assn08

June 4, 2025

1 1

设线性方程组

$$\begin{cases} 5x_1 + 2x_2 + x_3 = -12 \\ -x_1 + 4x_2 + 2x_3 = 20 \\ 2x_1 - 3x_2 + 10x_3 = 3 \end{cases}$$

(1) 考察用雅可比迭代法、高斯-塞德尔迭代法解此方程组的收敛性。

(2) 用雅可比迭代法及高斯-塞德尔迭代法解此方程组，要求当 $||\mathbf{x}^{k+1} - \mathbf{x}^k||_\infty < 10^{-4}$ 时迭代终止。

1.1 Solution

(1) 收敛性

检查矩阵 A 是否为严格对角占优矩阵。* 对于第一行：$|5| > |2| + |1| \implies 5 > 3$ (成立) * 对于第二行：$|4| > |-1| + |2| \implies 4 > 3$ (成立) * 对于第三行：$|10| > |2| + |-3| \implies 10 > 5$ (成立)

由于矩阵 A 是严格对角占优的，因此雅可比迭代法和高斯-塞德尔迭代法都收敛。

(2) 迭代求解

使用 Python 计算雅可比迭代和高斯-塞德尔迭代：

[ ]:
```
import numpy as np

def solve_linear_system():
    A = np.array([[5, 2, 1],
                  [-1, 4, 2],
                  [2, -3, 10]], dtype=float)
    b = np.array([-12, 20, 3], dtype=float)

    n = len(b)
    tol = 1e-4
    max_iter = 1000

    # --- 雅可比迭代 ---
    print("雅可比迭代:")
```

```python
    x_jacobi = np.zeros(n)
    for k_jacobi in range(max_iter):
        x_new_jacobi = np.zeros(n)
        for i in range(n):
            s = sum(A[i, j] * x_jacobi[j] for j in range(n) if j != i)
            x_new_jacobi[i] = (b[i] - s) / A[i, i]

        if np.linalg.norm(x_new_jacobi - x_jacobi, np.inf) < tol:
            x_jacobi = x_new_jacobi
            break
        x_jacobi = x_new_jacobi

    print(f"解: {x_jacobi}")
    print(f"迭代次数: {k_jacobi + 1}")
    print("-" * 30)

    # --- 高斯-塞德尔迭代 ---
    print("高斯-塞德尔迭代:")
    x_gs = np.zeros(n)
    for k_gs in range(max_iter):
        x_old_gs = x_gs.copy()
        for i in range(n):
            s1 = sum(A[i, j] * x_gs[j] for j in range(i))
            s2 = sum(A[i, j] * x_old_gs[j] for j in range(i + 1, n))
            x_gs[i] = (b[i] - s1 - s2) / A[i, i]

        if np.linalg.norm(x_gs - x_old_gs, np.inf) < tol:
            break

    print(f"解: {x_gs}")
    print(f"迭代次数: {k_gs + 1}")
    print("-" * 30)

solve_linear_system()
```

```
雅可比迭代:
解: [-3.99999642  2.99997389  1.99999989]
迭代次数: 18
------------------------------
高斯-塞德尔迭代:
解: [-4.00003333  2.99998307  2.00000159]
迭代次数: 8
------------------------------
```

2 4

设

$$A = \begin{pmatrix} 10 & a & 0 \\ b & 10 & b \\ 0 & a & 5 \end{pmatrix}, \quad \det(A) \neq 0$$

用 a, b 表示解线性方程组 $A\mathbf{x} = f$ 的雅可比迭代与高斯-塞德尔迭代收敛的充分必要条件。

2.1 Solution

1. 雅可比迭代法

雅可比迭代矩阵 $B_J = D^{-1}(L+U)$:

$$B_J = \begin{pmatrix} 0 & -a/10 & 0 \\ -b/10 & 0 & -b/10 \\ 0 & -a/5 & 0 \end{pmatrix}$$

计算特征多项式:

$$\det(B_J - \lambda I) = -\lambda^3 + \frac{3ab}{100}\lambda = -\lambda(\lambda^2 - \frac{3ab}{100})$$

特征值: $\lambda_1 = 0$, $\lambda_{2,3} = \pm\sqrt{\frac{3ab}{100}}$

谱半径: $\rho(B_J) = \sqrt{\frac{|3ab|}{100}}$

收敛条件: $\rho(B_J) < 1 \Rightarrow |ab| < \frac{100}{3}$

2. 高斯-塞德尔迭代法

对于具有一致排序性质的三对角矩阵，有性质:

$$\rho(B_{GS}) = [\rho(B_J)]^2$$

因此: $\rho(B_{GS}) = \frac{|3ab|}{100}$

收敛条件: $\rho(B_{GS}) < 1 \Rightarrow |ab| < \frac{100}{3}$

结论

雅可比迭代与高斯-塞德尔迭代收敛的充分必要条件都是:

$$|ab| < \frac{100}{3}$$

即:

$$-\frac{100}{3} < ab < \frac{100}{3}$$

3 5

对线性方程组

$$\begin{pmatrix} 3 & 2 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 3 \\ -1 \end{pmatrix}$$

若用迭代法

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha(A\mathbf{x}^{(k)} - \mathbf{b}), \quad k = 0, 1, \cdots$$

求解，问 α 在什么范围内取值可使迭代收敛; α 取什么值可使迭代收敛最快。

3.1 Solution

整理得：$\mathbf{x}^{(k+1)} = (I + \alpha A)\mathbf{x}^{(k)} - \alpha\mathbf{b}$

迭代矩阵为：$G = I + \alpha A$

其中 $A = \begin{pmatrix} 3 & 2 \\ 1 & 2 \end{pmatrix}$

特征多项式：$\det(A - \lambda I) = (3 - \lambda)(2 - \lambda) - 2 = \lambda^2 - 5\lambda + 4 = (\lambda - 1)(\lambda - 4)$

特征值：$\lambda_1 = 1, \lambda_2 = 4$

$G = I + \alpha A$ 的特征值为：$\mu_1 = 1 + \alpha, \mu_2 = 1 + 4\alpha$

迭代收敛的充分必要条件是 $\rho(G) < 1$，即：

$$|1 + \alpha| < 1 \quad \text{且} \quad |1 + 4\alpha| < 1$$

解不等式：- $|1 + \alpha| < 1 \Rightarrow -2 < \alpha < 0$ - $|1 + 4\alpha| < 1 \Rightarrow -\frac{1}{2} < \alpha < 0$

取交集得收敛范围：

$$-\frac{1}{2} < \alpha < 0$$

收敛速度由谱半径决定，$\rho(G) = \max\{|1 + \alpha|, |1 + 4\alpha|\}$

在区间 $(-\frac{1}{2}, 0)$ 内：- 当 $\alpha \in (-\frac{1}{2}, -\frac{1}{3})$ 时，$|1 + 4\alpha| > |1 + \alpha|$ - 当 $\alpha \in (-\frac{1}{3}, 0)$ 时，$|1 + \alpha| > |1 + 4\alpha|$

最优值在 $|1 + \alpha| = |1 + 4\alpha|$ 处取得：

$$1 + \alpha = -(1 + 4\alpha) \Rightarrow \alpha = -\frac{2}{5}$$

此时 $\rho(G) = |1 - \frac{2}{5}| = \frac{3}{5} = 0.6$

4 6

用雅可比迭代与高斯-塞德尔迭代解线性方程组 $A\mathbf{x} = \mathbf{b}$，证明若取

$$A = \begin{pmatrix} 3 & 0 & -2 \\ 0 & 2 & 1 \\ -2 & 1 & 2 \end{pmatrix}$$

则两种方法均收敛。试比较哪种方法收敛更快。

1. 收敛性证明

矩阵 $A = \begin{pmatrix} 3 & 0 & -2 \\ 0 & 2 & 1 \\ -2 & 1 & 2 \end{pmatrix}$ 是对称矩阵。

计算特征值：- 特征多项式：$\det(A - \lambda I) = (3-\lambda)(2-\lambda)^2 - 4(2-\lambda) = -(\lambda-1)(\lambda-2)(\lambda-4)$ - 特征值：$\lambda_1 = 1, \lambda_2 = 2, \lambda_3 = 4$ (均为正数)

因为 A 是对称正定矩阵，所以：- 雅可比迭代法收敛 - 高斯-塞德尔迭代法收敛

2. 收敛速度比较

对于对称正定矩阵，有结论：

$$\rho(B_{GS}) = [\rho(B_J)]^2$$

这意味着高斯-塞德尔迭代的谱半径是雅可比迭代谱半径的平方，因此：- $\rho(B_{GS}) < \rho(B_J)$（当 $\rho(B_J) < 1$ 时）- **高斯-塞德尔迭代收敛更快**

5 8

用 SOR 方法解线性方程组（取松弛因子 $\omega = 1.03, \omega = 1, \omega = 1.1$）

$$\begin{cases} 4x_1 - x_2 = 1 \\ -x_1 + 4x_2 - x_3 = 4 \\ -x_2 + 4x_3 = -3 \end{cases}$$

精确解 $x^* = (\frac{1}{2}, 1, -\frac{1}{2})^T$，要求当 $||x^* - x^{(k)}||_\infty < 5 \times 10^{-6}$ 时迭代终止，并且对于每一个 ω 值确定迭代次数。

5.1 Solution

SOR 方法的迭代公式为：

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

其中：- $\omega = 1$：高斯-塞德尔迭代 - $0 < \omega < 1$：低松弛 - $1 < \omega < 2$：超松弛

使用 python 求解，

[6]:

```python
import numpy as np

def sor_method():
    # 系数矩阵和右端向量
    A = np.array([[4, -1, 0],
                  [-1, 4, -1],
                  [0, -1, 4]], dtype=float)
    b = np.array([1, 4, -3], dtype=float)

    # 精确解
    x_exact = np.array([0.5, 1.0, -0.5])

    # 松弛因子
    omega_values = [1.03, 1.0, 1.1]

    # 收敛容差
    tol = 5e-6
    max_iter = 1000

    results = []

    for omega in omega_values:
        print(f"\nSOR 方法 (  = {omega}):")
        print("-" * 40)

        # 初始值
        x = np.zeros(3)
        n = len(b)

        # SOR 迭代
        for k in range(max_iter):
            x_old = x.copy()

            # SOR 迭代公式
            for i in range(n):
                # 计算前向替换部分 (已更新的分量)
                s1 = sum(A[i, j] * x[j] for j in range(i))
                # 计算后向替换部分 (未更新的分量)
                s2 = sum(A[i, j] * x_old[j] for j in range(i + 1, n))

                # SOR 更新公式
                x_new_i = (b[i] - s1 - s2) / A[i, i]
                x[i] = (1 - omega) * x_old[i] + omega * x_new_i

            # 计算误差
            error = np.linalg.norm(x_exact - x, np.inf)
```

```
            # 输出迭代过程（前几步和最后几步）
            if k < 5 or k % 20 == 0 or error < tol:
                print(f"k={k:3d}: x = [{x[0]:8.6f}, {x[1]:8.6f}, {x[2]:8.6f}], "
                      f"||x*-x||∞ = {error:.2e}")

            # 检查收敛
            if error < tol:
                iterations = k + 1
                final_error = error
                final_solution = x.copy()
                break
        else:
            iterations = max_iter
            final_error = error
            final_solution = x.copy()
            print("达到最大迭代次数，未收敛")

        print(f"\n最终结果:")
        print(f"迭代次数: {iterations}")
        print(f"最终解: x = [{final_solution[0]:.8f}, {final_solution[1]:.8f}, {final_solution[2]:.8f}]")
        print(f"最终误差: ||x* - x||∞ = {final_error:.2e}")

        results.append({
            'omega': omega,
            'iterations': iterations,
            'final_error': final_error,
            'solution': final_solution
        })

    # 比较结果
    print("\n" + "="*60)
    print("结果比较:")
    print("-" * 60)
    print(f"{'  值':<8} {'迭代次数':<10} {'最终误差':<15} {'收敛状态'}")
    print("-" * 60)

    for result in results:
        status = "收敛" if result['final_error'] < tol else "未收敛"
        print(f"{result['omega']:<8} {result['iterations']:<10} {result['final_error']:.2e}       {status}")

    # 找出最优松弛因子
    converged_results = [r for r in results if r['final_error'] < tol]
    if converged_results:
        best_result = min(converged_results, key=lambda x: x['iterations'])
        print(f"\n最优松弛因子:   = {best_result['omega']}")
```

```python
        print(f"最少迭代次数: {best_result['iterations']}")

sor_method()
```

```
SOR 方法 ( = 1.03):
----------------------------------------
k=   0: x = [0.257500, 1.096306, -0.490201], ||x*-x||∞ = 2.42e-01
k=   1: x = [0.532074, 1.007893, -0.498262], ||x*-x||∞ = 3.21e-02
k=   2: x = [0.501070, 1.000486, -0.499927], ||x*-x||∞ = 1.07e-03
k=   3: x = [0.500093, 1.000028, -0.499995], ||x*-x||∞ = 9.32e-05
k=   4: x = [0.500004, 1.000002, -0.500000], ||x*-x||∞ = 4.47e-06

最终结果:
迭代次数: 5
最终解: x = [0.50000447, 1.00000161, -0.49999974]
最终误差: ||x* - x||∞ = 4.47e-06

SOR 方法 ( = 1.0):
----------------------------------------
k=   0: x = [0.250000, 1.062500, -0.484375], ||x*-x||∞ = 2.50e-01
k=   1: x = [0.515625, 1.007812, -0.498047], ||x*-x||∞ = 1.56e-02
k=   2: x = [0.501953, 1.000977, -0.499756], ||x*-x||∞ = 1.95e-03
k=   3: x = [0.500244, 1.000122, -0.499969], ||x*-x||∞ = 2.44e-04
k=   4: x = [0.500031, 1.000015, -0.499996], ||x*-x||∞ = 3.05e-05
k=   5: x = [0.500004, 1.000002, -0.500000], ||x*-x||∞ = 3.81e-06

最终结果:
迭代次数: 6
最终解: x = [0.50000381, 1.00000191, -0.49999952]
最终误差: ||x* - x||∞ = 3.81e-06

SOR 方法 ( = 1.1):
----------------------------------------
k=   0: x = [0.275000, 1.175625, -0.501703], ||x*-x||∞ = 2.25e-01
k=   1: x = [0.570797, 1.001438, -0.499434], ||x*-x||∞ = 7.08e-02
k=   2: x = [0.493316, 0.998174, -0.500559], ||x*-x||∞ = 6.68e-03
k=   3: x = [0.500166, 1.000075, -0.499924], ||x*-x||∞ = 1.66e-04
k=   4: x = [0.500004, 1.000015, -0.500004], ||x*-x||∞ = 1.46e-05
k=   5: x = [0.500004, 0.999999, -0.500000], ||x*-x||∞ = 3.63e-06

最终结果:
迭代次数: 6
最终解: x = [0.50000363, 0.99999854, -0.50000004]
最终误差: ||x* - x||∞ = 3.63e-06

============================================================
结果比较:
```

```
------------------------------------------------------------
  值        迭代次数        最终误差              收敛状态
------------------------------------------------------------
1.03     5          4.47e-06      收敛
1.0      6          3.81e-06      收敛
1.1      6          3.63e-06      收敛

最优松弛因子:   = 1.03
最少迭代次数: 5
```

6 9

设有线性方程组 $A\mathbf{x} = \mathbf{b}$，其中 A 为对称正定矩阵，迭代公式

$$x^{(k+1)} = x^{(k)} + \omega(\mathbf{b} - Ax^{(k)}), \quad k = 0, 1, 2, \cdots$$

试证明当 $0 < \omega < \frac{2}{\beta}$ 时上述迭代法收敛，其中，$0 < \alpha \leq \lambda(A) \leq \beta$

6.1 Proof

整理得：

$$x^{(k+1)} = (I - \omega A)x^{(k)} + \omega\mathbf{b}$$

迭代矩阵为：$G = I - \omega A$

设 A 的特征值为 λ_i，对应的特征向量为 v_i，则：

$$Av_i = \lambda_i v_i$$

对于迭代矩阵 $G = I - \omega A$：

$$Gv_i = (I - \omega A)v_i = v_i - \omega Av_i = v_i - \omega\lambda_i v_i = (1 - \omega\lambda_i)v_i$$

因此，G 的特征值为：$\mu_i = 1 - \omega\lambda_i$

迭代收敛的充分必要条件是谱半径 $\rho(G) < 1$，即：

$$|1 - \omega\lambda_i| < 1, \quad \forall i$$

对于每个特征值 λ_i，需要：

$$|1 - \omega\lambda_i| < 1$$

这等价于：

$$-1 < 1 - \omega\lambda_i < 1$$

即：

$$-2 < -\omega\lambda_i < 0$$

$$0 < \omega\lambda_i < 2$$

由于 A 是对称正定矩阵，所有特征值 $\lambda_i > 0$，因此：

$$0 < \omega < \frac{2}{\lambda_i}, \quad \forall i$$

已知 $0 < \alpha \leq \lambda(A) \leq \beta$，即所有特征值都满足：

$$\alpha \leq \lambda_i \leq \beta$$

为了保证对所有特征值都有 $\omega < \frac{2}{\lambda_i}$，需要：

$$\omega < \min_i \frac{2}{\lambda_i} = \frac{2}{\max_i \lambda_i} = \frac{2}{\beta}$$

结合 $\omega > 0$ 的要求，得到：

$$0 < \omega < \frac{2}{\beta}$$

当 $0 < \omega < \frac{2}{\beta}$ 时，对于任意特征值 $\lambda_i \leq \beta$：

$$\omega\lambda_i < \frac{2}{\beta} \cdot \beta = 2$$

因此：

$$|1 - \omega\lambda_i| = |1 - \omega\lambda_i| < 1$$

（因为 $0 < \omega\lambda_i < 2$，所以 $-1 < 1 - \omega\lambda_i < 1$）

当 $0 < \omega < \frac{2}{\beta}$ 时，迭代矩阵 $G = I - \omega A$ 的谱半径 $\rho(G) < 1$，因此迭代法收敛。

证毕。