Linear Algebra

Pranav

May 19, 2020

Handling Matrices in C++

The best way to allocate and deallocation of matrices in C++ is using pointers. Recall that a matrix is represented by a double pointer that points to a memory segment holding a sequwncw of double* pointers. So each double* pointer point to a row in the matrix.

When we declare double** A, this means that A[i] is a pointer to the i+1-th row A[i] and A[i][j] is matrix entry (i,j).

This snippet of code shows how we allocate a matrix of n $\times$ n

```
int n;
double ** A;

A = new double*[n];

for(i=0;i<n;i++)
        A[i] = new double[N];
```

Linear Systems of equations

Gaussian Elimination

THe problem is to solve a system of N equations for N unknowns.

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1N}x_N - b_1 &= 0 \\
a_{21}x_1 + a_{22}x_2 + \cdots + a_{2N}x_N - b_2 &= 0 \\
&\vdots \\
a_{N1}x_1 + a_{N2}x_2 + \cdots + a_{NN}x_N - b_2 &= 0
\end{aligned}
$$

Or in matrix form,

$$\mathbf{Ax\text{ - }b} = 0$$

Where,

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & & \vdots & \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Gaussian Elimination works on two steps. Forward elimination and back substitution.

Backward substitution is done recursively starting from x_n. This can be mathematically expressed as:

$$x_m = \frac{1}{a_{mm}} \left(b_m - \sum_{k=m+1}^{n} a_{mk} x_k \right) \quad m = n-1, n-2, \dots, 1 \tag{1}$$

For N equations in N unknowns, the computation time for Gaussian elimination goes as N^3. Forsparse systems(where most coefficients are xero), this calculation time can be greatly reduced.

Pivoting

Sometimes in the matrix $\mathbf{A}$ we encounter very small numbers or even zero in the diagonal entries. This could be fatal as when we do forward elimination.

Consider the forward elimination for these equations.

$$\begin{aligned} \epsilon x_1 + x_2 + x_3 &= 5 \\ x_1 + x_2 &= 3 \\ x_1 + x_3 &= 4 \end{aligned}$$

$$\begin{aligned} \epsilon x_1 + x_2 + x_3 &= 5 \\ x_2 \left(1 - \frac{1}{\epsilon}\right) + \frac{x_3}{\epsilon} &= 3 - \frac{5}{\epsilon} \\ -(1/\epsilon) x_2 + (1 - 1/\epsilon) x_3 &= 4 - 5/\epsilon \end{aligned}$$

Algorithm for Gaussian Elimination

*** Forward Elimination**

```
DO K=1..N-1
DO I = K+1..N
COEFF = A(I,K)/A(K,K)
DO J = K+1,N
A(I,J) = A(I,J) - COEFF * A(I,J)
ENDO
A(I,K) = COEFF
B(I) = B(I)-COEFF*B(K)
ENDDO
ENDDO
```

*** Back-substitution**

```
X(N) = B(N)/A(N,N)
DO I = N-1...1
SUM = B(I)
DO J= I+1..N
SUM = SUM - A(I,J)*X(J)
ENDDO
X(I) = SUM/A(I,I)
ENDDO
```