Project 3

Name: Wang Fangyu Student ID: MAT2309449 Marks:_____

Computer Project (due at 11:59 pm on Jun 30).

1. (a) Write your own pure QR algorithm code by directly calling the QR decomposition function in Matlab. Use it to find all the eigenvalues of

$$\begin{bmatrix} 4 & 3 & 1 \\ 3 & -3 & 0 \\ 1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} -7 & -8 & 1 \\ 17 & 18 & -1 \\ -8 & -8 & 2 \end{bmatrix}.$$

[5 marks]

(b) Explain why the pure QR method does not work for

$$\begin{bmatrix} 4 & 3 & 1 \\ -5 & -3 & 0 \\ 3 & 2 & 1 \end{bmatrix}.$$

Extend the pure QR code so that it can handle the above matrix. Use it to find all the eigenvalues of the above matrix. [5 marks]

Solution

(a) **Julia Code**

```
using LinearAlgebra

function    qr_my(a)
    tol = 1e-14
    max_iterate = 1000
    n = size(a, 1)
    t = 1
    while maximum(abs(a[i,j]) for i in 2:n for j in 1:i-1) > tol && t < max_iterate
          F = qr(a)
          a = F.R * F.Q
          t = t + 1
    end
    println("%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if t < max_iterate
```

```julia
        println("Success after $t times of pure QR")
    else
        println("Not success after 1000 times of pure QR")
    end
    println("The Final matrix is:")
    display(round.(a, digits = 4))
    println("The eigenvalues are:")
    display(round.(diag(a), digits = 4))
    println("%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
end

A = [4.0   3.0   1.0;
     3.0  -3.0   0.0;
     1.0   0.0   1.0]
qr_my(A)

B = [-7.0  -8.0   1.0;
     17.0  18.0  -1.0;
     -8.0  -8.0   2.0]
qr_my(B)
```

Julia Output

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Success after 135 times of pure QR
The Final matrix is:
3×3 Matrix{Float64}:
 5.3143   0.0      0.0
 0.0     -4.1337  0.0
 0.0     -0.0      0.8194
The eigenvalues are:
3-element Vector{Float64}:
  5.3143
 -4.1337
  0.8194
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Success after 95 times of pure QR
```

```
The Final matrix is:
3×3 Matrix{Float64}:
 10.0  -22.6274  15.5885
  0.0    2.0       0.0
  0.0    0.0       1.0
The eigenvalues are:
3-element Vector{Float64}:
 10.0
  2.0
  1.0
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Therefore, we obtain the eigenvalues of $\begin{bmatrix} 4 & 3 & 1 \\ 3 & -3 & 0 \\ 1 & 0 & 1 \end{bmatrix}$ are $5.3143, -4.1337, 0.8194$, and the eigenvalues of $\begin{bmatrix} -7 & -8 & 1 \\ 17 & 18 & -1 \\ -8 & -8 & 2 \end{bmatrix}$ are $10.0, 2.0, 1.0$.

(b) The reason why pure QR method does not work for $\begin{bmatrix} 4 & 3 & 1 \\ -5 & -3 & 0 \\ 3 & 2 & 1 \end{bmatrix}$ is that there exist complex eigenvalues of this matrix.

Julia Code of Extended pure QR

```
using LinearAlgebra

function    qr_my(a)
    tol = 1e-14
    max_iterate = 1000
    n = size(a, 1)
    t = 1
    while maximum(abs(a[i,j]) for i in 2:n for j in 1:i-1) > tol && t < max_iterate
          F = qr(a)
          a = F.R * F.Q
          t = t + 1
    end
    println("%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if t < max_iterate
        println("Success after $t times of pure QR")
        println("The Final matrix is:")
```

```
        display(round.(a, digits = 4))
        println("The eigenvalues are:")
        display(round.(diag(a), digits = 4))
        println("%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    else
        println("Not success after 1000 times of pure QR, after use another method, we
        shiftedqr(a)
    end
end

function shiftedqr(a)
    tol = 1e-14
    kounttol = 500
    m = size(a, 1)
    lam = zeros(ComplexF64, m)
    n = m
    while n > 1
        kount = 0
        while maximum(abs.(a[n, 1:n-1])) > tol && kount < kounttol
            kount = kount + 1
            mu = a[n, n]
            F = qr(a - mu * I)
            a = F.R * F.Q + mu * I
        end
        if kount < kounttol
            lam[n] = a[n, n]
            n = n - 1
            a = a[1:n, 1:n]
        else
            disc = (a[n-1, n-1] - a[n, n])^2 + 4 * a[n, n-1] * a[n-1, n]
            lam[n] = (a[n-1, n-1] + a[n, n] + sqrt(Complex(disc))) / 2
            lam[n-1] = (a[n-1, n-1] + a[n, n] - sqrt(Complex(disc))) / 2
            n = n - 2
            a = a[1:n, 1:n]
        end
    end
    if n > 0
       lam[1] = a[1, 1]
```

```
    end
    println("%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    println("The eigenvalues are:")
    display(round.(lam, digits = 4))
    println("%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
end

C = [ 4.0   3.0   1.0;
     -5.0  -3.0   0.0;
      3.0   2.0   1.0]
qr_my(C)
```

Julia Output

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Not success after 1000 times of pure QR, after use another method, we obtain:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
The eigenvalues are:
3-element Vector{ComplexF64}:
  2.0 + 0.0im
 -0.0 - 1.0im
 -0.0 + 1.0im
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Therefore, we obtain the eigenvalues of above matrix are $2, -i, i$.