CompSci 206 PS3 NLA 29.1

Xiaoying Li

11/10/2021

(a)

```
import numpy as np


def tridiag(A):
    s = len(A)

    for p in range(1, s-1):
        vector = np.zeros((s, 1))
        vector[p:s, 0] = A[p:s, p-1]
        vector = vector/np.linalg.norm(vector)

        if vector[p, 0] < 0:
            vector = -vector

        vector[p, 0] = vector[p, 0] + 1
        alpha = -vector[p, 0]
        Q = np.dot(A, vector)/alpha
        beta = np.dot(Q.T, vector) / (2*alpha)
        Q = Q + beta * vector
        A = A + np.dot(vector, Q.T) + np.dot(Q, vector.T)
        A[p-1, p+1:s] = np.zeros((1, s-p-1))

    return A
```

Result of applying the program to A=hilb(4):

[[1.00000000e+00 -6.50854140e-01 0.00000000e+00 0.00000000e+00]

 [-6.50854140e-01 6.50585480e-01 6.39118800e-02 0.00000000e+00]

 [0.00000000e+00 6.39118800e-02 2.53201434e-02 -1.16520804e-03]

 [0.00000000e+00 -1.38777878e-17 -1.16520804e-03 2.84852680e-04]]

(b)

```
import numpy as np


def qralg(T):
    m = len(T)

    while abs(T[m-1, m-2]) >= 1e-12:
        Q, R = np.linalg.qr(T)
        T = np.dot(R, Q)
    Tnew = T

    return Tnew
```

Result of applying the program to A=hilb(4):

[[1.50021428e+00 -1.78075254e-05 -6.35504070e-17 7.86474555e-18]

 [-1.78075254e-05 1.69141220e-01 5.62874379e-09 -8.47223450e-18]

 [0.00000000e+00 5.62874380e-09 6.73827361e-03 6.88695494e-13]

 [0.00000000e+00 7.22389445e-34 6.88691517e-13 9.67023040e-05]]

[[1.00000000e+00 1.51398284e-01 6.55100063e-03 9.67023040e-05]]

(c)

```
import matplotlib.pyplot as plt
import numpy as np


# QR algorithm implementation with shifts of Rayleigh
def qralg_ray(A):
    n = len(A)
    vector = []

    while abs(A[n-1, n-2]) >= 1e-12:
        vector.append(abs(A[n-1, n-2]))
        Q, R = np.linalg.qr(A - np.dot(A[n-1, n-1], np.eye(n, n)))
        A = np.dot(R, Q) + np.dot(A[n-1, n-1], np.eye(n, n))

    lam = A[n-1, n-1]
    vector = vector
    newA = A

    return lam, vector, newA
```

```
# Driver program
if __name__ == "__main__":
    A = np.array([1, 1/2, 1/3, 1/4, 1/2, 1/3, 1/4, 1/5, 1/3, 1/4, 1/5,
1/6, 1/4, 1/5, 1/6, 1/7]).reshape(4, 4)
    T = tridiag(A)
    print("a result:\n", T)
    Tnew = qralg(T)
    print("b result:\n", Tnew)

    H = A
    sizeH = np.shape(H)[0]
    sub_diag = []
    eigenvalues = np.zeros((sizeH, 1))

    for i in range(sizeH-1, 0, -1):
        _lambda, vector, T1 = qralg_ray(T[0:i+1, 0:i+1])
        eigenvalues[i] = _lambda
        sub_diag.extend(vector)

    sub_diag.append(T[1, 0])
    eigenvalues[0] = T[0, 0]
    print(eigenvalues.T)

    plt.semilogy(range(0, len(sub_diag)), sub_diag)
    plt.show()
```

Output:

Eigenvalues = [[1.50020000e+00 1.51398284e-01 6.55100063e-03 9.67023040e-05]]