2024 秋数值代数-实验报告 #3

姓名：李奕萱　　学号：PB22000161

2024 年 11 月 2 日

运行环境：win11,vscode,py3

实验内容与要求

病态线性方程组的求解

问题提出：理论上的分析表明，求解病态的线性方程组是有困难的。实际情况是否如此，具体计算过程中究竟会出现怎样的现象呢？

实验内容：考虑线性方程组 $Hx = b$，其中 H 为 n 阶Hilbert 矩阵，即

$$H = (h_{ij})_{n\times n} \qquad h_{ij} = \frac{1}{i+j-1}, i, j = 1, 2, \cdots, n$$

这是一个著名的病态问题。通过先给定解（例如取 x 的各个分量为 1）, 再计算出右端向量 b 的办法给出一个精确解已知的问题

实验要求：

(1) 分别编写 Doolittle LU 分解法和 Cholesky 分解法的一般程序 (不得使用符号运算)；

(2) 先取阶数 n=6，分别用 LU 分解法、Cholesky 分解法去求解上述的病态方程组 $Hx = b$；分别报告它们的数值结果(即数值解) 以及它们在1-范数下的计算误差。

(3) 再分别取矩阵阶数 n=10 和 19，重复 (2)；仍然用上述的两种计算方法去求解，请分别报告各自的数值结果(即数值解) 以及计算过程中可能出现的问题；

(4) 对 LU 分解，请分别报告 n=6 和 10 时的 LU 分解的分解结果，即给出对应的三角矩阵 L 和 U.

(5) 适当地分析并比较两种计算方法，你能得出什么结论或经验教训.

1 计算结果

- n=6:

x 的精确解	如：(1,1,1,1,1,1)
LU 分解法的数值结果	(1. ,1. ,1., 1., 1., 1.)
LU 分解法在 1-范数下的计算误差	9.20683640437403e-10
Cholesky 分解法的数值结果	(1., 1., 1., 1., 1., 1.)
Cholesky 分解法在 1-范数下的计算误差	2.3233626134100405e-10

表 1: n=6

- n=10:

x 的精确解	如：(1,1,1,1,1,1,1,1,1,1)
LU 分解法的数值结果	(1. 1.00000003 0.99999926 1.00000669 0.99996817 1.00008732 0.99985697 1.00013801 0.99992765 1.00001589)
LU 分解法在 1-范数下的计算误差	0.0004958885204457975
Cholesky 分解法的数值结果	(1. 0.99999998 1.00000042 0.99999599 1.00001997 0.99994333 1.00009538 0.99990591 1.00005024 0.99998879)
Cholesky 分解法在 1-范数下的计算误差	0.00033200483286155436

表 2: n=10

- n=19:

x 的精确解	如：(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)
LU 分解法的数值结果	(1.00000001 0.99999748 1.00012839 0.99738737 1.02742007 0.83181672 1.64320434 -0.57756443 3.48150342 -1.45313159 2.54414427 0.17616262 1.47110881 0.96453137 1.19957548 -0.19082302 2.56961822 0.13691487 1.17800559)
LU 分解法在 1-范数下的计算误差	193.91944513459424
Cholesky 分解法的数值结果	(1.0000001 0.99998416 1.00060052 0.99029064 1.08272459 0.59141146 2.20175296 -0.98548086 2.24431885 2.31638633 -1.61863164 1.52550547 2.47849426 0.70162635 -0.10294406 1.24332358 1.77261918 0.44639229 1.11162575)
Cholesky 分解法在 1-范数下的计算误差	13.954703244325177

表 3: n=19

- n=6:

$$L = \begin{bmatrix} 1. & 0. & 0. & 0. & 0. & 0. \\ 0.5 & 1. & 0. & 0. & 0. & 0. \\ 0.33333333 & 1. & 1. & 0. & 0. & 0. \\ 0.25 & 0.9 & 1.5 & 1. & 0. & 0. \\ 0.2 & 0.8 & 1.71428571 & 2. & 1. & 0. \\ 0.16666667 & 0.71428571 & 1.78571429 & 2.77777778 & 2.5 & 1. \end{bmatrix}$$

$$U = \begin{bmatrix} 1.00000000e+00 & 5.00000000e-01 & 3.33333333e-01 & 2.50000000e-01 & 2.00000000e-01 & 1.66666667e-01 \\ 0.00000000e+00 & 8.33333333e-02 & 8.33333333e-02 & 7.50000000e-02 & 6.66666667e-02 & 5.95238095e-02 \\ 0.00000000e+00 & 0.00000000e+00 & 5.55555556e-03 & 8.33333333e-03 & 9.52380952e-03 & 9.92063492e-03 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 3.57142857e-04 & 7.14285714e-04 & 9.92063492e-04 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 2.26757370e-05 & 5.66893424e-05 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 1.43154905e-06 \end{bmatrix}$$

- n=10:

$$L = \begin{bmatrix} 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0.5 & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0.33333333 & 1. & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0.25 & 0.9 & 1.5 & 1. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0.2 & 0.8 & 1.71428571 & 2. & 1. & 0. & 0. & 0. & 0. & 0. \\ 0.16666667 & 0.71428571 & 1.78571429 & 2.77777778 & 2.5 & 1. & 0. & 0. & 0. & 0. \\ 0.14285714 & 0.64285714 & 1.78571429 & 3.33333333 & 4.09090909 & 3. & 1. & 0. & 0. & 0. \\ 0.125 & 0.58333333 & 1.75 & 3.71212121 & 5.56818182 & 5.65384615 & 3.49999997 & 1. & 0. & 0. \\ 0.11111111 & 0.53333333 & 1.6969697 & 3.95959596 & 6.85314685 & 8.61538461 & 7.46666659 & 3.99999893 & 1. & 0. \\ 0.1 & 0.49090909 & 1.63636364 & 4.11188811 & 7.93006993 & 11.63076923 & 12.59999985 & 9.52940824 & 4.4999617 & 1. \end{bmatrix}$$

$$U = \begin{bmatrix} 1.00000000e+00 & 5.00000000e-01 & 3.33333333e-01 & 2.50000000e-01 & 2.00000000e-01 & 1.66666667e-01 & 1.42857143e-01 & 1.25000000e-01 & 1.11111111e-01 & 1.00000000e-01 \\ 0.00000000e+00 & 8.33333333e-02 & 8.33333333e-02 & 7.50000000e-02 & 6.66666667e-02 & 5.95238095e-02 & 5.35714286e-02 & 4.86111111e-02 & 4.44444444e-02 & 4.09090909e-02 \\ 0.00000000e+00 & 0.00000000e+00 & 5.55555556e-03 & 8.33333333e-03 & 9.52380952e-03 & 9.92063492e-03 & 9.92063492e-03 & 9.72222222e-03 & 9.42760943e-03 & 9.09090909e-03 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 3.57142857e-04 & 7.14285714e-04 & 9.92063492e-04 & 1.19047619e-03 & 1.32575758e-03 & 1.41414141e-03 & 1.46853147e-03 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 2.26757370e-05 & 5.66893424e-05 & 9.27643785e-05 & 1.26262626e-04 & 1.55400155e-04 & 1.79820180e-04 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 1.43154905e-06 & 4.29464715e-06 & 8.09375810e-06 & 1.23333457e-05 & 1.66500167e-05 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 9.00974946e-08 & 3.15341229e-07 & 6.72727952e-07 & 1.13522842e-06 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 5.65997500e-09 & 2.26398940e-08 & 5.39362123e-08 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 3.55145365e-10 & 1.59814086e-09 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 2.22852500e-11 \end{bmatrix}$$

2 算法分析

1. LU 分解和 Cholesky 分解时间空间复杂度相同，但 LU 分解要存储两个矩阵，所需存储空间会更大。
2. n 变大时，LU 分解的误差会明显大于 Cholesky 分解。
3. n=19 时，H 非正定，此时 Cholesky 分解会出问题，需要加一个小常数保持矩阵正定。

3 实验小结

计算过程中可能出现的问题：

- 数值不稳定性：Hilbert 矩阵是经典的病态矩阵，随着矩阵维度的增加，其条件数迅速变大，可能导致在计算过程中出现数值不稳定性。这会影响到 LU 和 Cholesky 分解的精度，特别是在浮点数计算中，可能出现舍入误差。
- 非正定矩阵：Cholesky 分解要求矩阵是对称正定矩阵。如果输入的矩阵不满足正定条件，比如 Hilbert 矩阵在某些情况下会趋近于奇异矩阵，Cholesky 分解将无法进行，触发求解错误。而 LU 分解没有这种限制，适用于更广泛的矩阵类型。

分析比较两种算法：

- 适用范围：

  LU 分解是一种适用于任意方阵的分解方法，可以分解不一定是对称正定的矩阵。因此它的适用范围更广。即使是奇异矩阵，LU 分解仍然可以处理。Cholesky 分解仅适用于对称正定矩阵，其使用场景更为有限。如果矩阵不是正定矩阵，Cholesky 分解会失败。

- 计算效率：

  Cholesky 分解相比 LU 分解更高效，因为它仅需要对矩阵进行一次分解，计算复杂度是 $(^3/3)$。LU 分解需要更多的计算量，复杂度为 $(2^3/3)$，因为它需要分别计算上三角矩阵和下三角矩阵。因此，对于对称正定矩阵，Cholesky 分解更快，而对于一般的矩阵，必须使用 LU 分解。

- 数值稳定性：

  LU 分解在遇到病态矩阵（如 Hilbert 矩阵）时，可能会导致数值不稳定，尤其是在不进行部分选主元的情况下。为了提高稳定性，LU 分解通常结合选主元策略。Cholesky 分解的数值稳定性较高，前提是矩阵是正定的。当矩阵趋近于奇异时，Cholesky 分解容易失败。

总结：

- LU 分解是更通用的算法，适用于任意非奇异矩阵，但数值稳定性可能较差，特别是在处理病态矩阵时。
- Cholesky 分解是一种更高效且更稳定的分解方法，但仅限于对称正定矩阵。由于只需计算一个矩阵，计算量相较于 LU 分解减少一半。