**吉林大学数学实验中心实验报告**

**2021年 11月 16日**

|  |
| --- |
| **课程名称：科学计算方法实验实验 题目：乘幂法计算特征值**  **姓名：田泽禹 年级专业：2019**  **指导教师：王双**  **算法描述：使用乘幂法计算给定矩阵的特征值；在有最大模的特征值唯一时，使用规范乘幂法；在最大特征值共轭时应该使用多次拟合，最小二乘法解两特征值**  **相关内容如下**  **代码1 (MATLAB)：**  function [lam]=Power\_Method(A)  n=size(A,1);  x=ones(n,1);y=zeros(n,1);  for k=1:20  y=x/max(abs(x));  x=A\*y;  end  [~,index]=max(abs(x));  lam=x(index);  end  **运行结果1 (MATLAB)**：    但是当我们验证结果时，发现结果不正确  使用MatLab内置算法计算特征值    发现主特征值共轭，因此不能直接使用乘幂法  **代码2 (MATLAB)：**  function [lambda,eigx]=Power\_method\_H(A)  n=size(A,1);  z=ones(n,3);m=zeros(1,3);  for k=2:3  z(:,k)=A\*z(:,k-1);  m(k)=max(abs(z(:,k)));  z(:,k)=z(:,k)/m(k);  end    pq=[0,0;1,1];  pq(:,2)=pqk(z,m);  for k=1:200  z(:,1)=A\*z(:,3);  m(1)=max(abs(z(:,1)));  z(:,1)=z(:,1)/m(1);  z(:,[1,2,3])=z(:,[2,3,1]);  m([1,2,3])=m([2,3,1]);  pq(:,1)=pq(:,2);  pq(:,2)=pqk(z,m);  end  Relam=-pq(1,1)/2;  Imlam=1/2\*sqrt(4\*pq(2,1)-pq(1,1)^2);  lambda=Relam+1i\*Imlam;  eigx=z(:,1)+1i\*(Relam\*z(:,1)-m(2)\*z(:,2))/Imlam;  end    function [pq1]=pqk(z,m)  pq1=zeros(2,1);  delta=dot(z(:,2),z(:,2))\*dot(z(:,1),z(:,1))-dot(z(:,2),z(:,1))^2;  pq1(1)=-m(3)\*(dot(z(:,2),z(:,3))\*dot(z(:,1),z(:,1))-dot(z(:,1),z(:,2))\*dot(z(:,1),z(:,3)))/delta;  pq1(2)=-m(3)\*m(2)\*(dot(z(:,2),z(:,2))\*dot(z(:,1),z(:,3))-dot(z(:,1),z(:,2))\*dot(z(:,2),z(:,3)))/delta;  end  **运行结果2：**    **代码3 (C)：**本代码针对主特征值共轭的情况，矩阵阶数n任意  #include<stdio.h>  #include<stdlib.h>  #include<math.h>  double pqk1(double \*\*z,double m[3],int n);  double pqk2(double \*\*z,double m[3],int n);  void Mat\_multi(double \*\*Mat,double \*z,int n,double \*zk);  double fmaxabs(double \*z,int n);  void Division(double \*z,double m,int n);  void Multiplication(double \*z,double m,int n);  double dot(double \*z,double \*w,int n);  int main()  {  int n;  printf("please input matrix size n:\n");  scanf("%d",&n);  /\* input n \*/    double \*\*Mat;  Mat=(double \*\*)calloc(n,sizeof(double\*));  for(int i=0;i<n;i++)  {  Mat[i]=(double\*)calloc(n,sizeof(double));  }  for(int i=0;i<n;i++)  {  printf("Please input the %d row\n",i+1);  for(int j=0;j<n;j++)  {  scanf("%lf",&Mat[i][j]);  }  }    /\* initialize matrix A \*/    double \*z[3];  for(int i=0;i<3;i++)  {  z[i]=(double\*)calloc(n,sizeof(double));  }  for(int j=0;j<n;j++)  {  z[0][j]=1;  }  double m[3]={0};  /\* initialize z \*/    for(int k=1;k<3;k++)  {  Mat\_multi(Mat,z[k-1],n,z[k]);  m[k]=fmaxabs(z[k],n);  Division(z[k],m[k],n);  }    double pq[2][2]={{0,0},{1,1}};  pq[0][1]=pqk1(z,m,n);  pq[1][1]=pqk2(z,m,n);    for(int i=0;i<20;i++)  {  Mat\_multi(Mat,z[2],n,z[0]);  m[0]=fmaxabs(z[0],n);  Division(z[0],m[0],n);  {  double \*tempp;  tempp=z[0];  z[0]=z[1];  z[1]=z[2];  z[2]=tempp;  }  {  double temp;  temp=m[0];  m[0]=m[1];  m[1]=m[2];  m[2]=temp;  }  {  pq[0][0]=pq[0][1];  pq[1][0]=pq[1][1];  pq[0][1]=pqk1(z,m,n);  pq[1][1]=pqk2(z,m,n);  }  }    double Relam,Imlam;  Relam=-pq[0][0]/2.0;  Imlam=(sqrt(4\*pq[1][0]-pq[0][0]\*pq[0][0]))/2.0;  printf("Eigenvalue: %lf + i%lf\n",Relam,Imlam);    double \*Eigx[2];  Eigx[0]=z[0];  Eigx[1]=(double\*)calloc(n,sizeof(double));  for(int k=0;k<n;k++)  {  Eigx[1][k]=(Relam\*z[0][k]-m[1]\*z[1][k])/Imlam;  }  printf("The eigenvalue is:\n");  for(int k=0;k<n;k++)  {  printf("%lf + i\*%lf \n",Eigx[0][k],Eigx[1][k]);  }  }  void Mat\_multi(double \*\*Mat,double \*z,int n,double \*zk)  {  for(int i=0;i<n;i++)  {  double sum=0;  for(int j=0;j<n;j++)  {  sum+=Mat[i][j]\*z[j];  }  zk[i]=sum;  }  }  double fmaxabs(double \*z,int n)  {  double Max=0;  for(int i=0;i<n;i++)  {  double FABS=fabs(z[i]);  if(Max<FABS)  {  Max=FABS;  }  }  return Max;  }  double dot(double \*z,double \*w,int n)  {  double Sum=0;  for(int i=0;i<n;i++)  {  Sum+=z[i]\*w[i];  }  return Sum;  }  double pqk1(double \*\*z,double m[3],int n)  {  double delta=0;  double pq1;  delta=dot(z[1],z[1],n)\*dot(z[0],z[0],n)-dot(z[1],z[0],n)\*dot(z[1],z[0],n);  pq1=-m[2]\*(dot(z[1],z[2],n)\*dot(z[0],z[0],n)-dot(z[0],z[1],n)\*dot(z[0],z[2],n))/delta;  return pq1;  }  double pqk2(double \*\*z,double m[3],int n)  {  double delta=0;  double pq2;  delta=dot(z[1],z[1],n)\*dot(z[0],z[0],n)-dot(z[1],z[0],n)\*dot(z[1],z[0],n);  pq2=-m[2]\*m[1]\*(dot(z[1],z[1],n)\*dot(z[0],z[2],n)-dot(z[0],z[1],n)\*dot(z[1],z[2],n))/delta;  return pq2;  }  void Division(double \*z,double m,int n)  {  for(int i=0;i<n;i++)  {  z[i]=z[i]/m;  }  }  void Multiplication(double \*z,double m,int n)  {  for(int i=0;i<n;i++)  {  z[i]=z[i]\*m;  }  }  **运行结果3；**    **总结：**  使用乘幂法计算矩阵主特征值及对应特征向量时，应该对矩阵的特征值有一个预先估计；如本例中特征值共轭的情况，则不能直接使用乘幂法，进而要改用相应的特征值共轭算法。  当然此种方法具有很大的局限性与弊端，在不清楚矩阵特征值的分布的情况下，我们无法选择对应的算法来计算特征值及特征向量，因此多数情况下具有弊端。 |