

上机实验报告 2

姓名：付伟 学号：20190076

1. 主元的选取与算法的稳定性

1.1 问题描述

1.1.1 问题背景

Gauss 消去法是我们在线性代数中已经熟悉的。但由于计算机的数值运算是在一个有限的浮点数集合上进行的，如何才能确保 Gauss 消去法作为数值算法的稳定性呢？Gauss 消去法从理论算法到数值算法，其关键是主元的选择。主元的选择从数学理论上看起来平凡，它却是数值分析中十分典型的问题。

1.1.2 问题描述

考虑线性方程组

$$Ax = b, A \in R^{n \times n}, b \in R^n$$

编制一个能自动选取主元，又能手动选取主元的求解线性方程组的 Gauss 消去过程。要求如下：

1) 取矩阵 $A = \begin{bmatrix} 6 & 1 & & & \\ 8 & 6 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 8 & 6 & 1 \\ & & & 8 & 6 \end{bmatrix}$, $b = \begin{bmatrix} 7 \\ 15 \\ \vdots \\ 15 \\ 14 \end{bmatrix}$ 则方程有解 $x^* = (1, 1, \dots, 1)^T$ 。取

$n = 10$ 计算矩阵的条件数。让程序自动选取主元(顺序消元)，结果如何？

2) 现选择程序中手动选取主元的功能。每步消去过程总选取按模最小或按模尽可能小的元素作为主元，观察并记录计算结果。若每步消去过程总选取按模最大的元素作为主元，结果又如何？分析实验的结果。

3) 取矩阵阶数 $n = 20$ 或者更大，重复上述实验过程，观察记录并分析不同的问题及消去过程中选择不同的主元时计算结果的差异，说明主元素的选取在消去过程中的作用。

1.2 方案设计

根据要求，分别编写顺序高斯消元、按模最大列主元消元、按模最小列主元消元。高斯消元法主要包含两个步骤，简单描述如下：

1) 顺序消元 ($k = 1, 2, \dots, n - 1$)

$$\begin{cases} m_{ik} = a_{ik}^{(k)} / a_{kk}^{(k)}, i = k + 1, \dots, n \\ a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}, i, j = k + 1, \dots, n \\ b_i^{(k+1)} = b_i^{(k)} - m_{ik} b_k^{(k)}, i = k + 1, \dots, n \end{cases}$$

2) 回代

$$\begin{cases} x_n = b_n^{(n)} / a_{nn}^{(n)}, \\ x_i = \left(b_i^{(i)} - \sum_{j=i+1}^n a_{ij}^{(i)} x_j \right) / a_{ii}^{(i)} \end{cases}$$

当主元素很小时，会导致其他元素数量级的严重增长和舍入误差的扩散，主元的选取会对计算结果产生比较大的影响。为保持数值稳定性，在消元过程中，往往使用每一列按模最大的元素作为主元，即对矩阵进行初等行交换，使得主元的绝对值在每一列为最大。

1.3 问题求解

1.3.1 程序说明

gauss.m 文件可用于高斯消元法计算线性方程组的解，输入参数 mode 为 0 时为顺序消元，为 1 时采用列主元消元，为 2 时采用列最小不为零作为主元消元。

1.3.2 计算结果

- 1) 该三对角矩阵的条件数（二范数下）为：2557.5
得到顺序消元的解为

1	2	3	4	5	6	7	8	9	10
1.0000	1.0000	1.0000	1.0000	0.9999	1.0000	0.9999	1.0000	0.9999	1.0000
00000	00000	00000	00000	99999	00000	99999	00000	99999	00000
00000	00000	00000	00000	99999	00000	99999	00001	99997	00003
00	00	00	00	80	00	30	00	90	00

顺序消元得到的解和准确解的误差如下图所示

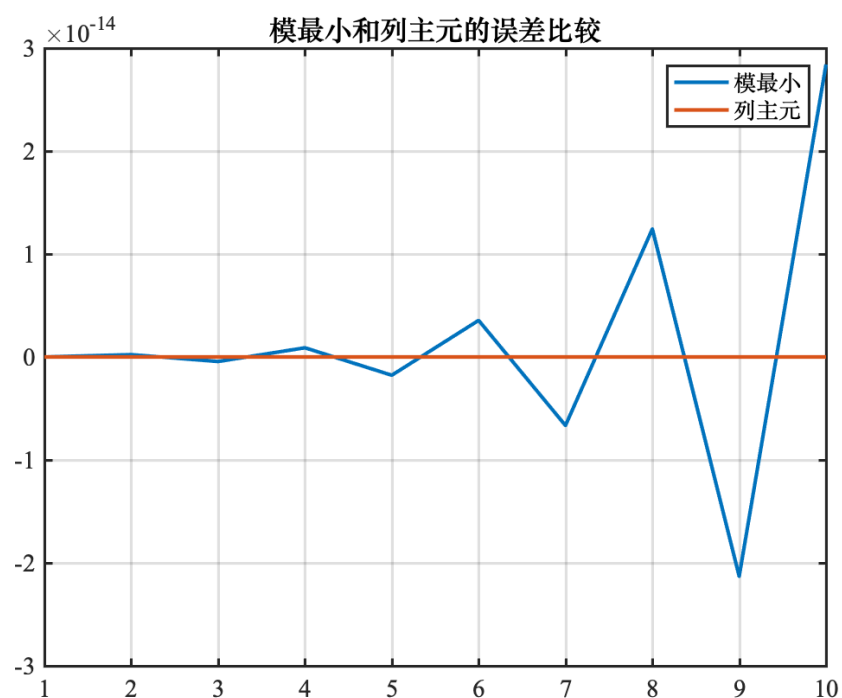
- 2) 按模最小得到的解如下表

1	2	3	4	5	6	7	8	9	10
1.0000	1.0000	1.0000	1.0000	0.9999	1.0000	0.9999	1.0000	0.9999	1.0000
00000	00000	00000	00000	99999	00000	99999	00000	99999	00000
00000	00000	00000	00000	99999	00000	99999	00001	99997	00003
00	00	00	00	80	00	30	00	90	00

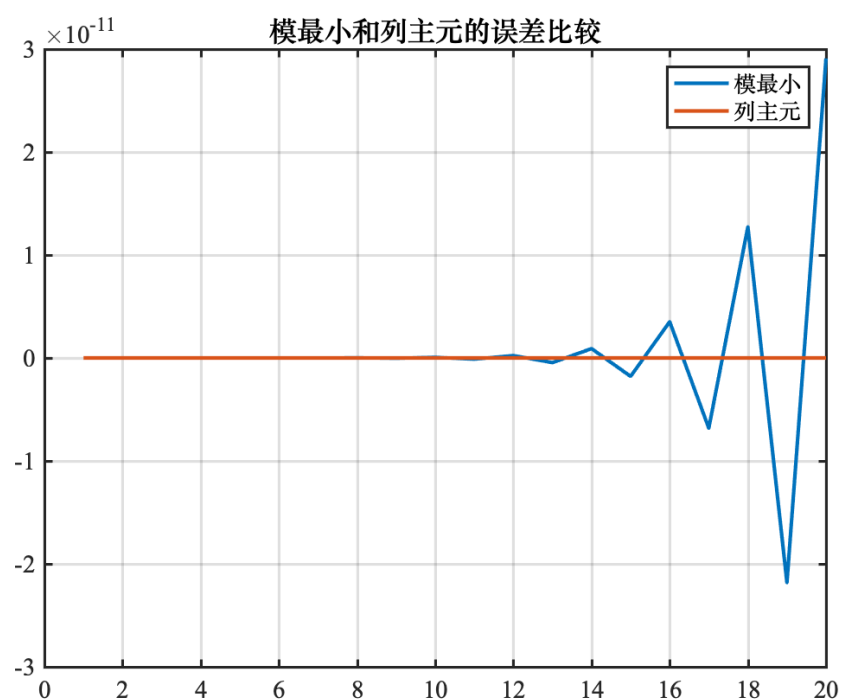
按模最大的解如下表

1	2	3	4	5	6	7	8	9	10
1	1	1	1	1	1	1	1	1	1

它们与标准解的误差如下图所示



3) 将 n 增大为 20 后，矩阵条件数变为 2621437.5，消元计算的结果如下图所示



可以看到，若选取模最小的元素作为主元，计算后几个解的误差会迅速变大，且随着 n 的增大，误差也在增大。这是由于若选取模最小的元素作为主元，除数过小会导致因子变大，出现“大数吃小数”的情况，积累舍入误差。

2. 线性代数方程组的性态与条件数的估计

2.1 问题描述

2.1.1 问题背景

理论上, 线性代数方程组 $Ax = b$ 的摄动满足

$$\frac{\|\Delta x\|}{\|x\|} \leq \frac{\text{cond}(A)}{1 - \|A^{-1}\| \|\Delta A\|} \left(\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta b\|}{\|b\|} \right)$$

矩阵的条件数确实是对矩阵病态性的刻画, 但在实际应用中直接计算它显然不现实, 因为计算 $\|A^{-1}\|$ 通常要比求解方程 $Ax = b$ 还困难。

2.1.2 问题描述

MATLAB 中提供有函数 “condest” 可以用来估计矩阵的条件数, 它给出的是按 1-范数的条件数。首先构造非奇异矩阵 A 和右端, 使得方程是可以精确求解的。再人为地引进系数矩阵和右端的摄动 ΔA 和 Δb , 使得 $\|\Delta A\|$ 和 $\|\Delta b\|$ 充分小。问题要求:

1) 假设方程 $Ax=b$ 的解为 x , 求解方程 $(A + \Delta A)\hat{x} = b + \Delta b$, 以 1-范数, 给出 $\frac{\|\Delta x\|}{\|x\|} = \frac{\|\hat{x}-x\|}{\|x\|}$ 的计算结果。

2) 选择一系列维数递增的矩阵 (可以是随机生成的), 比较函数 “condest” 所需机器时间的差别. 考虑若干逆是已知的矩阵, 借助函数 “eig” 很容易给出 $\text{cond2}(A)$ 的数值。将它与函数 “cond(A,2)” 所得到的结果进行比较。

3) 利用 “condest” 给出矩阵 A 条件数的估计, 针对 (1) 中的结果给出的 $\frac{\|\Delta x\|}{\|x\|}$ 理论估计, 并将它与 1) 给出的计算结果进行比较, 分析所得结果。

4) 估计著名的 Hilbert 矩阵的条件数。

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

2.2 方案设计

综合运用以下函数可以得到结果:

zeros(m,n)	生成 m 行, n 列的零矩阵
ones(m,n)	生成 m 行, n 列的元素全为 1 的矩阵
eye(n)	生成 n 阶单位矩阵
rand(m,n)	生成 m 行, n 列 (0,1) 上均匀分布的随机矩阵
diag(x)	返回由向量 x 的元素构成的对角矩阵
tril(A)	提取矩阵 A 的下三角部分生成下三角矩阵
triu(A)	提取矩阵 A 的上三角部分生成上三角矩阵
rank(A)	返回矩阵 A 的秩
det(A)	返回方阵 A 的行列式
inv(A)	返回可逆方阵 A 的逆矩阵
[V, D]=eig(A)	返回方阵 A 的特征值和特征向量
norm(A,p)	矩阵或向量的 p 范数
cond(A,p)	矩阵的条件数

Hi=hilb(n) 生成 n 阶 Hilbert 矩阵

对于 1)，选取实验 5.1 中的三对角阵，加上 rand 函数生成的微小扰动，采用命令直接求解方程；

对于 2)，矩阵的谱条件数为

$$\text{cond}(A)_2 = \frac{|\lambda_1|}{|\lambda_n|}$$

使用 eig 函数可以方便地计算谱条件数，并比较；

对于 3)，由于

$$\text{cond}(A) = \|A\| \|A^{-1}\|$$

如果知道了 $\|A\|$ 和条件数的估计，可以方便的给出误差上界；

对于 4)，使用 hilb 和 cond 函数可以简单求解条件数。

2.3 问题求解

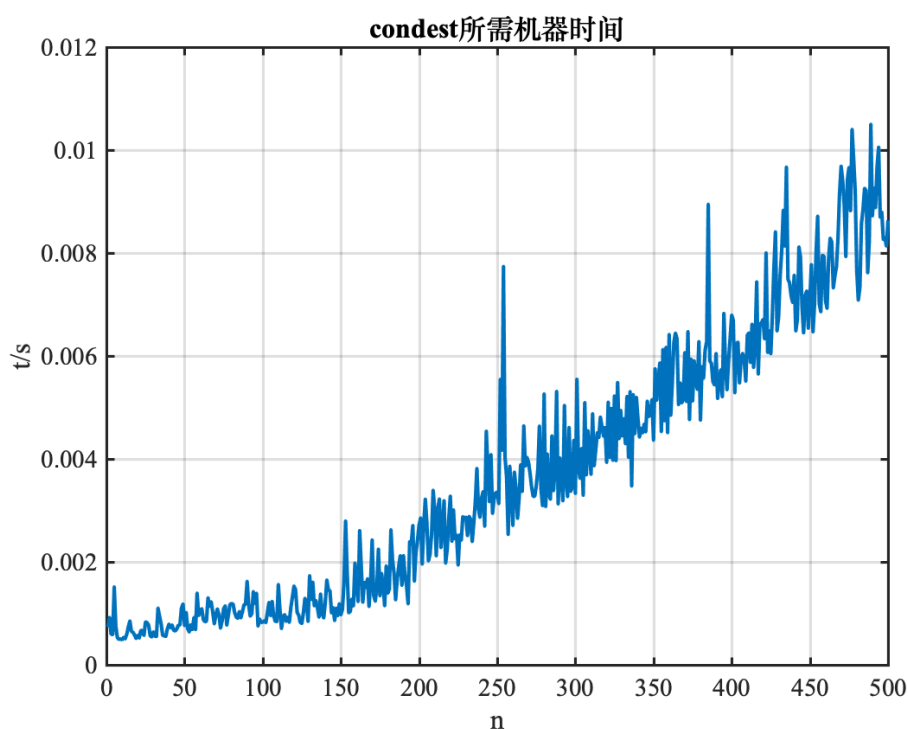
2.3.1 程序说明

main.m 的第二节即为计算实验 5.2 的代码。

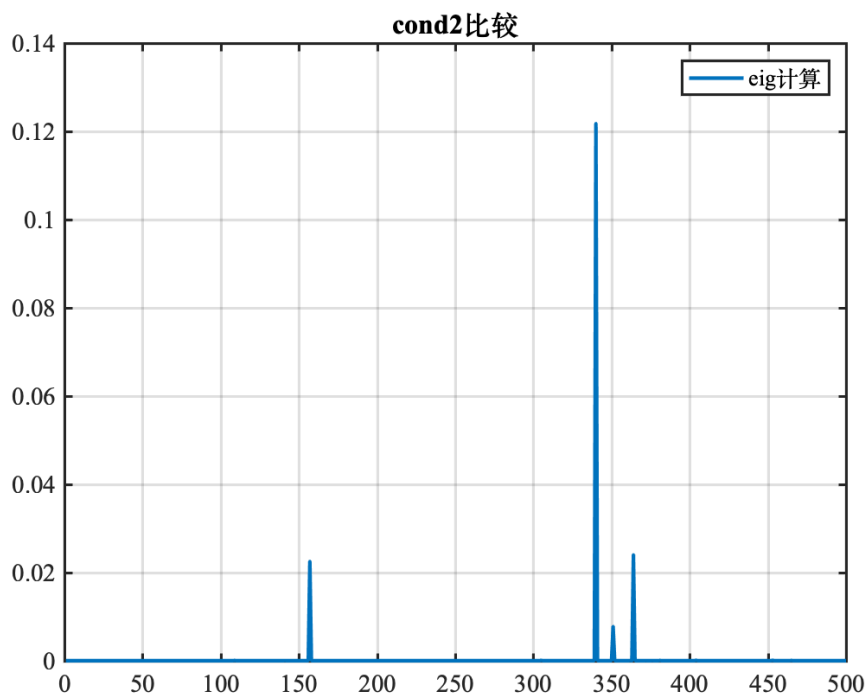
2.3.2 计算结果

1) 扰动量级为 $1e-6$ 时，计算得到的 $\frac{\|\Delta x\|}{\|x\|}$ 为 $7.05709741157690e-05$ 。

2) 不同维数的矩阵 condest 函数所需机器时间的变化情况如下图所示：



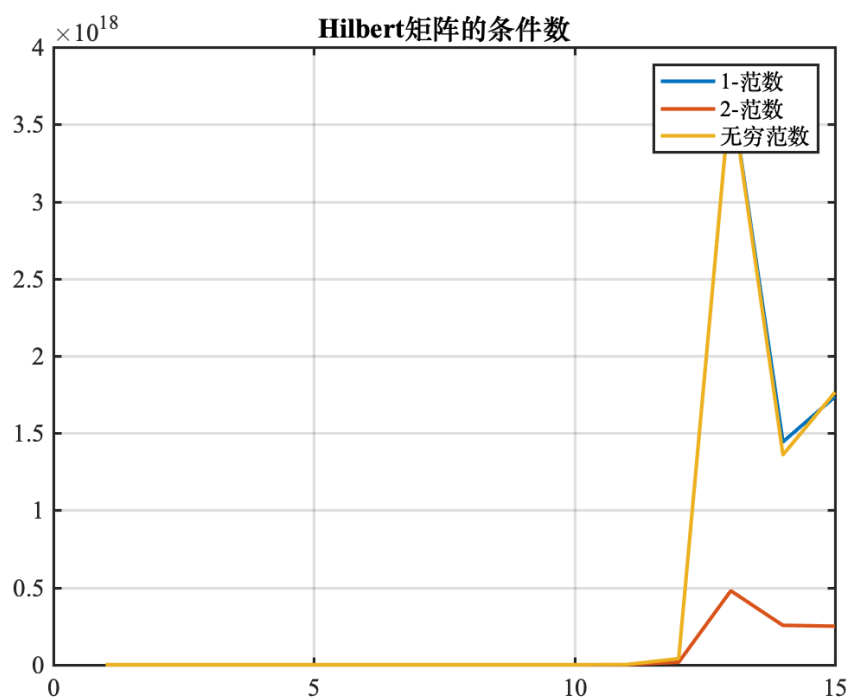
eig 和 cond 函数计算的谱条件数之差如下图所示：



可以看到除了少数几个差别较大的点，两种方法计算出的谱条件数可以认为一致。

3) 根据公式计算的到理论值为 0.00130535796854885 ，大于实际的误差 $7.05709741157690e-05$. 即证明了误差估计公式给出的是上限。

4) Hilbert 矩阵的条件数随维数的变化如下图所示：



得到的 1 范数条件数如下表：

1	2	3	4	5	6	7	8
1	27.0	748.0	28375.0	943655.9	29070279.0	985194886.4	33872788903.95

可以看到，矩阵增加 1 维，条件数增加几个量级，说明 Hilbert 矩阵病态非常严重。

3. 病态的线性方程组的求解

3.1 问题描述

3.1.1 问题背景

理论的分析表明，求解病态的线性方程组是困难的。实际情况是否如此，会出现怎样的现象呢？考虑方程组 $Hx=b$ 的求解，其中系数矩阵 H 为 Hilbert 矩阵，

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

这是一个著名的病态问题。通过首先给定解（例如取为各个分量均为 1）再计算出右端 b 的办法给出确定的问题。

3.1.2 问题描述

1) 选择问题的维数为 6，分别用 Gauss 消去法、列主元 Gauss 消去法、J 迭代法、GS 迭代法和 SOR 迭代法求解方程组，其各自的结果如何？将计算结果与问题的解比较，结论如何？

2) 逐步增大问题的维数（至少到 100），仍然用上述的方法来解它们，计算的结果如何？计算的结果说明了什么？

3) 讨论病态问题求解的算法。

3.2 方案设计

不同的迭代法的迭代公式分量形式如下：

1) jacobi 迭代

$$x_i^{(k+1)} = \left(b_i - \sum_{j=1, j \neq i}^n a_{ij} x_j^{(k)} \right) / a_{ii}, i = 1, 2, \dots, n$$

2) G-S 迭代

$$x_i^{(k+1)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij} x_j^{(k)} \right) / a_{ii}, i = 1, 2, \dots, n$$

3) SOR 迭代

$$x_i^{(k+1)} = x_i^{(k)} + \omega \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i}^n a_{ij} x_j^{(k)} \right) / a_{ii}, i = 1, 2, \dots, n$$

3.3 计算结果

3.3.1 程序文件

Jacobi.m 文件可用 Jacobi 迭代法计算线性方程组的解，输入参数为：矩阵 A ， b ，计算精度 ϵ ，最大迭代次数 $iter$ ；

GS.m 文件可用 G-S 迭代法计算线性方程组的解，输入参数为：矩阵 A ， b ，计算精度 ϵ ，最大迭代次数 $iter$ ；

SOR.m 文件可用 SOR 迭代法计算线性方程组的解，输入参数为：矩阵 A，b，计算精度 eps，最大迭代次数 iter，因子 omega。

3.3.2 计算结果

1) 矩阵维数为 6 时，解为（精确解为 $(1,1,1,1,1,1)^T$ ）

Gauss 消元法：

1	2	3	4	5	6
0.999999999 999228	1.000000000 002194	0.999999999 851792	1.000000000 038537	0.999999999 574584	1.000000000 016768

列主元 Gauss 消元法：

1	2	3	4	5	6
0.999999999 999072	1.000000000 002670	0.999999999 818267	1.000000000 047487	0.999999999 473962	1.000000000 020787

J 迭代法 ($\rho(J) = 4.3085 > 1$) 不收敛；

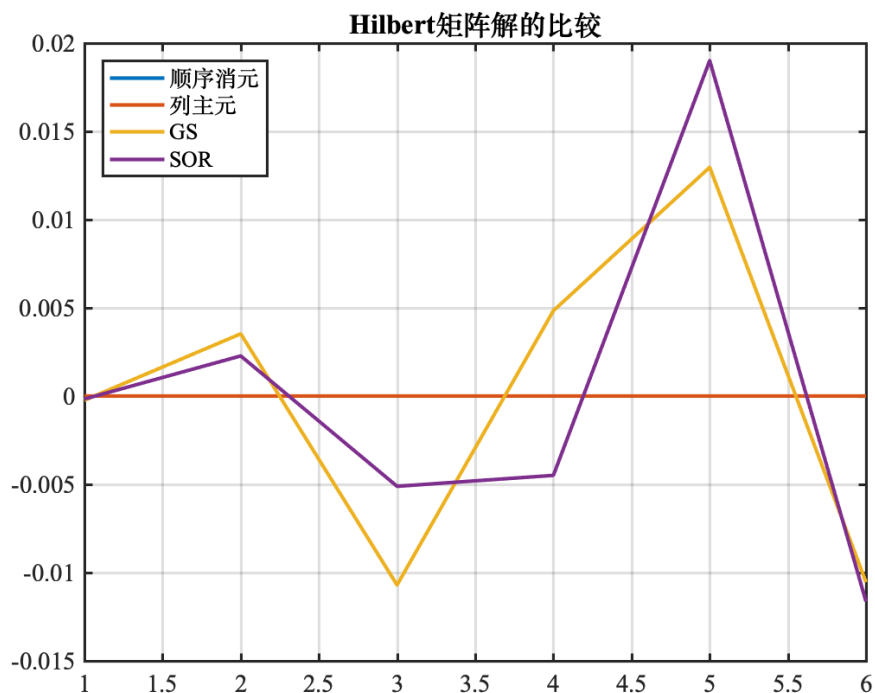
GS 迭代法（控制循环条件为 $1e-7$ ）：

1	2	3	4	5	6
0.999763387 886533	1.00352761 381646	0.989290774 029354	1.00483501 489063	1.01296417 844645	0.989445522 508985

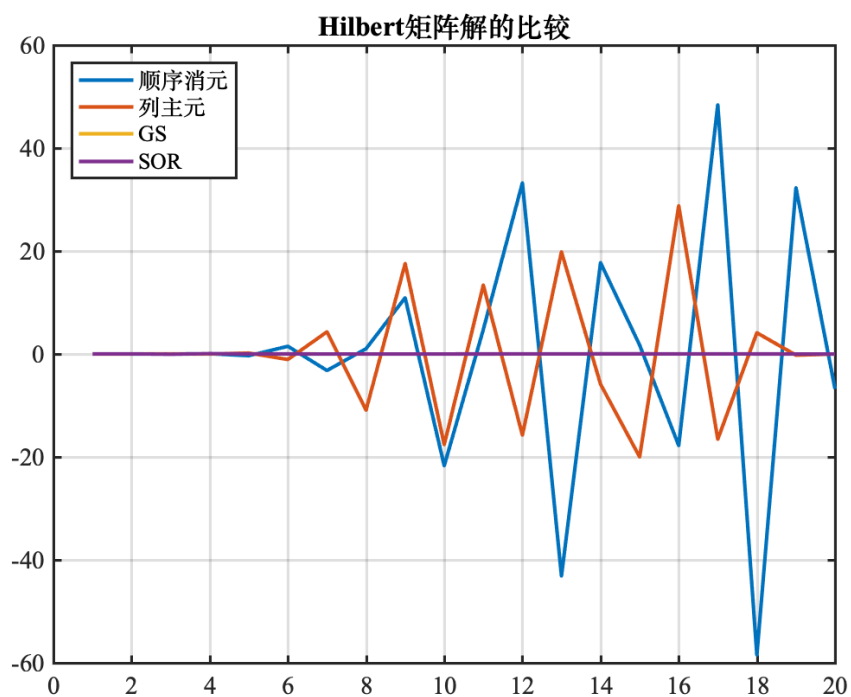
SOR 迭代法（控制循环条件为 $1e-7$ ，omega 取 1.5）：

1	2	3	4	5	6
0.999830928 771665	1.00227417 231049	0.994884825 312781	0.995503532 761870	1.01901249 905378	0.988351769 307922

和精确解的比较如下图所示：



2) 将维数增加到 20 维后，不同解法和精确解的误差如下图所示：



可以看到，迭代法解与精确解的偏差更小，而高斯消元法得到的偏差很大。结果说明 Hilbert 矩阵阶数较低时适合使用高斯消元法，阶数较高时适合使用迭代法求解。

3) 病态矩阵的解法

可以使用以下几种方法求解病态矩阵

a. 迭代改善法。直接法得到的解只是近似解，可以考虑采用迭代法对解进行改善。设 x_1 是 $Ax = b$ 的一个近似解， $r_1 = b - Ax_1$ 是 x_1 对应的余量，求解 $Ad_1 = r_1$ 得到 d_1 ，然后 $x_2 = x_1 + d_1$ 。重复这个过程，得到近似解的序列 $\{x_k\}$ 。然而，当矩阵 A 过分病态时， $\{x_k\}$ 有可能不收敛于精确解。

b. 加权迭代改善法

对于矩阵过分病态的情况，如 Hilbert 矩阵，可采用主元加权迭代改善算法。对于方程组 $Ax = b$ ，构造一个迭代过程：

$$(A + \alpha I)x^{(k+1)} = b + \alpha x^{(k)}$$

对于任意初始解 x_0 ，用 x_1 表示第一次计算出的解，则

$$(A + \alpha I)x^{(2)} = b + \alpha x^{(1)}$$

令 $x_2 = x_1 + e_1$ ，代入上式，得

$$(A + \alpha I)x^{(1)} + (A + \alpha I)e^{(1)} = b + \alpha x^{(1)}$$

即

$$(A + \alpha I)e^{(1)} = b - Ax^{(1)}$$

求解得到 e_1 。重复这个过程，得到近似解的序列 $\{x_k\}$ 。

c. 共轭梯度法是解决大型、稀疏、正定对称的线性方程组的有效方法。在矩阵的条件数很大时仍然能得到较好的结果。如果先采用预处理降低矩阵条件数，再用共轭梯度法求解，效果往往更好。

d. 遗传算法将矩阵求解问题转化为优化问题，然后用遗传算法求解。

4. 二维泊松方程的求解

4.1 问题描述

书上 P211 计算实习题 2，其中 $n=100$ ，第二小问改为用 Jacobi 迭代、G-S 迭代、红黑排序的 G-S 迭代求解，并比较他们之间的收敛速度；进一步，用 BSOR 迭代求解，试找出最优松弛因子。

4.2 方案设计

1) 建立方程

该问题为二维 Poisson 方程求解问题，可以使用五点公式建立线性方程组求解。根据差分方程可以得到：

$$4u_{ij} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} = h^2 f_{ij}$$

该问题和例 10 区别在于边值不为零，如果 $(x_i, y_j) \in \partial\Omega$ ，可以通过在方程 $Ax = b$ 的右端项添加边界点的取值，即可转化为边值为零的情况（A 矩阵不变）。

2) 迭代求解

4.3 问题求解

4.3.1 程序说明

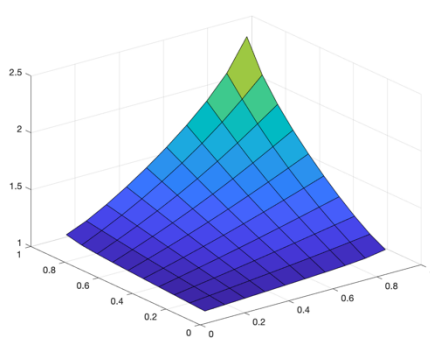
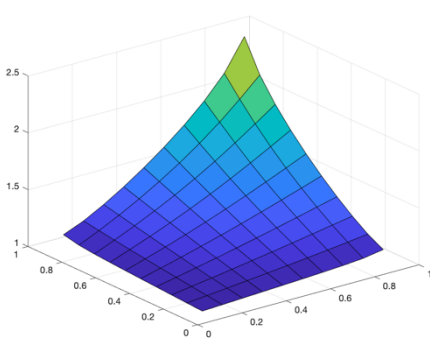
main.m 文件中 6.1 节即为求解二维泊松方程的代码。首先根据网格点之间的关系定义矩阵，对于边界点修改右端项，之后便可化为线性方程组求解。

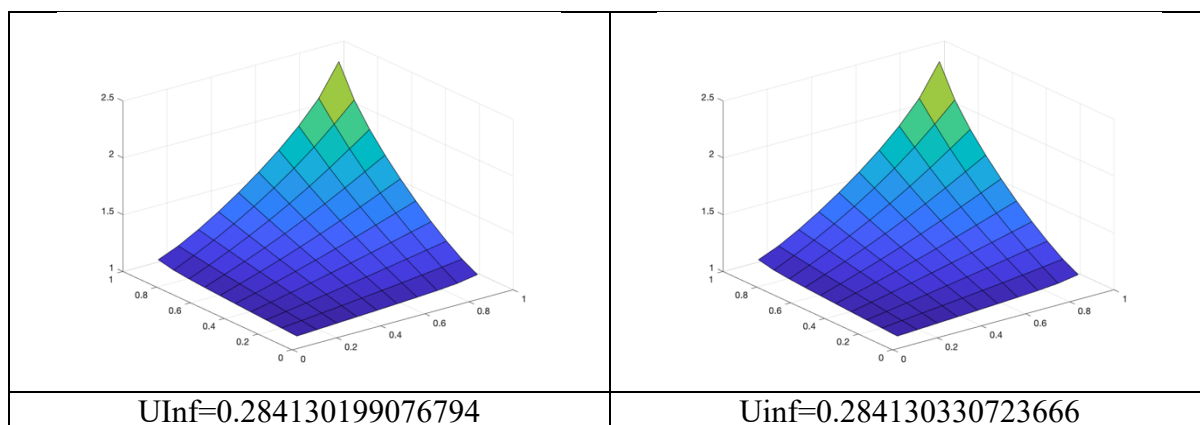
GS.m 文件中输入 mode 参数为 1 时，即为红黑排序法求解。

4.3.2 计算结果

1) 不同方法计算结果对比（ $n=100$ ），其中 U_{inf} 定义为：

$$U_{inf} = \|\hat{u} - u\|_{inf}$$

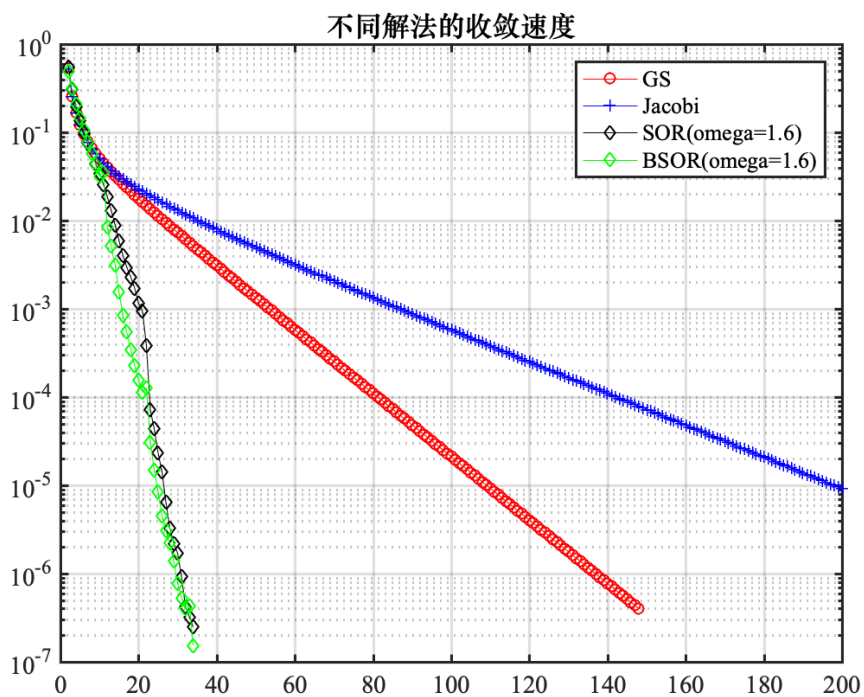
Jacobi	GS
	
$U_{inf}=0.283080816796088$	$U_{inf}=0.284111899362174$
SOR ($\omega=1.6$)	BSOR ($\omega=1.5$)



不同方法收敛速度对比，纵坐标为逐次迭代间偏差下降速度，定义为

$$\frac{\|x^{(k+1)} - x^{(k)}\|}{\|x^{(k)}\|}$$

画出图像如下图所示



可以看到，四种方法得到的 Uinf 几乎一致，但是 BSOR 方法收敛得更快。

2) 寻找最优松弛因子

可以画出矩阵 A 进行 SOR 迭代时迭代矩阵 L_ω 的谱半径随 ω 的变化关系如下图所示，得到最优的迭代因子大约为 1.57。

