计算方法第三次上机作业程序文档

5.11.2021 yawning-lion

一、任务介绍

给定对称正定矩阵矩阵，编制程序执行LDL^T分解，给定右端项，右端项列数不唯一，要求能一次求解。

二、公式说明

1、LDL^T分解

给定$n \times n$矩阵A，i行j列的元素表示为a_{ij} $(0 \leq i,j \leq n)$，单位下三角矩阵的分量相应表示为l_{ij}，对角矩阵对角线上元素表示为d_j,$0 \leq i,j \leq n$。

对于$j = 1,2,3,\ldots,n-1$循环

$$\begin{cases} a_{jk} = a_{jk} - \sum_{m=0}^{k-1} a_{jm} l_{km} \\ l_{jk} = \dfrac{a_{jk}}{d_k} \\ d_j = a_{jj} - \sum_{m=0}^{j-1} a_{jm} l_{jm} \end{cases}$$

完成单位下三角矩阵L和对角矩阵D的求解。

2、方程求解

给定$n \times m$右端矩阵B，j行i列的元素表示为b_j^i $(0 \leq j \leq n, 0 \leq i \leq m)$.

对于循环$i = 0,\ldots,m$,先求解下三角方程$Ly = \overrightarrow{b^i}$，求解格式为

> *进入循环*$j = 0,1,\ldots,n-1$,

> $$y_j = \overrightarrow{b^i}_j - \sum_{l=0}^{j-1} l_{jl} y_l,$$
>
> *对于*$l = 0, 1, \ldots, n-1$*,*
>
> $$z_l = \frac{y_l}{d_l},$$
>
> *接下来求解上三角形方程*$L^T x = z$*,进入循环*$j = n-1, \ldots, 0$*,完成右端项为*$\overrightarrow{b^i}$ *时的方程求解,*
>
> $$x_j = z_j - \sum_{l=j+1}^{n-1} l_{lj} z_l,$$

完成求解。

三、程序说明

1、运行环境

程序编译环境为mingw-w64-v8.0.0,g++,IDE为vscode，程序文档利用markdown写作。

2、使用说明

- **输入规范**

  本次输入要求放在与代码同级的code文件夹下，格式为txt文件，需要命名为Input.txt。

  第一行为一个整数，为对称正定矩阵的行数n。

  接下来依次n行，为矩阵的各行元素，一行内的元素以空格隔开，一行内n个元素。

  接下来一行为一个整数，为右端项的列数m.

  接下来一次n行，为右端项的各行元素，一行内的元素以空格隔开，一行内m个元素。

  要求n和m不大于64.

  示例如下

```
4
5 7 6 5
7 10 8 7
6 8 10 9
5 7 9 10
2
23 92
32 128
33 132
31 124
```

- 输出格式

  所有输出在与代码同级的code文件夹下，格式为txt文件，文件名为Output.txt。

  按行输出，元素间以空格隔开。

  示例如下

```
The  element of the matrixL are as follows
1 0 0 0
1.4 1 0 0
1.2 -2 1 0
1 0 1.5 1

The diagonal element of the matrixD are as follows
5 0.2 2 0.5

The element of the solution are as follows
1 4
1 4
1 4
1 4
```

3、程序结构

本程序包含三个头文件

```
#include<stdio.h>
#include<math.h>
#include<stdio.h>
```

随后定义矩阵的最大行列数

```
#define MAXNUM 64
```

定义执行LDL^T分解的函数`LDLTDecompose`，定义给定右端项后的求解函数`SolveB`，定义打印矩阵到文件里的函数`OutputMat`,下面进入主程序。

主程序先从Input.txt中读入行数`n`，定义二维数组存储待分解矩阵`matA` 下三角矩阵`matL`，尺寸都为`MAXNUM`×`MAXNUM`,实际上只用到前`n`×`n`的部分。定义数组存储对角矩阵的对角元素`matD`。

之后对各矩阵初始化，从Input.txt读入`matA`各元素，运行`LDLTDecompose`执行LDL^T分解，分解的结果分别储存进`matL` `matD`中，分别输出进Output.txt。读入右端项列数`m`，读入右端项入`matB`中，执行函数求解，解依然存储入`matB`中，执行输出，程序结束。

4、函数说明

`int LDLTDecompose(double matA[][MAXNUM],double matL[][MAXNUM],double *matD,int n)`

执行LDL^T算法，事实上不返回值，直接修改数组元素。算法实现详见公式说明部分。

`int SolveB(double matL[][MAXNUM],double *matD,double matB[][MAXNUM],int n,int m)`

执行求解算法求解，仅定义了中间数组`yTemp`，最终结果直接写入`matB`。算法实现详见公式说明部分。

`int OutputMat(double matB[][MAXNUM],int n,int m,FILE *pOutput)`

按行列输出矩阵元素进文件中，元素间以空格隔开。

三、算例展示

输入

```
4
5 7 6 5
7 10 8 7
6 8 10 9
5 7 9 10
2
23 92
32 128
33 132
31 124
```

到Output.txt中查看输出

```
The  element of the matrixL are as follows
1 0 0 0
1.4 1 0 0
1.2 -2 1 0
1 0 1.5 1

The diagonal element of the matrixD are as follows
5 0.2 2 0.5

The element of the solution are as follows
1 4
1 4
1 4
1 4
```

一次性解出

```
1 4
1 4
1 4
1 4
```

四、结果分析

本程序在求解部分有少许优化，将最终的结果直接存入存储原右端项的数组中，总的来看空间利用率仍有优化空间。