数值分析实验报告

张钰晖

2015011372, yuhui-zh15@mails.tsinghua.edu.cn, 185-3888-2881

2017 年 6 月 15 日

1 题目描述

4-1 考虑 10 阶 Hilbert 矩阵，作为系数阵的方程组

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

其中, $\boldsymbol{A}$ 的元素 $a_{ij} = \frac{1}{i+j-1}$, $\boldsymbol{b} = [1, 1/2, ..., 1/10]^T$. 取初始解 $\boldsymbol{x}^{(0)} = \boldsymbol{0}$, 编写程序用 Jacobi 与 SOR 迭代法求解改方程组，将 $||\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}||_\infty < 10^{-4}$ 作为终止迭代的判据.

(1) 分别用 Jacobi 与 SOR($\omega = 1.25$) 迭代法求解，观察收敛情况.

(2) 改变 ω 的值，试验 SOR 迭代法的效果，考察解的准确度.

2 解题思路

在求解线性方程组的时候，当对精度要求不高且要求快速获得大致解的时候，迭代求解法相比直接求解法便发挥出巨大的优势。

根据 Jacobi 迭代法和 SOR 迭代法的伪代码，可以直接编程实现。

在实际使用的时候，应该注意设置 SOR 迭代法的松弛因子 ω 在 0 2 范围内，这是收敛的必要条件。

3 实验结果

算法运行的结果如下，其中 cnt 是迭代次数：

(1) JACOBI: cnt = 349

NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

Jacobi 迭代法不收敛。

SOR: cnt = 187

1.0020 -0.0216 0.0515 -0.0284 -0.0063 -0.0062 -0.0005 0.0018 0.0036 0.0046

SOR 迭代法收敛，迭代次数为 187。

在实验中，经过 349 步迭代，Jacobi 迭代法发散。由书上定理 4.8，若 A 为 n 阶实对称矩阵，且对角线元素 $a_{ii} > 0 (i = 1, 2, ..., n)$，则求解线性方程组 $Ax = b$ 的雅可比迭代法收敛的充要条件是 A 和 $2D - A$ 都正定，其中，D 为取出 A 的对角线元素得到的对角阵。

Hibert 矩阵均为正定矩阵，但

$$2D - A = \begin{bmatrix} 1 & -\frac{1}{2} & \cdots & -\frac{1}{10} \\ -\frac{1}{2} & -\frac{1}{3} & \cdots & -\frac{1}{11} \\ \vdots & \vdots & \ddots & \vdots \\ -\frac{1}{10} & -\frac{1}{11} & \cdots & \frac{1}{19} \end{bmatrix}$$

非正定，取 $x = [1, 1, ..., 1]^T$，显然 $x^T Ax < 0$，因此 Jacobi 迭代法不收敛。

由书上定理 4.12，待求解的线性方程组为 $Ax = b$，若 A 对称正定，且 $0 < \omega < 2$，则相应的 SOR 迭代法收敛。Hilbert 矩阵均对称正定，因此 $0 < \omega < 2$ 时，必有 SOR 迭代法收敛。

(2) SOR: omega = 0.1, cnt = 1038

0.9826 0.1090 -0.0961 -0.0696 -0.0101 0.0303 0.0436 0.0344 0.0094 -0.0255

SOR: omega = 0.2, cnt = 657

0.9886 0.0813 -0.0936 -0.0482 0.0123 0.0432 0.0440 0.0241 -0.0072 -0.0429

SOR: omega = 0.3, cnt = 497

0.9916 0.0658 -0.0880 -0.0344 0.0216 0.0437 0.0376 0.0156 -0.0128 -0.0412

SOR: omega = 0.4, cnt = 397

0.9933 0.0552 -0.0807 -0.0239 0.0250 0.0392 0.0295 0.0090 -0.0137 -0.0340

SOR: omega = 0.5, cnt = 326

0.9945 0.0463 -0.0720 -0.0153 0.0250 0.0325 0.0215 0.0043 -0.0123 -0.0257

SOR: omega = 0.6, cnt = 271

0.9956 0.0381 -0.0619 -0.0082 0.0227 0.0249 0.0144 0.0014 -0.0098 -0.0180

SOR: omega = 0.7, cnt = 224

0.9965 0.0301 -0.0502 -0.0029 0.0183 0.0173 0.0088 -0.0000 -0.0069 -0.0115

SOR: omega = 0.8, cnt = 178

0.9973 0.0222 -0.0369 0.0002 0.0125 0.0103 0.0047 -0.0003 -0.0039 -0.0063

SOR: omega = 0.9, cnt = 121

0.9981 0.0142 -0.0217 0.0004 0.0057 0.0045 0.0022 0.0002 -0.0012 -0.0022

SOR: omega = 1.0, cnt = 2

1 0 0 0 0 0 0 0 0 0

SOR: omega = 1.1, cnt = 115

1.0016 -0.0133 0.0239 -0.0063 -0.0049 -0.0032 -0.0012 0.0002 0.0012 0.0019

SOR: omega = 1.2, cnt = 164

1.0019 -0.0192 0.0427 -0.0201 -0.0071 -0.0052 -0.0010 0.0014 0.0030 0.0039

SOR: omega = 1.3, cnt = 210

1.0021 -0.0239 0.0600 -0.0376 -0.0039 -0.0076 0.0002 0.0021 0.0042 0.0050

SOR: omega = 1.4, cnt = 270

1.0023 -0.0275 0.0749 -0.0575 0.0059 -0.0125 0.0036 0.0015 0.0051 0.0049

SOR: omega = 1.5, cnt = 376

1.0023 -0.0285 0.0834 -0.0766 0.0222 -0.0210 0.0112 -0.0019 0.0070 0.0027

SOR: omega = 1.6, cnt = 493

1.0021 -0.0284 0.0888 -0.0954 0.0434 -0.0356 0.0251 -0.0105 0.0134 -0.0023

SOR: omega = 1.7, cnt = 565

1.0021 -0.0297 0.0995 -0.1210 0.0716 -0.0621 0.0512 -0.0298 0.0325 -0.0139

SOR: omega = 1.8, cnt = 594

1.0022 -0.0334 0.1214 -0.1638 0.1144 -0.1118 0.1039 -0.0722 0.0847 -0.0451

SOR: omega = 1.9, cnt = 116

1.0030 -0.0468 0.1845 -0.2772 0.2165 -0.2364 0.2516 -0.1929 0.2558 -0.1594

由于误差限为 10^{-4} 较大，因此求出的 x 都和真实值有差异。

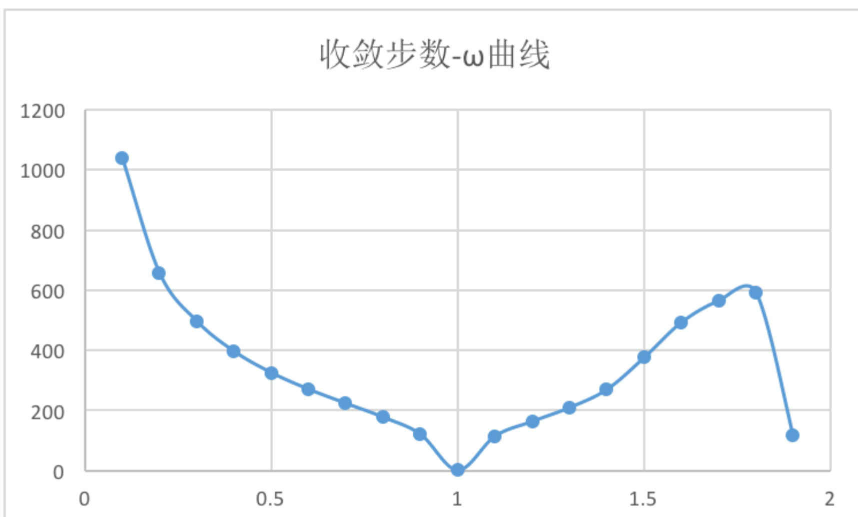

图 1: 收敛步数与松弛因子曲线

经过实验可知，对本题而言，$\omega = 1$ 时，SOR 迭代法收敛速度最快，解也最准确，但是，这是由于数据的特殊性造成的。$\omega < 0$ 或 $\omega > 2$，SOR 迭代法发散。收敛步数和松弛因子的关系曲线见上图。

4 实验心得

通过这个实验，我复习了不动点迭代法的相关知识点，加深了对收敛条件的认知，也实现了 Jacobi 迭代法和 SOR 迭代法的编写。

通过实验，我明白了 Jacobi 迭代法的缺陷，即在一些情况下无法收敛。而 SOR 迭代法可以很好的弥补 Jacobi 迭代法的这一缺点。当然，有时候 SOR 迭代法不一定有意义。比如这一道题目，我们发现 SOR 迭代法在 $\omega = 1$ 的时候效果最好，而此时 SOR 迭代法正是高斯-赛尔德迭代法。这说明 SOR 迭代法增大了不动点迭代法求解线性方程组的选择性，但不一定能增加高斯-赛尔德迭代法的准确性。

同时松弛因子的选择也是一个比较难的问题，根据相关论文指出，$\omega = \frac{2}{1+\sqrt{(1-\rho(J))}}$ 是最佳松弛因子。

5 源代码

```matlab
% Jacobi迭代法，输入向量x、b，矩阵A，迭代误差限epsilon，输出结果x，迭代次数cnt
function [x, cnt] = jacobi(x, A, b, epsilon)
n = size(x, 1);
y = inf(n, 1);
cnt = 0;
% 当不满足终止条件，进行迭代
while max(abs(y - x)) > epsilon
   y = x;
   for i = 1: n
      x(i) = (b(i) - A(i, :) * y) / A(i, i) + y(i);
   end
   cnt = cnt + 1;
end
```

```matlab
% SOR迭代法，输入向量x、b，矩阵A，松弛因子omega，迭代误差限epsilon，输出结果x，迭代次数cnt
function [x, cnt] = sor(x, A, b, omega, epsilon)
n = size(x, 1);
y = inf(n, 1);
cnt = 0;
% 当不满足终止条件，进行迭代
while max(abs(y - x)) > epsilon
   y = x;
   for i = 1: n
      x(i) = (1 - omega) * x(i) + omega * ((b(i) - A(i, :) * x) / A(i, i) + x(i));
   end
   cnt = cnt + 1;
end
```

```matlab
% 生成n阶Hilbert矩阵，输入n，输出H
function [H] = hilbert(n)
% 便捷的生成方法
a = repmat(1: n, n, 1);
H = 1 ./ (a + a' - 1);
end
```

```matlab
% 生成10阶Hilbert矩阵
n = 10;
H = hilbert(n);
b = 1 ./ (1: n)';
x = zeros(n, 1);

% 设置误差限和松弛因子，进行Jacobi迭代和SOR迭代
epsilon = 1e-4;
omega = 1.25;
[x1, cnt1] = jacobi(x, H, b, epsilon);
fprintf('JACOBI: cnt = %d\n', cnt1);
disp(x1');
[x2, cnt2] = sor(x, H, b, omega, epsilon);
fprintf('SOR: cnt = %d\n', cnt2);
disp(x2');

% 调整omega，观测迭代过程
for omega = 0.1: 0.1: 1.9
   [x3, cnt3] = sor(x, H, b, omega, epsilon);
   fprintf('SOR: omega = %.1f, cnt = %d\n', omega, cnt3);
   disp(x3');
end
```

参考文献

[1] 数值分析与算法（第二版），喻文健, 2015.