數值方法 Hw2. 實驗線性聯立方程組的演算法

tags: 數值方法 numerical 108-2

- 資工三乙 406262163 黃品翰

程式說明

- 用c++撰寫
- 測試維度2,3,4,7,10,20,30,50,70,100
- `random` 隨機生成亂數，但避開0出現的機會
- `caltime_begin` `caltime_end` 紀錄開始與結束的時間
- `generate` 把隨機數字放到陣列當中

過程

- 先把矩陣換成上三角矩陣
- 從最後一列，找出其他解

程式碼

```cpp
#include <bits/stdc++.h>

using namespace std;
double arr[105][105];
double tmp[105];
double x[105];

double caltime_begin()
{
        clock_t begin = clock();
        return begin;
}

double caltime_end(double begin)
{
        clock_t end = clock();
        double time_spent = (double)(end - begin) / CLOCKS_PER_SEC;
        return time_spent;
}

//-10 ~ 10
int random()
{
        /* 產生亂數 */
        int x = (rand() % 10) - 5;
        if(x == 0)
        {
                x = (rand() % 10) + 1;
        }
        return x;
}

void generate(int n)
{
        for (int i = 1; i <= n; i++)
        {
                for (int j = 1; j <= n+1; j++)
                {
                        arr[i][j] = random();
                }
        }
}

void print_array(int n)
{
        for (int i = 1; i <= n; i++)
        {
                for (int j = 1; j <= n+1; j++)
                {
                        std::cout << arr[i][j] << " ";
                }
                printf("\n");
        }
        printf("------------------------\n");
```

```
55  }
56
57
58
59  int main(int argc, char const *argv[])
60  {
61
62          /* 固定亂數種子 */
63          srand(time(NULL));
64          memset(arr, 0, sizeof(arr));
65          memset(x, 0, sizeof(x));
66
67          int Dimension[10] = {2,3,4,7,10,20,30,50,70,100};
68
69          for (unsigned int a = 0; a < 10; a++)
70          {
71                  unsigned int n = Dimension[a];
72                  string filename = to_string(n) + ".out";
73                  #ifdef DBG
74                  freopen(filename.c_str(), "w", stdout);
75                  #endif
76
77                  cout << "Dimension +" << Dimension[a] << ":" << '\n';
78
79                  generate(n);
80                  print_array(n);
81
82                  double begin = caltime_begin();
83                  //change upper matrix
84                  for (int k = 1; k <= n - 1; k++)
85                  {
86                          for(int i = k+1; i <= n; i++)
87                          {
88                                  tmp[i] = arr[i][k] / arr[k][k];
89                                  for(int j = 1; j <= n+1; j++)
90                                  {
91                                          arr[i][j] = arr[i][j] - (tmp[i] * arr[k][
92                                  }
93                          }
94                          print_array(n);
95                  }
96
97                  // solve
98                  double sum = 0;
99                  for(int i = n; i >= 1; i--)
100                 {
101                         sum = 0;
102                         for(int j = i+1; j <= n; j++)
103                         {
104                                 sum += arr[i][j] * x[j];
105                         }
106                         x[i] = (arr[i][n + 1] - sum) / arr[i][i];
107                 }
108                 double solve_total_time = caltime_end(begin);
109
```

```
110                 //print answer
111                 for(int i = 1; i <= n; i++)
112                 {
113                         cout << "x" << i << ": " << x[i] << '\n';
114                 }
115                 cout << "Calculate time: " << solve_total_time << "s" << '\n';
116         }
117
118
119         return 0;
120 }
```

程式結果

- 二維

```
Dimension +2:
-3 -2 -5
-4 2 4
-------------------------
-3 -2 -5
0 4.66667 10.6667
-------------------------
x1: 0.142857
x2: 2.28571
Calculate time: 0s
```

- 三維

```
Dimension +3:
1 -2 -4 5
-3 -4 -2 -5
2 2 -1 -5
-------------------------
1 -2 -4 5
0 -10 -14 10
0 6 7 -15
-------------------------
1 -2 -4 5
0 -10 -14 10
0 0 -1.4 -9
-------------------------
x1: 10.7143
x2: -10
x3: 6.42857
Calculate time: 0s
```

- 四維

```
Dimension +4:
2 -5 -1 -4 -1
2 2 -2 -5 -5
8 -2 -2 4 -2
-2 1 2 3 -3
-------------------------
2 -5 -1 -4 -1
0 7 -1 -1 -4
0 18 2 20 2
0 -4 1 -1 -4
-------------------------
2 -5 -1 -4 -1
0 7 -1 -1 -4
0 0 4.57143 22.5714 12.2857
0 0 0.428571 -1.57143 -6.28571
-------------------------
2 -5 -1 -4 -1
0 7 -1 -1 -4
0 0 4.57143 22.5714 12.2857
0 0 0 -3.6875 -7.4375
-------------------------
x1: -3.40678
x2: -1.32203
x3: -7.27119
x4: 2.01695
Calculate time: 0s
```

- 七維

Dimension +7:
-2 -3 3 -1 3 1 1 5
-2 -1 -5 -1 -3 2 -3 -2
2 1 -2 -3 -3 -4 2 -3
1 2 4 -2 -5 -1 2 2
7 -4 -2 2 2 -1 3 1
4 5 -3 1 3 -5 6 5
4 -5 1 -2 -1 -4 8 1

-2 -3 3 -1 3 1 1 5
0 2 -8 0 -6 1 -4 -7
0 -2 1 -4 0 -3 3 2
0 0.5 5.5 -2.5 -3.5 -0.5 2.5 4.5
0 -14.5 8.5 -1.5 12.5 2.5 6.5 18.5
0 -1 3 -1 9 -3 8 15
0 -11 7 -4 5 -2 10 11

-2 -3 3 -1 3 1 1 5
0 2 -8 0 -6 1 -4 -7
0 0 -7 -4 -6 -2 -1 -5
0 0 7.5 -2.5 -2 -0.75 3.5 6.25
0 0 -49.5 -1.5 -31 9.75 -22.5 -32.25
0 0 -1 -1 6 -2.5 6 11.5
0 0 -37 -4 -28 3.5 -12 -27.5

-2 -3 3 -1 3 1 1 5
0 2 -8 0 -6 1 -4 -7
0 0 -7 -4 -6 -2 -1 -5
0 0 0 -6.78571 -8.42857 -2.89286 2.42857 0.892857
0 0 0 26.7857 11.4286 23.8929 -15.4286 3.10714
0 0 0 -0.428571 6.85714 -2.21429 6.14286 12.2143
0 0 0 17.1429 3.71429 14.0714 -6.71429 -1.07143

-2 -3 3 -1 3 1 1 5
0 2 -8 0 -6 1 -4 -7
0 0 -7 -4 -6 -2 -1 -5
0 0 0 -6.78571 -8.42857 -2.89286 2.42857 0.892857
0 0 0 0 -21.8421 12.4737 -5.84211 6.63158
0 0 0 0 7.38947 -2.03158 5.98947 12.1579
0 0 0 0 -17.5789 6.76316 -0.578947 1.18421

-2 -3 3 -1 3 1 1 5
0 2 -8 0 -6 1 -4 -7
0 0 -7 -4 -6 -2 -1 -5
0 0 0 -6.78571 -8.42857 -2.89286 2.42857 0.892857
0 0 0 0 -21.8421 12.4737 -5.84211 6.63158
0 0 0 0 8.88178e-016 2.18843 4.01301 14.4014
0 0 0 0 0 -3.2759 4.12289 -4.15301

-2 -3 3 -1 3 1 1 5
0 2 -8 0 -6 1 -4 -7
0 0 -7 -4 -6 -2 -1 -5
0 0 0 -6.78571 -8.42857 -2.89286 2.42857 0.892857
0 0 0 0 -21.8421 12.4737 -5.84211 6.63158

```
0 0 0 0 8.88178e-016 2.18843 4.01301 14.4014
0 0 0 0 1.32953e-015 0 10.13 17.4048
-------------------------
x1: 0.963185
x2: 1.29641
x3: -0.127988
x4: -2.46418
x5: 1.19572
x6: 3.4301
x7: 1.71813
Calculate time: 0.002s
```

- 高維度的資料太大，提供連結參考
    - https://github.com/william31212/numerical/tree/master/HW2
      (https://github.com/william31212/numerical/tree/master/HW2)
- 觀察

> 越高維度算出來的隨機值，較有可能會遇到線性相關，導致求不出解
> 當維度較大的時，下三角矩陣可能沒辦法完全消成0