矩阵特征值的计算-上机作业

张晨，2017011307

5.1

【实验内容】
用幂法计算矩阵按模最大特征值及对应的特征向量。
【算法】
采用课本算法5.1的实用幂法。
【程序输出】

```
eigenvalue:         12.254327
eigenvector:        0.674020 -1.000000 0.889560
Value of A*x:       8.259659 -12.254318 10.900937
Value of lambda*x:  8.259659 -12.254327 10.900955

eigenvalue:         98.521699
eigenvector:        -0.603972 1.000000 -0.251135 0.148953
Value of A*x:       -59.504381 98.521698 -24.742259 14.675146
Value of lambda*x:  -59.504381 98.521699 -24.742260 14.675146
```

两段输出分别表示两个矩阵的求解结果
【关键代码】

```
function [] = solve(A)
    n = length(A);
    x = ones(n, 1);
    l_old = NaN;
    k = 0;
    while (1)
        x = A*x;
        k = k+1;
        l_new = max(abs(x));
        x = x/l_new;
        if (abs(l_old - l_new) < 1e-5), break; end
        l_old = l_new;
    end
    fprintf("eigenvalue:        %f\n", l_old);
    fprintf("eigenvector:       %f %f %f %f", x); fprintf("\n");
    fprintf("Value of A*x:      %f %f %f %f", (A*x)'); ...
        fprintf("\n");
    fprintf("Value of lambda*x: %f %f %f %f", (l_old*x)); ...
        fprintf("\n\n");
end
```

5.3

【算法】

使用Householder变换进行QR分解，然后使用基本QR算法进行迭代。

【实验结果】

经过一轮迭代后，矩阵A没有发生变化，故基本QR算法不会“基本收敛”于拟上三角阵。

【结果分析】

将A矩阵进行QR分解后，得到对称的正交阵Q和对角线元素为±1的对称对角阵R，故$A_1 = RQ = R^TQ^T = (QR)^T = A^T = A$，因而没有发生变化。

使用matlab自带的eig函数，得到A的特征值为-1,1,1,1，对模长1有±1两种特征值，不符合定理5.22中的第二条要求，因此有可能不收敛。

【关键代码】

```
function lambda = basicQR(A)
    for i = [1:1]
        [Q,R] = QR(A);
        A = R*Q;
    end
    lambda = diag(A);
end

function s = mySign(x)
    s = sign(x);
    if s == 0
        s = 1;
    end
end

function [Q, R] = QR(A)
    n = length(A);
    R = A;
    Q = eye(n);
    for k = (1:n)
        sig = mySign(R(k, k)) * sqrt(sum(R(k:n, k).^2));
        v = R(k:n, k) + sig * [1; zeros(n-k, 1)];
        b = v' * v;
        w = [zeros(k-1, 1); v/sqrt(b)];
        Q = Q * (eye(n) - 2 * w * w');
        R(k:n, k:n) = R(k:n, k:n) - 2/b * v * (v' * R(k:n, k:n));
    end
end
```

【程序输出】

```
standard
    -1.0000
     1.0000
     1.0000
     1.0000

 Iter 1

 Q =
```

```
    -0.5000   -0.5000   -0.5000    0.5000
    -0.5000   -0.5000    0.5000   -0.5000
    -0.5000    0.5000   -0.5000   -0.5000
    -0.5000    0.5000    0.5000    0.5000


R =

   -1.0000         0         0         0
         0   -1.0000    0.0000    0.0000
         0         0   -1.0000         0
         0         0         0    1.0000


A =

    0.5000    0.5000    0.5000   -0.5000
    0.5000    0.5000   -0.5000    0.5000
    0.5000   -0.5000    0.5000    0.5000
   -0.5000    0.5000    0.5000    0.5000
```

5.4

【算法】

带原点位移的QR算法，采用面向实对称矩阵的单位移技术，位移因子取$s_k = A_k(n,n)$。

【实验结果】

在4次迭代后，迭代过程基本收敛，求得A的特征值为-1,1,1,1。这说明原点位移技术能使QR算法对更一般的矩阵收敛。

【关键代码】

```
function lambda = biasQR(A)
    n = length(A);
    for i = 1:4
        s = A(n,n);
        [Q,R] = QR(A-s*eye(n));
        A = R*Q + s*eye(n);
        fprintf("Iter %d\n", i);
        disp(A);
    end
    lambda = diag(A);
end
```

【程序输出】

```
  Iter 1
    -0.5000    0.6708   -0.4392    0.3273
     0.6708    0.7000    0.1964   -0.1464
    -0.4392    0.1964    0.8714    0.0958
     0.3273   -0.1464    0.0958    0.9286
```

```
Iter 2
   -0.9991   -0.0349    0.0202   -0.0143
   -0.0349    0.9994    0.0004   -0.0002
    0.0202    0.0004    0.9998    0.0001
   -0.0143   -0.0002    0.0001    0.9999

Iter 3
   -1.0000    0.0000   -0.0000    0.0000
    0.0000    1.0000    0.0000   -0.0000
   -0.0000    0.0000    1.0000    0.0000
    0.0000   -0.0000    0.0000    1.0000

Iter 4
   -1.0000   -0.0000    0.0000    0.0000
   -0.0000    1.0000   -0.0000    0.0000
    0.0000   -0.0000    1.0000    0.0000
   -0.0000   -0.0000    0.0000    1.0000

lambda
   -1.0000
    1.0000
    1.0000
    1.0000
```