Resolución de sistema de ecuaciones lineales. Métodos directos

Juan Manuel Rabasedas

 

Sistemas lineales

Los sistemas de ecuaciones lineales aparecen en un gran número de áreas, tanto directamente modelizando situaciones físicas o indirectamente en la solución numérica de otros modelos matemáticos.

Sistemas lineales

Pretendemos resolver un sistema de forma $Ax = b$

$$\left\{\begin{array}{ccc} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = & b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & = & b_2 \\ \vdots \quad \vdots \quad \vdots & & \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n & = & b_n \end{array}\right. \tag{1}$$

con

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \in M_n(\mathbb{R}), \qquad b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \in \mathbb{R}^n. \tag{2}$$

Nos interesa resolver sistemas de n ecuaciones con n incógnitas, cuando exista solución y sea única. Estas condiciones se verifican si

$$\det(A) \neq 0\,.$$

Sistemas Diagonales

El sistema más sencillo que podemos resolver está dado por

$$\left\{\begin{array}{ccccccccc} a_{11}x_1 & + & 0 & + & \cdots & + & 0 & = & b_1 \\ 0 & + & a_{22}x_2 & + & \cdots & + & 0 & = & b_2 \\ \vdots & & 0 & & \ddots & & 0 & & \vdots \\ 0 & + & 0 & + & \cdots & + & a_{nn}x_n & = & b_n \end{array}\right.$$

donde la solución se calcula trivialmente haciendo

$$x_i = b_i/a_{ii}, \;\; i = 1, ..., n$$

Ejercicio: ¿Cómo podemos resolverlo en Scilab?
Esto se puede escribir en Scilab

```
x = b./diag(A)
```

En un sistema triangular dado por

$$\left\{\begin{array}{ccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ 0 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ \vdots & & 0 & & \ddots & & \vdots & & \vdots \\ 0 & + & 0 & + & \cdots & + & a_{nn}x_n & = & b_n \end{array}\right.$$

la última ecuación es de resolución inmediata $x_n = b_n/a_{nn}$ y luego se realiza una **sustitución regresiva** o **remonte** obteniéndose las componentes de la solución como

$$x_i = \frac{1}{a_{ii}}\left(b_i - \sum_{j=i+1}^{n} a_{ij}x_j\right)$$

El esquema del algoritmo es el siguiente:

$x_n \;=\; b_n/a_{nn}$

para $i = \; n-1,\ldots,1$

  $x_i \;=\; (b_i - A(i,i+1:n)x(i+1:n))/a_{ii}$

finpara

Esto se puede escribir en Scilab

```
x(n) = b(n)/A(n,n)
for i=n-1:-1:1
x(i) = ( b(i) - A(i,i+1:n)*x(i+1:n) )/A(i,i)
end
```


Ejercicio

Implementar en Scilab una función **remonte** *que tome como parámetros una matriz A triangular superior y un vector b y resuelva el sistema* $Ax = b$

```
function s1 = remonte(A, b)
  n = size(A,1)
  for i = n:-1:1
    x(i) = (b(i) - A(i,i+1:n)*x(i+1:n)) / A(i,i)
  end
  s1 = x
endfunction
```


Metodo de Gauss

La idea es transformar un sistema

$$\left\{\begin{array}{lllll} (e_1) & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = & b_1 \\ (e_2) & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & = & b_2 \\ \vdots & \vdots \quad \vdots \quad \vdots & & \vdots \\ (e_n) & a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n & = & b_n \end{array}\right. \tag{3}$$

en uno triangular más sencillo de resolver aplicando remonte. Si $a_{11} \neq 0$, podemos sustituir e_2 por $e_2 - \frac{a_{21}}{a_{11}}e_1$, eliminando el primer término como sigue

$$(e_2)\; 0 + (a_{22} - \frac{a_{21}}{a_{11}}a_{12})x_2 + \cdots + (a_{2n} - \frac{a_{21}}{a_{11}}a_{1n})x_n = b_2 - \frac{a_{21}}{a_{11}}b_1$$

Continuando con las sustituciones:

$$\begin{array}{ccc} e_3 & = & e_3 - \frac{a_{31}}{a_{11}}e_1 \\ \vdots & & \vdots \\ e_n & = & e_n - \frac{a_{n1}}{a_{11}}e_1 \end{array}$$

Metodo de Gauss

Se obtiene ceros en la primer columna:

$$\left\{\begin{array}{llll} (e_1) & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = & b_1 \\ (e_2) & 0x_1 + a'_{22}x_2 + \cdots + a'_{2n}x_n & = & b'_2 \\ \vdots & \vdots \quad \vdots \quad \vdots & & \vdots \\ (e_n) & 0x_1 + a'_{n2}x_2 + \cdots + a'_{nn}x_n & = & b'_n \end{array}\right. \tag{4}$$

Continuando con las sustituciones de la siguiente manera:

$$\begin{array}{lll} e_3 & = & e_3 - \frac{a_{32}}{a_{22}} e_2 \\ \vdots & & \vdots \\ e_n & = & e_n - \frac{a_{n2}}{a_{22}} e_2 \end{array}$$

Obtenemos ceros en la segunda columna

Podemos continuar hasta hacer cero todos los elementos por debajo de la diagonal.

Metodo de Gauss

En general, cuando ya hay ceros por debajo de la diagonal, en las columnas $1, 2, \ldots, k-1$ para obtener cero en la posición (j, k) se hace la operación: $e_j = e_j - a_{jk}/a_{kk} * e_k$

Esto lo podemos escribir es Scilab:

```
mjk = a(j,k)/a(k,k)
A(j, :) = A(j, :) - mjk * A(k, :)
b(j) = b(j) - mjk * b(k)
```

Si evitamos operar con los elementos que ya se han puesto en cero el esquema de la triangulación nos queda:

para $k = 1, \ldots, n-1$
  para $j = k+1, \ldots, n$
    $mjk \; = a_{jk}/a_{kk}$
    $a_{jk} \; = 0$
    $A(j, k+1:n) \; = A(j, k+1:n) - mjk * A(k, k+1:n)$
    $bj \; = bj - mjk * bk$
  finpara j
fin-para k

Metodo de Gauss

Luego en Scilab la triangulación nos queda:

```
function [s1, s2] = gauss(A, b)
  a = size(A)
  n = a(1)
  for i = 1:(n-1)
    for j = (i+1):a(1)
      mjk = A(j,i)/A(i,i)
      A(j,i)=0
      A(j,i+1:n) = A(j,i+1:n) - mjk*A(i,i+1:n)
      b(j) = b(j) - mjk*b(i)
      end
  end
  s1 = A
  s2 = b
endfunction
```


Factorización *LU*

Si resolvemos $Ax = b$ con eliminación Gaussiana (sin pivoteo) el sistema se transforma en $Ux = g$ donde

$$U = \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ 0 & u_{12} & \cdots & u_{1n} \\ & & \ddots & \\ 0 & 0 & \cdots & u_{nn} \end{pmatrix}, \quad con\ u_{ij} = a_{ij}^{(i)}$$

Podemos tener una matriz auxiliar triangular inferior, basada en los coeficientes $mjk = a_{jk}/akk$ definidos anteriormente.

$$L = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ m_{21} & 1 & \cdots & 0 \\ & & \ddots & \\ m_{n1} & m_{n2} & \cdots & 1 \end{pmatrix}$$

Teorema

Sea A una matriz no singular y sean L y U las matrices recién definidas. Entonces si U es producida sin pivoteo se tiene que LU = A, esto es lo que se llama la **factorización LU**

2 sistemas más sencillos

Esta factorización LU permite resolver el sistema a través de 2 sistemas más sencillos

$$LU = A \longleftrightarrow \begin{cases} Ly = b, \\ Ux = y. \end{cases}$$

Ambos sistemas son triangulares, por ende fáciles de resolver.
Este análisis es válido siempre que existan matrices L y U, triangular inferior la primera y triangular superior la segunda tales que $LU = A$.
Existen varios métodos para encontrar estas matrices:

- $l_{ii} = 1$, Doolite,
- $u_{ii} = 1$, Crout,
- Si $U = L^t$ i.e. $u_{ii} = l_{ii}$, $\forall i$, Cholesky.

Factorización de Cholesky

Sea

$$\begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix}$$

Entonces

$$\begin{pmatrix} u_{11} & 0 \\ u_{12} & u_{22} \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} \\ 0 & u_{22} \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix}$$

$$\begin{aligned} u_{11}^2 &= 1 \\ u_{11}u_{12} &= 2 \\ u_{12}^2 + u_{22}^2 &= 5 \end{aligned}$$

Se deduce que

$$\begin{aligned} u_{11} &= 1 \\ u_{12} &= 2 \\ u_{22} &= 1 \end{aligned}$$

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

Factorización de Cholesky

Sea

$$\begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}$$

Entonces

$$\begin{pmatrix} u_{11} & 0 \\ u_{12} & u_{22} \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} \\ 0 & u_{22} \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}$$

$$\begin{aligned} u_{11}^2 &= 1 \\ u_{11}u_{12} &= 2 \\ u_{12}^2 + u_{22}^2 &= 4 \end{aligned}$$

Se deduce que

$$\begin{aligned} u_{11} &= 1 \\ u_{12} &= 2 \\ u_{22} &= 0 \end{aligned}$$

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

Existe U pero no la factorización de Cholesky, U no es inversible.

Factorización de Cholesky

Sea

$$\begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix}$$

Entonces

$$\begin{pmatrix} u_{11} & 0 \\ u_{12} & u_{22} \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} \\ 0 & u_{22} \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix}$$

$$\begin{aligned} u_{11}^2 &= 1 \\ u_{11}u_{12} &= 2 \\ u_{12}^2 + u_{22}^2 &= 3 \end{aligned}$$

Se deduce que

$$\begin{aligned} u_{11} &= 1 \\ u_{12} &= 2 \\ u_{22} &= \sqrt{-1} \end{aligned}$$

Entonces no existe la factorización de Cholesky de A

Factorización de Cholesky

Caso general:

$$\begin{bmatrix} u_{11} & & & & & & \\ \vdots & & & & & & \\ u_{1k} & \cdots & u_{kk} & & & & \\ \vdots & & & & & & \\ u_{1j} & \cdots & u_{kj} & \cdots & u_{jj} & & \\ \vdots & & & & & & \\ u_{1n} & \cdots & u_{kn} & \cdots & u_{jn} & \cdots & u_{nn} \end{bmatrix} \begin{bmatrix} u_{11} & \cdots & u_{1k} & \cdots & u_{1j} & \cdots & u_{1n} \\ & & & & & & \vdots \\ & & u_{kk} & \cdots & u_{kj} & \cdots & u_{kn} \\ & & & & & & \vdots \\ & & & & u_{jj} & \cdots & u_{jn} \\ & & & & & & \vdots \\ & & & & & & u_{nn} \end{bmatrix}$$

El producto de la fila 1 de U^t por la columna 1 de U da: $u_{11}^2 = a_{11}$ luego:

$$u_{11} = \sqrt{a_{11}}$$

El producto de la fila 1 de U^t por la columna j de U da $u_{11}u_{1j} = a_{1j}$ luego:

$$u_{1j} = \frac{a_{1j}}{u_{11}}, \quad j = 2, \ldots, n$$

Factorización de Cholesky

Al hacer el producto de la fila 2 de U^t por la columna 2 de U, se puede calcular u_{22}. Al hacer el producto de la fila 2 de U^t por la columna j de U se puede calcular u_{2j}. El calculo de U se hace por fila.
Supongamos ahora que se conocen los elementos de las filas $1, 2, \ldots, k-1$ de U y se desea calcular los elementos de la fila k de U.
El producto de la fila k de U^t por la columna k de U da:

$$\sum_{i=1}^{k} u_{ik}^2 = a_{kk}, \quad \sum_{i=1}^{k-1} u_{ik}^2 + u_{kk}^2 = a_{kk}$$

Luego

$$u_{kk} = \sqrt{a_{kk} - \sum_{i=1}^{k-1} u_{ik}^2}, \quad k = 2, \ldots, n$$

Factorización de Cholesky

El producto de la fila k de U^t por la columna j de U da:

$$\sum_{i=1}^{k} u_{ik} u_{ij} = a_{kj}$$

Luego

$$u_{kj} = \frac{a_{kj} - \sum_{i=1}^{k-1} u_{ik} u_{ij}}{u_{kk}}, \quad k = 2, \ldots, n \; j = k+1, \ldots, n$$

Utilizando el producto de matrices las formulas anteriores se pueden escribir así:
$t = a_{kk} - U(1:k-1,k)^t U(1:k-1,k)$
$u_{kk} = \sqrt{t}$
$u_{kj} = \frac{a_{kj} - U(1:k-1,k)^t U(1:k-1,j)}{u_{kk}}$

Factorización de Cholesky

Podemos almacenar los valores u_{kk} y u_{kj} sobre los antiguos valores de a_{kk} y a_{kj}, de esta forma nos queda: $t = a_{kk} - U(1:k-1,k)^t U(1:k-1,k)$
$a_{kk} = \sqrt{t}$
$a_{kj} = \frac{a_{kj} - U(1:k-1,k)^t U(1:k-1,j)}{a_{kk}}$
El esquema del algoritmo para la factorización de Cholesky nos queda así:

```
datos: A, e
   para k = 1, ..., n
         cálculo de t
         si t < e   entonces salir
         akk = sqrt(t)
         para j = k + 1, ..., n
            cálculo de akj
   finpara j
finpara k
```

Factorización de Cholesky

```
function [U, ind] = Cholesky(A)
eps = 1.0e-8
n = size(A,1)
U = zeros(n,n)
for k = 1:n
    t = A(k,k) - U(1:k-1,k)'*U(1:k-1,k)
    if t <= eps
       printf("Matriz no definida positiva.\n")
       ind = 0
       return
    end
    U(k,k)= sqrt(t)
    for j = k+1:n
       U(k,j) = ( A(k,j) - U(1:k-1,k)'*U(1:k-1,j) )/U(k,k)
    end
end
ind = 1
endfunction
```