线性方程组求解

线性方程组求解

矩阵基本概念与性质

- 行列式

矩阵 $\boldsymbol{A} = \{a_{ij}\}$ 的行列式定义为:

$$\boldsymbol{D} = |\boldsymbol{A}| = \det(\boldsymbol{A}) = \sum(-1)^k a_{1k_1} a_{2k_2} \cdots a_{nk_n}$$

命令格式：d=det(A)

- 矩阵的迹

方阵 $\boldsymbol{A}=\{a_{ij}\}$, $i,j=1,2,\cdots,n$ 的迹定义为:

$$\mathrm{tr}(\boldsymbol{A})=\sum_{i=1}^{n}a_{ii}$$

格式： t=trace(A)

- 矩阵的秩

$$\mathrm{rank}(\boldsymbol{A})=r_c=r_r$$

其中 r_c 为列秩， r_r 为行秩

格式：r=rank(A) %用默认的精度求数值秩

r=rank(A, ε) %给定精度下求数值秩

矩阵的秩也表示该矩阵中行列式不等于0的子式的最大阶次。

可证行秩和列秩（线性无关的）应相等。

- 矩阵范数

函数$\rho(\boldsymbol{x})$ 为 $\boldsymbol{x}$ 向量的范数的条件：

(1) $\rho(\boldsymbol{x}) \geq 0$ 且 $\rho(\boldsymbol{x}) = 0$ 的充要条件是 $\boldsymbol{x} = 0$

(2) $\rho(a\boldsymbol{x}) = |a|\rho(\boldsymbol{x})$, a 为任意标量

(3) 对向量 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 有 $\rho(\boldsymbol{x} + \boldsymbol{y}) \leq \rho(\boldsymbol{x}) + \rho(\boldsymbol{y})$

此式满足范数的三个条件：

$$\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^{n} | x_i |^p \right)^{1/p}, \quad p = 1, 2, \cdots,$$

且 $\|\boldsymbol{x}\|_\infty = \max_{1 \leq i \leq n} | x_i |$

$\|\boldsymbol{x}\|_p$ 为向量范数的记号

- 矩阵的范数定义：

对于任意的非零向量 $\boldsymbol{x}$，矩阵 $\boldsymbol{A}$ 的范数为

$$\|\boldsymbol{A}\| = \sup_{\boldsymbol{x}\neq 0} \frac{\|\boldsymbol{A}\boldsymbol{x}\|}{\|\boldsymbol{x}\|}$$

常用范数：

- $$\|\boldsymbol{A}\|_1 = \max_{1\le j\le n} \sum_{i=1}^{n} |a_{ij}|,$$

$$\|\boldsymbol{A}\|_2 = \sqrt{s_{\max}(\boldsymbol{A}^T\boldsymbol{A})}$$

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

格式：

N=norm(A)　　％求解默认的2范数

N=norm(A，选项)　　％选项可为1,2,inf等

- 例：求一向量、矩阵的范数

```
>> a=[16 2 3 13];
[norm(a), norm(a,2), norm(a,1), norm(a,Inf)]
ans =
   2.092844953645635e+001   2.092844953645635e+001
   3.400000000000000e+001   1.600000000000000e+001

>> A=[16 2 3 13; 5 11 10 8; 9 7 6 12; 4 14 15 1];
[norm(A), norm(A,2), norm(A,1), norm(A,Inf)]
ans =
   34    34    34    34
```

符号运算工具箱未提供norm( )函数，需先用double( )函数转换成双精度数值矩阵，再调用norm( )函数。

- 特征多项式

矩阵 $\boldsymbol{A}$ 的特征多项式:

$$C(s) = \det(s\boldsymbol{I} - \boldsymbol{A}) = s^n + c_1 s^{n-1} + \cdots + c_{n-1} s + c_n$$

格式： C=poly(A)

例： >> A=[16 2 3 13; 5 11 10 8; 9 7 6 12; 4 14 15 1];

```
>> poly(A)    %直接求取
ans =
1.000000000000000e+000   -3.399999999999999e+001
 -7.999999999999986e+001    2.719999999999999e+003
  -2.819840539024018e-012
>> A=sym(A); poly(A)    %运用符号工具箱
 ans =
 x^4-34*x^3-80*x^2+2720*x
```

- 符号多项式与数值多项式的转换

向量 $\boldsymbol{P}=[a_1,a_2,\cdots,a_{n+1}]$

向量转换成多项式表示：

格式：

f=poly2sym(P) 或 f=poly2sym(P,x)

多项式符号表达式转换成向量：

格式：　P=sym2poly(f)

- 例：　用不同形式表示 $f = s^5+2s^4+3s^3+4s^2+5s+6$。

转换成符号型的多项式：

```
>> P=[1 2 3 4 5 6];  % 先由系数按降幂顺序排列表示多项式
>> f=poly2sym(P,'v')  % 以 v 为算子表示多项式
f =
 v^5+2*v^4+3*v^3+4*v^2+5*v+6
```

转换成数值形式的多项式：

```
>> P=sym2poly(f)
P =
     1     2     3     4     5     6
```

- 矩阵的相似变换与正交矩阵

$$X = B^{-1}AB$$

其中：A为一方阵，B矩阵非奇异。

相似变换后，X矩阵的秩、迹、行列式与特征值等均不发生变化，其值与A矩阵完全一致。

对于一类特殊的相似变换满足如下条件，称为正交基矩阵。

正交矩阵：

$$Q^*Q = I, \text{ 且 } QQ^* = I$$

$$Q=\mathrm{orth}(A)$$

例：

求 $A=\begin{bmatrix}5&9&8&3\\0&3&2&4\\2&3&5&9\\3&4&5&8\end{bmatrix}$ 的正交矩阵

```
>> A=[5,9,8,3; 0,3,2,4; 2,3,5,9; 3,4,5,8];
>> Q=orth(A)
Q =
   -0.6197    0.7738   -0.0262   -0.1286
   -0.2548   -0.1551    0.9490    0.1017
   -0.5198   -0.5298   -0.1563   -0.6517
   -0.5300   -0.3106   -0.2725    0.7406
>> norm(Q'*Q-eye(4))
ans =
  4.6395e-016
>> norm(Q*Q'-eye(4))
ans =
  4.9270e-016
```

例：

求 $A=\begin{bmatrix}16 & 2 & 3 & 13\\ 5 & 11 & 10 & 8\\ 9 & 7 & 6 & 12\\ 4 & 14 & 15 & 1\end{bmatrix}$ 的正交基矩阵

```
>> A=[16,2,3,13; 5,11,10,8; 9,7,6,12; 4,14,15,1];
>> Q=orth(A) %A为奇异矩阵，故得出的Q为长方形矩阵
Q =
   -0.5000    0.6708    0.5000
   -0.5000   -0.2236   -0.5000
   -0.5000    0.2236   -0.5000
   -0.5000   -0.6708    0.5000
>> norm(Q'*Q-eye(3))
ans =
  1.0140e-015
```

线性方程组直接解法

1线性方程组直接求解—矩阵除法

- 关于线性方程组的直接解法，如Gauss消去法、选主元消去法、平方根法、追赶法等等，在MATLAB中，只需用“／”或“\”就解决问题。它内部实际包含着许许多多的自适应算法，如对超定方程用最小二乘法，对欠定方程时它将给出范数最小的一个解，解三对角阵方程组时用追赶法等等。

  格式：　　x=A\b

- 例：解方程组

$$\begin{cases} 0.4096x_1 + 0.1234x_2 + 0.3678x_3 + 0.2943x_4 = 0.4043 \\ 0.2246x_1 + 0.3872x_2 + 0.4015x_3 + 0.1129x_4 = 0.1150 \\ 0.3645x_1 + 0.1920x_2 + 0.3781x_3 + 0.0643x_4 = 0.4240 \\ 0.1784x_1 + 0.4002x_2 + 0.2786x_3 + 0.3927x_4 = -0.2557 \end{cases}$$

```
>> A=[.4096,.1234,.3678,.2943;.2246,.3872,.4015,.1129;
.3645,.1920,.3781,.0643;.1784,.4002,.2786,.3927];
>> b=[0.4043 0.1550 0.4240 -0.2557]';
>> x=A\b; x'
ans =
   -0.1819   -1.6630    2.2172   -0.4467
```

线性方程组的直接求解分析

- LU分解

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$$

$$\text{其中 } \boldsymbol{L} = \begin{bmatrix} 1 & & & \\ l_{21} & 1 & & \\ \vdots & \vdots & \ddots & \\ l_{n1} & l_{n2} & \cdots & 1 \end{bmatrix}, \quad \boldsymbol{U} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ & u_{22} & \cdots & u_{2n} \\ & & \ddots & \vdots \\ & & & u_{nn} \end{bmatrix}$$

矩阵L和U与矩阵A的关系：

$$
\begin{array}{llll}
a_{11} = u_{11}, & a_{12} = u_{12}, & \cdots & a_{1n} = u_{1n} \\
a_{21} = l_{21}u_{11}, & a_{22} = l_{21}u_{12} + u_{22}, & \cdots & u_{2n} = l_{21}u_{1n} + u_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{n1} = l_{n1}u_{11}, & a_{n2} = l_{n1}u_{12} + l_{n2}u_{22}, & \cdots & a_{nn} = \sum_{k=1}^{n-1} l_{nk}u_{kn} + u_{nn}
\end{array}
$$

递推计算公式：

$$l_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} l_{ik}u_{kj}}{u_{jj}}, \quad (j < i)$$

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik}u_{kj}, \quad (j \geq i)$$

递推初值：

$$u_{1i} = a_{1i}, \quad i = 1, 2, \cdots, n$$

- 格式

[l,u,p]=lu(A)

L是一个单位下三角矩阵，u是一个上三角矩阵，

p是代表选主元的置换矩阵。

故：Ax=y => PAx=Py

=> LUx=Py => PA=LU

[l,u]=lu(A)

其中l等于P^{-1} L，u等于U，所以(P^{-1} L)U=A

- 例：对A进行LU分解

$$A=\begin{pmatrix}1 & 2 & 3\\ 2 & 4 & 1\\ 4 & 6 & 7\end{pmatrix}$$

```
>> A=[1 2 3; 2 4 1; 4 6 7];
>> [l,u,p]=lu(A)
l =
    1.0000         0         0
    0.5000    1.0000         0
    0.2500    0.5000    1.0000
u =
    4.0000    6.0000    7.0000
         0    1.0000   -2.5000
         0         0    2.5000
p =
     0     0     1
     0     1     0
     1     0     0
```

```
>> [l,u]=lu(A)                % l=P^-1 L

l =
    0.2500    0.5000    1.0000
    0.5000    1.0000         0
    1.0000         0         0

u =
    4.0000    6.0000    7.0000
         0    1.0000   -2.5000
         0         0    2.5000
```

- QR分解

  将矩阵A分解成一个正交矩阵与一个上三角矩阵的乘积。

  格式：

  [Q,R] = qr(A)

  求得正交矩阵Q和上三角阵R，Q和R满足A=QR。

- 例：

```
>> A =[ 1  2  3;4  5  6; 7  8  9; 10  11  12];
>> [Q,R] = qr(A)
Q =
   -0.0776   -0.8331    0.5456   -0.0478
   -0.3105   -0.4512   -0.6919    0.4704
   -0.5433   -0.0694   -0.2531   -0.7975
   -0.7762    0.3124    0.3994    0.3748
R =
  -12.8841  -14.5916  -16.2992
         0   -1.0413   -2.0826
         0         0   -0.0000
         0         0         0
```

- Cholesky 分解

若矩阵A为 n阶对称正定阵,则存在唯一的对角元素为正的三角阵D，使得

$$A = D^T D = \begin{bmatrix} d_{11} & & & \\ d_{21} & d_{22} & & \\ \vdots & \vdots & \ddots & \\ d_{n1} & d_{n2} & \cdots & d_{nn} \end{bmatrix} \begin{bmatrix} d_{11} & d_{21} & \cdots & d_{n1} \\ & d_{22} & \cdots & d_{n2} \\ & & \ddots & \vdots \\ & & & d_{nn} \end{bmatrix}$$

其中 A 为对称矩阵

对称矩阵的 Cholesky 分解算法：

$$d_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} d_{ik}^2}, \qquad d_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} d_{ik} d_{jk}}{l_{jj}}, \ (j < i)$$

格式：

D=chol(A)

例：进行Cholesky分解。

$$A=\begin{pmatrix}16 & 4 & 8\\ 4 & 5 & -4\\ 8 & -4 & 22\end{pmatrix}$$

```
>> A=[16 4 8; 4 5 -4; 8 -4 22];
>> D=chol(A)
D =
     4     1     2
     0     2    -3
     0     0     3
```

利用矩阵的LU、QR和cholesky分解求方程组的解

（1）LU分解：

A*X=b　　变成　L*U*X=b

所以 X=U\(L\b) 这样可以大大提高运算速度。

（2）Cholesky 分解

若A为对称正定矩阵，则Cholesky分解可将矩阵A分解成上三角矩阵和其转置的乘积，

方程　A*X=b　　变成　　R'*R*X=b

所以　　X=R\(R'\b)

（3）QR 分解

对于任何长方矩阵A，都可以进行QR分解，其中Q为正交矩阵，R为上三角矩阵的初等变换形式，即：A=QR

方程　A*X=b　变形成　QRX=b

所以　　X=R\(Q\b)

这三种分解，在求解大型方程组时很有用。其优点是运算速度快、可以节省磁盘空间、节省内存。

- 三个变换

在线性方程组的迭代求解中，要用到系数矩阵A的上三角矩阵、对角阵和下三角矩阵。此三个变换在MATLAB中可由以下函数实现。

– 上三角变换：

格式 triu(A,1)

– 对角变换：

格式 diag(A)

– 下三角变换：

格式 tril(A,-1)

例：对此矩阵做三种变换。 $A=\begin{pmatrix}1 & 2 & -2\\ 1 & 1 & 1\\ 2 & 2 & 1\end{pmatrix}$

```
>> A=[1 2 -2;1 1 1;2 2 1];  %
>> triu(A,1)
ans =
     0     2    -2
     0     0     1
     0     0     0
>> tril(A,-1)
ans =
     0     0     0
     1     0     0
     2     2     0
>> b=diag(A); b'
ans =
     1     1     1
```

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

迭代解法的几种形式
1 Jacobi迭代法

- 方程组 $Ax=b$

  A可写成 $A=D-L-U$

  其中:$D=diag[a_{11},a_{22},\ldots,a_{nn}]$, -L、-U分别为A的严格下、上三角部分（不包括对角线元素）.

  由 $Ax=b \Rightarrow x=Bx+f$

由此可构造迭代法:

$$x^{(k+1)}=Bx^{(k)}+f$$

其中： $B=D^{-1}(L+U)=I-D^{-1}A$, $f=D^{-1}b$.

```
function y=jacobi(a,b,x0)
D=diag(diag(a));  U=-triu(a,1); L=-tril(a,-1);
B=D\(L+U);   f=D\b;
y=B*x0+f;
n=1;
while norm(y-x0)>=1.0e-6
    x0=y;
    y=B*x0+f;
    n=n+1;
end
n
```

- 例： 用Jacobi方法求解，

设x(0)=0,精度为10^{-6}。

$$\begin{cases} 10x_1 - x_2 = 9 \\ -x_1 + 10x_2 - 2x_3 = 7 \\ -2x_2 + 10x_3 = 6 \end{cases}$$

```
>> a=[10 -1 0; -1 10 -2; 0 -2 10];
>> b=[9; 7; 6];
>> jacobi(a,b,[0;0;0])
n =
    11
ans =
    0.9958
    0.9579
    0.7916
```

2 Gauss-Seidel迭代法

由原方程构造迭代方程

$$x^{(k+1)}=G\,x^{(k)}+f$$

其中：$G=(D-L)^{-1}U$，　　$f=(D-L)^{-1}b$

$D=\mathrm{diag}[a_{11},a_{22},\ldots,a_{nn}]$,

-L、-U分别为A的严格下、上三角部分（不包括对角线元素）.

```
function y=seidel(a,b,x0)
D=diag(diag(a));U=-triu(a,1);L=-tril(a,-1);
G=(D-L)\U  ;f=(D-L)\b;
y=G*x0+f;  n=1;
while norm(y-x0)>=1.0e-6
    x0=y;
    y=G*x0+f;
    n=n+1;
end
n
```

- 例：对上例用 Gauss-Seidel 迭代法求解

```
>>  a=[10 -1 0; -1 10 -2; 0 -2 10];
>> b=[9; 7; 6];
>> seidel(a,b,[0;0;0])
n =
     7
ans =
    0.9958
    0.9579
    0.7916
```

例：分别用Jacobi和G-S法迭代求解,看是否收敛。

$$\begin{pmatrix} 1 & 2 & -2 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 9 \\ 7 \\ 6 \end{pmatrix}$$

```
>> a=[1 2 -2; 1 1 1; 2 2 1];  b=[9; 7; 6];
>> jacobi(a,b,[0;0;0])
n =
     4
ans =
   -27
    26
     8
>>  seidel(a,b,[0;0;0])
n =
        1011
ans =
  1.0e+305 *
      -Inf
       Inf
   -1.7556
```

3 SOR迭代法

在很多情况下，J法和G-S法收敛较慢，所以考虑对G-S法进行改进。于是引入一种新的迭代法—逐次超松弛迭代法（Successive Over-Relaxation),记为SQR法。

迭代公式为：

$$X^{(k+1)}= (D-wL)^{-1}((1-w)D+wU)x^{(k)} + w(D-wL)^{-1}\, b$$

其中：w最佳值在[1, 2）之间，不易计算得到，因此 w通常有经验给出。

```
function y=sor(a,b,w,x0)
D=diag(diag(a));U=-triu(a,1);L=-tril(a,-1);
M=(D-w*L)\((1-w)*D+w*U); f=(D-w*L)\b*w;
y=M*x0+f; n=1;
while norm(y-x0)>=1.0e-6
    x0=y;
    y=M*x0+f;
    n=n+1;
end
n
```

例：上例中，当w=1.103时，用SOR法求解原方程。

```
>>  a=[10 -1 0; -1 10 -2; 0 -2 10];
>> b=[9; 7; 6];
>> sor(a,b,1.103,[0;0;0])
n =
     8
ans =
    0.9958
    0.9579
    0.7916
```

4 两步迭代法

当线性方程系数矩阵为对称正定时，可用一种特殊的迭代法来解决，其迭代公式为：

$$(D-L)\,x^{(k+1/2)} = U\,x^{(k)} + b$$

$$(D-U)\,x^{(k+1)} = L x^{(k+1/2)} + b$$

$\Rightarrow$

$$x^{(k+1/2)} = (D-L)^{-1}\,U\,x^{(k)} + (D-L)^{-1}\,b$$

$$x^{(k+1)} = (D-U)^{-1}\,L x^{(k+1/2)} + (D-U)^{-1}\,b$$

```
function y=twostp(a,b,x0)
D=diag(diag(a));U=-triu(a,1);L=-tril(a,-1);
G1=(D-L)\U; f1=(D-L)\b;
G2=(D-U)\L; f2=(D-U)\b;
y=G1*x0+f1; y=G2*y+f2; n=1;
while norm(y-x0)>=1.0e-6
    x0=y;
    y=G1*x0+f1; y=G2*y+f2;
    n=n+1;
end
n
```

- 例：求解方程组

$$\begin{pmatrix} 10 & -1 & 2 & 0 \\ -1 & 11 & -1 & 3 \\ 2 & -1 & 10 & -1 \\ 0 & 3 & -1 & 8 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 6 \\ 25 \\ -11 \\ 15 \end{pmatrix}$$

```
>> a=[10 -1 2 0; -1 11 -1 3; 2 -1 10 3; 0 3 -1 8];
>> b=[6; 25; -11; 15];
>> twostp(a, b, [0; 0; 0; 0])
n =
     7
ans =
    1.0791
    1.9824
   -1.4044
    0.9560
```