Assignment 3

Arifur Rahman
400300356
Feb 11, 2023

```
import numpy as np

def column_convertor(x):
    x.shape = (1, x.shape[0])
    return x

def get_norm(x):
    return np.sqrt(np.sum(np.square(x)))

def householder_transformation(v):
    vector_size = v.shape[1]
    e = np.zeros_like(v)
    e[0, 0] = 1
    vector = get_norm(v) * e
    if v[0,0] < 0:
        vector = - vector
    updatedV = (v + vector).astype(np.float32)
    H = np.identity(vector_size) - ((2 * np.matmul(np.transpose(updatedV),
updatedV)) / np.matmul(updatedV, np.transpose(updatedV)))
    return H

def qr_factorization(A):
    n, m = A.shape
    Q = np.identity(n)
    R = A.astype(np.float32)
    for i in range(min(n, m)):
        v = column_convertor(R[i:, i])
        Hbar = householder_transformation(v)
        H = np.identity(n)
        H[i:, i:] = Hbar
        R = np.matmul(H, R)
        Q = np.matmul(Q, H)
        R = np.around(R, decimals=5)
        Q = np.around(Q, decimals=5)
        print(f"Step ======= {i+1} =========")
        print(f"Q: {Q} \n R: {R}")

    return Q, R

if __name__ == "__main__":
    A = np.array([[1, -1, 4], [1, 4, -2], [1, 4, 2], [1, -1, 0]])
    Q, R = qr_factorization(A)
```

```
    R = np.around(R, decimals=5)
    Q = np.around(Q, decimals=5)
    print('After QR factorization')
    print('R matrix:')
    print(R, '\n')
    print('Q matrix:')
    print(Q)
```

```
OUTPUT:
Step ======= 1 =========
Q: [[-0.5     -0.5     -0.5     -0.5    ]
 [-0.5      0.83333 -0.16667 -0.16667]
 [-0.5     -0.16667  0.83333 -0.16667]
 [-0.5     -0.16667 -0.16667  0.83333]]
 R: [[-2.      -3.      -2.     ]
 [-0.       3.33333 -4.     ]
 [-0.       3.33333  0.     ]
 [-0.      -1.66667 -2.     ]]
Step ======= 2 =========
Q: [[-0.5      0.5     -0.1     -0.7    ]
 [-0.5     -0.5     -0.7      0.1    ]
 [-0.5     -0.5      0.7     -0.1    ]
 [-0.5      0.5      0.1      0.69999]]
 R: [[-2.  -3.  -2. ]
 [ 0.  -5.   2. ]
 [ 0.  -0.   2.4]
 [ 0.  -0.  -3.2]]
Step ======= 3 =========
Q: [[-0.5      0.5     -0.5     -0.5    ]
 [-0.5     -0.5      0.5     -0.5    ]
 [-0.5     -0.5     -0.5      0.5    ]
 [-0.5      0.5      0.49999  0.49999]]
 R: [[-2. -3. -2.]
 [ 0. -5.  2.]
 [ 0.  0. -4.]
 [ 0.  0.  0.]]
```

```
After QR factorization
R matrix:
[[-2. -3. -2.]
 [ 0. -5.  2.]
 [ 0.  0. -4.]
 [ 0.  0.  0.]]

Q matrix:
[[-0.5      0.5     -0.5     -0.5    ]
 [-0.5     -0.5      0.5     -0.5    ]
 [-0.5     -0.5     -0.5      0.5    ]
 [-0.5      0.5      0.49999  0.49999]]
```