# 实验四 高斯消元法(2 课时)

## 一、实验目的

1.掌握高斯选主元消去法公式的用法，适用范围及精确度。

2.通过高斯选主元消去法求矩阵方程的解，验证高斯消去法。

## 二、实验要求

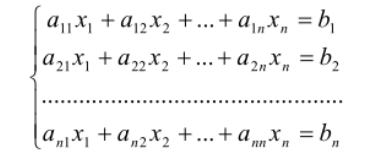
1.写出高斯选主元消去法解线性方程组算法，编写程序上机调试出结果

2.进一步加深对高斯消去法的理解。

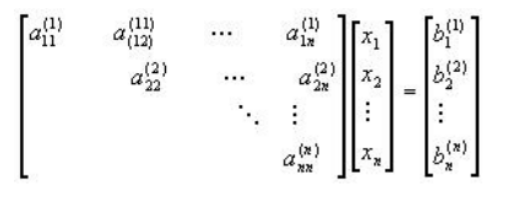
## 三、实验原理

**(一)高斯消元法**

Gauss消去法就是将方程组



通过(n-1)步消元，将上述方程组转化为上三角方程组



再回代求此方程组的解。

**(二)选列主元素高斯消去法**

给定线性方程组 Ax=b，记 A(1)=A，b(1)=b，列主元 Gauss 消去法的具体过程如下:

首先在增广矩阵 B(1)=(A(1),b(1))的第一列元素中，取



然后进行第一步消元得增广矩阵 B(2)=(A(2),b(2))。 再在矩阵 B(2)=(A(2),b(2))的第

二列元素中，取



然后进行第二步消元得增广矩阵B(3)=(A(3),b(3))。按此方法继续进行下去，经过n-1

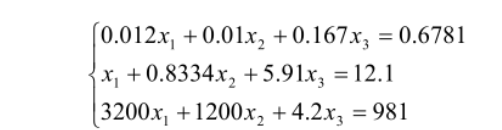
步选主元和消元运算,得到增广矩阵 B(n)=(A(n),b(n)).则方程组 A(n)x=b(n)是与原方程组

等价的上三角形方程组,可进行回代求解.

易证，只要A|≠0,列主元 Gauss 消去法就可顺利进行

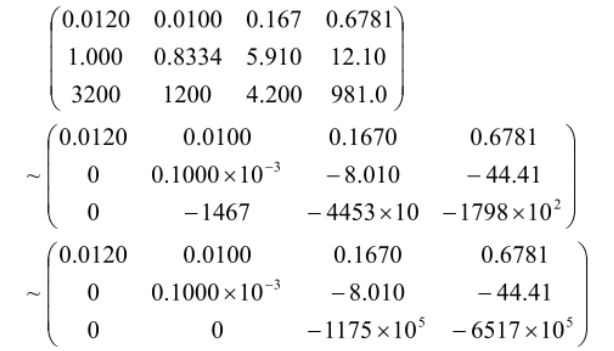
**(三)例子**

采用4位十进制浮点计算，分别用顺序Gauss消去法和列主元Gauss 消去法求解线性方程组:



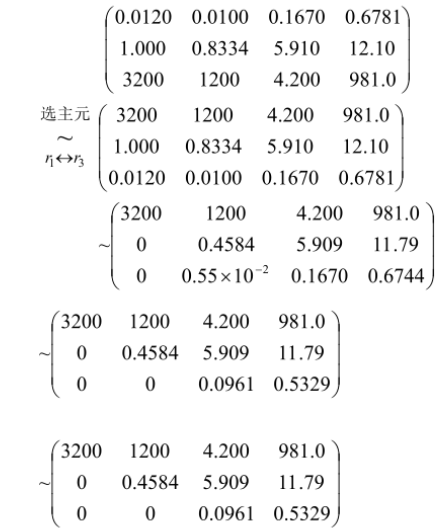
方程组具有四位有效数字的精确解为x1\*=17.46，x2\*=-45.76，x3\*=5.546

解(1)用顺序 Gauss 消去法求解，消元过程为



回代得:x3=5.546，x2=100.0，x1=-104.0

(2)用列主元 Gauss 消去法求解，消元过程为

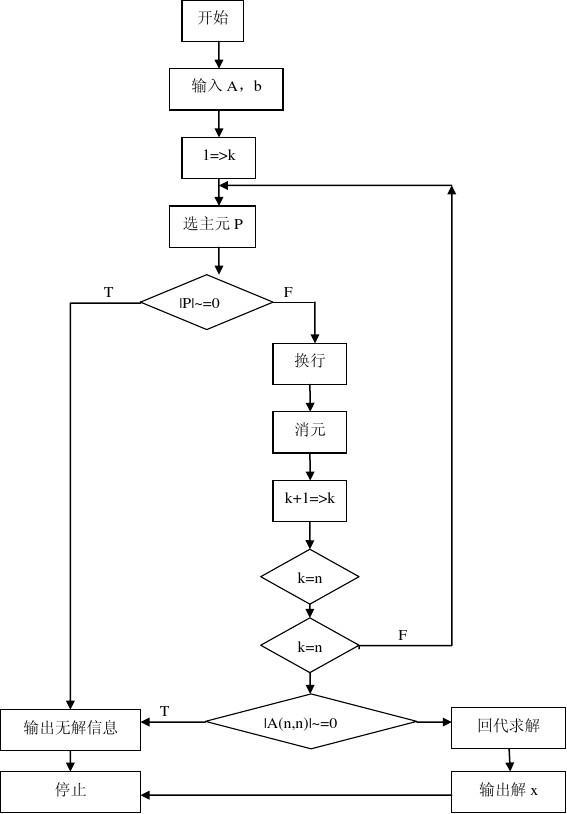


回代得:x3=5.545，x2=-45.77，x1=17.46

## 四、实验内容

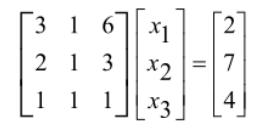
**(一)算法流程图**

高斯列主元消去法N-S图



**(二)编程作业**

编写选列主元的高斯消去法。求出下列线性方程组Ax=b的解x。



*#include* <iostream>  
*#include* <cmath>  
  
*using namespace* std;  
  
  
*int* main() {  
 *int* n = 3;  
 *double* A[3][3] = {{3, 1, 6},  
 {2, 1, 3},  
 {1, 1, 1}}; *// 系数矩阵 A  
 double* b[] = {2, 7, 4}; *// 常数项向量 b   
 // 高斯列主元消去法函数求解线性方程组  
 for* (*int* k = 0; k < n - 1; k++) {  
 *// 寻找主元所在的行  
 int* maxRow = k;  
 *double* maxVal = abs(A[k][k]);  
 *for* (*int* i = k + 1; i < n; i++) {  
 *if* (abs(A[i][k]) > maxVal) {  
 maxRow = i;  
 maxVal = abs(A[i][k]);  
 }  
 }  
 *// 交换最大行和当前行* swap(A[k], A[maxRow]);  
 swap(b[k], b[maxRow]);  
 *// 消元  
 for* (*int* i = k + 1; i < n; i++) {  
 *double* factor = A[i][k] / A[k][k];  
 *for* (*int* j = k; j < n; j++) {  
 A[i][j] -= factor \* A[k][j];  
 }  
 b[i] -= factor \* b[k];  
 }  
 }  
 *// 回代  
 double* x[n];  
 *for* (*int* i = n - 1; i >= 0; i--) {  
 x[i] = b[i];  
 *for* (*int* j = i + 1; j < n; j++) {  
 x[i] -= A[i][j] \* x[j];  
 }  
 x[i] /= A[i][i];  
 }  
 *// 输出解向量* cout << "解向量 x: " << endl;  
 *for* (*int* i = 0; i < n; i++) {  
 cout << "x" << i + 1 << " = " << x[i] << endl;  
 }  
 *return* 0;  
}

