第七章 常微分方程初值问题的数值解法

7.1 问题的提出

考虑常微分方程中最简 单的一类问题:一阶方程的初值问题.

$$\begin{cases} \dfrac{dy}{dx} = f(x,y) & x > x_0 \\ y(x_0) = y_0 \end{cases} \qquad (7.1)$$

当微分方程较复杂时，解函数$y = y(x)$的求解很困难。

如何解决？-近似解。

求（7.1）的数值解，即寻求解 $y(x)$在一系列离散点

$$a = x_0 < x_1 < x_2 < \cdots < x_n < x_n = b$$

上的值$y(x_1), y(x_2), \cdots, y(x_n)$

的近似值$y_1,\quad y_2,\quad \cdots,\quad y_n$ $\Big\}$ $y_n \approx y(x_n)$

相邻2个节点的间距 $h_i = x_{i+1} - x_i$称为步长，

一般取为常 数,即$h_i = h.$

初值问题（7.1）的数值解法过程：按 节点顺序依次求解

$y_0, y_1, \cdots, y_i, \cdots$

7.2 Euler法

7.2.1 欧拉公式

(7.1)式两边在区间 $[x_i, x_{i+1}]$上积分，得

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

即$$y(x_{i+1}) = y(x_i) + \int_{x_i}^{x_{i+1}} f(x, y(x))dx \quad (7.2)$$

应用左矩形公式，得

$$y(x_{i+1}) = y(x_i) + hf(x_i, y(x_i)) + R_{i+1}(\text{截断误差})$$

7.2 Euler法

略去截断误差，并用 y_i, y_{i+1} 代替 $y(x_i), y(x_{i+1})$，得

$$\left.\begin{aligned} y_{i+1} &= y_i + hf(x_i, y_i) \qquad i = 0,1,2,\cdots,n-1 \\ y_0 &= y(x_0) = y(a) = \eta \end{aligned}\right\}\text{--欧拉公式}$$

利用欧拉公式可依次求解 $y_1, y_2, \cdots, y_n$。

7.2.2 梯形公式

在（7.2）式中，对右端积分采 用梯形求积公式得：

$$y(x_{i+1}) \approx y(x_i) + \frac{h}{2}[f(x_i, y(x_i)) + f(x_{i+1}, y(x_{i+1}))]$$

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

$y_{i+1} = y_i + hf(x_i, y_i)$ 欧拉公式

观察欧拉公式和梯形公式递推计算过程及表达式的异同点.

欧拉公式：计算 y_{i+1}时用到 y_i,将 y_i带入公式，

可直接得 y_{i+1} --显格式；

梯形公式：计算 y_{i+1}时用到 y_i,将 y_i带入公式，

不可直接得 y_{i+1} --隐格式

如何将两者的优点结合起来？？？

7.2.3 改进欧拉公式（ *Euler*预估—校正法）

$$\begin{cases} \tilde{y}_{i+1} = y_i + hf(x_i, y_i) \\ y_{i+1} = y_i + \dfrac{h}{2}[f(x_i, y_i) + f(x_{i+1}, \tilde{y}_{i+1})] \end{cases} \qquad (7.3)$$

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

当y_i已知时，由第一式预估 出初值$\tilde{y}_{i+1}$，再代入第二式反复校正（迭代）。

7.2.4.局部截断误差和整体截断误差

1.单步显式公式的一般形式为：

$$\begin{cases} y_{i+1} = y_i + h\varphi(x_i, y_i, h) \quad i = 0,1,\cdots,n-1 \\ y_0 = \eta \end{cases}$$

（其中$\varphi(x, y, h)$为增量函数），其局部截断误差为

$$R_{i+1} = y(x_{i+1}) - y(x_i) - h\varphi(x_i, y(x_i), h).$$

于是，*Euler*公式的局部截断误差为：

$$\begin{aligned} R_{i+1} &= y(x_{i+1}) - y(x_i) - hf(x_i, y(x_i)) \\ &= y(x_i) + hy'(x_i) + \frac{h^2}{2} y''(\xi_i) - y(x_i) - hy'(x_i) \\ &= \frac{h^2}{2} y''(\xi_i) = O(h^2) \end{aligned}$$

2.单步隐式公式的一般形式为：

$$\begin{cases} y_{i+1} = y_i + h\psi(x_i, y_i, y_{i+1}, h) & i = 0,1,\cdots,n-1 \\ y_0 = \eta \end{cases}$$

（其中$\psi(x, y, \tilde{y}, h)$为增量函数），其局部截断误差为

$$R_{i+1} = y(x_{i+1}) - y(x_i) - h\psi(x_i, y(x_i), y(x_{i+1}), h).$$

于是，梯形公式的局部截断误差为：

$$R_{i+1} = y(x_{i+1}) - y(x_i) - \frac{h}{2}[f(x_i, y(x_i)) + f(x_{i+1}, y(x_{i+1}))]$$

$$= -\frac{h^3}{12} y'''(\xi_i) = O(h^3)$$

改进*Euler*公式也可表示为：

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

或
$$\begin{cases} y_{i+1}=y_i+\dfrac{h}{2}(k_1+k_2) \\ k_1=f(x_i,y_i) \\ k_2=f(x_{i+1},y_i+hk_1) \end{cases}$$

因而改进*Euler*公式本质上是单步显式公式，其局部截断误差为：

$$R_{i+1}=y(x_{i+1})-y(x_i)-\frac{h}{2}[f(x_i,y_i)+f(x_{i+1},y_i+hf(x_i,y_i))]$$

$$=O(h^3)$$

3.整体截断误差及精度

定义1: 设$y(x_1), y(x_2), \cdots, y(x_n)$为微分方程问题的解在节点处的值, $y_1(h), y_2(h), \cdots, y_n(h)$为用某种数值方法求得的近似解.称每一节点上误差的最大值,即$E(h) = \max\limits_{1 \le i \le n} | y(x_i) - y_i(h) |$为该方法的整体截断误差.

如果$\lim\limits_{h \to 0} E(h) = 0$则称该数值方法是收敛的.

定义2: 如果一个求解公式的局部截断误差为$O(h^{p+1})$, 则称该求解公式是p阶的，或具有p阶精度。

*Euler*公式，梯形公式和改进*Euler*公式的精度？

*Euler*公式具有一阶精度，梯形公式和改进*Euler*公式具有二阶精度。

7.2.5.稳定性

定义　用某方法固定步长h作计算，由初值y_0严格得出y_n,假设初值有微小误差δ_0(实际初值为$y_0+\delta o$),则引起第n步计算值有误差δ_n（实际为y_n+s_n）.若

$|\delta_n|\leq|\delta_0|$　　（$n=1,2,.....$）

则称该方法关于步长h绝对稳定

(即$|\delta_n|$不随n无限扩大).

稳定性比较复杂，通常只考虑“试验方程”

$y'=\lambda y$（$\lambda<0$，λ为常数）

举例：对*Euler*法

$$y_{n+1} = y_n + h(\lambda y_n) = (1+\lambda h)y_n = \cdots\cdots = (1+\lambda h)^{n+1} y_0$$

设初值有小扰动δ_0,则

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

与上式相减得

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

显然，*Euler*方法绝对稳定$\Leftrightarrow |1+\lambda h| \le 1$

Ex：对任意步长h,梯形法绝对稳定。

例1：分别写出求解下列初值问题的*Euler*公式和改进*Euler*公式：

$$\begin{cases} y' = -2xy \quad 0 \le x \le 1.8 \\ y(0) = 1 \end{cases}$$

取步长$h = 0.1$.

解：*Euler*公式为：

$$y_{i+1} = y_i + h(-2x_i y_i) = (1 - 0.2x_i)y_i;$$

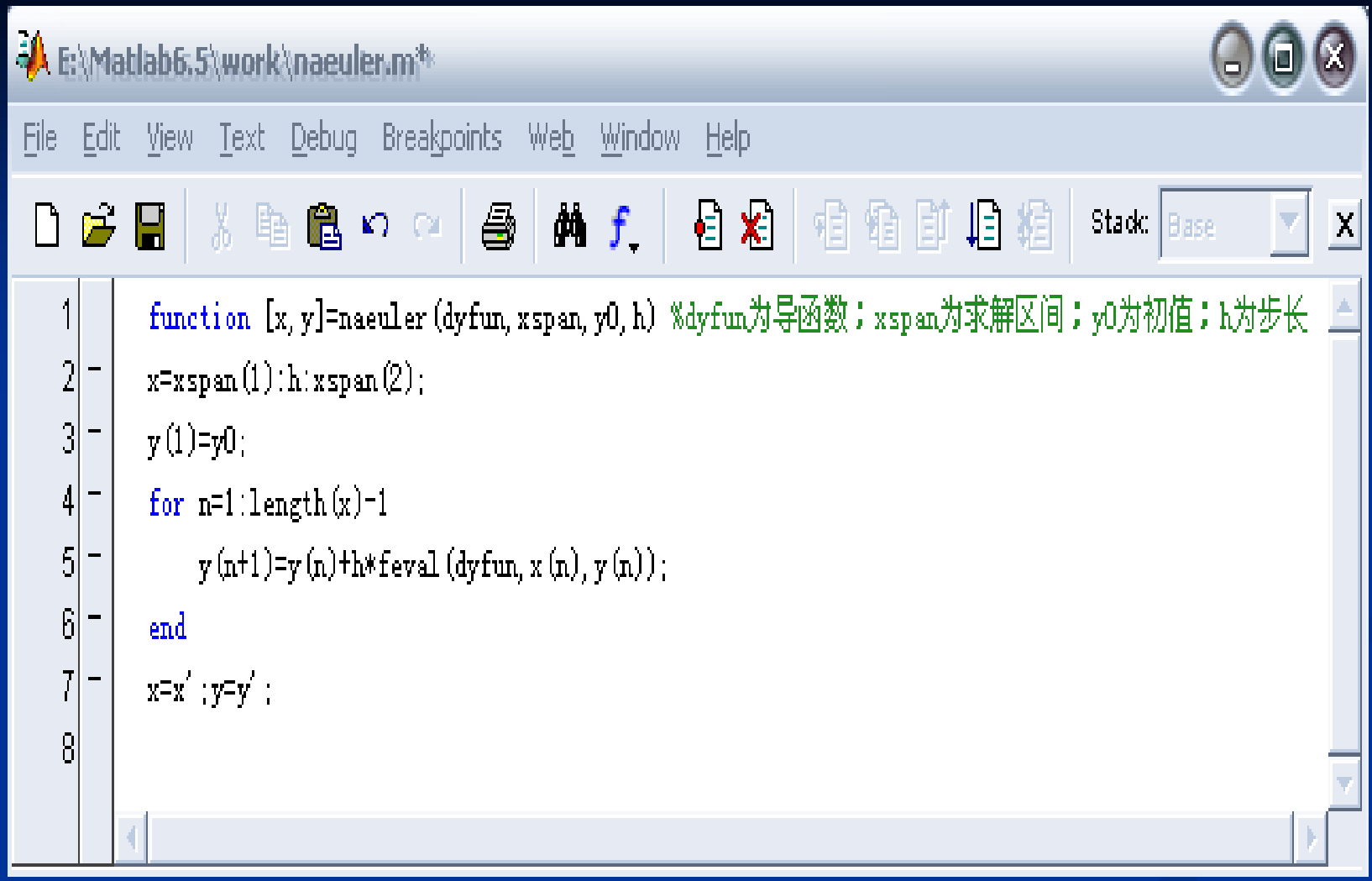
为了编程上机计算，改进*Euler*公式可改写。

$$\begin{cases} y_p = y_i + h(-2x_i y_i) = (1 - 0.2x_i)y_i \\ y_c = y_i + h(-2x_{i+1} y_p) = y_i - 0.2(x_i + 0.1)y_p \\ y_{i+1} = \dfrac{1}{2}(y_p + y_c) \end{cases}$$

用Matlab实现用Euler格式解常微分方程

在Matlab程序编辑器中输入：

```
function [x,y]=naeuler(dyfun,xspan,y0,h) %dyfun为导函数；xspan为求解区间；y0为初值；h为步长
x=xspan(1):h:xspan(2);
y(1)=y0;
for n=1:length(x)-1
    y(n+1)=y(n)+h*feval(dyfun,x(n),y(n));
end
x=x';y=y';
```


E:\Matlab6.5\work\naeuler.m*
File Edit View Text Debug Breakpoints Web Window Help
Stack: Base
1 function [x,y]=naeuler(dyfun,xspan,y0,h) %dyfun为导函数；xspan为求解区间；y0为初值；h为步长
2 x=xspan(1):h:xspan(2);
3 y(1)=y0;
4 for n=1:length(x)-1
5 y(n+1)=y(n)+h*feval(dyfun,x(n),y(n));
6 end
7 x=x';y=y';
8

Euler公式

求解得：

Command Window

```
>> clear;dyfun=inline('-2*x*y');
>> [x,y]=naeuler(dyfun,[0,1.8],1,0.1);[x,y]

ans =

         0    1.0000
    0.1000    1.0000
    0.2000    0.9800
    0.3000    0.9408
    0.4000    0.8844
    0.5000    0.8136
    0.6000    0.7322
    0.7000    0.6444
    0.8000    0.5542
    0.9000    0.4655
    1.0000    0.3817
    1.1000    0.3054
    1.2000    0.2382
    1.3000    0.1810
    1.4000    0.1340
    1.5000    0.0964
    1.6000    0.0675
    1.7000    0.0459
    1.8000    0.0303

>>
```

用Matlab实现用改进Euler公式解常微分方程

在Matlab程序编辑器中输入：

```
function [x,y]=naeuler2(dyfun,xspan,y0,h) %dyfun为导函数；xspan为求解区间；y0为初值；h为步长
x=xspan(1):h:xspan(2);y(1)=y0;
for n=1:length(x)-1
    k1=feval(dyfun,x(n),y(n));
    y(n+1)=y(n)+h*k1;
    k2=feval(dyfun,x(n+1),y(n+1));
    y(n+1)=y(n)+h*(k1+k2)/2;
end
x=x';y=y';
```

E:\Matlab6.5\work\naeuler2.m*

File Edit View Text Debug Breakpoints Web Window Help

Stack: Base

```
1   function [x,y]=naeuler2(dyfun,xspan,y0,h) %dyfun为导函数；xspan为求解区间；y0为初值；h为步长
2   x=xspan(1):h:xspan(2);y(1)=y0;
3   for n=1:length(x)-1
4       k1=feval(dyfun,x(n),y(n));
5       y(n+1)=y(n)+h*k1;
6       k2=feval(dyfun,x(n+1),y(n+1));
7       y(n+1)=y(n)+h*(k1+k2)/2;
8   end
9   x=x';y=y';
```

用改进Euler公式求得：

Command Window

```
>> clear;dyfun=inline('-2*x*y');
>> [x,y]=naeuler2(dyfun,[0,1.8],1,0.1);[x,y]

ans =

         0    1.0000
    0.1000    0.9900
    0.2000    0.9607
    0.3000    0.9138
    0.4000    0.8520
    0.5000    0.7788
    0.6000    0.6978
    0.7000    0.6129
    0.8000    0.5279
    0.9000    0.4457
    1.0000    0.3691
    1.1000    0.2997
    1.2000    0.2387
    1.3000    0.1864
    1.4000    0.1429
    1.5000    0.1075
    1.6000    0.0793
    1.7000    0.0574
    1.8000    0.0409

>>
```

基本要求:

1.掌握Euler公式、梯形公式及改进Euler公式;

2.会应用局部截断误差.

作业:见课堂在线

练习：实习部分例题

7.3 龙格-库塔方法

7.3.1 Runge – Kutta方法的基本思想：

由微分中值定理 $\frac{y(x_{i+1})-y(x_i)}{h}=y'(x_i+\theta h),$

及微分方程 $y'=f(x,y)$可得

$$y(x_{i+1})=y(x_i)+hf(x_i+\theta h,y(x_i+\theta h)),$$

这里$f(x_i+\theta h,y(x_i+\theta h))$称作区间$(x_i,x_{i+1})$上的平均斜率，记作$k^*$.可以看出只要对 k^*提供一种算法,就可得到一种微分方程的数值计算公式。用这个观点可以发 现*Euler*公式和改进*Euler*公式分别是以一点处的斜率和两点处的斜率值的平均值作为平均斜率。因此，精度不同。怎样达到最高？

7.3.2 二阶Runge – Kutta公式（改进*Euler*公式的推广）

考察区间$[x_i, x_{i+1}]$内的任一点$x_{i+l} = x_i + lh \quad 0 < l \le 1;$

我们希望用x_i和x_{i+l}两点的斜率值k_1和k_2加权平均作为平均斜率k^*近似值,即$k^* = \lambda_1 k_1 + \lambda_2 k_2$,从而取$y_{i+1} = y_i + h(\lambda_1 k_1 + \lambda_2 k_2)$,其中$\lambda_1, \lambda_2$是待定常数，故二阶Runge – Kutta公式具有如下形式：

$$\begin{cases} y_{i+1} = y_i + h(\lambda_1 k_1 + \lambda_2 k_2) \\ k_1 = f(x_i, y_i) \\ k_2 = f(x_{i+l}, y_i + lhk_1) \end{cases} \qquad (7.4)$$

公式中有三个待定参数,我们希望适当选取这些参数值，使得公式(7.4)具有二阶精度。

要使公式(7.4)具有二阶精度,

即截断误差$R_{n+1}=y(x_{n+1})-y_{n+1}=O(h^3)$,

得:
$$\begin{cases}\lambda_1+\lambda_2=1\\ l\lambda_2=\dfrac{1}{2}\end{cases}\qquad(7.5)$$

故满足(7.5)的3个参数均可使原格式具有二阶精度,

这些公式统称为二阶龙格-库塔公式.

式(7.5)的解为: $\lambda_1=1-\dfrac{1}{2l},\lambda_2=\dfrac{1}{2l}.$

特别,当$l=1$时,即$x_{i+l}=x_{i+1},\lambda_1=\lambda_2=\dfrac{1}{2}$时

二阶龙格-库塔公式成为改进的*Euler*公式.

特例：(1)$\begin{cases} y_{i+1} = y_i + hk_2 \\ k_1 = f(x_i, y_i) \\ k_2 = f(x_{i+\frac{1}{2}}, y_i + \frac{h}{2}k_1) \end{cases}$；

(2)$\begin{cases} y_{n+1} = y_n + \frac{1}{4}(k_1 + 3k_2) \\ k_1 = hf(x_n, y_n) \\ k_2 = hf(x_n + \frac{2}{3}h, y_n + \frac{2}{3}k_1) \end{cases}$

其中(1)为$l = \frac{1}{2}$的结果，称为变形的Euler公式；

(2)称为二阶*Heun*(休恩)格式.

7.3.3 高阶R - K方法介绍

(1) 常用三阶R - K方法

$$\begin{cases} y_{i+1} = y_i + \dfrac{h}{6}(k_1 + 4k_2 + k_3) \\ k_1 = f(x_i, y_i) \\ k_2 = f(x_i + \dfrac{h}{2}, y_i + \dfrac{hk_1}{2}) \\ k_3 = f(x_i + h, y_i - hk_1 + 2hk_2) \end{cases}$$

其截断误差均为$O(h^4)$.

(2) 经典(标准)四阶R - K方法(实用)

$$\begin{cases} y_{i+1} = y_i + \dfrac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\ k_1 = f(x_i, y_i) \\ k_2 = f(x_i + \dfrac{h}{2}, y_i + \dfrac{hk_1}{2}) \\ k_3 = f(x_i + \dfrac{h}{2}, y_i + \dfrac{hk_2}{2}) \\ k_4 = f(x_i + h, y_i + hk_3) \end{cases}$$

其截断误差均为$O(h^5)$.

(3) Gill(基尔)公式 $\rightarrow$ 稳定性好

$$\begin{cases} y_{i+1} = y_i + \dfrac{h}{6}(k_1 + (2-\sqrt{2})\ k_2 + (2+\sqrt{2})\ k_3 + k_4) \\ k_1 = f(x_i, y_i) \\ k_2 = f(x_i + \dfrac{h}{2}, y_i + \dfrac{hk_1}{2}) \\ k_3 = f(x_i + \dfrac{h}{2}, y_i + \dfrac{\sqrt{2}-1}{2} hk_2 + (1-\dfrac{\sqrt{2}}{2})\ hk_2) \\ k_4 = f(x_i + h, y_i - \dfrac{\sqrt{2}}{2} hk_2 + (1+\dfrac{\sqrt{2}}{2})\ hk_3) \end{cases}$$

其截断误差均为$O(h^5)$.

例：写出求解下列初值问题的四阶龙格-库塔公式：

$$\begin{cases} y' = -2xy \quad 0 \le x \le 1.8 \\ y(0) = 1 \end{cases}$$

取步长$h = 0.2.$

解：具体形式为

$$\begin{cases} y_{i+1} = y_i + \dfrac{0.2}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\ k_1 = -2x_i y_i \\ k_2 = -2(x_i + 0.1)(y_i + 0.1k_1) \\ k_3 = -2(x_i + 0.1)(y_i + 0.1k_2) \\ k_4 = -2(x_i + 0.2)(y_i + 0.2k_3) \end{cases}$$

x_i	y_i	$y(x_i)$	$\lvert y(x_i)-y_i \rvert$
0	1.0000000	1.0000000	0.0000000
0.2	0.9607893	0.9607894	0.0000001
0.4	0.8521429	0.8521438	0.0000008
0.6	0.6976755	0.6976763	0.0000008
0.8	0.5272977	0.5272924	0.0000053
1.0	0.3679036	0.3678795	0.0000242
1.2	0.2369857	0.2369277	0.0000579
1.4	0.1409576	0.1408584	0.0000992
1.6	0.0774387	0.0773047	0.0001340
1.8	0.0393135	0.0391639	0.0001496

对这表的写法有疑问吗?

求初值公式的出发点是什么?

理论上来说，可以构造任意高阶的龙格-库塔公式，但是要注意精度的阶数与计算函数值$f(x,y)$的次数间的关系不是等量增加的，且精度越高表达式越复杂。四阶龙格-库塔公式是兼顾了两者，见下表。

每步计算f的次数	2	3	4	5	6	7	8	9
精度的阶数	2	3	4	4	5	6	6	7

注意事项：龙格-库塔方法是基于泰勒展开方法的，因而要求所求微分方程问题的解具有较好的光滑性质。假若解的光滑性差，那么使用四阶龙格-库塔公式求得的数值解，其精度可能反而不如改进欧拉公式。

7.3.4 变步长使用R - K公式：

取定步长h(可稍大)，用某R - K方法由y_i算出$y(x_{i+1})$的近似值，记为$y_{i+1}^{(h)}$。现将步长h减半，以$\frac{h}{2}$为步长，经两步算出$y(x_{i+1})$的近似值，记为$y_{i+1}^{(h/2)}$。若

$$\left|y_{i+1}^{(h/2)} - y_{i+1}^{(h)}\right| \le \varepsilon$$

则取$y(x_{i+1}) = y_{i+1}^{(h/2)}$，否则继续将步长减半，经计算出$y(x_{i+1})$的近似值$y_{i+1}^{(h/4)}$，并检验

$$\left|y_{i+1}^{(h/4)} - y_{i+1}^{(h/2)}\right| \le \varepsilon$$

是否成立。若成立，则 取$y_{i+1} = y_{i+1}^{(h/4)}$，否则将步长减半，直到相邻两次计算结果满足误差要求为止.

用Matlab实现四阶Runge-Kutta格式求解常微分方程

在Matlab程序编辑器中输入：

```
function [x,y]=nark4(dyfun,xspan,y0,h)
x=xspan(1):h:xspan(2);
y(1)=y0;
for n=1:length(x)-1
    k1=feval(dyfun,x(n),y(n));
    k2=feval(dyfun,x(n)+h/2,y(n)+h/2*k1);
    k3=feval(dyfun,x(n)+h/2,y(n)+h/2*k2);
    k4=feval(dyfun,x(n)+h,y(n)+h*k3);
    y(n+1)=y(n)+h/6*(k1+2*k2+2*k3+k4);
end
x=x';y=y';
```

E:\Matlab6.5\work\nark4.m

File Edit View Text Debug Breakpoints Web Window Help

```
1   function [x,y]=nark4(dyfun,xspan,y0,h)
2   x=xspan(1):h:xspan(2);
3   y(1)=y0;
4   for n=1:length(x)-1
5       k1=feval(dyfun,x(n),y(n));
6       k2=feval(dyfun,x(n)+h/2,y(n)+h/2*k1);
7       k3=feval(dyfun,x(n)+h/2,y(n)+h/2*k2);
8       k4=feval(dyfun,x(n)+h,y(n)+h*k3);
9       y(n+1)=y(n)+h/6*(k1+2*k2+2*k3+k4);
10  end
11  x=x';y=y';
12
```

Runge-Kutta

格式求解得：

Command Window

```
>> clear;dyfun=inline('-2*x*y');
>> [x,y]=mark4(dyfun,[0,1.8],1,0.2);[x,y]

ans =

         0    1.0000
    0.2000    0.9608
    0.4000    0.8521
    0.6000    0.6977
    0.8000    0.5273
    1.0000    0.3679
    1.2000    0.2370
    1.4000    0.1410
    1.6000    0.0774
    1.8000    0.0393

>>
```

7.4 线性多步法

前述方法特点：计算 y_{i+1}时主要使用了y_i－单步法；

事实上，计算y_{i+1}时，已经求出了$y_{0,}y_1,\cdots,y_i$，

如果能充分利用第$i+1$步前面已求得的多步信息预测y_{i+1}，

以期望获得较高的精度，这就是线性多步法的基本思想。

线性r步公式可表示为

$$y_{i+1}=\sum_{j=0}^{r-1}\alpha_j y_{i-j}+h\sum_{j=-1}^{r-1}\beta_j f(x_{i-j},y_{i-j}) \qquad (7.6)$$

式中α_j，β_j为常数，$|\alpha_{r-1}|+|\beta_{r-1}|\neq 0$时，（7.6）为显式格式，

当$\beta_{-1}\neq 0$时，（7.6）为隐式格式。

线性多步法的构造-数值积分法

选取节点x_{i-2}，x_{i-1}，x_i，x_{i+1}对应的插值多项式代入（7.2）得阿当姆斯内插公式：

$$y_{i+1} = y_i + \frac{h}{24}[9f(x_{i+1}, y_{i+1}) + 19f(x_i, y_i) - 5f(x_{i-1}, y_{i-1}) + f(x_{i-2}, y_{i-2})]$$

同样，若选取$x_{i-3}, x_{i-2}, x_{i-1}, x_i$作为插值节点，则得阿当姆斯外推公式：

$$y_{i+1} = y_i + \frac{h}{24}[55f(x_i, y_i) - 59f(x_{i-1}, y_{i-1}) + 37f(x_{i-2}, y_{i-2}) - 9f(x_{i-3}, y_{i-3})]$$

注意：

（1）出发值计算：线性多步法不能自动开始计算，一般先由同阶R-K单步法或其他公式求出所需初值。

（2）阿当姆斯内插公式是隐格式，阿当姆斯外推公式是显格式

（3）一般内插公式比外推公式误差小，所以？？

阿当姆斯预测校正公式=阿当姆斯内插公式+阿当姆斯外推公式

$$\begin{cases} \tilde{y}_{i+1} = y_i + \dfrac{h}{24}[55 f(x_i, y_i) + 59 f(x_{i-1}, y_{i-1}) \\ \qquad + 37 f(x_{i-2}, y_{i-2}) - 9 f(x_{i-3}, y_{i-3})] \\ y_{i+1} = y_i + \dfrac{h}{24}[9 f(x_{i+1}, \tilde{y}_{i+1}) + 19 f(x_i, y_i) \\ \qquad - 5 f(x_{i-1}, y_{i-1}) + f(x_{i-2}, y_{i-2})] \end{cases}$$

基本要求:

1.熟悉二阶R-K公式;
2.了解阿当姆斯公式;
3.会编程在计算机上使用各种方法.

作业: 课堂在线
练习：实习部分例题

第八章　矩阵特征值和特征向量计算

8.1　问题的提出

数学和物理中，很多问题需要计算矩阵的特征值和特征向量。

当矩阵A的阶数较高时，求解过程会变得比较复杂。

8.2 按模最大与最小特征值的求法

一．幂法——求A的主特征值（按模最大者）及其相应的特征向量

设A的特征值为$\lambda_1 \quad \lambda_2 \quad \cdots \quad \lambda_n$，且$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n|$

情况一：当$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$

假设$x_1, x_2, \cdots, x_n$为矩阵A的n个线性无关特征向量

$$Ax_i = \lambda_i x_i \quad (i = 1,2,\cdots,n)$$

任取初始向量 v_0，则有

$v_0 = \alpha_1 x_1 + \alpha_2 x_2 + \cdots \alpha_n x_n$，并设$\alpha_1 \neq 0$.

利用$v_{k+1}=Av_k\,(k=0,1,2,\cdots)$进行迭代，得

$$v_1=Av_0=A(\alpha_1x_1+\alpha_2x_2+\cdots+\alpha_nx_n)$$

$$=\alpha_1\lambda_1x_1+\alpha_2\lambda_2x_2+\cdots+\alpha_n\lambda_nx_n$$

$$v_2=Av_1=A^2v_0$$

$$=\alpha_1\lambda_1^{\,2}x_1+\alpha_2\lambda_2^{\,2}x_2+\cdots+\alpha_n\lambda_n^{\,2}x_n$$

$$\vdots$$

$$v_k=A^kv_0=\alpha_1\lambda_1^{\,k}x_1+\alpha_2\lambda_2^{\,k}x_2+\cdots+\alpha_n\lambda_n^{\,k}x_n$$

改写成：$v_k = \lambda_1^k [\alpha_1 x_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k x_2 + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1}\right)^k x_n]$

同理$v_{k+1} = \lambda_1^{k+1} [\alpha_1 x_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^{k+1} x_2 + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1}\right)^{k+1} x_n]$

因为$|\lambda_i| < |\lambda_1|$ ，得$\lim_{k\to\infty} \left(\frac{\lambda_i}{\lambda_1}\right)^k = 0, \quad i = 2,3,\cdots,n$

可得$v_{k+1} \approx \lambda_1 v_k$

可见v_{k+1}和v_k应近似线性相关，常数 λ_1就是按模最大特征值，可利用$\frac{(v_{k+1})_j}{(v_k)_j} \approx \lambda_1$

乘幂计算过程中,当K充分大时,若$|\lambda_1|>1$,则v_k分量绝对值可能很大;若$|\lambda_1|<1$,则v_k分量绝对值过小.为防止此现象发生,实用中常常是迭代 m步后对v_k作一次规范化,

用$u_k=\dfrac{v_k}{(v_k)_{\max}}$或$u_k=\dfrac{v_k}{(v_k)_{\min}}$代替$v_k$继续迭代,

这里$(v_k)_{\max}$和$(v_k)_{\min}$分别表示向量v_k的按模最大分量和最小分量。

课本上使用$\max(v_k)$和$\min(v_k)$分别表示向量 v_k的按模最大分量和最小分量。

于是，改写后的算法为:

(1)任取一个初始向量$v_0 \neq 0$

(2)构造迭代序列

$$\begin{cases} u_0 = v_0 \\ v_k = Au_{k-1} \quad (k = 1,2,3,...) \\ m_k = \max(v_k) \\ u_k = v_k / m_k \end{cases}$$

式中$m_k = \max(v_k)$表示v_k中首次出现的模最大的分量，

如$v_k = (3,-5,2)^{\mathrm{T}}$，则 $\max(v_k) = -5$，于是规一化后所得向量为$u_k = (-\frac{3}{5},1,-\frac{2}{5})^{\mathrm{T}}$.

(3)取$\lim\limits_{k\to\infty} m_k = \lambda_1, \lim\limits_{k\to\infty} u_k = \dfrac{x_1}{\max(x_1)}$

算法中两个公式的证明:

$$(1)u_k=\frac{A^k u_0}{m_k m_{k-1}\cdots m_1}=\frac{A^k v_0}{\max(A^k v_0)}=\frac{\lambda_1^{\,k}[\alpha_1 x_1+\sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^k x_i]}{\max[\lambda_1^{\,k}(\alpha_1 x_1+\sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^k x_i)]}$$

$$=\frac{\lambda_1^{\,k}[\alpha_1 x_1+\sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^k x_i]}{\lambda_1^{\,k}\max[\alpha_1 x_1+\sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^k x_i]}=\frac{\alpha_1 x_1+\sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^k x_i}{\max[\alpha_1 x_1+\sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^k x_i]}$$

于是, $\lim_{k\to\infty}u_k=\frac{x_1}{\max(x_1)}$.

$$(2)v_k = Au_{k-1} = A\frac{A^{k-1}v_0}{\max(A^{k-1}v_0)} = \frac{\lambda_1^{\,k}[\alpha_1 x_1 + \sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^k x_i]}{\max[\lambda_1^{\,k-1}(\alpha_1 x_1 + \sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^{k-1} x_i)]}$$

$$= \frac{\lambda_1^{\,k}[\alpha_1 x_1 + \sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^k x_i]}{\lambda_1^{\,k-1}\max[\alpha_1 x_1 + \sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^k x_i]}$$,取分量模的最大值，得到

$$\max(v_k) = \frac{\lambda_1 \max[\alpha_1 x_1 + \sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^k x_i]}{\max[\alpha_1 x_1 + \sum_{i=2}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^{k-1} x_i]},$$

当$k \to \infty$时，$\max(v_k) \to \lambda_1$.

例1：用幂法求矩阵

$$A=\begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}$$

模最大的特征值及其对应的特征向量。

解：迭代两次为例.

情况二：若特征值满足$|\lambda_1|=|\lambda_2|>|\lambda_3|\geq\cdots\geq|\lambda_n|$,则得

$$v_k=\lambda_1^k[\alpha_1x_1+\alpha_2(\frac{\lambda_2}{\lambda_1})^kx_2+\sum_{i=3}^{n}\alpha_i(\frac{\lambda_i}{\lambda_1})^kx_i]$$

因为$|\lambda_1|>|\lambda_i|(i=3,4,...,n)$，所以$\lim_{k\to\infty}(\frac{\lambda_i}{\lambda_1})^k=0(i=3,4,...,n)$

则有$v_k\approx\lambda_1^k\alpha_1x_1+\lambda_2^k\alpha_2x_2$；同理，

$v_{k+1}\approx\lambda_1^{k+1}\alpha_1x_1+\lambda_2^{k+1}\alpha_2x_2$，$v_{k+2}\approx\lambda_1^{k+2}\alpha_1x_1+\lambda_2^{k+2}\alpha_2x_2$，

于是，$v_{k+2}-(\lambda_1+\lambda_2)v_{k+1}+\lambda_1\lambda_2v_k$

$\approx[\lambda_1^{k+2}-(\lambda_1+\lambda_2)\lambda_1^{k+1}+\lambda_1\lambda_2\lambda_1^k]\alpha_1x_1$

$+[\lambda_2^{k+2}-(\lambda_1+\lambda_2)\lambda_2^{k+1}+\lambda_1\lambda_2\lambda_2^k]\alpha_2x_2=0$，

即v_k，v_{k+1}，v_{k+2}三个向量大体上线性相关.

令$p=-(\lambda_1+\lambda_2), q=\lambda_1\lambda_2$,可用最小二乘法确定$p,q$.

从而$\lambda_1=-\frac{p}{2}+\sqrt{\frac{p^2}{4}-q}, \lambda_2=-\frac{p}{2}-\sqrt{\frac{p^2}{4}-q},$

当$\lambda_1\neq\lambda_2$，有$v_{k+1}-\lambda_2 v_k\approx\lambda_1^k(\lambda_1-\lambda_2)\ \alpha_1 x_1$及

$v_{k+1}-\lambda_1 v_k\approx\lambda_2^k(\lambda_2-\lambda_1)\ \alpha_2 x_2$,即$v_{k+1}-\lambda_2 v_k$与$x_1$成比例，

$v_{k+1}-\lambda_1 v_k$与x_2成比例，可以作为对应于λ_1及λ_2的特征向量。

当$\lambda_1=\lambda_2$时，只能求得一个特征向量；如果要求另一特征向量，可以不同的初始值v_0再作迭代。

说明：初始向量的选取对迭代次数是有影响的，若选取的v_0中α_1较小，则迭代次数就可能增加。（见课本例1说明）

乘幂法的速度与比值$\left|\frac{\lambda_2}{\lambda_1}\right|$有关,该比值越小,

收敛越快 – – ***Rayleigh***商加速法(了解，练习实习8).

二．反幂法——求A的按模最小的特征值

设A可逆，特征值$|\lambda_1| \geq \cdots \geq |\lambda_{n-1}| \geq |\lambda_n| > 0.$

由$Ax_j = \lambda_j x_j$,得$A^{-1}x_j = \frac{1}{\lambda_j} x_j$，且$\frac{1}{\lambda_n}$一定是$A^{-1}$的按模最大的特征值，故对$A^{-1}$用乘幂法可得$\frac{1}{\lambda_n}$的近似值，就是对$A$求按模最小特征值及特征向量，用$A^{-1}$代替$A$作幂法称为反幂法)

二. 反幂法——求A的按模最小的特征值

即任给初始向量 v_0,作如下迭代：$v_k = A^{-1}u_{k-1}(k = 0,1,2\cdots)$

由于A^{-1}需要计算，故可改写为：$Av_k = u_{k-1}$.

若采用“规一化”方法，得
$$\begin{cases} Av_k = u_{k-1} \\ m_k = \max(v_k) \\ u_k = v_k / m_k \\ u_0 = v_0 \end{cases}$$

注：（1）每迭代一次需要解 1个线性方程组$Av_k = u_{k-1}$，所以计算量较大，具体计算时，先把A作LU分解，这样每次迭代只要解 2个三角方程组。

（2）反幂法收敛速度取决于$|\frac{\lambda_n}{\lambda_{n-1}}|$.如果已知一个较好的初始向量，反幂法可以用来求任意特征值。

例2：用反幂法求矩阵

$$A=\begin{pmatrix}2 & 8 & 9\\ 8 & 3 & 4\\ 9 & 4 & 7\end{pmatrix}$$

按模最小的特征值及其对应的特征向量。

基本要求：

1. 掌握乘幂法求主特征值及其特征向量；
2. 了解反幂法的思路

作业：习题8- 1.

练习：实习部分例题