第三章 线性方程组数值解法

3.1 问题的提出

解线性方程组 $Ax = b$

直接法：理论，无舍入误差，有限步，精确解

迭代法：格式，无穷序列 $\rightarrow$ 解向量 x

3.2.1 三角方程组的解法

- 什么是三角方程组？

$$
\begin{cases}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = y_1 \\
\qquad a_{22}x_2 + \cdots + a_{2n}x_n = y_2 \\
\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
\qquad\qquad a_{ii}x_i + \cdots + a_{in}x_n = y_i \\
\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
\qquad\qquad\qquad a_{nn}x_n = y_n
\end{cases}
$$

- 回代，求解。

3.2.2 高斯消去法

设有

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = a_{1,n+1} \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = a_{2,n+1} \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n = a_{i,n+1} \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = a_{n,n+1} \end{cases} \tag{3.1}$$

消 元： 用a_{11}将$a_{i1}(i = 2,\cdots,n)$化为零；

把$\left(-\dfrac{a_{i1}}{a_{11}}\right)$×第1行，加到第$i$ 行。

问 题： $a_{11} = 0$或$|a_{11}| \approx 0$?以后各步类似。

Gauss消去法-Matlab实现

- Function x=nagauss(a,b,flag)　%解线形方程组ax=b，a为系数矩阵，b为右端列向量，flag若为0，则显示中间过程，否则不显示，默认为0，x为解向量
- if nargin<3,flag=0;end
- n=length(b); a=[a,b];
- % 消元
- for k=1:(n-1)
- 　　a((k+1):n,(k+1):(n+1))=a((k+1):n,(k+1):(n+1))-a((k+1):n,k)/a(k,k)*a(k,(k+1):(n+1));
- 　　a((k+1):n,k)=zeros(n-k,1);
- 　　if flag==0,a,end
- end

```
% 回代
x=zeros(n, 1);
x(n)=a(n, n+1)/a(n, n);
for  k=n-1:-1:1
      x(k, :)=(a(k, n+1)-
a(k, (k+1):n)*x((k+1):n))/a(k, k);
end
```

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

File Edit View Text Debug Breakpoints Web Window Help

Stack: Base

```
function x=nagauss(a,b,flag)  %解线形方程组ax=b，a为系数矩阵，b为右端列向量，flag若为0，则
                              %显示中间过程，否则不显示，默认为0，x为解向量
if nargin<3,flag=0;end
n=length(b); a=[a,b];
% 消元
for k=1:(n-1)
   a((k+1):n,(k+1):(n+1))=a((k+1):n,(k+1):(n+1))-a((k+1):n,k)/a(k,k)*a(k,(k+1):(n+1));
   a((k+1):n,k)=zeros(n-k,1);
   if flag==0,a,end
end
% 回代
x=zeros(n,1);
x(n)=a(n,n+1)/a(n,n);
for k=n-1:-1:1
    x(k,:)=(a(k,n+1)-a(k,(k+1):n)*x((k+1):n))/a(k,k);
end
```

程序运行

结果：

Command Window

```
>> a=[1 1 1;-1 3 1;2 -6 1];b=[6;4;5];
>> x=nagauss(a,b)

a =

     1     1     1     6
     0     4     2    10
     0    -8    -1    -7


a =

     1     1     1     6
     0     4     2    10
     0     0     3    13


x =

    1.3333
    0.3333
    4.3333

>>
```

3.2.3 高斯消去法解三对角方程组——追赶法

给定
$$\begin{bmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & a_n & b_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

（三对角方程组）

且按行对角占优：$|b_1| > |c_1| > 0$，$|b_i| \geq |a_i| + |c_i| (a_i c_i \neq 0)$，

$|b_n| \geq |a_n|$　　$(i = 2, \cdots, n-1)$

- 利用Gauss消元法得到同解的三角方程为

$$\begin{bmatrix} \beta_1 & c_1 & & & & y_1 \\ & \beta_2 & c_2 & & & y_2 \\ & & \ddots & \ddots & & \vdots \\ & & & \beta_{n-1} & c_{n-1} & y_{n-1} \\ & & & & \beta_n & y_n \end{bmatrix}$$

消元过程：

$$\begin{cases} \beta_1 = b_1, \quad y_1 = d_1 \\ l_i = \alpha_i / \beta_{i-1}, \beta_i = b_i - l_i c_{i-1} \\ y_i = d_i - l_i y_{i-1} \quad i = 2,3,\cdots,n \end{cases}$$

回代过程为

$$\begin{cases} x_n = y_n / \beta_n \\ x_i = (y_i - c_i x_{i+1}) / \beta_i \\ i = n-1, n-2, \cdots, 1 \end{cases}$$

以上两个递推公式表示出来的方法被形象地叫做追赶法。

3.2.4 列主元素Gauss消去法---计算结果可靠

(1)找行号r_1 使$\left|a_{r_1 1}\right| = \max\limits_{1\le i\le n}\left|a_{i1}\right|$， 对调$1 \leftrightarrow r_1$行：

消元： 用a_{11}消a_{i1}为0：

第1行$\times\left(-\dfrac{a_{i1}}{a_{11}}\right)$加到第$i$行，第$i$行第$j$个元素成为

$$a_{ij} + a_{1j}\left(-\frac{a_{i1}}{a_{11}}\right) \Rightarrow a_{ij}$$

$(i = 2,3,\cdots,n;\ \ j = 1,2,\cdots,n+1)$

到此原方程组化为

$$
\begin{cases}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = a_{1,n+1} \\
a_{22}x_2 + \cdots + a_{2n}x_n = a_{2,n+1} \\
\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
a_{i2}x_2 + \cdots + a_{in}x_n = a_{i,n+1} \\
\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
a_{n2}x_2 + \cdots + a_{nn}x_n = a_{n,n+1}
\end{cases}
$$

(2) 找r_2，使$\left|a_{r_2 2}\right| = \max\limits_{2\le i\le n}\left|a_{i2}\right|$，

对调$2 \leftrightarrow r_2$行.

消元：用a_{22}把a_{i2}消为0 $(i = 3,4,\cdots,n)$：

$$\text{第}2\text{行}\times\left(-\frac{a_{i2}}{a_{22}}\right)+\text{第}i\text{行}，\text{则}$$

$$a_{ij}+a_{2j}\left(-\frac{a_{i2}}{a_{22}}\right)\Rightarrow a_{ij}$$

$(i = 3,4,\cdots,n;\ \ j = 2,3,\cdots,n+1)$

直到(n-1)步， 原方程组化为

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = a_{1,n+1} \\ a_{22}x_2 + \cdots + a_{2n}x_n = a_{2,n+1} \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{nn}x_n = a_{n,n+1} \end{cases}$$

（上三角方程组）

以上为消元过程。

（n）回代求解公式

$$\begin{cases} x_n = a_{n,n+1} \Big/ a_{nn} \\ x_k = \dfrac{1}{a_{kk}}\left[a_{k,n+1} - \sum_{j=k+1}^{n} a_{kj}x_j\right] \end{cases}$$

$$(k = n-1, n-2, \ldots, 1)$$

说明：

(1) 也可采用无回代的列主元消去法(叫Gauss-Jordan消去法)，但比有回代的列主元消去法的乘除运算次数多。

(2) 有回代的列主元消去法所进行的乘除运算次数为 $\frac{1}{3}n^3+n^2-\frac{1}{3}n$，量很小。

Gauss 列主元消去法：

优点 ------ 计算结果更可靠；

缺点 ------ 挑主元耗时更多，变量次序有变动，程序复杂。

- 系数矩阵为对称正定阵或严格对角占优阵的方程组按高斯消去法计算是数值稳定的，因而不必选主元。
- 正定矩阵:设M是n阶方阵，如果对任何非零向量z，都有$z^T M z > 0$，其中z^T表示z的转置，就称M正定矩阵。
- 严格对角占优阵：行(列)严格对角占优阵
- 行严格对角占优阵：$|a_{ii}| > \sum_{j=1, i\neq j}^{n} |a_{ij}|, i = 1,2,\cdots,n$
- 列严格对角占优阵？？

用Matlab实现选列主元Gauss消去法解线性方程组

在Matlab程序编辑器中输入：

```
function x=nagauss2(a,b,flag)  %a为系数矩阵；b为右端列向量；flag若为0，则显示中间过程，否则不显示
if nargin<3,flag=0;end
n=length(b); a=[a,b];
% 选主元
for k=1:(n-1)
[ap,p]=max(abs(a(k:n,k)));p=p+k-1;
if p>k,t=a(k,:); a(k,:)=a(p,:); a(p,:)=t; end
% 消元
    a((k+1):n,(k+1):(n+1))=a((k+1):n,(k+1):(n+1))-a((k+1):n,k)/a(k,k)*a(k,(k+1):(n+1));
    a((k+1):n,k)=zeros(n-k,1);
```

```
if flag==0,a,end
end
%回代
x=zeros(n,1);
x(n)=a(n,n+1)/a(n,n);
for  k=n-1:-1:1
      x(k,:)=(a(k,n+1)-
a(k,(k+1):n)*x((k+1):n))/a(k,k);
end
```

E:\Matlab6.5\work\nagauss2.m

File Edit View Text Debug Breakpoints Web Window Help

Stack: Base

```
function x=nagauss2(a,b,flag)  %a为系数矩阵；b为右端列向量；flag若为0，则显示中间过程，否则不显示
if nargin<3,flag=0;end
n=length(b); a=[a,b];
% 选主元
for k=1:(n-1)
[ap,p]=max(abs(a(k:n,k)));p=p+k-1;
if p>k,t=a(k,:); a(k,:)=a(p,:); a(p,:)=t; end
 %消元
   a((k+1):n,(k+1):(n+1))=a((k+1):n,(k+1):(n+1))-a((k+1):n,k)/a(k,k)*a(k,(k+1):(n+1));
   a((k+1):n,k)=zeros(n-k,1);
   if flag==0,a,end
end
%回代
x=zeros(n,1);
x(n)=a(n,n+1)/a(n,n);
for k=n-1:-1:1
    x(k,:)=(a(k,n+1)-a(k,(k+1):n)*x((k+1):n))/a(k,k);
end
```

nagauss2 Ln 13 Col 4

程序运行

结果：

Command Window

```
>> a=[1 1 1;-1 3 1;2 -6 1];b=[6;4;-5];
>> x=nagauss2(a,b)

a =

    2.0000   -6.0000    1.0000   -5.0000
         0         0    1.5000    1.5000
         0    4.0000    0.5000    8.5000


a =

    2.0000   -6.0000    1.0000   -5.0000
         0    4.0000    0.5000    8.5000
         0         0    1.5000    1.5000


x =

     3
     2
     1
```

3.3. 矩阵的直接分解及其在解方程中的应用

3.3.1 矩阵分解的紧凑格式

（考虑Gauss消元法的过程用初等矩阵刻画）

$(A|b) \to \cdots \to (U|g)$上三角形矩阵

$L_k L_{k-1} \cdots L_2 L_1 (A|b) = (U|g)$

$L_k L_{k-1} \cdots L_2 L_1 A = U$

记 $(L_k L_{k-1} \cdots L_2 L_1)^{-1} = L$, 则 $A = (L_k L_{k-1} \cdots L_2 L_1)^{-1} U$

$A = LU$, U是上三角形矩阵，L呢？

练习：习题3-5,6

定义3.1 $A=LU$ 叫 A 的三角（因子）分解，其中 L 是下三角，U 是上三角。

定义3.2 若 L 为单位下三角阵（对角元全为1），U 为上三角阵，则称 $A=LU$ 为Doolittle分解；若 L 是下三角，U 是单位上三角，则称 $A=LU$ 为Crout分解。

为什么要讨论三角分解？若在消元法进行前能实现三角分解 $A = LU$ ， 则

$$Ax = b \Leftrightarrow (LU)x = b \Leftrightarrow$$

$Ly = b$ (下三角方程组）

$Ux = y$ (上三角方程组）

从而容易回代求解。

直接三角分解法(Doolittle分解为例)

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} =$$

$$\begin{pmatrix} 1 & & & \\ l_{21} & 1 & & \\ \cdots & \cdots & \cdots & \\ l_{n1} & l_{n2} & \cdots & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ & u_{22} & \cdots & u_{2n} \\ & & \cdots & \cdots \\ & & \cdots & u_{nn} \end{pmatrix}$$

由矩阵乘法

(1) 1) 求u的第1行：用L的第一行$\times u$的第j列

$$1.\ \mathrm{u}_{1j} = a_{1j} \Rightarrow u_{1j} = a_{1j} \qquad (\mathrm{j} = 1,2,\cdots,\mathrm{n})$$

2) 求L的第1列：用L的第i行$\times u$的第1列

$$l_{i1} \cdot u_{11} = a_{i1} \Rightarrow l_{i1} = \frac{a_{i1}}{u_{11}} \qquad (i = 2,3,\cdots,n)$$

(2) 1)求u的第2行：用L的第2行$\times u$的第j列

$$(j = 2,\cdots,\mathrm{n})\ l_{21} \cdot u_{1j} + 1 \cdot u_{2j} = a_{2j} \Rightarrow u_{2j} = a_{2j} - l_{21}u_{1j}$$

2)求L的第2列：用L的第i行$\times u$的第2列

$$(i = 3,4,\cdots,\mathrm{n})$$

$$l_{i1} \cdot u_{12} + l_{i2} \cdot u_{22} = a_{i2} \Rightarrow l_{i2} = (a_{i2} - l_{i1}u_{12}) / u_{22}$$

……

……

(k)

1）求u的第k行：用L的第k行$\times u$的第j列

$(\mathrm{j}=\mathrm{k},\mathrm{k}+1,\cdots,\mathrm{n})$

$$(l_{k1},l_{k2},\cdots,l_{kk},0\cdots,0)\cdot(u_{1j},u_{2j},\cdots,u_{jj},0\cdots,0)'=a_{kj}$$

$$\sum_{q=1}^{k-1}l_{kq}u_{qj}+1\cdot u_{kj}=a_{kj}\Rightarrow u_{kj}=a_{kj}-\sum_{q=1}^{k-1}l_{kq}u_{qj}$$

2）求L的第k列：用L的第i行×u的第k列$(\mathrm{i}=\mathrm{k}+1,\cdots,\mathrm{n})$，即

$$(l_{i1},\cdots,l_{ik},\cdots l_{kk},0\cdots 0)\cdot(u_{1k},u_{2k},\cdots,u_{kk},0\cdots 0)'$$

$$=a_{ik}$$

$$\sum_{q=1}^{k-1} l_{iq}u_{qk}+l_{ik}u_{kk}=a_{ik}$$

$$l_{ik}=\left[a_{ik}-\sum_{q=1}^{k-1} l_{iq}u_{qk}\right]\Big/u_{kk}$$

*LU*分解式：

$$
\begin{cases}
u_{1j} = a_{1j} \quad (j = 1,2,\cdots n) \\
l_{i1} = a_{i1} / u_{11} \quad (i = 2,3,\cdots,n) \\
u_{kj} = a_{kj} - \sum_{q=1}^{k-1} l_{kq} u_{qj} \\
\quad (j = k, k+1, \cdots, n) \\
l_{ik} = \left(a_{ik} - \sum_{q=1}^{k-1} l_{iq} u_{qk} \right) \Big/ u_{kk} \\
\quad (i = k+1, \cdots, n) \\
\quad (k = 2,3,\cdots,n)
\end{cases}
$$

例3．1

$$\left(\begin{array}{ccc|c} 2 & 1 & 2 & 6 \\ 4 & 5 & 4 & 18 \\ 6 & -3 & 5 & 5 \end{array}\right) \rightarrow \left(\begin{array}{ccc|c} 2 & 1 & 2 & 6 \\ 2 & 3 & 0 & 6 \\ 3 & -2 & -1 & -1 \end{array}\right)$$

所以

$$x_3 = \frac{-1}{-1} = 1, \quad x_2 = (6 - 0x_3)/3 = 2$$

$$x_1 = (6 - 2x_3 - 1x_2)/2 = 1$$

用Matlab实现LU分解

在Matlab程序编辑器中输入：

```
function [L,U]=nalu(a) % a为可逆方阵；L返回单位下三角矩阵；U返回上三角矩阵
n=length(a);
U=zeros(n,n);L=eye(n,n);
U(1,:)=a(1,:);L(2:n,1)=a(2:n,1)/U(1,1);
for k=2:n
    U(k,k:n)=a(k,k:n)-L(k,1:k-1)*U(1:k-1,k:n);
    L(k+1:n,k)=(a(k+1:n,k)-L(k+1:n,1:k-1)*U(1:k-1,k))/U(k,k);
end
```

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

File Edit View Text Debug Breakpoints Web Window Help

Stack:

```
1   function [L,U]=nalu(a) % a为可逆方阵；L返回单位下三角矩阵；U返回上三角矩阵
2   n=length(a);
3   U=zeros(n,n);L=eye(n,n);
4   U(1,:)=a(1,:);L(2:n,1)=a(2:n,1)/U(1,1);
5   for k=2:n
6       U(k,k:n)=a(k,k:n)-L(k,1:k-1)*U(1:k-1,k:n);
7       L(k+1:n,k)=(a(k+1:n,k)-L(k+1:n,1:k-1)*U(1:k-1,k))/U(k,k);
8   end
```

程序运行结果：

Command Window

```
>> a=[1 1 1;-1 3 1;2 -6 1];
>> [L,U]=nalu(a)

L =

     1     0     0
    -1     1     0
     2    -2     1


U =

     1     1     1
     0     4     2
     0     0     3

>>
```

3.3.2. 平方根法（Cholesky）

定理：设A对称正定，则有非奇异下三角阵L，使

$$A = LL^T。$$

分解方法：设

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & & & \\ l_{21} & l_{22} & & \\ \vdots & \vdots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & \cdots & l_{n1} \\ & l_{22} & \cdots & l_{n2} \\ & & \ddots & \vdots \\ & & & l_{nn} \end{bmatrix}$$

其中$a_{ij} = a_{ji}$

由矩阵乘法

(1) 1) $l_{11}^{\ 2} = a_{11} \Rightarrow l_{11} = \sqrt{a_{11}}$ (取正)

2) L第1行 $\times L^T$ 第j列 (j = 2,⋯,n)

$$l_{11}l_{j1} = a_{1j} = a_{j1} \Rightarrow l_{j1} = \frac{a_{j1}}{l_{11}}$$

(2) 1) L第2行 $\times L^T$ 第2列

$$l_{21}^{\ 2} + l_{22}^{\ 2} = a_{22} \Rightarrow l_{22} = \sqrt{a_{22} - l_{21}^{\ 2}}$$

2) L第2行 $\times L^T$ 第j列 (j = 3,4,⋯,n)

$$l_{21}l_{j1} + l_{22}l_{j2} = a_{2j} = a_{j2} \Rightarrow l_{j2} = \frac{a_{j2} - l_{21}l_{j1}}{l_{22}}$$

……

(k)　　可得：　$l_{kk}=\sqrt{a_{kk}-\sum_{q=1}^{k-1}l_{kq}^{\ 2}}$　，

$$l_{jk}=\frac{1}{l_{kk}}\left[a_{jk}-\sum_{q=1}^{k-1}l_{kq}l_{jq}\right]$$

$(j=k+1,\cdots,n)$

练习：习题3-9

优点：1、计算量小;

2、第k行×第k列：　$a_{kk}=\sum_{q=1}^{k} l_{kq}^{2}$

故　$l_{kq}^{2} \le a_{kk} \le \max\limits_{1\le k\le n} a_{kk}$

$$\left|l_{kq}\right| \le \max_{1\le k\le n} \sqrt{a_{kk}}\,.$$

这说明在分解过程中$\left|l_{kq}\right|^{2}$的值不会超过A的最大元,所以说舍入误差的放大受到控制$\longrightarrow$平方根法稳定，不必先主元。

缺点：开方运算。$\Rightarrow$改进平方根法。

什么是改进平方根法呢？

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & & & \\ l_{21} & l_{22} & & \\ \vdots & \vdots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} u_{11} & u_{21} & \cdots & u_{n1} \\ & u_{22} & \cdots & u_{n2} \\ & & \ddots & \vdots \\ & & & u_{nn} \end{bmatrix}$$

其中$a_{ij} = a_{ji}$

$A = LL^T = QDL^T = QU$,其中D是三角形矩阵。

Q中的元素与U的元素有什么关系？

$$
\begin{cases}
u_{1i} = a_{1i} \quad (i = 1,2,\cdots n) \\
l_{i1} = a_{i1}/u_{11} = u_{1i}/u_{11} \quad (i = 2,3,\cdots,n) \\
u_{ki} = a_{ki} - \sum_{q=1}^{k-1} l_{kq}u_{qi} = a_{ki} - \sum_{q=1}^{k-1} \frac{u_{qk}}{u_{qq}}u_{qi} \\
(i = k, k+1,\cdots,n) \\
l_{ik} = \dfrac{u_{ki}}{u_{kk}} \\
(i = k+1,\cdots,n) \\
\quad (k = 2,3,\cdots,n)
\end{cases}
$$

改进平方根法：1）无需计算平方根；

2）计算量与平方根法相比同阶量。

3.3.3 列主元的三角分解法

设方程组$Ax=b$,对其增广矩阵作LU分解时,为了避免用小u_{kk}作除数,引进量$s_i=a_{ik}-\sum_{q=1}^{k-1}l_{iq}u_{qk}\ (i=k,k+1,\cdots,n)$

于是$u_{kk}=s_k$,比较$|s_i|$的大小,取$\max\limits_{k\le i\le n}|s_i|\ (=|s_t|)$为$u_{kk}$,并交换矩阵的第$t$行与第$k$行,且元素的足码也相应改变。

于是$u_{kk}=s_k\ (s_k$为交换前的$s_t)$,$l_{ik}=s_i/s_k$.此时$|l_{ik}|\le 1$,可以控制舍入误差的增大。

例4 对下面的增广矩阵用列主元三角分解法分解。练习并讲解。

$$\overline{A}=\begin{bmatrix} 12 & -3 & 3 & 15 \\ 18 & -3 & 1 & 15 \\ -1 & 2 & 1 & 6 \end{bmatrix}$$

用Matlab实现追赶法求解线性方程组

在Matlab程序编辑器中输入：

```
function x=pursue(a,b,flag) % a为系数矩阵；b为右端列向量；flag为0则显示过程
if nargin<3,flag=0;end    % nargin为计算输入变量个数
n=length(b); a=[a,b];
for k=1:(n-1)
    a(k+1,k+1)=a(k+1,k+1)-
a(k+1,k)/a(k,k)*a(k,k+1);
    a(k+1,n+1)=a(k+1,n+1)-
a(k+1,k)/a(k,k)*a(k,n+1);
    a(k+1,k)=zeros(1,1);
    if flag==0,a,end
end
```

```
% 回代
x(n)=a(n, n+1)/a(n, n);
for  k=n-1:-1:1
     x(k)=(a(k, n+1)-a(k, k+1)*x(k+1))/a(k, k)
end
```

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

File Edit View Text Debug Breakpoints Web Window Help

Stack

```
1   function x=pursue(a,b,flag) % a为系数矩阵；b为右端列向量；flag为0则显示过程
2   if nargin<3,flag=0;end    % nargin为计算输入变量个数
3   n=length(b); a=[a,b];
4   for k=1:(n-1)
5       a(k+1,k+1)=a(k+1,k+1)-a(k+1,k)/a(k,k)*a(k,k+1);
6       a(k+1,n+1)=a(k+1,n+1)-a(k+1,k)/a(k,k)*a(k,n+1);
7       a(k+1,k)=zeros(1,1);
8       if flag==0,a,end
9   end
10  % 回代
11  x(n)=a(n,n+1)/a(n,n);
12  for k=n-1:-1:1
13      x(k)=(a(k,n+1)-a(k,k+1)*x(k+1))/a(k,k)
14  end
```

pursue Ln 1 Col 1

程序运行结果：

Command Window

```
>> a=[2 1 0 0 0;5/14 2 9/14 0 0;0 3/5 2 2/5 0;0 0 3/7 2 4/7;0 0 0 1 2];
>> b=[-5.5200;-4.3144;-3.2664;-2.4287;-2.1150];
>> x=pursue(a,b)

a =

    2.0000    1.0000         0         0         0   -5.5200
         0    1.8214    0.6429         0         0   -3.3287
         0    0.6000    2.0000    0.4000         0   -3.2664
         0         0    0.4286    2.0000    0.5714   -2.4287
         0         0         0    1.0000    2.0000   -2.1150


a =

    2.0000    1.0000         0         0         0   -5.5200
         0    1.8214    0.6429         0         0   -3.3287
         0         0    1.7882    0.4000         0   -2.1699
         0         0    0.4286    2.0000    0.5714   -2.4287
         0         0         0    1.0000    2.0000   -2.1150
```

Command Window

```
a =

    2.0000    1.0000         0         0         0   -5.5200
         0    1.8214    0.6429         0         0   -3.3287
         0         0    1.7882    0.4000         0   -2.1699
         0         0         0    1.9041    0.5714   -1.9087
         0         0         0    1.0000    2.0000   -2.1150

a =

    2.0000    1.0000         0         0         0   -5.5200
         0    1.8214    0.6429         0         0   -3.3287
         0         0    1.7882    0.4000         0   -2.1699
         0         0         0    1.9041    0.5714   -1.9087
         0         0         0         0    1.6999   -1.1126
```

Command Window

```
x =

         0         0         0   -0.8060   -0.6545


x =

         0         0   -1.0331   -0.8060   -0.6545


x =

         0   -1.4629   -1.0331   -0.8060   -0.6545


x =

   -2.0286   -1.4629   -1.0331   -0.8060   -0.6545


x =

   -2.0286   -1.4629   -1.0331   -0.8060   -0.6545

>>
```

用Matlab实现改进平方根法求解线性方程组

在Matlab程序编辑器中输入：

```
function x=squareroot(a,b,flag)
if nargin<3,flag=0;end
n=length(b);a=[a,b];
for k=1:(n-1)
    a(k+1,1)=a(k+1,1)/a(1,1);
end
 a
for k=1:(n-2)
    for j=k+1:n+1
        a(k+1, j)=a(k+1, j)-a(1:k, j)*a(k+1, 1:k);
    end
    for l=k+2:n
        a(l,k+1)=(a(l,k+1)-
a(1:k,k+1)*a(l,1:k))/a(k+1,k+1);
```

```
end
    a
end
for  i=1:(n-1)
  a(n,n)=a(n,n)-a(i,n)*a(n,i);
  a(n,n+1)=a(n,n+1)-a(i,n+1)*a(n,i);
end
a
x=zeros(n,1);
x(n)=a(n,n+1)/a(n,n);
for  k=n-1:-1:1
     x(k)=(a(k,n+1)-
a(k,(k+1):n)*x((k+1):n))/a(k,k);
end
```

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

File Edit View Text Debug Breakpoints Web Window Help

Stack: Base

```
function x=squareroot(a,b,flag)
if nargin<3,flag=0;end
n=length(b);a=[a,b];
for k=1:(n-1)
    a(k+1,1)=a(k+1,1)/a(1,1);
end
 a
for k=1:(n-2)
    for j=k+1:n+1
        a(k+1,j)=a(k+1,j)-a(1:k,j)*a(k+1,1:k);
    end
    for l=k+2:n
        a(l,k+1)=(a(l,k+1)-a(1:k,k+1)*a(l,1:k))/a(k+1,k+1);
    end
    a
end
for i=1:(n-1)
  a(n,n)=a(n,n)-a(i,n)*a(n,i);
  a(n,n+1)=a(n,n+1)-a(i,n+1)*a(n,i);
end
a
x=zeros(n,1);
x(n)=a(n,n+1)/a(n,n);
for k=n-1:-1:1
    x(k)=(a(k,n+1)-a(k,(k+1):n)*x((k+1):n))/a(k,k);
end
```

程序运行结果：

```
>> a=[4 -2 -4;-2 17 10;-4 10 9];
>> b=[10;3;-7];
>> x=squareroot(a,b)

a =

    4.0000   -2.0000   -4.0000   10.0000
   -0.5000   17.0000   10.0000    3.0000
   -1.0000   10.0000    9.0000   -7.0000


a =

    4.0000   -2.0000   -4.0000   10.0000
   -0.5000   16.0000    8.0000    8.0000
   -1.0000    0.5000    9.0000   -7.0000


a =

    4.0000   -2.0000   -4.0000   10.0000
   -0.5000   16.0000    8.0000    8.0000
   -1.0000    0.5000    1.0000   -1.0000


x =

     2
     1
    -1
```

3.4 向量范数和矩阵范数

定义：设$f(x)=\|x\|$是定义在R^n上的实函数，如果它满足以下3个条件：

(1) 对任意$x\in R^n, \|x\|\geq 0; \|x\|=0$当且仅当$x=0$（非负性);

(2) 对任意实常数c和任意$x\in R^n, \|cx\|=|c|\cdot\|x\|$（奇次性）；

(3) 对任意$x, y\in R^n$有$\|x+y\|\leq\|x\|+\|y\|$（三角不等式性）。

则称$\|\bullet\|$为上R^n的向量范数。

最常用的是如下三种范数：

向量的1-范数：$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|$；

向量的2-范数：$\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$

向量的∞-范数：$\|\mathbf{x}\|_\infty = \max_{1 \le i \le n} |x_i|$.

矩阵范数：设$A \in R^{n\times n}$,$\|\cdot\|$是R^n上的任一向量范数，称$\max\limits_{\|x\|=1}\|AX\|$为$A$的矩阵范数。记作$\|A\|$，

即$\|A\|=\max\limits_{\|x\|=1}\|AX\|$.

最常用的是如下三种矩阵范数：

$\|A\|_1=\max\limits_{\|x\|_1=1}\|AX\|_1$;

$\|A\|_2=\max\limits_{\|x\|_2=1}\|AX\|_2$;

$\|A\|_\infty=\max\limits_{\|x\|_\infty=1}\|AX\|_\infty$.

可以证明：

$$\| A \|_1 = \max_{1 \le j \le n} \sum_{i=1}^{n} | a_{ij} |; \| A \|_\infty = \max_{1 \le i \le n} \sum_{j=1}^{n} | a_{ij} |$$

$\| A \|_2 = \sqrt{\rho(A^{\mathrm{T}} A)}$(其中为矩阵的谱半径，即

$\rho(B) = \max\{ | \lambda | \, | \lambda I - B | = 0 \}$.

例3.2：设$A = \begin{pmatrix} 1 & -3 \\ -1 & 2 \end{pmatrix}$,求$\| A \|_1, \| A \|_\infty$及$\| A \|_2$.

练习：习题3-12

性质：（1）对任意$A \in R^{n\times n}, \| A \| \geq 0; \| A \| = 0$
当且仅当$A = 0;$

（2）对任意实常数c和任意$A \in R^{n\times n}, \| cA \| = | c | \cdot \| A \|$；

(3)对任意$A, B \in R^{n\times n}$有$\| A + B \| \leq \| A \| + \| B \|;$

(4)对任意向量$x \in R^n, A \in R^{n\times n}$有$\| Ax \| \leq \| A \| \cdot \| x \|$

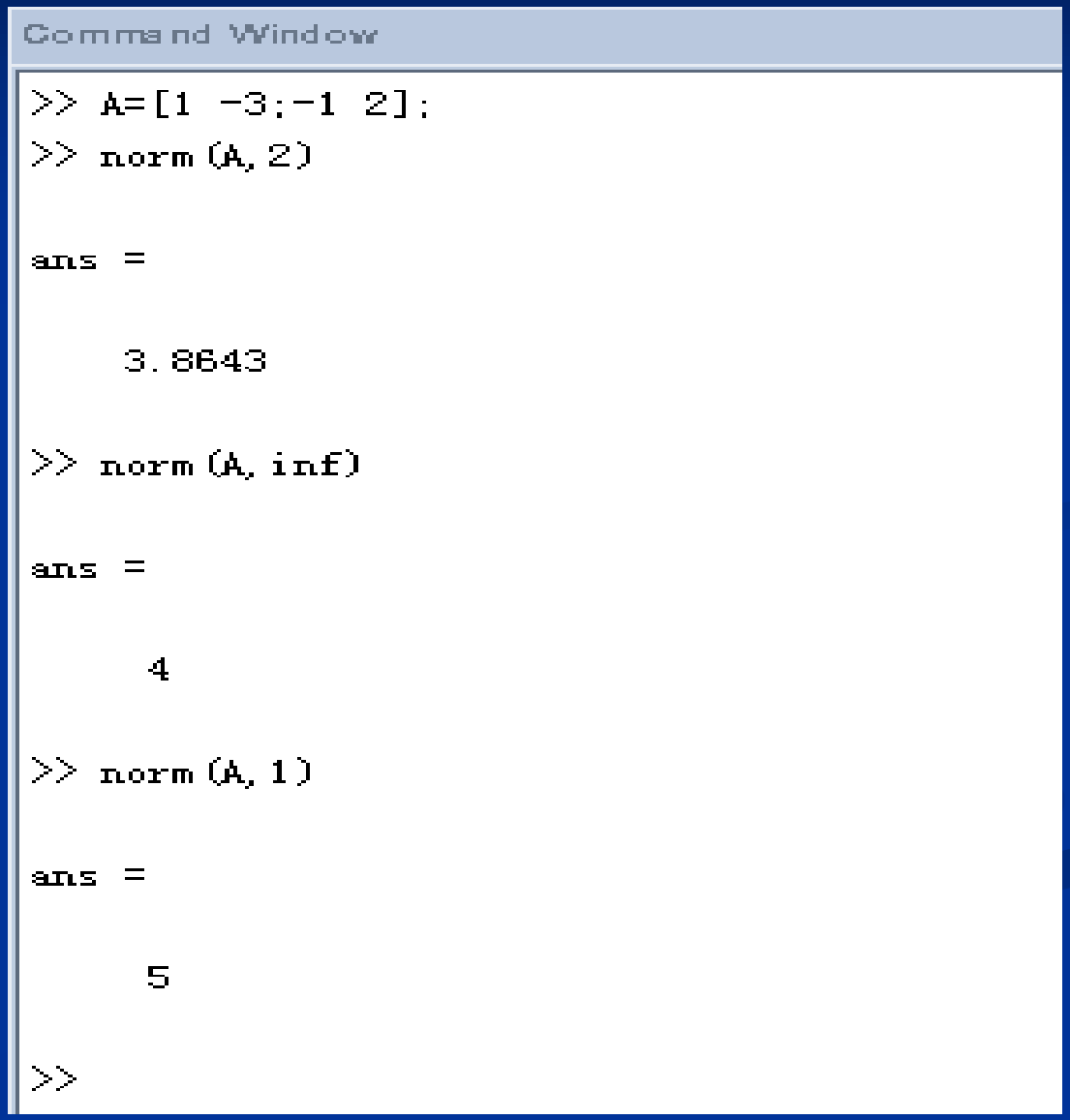
（5）对任意$A, B \in R^{n\times n}$有$\| AB \| \leq \| A \| \cdot \| B \|$.

定理3.1：设$A \in R^{n\times n}, \| \cdot \|$为任一矩阵范数，
则$\rho(A) \leq \| A \|$.
即矩阵的任一范数均可作为矩阵特征值的上界。

在Matlab中可以用命令 norm(A,p)来求范数。

p为(2,1,inf)分别对应：2-范数，1-范数和∞-范数。

举例如右图：



3.5 解线性方程组的迭代法

3.5.1 迭代法及收敛性

1. 迭代法建立. 考虑 $Ax = b$

$$Ax = b \Leftrightarrow x = Bx + g \quad \text{（矩阵B不唯一）}$$

对应写出

$$\begin{cases} x^{(k+1)} = Bx^{(k)} + g \\ (k = 0,1,2,\cdots) \\ \text{取定初始向量} x^{(0)} \end{cases}$$

迭代格式

产生向量序列 $x^{(1)}, x^{(2)}, \cdots, x^{(k)}, x^{(k+1)}, \cdots$

若序列收敛, 记 $\lim\limits_{k\to\infty} x^{(k+1)} = \bar{x}$

两端取极限有：

$$\bar{x} = B\bar{x} + g\,,$$

上式说明：$\bar{x}$ 是解向量x，从而当k充分大时

$$x^{(k+1)} \approx \text{解向量 } x$$

——简单迭代法，B叫迭代矩阵。

注意：迭代阵B不唯一，影响收敛性。

例3.3 用迭代法解方程组

$$\begin{cases} x_1 + 0.5x_2 = 0.5 \\ 0.5x_1 + x_2 = -0.5 \end{cases}$$

练习，给出不同迭代格式。

定理3.2(充分条件判别法)给定方程组，如果$\| B \| < 1$,则(1)方程组有唯一解x^*.

(2)对任意初始向量$x^{(0)} \in R^n$,迭代格式收敛于x^*，且有

$$\| x^{(k+1)} - x^* \| \leq \| B \| \cdot \| x^{(k)} - x^* \| \quad k = 0,1,2,\cdots$$

$$(3) \| x^{(k)} - x^* \| \leq \frac{\| B \|}{1 - \| B \|} \cdot \| x^{(k)} - x^{(k-1)} \| \quad k = 1,2,\cdots$$

$$(4) \| x^{(k)} - x^* \| \leq \frac{\| B \|^k}{1 - \| B \|} \cdot \| x^{(1)} - x^{(0)} \| \quad k = 1,2,\cdots$$

定理3.4(充要条件）简单迭代法 $x^{(k+1)} = Bx^{(k)} + g$ 对任意初始向量$x^{(0)}$都收敛 $\Leftrightarrow \rho(B) < 1$.

注意：$\rho(B) \geq 1$ 时不能说对任意 $x^{(0)}$都不收敛.

3.5.2. Jacobi迭代法

$$
\begin{cases}
x_1^{(k+1)} = \dfrac{1}{a_{11}}\left[b_1 - 0x_1^{(k)} - a_{12}x_2^{(k)} - \cdots - a_{1n}x_n^{(k)}\right] \\
x_2^{(k+1)} = \dfrac{1}{a_{22}}\left[b_2 - a_{21}x_1^{(k)} - 0x_2^{(k)} - \cdots - a_{2n}x_n^{(k)}\right] \\
\cdots \quad \cdots \quad \cdots \\
x_n^{(k+1)} = \dfrac{1}{a_{nn}}\left[b_n - a_{n1}x_1^{(k)} - a_{n2}x_2^{(k)} - \cdots - 0x_n^{(k)}\right]
\end{cases}
$$

(k=0, 1, 2, …)

练习例3.7

$$\begin{cases} 10x_1 - 2x_2 - x_3 = 3 \\ -2x_1 + 10x_2 - x_3 = 15 \\ -x_1 - 2x_2 + 5x_3 = 10 \end{cases}$$

矩阵形式为：

$$\begin{pmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \\ \vdots \\ x_n^{(k+1)} \end{pmatrix} = \begin{pmatrix} 0 & -\frac{a_{12}}{a_{11}} & \cdots & -\frac{a_{1n}}{a_{11}} \\ -\frac{a_{21}}{a_{22}} & 0 & \cdots & -\frac{a_{2n}}{a_{22}} \\ \cdots & \cdots & \cdots & \cdots \\ -\frac{a_{n1}}{a_{nn}} & -\frac{a_{n2}}{a_{nn}} & \cdots & 0 \end{pmatrix} \begin{pmatrix} x_1^{(k)} \\ x_2^{(k)} \\ \vdots \\ x_n^{(k)} \end{pmatrix} + \begin{pmatrix} \frac{b_1}{a_{11}} \\ \frac{b_2}{a_{22}} \\ \vdots \\ \frac{b_n}{a_{nn}} \end{pmatrix}$$

简记为$x^{(k+1)} = Jx^{(k)} + g$

收敛性：*Jacobi*迭代法对任意$x^{(0)}$收敛

$\Leftrightarrow \rho(J) < 1$

3.5.2. *Gauss – seidel*迭代法：

设方程组*Jacobi*迭代格式为

$$\begin{cases} x_1^{(k+1)} = b_{11} x_1^{(k)} + b_{12} x_2^{(k)} + \cdots + b_{1n} x_n^{(k)} + g_1 \\ x_2^{(k+1)} = b_{21} x_1^{(k)} + b_{22} x_2^{(k)} + \cdots + b_{2n} x_n^{(k)} + g_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ x_n^{(k+1)} = b_{n1} x_1^{(k)} + b_{n2} x_2^{(k)} + \cdots + b_{nn} x_n^{(k)} + g_n \end{cases}$$

称如下迭代法

$$\begin{cases} x_1^{(k+1)} = b_{11}x_1^{(k)} + b_{12}x_2^{(k)} + \cdots + b_{1n}x_n^{(k)} + g_1 \\ x_2^{(k+1)} = b_{21}x_1^{(k+1)} + b_{22}x_2^{(k)} + \cdots + b_{2n}x_n^{(k)} + g_2 \\ \cdots\cdots\cdots \\ x_n^{(k+1)} = b_{n1}x_1^{(k+1)} + b_{n2}x_2^{(k+1)} + \cdots + b_{nn}x_n^{(k)} + g_n \end{cases}$$

为高斯–赛德尔迭代法。

Gauss – Seidel 迭代法的收敛性

*Gauss – Seidel*迭代法对任意$x^{(0)}$收敛的充要条件是

$\rho(S)<1$.

定理3.4(充分条件判别法)对于给定的线性方程组：

(1)若系数矩阵*A*为严格对角占优，则*Jacobi*迭代法和*Gauss – seidel*迭代法均收敛.

(2)若*A*为对称正定阵，则*Gauss – seidel*迭代法收敛.

例3.4 分别用Jacobi迭代法和Gauss-Seidel迭代法求解。

$$\begin{bmatrix} 10 & -2 & -1 \\ -2 & 10 & -1 \\ -1 & -2 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ 15 \\ 10 \end{bmatrix}$$

取初始向量 $x^{(0)} = (0,0,0)^T$ ，要求 $\max\limits_{1 \le i \le 3} | x_i^{(k+1)} - x_i^{(k)} | \le 10^{-3}$ 时迭代终止。

解：因为系数矩阵严格对角占优，故Jacobi迭代法和Gauss-Seidel方法都收敛。

首先，Jacobi迭代格式为

$$\begin{cases} x_1^{(k+1)} = 0.2x_2^{(k)} + 0.1x_3^{(k)} + 0.3 \\ x_2^{(k+1)} = 0.2x_1^{(k)} + 0.1x_3^{(k)} + 1.5 \\ x_3^{(k+1)} = 0.2x_1^{(k)} + 0.4x_2^{(k)} + 2 \end{cases} \quad (k = 0,1,2,\cdots)$$

计算结果略.

其次，*Gauss – seidel*迭代法为

$$\begin{cases} x_1^{(k+1)} = 0.2x_2^{(k)} + 0.1x_3^{(k)} + 0.3 \\ x_2^{(k+1)} = 0.2x_1^{(k+1)} + 0.1x_3^{(k)} + 1.5 \\ x_3^{(k+1)} = 0.2x_1^{(k+1)} + 0.4x_2^{(k+1)} + 2 \end{cases} \quad (k = 0,1,2,\cdots)$$

计算结果可列表如下

k	$x^{(k)}=(x_1^{(k)},x_2^{(k)},x_3^{(k)})^T$
0	$(0 \quad 0 \quad 0)^T$
1	$(0.30000 \quad 1.56000 \quad 2.68400)^T$
2	$(0.88040 \quad 1.94448 \quad 2.95387)^T$
3	$(0.98428 \quad 1.99224 \quad 2.99375)^T$
4	$(0.99782 \quad 1.99894 \quad 2.99914)^T$
5	$(0.99970 \quad 1.99985 \quad 2.99988)^T$
6	$(0.99996 \quad 1.99998 \quad 2.99998)^T$

因为$|x_i^{(6)}-x_i^{(5)}|\le 10^{-3}\ (i=1,2,3)$,故$x^{(6)}$为近似解，即

$x_1\approx 0.99996$ ， $x_2\approx 1.99998$ ， $x_3\approx 2.99998$

注意：Gauss-Seidel方法未必一定比Jacobi方法好。参看习题3-16。

- 对比课本76页，79页结论，观察课本上求解过程，有什么疑问？

- 为了避免分数计算带来的误差等不便，对特征方程进行了转换。

若记$\tilde{\mathrm{L}}=\begin{bmatrix} 0 & & & & \\ a_{21} & 0 & & & \\ a_{31} & a_{32} & 0 & & \\ \vdots & \vdots & \vdots & 0 & \\ a_{n1} & a_{n2} & \cdots & a_{nn-1} & 0 \end{bmatrix}$ $\mathrm{D}=\begin{bmatrix} a_{11} & & & \\ & a_{22} & & \\ & & \ddots & \\ & & & a_{nn} \end{bmatrix}$

$\tilde{U}=\begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ & 0 & \cdots & a_{2n} \\ & & \vdots & \vdots \\ & & 0 & a_{n-1n} \\ & & & 0 \end{bmatrix}$

- 三个矩阵与A之间有什么关系？

- 转换过程推导（Jacobi和Gauss-seidel雷同）

已知$Ax=b$，$A=\tilde{L}+D+\tilde{U}$,

则雅克比迭代格式的矩阵表示形式为：

$$x^{(k+1)}=-D^{-1}(\tilde{L}+\tilde{U})x^{(k)}+D^{-1}b$$

其迭代矩阵$J=-D^{-1}(\tilde{L}+\tilde{U})$,

特征方程为$|\lambda I-J|=0$

$$|\lambda I+D^{-1}(\tilde{L}+\tilde{U})|=0,|D^{-1}(\lambda D+\tilde{L}+\tilde{U})|=0$$

因为$|D^{-1}|\neq 0$，所以$|\tilde{L}+\lambda D+\tilde{U}|=0.$

即等式左侧为直接在系数矩阵的对角线元素乘以λ后所得新矩阵的行列式。

补充：逐次超松弛迭代法(SOR法)该方法了解即可，无需掌握

$$x_i^{(k+1)} = (1-\omega)\, x_i^{(k)} +$$

$$\omega \cdot \frac{1}{a_{ii}} \left[b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right]$$

$$(i = 1,2,\cdots,n) \quad (3.11)$$

ω — 松弛因子，ω =1 即Seidel方法(3.10)，

(3.11)是一种加权平均。

SOR方法的收敛性如下（不加证明）：

(1) SOR方法对任意 $x^{(0)}$ 都收敛的必要条件是：

$0 < \omega < 2$

(2) 若系数矩阵A对称正定，则 $0 < \omega < 2$ 时SOR方法求解 $Ax = b$ 对任意 $x^{(0)}$ 收敛；

(3) 若系数矩阵A按行（或按列）严格对角占优，则 $0 < \omega \leq 1$ 时SOR方法对任意 $x^{(0)}$ 收敛.

最佳松弛因子 ω_{opt} 选取问题。

基本要求：

1. 掌握Gauss消去法及列主元Gauss消去法。
2. 掌握矩阵的直接三角分解法的过程；
3. 熟悉平方根法的计算过程及其优缺点；
4. 掌握简单迭代法及其收敛条件的使用；
5. 掌握Jacobi迭代法及其相应的Gauss-seidel迭代法的计算公式以及它们的收敛条件。

本章作业：见课堂在线

第四章　插值法

4.1 问题的提出

4.1.1. 问题：实际中，连续 $y = f(x)$存在，但未知，

只知离散数据 $y_i = f(x_i)$ $(i = 0,1,2,\ldots,n)$

希 望：用简单函数$P(x)$——插值函数

近似代替$f(x)$——被插函数，

使$P(x_i) = f(x_i) = y_i$——插值条件，(4.1)

———————插值法

x_i叫插值结点，$R(x) = f(x) - P(x)$叫截断误差

通常取$P(x)$为三角函数$\rightarrow$三角插值或

取$P(x)$为多项式函数$\rightarrow$代数插值(多项式插值)

即若$P(x)=P_n(x)=a_0+a_1x+a_2x^2+\cdots+a_nx^n$

是次数不超过n的多项式，则称$P_n(x)$为n次插值多项式，本章主要讨论$P(x)$为多项式时的情况。

特别，当$n=1$时，为通过两点的一次插值多项式，称相应的插值问题为线性插值；当$n=2$时，为通过三点的一次插值多项式，称相应的插值问题为抛物插值。

4.1.2. 插值多项式的存在唯一性

定理4.1: 在 $n+1$ 个 互异 节点 x_k 处满足插值条件

$$p_n(x_k) = f(x_k) \quad (k = 0,1,2,\ldots,n)$$

的次数不超过 n 的多项式 $p_n(x)$ 存在且唯一.

证： 设 $\boldsymbol{p_n(x) = a_0 + a_1 x + \ldots + a_n x^n}$.

代入插值条件得:

$$\begin{aligned} a_0 + a_1 x_0 + \ldots + a_n x_0^n &= f(x_0) \\ a_0 + a_1 x_1 + \ldots + a_n x_1^n &= f(x_1) \quad (4.2) \\ &\cdots\cdots\cdots\cdots \\ a_0 + a_1 x_n + \ldots + a_n x_n^n &= f(x_n) \end{aligned}$$

系数行列式：$\begin{vmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \cdots & x_n^n \end{vmatrix} = \prod_{0 \le j < i \le n} (x_i - x_j) \neq 0$

故由*Crammer*法则知, 该方程组解存在唯一, 即多项式(系数)存在唯一.

注意： 如果不限制多项式的次数, 插值多项式并不唯一.事实上, 设α是任意实数, 若$P(x)$是满足(4.1)式的一个插值多项式, 则$P(x) + \alpha \prod_{i=0}^{n} (x - x_i)$也是满足(4.1)式的一个插值多项式, 而且当$\alpha$不为零时次数严格大于$n$.

4.2 Lagrange插值公式

4. 2. 1. 基本插值多项式

当$n=1$时,(4.2)式为$\begin{cases} a_0+a_1x_0=y_0 \\ a_0+a_1x_1=y_1 \end{cases}$,

解之，得$a_0=\dfrac{x_1y_0-x_0y_1}{x_1-x_0}, a_1=\dfrac{y_1-y_0}{x_1-x_0}$

因而$P_1(x)=a_0+a_1x$

$$=\frac{x_1y_0-x_0y_1}{x_1-x_0}+\frac{y_1-y_0}{x_1-x_0}x$$

$$=y_0+\frac{y_1-y_0}{x_1-x_0}(x-x_0)$$

若令$l_0(x)=\dfrac{x-x_1}{x_0-x_1}, l_1(x)=\dfrac{x-x_0}{x_1-x_0}$,

则有$P_1(x)=y_0l_0(x)+y_1l_1(x)$

而$l_0(x)$和$l_1(x)$可以看作满足插值条件

$l_0(x_0)=1,\ \ l_0(x_1)=0$及$l_1(x_0)=0,\ \ l_1(x_1)=1$

的一次插值多项式。这2个特殊的插值多项式称作一次插值的基本插值多项式。

可以看出一次插值多项式可以通过基本插值多项式的线性组合得到，且其系数恰为所给数据y_0和y_1。

定义4.1：

若$n+1$个n次多项式$l_k(x)$ $(k=0,1,2,\ldots,n)$

在$n+1$个结点x_i上满足

$$l_k(x_i)=\begin{cases}1 & \text{当}i=k\\ 0 & \text{当}i\neq k\end{cases},$$

则它为基本插值多项式，

也称为插值基函数。

怎么求得n次插值多项式？

首先，$x_0,\cdots,\ x_{k-1},\ x_{k+1},\cdots x_n$为$n$次多项式$l_k(x)$的零点，所以$l_k(x)$含有如下$n$个一次因子：

$$x-x_0,\cdots,\ x-x_{k-1},\ x-x_{k+1},\cdots x-x_n$$

于是$l_k(x)$可以写成

$$l_k(x)=A_k(x-x_0)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_n)$$

其中A_k为待定系数。由$l_k(x_k)=1$,得到

$$A_k=\frac{1}{\prod\limits_{\substack{i=0\\ i\neq k}}^{n}(x_k-x_i)}$$,从而$$l_k(x)=\prod_{\substack{i=0\\ i\neq k}}^{n}\frac{x-x_i}{x_k-x_i}.$$

$l_k(x)$称为n次插值问题的(第k个)基本插值多项式。

4. 2. 2. Lagrange插值多项式

满足插值条件 $L_n(x_k)=y_k\ (\mathrm{k}=0,1,2,\cdots,\mathrm{n})$

的次数不超过n的多项式为:

$$L_n(x)=l_0(x)y_0+l_1(x)y_1+\cdots+l_n(x)y_n$$

这是因为 $L_n(x_k)=l_k(x_k)y_k=y_k$

$$(\mathrm{k}=0,1,2,\cdots,\mathrm{n})$$

注意: $L_n(x)\in P^n=\{\mathrm{p}_n(x)\,|\,\mathrm{p}_n$为次数$\le n$的多项式$\}$,

故可表为基函数 $l_0,l_1,\cdots,l_n$ 的线性组合。

4.2.3. 插值余项

通过$n+1$个节点的n次插值多项式，在节点处有

$L_n(x_j)=f(x_j)\quad j=0,1,\cdots,n$

而在其他点上，均是$f(x)$的近似值.记

$R_n(x)=f(x)-L_n(x)$称$R_n(x)$为插值多项式的余项。

定理4.2设$f^{(n)}(x)\in C[a,b]$，$f^{(n+1)}(x)$在(a,b)上存在，节点$a\le x_0\le\cdots\le x_n\le b$，$L_n(x)$是满足插值条件$L_n(x_j)=f(x_j)$$(j=0,1,\cdots,n)$的$n$次插值多项式，则对任意$x\in(a,b)$，

插值余项$R_n(x)=f(x)-L_n(x)=\dfrac{f^{(n+1)}(\xi)}{(n+1)!}\omega_{n+1}(x)$。

其中$\omega_{n+1}=\prod_{i=0}^{n}(x-x_i);\xi\in[a,b]$而且依赖于$x$的位置。

证：因$R(x_i)=0\quad(\mathrm{i}=0,1,\cdots,\mathrm{n})$，故有形式

$$R(x)=k(x)(x-x_0)\cdots(x-x_n)\qquad(4.3)$$

引入 $\varphi(t)=f(t)-L_n(t)-k(x)(t-x_0)(t-x_1)\cdots(t-x_n)$,

显然在$x_0,x_1,\cdots,x_n,x$处为零。

故由*Rolle*定理(反复使用)

有$\xi\in(x_0,x_n)$,使$\varphi^{(n+1)}(\xi)=0$,

即$f^{(n+1)}(\xi)-k(x)(n+1)!=0$

故$k(x)=\dfrac{f^{(n+1)}(\xi)}{(n+1)!}$，代入（4.3）即得证。

说明:(1) 当$f(x)$次数不超过n时，$R_n(x)=0.$

(2) 余项表达式只有在$f(x)$的$n+1$阶导数存在时才能使用，一般利用$M_{n+1}=\max\limits_{x_0\le x\le x_n}\left|f^{(n+1)}(x)\right|$求误差限，即

$$\left|\mathrm{R}(x)\right|\le\frac{M_{n+1}}{(n+1)!}\left|\omega_{n+1}(x)\right|。$$

例1 **已知特殊角**$30^o,45^o,60^o$的正弦函数值为$\frac{1}{2},\frac{\sqrt{2}}{2}$及$\frac{\sqrt{3}}{2}$,用一次插值多项式、二次插值多项式近似$\sin x$,并用近似式求出$\sin 50^o$.

解：若取30^o和45^o作为节点作一次插值，得

$$L_1(x)=\frac{x-45}{30-45}\frac{1}{2}+\frac{x-30}{45-30}\frac{\sqrt{2}}{2}$$

则$$L_1(50)=\frac{50-45}{30-45}\frac{1}{2}+\frac{50-30}{45-30}\frac{\sqrt{2}}{2}=0.77614\approx\sin 50^o$$

若取60^o和45^o作为节点作一次插值，得

$$\tilde{L}_1(x)=\frac{x-60}{45-60}\frac{\sqrt{2}}{2}+\frac{x-45}{60-45}\frac{\sqrt{3}}{2}$$

则$$\tilde{L}_1(50)=\frac{50-60}{45-60}\frac{\sqrt{2}}{2}+\frac{50-45}{60-45}\frac{\sqrt{3}}{2}=0.76008\approx\sin 50^o$$

取$30^o,45^o$和60^o为节点，作二次插值，得

（自己练习）

误差估计（内插比外推精度高）：

先求线性插值的误差：$f(x)=\sin x,\ f'(x)=\cos x,$

$f''(x)=-\sin x$.并把度化为弧度，得

$$\sin 50^o - L_1(50) = \frac{1}{2}(-\sin\xi)(\frac{\pi}{180})^2(50-30)(50-45)$$

$(30^o < \xi < 50^o)$,所以

$$|\sin 50^o - L_1(50)| \le \frac{1}{2}\times\frac{\sqrt{3}}{2}\times(\frac{\pi}{180})^2\times 20\times 5 = 0.013190$$

同理，$\sin 50^o - \tilde{L}_1(50) = \frac{1}{2}(-\sin\xi)(\frac{\pi}{180})^2(50-45)(50-60)$

$(45^o < \xi < 60^o)$,所以

$$|\sin 50^o - \tilde{L}_1(50)| \le \frac{1}{2}\times\frac{\sqrt{3}}{2}\times(\frac{\pi}{180})^2\times 5\times 10 = 0.006595$$

误差估计（二次插值比一次插值精度高）：

再求二次插值的误差：

$$\sin 50^o - L_2(50) = \frac{1}{3!}(-\cos\xi)(\frac{\pi}{180})^2(50-30)(50-45)(50-60)$$

$(30^o < \xi < 60^o)$, 所以

$$|\sin 50^o - L_2(50)| \le \frac{1}{6} \times \frac{\sqrt{3}}{2} \times (\frac{\pi}{180})^3 \times 20 \times 5 \times 10 = 0.000767.$$

事实上，$|\sin 50^o - L_1(50)| = 0.01010;$

$|\sin 50^o - \tilde{L}_1(50)| = 0.00596; |\sin 50^o - L_2(50)| = 0.0006.$

总结:(1)内插法(用45^o和60^o)比外推法(用30^o和45^o)作线性插值精确度高。(2)二次插值要比一次插值精确度高。

用Matlab实现 Lagrange插值：

```
function yy=nalagr(x,y,xx)   % x为节点；y为节点值；xx为插值点；yy为返回值
m=length(x);n=length(y);
if m~=n,error('向量x与y长度必须一致');end
s=0;
for i=1:n
    t=ones(1,length(xx));
    for j=1:n
        if j~=i,
            t=t.*(xx-x(j))/(x(i)-x(j));
        end
    end
    s=s+t*y(i);
end
```

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

File Edit View Text Debug Breakpoints Web Window Help

```
function yy=nalagr(x,y,xx)  % x为节点；y为节点值；xx为插值点；yy为返回值
m=length(x);n=length(y);
if m~=n,error('向量x与y长度必须一致');end
s=0;
for i=1:n
    t=ones(1,length(xx));
    for j=1:n
        if j~=i,
            t=t.*(xx-x(j))/(x(i)-x(j));
        end
    end
    s=s+t*y(i);
end
yy=s;
```

求解例题得

一次插值和

二次插值

如右图所示：

Command Window

```
>> x1=pi*[1/6 1/4];y1=[0.5 0.7071];xx=5*pi/18;
>> yy1=nalagr(x1,y1,xx)

yy1 =

    0.776133333333333

>> x2=pi*[1/6 1/4 1/3];y2=[0.5 0.7071 0.866];
>> yy2=nalagr(x2,y2,xx)

yy2 =

    0.765422222222222

>>
```

4.2.4 一类带导数的插值

要求被插函数与插值多项式在某些节点不仅有相同函数值和而且有相同的导数值。

例1 给定函数值表如下：

x_i	1	2	3
$f(x_i)$	2	4	12
$f'(x_i)$		3	

求一次数不超过3的多项式$H_3(x)$,

使之满足如下插值条件：

$$H_3(i)=f(i)\quad(i=1,2,3)\ ,\quad H_3'(2)=3$$

并写出截断误差$R(x)=f(x)-H_3(x)$的表达式.

解：先求满足插值条件

$p_2(i)=f(i)\quad (i=1,2,3)$ 的二次多项式，得

$$p_2(x)=3x^2-7x+6$$

设所求多项式为

$$H_3(x)=p_2(x)+k(x-1)(x-2)(x-3)$$

由条件 $H_3'(2)=3$，得 $k=2$. 故

$$H_3(x)=2x^3-9x^2+15x-6$$

截断误差为：$R(x)=\dfrac{f^{(4)}(\xi)}{4!}(x-1)(x-2)^2(x-3)$

问题：*Lagrange*插值多项式以取多少节点为宜？

如果增加节点个数，前后两次的数值之间能否建立联系？即当$L_{k-1}(x)$近似代替$f(x)$求出的近似值不能满足要求，从而需要至少增加一个节点时用$L_k(x)$求$f(x)$的函数值的近似值时能否利用已有的$L_i(x)(i \le k-1)$,从而减少运算量?

如果对*Lagrange*插值多项式修正,使得$L_k(x)$和$L_{k-1}(x)$之间建立密切联系,问题可解决。

设$h(x) = L_k(x) - L_{k-1}(x)$,故$h(x)$为次数不高于$k$的多项式，且$x_0, x_1, \cdots, x_{k-1}$均是$h(x)$的零点.因而存在一个常数$a_k$,使得$h(x) = a_k(x-x_0)(x-x_1)\cdots(x-x_{k-1})$

则$L_k(x)=L_{k-1}(x)+h(x)=L_{k-1}(x)+a_k(x-x_0)(x-x_1)\cdots(x-x_{k-1})$

继续该过程，则得

$$L_n(x)=a_0(x)+a_1(x-x_0)+a_2(x-x_0)(x-x_1)+\cdots$$
$$+a_n(x-x_0)(x-x_1)\cdots(x-x_{n-1})$$

求a_k,取$x=x_k$,

则$L_k(x_k)=L_{k-1}(x_k)+a_k(x_k-x_0)(x_k-x_1)\cdots(x_k-x_{k-1})$

$$a_k=\frac{L_k(x_k)-L_{k-1}(x_k)}{(x_k-x_0)(x_k-x_1)\cdots(x_k-x_{k-1})}$$

$$f(x_k)-\sum_{m=0}^{k-1}\prod_{\substack{i=0\\ i\neq m}}^{k-1}\frac{x_k-x_i}{x_m-x_i}f(x_m)$$

$$=\frac{}{\prod_{i=0}^{k-1}(x_k-x_i)}=\sum_{m=0}^{k}\frac{f(x_m)}{\prod_{\substack{i=0\\ i\neq m}}^{k}(x_m-x_i)}$$

4.3 差商、差分和Newton插值公式

4.3.1. 差商及牛顿插值公式

$f[x_0, x_1] = \dfrac{f(x_0) - f(x_1)}{x_0 - x_1}$ 叫 f 在 x_0, x_1 的一阶差商（均差）

$f[x_1, x_2] = \dfrac{f(x_1) - f(x_2)}{x_1 - x_2}$ 叫 f 在 x_1, x_2 的一阶差商

.....

$f[x_0, x_1, x_2] = \dfrac{f[x_0, x_1] - f[x_1, x_2]}{x_0 - x_2}$ 叫 f 在 x_0, x_1, x_2 的二阶差商

.....

4.3 差商、差分和Newton插值公式

4.3.1. 差商及牛顿插值公式

一般

$$f[x_0, x_1, \cdots x_k] = \frac{f[x_0, x_1, \cdots, x_{k-1}] - f[x_1, x_2, \cdots, x_k]}{x_0 - x_k}$$

叫 f 在$x_0, x_1, \cdots, x_k$的k阶差商，是由k-1阶差商定义的。

造差商表(实用)

$$
\begin{array}{lllll}
x_0 & f_0 & & & \\
 & & f[x_0,x_1] & & \\
x_1 & f_1 & & f[x_0,x_1,x_2] & \\
 & & f[x_1,x_2] & & f[x_0,x_1,x_2,x_3] \\
x_2 & f_2 & & f[x_1,x_2,x_3] & \\
 & & f[x_2,x_3] & \cdots\cdots & \cdots\cdots \\
x_3 & f_3 & \cdots & & \\
\vdots & \vdots & & &
\end{array}
$$

Newton插值公式

记 $N_n(x)=f(x_0)+f[x_0,x_1](x-x_0)+f[x_0,x_1,x_2](x-x_0)(x-x_1)$

$$+\cdots+f[x_0,x_1,\cdots,x_n](x-x_0)(x-x_1)\cdots(x-x_{n-1}) \quad (4.4)$$

(n次牛顿插值多项式)

性质1. $f[x_0, x_1, \cdots, x_k] = \sum_{m=0}^{k} \frac{f(x_m)}{\prod_{\substack{i=0 \\ i \neq m}}^{k} (x_m - x_i)}$

即k阶差商可表为函数值 $f(x_0), \cdots, f(x_k)$的线性组合。

性质2.(对称 性) $f[x_o \cdots, x_i, \cdots, x_j, \cdots] = f[x_o \cdots, x_j, \cdots, x_i, \cdots]$.

性质3. $f[x_0, x_1, \cdots, x_k] = \frac{f^{(k)}(\eta)}{k!}$,

$\eta \in (\min\{x_o, x_1, \cdots, x_k\}, \max\{x_o, x_1, \cdots, x_k\})$

且 $\lim_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0} = \lim_{x \to x_0} f[x, x_0] = f'(x_0)$,

故差商是微商的离散形式.

例 2　已知 $f(x)=e^{-x}$ 在 $x=1,2,3$ 的值如下

x	1	2	3
e^{-x}	0.367879441	0.135335283	0.049787068

试用二次 *Lagrange* 插值公式求 $e^{-2.1}$ 的近似值，并进行 误差估计.

解：二次插值多项式为

$$L_2(x)=\frac{(x-2)(x-3)}{(1-2)(1-3)}e^{-1}+\frac{(x-1)(x-3)}{(2-1)(2-3)}e^{-2}+\frac{(x-1)(x-2)}{(3-1)(3-2)}e^{-3}$$

$$e^{-2.1}\approx L_2(2.1)=\frac{(2.1-2)(2.1-3)}{(1-2)(1-3)}e^{-1}+\frac{(2.1-1)(2.1-3)}{(2-1)(2-3)}e^{-2}$$

$$+\frac{(2.1-1)(2.1-2)}{(3-1)(3-2)}e^{-3}=0.120165644$$

因 $\max\limits_{1\le x\le 2}\left|\left(e^{-x}\right)^{(3)}\right|\le e^{-1}$, 故有误差估计

$$\left|R(2.1)\right|=\left|e^{-2.1}-L_2(2.1)\right|\le\frac{e^{-1}}{3!}\left|(2.1-1)(2.1-2)(2.1-3)\right|$$

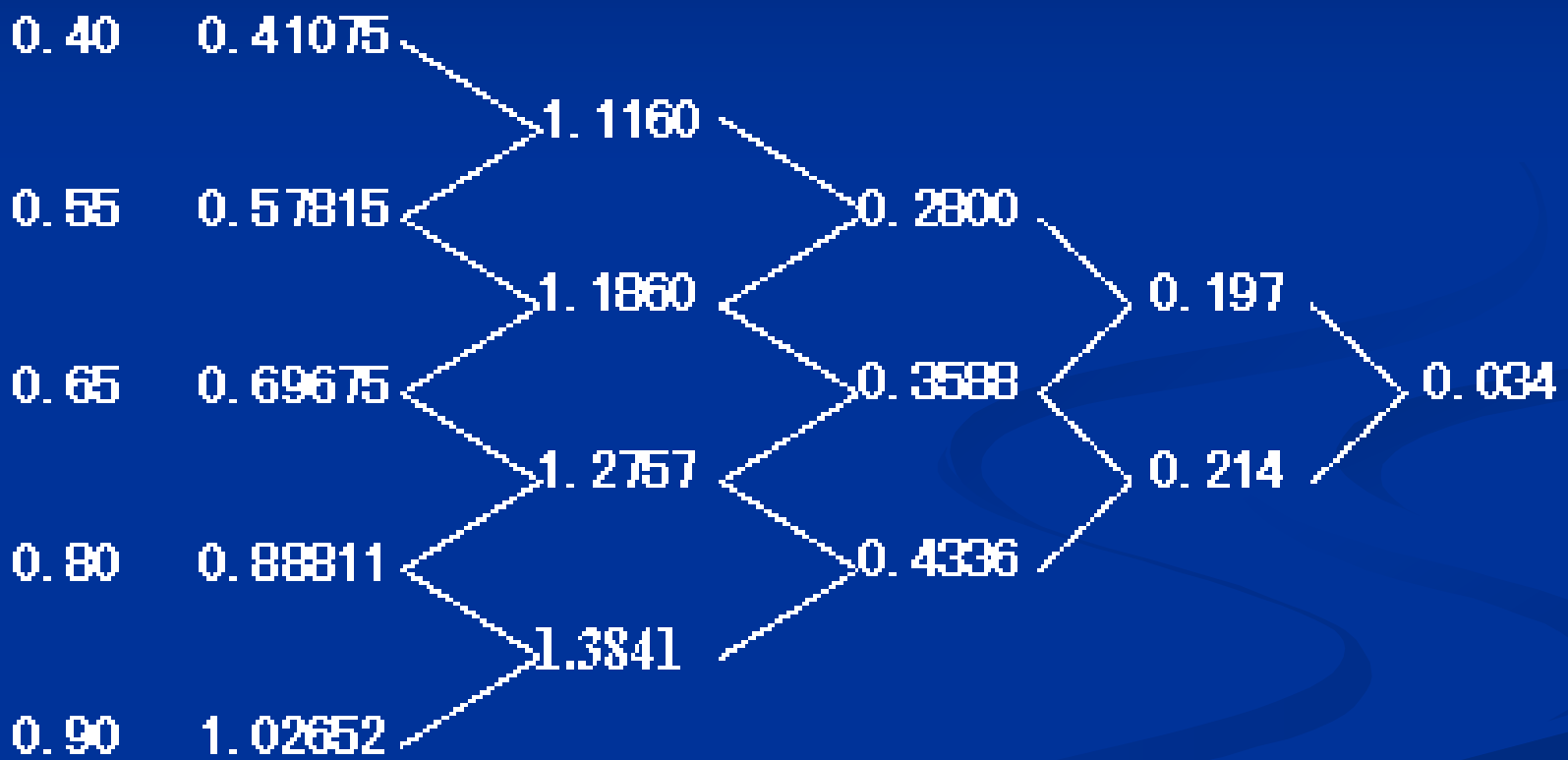
$$\approx 0.00607001$$

例3 给定数值表如下：

x_k	0.40	0.55	0.65	0.80	0.90
$f(x_k)$	0.41075	0.57815	0.69675	0.88811	1.02652

用 *Newton* 插值公式求 f (0.596) 的近似值。

解：先造差商表


0.40
0.41075
1.1160
0.55
0.57815
0.2800
1.1860
0.197
0.65
0.69675
0.3588
0.034
1.2757
0.214
0.80
0.88811
0.4336
1.3841
0.90
1.02652

由Newton公式得四次插值多项式为：

$$N_4 = 0.41075 + 1.1160(x - 0.40)$$
$$+ 0.2800(x - 0.40)(x - 0.55)$$
$$+ 0.197(x - 0.40)(x - 0.55)(x - 0.65)$$
$$+ 0.034(x - 0.40)(x - 0.55)(x - 0.65)(x - 0.80)$$

将$x = 0.596$代入得：

$$f(0.596) \approx N_4(0.596) \approx 0.63192$$

4.3.2 . 差分及等距结点插值公式

牛顿基本插值公式对结点是否等距没有限制. 不过当结点等距时前述牛顿插值公式可进行简化. 首先介绍差分概念.

1)差分

设 $x_i = x_0 + ih \ (i = 0,1,\cdots,n), \quad h = \dfrac{x_n - x_0}{n}$ 叫步长。

设已知函数$f(x)$在等距节点$x_i (i = 0,1,\cdots,n)$上的函数值为$f(x_i) = f_i$,称$f_{i+1} - f_i$为$f(x)$在x_i处以h为步长的一阶向前差分,简称一阶差分，记作Δf_i，即$\Delta f_i = f_{i+1} - f_i$。

类似地，称$\Delta^m f_i = \Delta^{m-1} f_{i+1} - \Delta^{m-1} f_i$为$f(x)$在$x_i$处以$h$为步长的$m$阶向前差分,简称$m$阶差分.

2)性质:

（1） 差分可表为函数值的线性组合 （证略）

(2) $f[x_i, x_{i+1}, \cdots, x_{i+k}] = \dfrac{\Delta^k f_i}{k! h^k}$ (证明用归纳法,略)

(3) $\Delta^k f = h^k f^{(k)}(\xi)$ $\xi \in (x_0, x_k)$

3) 差分表 (实用)

x_i	y_i	一阶差分	二阶差分	三阶差分	…
x_0	y_0				
x_1	y_1	Δy_0			
x_2	y_2	Δy_1	$\Delta^2 y_0$		
x_3	y_3	Δy_2	$\Delta^2 y_1$	$\Delta^3 y_0$	
…	…	…	…	…	…

4）等矩结点插值公式:

设 $x = x_0 + th \quad (0 < t < \mathrm{n}),\ x_i = x_0 + ih,$

将Newton插值公式

$$N_n(x) = f(x_0) + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1)$$
$$+ \cdots + f[x_0, x_1, \cdots, x_n](x - x_0)(x - x_1)\cdots(x - x_{n-1})$$

中的差商用性质(2)换为差分，可整理为如下的（n次）Newton前插公式

$$N_n(x) = N_n(x_0 + th) = f_0 + \frac{\Delta f_0}{1!}t + \frac{\Delta^2 f_0}{2!}t(t-1)$$
$$+ \cdots + \frac{\Delta^n f_0}{n!}t(t-1)\cdots(t-n+1)$$

截断误差可表示为

$$R(x)=R(x_0+th)=\frac{t(t-1)\cdots(t-n)}{(n+1)!}h^{n+1}f^{(n+1)}(\xi)$$

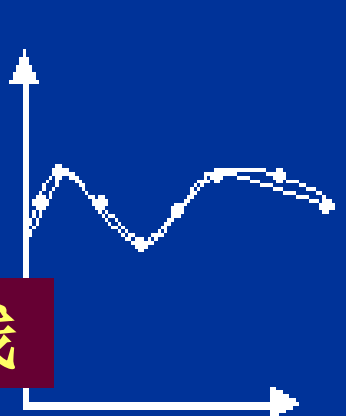
Newton向后插值公式及Bessel 插值公式略。

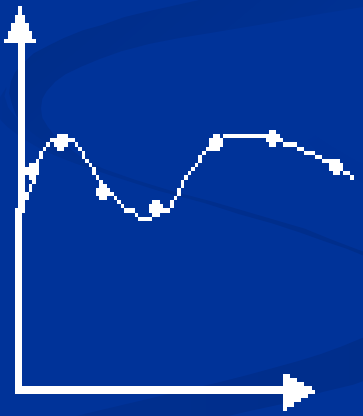
4.4 高次插值的缺点及分段插值

问题：结点增多，多项式次数增高，插值结果逼近精度越好？未必！多结点高次插值往往在局部误差更大——Runge（龙格）现象。

如何解决？---采用分段低次插值

有分段线性，分段二次插值等





折线代替曲线

缺点：分段插值函数只能保证连续性，不能保证光滑性。

4.5 三次样条插值

分段插值可以得到整体连续函数，但在连接点处一般不光滑。实际问题中要求更多，需要分段三次曲线插值，同时连接点处有二阶连续导数。

希望：

分段插值，又想在结点处保持光滑，甚至二阶光滑——三次样条函数。

定义：在[a, b]上取n+1个点 $a = x_0 < x_1 \cdots < x_n = b$

若函数$S(x)$满足：

(1) $S(x_i) = y_i$ ——此时$S(x)$叫插值函数；

(2) 在每个小区间 $[x_{i-1}, x_i]$上是三次多项式；

(3) 在内结点或在整个区间上具二阶连续导数。

则称S(x)为y=f（x）的三次样条插值函数。

构造：导出三对角方程组，再用追 赶法求解。

基本要求：

1. 掌握插值法的含义及其几何意义；
2.掌握Lagrange插值公式及其余项的使用；
3. 会造差商表，并掌握Nenton插值公式的使用；
4. 熟悉差商与导数的关系式。
5.了解三次样条插值。