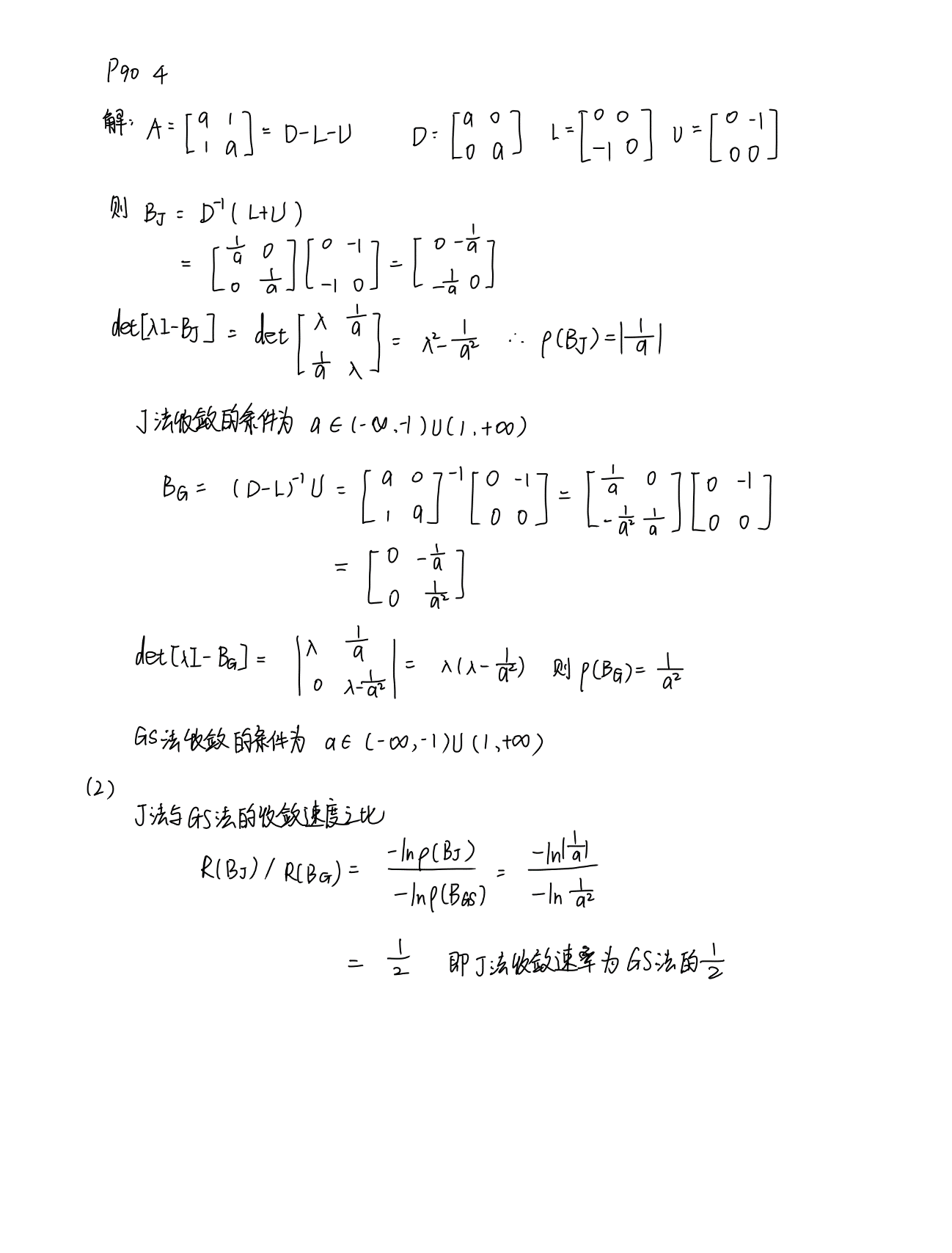
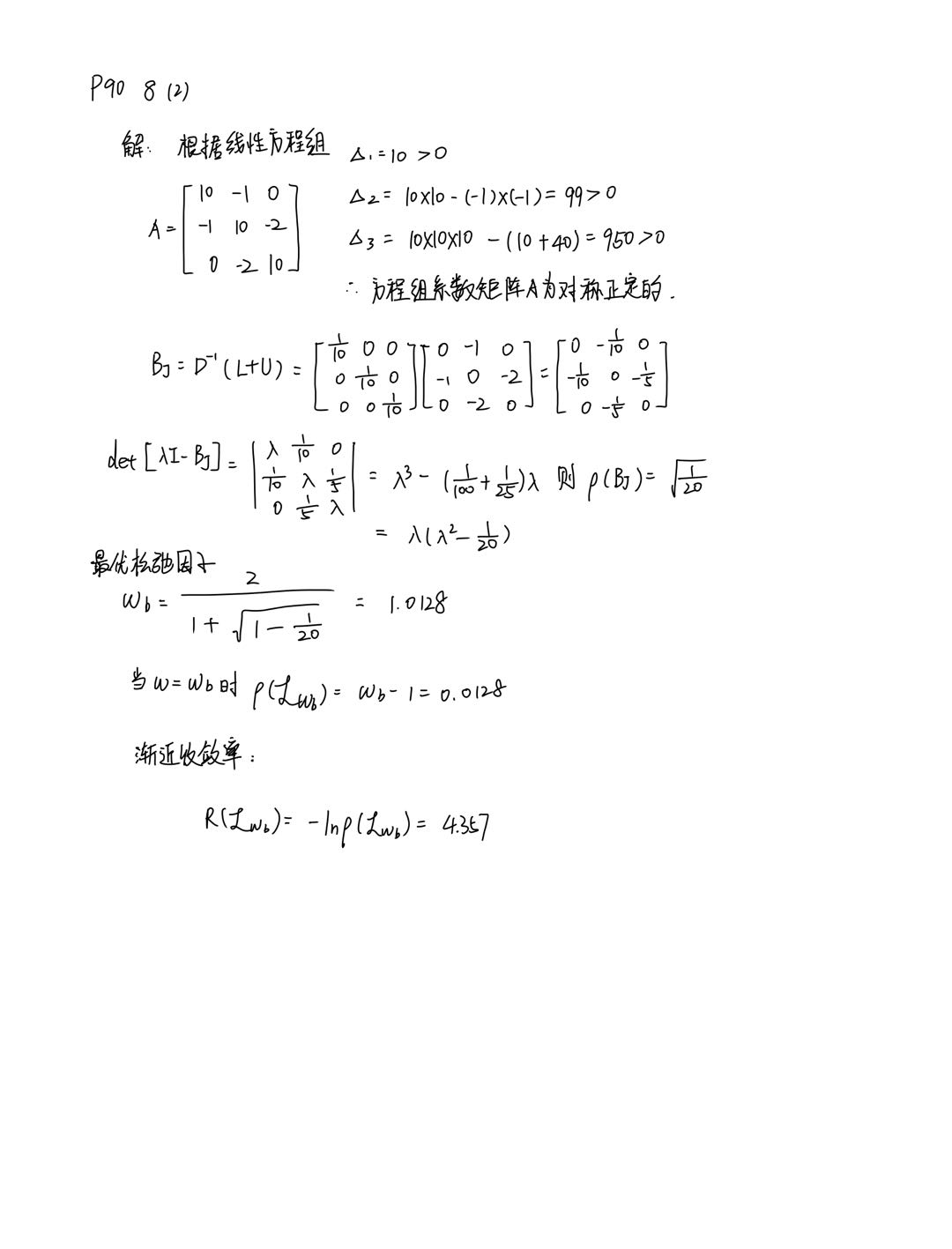
第四次作业

赖显松 2021214726

1 书面作业题目





2 编程题目

2.1 五对角矩阵

编写生成五对角矩阵的函数（输入：对角元素列表nums，阶次*n*；返回：*n*阶的多对角矩阵矩阵）。代码如下：

1. **def** geneDlgMat(nums: list, n: int):
2. ap = len(nums)
3. **if** ap % 2 == 1:
4. A = np.zeros((n, n + ap - 1))
5. **for** i **in** range(n):
6. **for** j **in** range(ap):
7. A[i,i + j] = nums[j]
8. A = A[:,int((ap-1)/2):n+int((ap-1)/2)]
9. **return** A
10. **else**:
11. **print**('请输入一个奇数')
12. **return** None

生成一个10阶五对角矩阵，对角元素为（1，-8，20，-8，1），如图2. 1所示：

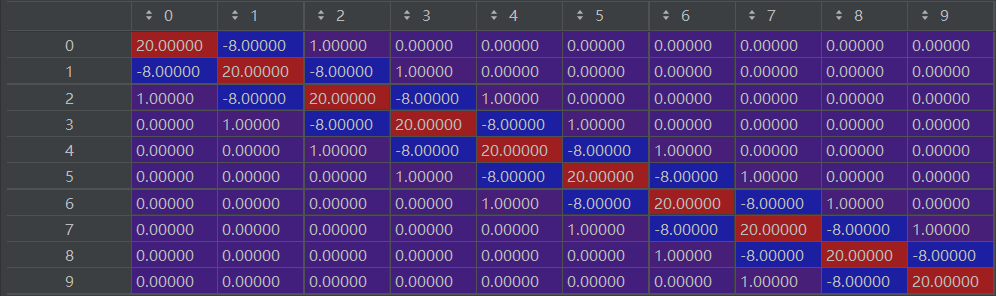


图2. 1 10阶五对角矩阵

2.2 迭代法求解线性方程组

迭代法解线性方程组的思想为[1]：通过构造迭代函数，使得通过这种方式产生的序列若能满足(2 - 1)：



(2 - 1)

则晨构造函数迭代法为收敛的，且经过一定次数的迭代，随机的序列可以近似等于至线性方程组解的真实值。

数值方法这门课程主要涉及的迭代方法有Jacobi迭代法、Gauss-Seidel迭代法和超松弛迭代法。

2.3 Jacobi迭代法及python程序实现

迭代法需要将系数矩阵先拆分为3个不同的矩阵部分作为构造迭代函数用到的元素：对角元矩阵，零对角上三角矩阵和零对角下三角矩阵，并且满足。

获取矩阵迭代函数元素的python代码如下：

1. **def** Mat2LUD(A):
2. n = A.shape[0]
3. D = np.zeros((n, n))
4. L = np.zeros((n, n))
5. U = np.zeros((n, n))
6. **for** i **in** range(n):
7. D[i, i] = A[i, i]
8. idx1 = np.arange(0,i)
9. idx2 = np.arange(i+1,n)
10. L[i, idx1] = -A[i, idx1]
11. U[i, idx2] = -A[i, idx2]
12. **return** L, U, D

Jacobi迭代法的迭代函数为(2 - 2)：



(2 - 2)

Jacobi迭代法迭代函数获取的Python代码如下：

1. **def** BJandfJ(Mat\_A, b):
2. n = Mat\_A.shape[0]
3. E = np.eye(n)
4. Mat\_L, Mat\_U, Mat\_D = Mat2LUD(Mat\_A)
5. BJ = E-(np.linalg.inv(Mat\_D))@(Mat\_A)
6. fJ = np.linalg.inv(Mat\_D)@b
7. **return** BJ, fJ

题目通过Jacobi迭代法求矩线性方程组的流程为：

1 生成系数矩阵、和

2 取

3 设置迭代误差范数计算方法、迭代停止精度、初始化迭代误差

4 循环：在误差未满足精度前，通过迭代函数获取新的序列，比较新旧序列差值范数是否小于精度：若是，退出循环；若否将新序列作为旧序列继续迭代。

具体python代码实现如下（精度设置为10-6），其中设置一个循环计算时五对角矩阵线性方程组的结果：

1. **for** n **in** [10, 20, 40]:
2. A = SM.geneDlgMat([1, -8, 20, -8, 1], n)
3. b = np.zeros(n)
4. xOld = np.ones(n)
5. xTrue = li.solve(A, b)
6. s = -6  # 选择精度
7. method = 'inf'
8. itNumJ = 0
9. eJ = IS.iterNorm(np.zeros(n), xOld, method)  # 初始值
11. B, f = IS.BJandfJ(A, b)  # J法收敛次数
12. **while** eJ > 10 \*\* s:
13. xNewJ = B @ xOld + f
14. eJ = IS.iterNorm(xOld, xNewJ, method)
15. xOld = xNewJ
16. itNumJ += 1
17. **print**('J法>>', n, '阶5对角的迭代次数：', itNumJ)

结果如图2. 2所示：

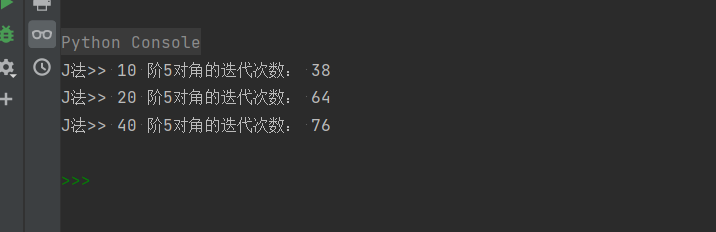


图2. 2 10，20，40阶五对角矩阵线性方程组的迭代次数

10、20、40阶五对角矩阵线性方程组的收敛次数约为38、64和76。可见，Jacobi迭代法的是收敛的，且随着五对角矩阵阶次的增加，收敛的次数增多。

2.4 SOR迭代法及python程序实现

SOR迭代法的迭代函数为(2 - 3)：



(2 - 3)

其中为松弛因子，是影响SOR迭代法收敛速度的一个重要参数。

SOR迭代法迭代函数获取的Python代码如下：

1. **def** SOR(A, b, omiga):  # 迭代因子omiga
2. n = A.shape[0]
3. E = np.eye(n)
4. A\_L, A\_U, A\_D = Mat2LUD(A)
5. Lw\_B = (np.linalg.inv(A\_D-omiga\*A\_L)) @ (omiga\*A\_U+(1-omiga)\*A\_D)
6. Lw\_f = omiga\*np.linalg.inv(A\_D-omiga\*A\_L) @ b
7. **return** Lw\_B, Lw\_f

题目通过SOR迭代法求矩线性方程组的流程为：

1 生成系数矩阵、和

2 取

3 设置迭代误差范数计算方法、迭代停止精度、初始化迭代误差

4 循环：在误差未满足精度前，通过迭代函数获取新的序列，比较新旧序列差值范数是否小于精度：若是，退出循环；若否将新序列作为旧序列继续迭代。

具体python代码实现如下（精度设置为10-6），其中设置一个循环计算,从中间隔0.05取值，并且时五对角矩阵线性方程组的结果，计算迭代的渐进收敛速度，最后绘制收敛速度和松弛因子取值的关系曲线：

1. w = np.arange(0.5, 2, 0.05)
2. k = np.zeros(len(w))
3. RB\_Lw\_inf = np.zeros(len(w))
4. **for** i **in** range(len(w)):
5. xOld = np.ones(n)
6. itNumJ = 0
7. eJ = IS.iterNorm(np.zeros(n), xOld, method)  # 初始值
8. B, f = IS.SOR(A, b, w[i])
9. RB\_Lw\_inf[i] = -np.log(np.linalg.norm(B, np.inf))
10. **while** eJ > 10 \*\* s:
11. xNewJ = B @ xOld + f
12. eJ = IS.iterNorm(xOld, xNewJ, method)
13. xOld = xNewJ
14. itNumJ += 1
15. # eJFinal = IS.iterNorm(xTrue, xNewJ, method)  # 最终误差
16. **print**('SOR法>>w==', w[i], n, '阶5对角的迭代次数：', itNumJ)
17. k[i] = itNumJ
19. plt.rcParams['font.sans-serif'] = ['SimHei']  # 用来正常显示中文标签
20. plt.rcParams['axes.unicode\_minus'] = False  # 用来正常显示负号
21. plt.subplot(1, 2, 1)
22. plt.plot(w, k)
23. plt.xlabel('松弛因子')
24. plt.ylabel('迭代次数k')
25. plt.subplot(1, 2, 2)
26. plt.plot(w, RB\_Lw\_inf)
27. plt.xlabel('松弛因子')
28. plt.ylabel('渐进收敛速度R(B)')
29. plt.show()

结果如图2. 3所示：

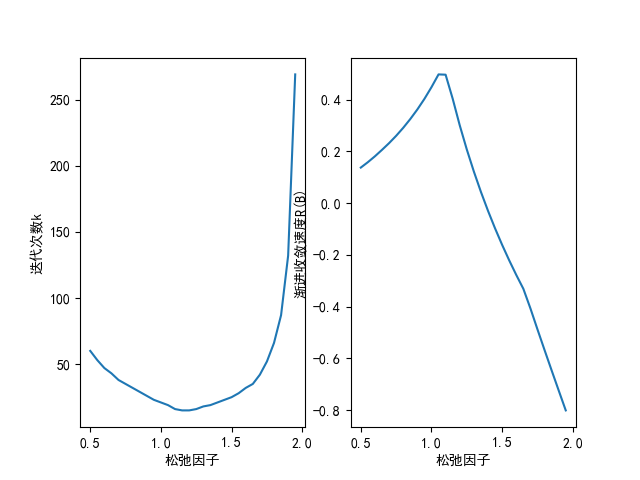


图2. 3 10阶五对角矩阵线性方程组的迭代次数随松弛因子变化

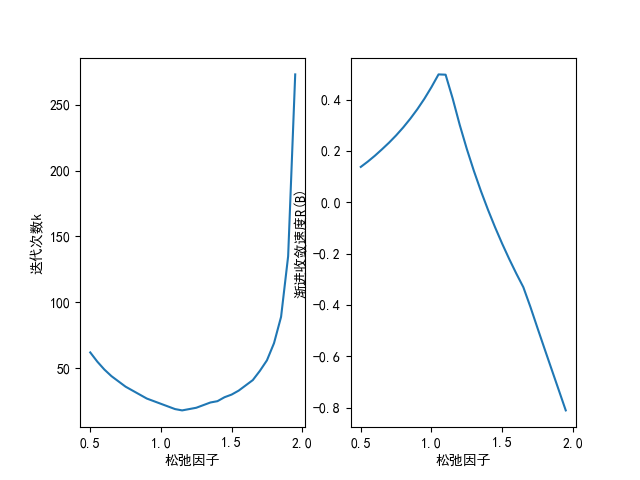


图2. 4 20阶五对角矩阵线性方程组的迭代次数随松弛因子变化

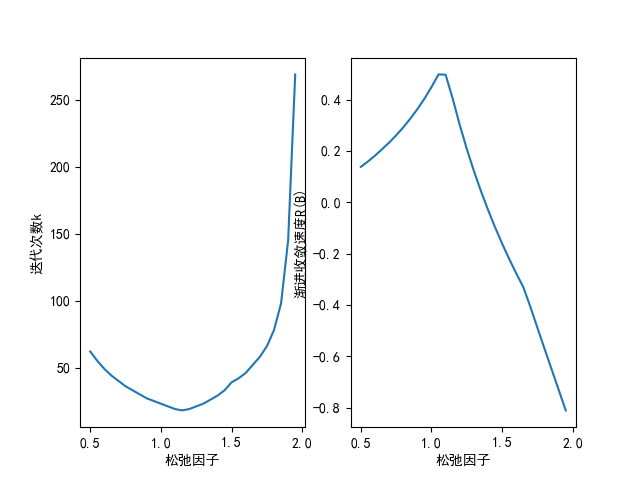


图2. 5 40阶五对角矩阵线性方程组的迭代次数随松弛因子变化

可见，SOR迭代法的是收敛的，且随着五对角矩阵阶次的增加，收敛的次数并没有什么的变化。10、20、40阶五对角矩阵线性方程组的最低收敛次数约为15、18和18。而且SOR迭代法的收敛速度（渐进收敛率）随松弛因子在内变化呈现先下降后上升的趋势，在松弛因子约为1.2左右时速度最快。

参考文献

1. 关冶．数值方法[M]．北京：清华大学出版社，2006：66-92．

附录

见附件python源代码。