

# 第五章 常微分方程数值解

/\* Numerical Methods for Ordinary Differential Equations \*/

考虑一阶常微分方程的初值问题 /\* Initial-Value Problem \*/:

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

例如: 
$$\begin{cases} y' = x + y, & x \in [0, 1] \\ y(0) = 1 \end{cases}$$

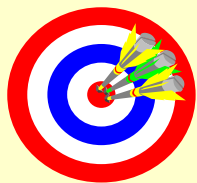
其解析解为  $y = -x - 1 + 2e^x$

但是,只有一些特殊类型的微分方程问题能够得到用解析表达式表示的函数解,而大量的微分方程问题很难得到其解析解,因此,只能依赖于数值方法去获得微分方程的数值解。

例如: 
$$\begin{cases} y' = e^{-x^2}, & x \in [0, 1] \\ y(0) = 1 \end{cases}$$

其解析解为:  $y = 1 + \int_0^x e^{-t^2} dt, x \in [0, 1]$  很难得到其解析解。

本章的任务：计算出解函数  $y(x)$  在一系列节点  $a = x_0 < x_1 < \dots < x_n = b$   $y_n \approx y(x_n)$  ( $n = 1, \dots, N$ ) 处的近似值。



## 建立常微分方程数值方法的基本思想

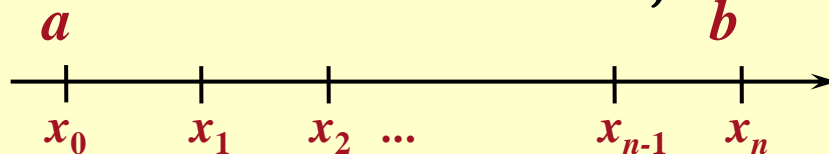
微分方程数值解法，其实是求出方程的解  $y(x)$  在一系列离散点上的近似值。则微分方程数值解的基本思想是：求解区间和方程离散化。

### ► 求解区间离散化

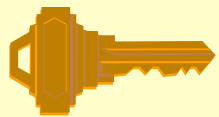
将求解区间  $[a, b]$  离散化，是在  $[a, b]$  上插入一系列的分点  $\{x_k\}$  使  $a = x_0 < x_1 < \dots < x_n < x_{n+1} < \dots < x_N = b$

记  $h_n = x_{n+1} - x_n$  ( $n = 0, 1, \dots, N-1$ ) 称为步长，一般取  $h_n = h$  (常数)，节点为  $x_n = x_0 + nh$  ( $n = 0, 1, 2, \dots, N$ )  $h = \frac{b-a}{N}$

称为等步长节点。



## ➤ 将微分方程离散化



将微分方程离散化，通常有下述方法：

### (1) 差商逼近法



即是用适当的差商逼近导数值。

### (2) 数值积分法



基本思想是先将问题转化为积分方程

$$y(x_m) - y(x_n) = \int_{x_n}^{x_m} f(x, y(x)) dx \quad (y(x_0) = y_0)$$

然后将上式右端采用第四章介绍的数值积分离散化，从而获得原初值问题的一个离散差分格式。

### (3) Taylor展开法



见后面的叙述。

# § 1 欧拉方法

➤ 欧拉方法  
向前差商

亦称为欧拉折线法

/\* Euler's polygonal arc method \*/

$$y(x_1) \approx y(x_0) + h y'(x_0) = y_0 + h f(x_0, y_0) \quad \text{记为}$$

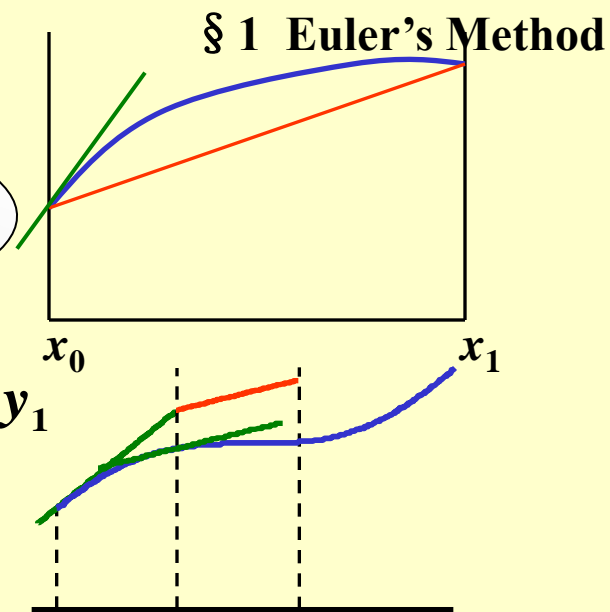
$$y_{n+1} = y_n + h f(x_n, y_n) \quad (n = 0, \dots, N-1)$$

例. 求初值问题

$$\begin{cases} y' = y - \frac{2x}{y} & (0 < x < 1) \\ y(0) = 1 \end{cases}$$

解: 本题的Euler公式的具体形式为

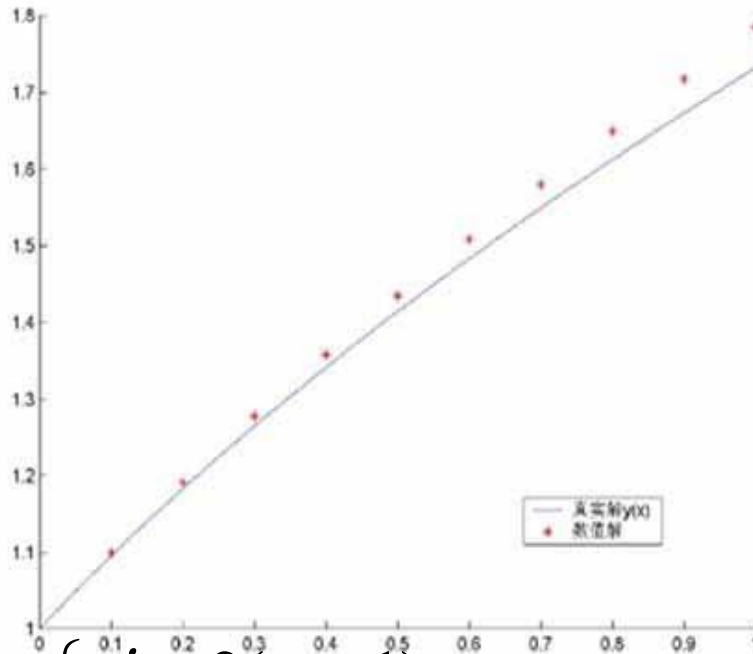
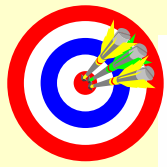
$$y_{n+1} = y_n + h \left( y_n - \frac{2x_n}{y_n} \right)$$



取步长 $h=0.1$ 。我们将计算结果与其解析解的精确值一同列在下表中，其中  $x_n$  是节点， $y_n$  是节点上的近似值， $y(x_n)$  是精确值，结果见下表：

$x_n$	$y_n$	$y(x_n)$	$x_n$	$y_n$	$y(x_n)$
0.1	1.1000	1.0954	0.6	1.5090	1.4832
0.2	1.1918	1.1832	0.7	1.5803	1.5492
0.3	1.2774	1.2649	0.8	1.6498	1.6125
0.4	1.3582	1.2416	0.9	1.7178	1.6733
0.5	1.4351	1.4142	1.0	1.7848	1.7321

如果说表格仍不够直观的话，我们用Matlab做出积分曲线与近似值的图，如下：



在图上似乎数值解与曲线的偏差不是很大，但不要忘记这只是在0到1范围内的。通过后面用其他方法解本题，大家便会发现Euler方法误差其实是很大的。

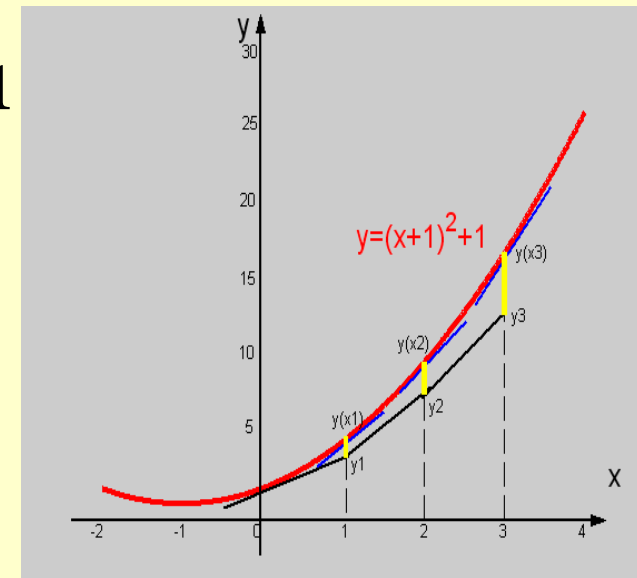
例：  $\begin{cases} y' = 2(x+1) \\ y(0) = 2 \end{cases}$  其精确解为  $y = (x+1)^2 + 1$

由  $y(0)=2$ ，过该曲线上一点  $(0,2)$  作曲线的切线，其斜率： $\left. \frac{dy}{dx} \right|_{x=0} = f(0,2)$

切线为： $y - 2 = f(0,2)(x - 0)$

由此，可计算出  $y_1$ ， $y_1 = 2 + f(0,2) * 1 = 4$

类似地，可计算出  $y_2, \dots$  故欧拉法又称欧拉折线法。

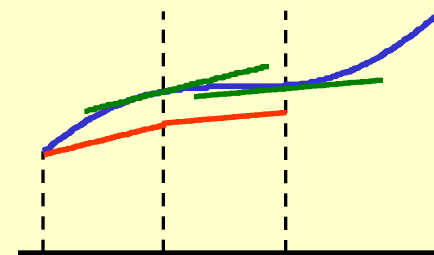
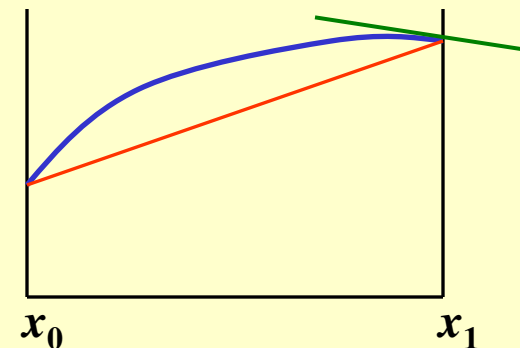


✂ 隐式欧拉法 /\* implicit Euler method \*/

向后差商近似导数  $\rightarrow y'(x_1) \approx \frac{y(x_1) - y(x_0)}{h}$

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

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



由于未知数  $y_{n+1}$  同时出现在等式的两边，

不能直接得到，故称为隐式 /\* implicit \*/ 欧拉公式，而前者称为显式 /\* explicit \*/ 欧拉公式。

一般先用显式计算一个初值，再迭代求解。即

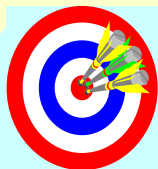
$$y_{n+1}^{(0)} = y_n + h f(x_n, y_n)$$

$$y_{n+1}^{(k+1)} = y_n + h f(x_{n+1}, y_{n+1}^{(k)}) \quad k = 0, 1, 2, \dots$$

如果迭代过程收敛，则某步后  $y_{n+1}^{(k)}$  就可以作为  $y_{n+1}$ ，从而进行下一步的计算。

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

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

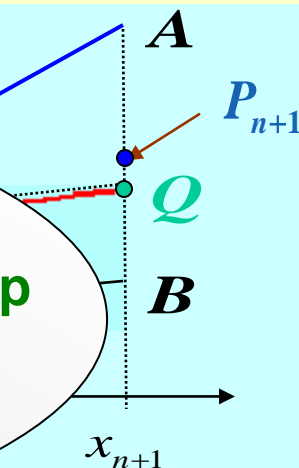


梯形方法的平均

可以借助

Euler

需要2个初值  $y_0$  和  $y_1$  来启动递推过程，这样的算法称为**两步法** **/\* double-step method \*/**，而前面的三种算法都是**单步法** **/\* single-step method \*/**。



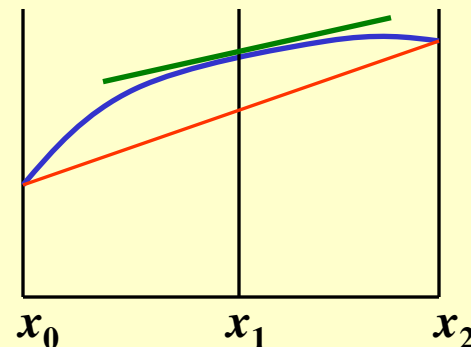
✍ 两步欧拉公式

中心差商近似

$$y'(x_1) \approx \frac{y(x_2) - y(x_0)}{2h}$$

$$\rightarrow y(x_2) = y(x_0) + 2h f(x_1, y(x_1))$$

$$y_{n+1} = y_{n-1} + 2h f(x_n, y_n) \quad n = 1, \dots, N-1$$





## 改进欧拉法 /\* modified Euler's method \*/

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

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

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

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

$$y_p = y_n + h f(x_n, y_n), y_c = y_n + h f(x_{n+1}, y_p)$$

$$y_{n+1} = \frac{1}{2}(y_p + y_c)$$

**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。

**例** 作为比较，我们仍用Euler法中的那个例子。

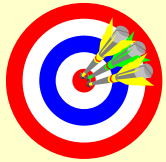
$$\begin{cases} y' = y - \frac{2x}{y} & (0 < x < 1) \\ y(0) = 1 \end{cases}$$

**解：** 改进的Euler公式为

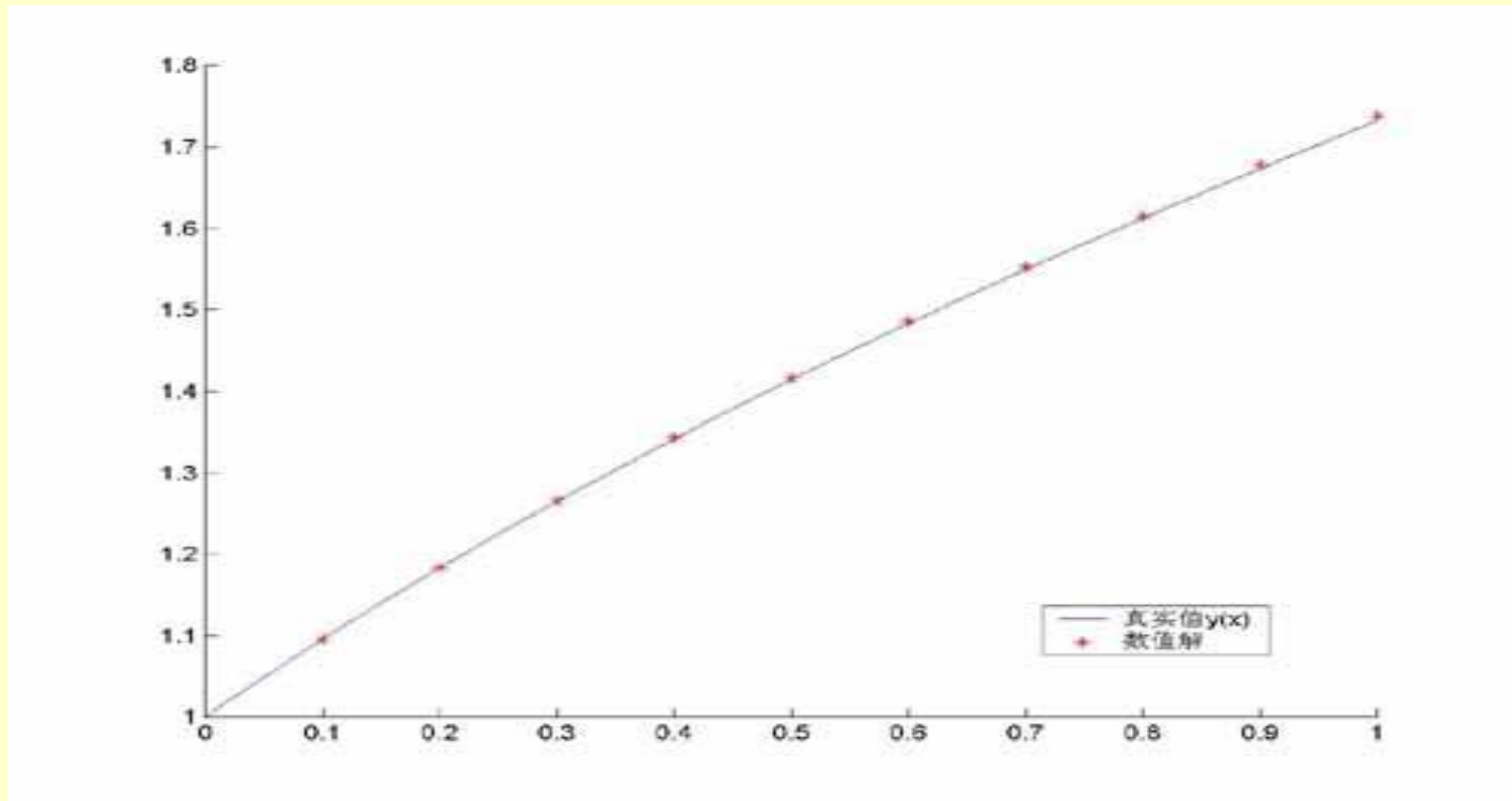
$$\begin{cases} y_p = y_n + h(y_n - \frac{2x_n}{y_n}) \\ y_c = y_n + h(y_p - \frac{2x_{n+1}}{y_p}) \\ y_{n+1} = \frac{1}{2}(y_p + y_c) \end{cases}$$

我们仍取步长 $h=0.1$ ，计算结果见右表：

$x_n$	$y_n$	$y(x_n)$
0.1	1.0959	1.0954
0.2	1.1841	1.1832
0.3	1.2662	1.2649
0.4	1.3434	1.3416
0.5	1.4164	1.4142
0.6	1.4860	1.4832
0.7	1.5525	1.5492
0.8	1.6153	1.6125
0.9	1.6782	1.6733
1.0	1.7379	1.7321



## Matlab作图显示



从表和图可以看出，改进的Euler法的精度提高了不少。

### *Euler*公式的*m*函数程序:

```
function y=Euler(f,a,b,n,ya)
    h=(b-a)/n;y(1)=ya;
    for i=1:n
        x=a+(i-1)*h;
        y(i+1)=y(i)+h*feval(f,x,y(i));
    End
```

### 改进欧拉法的*m*函数程序:

```
function z=Modified_Euler(f,a,b,n,ya)
    z(1)=ya; h=(b-a)/n;
    for i=1:n
        x=a+(i-1)*h;
        y(i+1)=z(i) + h*feval(f,x,z(i));
        z(i+1)=z(i)+1/2*h*(feval(f,x,z(i))+feval(f,x+h,y(i+1)));
    end
```

**算例3：** 分别用Euler公式和改进的Euler公式求解：

$$(1) \quad \begin{cases} y' = x^2 + x - y, 0 \leq x \leq 1 \\ y(0) = 0 \end{cases} \quad \text{精确解: } y = x^2 - x + 1 - e^{-x}$$

取步长  $h=0.1$ ，计算 $y(0.5)$ 的近似值

**解：** 欧拉公式： 
$$\begin{cases} y_{i+1} = y_i + 0.1(x_i^2 + x_i - y_i) , i = 0, 1, \dots, 9 \\ y_0 = 0 \end{cases}$$

改进的Euler公式：

$$\begin{cases} \bar{y}_{i+1} = y_i + 0.1(x_i^2 + x_i - y_i) \\ y_{i+1} = y_i + \frac{0.1}{2}(x_i^2 + x_i - y_i + x_{i+1}^2 + x_{i+1} - \bar{y}_{i+1}) \\ y_0 = 0 \end{cases} \quad i = 0, 1, 2, \dots, 9$$

## ➤ 局部截断误差和方法的阶

初值问题的单步法可用一般形式表示为：

$\varphi$  称为增量函数

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

$\varphi$  含有  $y_{n+1}$  时，方法是隐式的，  
 $\varphi$  不含有  $y_{n+1}$  时，方法是显式的。

例如对欧拉法有  $\varphi(x_n, y_n, h) = f(x_n, y_n)$ ，隐式欧拉法有  $\varphi(x_n, y_n, y_{n+1}, h) = f(x_{n+1}, y_{n+1})$

从  $x_0$  开始计算，如果考虑每一步产生的误差，直到  $x_n$  则有误差  $e_n = y(x_n) - y_n$ ，称为该方法在  $x_n$  的整体截断误差，分析和求得整体截断误差是复杂的。为此，我们仅考虑从  $x_n$  到  $x_{n+1}$  的局部情况，并假设  $x_n$  之前的计算没有误差，即  $y_n = y(x_n)$ ，下面在给出单步法的局部截断误差概念前，先给出Taylor展式的概念。

## ► Taylor级数

函数 $y(x)$ 在展开点 $x_0$ 作Taylor级数展开, $x$ 是被展开点。

$$y(x) = y(x_0) + y'(x_0)(x - x_0) + \frac{1}{2!} y''(x_0)(x - x_0)^2 + \cdots + \frac{1}{n!} y^{(n)}(x_0)(x - x_0)^n + \cdots$$

则:

$$y(x_{i+1}) = y(x_i + h) = y(x_i) + hy'(x_i) + \frac{h^2}{2!} y''(x_i) + \cdots + \frac{h^n}{n!} y^{(n)}(x_i) + \cdots$$

$$y(x_{i-1}) = y(x_i - h) = y(x_i) - hy'(x_i) + \frac{h^2}{2!} y''(x_i) + \cdots + \frac{(-h)^n}{n!} y^{(n)}(x_i) + \cdots$$

$$y'(x_{i+1}) = y'(x_i + h) = y'(x_i) + hy''(x_i) + \frac{h^2}{2!} y'''(x_i) + \cdots + \frac{h^n}{n!} y^{(n+1)}(x_i) + \cdots$$

$$y'(x_{i-1}) = y'(x_i - h) = y'(x_i) - hy''(x_i) + \frac{h^2}{2!} y'''(x_i) + \cdots + \frac{(-h)^n}{n!} y^{(n+1)}(x_i) + \cdots$$

注意到,一般地,对以上各式,随着 $n$ 的增大,累加式中后面的各项将越来越小;如果 $h$ 取得比较小,譬如 $h=0.1$ 或 $h=0.01$ 或 $h=0.001$ 等等,则随着 $n$ 的增大 $h^n$ 也将越来越小。

## 求局部截断误差的步骤

- 1) 作局部化假设, 即设  $y_n = y(x_n)$ ;
- 2) 将差分  $y_{n+1}$  在  $x_n$  处展开;
- 3) 将准确解  $y(x_{n+1})$  在  $x_n$  处展开;
- 4) 比较  $T_{n+1} = y(x_{n+1}) - y_{n+1}$

h)

确的前提下,

断误差 /\* local

**定义** 若某算法的局部截断误差为  $O(h^{p+1})$ , 则称该算法有  $p$

阶精度。

 $T_{n+1}$  的主项

/\* leading term \*/

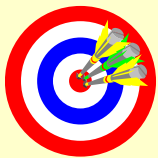
☞ 欧拉法的局部截断误差:

$$\begin{aligned}
 T_{n+1} &= y(x_{n+1}) - y_{n+1} = [\cancel{y(x_n)} + \cancel{hy'(x_n)} + \frac{h^2}{2} y''(x_n) + O(h^3)] - [\cancel{y_n} + \cancel{hf(x_n, y_n)}] \\
 &= \frac{h^2}{2} y''(x_n) + O(h^3) \quad \text{欧拉法具有 1 阶精度。}
 \end{aligned}$$

☞ 两步欧拉法的局部截断误差:

$$T_{n+1} = y(x_{n+1}) - y_{n+1} = O(h^3) \quad \text{即两步欧拉公式具有 2 阶精度。}$$





以上定义对隐式单步法也是适用的。

例 求隐式欧拉格式  $T_{n+1}$  的主项  $y_{n+1}$  的局部截断误差。

$$\begin{aligned}
 T_{n+1} &= y_{n+1} - y(x_{n+1}) \\
 &= h y'(x_n) + \frac{h^2}{2} y''(x_n) + O(h^3) - h [y'(x_n) + h y''(x_n) + O(h^2)] \\
 &= -\frac{h^2}{2} y''(x_n) + O(h^3) \text{ 具有 1 阶精度。}
 \end{aligned}$$

*T<sub>n+1</sub> 的主项*  
/\* leading term \*/

例 求梯形格式  $T_{n+1}$  的主项  $y_{n+1}$  的局部截断误差。

Hey! Isn't the *leading term* of the *local truncation error* of Euler's method  $\frac{h^2}{2} y''(x_i)$ ?  
Seems that we can make a good use of it ...

$$\begin{aligned}
 T_{n+1} &= h y'(x_n) + \frac{h^2}{2} y''(x_n) + \frac{h^3}{6} y'''(x_n) + O(h^4) - \frac{h}{2} [y'(x_n) + h y''(x_n) + \frac{h^2}{2} y'''(x_n)] + O(h^4) \\
 &= -\frac{h^3}{12} y'''(x_n) + O(h^4) \text{ 具有 2 阶精度。}
 \end{aligned}$$

*T<sub>n+1</sub> 的主项*  
/\* leading term \*/



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
两步欧拉公式	精度提高, 显式	多一个初值, 可能影响精度

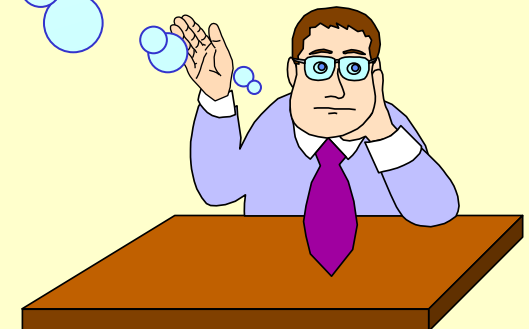
C:  
with

OK, let's  
make it  
possible.

edy...  
any



HW:  
p.152-153 #2, # 3



## 附：人物介绍——欧拉



欧 拉

Leonhard Euler

(1707~1783)

瑞士数学家、自然科学家

- 十八世纪数学界最杰出的人物之一。
- 数学史上最多产的数学家。
- 不但为数学界作出贡献，  
而且把数学推至几乎整个物理领域。

## 附：人物介绍——欧拉

- 欧拉是科学史上最多产的一位杰出的数学家。

以每年平均 800 页的速度写出创造性论文。

一生共写下了 886 本书籍和论文。

其中 分析、代数、数论占40%，几何占18%，

物理和力学占28%，天文学占11%，

弹道学、航海学、建筑学等占

3%，

- 彼得堡科学院为了整理他的著作，足足忙碌了 47 年

整理出他的研究成果多达 74 卷。

(牛顿全集 8 卷，高斯全集 12 卷)

## 附：人物介绍——欧拉

- 欧拉编写了大量的力学、分析学、几何学的教科书。《无穷小分析引论》、《微分学原理》以及《积分学原理》都成为数学中的经典著作。
- 课本上常见的如  $i$ ,  $e$ ,  $\sin$ ,  $\cos$ ,  $\text{tg}$ ,  $\Delta x$ ,  $\Sigma$ ,  $f(x)$  等等，也都是他创立并推广的。
- 有的学者认为，自从 1784 年以后，微积分的教科书基本上都抄袭欧拉的书。

## 附：人物介绍——欧拉

● 如今几乎每一个数学领域都可以看到欧拉的名字：

初等几何的欧拉线

多面体的欧拉定理

解析几何的欧拉变换

四次方程的欧拉解法

数论中的欧拉函数

微分方程的欧拉方程

复变函数的欧拉公式

变分学的欧拉方程

级数论的欧拉常数

.....

## 附：人物介绍——欧拉

### ● 欧拉的记忆力惊人！

能背诵罗马诗人维吉尔(Virgil)的史诗Aeneid,

能背诵前一百个质数的前十次幂,

能背诵“全部”的数学公式,

直至晚年，还能复述年轻时的笔记的“全部”内容。

## 附：人物介绍——欧拉

● 欧拉的心算能力罕见！

道听途说 欧拉的两个学生把一个复杂的收敛级数的前 17 项加起来，算到第 50 位数字，两人相差一个单位；

欧拉为了确定究竟谁对，用心算进行了全部运算，最后把错误找了出来。



## 附：人物介绍——欧拉

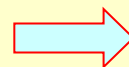
### ● 欧拉的毅力极其顽强！

可以在任何不良的环境中工作。

常常抱着孩子在膝上完成论文。

在双目失明以后，也没有停止对数学的研究。

在失明后的 17 年间，还口述了400 篇左右的论文。



(返回)



## § 2 龙格 - 库塔法 /\* Runge-Kutta Method \*/



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

$$\begin{cases} y_{n+1} \\ K_1 \\ K_2 \end{cases} = \begin{cases} \frac{y(x_{n+1}) - y(x_n)}{h} = y'(\xi), \xi \in (x_n, x_{n+1}) \\ y(x_{n+1}) = y(x_n) + hf(\xi, y(\xi)) \end{cases}$$

斜率

一定取  $K_1, K_2$  的平均值吗?

显然， $k_1, k_2$  是在点  $x_n$  的是两个点的斜率的加权平均，新的途径。Runge-Kutta方法就是这种思想的体现和发展。

步长一定是一个  $h$  吗?

将改进欧拉法推广为:

$$\begin{cases} y_{n+1} = y_n + h[\lambda_1 K_1 + \lambda_2 K_2] \\ K_1 = f(x_n, y_n) \\ K_2 = f(x_n + ph, y_n + phK_1) \end{cases} \quad \lambda_1 + \lambda_2 = 1$$

首先希望能确定系数  $\lambda_1$ 、 $\lambda_2$  精度, 即在  $y_n = y(x_n)$  的前

$$T_{n+1} = y(x_{n+1}) - y_n$$

**Step 1:** 将  $K_2$  在  $(x_n, y_n)$  点作 Taylor

$$\begin{aligned} K_2 &= f(x_n + ph, y_n + phK_1) \\ &= f(x_n, y_n) + phf_x(x_n, y_n) + phK_1f_y(x_n, y_n) + O(h^2) \\ &= y'(x_n) + phy''(x_n) + O(h^2) \end{aligned}$$

**Step 2:** 将  $K_2$  代入第1式, 得到

$$\begin{aligned} y_{n+1} &= y_n + h \left\{ \lambda_1 y'(x_n) + \lambda_2 [y'(x_n) + phy''(x_n) + O(h^2)] \right\} \\ &= y_n + (\lambda_1 + \lambda_2)hy'(x_n) + \lambda_2 ph^2 y''(x_n) + O(h^3) \end{aligned}$$

$$y''(x) = \frac{d}{dx} f(x, y)$$

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

$$= f_x(x, y) + f_y(x, y)f(x, y)$$

**Step 3:** 将  $y_{n+1}$  与  $y(x_{n+1})$  在  $x_n$  点的泰勒展开作比较

$$y_{n+1} = y_n + (\lambda_1 + \lambda_2)hy'(x_n) + \lambda_2 ph^2 y''(x_n) + O(h^3)$$

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

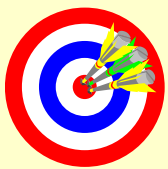
要求  $T_{n+1} = y(x_{n+1}) - y_{n+1} = O(h^3)$  则必须有:

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

这里有 3 个未知数, 2 个方程。

存在无穷多个解。所有满足上式的格式统称为2阶龙格 - 库塔格式。注意到,  $p=1, \lambda_1 = \lambda_2 = \frac{1}{2}$  就是改进的欧拉法。

**Q:** 为获得更高的精度, 应该如何进一步推广?



## Runge-Kutta方法的一般形式

$$y_{n+1} = y_n + h \sum_{i=1}^L \lambda_i k_i$$

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

$$k_2 = f(x_n + c_2 h, y_n + c_2 h k_1)$$

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

.....

$$k_i = f(x_n + c_i h, y_n + c_i h \sum_{j=1}^{i-1} a_{ij} k_j) \quad i = 2, 3, \dots, L$$

其中  $\sum_{i=1}^L \lambda_i = 1$  ,  $c_i \leq 1$  ,  $\sum_{j=1}^{i-1} a_{ij} = 1$  均为待定系数, 确定这些系数的步骤与前面相似。

- 最常用为四级4阶经典龙格-库塔法 /\* Classical Runge-Kutta Method \*/ :

$$\begin{cases} y_{n+1} &= y_n + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ K_1 &= f(x_n, y_n) \\ K_2 &= f(x_n + \frac{h}{2}, y_n + \frac{h}{2}K_1) \\ K_3 &= f(x_n + \frac{h}{2}, y_n + \frac{h}{2}K_2) \\ K_4 &= f(x_n + h, y_n + hK_3) \end{cases}$$

我们仍用前面的例子来看看四阶Runge-Kutta方法的效果。

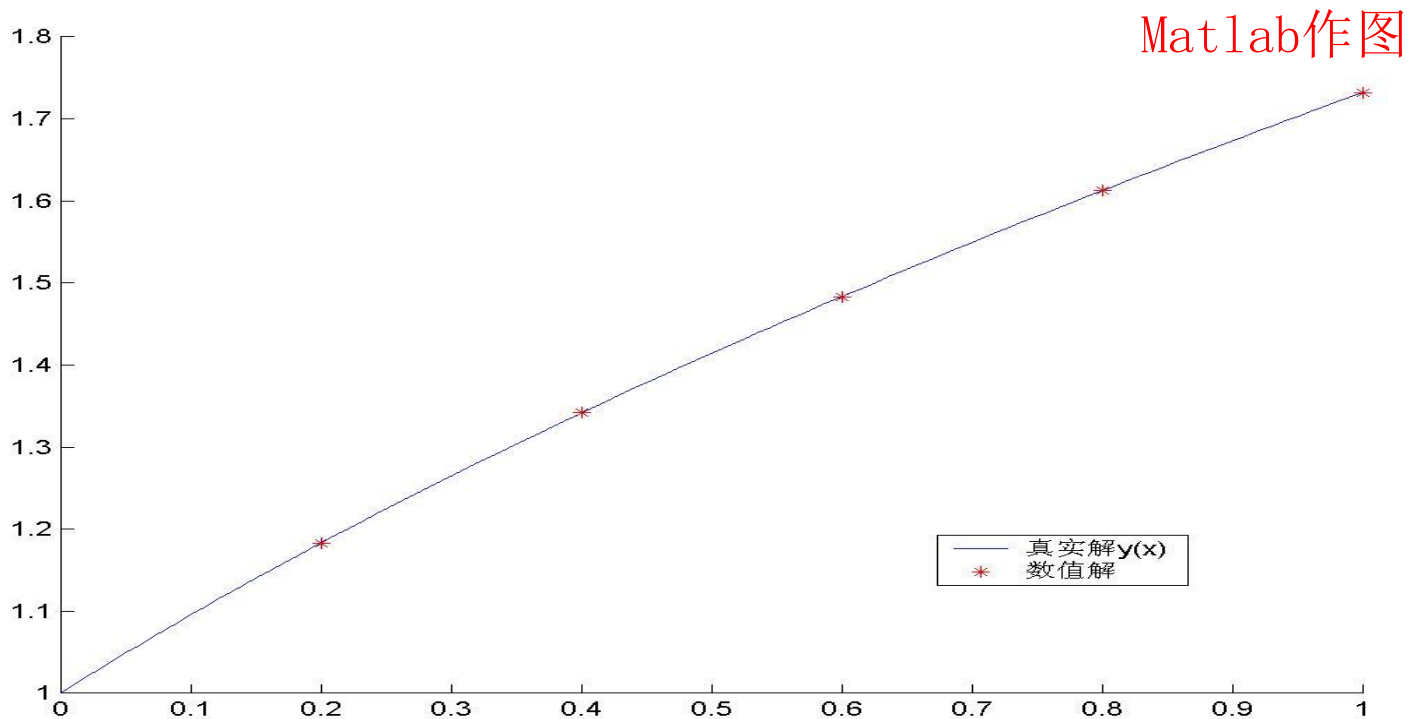
$$\begin{cases} y' = y - \frac{2x}{y} & (0 < x < 1) \\ y(0) = 1 \end{cases}$$

解：对于本题，经典的四阶Runge-Kutta方法具有以下形式：

$$\left\{ \begin{array}{l} y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\ k_1 = y_n - \frac{2x_n}{y_n} \\ k_2 = y_n + \frac{h}{2}k_1 - \frac{2x_n + h}{y_n + \frac{h}{2}k_1} \\ k_3 = y_n + \frac{h}{2}k_2 - \frac{2x_n + h}{y_n + \frac{h}{2}k_2} \\ k_4 = y_n + hk_3 - \frac{2(x_n + h)}{y_n + hk_3} \end{array} \right.$$

这里，我们取步长  
h=0.2，下面是计算结果（符号的意义同前）

$x_n$	$y_n$	$y(x_n)$
0.2	1.1832	1.1832
0.4	1.3417	1.3416
0.6	1.4833	1.4832
0.8	1.6125	1.6125
1.0	1.7321	1.7321



从上图可以看出，每一个数值解都“准确”的落在了真实解的曲线上。与改进的Euler法所做出来的图比较，似乎精确性没有很明显的提高，但是不要忘了在用Runge-Kutta方法时，我们取的步长是 $h=0.2$ 。实际上，Runge-Kutta方法的精确性是要高很多。



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。

HW:

p.152-153 #2, # 3, # 4


## § 3 收敛性与稳定性 /\* Convergency and Stability \*/

### ➤ 收敛性 /\* Convergency \*/

**定义** 若某算法对于任意固定的  $x = x_n = x_0 + n h$ , 当  $h \rightarrow 0$  (同时  $n \rightarrow \infty$ ) 时有  $y_n \rightarrow y(x_n)$ , 则称该算法是收敛的。

**例:** 就初值问题  $\begin{cases} y' = \lambda y \\ y(0) = y_0 \end{cases}$  考察欧拉显式格式的收敛性。

**解:** 该问题的精确解为  $y(x) = y_0 e^{\lambda x}$

欧拉公式为  $y_{n+1} = y_n + h \lambda y_n = (1 + \lambda h) y_n$    $\lim_{h \rightarrow 0} (1 + \lambda h)^{1/\lambda h} = e$

对任意固定的  $x = x_n = n h$ , 有

$$y_n = y_0 (1 + \lambda h)^{x_n/h} = y_0 [(1 + \lambda h)^{1/\lambda h}]^{\lambda x_n}$$

$$\rightarrow y_0 e^{\lambda x_n} = y(x_n) \quad \checkmark$$

# 回顾

考虑一阶常微分方程的初值问题 /\* Initial-Value Problem \*/:

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

只要  $f(x, y)$  在  $[a, b] \times \mathbb{R}^1$  上连续, 且关于  $y$  满足 **Lipschitz 条件**, 即存在与  $x, y$  无关的常数  $L$  使  $|f(x, y_1) - f(x, y_2)| \leq L|y_1 - y_2|$  对任意定义在  $[a, b]$  上的  $y_1(x)$  和  $y_2(x)$  都成立, 则上述 IVP 存在唯一解。

初值问题的单步法可用一般形式表示为:

$\varphi$  称为增量函数

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

例如对欧拉法有  $\varphi(x_n, y_n, h) = f(x_n, y_n)$ , 隐式欧拉法有  $\varphi(x_n, y_n, y_{n+1}, h) = f(x_{n+1}, y_{n+1})$

**定理** 对于一个 $p$ 阶的显式单步法，若满足如下条件

(1) 增量函数 $\varphi$ 关于 $y$ 满足Lipschitz条件，即存在常数 $L_\varphi > 0$ ，使

$$|\varphi(x, y, h) - \varphi(x, \bar{y}, h)| \leq L_\varphi |y - \bar{y}|, \quad \forall y, \bar{y} \in R$$

成立；

(2) 微分方程的初值是精确的。

则该方法收敛，其整体截断误差为：

$$|e_n| = |y(x_n) - y_n| = O(h^p)。$$

注：1、判断显式单步格式的收敛性，归结为验证增量函数 $\varphi$ 是否满足Lipschitz条件。

2、单步格式的整体截断误差由初值误差及局部截断误差决定，整体截断误差比局部截断误差的阶数低一阶。

3、要构造高精度的计算方法，只需设法提高局部截断误差的阶即可。

更详细的请参看教材p137

## ► 稳定性 /\* Stability \*/

例：考察初值问题  $\begin{cases} y'(x) = -30y(x) \\ y(0) = 1 \end{cases}$  在区间  $[0, 0.5]$  上的解。

分别用欧拉显、隐式格式和改进的欧拉格式计算数值解。

节点

An Engineer complains: "Math theorems are so unstable that a small perturbation on the conditions will cause a crash on the conclusions!"



0.5

0.4

0.5

$1.6000 \times 10^1$	$3.9063 \times 10^1$	$3.9063 \times 10^1$	$6.1442 \times 10^{-6}$
$-3.2000 \times 10^1$	$9.7656 \times 10^{-4}$	$9.7656 \times 10^1$	$3.0590 \times 10^{-7}$

What is wrong ???

**定义** 若某算法在计算过程中任一步产生的误差在以后的计算中都**逐步衰减**，则称该算法是**绝对稳定的** /\*absolutely stable \*/。

常数，可以是复数

一般分析时为简单起见，只考虑**试验方程** /\* test equation \*/

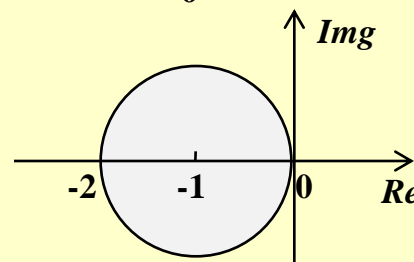
$$y' = \lambda y \quad \text{Re}(\lambda) < 0$$

当步长取为  $h$  时，将某算法应用于上式，并假设只在初值产生误差  $\varepsilon_0 = y_0 - \bar{y}_0$ ，则若此误差以后逐步衰减，就称该算法相对于  $\bar{h} = \lambda h$  **绝对稳定**， $\bar{h}$  的全体构成**绝对稳定区域**。我们称**算法A比算法B稳定**，就是指 A 的绝对稳定区域比 B 的大。

例：考察显式欧拉法  $y_{i+1} = y_i + h\lambda y_i = (1+\bar{h})^{i+1} y_0$

$$\varepsilon_i = y_i - \bar{y}_i \quad \rightarrow \quad \bar{y}_{i+1} = (1+\bar{h})\bar{y}_i$$

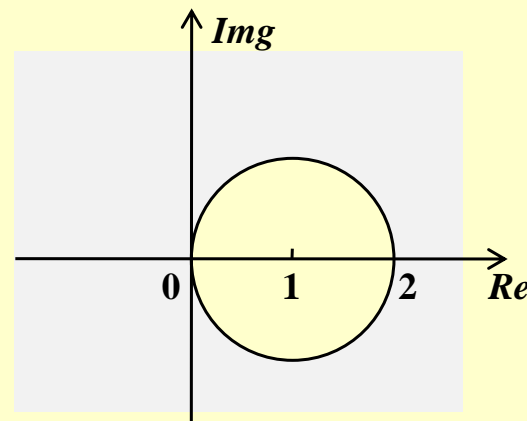
$$\rightarrow \quad \varepsilon_{i+1} = y_{i+1} - \bar{y}_{i+1} = (1+\bar{h})\varepsilon_i$$



由此可见，要保证初始误差 $\varepsilon_0$ 以后逐步衰减， $\bar{h} = \lambda h$ 必须满足： $|1+\bar{h}| < 1$

例：考察隐式欧拉法  $y_{i+1} = y_i + h\lambda y_{i+1}$

$$y_{i+1} = \left( \frac{1}{1-\bar{h}} \right) y_i \quad \rightarrow \quad \varepsilon_{i+1} = \left( \frac{1}{1-\bar{h}} \right)^{i+1} \varepsilon_0$$



可见绝对稳定区域为： $|1-\bar{h}| > 1$

注：一般来说，隐式欧拉法的绝对稳定性比同阶的显式法的好。

例：考察梯形公式  $y_{i+1} = y_i + \frac{h}{2}(\lambda y_i + \lambda y_{i+1})$

$$y_{i+1} = \left( \frac{1 + \bar{h}/2}{1 - \bar{h}/2} \right) y_i \rightarrow \varepsilon_{i+1} = \left( \frac{1 + \bar{h}/2}{1 - \bar{h}/2} \right) \varepsilon_i$$

当  $\lambda < 0$  时，总有  $\left| \frac{1 + \lambda h/2}{1 - \lambda h/2} \right| < 1$

可见梯形公式是无条件稳定的。

思考：设有常微分方程初值问题  $\begin{cases} y' = f(x, y) \\ y(x_0) = y_0 \end{cases}$  的单步法

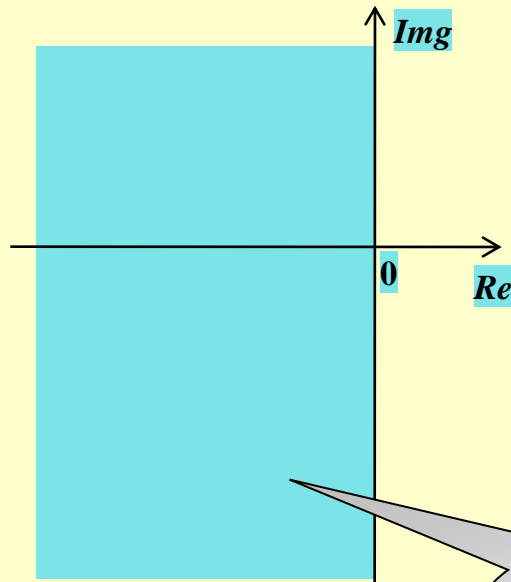
$$y_{n+1} = y_n + \frac{h}{3}[f(x_n, y_n) + 2f(x_{n+1}, y_{n+1})]$$

证明该方法是无条件稳定的。

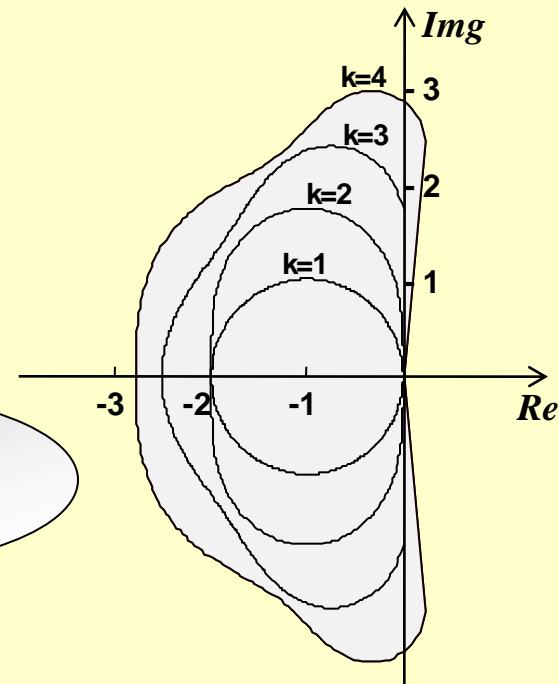


例：隐式龙格-库塔法 
$$\begin{cases} y_{i+1} = y_i + h[\lambda_1 K_1 + \dots + \lambda_m K_m] \\ K_j = f(x_i + \alpha_j h, y_i + \beta_{j1} h K_1 + \dots + \beta_{jm} h K_m) \\ (j = 1, \dots, m) \end{cases}$$

其中2阶方法 
$$\begin{cases} y_{i+1} = y_i + h K_1 \\ K_1 = f(x_i + \frac{h}{2}, y_i + \frac{h}{2} K_1) \end{cases}$$
 的绝对稳定区域为



而显式 1~4 阶方法的绝对稳定区域为



无条件稳定

HW:  
p.153 #6,8

## 改进欧拉法 /\* modified Euler's method \*/

**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

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

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

## 预估-改进-校正-改进系统

**Step 1:** 先用欧拉二步法作预测，算出  $\bar{y}_{i+1} = y_{i-1} + 2hf(x_i, y_i)$

欧拉二步法的局部截断误差：

$$y(x_{i+1}) - \bar{y}_{i+1} = \frac{h^3}{3} y'''(x_i) + \dots \approx \frac{h^3}{3} y'''(x_i)$$

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

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

梯形公式的局部截断误差:

$$y(x_{i+1}) - y_{i+1} = -\frac{h^3}{12} y'''(x_i) + \cdots \approx -\frac{h^3}{12} y'''(x_i)$$

二式相除, 有:

$$\frac{y(x_{i+1}) - y_{i+1}}{y(x_{i+1}) - \bar{y}_{i+1}} \approx -\frac{1}{4}$$

由此式可推导出下列2个事后误差估计式:

$$\begin{aligned} y(x_{i+1}) - \bar{y}_{i+1} &\approx -\frac{4}{5} (\bar{y}_{i+1} - y_{i+1}) \\ y(x_{i+1}) - y_{i+1} &\approx \frac{1}{5} (\bar{y}_{i+1} - y_{i+1}) \end{aligned}$$

把事后误差加在估计值的后面，可得如下计算方案：

预测：

$$p_{i+1} = y_{i-1} + 2hf(x_i, y_i)$$

改进：

$$M_{i+1} = P_{i+1} - \frac{4}{5}(p_{i+1} - C_{i+1}) \approx P_{i+1} - \frac{4}{5}(P_i - C_i)$$

校正：

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

改进：

$$y_{i+1} = C_{i+1} + \frac{1}{5}(p_{i+1} - C_{i+1})$$

开始时取  $P_0 - C_0 = 0$ ，即第一步计算时不做改进。

```

function m_y=Modified_rectify_Euler(f,a,b,n,ya)
    h=(b-a)/n; m_y(1)=ya;
    m_y(2)=ya+h*feval(f,a,ya); %initiate
    for i=1:n-1
        p=m_y(i)+2*h*feval(f,a+i*h,m_y(i+1)); %forecast
        if i==1; m=p;
        else
            m=p-4/5*(p1-c1);
        end
        c=m_y(i+1)+h/2*(feval(f,a+i*h,m_y(i+1))+feval(f,a+(i+1)*
            h,m)); %rectify
        m_y(i+2)=1/5*p+4/5*c;
        p1=p;c1=c;
    end
end

```

