Numerical Analysis

Mathematics of Scientific Computing

主讲人 孙晓庆
幻灯片制作 孙晓庆

中国海洋大学 信息科学与工程学院

2013 年 10 月 11 日

目录

内容提要

下一节内容

let us consider a system of n unknows x_1, x_2, ..., x_n. It can be written in the form

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ a_{31} & a_{32} & \dots & a_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}$$

The matrices in this equation are denoted by A, x, and b. Thus, our system is simply $Ax = b$

Matrix A has a **diagonal structure**. This mains that all the nonzero elements of A are on the main diagonal.

$$\begin{bmatrix} a_{11} & 0 & 0 & \dots & 0 \\ 0 & a_{22} & 0 & \dots & 0 \\ 0 & 0 & a_{33} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}$$

The solution is

$$x = \begin{bmatrix} b_1/a_{11} \\ b_2/a_{22} \\ b_3/a_{33} \\ \vdots \\ b_n/a_{nn} \end{bmatrix}$$

lower triangular structure

We assume a **lower triangular structure** for A. This means that all the nonzero elements of A are situated on or below the main diagoal, the system is

$$\begin{bmatrix} a_{11} & 0 & 0 & \dots & 0 \\ a_{21} & a_{22} & 0 & \dots & 0 \\ a_{31} & a_{32} & a_{33} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}$$

To solve this, assume that $a_{ii} \neq 0$ for all i; then obtain x_1 from the first equation. With the value of x_1 substituted into the second equation, solve the second equation for x_2. We proceed in the same way, obtaining x_1, x_2, ..., x_n, one at a time and in this order. A formal argorithm for the solution in this case is called **forward substitution**:

input n, (a_{ij}), (b_i)
for $i = 1$ to n do
$x_i \leftarrow (b_i - \sum_{j=1}^{i-1} a_{ij} x_j)/a_{ii}$
end do
output (x_i)
As is customary, any sum of the type $\sum_{j=\alpha}^{\beta} x_i$ in which $\beta < \alpha$ is interpreted to be 0.

EXAMPLE 1

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ -3 & 1 & 0 & 0 \\ -2 & 2 & 1 & 0 \\ 3 & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ 0 \\ -1 \end{bmatrix}$$

EXAMPLE 1

```
function x=lowerang(n,a,b)
for i=1:n
    s=b(i)
    for j=1:(i-1)
        s=s-a(i,j)*x(j)
    end
    x(i)=s/a(i,i)
end
```

```
clc
a=[1 0 0 0;-3 1 0 0;-2 2 1 0;3 0 -1 1]; b=[0 -1 0 -1];n=4
 x=lowerang(n,a,b)
```

upper triangular structure

Upper triangular structure has the form

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ 0 & a_{22} & a_{23} & \dots & a_{2n} \\ 0 & 0 & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}$$

It must be assumed that $a_{ii} \neq 0$ for $1 \leq i \leq n$. The formal algorithm to solve for x is as follows and is called **back substitution**.

input n, (a_{ij}), (b_i)
for $i = n$ to 1 step -1 do
$x_i \leftarrow (b_i - \sum_{j=i+1}^{n} a_{ij}x_j)/a_{ii}$
end do
output (x_i)

EXAMPLE 2

$$\begin{bmatrix} 3 & 0 & -1 & 1 \\ 0 & -2 & 2 & 1 \\ 0 & 0 & -3 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ 0 \\ -1 \end{bmatrix}$$

EXAMPLE 2

```
function x=upperang(n,a,b)
for i=n:-1:1
    s=b(i);
    for j=(i+1):n
        s=s-a(i,j)*x(j);
    end
    x(i)=s/a(i,i);
end
```

```
clc;
a=[1 -1 0 3;0 1 2 2; 0 0 1 3; 0 0 0 1];b=[-1 0 -1 0];n=4;
x=upperang(n,a,b)
```

$$\begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & a_{23} \\ a_{31} & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

If we simply reorder these equations, we can get a lower trianglar system:

$$\begin{bmatrix} a_{31} & 0 & 0 \\ a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_3 \\ b_1 \\ b_2 \end{bmatrix}$$

Putting this another way, we should solve the equations of System not in the usual order 1, 2, 3, but in the order 3, 1, 2.

We wish to say that one row of A, say row p_1, has zero in positions 2, 3, ...,n. Then another row p_2, has zeros in positions 3, 4, ...,n, and so on. Let's assume that the **permulation vector** $(p_1, p_2, ..., p_n)$ is known or has been determined somehow beforehand. Modifying our pervious algorithms, we arrive at **forward substitution** for a **permuted lower triangular system**:

input n, (a_{ij}), (b_i), (p_i)
for $i = 1$ to n do
$x_i \leftarrow (b_{p_i} - \sum_{j=1}^{i-1} a_{p_i j} x_j)/a_{p_i i}$
end do
output(x_i)

EXAMPLE 2

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ -2 & 2 & 1 & 0 \\ 3 & 0 & -1 & 1 \\ -3 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -1 \\ -1 \end{bmatrix}$$

```
function x=lowertriangular(n,a,b,p)
for i=1:n
    s=b(p(i))
    for j=1:(i-1)
        s=s-a(p(i),j)*x(j)
    end
    x(i)=s/a(p(i),i)
end
```

```
clc
n=4,a=[1 0 0 0;-2 2 1 0;3 0 -1 1;-3 1 0 0]; b=[0 0 -1 -1];
p=[1 4 2 3]
x=lowertriangular(n,a,b,p)
```

Similarly,**back sustitution** for a **permuted upper triangular system** is as follows:

input n, (a_{ij}), (b_i), (p_i)
for $i = n$ to 1 step -1 do
$x_i \leftarrow (b_{p_i} - \sum_{j=i+1}^{n} a_{p_i j} x_j) / a_{p_i i}$
end do
output(x_i)

下一节内容

- Suppose that A can be factored into the product of a lower triangular matrix L and an upper triangular matrix U: $A = LU$.
- Then to solve the system of equations $Ax = b$,
  $Lz = b$ solve for z
  $Ux = z$ solve for x

We begin with an $n \times n$ matrix A and search for matrices

$$L = \begin{bmatrix} l_{11} & 0 & 0 & \dots & 0 \\ l_{21} & l_{22} & 0 & \dots & 0 \\ l_{31} & l_{32} & l_{33} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \dots & l_{nn} \end{bmatrix}$$

$$U = \begin{bmatrix} u_{11} & u_{12} & u_{13} & \dots & u_{1n} \\ 0 & u_{22} & u_{23} & \dots & u_{2n} \\ 0 & 0 & u_{33} & \dots & u_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & u_{nn} \end{bmatrix}$$

Such that

$$A = LU$$

- We start with the formula for matrix multiplication:

$$a_{ij} = \sum_{k=1}^{n} l_{is} u_{sj} = \sum_{k=1}^{min(i,j)} l_{is} u_{sj}$$

- If u_{kk} or l_{kk} has specified, we can use the Equation $a_{kk} = \sum_{s=1}^{k-1} l_{ks} u_{sk} + l_{kk} u_{kk}$ to detetermine the other.
- If $l_{kk} \neq 0$ Equation(1) can be used to obtain the elements u_{kj}, If $u_{kk} \neq 0$ Equation(2) can be used to obtain the elements l_{ik}.

$$a_{kj} = \sum_{s=1}^{k-1} l_{ks} u_{sj} + l_{kk} u_{kj} \qquad (k+1 \leq j \leq n) \tag{1}$$

$$a_{ik} = \sum_{s=1}^{k-1} l_{is} u_{sk} + l_{ik} u_{kk} \qquad (k+1 \leq i \leq n) \tag{2}$$

general LU-factorization

input n, (a_{ij})
for $k = 1$ to n do
Specify a nonzero value for either l_{kk} or u_{kk} and compute the other form
$l_{kk}u_{kk} = a_{kk} - \sum_{s=1}^{k-1} l_{ks}u_{sk}$
for $j = k+1$ to n do
$u_{kj} \leftarrow (a_{kj} - \sum_{s=1}^{k-1} l_{ks}u_{sj})/l_{kk}$
end do
for $i = k+1$ to n do
$l_{ik} \leftarrow (a_{ik} - \sum_{s=1}^{k-1} l_{is}u_{sk})/u_{kk}$
end do
end do
output (l_{ij}), (u_{ij})

three factorizations

- When L is unit lower triangular $(l_{ii} = 1)$ for $(1 \leq i \leq n)$, the algorithm is called **Doolittle's factorization**.
- When U is unit upper triangular $(u_{ii} = 1)$ for $(1 \leq i \leq n)$, the algorithm is called **Crout's factorization**.
- When $U = L^T$ so that $l_{ii} = u_{ii}$ for $(1 \leq i \leq n)$, the algorithm is called **Cholesky's factorization**.

EXAMPLE 3

Find the Doolittle, Crout, and Cholesky factorizations of the matrix

$$A = \begin{bmatrix} 60 & 30 & 20 \\ 30 & 20 & 15 \\ 20 & 15 & 12 \end{bmatrix}$$

```
clc
a=[60 30 20;30 20 15;20 15 12]; n=3
 [l,u]=doolittle(n,a)
```

```
function [l,u]=doolittle(n,a)
for k=1:n
    l(k,k)=1;
    for j=k:n
        p=a(k,j);
        for s=1:(k-1)
            p=p-l(k,s)*u(s,j);
        end
        u(k,j)=p
        for i=(k+1):n
            q=a(i,k)
            for s=1:(k-1)
                q=q-l(i,s)*u(s,k)
            end
            l(i,k)=q/u(k,k)
        end
    end
end
```

```
function [l,u]=crout(n,a)
for k=1:n
    u(k,k)=1;
    for i=k:n
        p=a(i,k);
        for s=1:(k-1)
        p=p-l(i,s)*u(s,k);
        end
    l(i,k)=p;
    end
    for j=(k+1):n
        q=a(k,j);
        for m=1:(k-1)
        q=q-l(k,m)*u(m,j);
        end
        u(k,j)=q/l(k,k);
    end
    format rat
end
```

The algorithm for the **Cholesky factorization** will then be as follows:

input n, (a_{ij})

for $k = 1$ to n do

$l_{kk} \leftarrow (a_{kk} - \sum_{s=1}^{k-1} l_{ks}^2)^{1/2}$

for $i = k+1$ to n do

$l_{ik} \leftarrow (a_{ik} - \sum_{s=1}^{k-1} l_{is} l_{ks}) / l_{kk}$

end do

end do

output (l_{ij})

```
function l=cholesky(n,a)
for k=1:n
    p=a(k,k);
    for s=1:(k-1)
        p=p-[l(k,s)]^2;
    end
    l(k,k)=p^(1/2);
    for i=(k+1):n
        q=a(i,k);
        for m=1:(k-1)
            q=q-l(i,m)*l(k,m);
        end
        l(i,k)=q/l(k,k);
    end
end
```

THEOREM 1 (Theorem on LU-Decompostition)

If all n leading principal minors of the $n \times n$ matrix A are nonsigular, then A has LU-decompostition.

Proof.

Recall that the kth leading **principal minor** of the matrix A is the matrix

$$A_k = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1k} \\ a_{21} & a_{22} & \dots & a_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k1} & a_{k2} & \dots & a_{kk} \end{bmatrix}$$

Suppose that L_{k-1} and U_{k-1} have been obtained. Hence, Equation $a_{ij} = \sum_{k=1}^{n} l_{is}u_{sj} = \sum_{k=1}^{min(i,j)} l_{is}u_{sj}$ states that $A_{k-1} = L_{k-1}U_{k-1}$. Since A_{k-1} is nonsigular by hypothesis, L_{k-1} and U_{k-1} are also nonsinglar. □

Proof.

Since L_{k-1} is nonsigular, we can solve the system $\sum_{s=1}^{k-1} l_{is}u_{sk} = a_{ik}$ $(1 \leq i \leq k-1)$ for the quantities u_{sk} with $(1 \leq s \leq k-1)$. These elements lie in the kth column of U.

Since U_{k-1} is nonsigular, We can solve the system $\sum_{s=1}^{k-1} l_{ks}u_{sj} = a_{kj}$ $(1 \leq j \leq k-1)$ for the quantities l_{ks} with $(1 \leq s \leq k-1)$. These elements lie in the kth row of L.

From the requirement $a_{kk} = \sum_{s=1}^{k-1} l_{ks}u_{sk} + l_{kk}u_{kk}$ we can obtain u_{kk} since l_{kk} has been specified as unity.

Thus, all the new elements necessary to form L_k and U_k have been defined.

The induction is completed by nothing that $l_{11}u_{11} = a_{11}$ and, therefore, $l_{11} = 1$ and $u_{11} = a_{11}$. □

THEOREM 2 (Cholesky Theorem on LL^T-Factorization)

If A is a real, symmetric, and positive definite matrix, then it has a unique factorization, $A = LL^T$, in which L is lower triangular with a positive diagonal.

Proof.

A matrix A is symmertic and positive definite if $A = A^T$ and $x^T Ax > 0$ for every nonzero vector x. By considering special vectors of the form $x = (x_1, x_2, x_3, , ..., x_k, 0, 0, ..., 0)^T$, we see that the leading principal minors of A are positive definite. Therom 1 implies that A has LU-decomposition. By the symmetry of A, we have $LU = A = A^T = U^T L^T$,This implies that $U(L^T)^{-1} = L^{-1}U^T$. The left of the equation is upper trigular, whereas the right member is lower triangular. There is a diagoal matrix D such $D = U(L^T)^{-1}$. $U = DL^T$ and $A = LDL^T$. D is positive definite. $A = L'L'^T$, where $L' = LD^{1/2}$. □

Proof.

Uniqueness proof:
Suppose that there exist L_1 such that $A = LL^T = L_1 L_1^T$. Let's show that $L = L_1$.

$$A = LL^T = L_1 L_1^T$$
$$L^{-1}LL^T = L^{-1}L_1 L_1^T$$
$$L^T = L^{-1}L_1 L_1^T$$
$$L^T(L_1^T)^{-1} = L^{-1}L_1 = D^*$$
$$L_1 = LD^*$$
$$L^T = D^* L_1^T$$
$$D^*(LD^*)^T = D^* {D^*}^T L^T = (D^*)^2 L^T$$
$$D^*(LD^*)^T = D^* L_1^T = L^T$$
$$(D^*)^2 L^T = L^T$$
$$(D^*)^2 = I$$
$$D^* = I = L^{-1}L_1$$
$$L = L_1$$

So it has a unique factorization. □