原 高斯消元算法

2018年08月30日 17:47:37 数学家是我理想 阅读数 614

一、基本描述

高斯消元主要用来**求解线性方程组**，也可以求解**矩阵的秩**、**矩阵的逆**。它的时间复杂度是**n^3**，主要与方程组的个数， 的个数有关

二、算法过程

其实高斯消元的过程就是**手算解方程组的过程**，分两步：1.加减消元，2.回代求未知数值。下面通过一个例子来具体说明

三、示例

有样一个**三元一次**方程组：

$$\begin{cases} 2x+y+z=1 & ① \\ 6x+2y+z=-1 & ② \\ -2x+2y+z=7 & ③ \end{cases}$$

1.消去x

①×(-3)+②得到 0x-y-2z=-4

①+③得到 0x+3y+2z=8

从而得到

$$\begin{cases} 2x+y+z=1 & ① \\ 0x-y-2z=-4 & ② \\ 0x+3y+2z=8 & ③ \end{cases}$$

2.消去y

②×3+③得到 0x+0y-4z=-4

进而得到

$$\begin{cases} 2x+y+z=1 & ① \\ 0x-y-2z=-4 & ② \\ 0x+0y-4z=-4 & ③ \end{cases}$$

至此，我们已经求出了z=1，下一步进行回代

3.回代求解x

将z=1,y=2带入①，求得x=-1

最终得到

$$\begin{cases} x=-1 \\ y=2 \\ z=1 \end{cases}$$

四、再说算法

对于方程组，其系数是具体存在矩阵(数组)里的，下面在给出实际在矩阵中的表示

$$\begin{bmatrix} x & y & z & val \\ 2 & 1 & 1 & 1 \\ 6 & 2 & 1 & -1 \\ -2 & 2 & 1 & 7 \end{bmatrix}$$

1.消去x

$$\begin{bmatrix} x & y & z & val \\ 2 & 1 & 1 & 1 \\ 0 & -1 & -2 & -4 \\ 0 & 3 & 2 & 8 \end{bmatrix}$$

2.消去y

$$\begin{bmatrix} x & y & z & val \\ 2 & 1 & 1 & 1 \\ 0 & -1 & -2 & -4 \\ 0 & 0 & -4 & -4 \end{bmatrix}$$

3.回代求解y

回代的时候，记录各个变量的结果将保存在另外一个数组当中，故保存矩阵的数组值不会发生改变，该矩阵主要进行消元过程。

五、有几个问题

Q1：系数不一定是整数啊？
A1：这时候数组就要用到浮点数了！不能是整数！

Q2：什么时候无解啊？
A2：消元完了，发现有一行系数都为0，但是常数项不为0，当然无解啦！比如：

$$\begin{bmatrix} x & y & z & val \\ 2 & 1 & 1 & 1 \\ 0 & -1 & -2 & -4 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

Q3：什么时候多解啊？
A3：消元完了，发现有好几行系数为0，常数项也为0，这样就多解了！有几行为全为0，就有几个自由元，所谓自由元，就是这些变量的值可以随意种情况可以满足给出的方程组，比如：

$$\begin{bmatrix} x & y & z & val \\ 2 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Q4：那什么时候解是唯一的啊！
A4：您做一下排除法，不满足2和3的，不就是解释唯一的嘛！其实也就是说我们的系数矩阵可以化成上三角阵。

六、代码实现

c/c++高斯消元模板

```
#include<stdio.h>
#include<algorithm>
#include<iostream>
#include<string.h>
#include<math.h>
using namespace std;
const int MAXN=50;
int a[MAXN][MAXN];//增广矩阵
int x[MAXN];//解集
bool free_x[MAXN];//标记是否是不确定的变元
int gcd(int a,int b) {
    if(b == 0)
        return a;
    else
        return gcd(b,a % b);
}
inline int lcm(int a,int b) {
    return a / gcd(a,b) * b;//先除后乘防溢出
}
//高斯消元法解方程组(Gauss-Jordan elimination).(-2表示有浮点数解，但无整数解，
//-1表示无解，0表示唯一解，大于0表示无穷解，并返回自由变元的个数)
//有equ个方程，var个变元。增广矩阵行数为equ,分别为0到equ-1,列数为var+1,分别为0到var.
int Gauss(int equ,int var) {
```

```
    int max_r;// 当前这列绝对值最大的行.
    int col;//当前处理的列
    int ta,tb;
    int LCM;
    int temp;
    int free_x_num;
    int free_index;

    for(int i = 0;i <= var;i++){
        x[i] = 0;
        free_x[i] = true;
    }

    //转换为阶梯阵.
    col = 0; // 当前处理的列
    for(k = 0;k < equ && col < var;k++,col++) {// 枚举当前处理的行.
    // 找到该col列元素绝对值最大的那行与第k行交换.(为了在除法时减小误差)
        max_r = k;
        for(i = k + 1;i < equ;i++)
            if(abs(a[i][col]) > abs(a[max_r][col]))
                max_r=i;

        if(max_r != k)// 与第k行交换.
            for(j = k;j < var + 1;j++)
                swap(a[k][j],a[max_r][j]);

        if(a[k][col] == 0){// 说明该col列第k行以下全是0了，则处理当前行的下一列.
            k--;
            continue;
        }
        for(i = k + 1;i < equ;i++) {// 枚举要删去的行.
            if(a[i][col] != 0) {
                LCM = lcm(abs(a[i][col]),abs(a[k][col]));
                ta = LCM / abs(a[i][col]);
                tb = LCM / abs(a[k][col]);
                if(a[i][col] * a[k][col] < 0)
                    tb =- tb;//异号的情况是相加
                for(j = col;j < var + 1;j++)
                    a[i][j] = a[i][j] * ta - a[k][j] * tb;
            }
        }
    }
    // 1. 无解的情况: 化简的增广阵中存在(0, 0, ..., a)这样的行(a != 0).
    for (i = k; i < equ; i++) // 对于无穷解来说，如果要判断哪些是自由变元，那么初等行变换中的交换就会影响，则要记录交换.
        if (a[i][col] != 0)
            return -1;

        // 2. 无穷解的情况: 在var * (var + 1)的增广阵中出现(0, 0, ..., 0)这样的行，即说明没有形成严格的上三角阵.
        // 且出现的行数即为自由变元的个数.
    if (k < var)
        return var - k; // 自由变元有var - k个.

        // 3. 唯一解的情况: 在var * (var + 1)的增广阵中形成严格的上三角阵.
        // 计算出Xn-1, Xn-2 ... X0.
    for (i = var - 1; i >= 0; i--) {
        temp = a[i][var];
        for (j = i + 1; j < var; j++)
            if (a[i][j] != 0)
                temp -= a[i][j] * x[j];
        if (temp % a[i][i] != 0)
            return -2; // 说明有浮点数解，但无整数解.
        x[i] = temp / a[i][i];
    }
    return 0;
}
int main(void) {
    int i, j;
    int equ,var;
```

```
        memset(a, 0, sizeof(a));
        for (i = 0; i < equ; i++)
            for (j = 0; j < var + 1; j++)
                scanf("%d", &a[i][j]);

        int free_num = Gauss(equ,var);
        if (free_num == -1)
            printf("无解!\n");
        else if (free_num == -2)
            printf("有浮点数解，无整数解!\n");
        else if (free_num > 0) {
            printf("无穷多解! 自由变元个数为%d\n", free_num);
            for (i = 0; i < var; i++) {
                if (free_x[i])
                    printf("x%d 是不确定的\n", i + 1);
                else
                    printf("x%d: %d\n", i + 1, x[i]);
            }
        } else
            for (i = 0; i < var; i++)
                printf("x%d: %d\n", i + 1, x[i]);
            printf("\n");
    }
    return 0;
}
```


想对作者说点什么

高斯消元(and牛顿迭代法)详解及模板 阅读数 1113

高斯消元的最主要作用为求解线性方程组，说白了，就是解方程......消元的过程就是模拟人手算解方程的过程... 博文 来自： 风灯记的博客

高斯消元 阅读数 739

题目背景Gauss消元题目描述给定一个线性方程组，对其求解输入输出格式输入格式： 第一行，一个正整数 n第二至 ... 博文 来自： sslz_fsy的博客

高斯消元快速入门 阅读数 2万+

高斯消元快速入门一、基本描述学习一个算法/技能，首先要知道它是干什么的，那么高斯消元是干啥的呢？高斯消... 博文 来自： Pengwill's Blog

高斯消元法 阅读数 1937

高斯消去法是一种常用的求解线性方程组的方法，通过逐次消元后，在回代求解，实际计算中常用的一种方法。顺序... 博文 来自： tyxr

高斯消元法(python实现) 阅读数 1072

今天在看一篇论文是讲fastmultipolemethod的。好久之前就看过相关的知识，主要是用来加速矩阵和向量之间的乘... 博文 来自： bengepai的博客

Java的高斯消元法 阅读数 38

算法虽然会，但是用数组的形式进行求值是一件非常非常恶心的事情。更恶心的是，公司里面的逻辑居然要用到方程... 博文 来自： 柠檬不萌

计算行列式（高斯消元？+Java+工具） 阅读数 860

功能：计算行列式并输出用法：首先输入行列式的阶数，然后以输入行列式内容。例如：输入：412-1323-12-1110... 博文 来自： ACM_Fish的博客

高斯消元法解线性方程--Java实现 阅读数 2188

我想当你看到这篇文章的时候，已经对高斯消元法进行了一些了解了，如果还有不明白的地方，请大家自行百度，我... 博文 来自： 郑小小小小源的博客

高斯消元法求解方程组 阅读数 1611