LAB03

Sparse Matrix & Gauss-Seidel

计算摄影学 2025春夏

2025/3/4

Computation Photography

BEFORE

- 在提交 LAB02 时，无需将动态链接库(.dll)打包上传，在 README 中注明使用的文件名(如opencv_world345d.dll) 和放置位置即可。
- 在提交版本中请注释图片展示窗口。
- 请按照课程作业打分细则完成作业。
- 本次实验无需使用opencv库，你可以使用原来 FDS / ADS /OOP 的 c++ 环境。

BEFORE

- 本次实验内容需要在学在浙大上提交作业，本次作业计入成绩。
- 提交要求：
  - 将**源代码和实验文档**打包
  - 请在 README中注明代码的**编译方法。**
  - 压缩包名称为 Lab3-学号-姓名.zip/7z。
  - 在**3月18日23:59:59**前提交至学在浙大。

实验任务

- **实现一种稀疏矩阵**
  - Compressed Row Storage (CRS)
  - Compressed Sparse Column (CSC)
  - 其他形式
- **基于稀疏矩阵实现Gauss-Seidel迭代法**
  - **Bonus**：Conjugate_Gradient 迭代法

稀疏矩阵

对于一个 $m \times n$ 的矩阵，矩阵中的零元素的数目远多于非零元素的数目，则称该矩阵为稀疏矩阵。

使用二维数组存储稀疏矩阵会导致空间消耗和无效运算。一个经典的稀疏矩阵使用场景是：稀疏边的图。

稀疏矩阵(Cont.)

稀疏矩阵常见的实现方式有：

- 压缩行存储 Compressed Row Storage (CRS)
- 压缩列存储 Compressed Sparse Column (CSC)

稀疏矩阵(Cont.)

对一个 $m \times n$ 矩阵，压缩行存储 (CRS) 的朴素实现包括：

- val：按行优先顺序存储所有非零元素的值。
- col_ind：存储每个非零元素对应的列索引。
- row_ptr：长度为 $m + 1$ 的数组，其中索引为 i ($i < m$)的元素存储第 i 行第一个非零元素在 val 数组中的索引；索引为 m 元素存储矩阵中非零元素的总数。

通过 row_ptr 可以确定每一行的非零元素数量。

稀疏矩阵(Cont.)

类似地，压缩列存储 (CCS) 的朴素实现包括：

- val：按列优先顺序存储所有非零元素的值。
- raw_ind：存储每个非零元素对应的行索引。
- col_ptr：长度为 $n+1$ 的数组，其中索引为 i ($i < n$)的元素存储第 i 行第一个非零元素在 val 数组中的索引；索引为 n 元素存储矩阵中非零元素的总数。

稀疏矩阵(Cont.)

这是一个简单的例子。对于矩阵 $A = \begin{pmatrix} 5 & 1 & 0 & 0 \\ 0 & 8 & 0 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 \end{pmatrix}$，它的CRS对应数组如下：

```
val = {5, 1, 8, 2, 3}
col_id = {0, 1, 1, 3, 2}
row_ptr = {0, 2, 3, 4, 5}
```

CCS 实现对应数组如下：

```
val = {5, 1, 8, 2, 3}
row_id = {0, 0, 1, 3, 2}
col_ptr = {0, 1, 3, 4, 5}
```

稀疏矩阵(Cont.)

本次实验要求基于 sparse.h 手工实现 Sparse 类，实现的 Sparse 类至少具备以下 public 成员函数：

```
double at(int row, int col) const;
void insert(double val, int row, int col);
void initializeFromVector(const Veci& rows,
                          const Veci& cols,
                          const Vecd& vals);
```

```
typedef std::vector<double> Vecd;
typedef std::vector<int> Veci;
```

稀疏矩阵(Cont.)

本次实验要求基于 sparse.h 手工实现 Sparse 类，实现的 Sparse 类至少具备以下 public 成员函数：

1. **查询元素 at(row, col)**：返回 A[row][col]
2. **插入/替换元素 insert(val, row, col)**：A[row][col] = val
3. **初始化 initializeFromVector(rows, cols, vals)**：对 A 进行初始化，A[row[i]][col[i]] = val[i]

且我们约定：

1. 矩阵的非零元素的数量少于 int。
2. 矩阵元素的精度在 1e-10 以内。

Gauss-Seidel迭代法

RECALL 高斯消元求解线性方程组

对于线性方程组 Ax = b，高斯消元法通过对增广矩阵 [A|b] 进行初等行变换，将系数矩阵 A 转化为上三角矩阵，然后通过回代求解。高斯约旦消元直接将 A 转化为对角矩阵，从而不需要繁琐的回代。

期间方阵 A 需要 n 次主消元步骤，每次主消元需要操作 O(n) 行，每次操作需要使用 at 修改 O(n) 次。

- 时间复杂度 O(n^3)
- 代码构造比较复杂，且在消元过程中可能形成稠密矩阵

Gauss-Seidel迭代法(Cont.)

迭代法

通过多步迭代，找到精确解 x^* 的一个近似解 x: $||x - x^*|| < \epsilon$。通过改造原方程为等价迭代形式，并迭代求解，得到解序列。若解序列收敛，则能在有限步内找到满足误差精度的近似解。

迭代格式构造

对于线性方程组 $Ax = b$，改写成

$$Ax - b = 0 \Leftrightarrow x - C * (Ax - b) = x \Leftrightarrow I * x - C * (Ax - b) = x$$

然后就得到迭代格式：

$$x = (I - CA)x + Cb$$

Gauss-Seidel迭代法(Cont.)

迭代格式收敛

对于形似 $x = Bx + f$ 的迭代格式，对任意初始值 $x^{(0)}$收敛，等价于：

存在某种范数 $\|\cdot\|_\diamond$ 满足

$$\|B\|_\diamond < 1$$

结合迭代格式 $x = (I - CA)x + Cb$，我们的目标是找到良好的 C 使得 $||I - CA|| = q$ 在小于 1 的同时尽可能小。称这样的 C 为 A 的广义逆矩阵。

Gauss-Seidel迭代法(Cont.)

Jacobi迭代

Jacobi 迭代法把稀疏矩阵 A 拆分为上三角 U 、下三角 L 和对角矩阵 D。

$$(D + L + U)x - b = 0 \Leftrightarrow Dx = b - (L + U)x$$
$$\Leftrightarrow x = D^{-1}(b - (L + U))x$$

则迭代格式为

$$x = -D^{-1}(L + U)x + D^{-1}b$$

对于稀疏矩阵，上三角和下三角绝大多数元素为零，因而每步迭代是 O（n）的。

求对角矩阵的逆非常容易。
理论课件中把 L + U 写成 R。

Gauss-Seidel迭代法(Cont.)

Gauss-Seidel迭代

Jacobi 迭代法每轮迭代用到了上一轮的信息：

$$x_i^{(k+1)} = -\frac{1}{a_{ii}} \sum_{i \neq j} a_{ij} x_j^{(k)} + \frac{1}{a_{ii}} b_i$$

而Gauss-Seidel则利用了本步迭代中已经计算出的值：

$$x_i^{(k+1)} = -\frac{1}{a_{ii}} \sum_{j<i} a_{ij} x_j^{(k+1)} - \frac{1}{a_{ii}} \sum_{i<j} a_{ij} x_j^{(k)} + \frac{1}{a_{ii}} b_i$$

Gauss-Seidel迭代法(Cont.)

Gauss-Seidel迭代伪代码

```
INPUT: a,b
RETURN: phi

initialize phi
repeat until convergence
        for i from 1 until n do                    #依次更新x0,x1,…,xn
                sigma ← 0
                for j from 1 until n do if j ≠ i then
                        if j ≠ i then
                                sigma ← sigma + a[i][j]phi[j]
                        end if
                end (j - loop)
                phi ← (b[i] - sigma)/a[i][i]
        end (i-loop)
        check if convergence is reached
end (repeat)
```

Gauss-Seidel迭代法(Cont.)

本次实验要求基于 h3_solve.h 所定义的接口实现 Gauss-Seidel 迭代法。

```
Vecd Gauss_Seidel(const Sparse& A,    //系数矩阵
                  const Vecd& b,      //
                  double error);      //收敛标准
```

并以前后两次迭代的解向量的差的无穷范数是否不超过 error 作为是否已经收敛的标准:

$$\| x_{k+1} - x_k \|_\infty <= error$$

Bonus: 共轭梯度法

由余力的同学，可以参考课件和 wiki 求基于 h3_solve.h 所定义的接口实现共轭梯度法。

```
Vecd Conjugate_Gradient(const Sparse& A, const Vecd& b,
                        double error, //收敛标准
                        int kmax);    //最大迭代步数
```

并以残差的2范数平方是否不超过 error 作为是否已经收敛的标准。

测试与评估

实验提供 main.cpp 对稀疏矩阵的实现、G-S 迭代求解进行基础的正确性验证。并提供一个 makefile 文件以便编译。

你需要完善自己的测试样例，并在实验报告中说明。在作业批改时，将会对你的实现进行更复杂的测试，包括最高 30000 × 30000 的随机矩阵求解。

TIPS

- CRS 和 CCS 分别擅长处理什么任务？对于本实验的方程组求解，更适合使用哪种实现方式?
- 初始化的 cols 和 rows 如果不是递增给出的，应该如何处理?
  - HINT：Heap
- **Bonus**：Gauss-Seidel迭代和共轭迭代的收敛条件有什么区别?观察你的实验结果，解释两种方法迭代步数差别的存在。
- 参考网页：
  - https://zhuanlan.zhihu.com/p/389389672
  - https://en.wikipedia.org/wiki/Conjugate_gradient_method

AGAIN

- 在**3月18日23:59:59**前提交至学在浙大。
- 打包格式：
  - Lab3-学号-姓名.zip/7z
    - README
    - h3_solve.h
    - h3_solve.cpp
    - sparse.h
    - sparse.cpp
    - main.cpp

Questions are welcome

计算摄影学 2025春夏

2025/3/4

Computation Photography