

EEE5062计算方法 作业九

习题P275: 1、3(1)

计算实习题P277: 1 (代码、输出、备注等, 清晰截图)

作业提交DDL: 2022/5/24 16:20前

姓名: 江宇辰

学号: 11812419

提交时间: 2022.05.22

Q1

1. 利用格什戈林圆盘定理估计下面矩阵特征值的界:

$$(1) \begin{pmatrix} -1 & 0 & 0 \\ -1 & 0 & 1 \\ -1 & -1 & 2 \end{pmatrix}; \quad (2) \begin{pmatrix} 4 & -1 & & & \\ -1 & 4 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 4 & -1 \\ & & & -1 & 4 \end{pmatrix}.$$

解: (1) 根据格什戈林圆盘定理, 矩阵的特征值 λ_i 分别位于圆盘 $|\lambda_1 - (-1)| \leq 2, |\lambda_2 - 0| \leq 1, |\lambda_3 - 2| \leq 1$ 之内

即矩阵特征值的界为 $-3 \leq \lambda_1 \leq 1, -1 \leq \lambda_2 \leq 1, 1 \leq \lambda_3 \leq 3$

(2) 根据格什戈林圆盘定理, 矩阵的特征值 λ_i 位于圆盘 $|\lambda_i - 4| \leq 2, i = 1, 2, \dots, n$ 之内

即矩阵特征值的界为 $2 \leq \lambda_i \leq 6, i = 1, 2, \dots, n$

Q3 (1)

3. 用幂法计算下列矩阵的主特征值及对应的特征向量:

$$(1) A_1 = \begin{pmatrix} 7 & 3 & -2 \\ 3 & 4 & -1 \\ -2 & -1 & 3 \end{pmatrix};$$

当特征值有 3 位小数稳定时迭代终止.

解: 使用幂法公式, $u_0 \neq 0, v_k = Au_{k-1}, u_k = \frac{v_k}{\max(v_k)}, k = 1, 2, \dots$

取 $u_0 = (1, 1, 1)^T \neq 0$, 根据给定的 A_1 , 代入得:

k	u_k^T	$\max(v_k)$
1	[1., 0.75, 0.]	8
2	[1., 0.64864865, -0.2972973]	9.25
3	[1., 0.61756374, -0.37110482]	9.54054054054054
4	[1., 0.60879835, -0.38883968]	9.594900849858357
5	[1., 0.60641274, -0.39309539]	9.604074402125775
6	[1., 0.60577683, -0.39412075]	9.605429001813766
7	[1., 0.60560975, -0.39436892]	9.60557200236834

故 A_1 的主特征值 $\lambda_1 = 9.60557200236834$, 特征向量 $x_1 = [1, 0.60560975, -0.39436892]^T$

计算实习题Q1

1. 已知矩阵

$$A = \begin{bmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{bmatrix}, \quad B = \begin{bmatrix} 2 & 3 & 4 & 5 & 6 \\ 4 & 4 & 5 & 6 & 7 \\ 0 & 3 & 6 & 7 & 8 \\ 0 & 0 & 2 & 8 & 9 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad H_6 = \begin{bmatrix} 1 & \frac{1}{2} & \cdots & \frac{1}{6} \\ \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{7} \\ \vdots & \vdots & & \vdots \\ \frac{1}{6} & \frac{1}{7} & \cdots & \frac{1}{11} \end{bmatrix}.$$

(1) 用 MATLAB 函数“eig”求矩阵全部特征值.

(2) 用基本 QR 算法求全部特征值(可用 MATLAB 函数“qr”实现矩阵的 QR 分解).

(1) Using python, and numpy.linalg.eig() provides us with the same function as eig in matlab.

```
import numpy as np

# load target matrix
A = np.array([[10, 7, 8, 7], [7, 5, 6, 5], [8, 6, 10, 9], [7, 5, 9, 10]])
B = np.array([[2, 3, 4, 5, 6], [4, 4, 5, 6, 7], [0, 3, 6, 7, 8], [0, 0, 2, 8, 9], [0, 0, 0, 1, 0]])
H_6 = np.array([[1, 1 / 2, 1 / 3, 1 / 4, 1 / 5, 1 / 6], [1 / 2, 1 / 3, 1 / 4, 1 / 5, 1 / 6, 1 / 7],
                 [1 / 3, 1 / 4, 1 / 5, 1 / 6, 1 / 7, 1 / 8],
                 [1 / 4, 1 / 5, 1 / 6, 1 / 7, 1 / 8, 1 / 9], [1 / 5, 1 / 6, 1 / 7, 1 / 8, 1 / 9, 1 / 10],
                 [1 / 6, 1 / 7, 1 / 8, 1 / 9, 1 / 10, 1 / 11]])

# get eigen values by numpy.linalg.eig() function
# the function returns eigen values and vectors in order, choose first item to print
eigen_A = np.linalg.eig(A)
print(eigen_A[0]) # [3.02886853e+01 3.85805746e+00 1.01500484e-02 8.43107150e-01]
eigen_B = np.linalg.eig(B)
```

```

print(eigen_B[0]) # [13.1723514  6.55187835  1.59565457 -0.39078805
-0.92909628]
eigen_H_6 = np.linalg.eig(H_6)
print(eigen_H_6[0]) # [1.61889986e+00 2.42360871e-01 1.63215213e-02
6.15748354e-04 1.25707571e-05 1.08279948e-07]

```

Output screenshot:

```

import numpy as np
# load target matrix
A = np.array([[10, 7, 8, 7], [7, 5, 6, 5], [8, 6, 10, 9], [7, 5, 9, 10]])
B = np.array([[2, 3, 4, 5, 6], [4, 4, 5, 6, 7], [0, 3, 6, 7, 8], [0, 0, 2, 8, 9], [0, 0, 0, 1, 0]])
H_6 = np.array([[1, 1 / 2, 1 / 3, 1 / 4, 1 / 5, 1 / 6], [1 / 2, 1 / 3, 1 / 4, 1 / 5, 1 / 6, 1 / 7],
[1 / 3, 1 / 4, 1 / 5, 1 / 6, 1 / 7, 1 / 8],
[1 / 4, 1 / 5, 1 / 6, 1 / 7, 1 / 8, 1 / 9], [1 / 5, 1 / 6, 1 / 7, 1 / 8, 1 / 9, 1 / 10],
[1 / 6, 1 / 7, 1 / 8, 1 / 9, 1 / 10, 1 / 11]])

# get eigen values by numpy.linalg.eig() function
# the function returns eigen values and vectors in order, choose first item to print
eigen_A = np.linalg.eig(A)
print(eigen_A[0]) # [3.02886853e+01 3.85805746e+00 1.01500484e-02 8.43107150e-01]
eigen_B = np.linalg.eig(B)
print(eigen_B[0]) # [13.1723514  6.55187835  1.59565457 -0.39078805 -0.92909628]
eigen_H_6 = np.linalg.eig(H_6)
print(eigen_H_6[0]) # [1.61889986e+00 2.42360871e-01 1.63215213e-02 6.15748354e-04 1.25707571e-05 1.08279948e-07]

```

Run: hw9_eigen.py

```

D:\Anaconda3\envs\zyfd\python.exe D:/SUSTech/PreStudy-S0/EEE5062计算方法/hw9_eigen.py
[3.02886853e+01 3.85805746e+00 1.01500484e-02 8.43107150e-01]
[13.1723514  6.55187835  1.59565457 -0.39078805 -0.92909628]
[1.61889986e+00 2.42360871e-01 1.63215213e-02 6.15748354e-04 1.25707571e-05 1.08279948e-07]
Process finished with exit code 0

```

(2) Using python, and `numpy.linalg.qr()` provides us with the same function as `qr` in matlab. Then iterations start until reaching limitations.

```

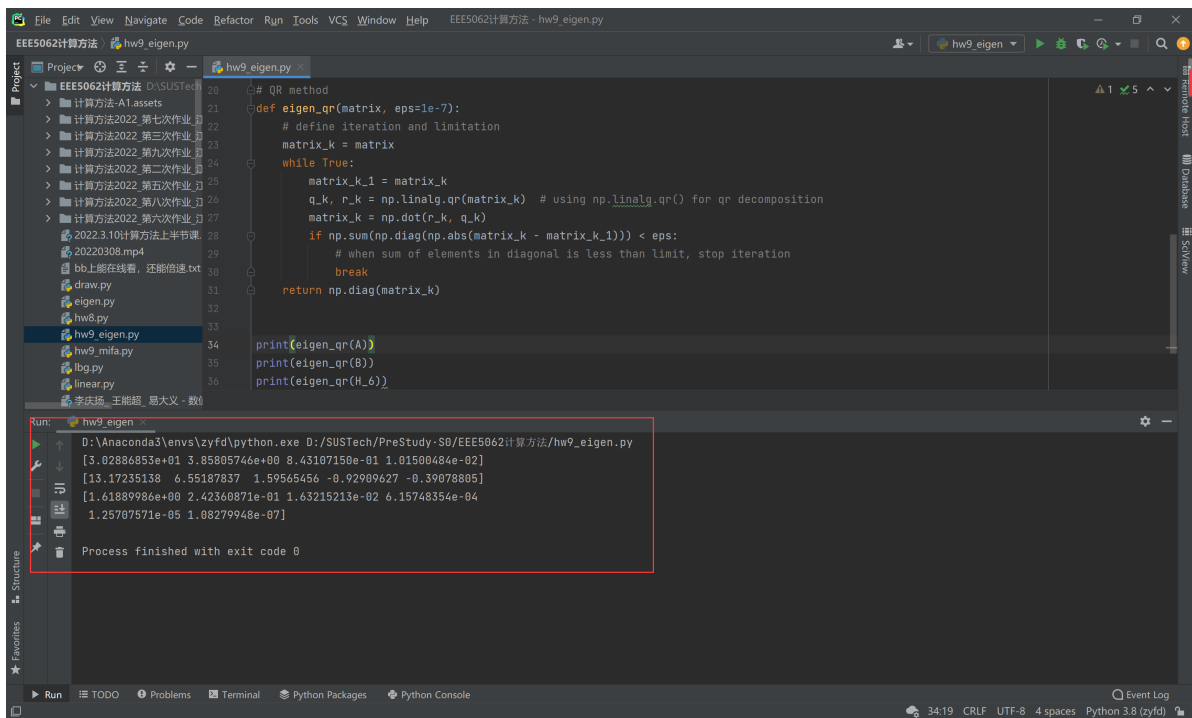
import numpy as np
# load target matrix
A = np.array([[10, 7, 8, 7], [7, 5, 6, 5], [8, 6, 10, 9], [7, 5, 9, 10]])
B = np.array([[2, 3, 4, 5, 6], [4, 4, 5, 6, 7], [0, 3, 6, 7, 8], [0, 0, 2, 8, 9], [0, 0, 0, 1, 0]])
H_6 = np.array([[1, 1 / 2, 1 / 3, 1 / 4, 1 / 5, 1 / 6], [1 / 2, 1 / 3, 1 / 4, 1 / 5, 1 / 6, 1 / 7],
[1 / 3, 1 / 4, 1 / 5, 1 / 6, 1 / 7, 1 / 8],
[1 / 4, 1 / 5, 1 / 6, 1 / 7, 1 / 8, 1 / 9], [1 / 5, 1 / 6, 1 / 7, 1 / 8, 1 / 9, 1 / 10],
[1 / 6, 1 / 7, 1 / 8, 1 / 9, 1 / 10, 1 / 11]])

# QR method
def eigen_qr(matrix, eps=1e-7):
    # define iteration and limitation
    matrix_k = matrix
    while True:
        matrix_k_1 = matrix_k
        q_k, r_k = np.linalg.qr(matrix_k) # using np.linalg.qr() for qr decomposition
        matrix_k = np.dot(r_k, q_k)
        if np.sum(np.diag(np.abs(matrix_k - matrix_k_1))) < eps:
            # when sum of elements in diagonal is less than limit, stop iteration
            break
    return np.diag(matrix_k)

```

```
print(eigen_qr(A))
print(eigen_qr(B))
print(eigen_qr(H_6))
```

Output screenshot:



The screenshot shows an IDE window titled "EEE5062计算方法 - hw9_eigen.py". The editor displays a Python script for QR decomposition. The script defines a function `eigen_qr(matrix, eps=1e-7)` that iteratively refines a matrix `matrix_k` using QR decomposition until the sum of absolute values of the off-diagonal elements is less than `eps`. The main code calls `eigen_qr(A)`, `eigen_qr(B)`, and `eigen_qr(H_6)`.

The Run window shows the output of the script:

```
D:\Anaconda3\envs\zyfd\python.exe D:/SUSTech/PreStudy/S0/EEE5062计算方法/hw9_eigen.py
[3.02886853e+01 3.85805746e+00 8.43107150e-01 1.01500484e-02]
[13.17235138  6.55187837  1.59565456 -0.92909627 -0.39078805]
[1.61889986e+00 2.42360871e-01 1.63215213e-02 6.15748354e-04
 1.25707571e-05 1.08279948e-07]
Process finished with exit code 0
```