Student name: Abao Zhang (张宝)

Student number: 12332459

Homework 3

Question 1

(a) 是对称矩阵

(b) 不是对称矩阵

(c) 是对称矩阵

(d) 不是对称矩阵

Question 2

(a)

A 为 M x N 矩阵，X 为列向量，A 的每一行乘以 X，需要 N 次乘法，共有 M 行，总共需要进行$\boldsymbol{M \cdot N}$次乘法

(b)

A 为 M x N 矩阵，X 为列向量，A 的每一行乘以 X 再累加求和，每次求和需要$N-1$次加法，共有 M 行，总共需要进行$\boldsymbol{M \cdot (N-1)}$次乘法

Question 3

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1(N-2)} & a_{1(N-1)} & a_{1N} \\ 0 & a_{22} & \dots & a_{2(N-2)} & a_{2(N-1)} & a_{1N} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & a_{(N-1)(N-1)} & a_{(N-1)N} \\ 0 & 0 & \dots & 0 & 0 & a_{NN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_N \end{bmatrix}$$

如上公式，当时使用回代法求解每个变量x_i时，需要顺序计算$x_N, x_{N-1}, ..., x_2, x_1$。容易知道在求$x_N$，需要 1 次除法，0 次乘法，0 次加减法，

假设此时已知道x_N, x_{N-1}, x_k，要求解x_{k-1}，需要将$x_N, x_{N-1}, ..., x_k$的值代入方程$a_{k-1}x_{k-1} + a_k x_k + ... + a_{N-1}x_{N-1} + a_N x_N = y_{k-1}$中。此时需要 1 次除法，$N-k+1$次除法，$N-k+1$次加减法，

归纳并求和：

- 总除法次数 $= \overbrace{1+1+...+1}^{\text{N 个 1}} = N$
- 总乘法次数 $= 0 + 1 + ... + (N-2) + (N-1) = \frac{N(N-1)}{2}$
- 总加减法次数 $= 0 + 1 + ... + (N-2) + (N-1) = \frac{N(N-1)}{2}$

Question 4

(a)

在本问中，由于计算较为繁琐，且都是重复性计算，所以我编写了 `question-4.py` 进行计算验证，手写了高斯消元法，包括反代法、Partial pivoting、Scaled partial pivoting。下面仅给出 Scaled partial pivoting 的实现

```python
def gs_partial_scaled_pivot(A: np.ndarray, B: np.ndarray):
    M, M = A.shape
    AB = np.hstack((A, B))

    for p in range(M - 1):
        k = np.argmax(np.abs(AB[p:, p] / np.max(np.abs(AB[p:, p:]), axis=1))) + p
        if k > p:
            AB[[p, k], :] = AB[[k, p], :]

        for i in range(p + 1, M):
            AB[i, :] = AB[i, :] - AB[p, :] * AB[i, p] / AB[p, p]
    return back_substitution(AB)
```

下面给出每个小题的计算过程:

(i) Gaussian elimination with partial pivoting

$$[A|B] = \left[\begin{array}{ccc|c} \underline{2} & -3 & 100 & 1 \\ 1 & 10 & -0.001 & 0 \\ 3 & -100 & 0.01 & 0 \end{array}\right]$$

$$\xRightarrow{\text{pivot 1, 3 row}} \left[\begin{array}{ccc|c} \underline{3} & -100 & 0.01 & 0 \\ 1 & 10 & -0.001 & 0 \\ 2 & -3 & 100 & 1 \end{array}\right] \xRightarrow{m_{21}=\frac{1}{3}, m_{31}=\frac{2}{3}} \left[\begin{array}{ccc|c} 3 & -100 & 0.01 & 0 \\ 0 & \underline{43.3333} & -0.0043 & 0 \\ 0 & 63.6667 & 99.9933 & 1 \end{array}\right]$$

$$\xRightarrow{\text{pivot 2, 3 row}} \left[\begin{array}{ccc|c} 3 & -100 & 0.01 & 0 \\ 0 & \underline{63.6667} & 99.9933 & 1 \\ 0 & 43.3333 & -0.0043 & 0 \end{array}\right] \xRightarrow{m_{32}=\frac{130}{191}} \left[\begin{array}{ccc|c} 3 & -100 & 0.01 & 0 \\ 0 & 63.6667 & 99.9933 & 1 \\ 0 & 0 & -68.4208 & -0.6842 \end{array}\right]$$

使用反代法，可得

$$X = [0.0000 \ \ 0.0001 \ \ 0.010]^T$$

(ii) Gaussian elimination with scaled partial pivoting

$$[A|B] = \left[\begin{array}{ccc|c} \underline{2} & -3 & 100 & 1 \\ 1 & 10 & -0.001 & 0 \\ 3 & -100 & 0.01 & 0 \end{array}\right]$$

$$\xRightarrow{\text{pivot 1, 2 row}} \left[\begin{array}{ccc|c} \underline{1} & 10 & -0.001 & 0 \\ 2 & -3 & 100 & 1 \\ 3 & -100 & 0.01 & 0 \end{array}\right] \xRightarrow{m_{21}=2, m_{31}=3} \left[\begin{array}{ccc|c} 1 & 10 & -0.001 & 0 \\ 0 & \underline{-23} & 100.002 & 1 \\ 0 & -130 & 0.013 & 0 \end{array}\right]$$

$$\xRightarrow{\text{pivot 2, 3 row}} \left[\begin{array}{ccc|c} 1 & 10 & -0.001 & 0 \\ 0 & \underline{-130} & 0.013 & 0 \\ 0 & -23 & 100.002 & 1 \end{array}\right] \xRightarrow{m_{32}=\frac{23}{130}} \left[\begin{array}{ccc|c} 1 & 10 & -0.001 & 0 \\ 0 & \underline{-130} & 0.013 & 0 \\ 0 & 0 & 99.9997 & 1 \end{array}\right]$$

使用反代法，可得

$$X = [0.0000\ \ 0.0000\ \ 0.010]^T$$

(b)

(i) Gaussian elimination with partial pivoting

$$[A|B] = \left[\begin{array}{cccc|c} \underline{1} & 20 & -1 & 0.001 & 0 \\ 2 & -5 & 30 & -0.1 & 1 \\ 5 & 1 & -100 & -10 & 0 \\ 2 & -100 & -1 & 1 & 0 \end{array}\right]$$

$$\xRightarrow{\text{pivot 1, 3 row}} \left[\begin{array}{cccc|c} \underline{5} & 1 & -100 & -10 & 0 \\ 2 & -5 & 30 & -0.1 & 1 \\ 1 & 20 & -1 & 0.001 & 0 \\ 2 & -100 & -1 & 1 & 0 \end{array}\right] \xRightarrow{m_{21}=\frac{2}{5}, m_{31}=\frac{1}{5}, m_{41}=\frac{2}{5}} \left[\begin{array}{cccc|c} 5 & -1 & -100 & -10 & 0 \\ 0 & -4.6 & 70 & 3.9 & 1 \\ 0 & 20.2 & 19 & 2.001 & 0 \\ 0 & -99.6 & 39 & 3 & 0 \end{array}\right]$$

$$\xRightarrow{\text{pivot 2, 4 row}} \left[\begin{array}{cccc|c} 5 & -1 & -100 & -10 & 0 \\ 0 & -99.6 & 39 & 3 & 0 \\ 0 & 20.2 & 19 & 2.001 & 0 \\ 0 & -4.6 & 70 & 3.9 & 1 \end{array}\right] \xRightarrow{\text{消元}} \left[\begin{array}{cccc|c} 5 & -1 & -100 & -10 & 0 \\ 0 & -99.6 & 39 & 3 & 0 \\ 0 & 0 & 26.9096 & 2.6094 & 0 \\ 0 & 0 & 68.1988 & 3.7614 & 1 \end{array}\right]$$

$$\xRightarrow{\text{pivot 3, 4 row}} \left[\begin{array}{cccc|c} 5 & -1 & -100 & -10 & 0 \\ 0 & -99.6 & 39 & 3 & 0 \\ 0 & 0 & 68.1988 & 3.7614 & 1 \\ 0 & 0 & 26.9096 & 2.6094 & 0 \end{array}\right] \xRightarrow{\text{消元}} \begin{bmatrix} 5 & -1 & -100 & -10 & 0 \\ 0 & -99.6 & 39 & 3 & 0 \\ 0 & 0 & 68.1988 & 3.7614 & 1 \\ 0 & 0 & 0 & 1.1253 & -0.3946 \end{bmatrix}$$

使用反代法，可得

$$X = [-0.0207\ \ 0.0028\ \ 0.0340\ \ -0.3508]^T$$

(ii) Gaussian elimination with scaled partial pivoting

$$[A|B] = \left[\begin{array}{cccc|c} \underline{1} & 20 & -1 & 0.001 & 0 \\ 2 & -5 & 30 & -0.1 & 1 \\ 5 & 1 & -100 & -10 & 0 \\ 2 & -100 & -1 & 1 & 0 \end{array}\right]$$

$$\xrightarrow{\text{pivot 1, 2 row}} \left[\begin{array}{cccc|c} 2 & -5 & 30 & -0.1 & 1 \\ 1 & 20 & -1 & 0.001 & 0 \\ 5 & 1 & -100 & -10 & 0 \\ 2 & -100 & -1 & 1 & 0 \end{array}\right] \xrightarrow{\text{消元}} \left[\begin{array}{cccc|c} 2 & -5 & 30 & -0.1 & 1. \\ 0 & 22.5 & -16 & 0.051 & -0.5 \\ 0 & 11.5 & -175 & -9.75 & -2.5 \\ 0 & -95. & -31 & -0.9 & -1. \end{array}\right]$$

$$\xrightarrow{\text{消元}} \left[\begin{array}{cccc|c} 2 & -5 & 30 & -0.1 & 1 \\ 0 & 22.5 & -16 & 0.051 & -0.5 \\ 0 & 0 & -166.8222 & -9.7761 & -2.2444 \\ 0 & 0 & -98.5556 & -0.6847 & -3.1111 \end{array}\right]$$

$$\xrightarrow{\text{消元}} \left[\begin{array}{cccc|c} 2 & -5 & 30. & -0.1 & 1 \\ 0 & 22.5 & -16. & 0.051 & -0.5 \\ 0 & 0 & -166.8222 & -9.7761 & -2.2444 \\ 0 & 0 & 0 & 5.0909 & -1.7851 \end{array}\right]$$

使用反代法，可得

$$X = [-0.0207\ \ 0.0028\ \ 0.03400\ \ -0.3507]^T$$

Question 5

$$A=\begin{bmatrix}1 & 1 & 0 & 4\\ 2 & -1 & 5 & 0\\ 5 & 2 & 1 & 2\\ -3 & 0 & 2 & 6\end{bmatrix}$$

根据按列消元的逻辑有

$$A-\begin{bmatrix}1\\2\\5\\-3\end{bmatrix}\begin{bmatrix}1 & 1 & 0 & 4\end{bmatrix}=\begin{bmatrix}0 & 0 & 0 & 0\\ 0 & -3 & 5 & -8\\ 0 & -3 & 1 & 18\\ 0 & 3 & 2 & 18\end{bmatrix}\overset{\text{def}}{=}A_1$$

$$A_1-\begin{bmatrix}0\\1\\1\\-1\end{bmatrix}\begin{bmatrix}0 & -3 & 5 & -8\end{bmatrix}=\begin{bmatrix}0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & -4 & -10\\ 0 & 0 & 7 & 10\end{bmatrix}\overset{\text{def}}{=}A_2$$

$$A_2-\begin{bmatrix}0\\0\\1\\-\frac{7}{4}\end{bmatrix}\begin{bmatrix}0 & 0 & -4 & -10\end{bmatrix}=\begin{bmatrix}0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & -\frac{15}{2}\end{bmatrix}\overset{\text{def}}{=}A_3$$

$$A_3-\begin{bmatrix}0\\0\\0\\1\end{bmatrix}\begin{bmatrix}0 & 0 & 0 & -\frac{15}{2}\end{bmatrix}=\mathrm{O}$$

也就是

$$A=\begin{bmatrix}1\\2\\5\\-3\end{bmatrix}\begin{bmatrix}1 & 1 & 0 & 4\end{bmatrix}+\begin{bmatrix}0\\1\\1\\-1\end{bmatrix}\begin{bmatrix}0 & -3 & 5 & 8\end{bmatrix}+\begin{bmatrix}0\\0\\1\\-\frac{7}{4}\end{bmatrix}\begin{bmatrix}0 & 0 & -4 & -10\end{bmatrix}+\begin{bmatrix}0\\0\\0\\1\end{bmatrix}\begin{bmatrix}0 & 0 & 0 & -\frac{15}{2}\end{bmatrix}$$

将右侧改写成矩阵相乘形式即为

$$L=\begin{bmatrix}1 & 0 & 0 & 0\\ 2 & 1 & 0 & 0\\ 5 & 1 & 1 & 0\\ -3 & -1 & -\frac{7}{4} & 1\end{bmatrix}$$

$$U=\begin{bmatrix}1 & 1 & 0 & 4\\ 0 & -3 & 5 & -8\\ 0 & 0 & -4 & -10\\ 0 & 0 & 0 & -\frac{15}{2}\end{bmatrix}$$

Question 6

本文可以使用计算器进行计算，我使用脚本 `question-6.py` 进行计算== (a) Jacobi iteration 根据以下迭代公式：

$$
\begin{aligned}
x_{k+1} &= \frac{1}{4}(13 - y_k + z_k) \\
y_{k+1} &= \frac{1}{5}(x_k - z_k + 8) \\
z_{k+1} &= \frac{1}{6}(2x_k - y_k + 2)
\end{aligned}
$$

可以计算（保留至多四位有效小数）:

$$
\begin{aligned}
P_1 &= (x_1, y_1, z_1) = (3.25, 1.6, 0.3333) \\
P_2 &= (x_2, y_2, z_2) = (2.9333, 2.1833, 1.15) \\
P_3 &= (x_3, y_3, z_3) = (2.9916, 1.9567, 0.9472)
\end{aligned}
$$

迭代能收敛

(b) Gauss-Seidel iteration

根据以下迭代公式：

$$
\begin{aligned}
x_{k+1} &= \frac{1}{4}(13 - y_k + z_k) \\
y_{k+1} &= \frac{1}{5}(x_{k+1} - z_k + 8) \\
z_{k+1} &= \frac{1}{6}(2x_{k+1} - y_{k+1} + 2)
\end{aligned}
$$

可以计算（保留至多四位有效小数）:

$$
\begin{aligned}
P_1 &= (x_1, y_1, z_1) = (3.25, 2.25, 1.0417) \\
P_2 &= (x_2, y_2, z_2) = (2.9479, 1.9812, 0.9858) \\
P_3 &= (x_3, y_3, z_3) = (3.0011, 2.0030, 0.9999)
\end{aligned}
$$

迭代能收敛

Question 7

本题核心思路为通分，然后通过合并同类项得到多项式方程，通过多项式对应系数建立方程组。

在计算中，使用 python 中的 sympy 库进行通分、合并同类型的计算。 具体代码如下

```python
import sympy as sp
x, A1, A2, A3, A4, A5, A6 = sp.symbols("x, A1, A2,A3, A4, A5, A6")

left = (x**2 + x + 1) / ((x - 1) * (x - 2) * (x - 3) ** 2 * (x**2 + 1))

right = (
    (1 / (x - 1)) * A1
    + (1 / (x - 2)) * A2
    + (1 / (x - 3) ** 2) * A3
    + A4 / (x - 3)
    + (A5 * x + A6) / (x**2 + 1)
)
# 同时乘以分母并简化表达式
left = sp.simplify(left * ((x - 1) * (x - 2) * (x - 3) ** 2 * (x**2 + 1)))
right = sp.simplify(right * ((x - 1) * (x - 2) * (x - 3) ** 2 * (x**2 + 1)))
# 等式右边按照 x 的多项式进行表达
right = right.expand().collect(x)
```

结果如下：

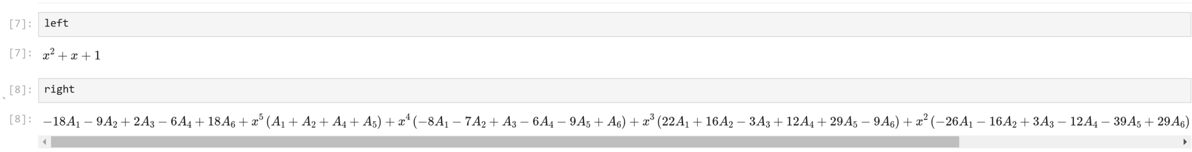

Figure 1: 通分后的关于x的多项式表达式（Jupyter Notebook 渲染）

根据多项式对应系数相等构造关于$\mathbb{A}=(A_1,A_2,A_3,A_4,A_5,A_6)^T$的线性方程组，这里简单使用 sympy 的 API 即可

```python
A = [A1, A2, A3, A4, A5, A6]

b = [left.coeff(x, i) for i in reversed(range(6))]
E = [right.coeff(x, i) for i in reversed(range(6))]
```

```
E
```

[11]:

```
[A1 + A2 + A4 + A5,
 -8*A1 - 7*A2 + A3 - 6*A4 - 9*A5 + A6,
 22*A1 + 16*A2 - 3*A3 + 12*A4 + 29*A5 - 9*A6,
 -26*A1 - 16*A2 + 3*A3 - 12*A4 - 39*A5 + 29*A6,
 21*A1 + 15*A2 - 3*A3 + 11*A4 + 18*A5 - 39*A6,
 -18*A1 - 9*A2 + 2*A3 - 6*A4 + 18*A6]
```

[12]:

```
b
```

[12]:

```
[0, 0, 0, 1, 1, 1]
```

Figure 2: 关于$\mathbb{A}$的线性方程组（Jupyter Notebook 渲染）

转换为矩阵表达

```
M = [[right.coeff(x, i).coeff(A[j]) for j in range(6)] for i in reversed(range(5))]
```

得到：

$$M = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 \\ -8 & -7 & 1 & -6 & -9 & 1 \\ 22 & 16 & -3 & 12 & 29 & -9 \\ -26 & -16 & 3 & -12 & -39 & 29 \\ 21 & 15 & -3 & 11 & 18 & -39 \\ -18 & -9 & 2 & -6 & 0 & 18 \end{bmatrix}$$

$$b = [0 \;\; 0 \;\; 0 \;\; 1 \;\; 1 \;\; 1]^T$$

也就是求解$M\mathbb{A} = b$，由于 Program 3-3 为 Matlab 代码，这里使用 sympy.solve 进行矩阵的方程组的求解

```
result = sp.solve([e - i for e, i in zip(E, b)], A)
```

结果为

$$\mathbb{A} = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \\ A_5 \\ A_6 \end{bmatrix} = \begin{bmatrix} -\frac{3}{8} \\ \frac{7}{5} \\ \frac{13}{20} \\ -\frac{203}{200} \\ -\frac{1}{100} \\ -\frac{3}{100} \end{bmatrix}$$

具体代码请参考 question-7.py

Question 8

先介绍算法实现，采用$X_0 = 0$作为初始化解。:

```python
def gs_iteration(A: np.ndarray, B: np.ndarray, max_iter=10000, max_residual=0.0001):
    X = np.zeros(N)  # 初始化 X_0
    # 最多迭代 max_iter 次
    for i in range(max_iter):
        X = X.copy()  # 拷贝一份 Xi
        for j in range(N):
            # 顺序更新 xi (即每个未知数)
            X[j] = (
                B[j] - np.sum(A[j, :j] * X[:j]) - np.sum(A[j, j + 1 :] * X[j + 1 :])
            ) / A[j, j]
        # 所有的未知数都更新了一遍，计算误差（2-norm）
        residual = np.linalg.norm(A @ X - B)
        # 当误差小于要求值
        if max_residual > residual:
            return X

    return X
```

对于题目给出的方程组，可以发现其是对称带状矩阵，根据其特性可以用代码推算出矩阵A:

```python
N = 50
A = np.zeros((N, N))
for j in range(N):
    A[j][j] = 12
    if j > 0:
        A[j][j - 1] = -2
    if j > 1:
        A[j][j - 2] = 1

    if j < N - 1:
        A[j][j + 1] = -2
    if j < N - 2:
        A[j][j + 2] = 1


B = np.ones(N) * 5
```

调用 gs_iteration()函数：结果如下

```
[3]: gs_iteration(A, B)

[3]: array([0.4637955 , 0.53728458, 0.5090229 , 0.49822161, 0.49894183,
           0.49998533, 0.5000887 , 0.50001529, 0.49999477, 0.49999783,
           0.50000008, 0.50000018, 0.5       , 0.49999996, 0.49999997,
           0.49999997, 0.49999997, 0.49999997, 0.49999997, 0.49999997,
           0.49999997, 0.49999997, 0.49999997, 0.49999997, 0.49999997,
           0.49999997, 0.49999997, 0.49999997, 0.49999997, 0.49999997,
           0.49999997, 0.49999997, 0.49999997, 0.49999997, 0.49999997,
           0.49999997, 0.49999997, 0.49999997, 0.49999982, 0.5000015 ,
           0.49999598, 0.49999486, 0.50001679, 0.50008847, 0.49998452,
           0.49894174, 0.49822182, 0.50902299, 0.53728459, 0.46379552])
```

Figure 3: 高斯-塞德尔迭代法在给定误差/最大迭代次数下的X结果