法做校正。

Adam法：显示Vs 隐式

$$y_{n+1} = \sum_{i=0}^{p} a_i y_{n-i} + h \sum_{i=-1}^{p} b_i f(x_{n-i}, y_{n-i})$$

$a_0=1$，$a_1=a_2=\ldots=a_p=0$ 显示和隐式

显示

$$\begin{pmatrix} (0)^0 & (-1)^0 & (-2)^0 & \cdots & (-p)^0 \\ (0)^1 & (-1)^1 & (-2)^1 & \cdots & (-p)^1 \\ (0)^2 & (-1)^2 & (-2)^2 & \cdots & (-p)^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (0)^p & (-1)^p & (-2)^p & \cdots & (-p)^p \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ b_p \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \\ \vdots \\ \frac{1}{p+1} \end{pmatrix}$$

隐式

$$\begin{pmatrix} (1)^0 & (0)^0 & (-1)^0 & (-2)^0 & \cdots & (-p)^0 \\ (1)^1 & (0)^1 & (-1)^1 & (-2)^1 & \cdots & (-p)^1 \\ (1)^2 & (0)^2 & (-1)^2 & (-2)^2 & \cdots & (-p)^2 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ (1)^{p+1} & (0)^{p+1} & (-1)^{p+1} & (-2)^{p+1} & \cdots & (-p)^{p+1} \end{pmatrix} \begin{pmatrix} b_{-1} \\ b_0 \\ b_1 \\ \vdots \\ b_p \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \\ \vdots \\ \frac{1}{p+2} \end{pmatrix}$$

第7节

Gear法

基本思路（1）

Gear法：除b_{-1}外所有b_i都为零。

因为$b_{-1}\neq 0$，所以所有的Gear法都为隐式法。

k阶Gear公式的推导为：

$$y_{n+1}=a_0y_n+a_1y_{n-1}+\ldots+a_{k-1}y_{n-k+1}+hb_{-1}f(x_{n+1},y_{n+1}) \qquad (15)$$

令p=k-1和$b_0=b_1=\ldots=0$，得k阶的Gear法：

基本思路（2）

与Adam法类似，式（15）中的k+1个系数可由下式得出：

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 & 0 \\ 0 & -1 & -2 & \cdots & -(k-1) & 1 \\ 0 & 1 & 4 & \cdots & [-(k-1)]^2 & 2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & (-1)^k & (-2)^k & \cdots & [-(k-1)]^k & k \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ b_{-1} \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \quad (16)$$

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 & 0 \\ 0 & -1 & -2 & \cdots & -(k-1) & 1 \\ 0 & 1 & 4 & \cdots & [-(k-1)]^2 & 2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & (-1)^k & (-2)^k & \cdots & [-(k-1)]^k & k \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ b_{-1} \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 & 0 \\ (0)^1 & (-1)^1 & (-2)^1 & \cdots & [-(k-1)]^1 & 1 \\ (0)^2 & (-1)^2 & (-2)^2 & \cdots & [-(k-1)]^2 & 2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ (0)^k & (-1)^k & (-2)^k & \cdots & [-(k-1)]^k & k \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ b_{-1} \end{pmatrix}$$

Gear法—例子（1）

试推导三阶的Gear法。

解：令k=3，由式（16）得：

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & -1 & -2 & 1 \\ 0 & 1 & 4 & 2 \\ 0 & -1 & -8 & 3 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ b_{-1} \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

计算得到

$$b_{-1} = \frac{6}{11}, a_0 = \frac{18}{11}, a_1 = -\frac{9}{11}, a_2 = \frac{2}{11}$$

Gear法—例子（2）

从而求得三阶Gear法为：

$$y_{n+1} = \frac{18}{11} y_n - \frac{9}{11} y_{n-1} + \frac{2}{11} y_{n-2} + \frac{6}{11} hf(x_{n+1}, y_{n+1})$$

注意到在算法的执行过程中，内存中要保留y_n，y_{n-1}，y_{n-2}的数据，并且需要采用迭代解法，因为是隐式的。

第8 节
数值稳定性

问题的提出

由上节的讨论可知，步长大小的选择会直接影响到局部截断误差，本节则要介绍步长大小的选择对数值计算方法稳定性的影响。

数值计算方法稳定性：任一步产生的误差在以后均能逐步衰减，则称这种方法是稳定的。

对于稳定的数值计算方法，积分步长的选择将只决定于局部截断误差。

数值稳定性分析的方法一测试方程法

为了分析步长大小的选择对数值计算方法稳定性的影响，引入下面简单的测试方程：

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

容易得出其解为 $$y(x) = y_0 e^{(\lambda x)}$$

当λ<0时，y(x)随着时间的推移而趋向于零；当λ>0时，y(x)随着时间的推移而趋向无穷大。

下面将讨论应用欧拉法、后退欧拉法、多步法数值积分计算时的稳定区域。

$$y'(x) = f(y) = \lambda y(x)$$

欧拉法的数值稳定性分析（I）

对测试方程应用欧拉法：

有 $$y_{n+1} = y_n + h\lambda y_n = (1+h\lambda)y_n$$

$$y_1 = (1+h\lambda)y_0$$

$$y_2 = (1+h\lambda)y_1 = (1+h\lambda)^2 y_0$$

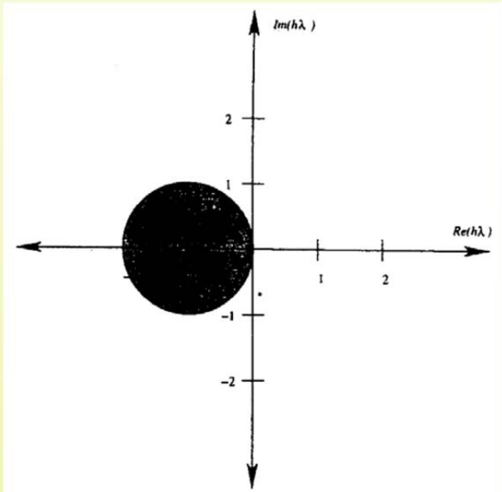
$$\vdots$$

$$y_n = (1+h\lambda)^n y_0$$

可以看出只有当|1+hλ|<1时，才能使当λ<0时，y(x)随着时间的推移而趋向于零。

欧拉法的数值稳定性分析（2）

所以对于λ<0，系统稳定的条件是hλ落在以（－1，0）为圆心的单位圆内，如图所示。



可以得出λ值越大，步长就必须选择得越小。

$$y'(x) = f(y) = \lambda y(x)$$

后退欧拉法的数值稳定性分析（I）

相似地，对测试方程使用后退欧拉法：

有 $$y_{n+1} = y_n + h\lambda y_{n+1} = \frac{y_n}{(1-h\lambda)}$$

$$y_1 = \frac{y_0}{(1-h\lambda)}$$

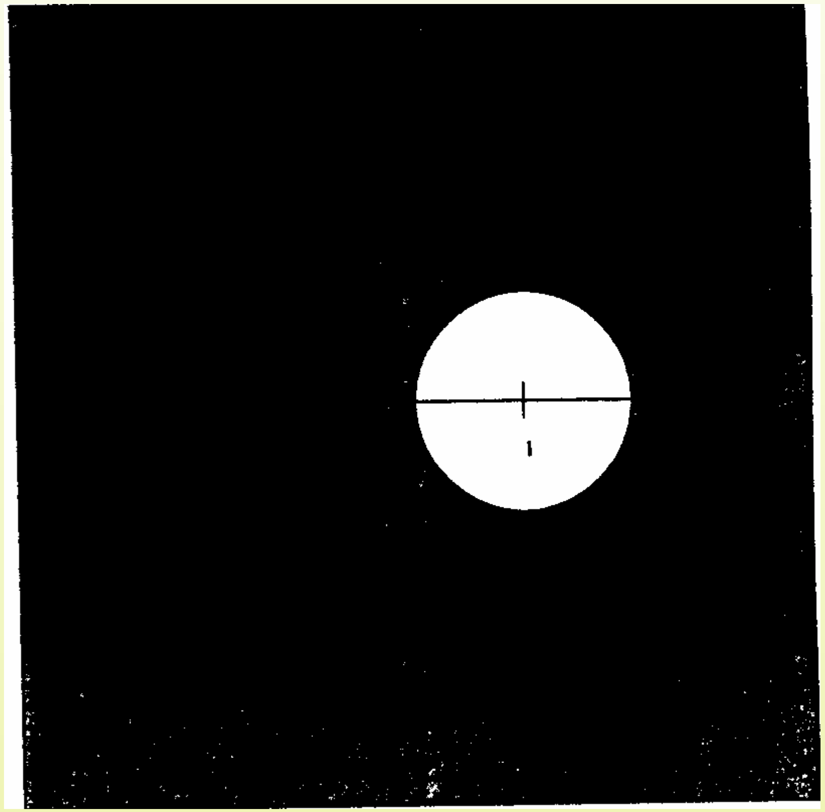
$$y_2 = \frac{y_1}{(1-h\lambda)} = \frac{y_0}{(1-h\lambda)^2}$$

$$\vdots$$

$$y_n = \frac{y_0}{(1-h\lambda)^n}$$

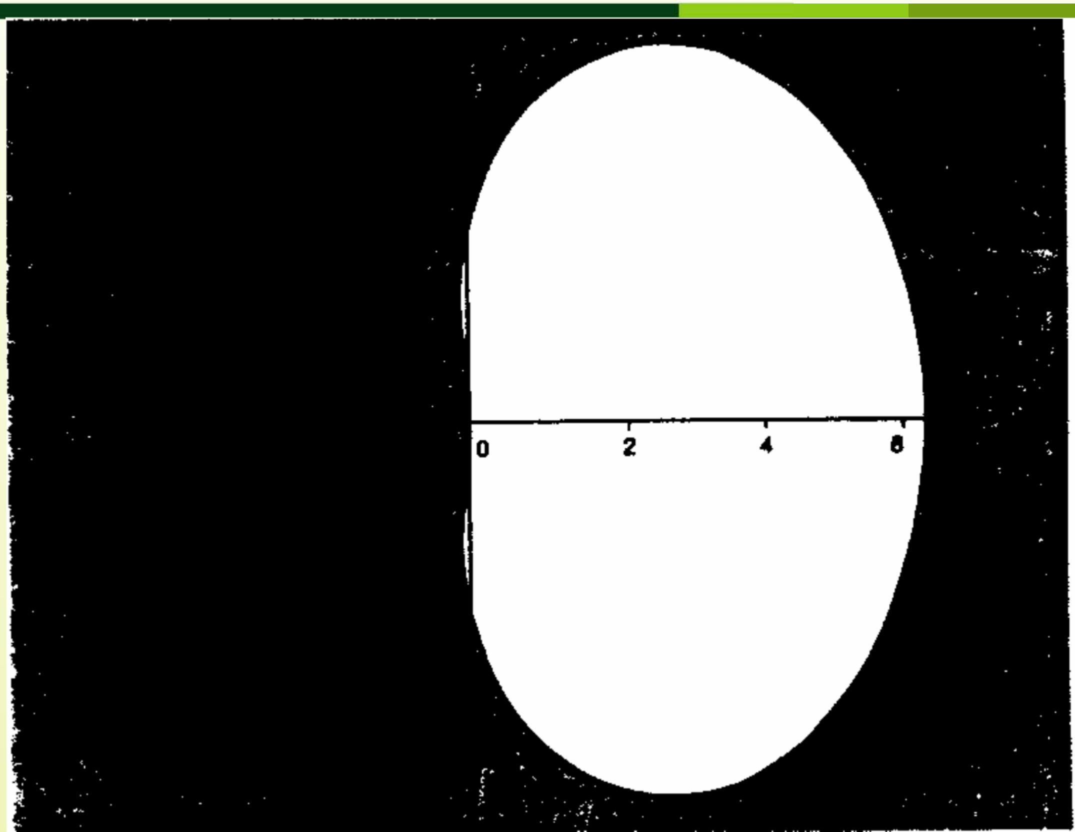
后退欧拉法的数值稳定性分析（2）

因为当$\lambda<0$时，$y(x)$随着时间的推移而趋向于零，所以必须使$|1-h\lambda|>1$。即对于$\lambda<0$，系统稳定的条件是$h\lambda$落在以（1，0）为圆心的单位圆外，如图所示。



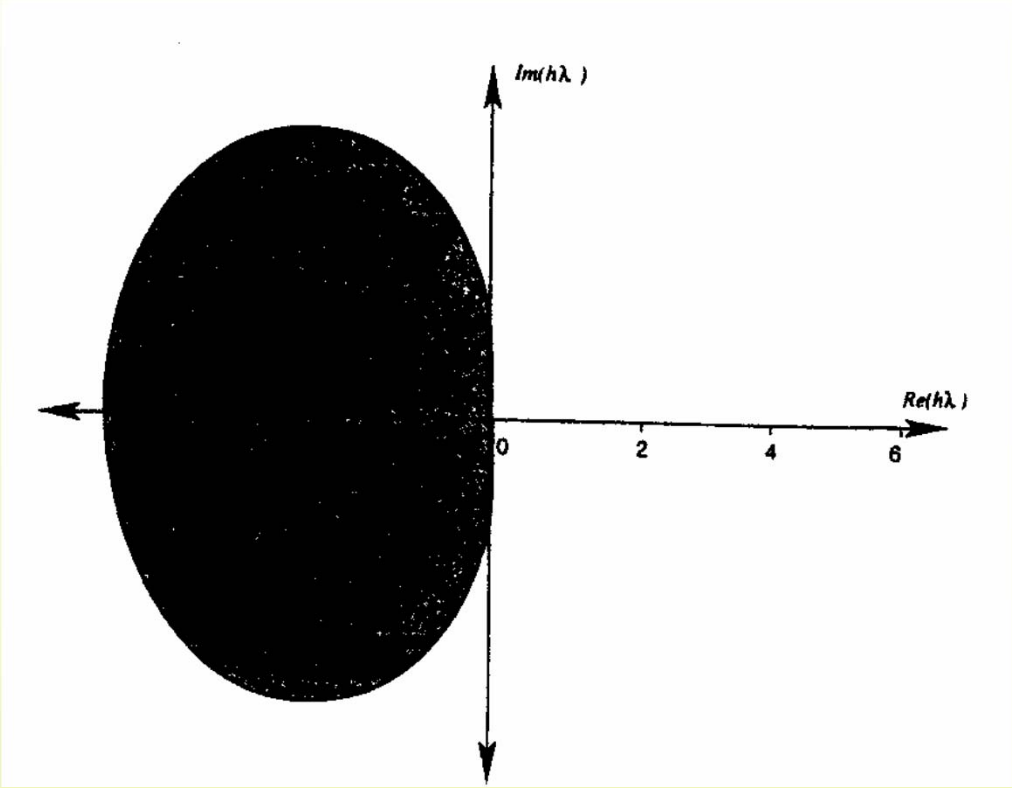
可以看出使用后退欧拉法时积分步长可以取得任意大，而不至于影响到解的稳定性，这时步长的选择只取决于局部截断误差。

多步法的数值稳定性—例子



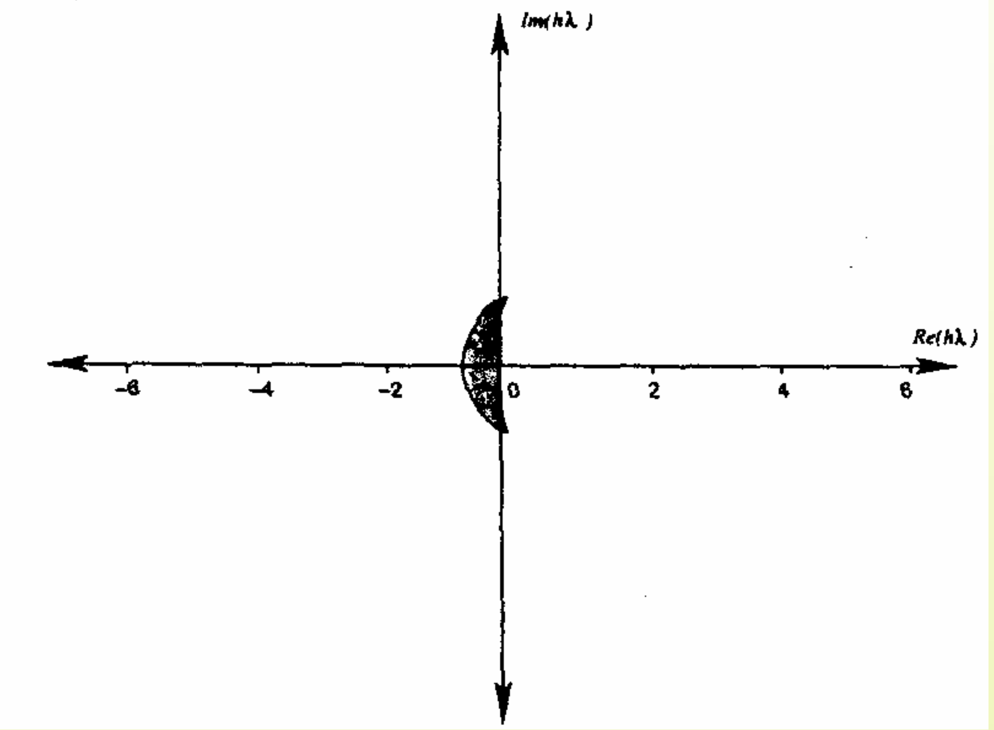
三阶的Gear法的绝对稳定域

Adam's-moulton法的稳定性分析



三阶的Adam's-moulton法的绝对稳定域

Adam's-Bashforth法的稳定性分析



三阶的Adam's-Bashforth法的绝对稳定域

多步法的数值稳定性—总结

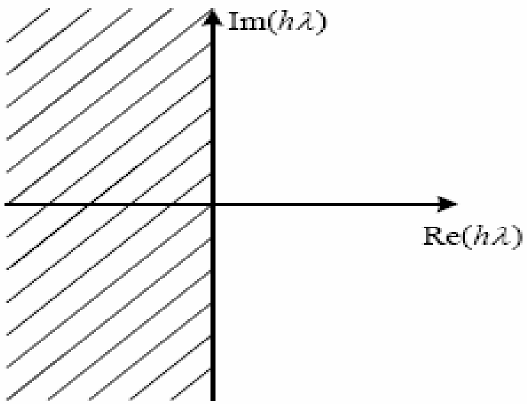
本例用图像表明了隐式法和显式法的主要差别，隐式法（Gear法和Adam's-moulton法）的绝对稳定区域要比显式法（Adam's-Bashforth法）大得多。如Gear法的绝对稳定区域几乎包括整个hλ的左半平面，所以积分步长可以取得任意大，而不至于影响到解的稳定性。正因为这个原因，各种商业软件中往往会采用隐式法，其积分步长的选择只取决于局部截断误差。

另外，随着阶数的增加，绝对稳定区域会不断缩小，但精确度却会增加。

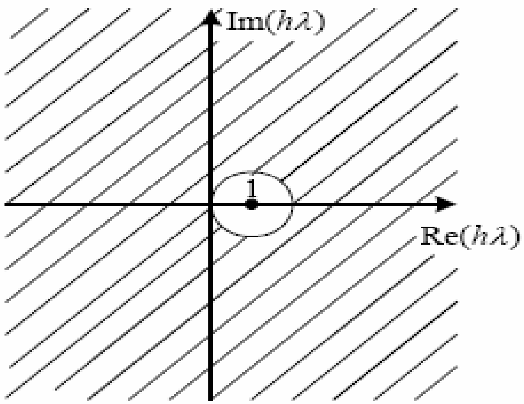
电气工程中常用数值积分公式的特性

梯形公式和 Gear 公式的形式及其局部截断误差

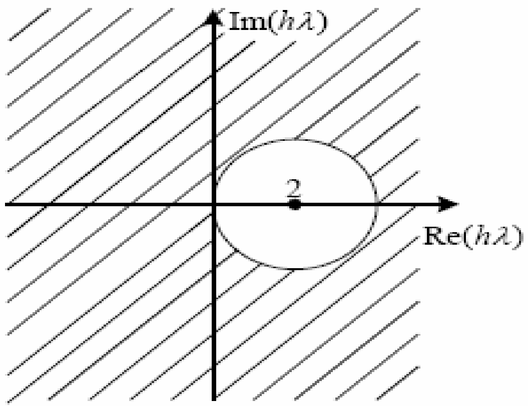
数值积分公式	形式	局部截断误差
梯形公式	$X_{n+1} = X_n + \frac{h}{2}(X'_{n+1} + X'_n)$	$O(h^3)$
一阶 Gear 公式	$X_{n+1} = X_n + hX'_{n+1}$	$O(h^2)$
二阶 Gear 公式	$X_{n+1} = \frac{4}{3}X_n - \frac{1}{3}X_{n-1} + \frac{2}{3}hX'_{n+1}$	$O(h^3)$



(a) 梯形公式稳定域



(b) 一阶 Gear 公式稳定域



(c) 二阶 Gear 公式稳定域