数值分析实验报告

第五章

姓名	王琛
学号	2016011360
班级	计 65
实验日期	2019 年 6 月 9 日
报告日期	2019 年 6 月 9 日

第 5 章第 1 题

问题描述

用幂法求矩阵按模最大的特征值 λ_1 及其对应的特征向量 $\boldsymbol{x}_1$，使 $|(\lambda_1)_{k+1} - (\lambda_1)_k| < 10^{-5}$

解题思路

初始化一个任意向量，将矩阵乘以该向量，然后归一化，若两次得到的向量差值小于阈值则停止。

实验结果

计算所得的矩阵的特征向量和特征值如下：

```
>> main
Num of iterations=16
Matrix A
   -0.6740
    1.0000
   -0.8896

   12.2543

Num of iterations=6
Matrix B
   -0.6040
    1.0000
   -0.2511
    0.1490

   98.5217
```

和使用 matlab eig 命令算出的结果相同。

第 5 章第 3 题

问题描述

在 MATLAB 中实现基本的 QR 算法，观察矩阵序列收敛的情况，然后解释观察到的现象。

解题思路

首先使用 Householder 变换对矩阵实现正交三角化，实现 QR 分解。然后通过迭代，观察主对角线元素变化，最终获得特征值。

实验结果

无论如何迭代，每一步计算得到的 $A_{k+1} = RQ$ 都永远等于初始矩阵 A，永远无法收敛到特征值。因为矩阵 A 不满足所有的特征值绝对值不相同，因此不收敛。

第 5 章第 4 题

问题描述

采用带原点位移的 QR 算法计算 A 的特征值，观察迭代过程的收敛情况。

解题思路

每一步迭代将矩阵 A 减去 sI，QR 分解后，矩阵 A 加上 sI，如果 $a_{n,n-1}$ 趋近于 0，删除第 n 行和第 n 列，对子矩阵采用同样的方法。

实验结论

```
qr_offset_eig(A, 20)
 1.0000    0.0000         0         0
 0.0000    1.0000         0         0
      0         0   -0.9286    0.3712
      0         0    0.3712    0.9286

 1.0000    0.0000         0         0
 0.0000    1.0000         0         0
      0         0   -0.9999    0.0143
      0         0    0.0143    0.9999

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

 1.0000    0.0000         0
 0.0000    1.0000         0
      0         0   -1.0000

 1.0000    0.0000
 0.0000    1.0000
```

可以看出，求第一个特征值时迭代了 3 次，求之后的每个特征值都只需迭代一次。求得的特征值为 [1,-1,1,1]，结果正确。相比不用原点位移的方法，不仅收敛，而且速度很快。

实验总结

这次实验主要是针对特征值，包括幂法和 QR 算法求特征值。我认识到了一般 QR 算法的局限性以及原点位移 QR 算法的优越性所在。

主要代码

幂法

```
function [x, lam] = power_iteration(A)
%power_iteration
%Get the A's eigenvalue with maximum absolute value
n=size(A,1);
u=rand(n,1);
```

```
lam=0;
lam0=inf;
cnt=0;
while norm(lam0-lam, inf)>1e-5
    lam0=lam;
    v=A*u;
    lam=max(abs(v));
    u=v/lam;
    cnt=cnt+1;
end
fprintf("Num of iterations=%d\n", cnt);
x=u;
```

QR 算法

```
function [Q, R] = my_qr(A)
[m,n]=size(A);
V=zeros(m,n);
Q=eye(m);
for k=1:n
    sigma=0;
    for j=k:m
        sigma=sigma+A(j,k)*A(j,k);
    end
    sigma=sqrt(sigma);
    if(A(k,k) < 0)
        sigma=-sigma;
    end
    if sigma==A(k,k)
        continue
    end
    e=zeros(m,1);
    e(k,1)=1;
    for j=k:m
        V(j,k)=A(j,k);
```

```
    end
    V(:,k)=V(:,k)+sigma*e;
    beta=V(:,k)'*V(:,k);
    for j=k:n
        gamma=V(:,k)'*A(:,j);
        A(:,j)=A(:,j)-2*gamma/beta*V(:,k);
    end
    omega=V(:,k)/norm(V(:,k),2);
    H=eye(m)-2*(omega*omega');
    Q=Q*H;
%     disp(A);
end
R=A;
```

QR 算法求特征值

```
function[lam]=qr_eig(A, iter)
for i=1:iter
    [Q,R]=my_qr(A);
    A=R*Q;
    disp(A);
end
lam=diag(A);
```

原点位移 QR 算法求特征值

```
function l = qr_offset_eig(A,iter)
A=hess(A);
n=size(A,1);
l=zeros(n,1);
num=1;
for i=1:iter
    if(abs(A(n,n-1))<1e-3)
        l(num,1)=A(n,n);
        num=num+1;
        n=n-1;
```

```
        if n==1
            break;
        end
        A=A(1:n,1:n);
    end
    u=A(n,n);
    [q,r]=my_qr(A-u*eye(n,n));
    A=r*q+u*eye(n,n);
    disp(A);
end
l(num,1)=A(1,1);
```