计算方法编程作业 2 实验报告

崔士强 PB22151743

2024 年 4 月 1 日

1 原理

1.1 初始化

根据实验文档给出的原理，我们可以为解方程组做如下准备：

1. 初始化系数矩阵 $\mathbf{A}$.

   首先创建一个 $n-1$ 行全 0 方阵，再遍历每个位置，对应设置数值.

2. 初始化向量 $\mathbf{b}$.

   由给出的差分方程可以知道，$\mathbf{b}$ 除了最后一个分量外全为 ah^2，最后一个分量为 $ah^2 - (\epsilon + h)$，由此可以初始化 $\mathbf{b}$.

1.2 列主元 Gauss 消元法

在程序中定义一个类`GaussianElimination`用于列主元 Gauss 消元法的计算,其中定义了`void Solve()`成员函数，首先通过另一个成员函数`int SelectMax(int i)`选出最大行，与当前行进行交换，然后消去接下来所有行的首个非零元素，最后回代.

1.3 Gauss-Seidel 迭代

构造函数接收 $\mathbf{A}$, $\mathbf{b}$, 以及算法的容差（用于结束判断）. 初始向量设为全 0 向量，然后通过一个`while`循环进行迭代，如果两次结果各个分量差方和的平方根小于容差则退出循环.

2 实验结果

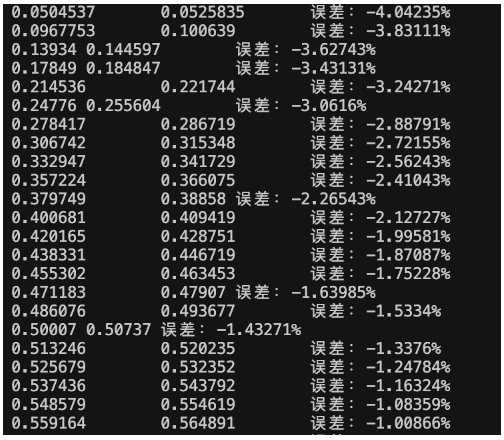
2.1 数值结果

大部分结果误差较小，在 0.1% 以下，另外观察到以下规律：

1. 对于两种方法，均发现越靠近 0，百分比误差越大，原因是更小的数值对误差更敏感.

2. 总体上来说，随着 ϵ 的减小，误差随之减小，然而观察到在最接近 0 的几个点处误差非常大，超过了 3%，推测原因与上一条所述类似.

3. 两种方法的误差没有明显区别.



```
0.0504537       0.0525835       误差： -4.04235%
0.0967753       0.100639        误差： -3.83111%
0.13934 0.144597        误差： -3.62743%
0.17849 0.184847        误差： -3.43131%
0.214536        0.221744        误差： -3.24271%
0.24776 0.255604        误差： -3.0616%
0.278417        0.286719        误差： -2.88791%
0.306742        0.315348        误差： -2.72155%
0.332947        0.341729        误差： -2.56243%
0.357224        0.366075        误差： -2.41043%
0.379749        0.38858 误差： -2.26543%
0.400681        0.409419        误差： -2.12727%
0.420165        0.428751        误差： -1.99581%
0.438331        0.446719        误差： -1.87087%
0.455302        0.463453        误差： -1.75228%
0.471183        0.47907 误差： -1.63985%
0.486076        0.493677        误差： -1.5334%
0.50007 0.50737 误差： -1.43271%
0.513246        0.520235        误差： -1.3376%
0.525679        0.532352        误差： -1.24784%
0.537436        0.543792        误差： -1.16324%
0.548579        0.554619        误差： -1.08359%
0.559164        0.564891        误差： -1.00866%
```

图 1: 靠近 0 处较大的误差

2.2 运行时间

对于几组不同 ϵ 的输入，两种算法的运行时间 (μs) 如下表所示.

ϵ	1	0.1	0.01	0.0001
Gauss 消元	4992	1476	1542	1475
Gauss-Seidel 迭代	386444	147130	5224	4324

总体上来说，Gauss 消元法的运行速度更快，Gauss-Seidel 迭代在处理 ϵ 值更小的矩阵时时间明显缩短，推测原因是此时矩阵谱半径更大.