Computational Linear Algebra

SUSTech-CPC div1

Yee_172

2019.01.16

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}, B = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$$

$$\text{Let } C = A \times B = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix},$$

$$\text{where } c_{ij} = \sum_{k=1}^{3} a_{ik} b_{kj}$$

```
We have A[][], B[][], C[][]

We initialize C first

for (int i = 0; i < n; ++i)
    for (int j = 0; j < n; ++j)
        for (int k = 0; k < n; ++k)
            C[i][j] += A[i][k] * B[k][j];

Change the order of i, j, k is also correct

for (int k = 0; k < n; ++k)
    for (int i = 0; i < n; ++i)
        for (int j = 0; j < n; ++j)
            C[i][j] += A[i][k] * B[k][j];
```

$O(n^3)$

Optimize a little

```
for (int k = 0; k < n; ++k)
    for (int i = 0; i < n; ++i)
        if (sign(A[i][k]) != 0)
            for (int j = 0; j < n; ++j)
                C[i][j] += A[i][k] * B[k][j]
```

Same as the Fast Power Algorithm, we can also fast power other types of element

```
ElementType fast_power(ElementType a, int b)
{
    ElementType r = ElementType_Identity;
    for (; b; b >>= 1, a = __mul__(a, a))
        if (b & 1) r = __mul__(r, a);
    return r;
}
```

Gaussian Elimination with Backward Substitution

To solve the $n \times n$ linear system

$$\begin{aligned}
E_1 &: \quad a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = a_{1,n+1} \\
E_2 &: \quad a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = a_{2,n+1} \\
&\vdots \qquad \vdots \qquad \vdots \qquad\qquad \vdots \qquad\quad \vdots \\
E_n &: \quad a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = a_{n,n+1}
\end{aligned}$$

INPUT number of unknowns and equations n; augmented matrix $A = [a_{ij}]$, where $1 \leq i \leq n$ and $1 \leq j \leq n + 1$.

OUTPUT solution $x_1, x_2, \ldots, x_n$ or message that the linear system has no unique solution.

Step 1 For $i = 1, \ldots, n - 1$ do Steps 2–4. (*Elimination process.*)

Step 2 Let p be the smallest integer with $i \leq p \leq n$ and $a_{pi} \neq 0$.
If no integer p can be found
then OUTPUT ('no unique solution exists');
STOP.

Step 3 If $p \neq i$ then perform $(E_p) \leftrightarrow (E_i)$.

Step 4 For $j = i + 1, \ldots, n$ do Steps 5 and 6.

Step 5 Set $m_{ji} = a_{ji}/a_{ii}$.

Step 6 Perform $(E_j - m_{ji}E_i) \rightarrow (E_j)$;

Step 7 If $a_{nn} = 0$ then OUTPUT ('no unique solution exists');
STOP.

Step 8 Set $x_n = a_{n,n+1}/a_{nn}$. (*Start backward substitution.*)

Step 9 For $i = n-1, \ldots, 1$ set $x_i = \left[a_{i,n+1} - \sum_{j=i+1}^{n} a_{ij}x_j\right] \Big/ a_{ii}$.

Step 10 OUTPUT $(x_1, \ldots, x_n)$; (*Procedure completed successfully.*)
STOP.

Gauss-Jordan Method: This method is described as follows. Use the ith equation to eliminate not only x_i from the equations $E_{i+1}, E_{i+2}, \ldots, E_n$, as was done in the Gaussian elimination method, but also from $E_1, E_2, \ldots, E_{i-1}$. Upon reducing $[A, \mathbf{b}]$ to:

$$\begin{bmatrix} a_{11}^{(1)} & 0 & \ldots & 0 & \vdots & a_{1,n+1}^{(1)} \\ 0 & a_{22}^{(2)} & \ddots & \vdots & \vdots & a_{2,n+1}^{(2)} \\ \vdots & \ddots & \ddots & 0 & \vdots & \vdots \\ 0 & \vdots & 0 & a_{nn}^{(n)} & \vdots & a_{n,n+1}^{(n)} \end{bmatrix},$$

the solution is obtained by setting

$$x_i = \frac{a_{i,n+1}^{(i)}}{a_{ii}^{(i)}},$$

for each $i = 1, 2, \ldots, n$. This procedure circumvents the backward substitution in the Gaussian elimination.

$$
\begin{aligned}
4x_1 - 1x_2 + 1x_3 &= 8, \\
2x_1 + 5x_2 + 2x_3 &= 3, \\
1x_1 + 2x_2 + 4x_3 &= 11.
\end{aligned}
$$

two-digit rounding arithmetic

$$\begin{bmatrix} 4 & -1 & 1 & 8 \\ 2 & 5 & 2 & 3 \\ 1 & 2 & 4 & 11 \end{bmatrix} \rightarrow \begin{bmatrix} 4 & -1 & 1 & 8 \\ 0 & 5.5 & 1.5 & -1 \\ 0 & 2.3 & 3.8 & 9 \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} 4 & 0 & 1.3 & 7.8 \\ 0 & 5.5 & 1.5 & -1 \\ 0 & 0 & 3.2 & 9.4 \end{bmatrix} \rightarrow \begin{bmatrix} 4 & 0 & 0 & 3.9 \\ 0 & 5.5 & 0 & -5.4 \\ 0 & 0 & 3.2 & 9.4 \end{bmatrix}$$

$x_1 = 0.98, x_2 = -0.98, x_3 = 2.9.$

Gaussian Elimination with Partial Pivoting

To solve the $n \times n$ linear system

$$\begin{aligned} E_1: &\quad a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = a_{1,n+1} \\ E_2: &\quad a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = a_{2,n+1} \\ &\qquad\qquad \vdots \qquad\qquad\qquad\qquad \vdots \\ E_n: &\quad a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = a_{n,n+1} \end{aligned}$$

INPUT number of unknowns and equations n; augmented matrix $A = [a_{ij}]$ where $1 \le i \le n$ and $1 \le j \le n + 1$.

OUTPUT solution $x_1, \ldots, x_n$ or message that the linear system has no unique solution.

Step 1 For $i = 1, \ldots, n$ set $NROW(i) = i$. (*Initialize row pointer.*)

Step 2 For $i = 1, \ldots, n - 1$ do Steps 3–6. (*Elimination process.*)

Step 3 Let p be the smallest integer with $i \le p \le n$ and
$|a(NROW(p), i)| = \max_{i \le j \le n} |a(NROW(j), i)|$.
(*Notation:* $a(NROW(i), j) \equiv a_{NROW_i, j}$.)

Step 4 If $a(NROW(p), i) = 0$ then OUTPUT ('no unique solution exists');
STOP.

Step 5 If $NROW(i) \neq NROW(p)$ then set $NCOPY = NROW(i)$;
$NROW(i) = NROW(p)$;
$NROW(p) = NCOPY$.
(*Simulated row interchange.*)

Step 6 For $j = i + 1, \ldots, n$ do Steps 7 and 8.

Step 7 Set $m(NROW(j), i) = a(NROW(j), i)/a(NROW(i), i)$.

Step 8 Perform $(E_{NROW(j)} - m(NROW(j), i) \cdot E_{NROW(i)}) \rightarrow (E_{NROW(j)})$.

Step 9 If $a(NROW(n), n) = 0$ then OUTPUT ('no unique solution exists');
STOP.

Step 10 Set $x_n = a(NROW(n), n + 1)/a(NROW(n), n)$.
(*Start backward substitution.*)

Step 11 For $i = n - 1, \ldots, 1$

$$\text{set } x_i = \frac{a(NROW(i), n + 1) - \sum_{j=i+1}^{n} a(NROW(i), j) \cdot x_j}{a(NROW(i), i)}.$$

Step 12 OUTPUT $(x_1, \ldots, x_n)$; (*Procedure completed successfully.*)
STOP.

$$0.03x_1 + 58.9x_2 = 59.2$$
$$5.31x_1 - 6.10x_2 = 47.0$$

Actual solution $[10, 1]$

three-digit rounding arithmetic

$$\begin{bmatrix} 0.03 & 58.9 & 59.2 \\ 5.31 & -6.10 & 47.0 \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} 0.03 & 58.9 & 59.2 \\ 0 & -0.104 \times 10^5 & -0.105 \times 10^5 \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} 0.03 & 0 & -0.300 \\ 0 & -0.104 \times 10^5 & -0.105 \times 10^5 \end{bmatrix}$$

Therefore, we have the solution is $[-6.67, 1.01]$. x_1 is not quite accurate.

partial pivoting and three-digit rounding arithmetic

$$\begin{bmatrix} 0.03 & 58.9 & 59.2 \\ 5.31 & -6.10 & 47.0 \end{bmatrix} \rightarrow \begin{bmatrix} 5.31 & -6.10 & 47.0 \\ 0.03 & 58.9 & 59.2 \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} 5.31 & -6.10 & 47.0 \\ 0 & 58.9 & 58.9 \end{bmatrix} \rightarrow \begin{bmatrix} 5.31 & 0 & 53.1 \\ 0 & 58.9 & 58.9 \end{bmatrix}$$

Therefore, we have the solution is $[10.0, 1.00]$. Accurate.

$\mathbf{b}$ could be a matrix

Calculating the inverse of a matrix

$$AX = I$$

Optimization: LU decomposition

Lagrange Interpolating Polynomials

The problem of determining a polynomial of degree one that passes through the distinct points (x_0, y_0) and (x_1, y_1) is the same as approximating a function f for which $f(x_0) = y_0$ and $f(x_1) = y_1$ by means of a first-degree polynomial **interpolating**, or agreeing with, the values of f at the given points. Using this polynomial for approximation within the interval given by the endpoints is called polynomial **interpolation**.

Define the functions

$$L_0(x) = \frac{x - x_1}{x_0 - x_1} \quad \text{and} \quad L_1(x) = \frac{x - x_0}{x_1 - x_0}.$$

The linear **Lagrange interpolating polynomial** through (x_0, y_0) and (x_1, y_1) is

$$P(x) = L_0(x)f(x_0) + L_1(x)f(x_1) = \frac{x - x_1}{x_0 - x_1}f(x_0) + \frac{x - x_0}{x_1 - x_0}f(x_1).$$

Note that

$$L_0(x_0) = 1, \quad L_0(x_1) = 0, \quad L_1(x_0) = 0, \quad \text{and} \quad L_1(x_1) = 1,$$

Some of the common **closed Newton-Cotes formulas** with their error terms are listed. Note that in each case the unknown value ξ lies in (a, b).

$n = 1$: Trapezoidal rule

$$\int_{x_0}^{x_1} f(x)\,dx = \frac{h}{2}[f(x_0) + f(x_1)] - \frac{h^3}{12} f''(\xi), \quad \text{where} \quad x_0 < \xi < x_1. \tag{4.25}$$

$n = 2$: Simpson's rule

$$\int_{x_0}^{x_2} f(x)\,dx = \frac{h}{3}[f(x_0) + 4f(x_1) + f(x_2)] - \frac{h^5}{90} f^{(4)}(\xi), \quad \text{where} \quad x_0 < \xi < x_2. \tag{4.26}$$

$n = 3$: Simpson's Three-Eighths rule

$$\int_{x_0}^{x_3} f(x)\,dx = \frac{3h}{8}[f(x_0) + 3f(x_1) + 3f(x_2) + f(x_3)] - \frac{3h^5}{80} f^{(4)}(\xi), \qquad (4.27)$$

where $x_0 < \xi < x_3$.

$n = 4$:

$$\int_{x_0}^{x_4} f(x)\,dx = \frac{2h}{45}[7f(x_0) + 32f(x_1) + 12f(x_2) + 32f(x_3) + 7f(x_4)] - \frac{8h^7}{945} f^{(6)}(\xi),$$

where $x_0 < \xi < x_4$. (4.28)

THANKS