Zadanie nr 5

Autor: Mateusz Laskowski

Obliczyc 2 najwieksze wartości własne i wektory własne dla macierzy:

{{19, 13, 10, 10, 13, -17},

{13, 13, 10, 10, -11, 13},

{10, 10, 10, -2, 10, 10},

{10, 10, -2, 10, 10, 10},

{13, -11, 10, 10, 13, 13},

{-17, 13, 10, 10, 13, 19}}

Pomnożonej przez 1/12

Wykorzystalem metode Jacobiego dla oliczenia wartości własnych – w jezyku java.

Wektory obliczyłem w programie Mathematica.

Kod:

class Matrix

{

public static final void eigenvalues(final double A[][],

double V[][], double Y[])

{

int n=A.length;

double AA[][] = new double[n][n];

double norm;

double c[] = new double[1];

double s[] = new double[1];

c[0] = 1.0;

s[0] = 0.0;

for(int i=0; i<n; i++)

{

for(int j=0; j<n; j++) V[i][j]=0.0;

V[i][i]=1.0;

}

copy(A, AA);

for(int k=0; k<n; k++)

{

norm=norm4(AA);

for(int i=0; i<n-1; i++)

{

for(int j=i+1; j<n; j++)

{

schur2(AA, i, j, c, s);

mat44(i, j, c, s, AA, V);

}

}

}

norm = norm4(AA);

for(int i=0; i<n; i++)

Y[i] = AA[i][i];

}

static double norm4(final double A[][])

{

int n=A.length;

int nr=A[0].length;

double nrm=0.0;

for(int i=0; i<n-1; i++)

{

for(int j=i+1; j<n; j++)

{

nrm=nrm+Math.abs(A[i][j])+Math.abs(A[j][i]);

}

}

return nrm/(n\*n-n);

}

public static final double norm2(final double A[][])

{

double r=0.0;

int n=A.length;

double B[][] = new double[n][n];

double V[][] = new double[n][n];

double BI[] = new double[n];

for(int i=0; i<n; i++) // B = A^T \* A

{

for(int j=0; j<n; j++)

{

B[i][j]=0.0;

for(int k=0; k<n; k++)

B[i][j] = B[i][j] + A[k][i]\*A[k][j];

}

}

eigenvalues(B, V, BI);

for(int i=0; i<n; i++) r=Math.max(r,BI[i]);

return Math.sqrt(r);

}

public static final void print(double A[][])

{

int N = A.length;

for(int i=0; i<N; i++)

for(int j=0; j<N; j++)

System.out.println("A["+i+"]["+j+"]="+A[i][j]);

}

static void mat44(final int p, final int q, final double c[], final double s[],

final double A[][], double V[][])

{

int n = A.length;

double B[][] = new double[n][n];

double J[][] = new double[n][n];

for(int i=0; i<n; i++)

{

for(int j=0; j<n; j++)

{

J[i][j]=0.0;

}

J[i][i]=1.0;

}

J[p][p]=c[0];

J[p][q]=-s[0];

J[q][q]=c[0];

J[q][p]=s[0];

multiply(J, A, B);

J[p][q]=s[0];

J[q][p]=-s[0];

multiply(B, J, A);

multiply(V, J, B);

copy(B, V);

}

static void schur2(final double A[][], final int p, final int q,

double c[], double s[])

{

double tau;

double t;

if(A[p][q]!=0.0)

{

tau=(A[q][q]-A[p][p])/(2.0\*A[p][q]);

if(tau>=0.0)

t=1.0/(tau+Math.sqrt(1.0+tau\*tau));

else

t=-1.0/((-tau)+Math.sqrt(1.0+tau\*tau));

c[0]=1.0/Math.sqrt(1.0+t\*t);

s[0]=t \* c[0];

}

else

{

c[0]=1.0;

s[0]=0.0;

}

}

public static final void multiply(final double A[][], final double B[][],

double C[][])

{

int ni = A.length;

int nk = A[0].length;

int nj = B[0].length;

for(int i=0; i<ni; i++)

for(int j=0; j<nj; j++)

{

C[i][j] = 0.0;

for(int k=0; k<nk; k++)

C[i][j] = C[i][j] + A[i][k] \* B[k][j];

}

}

public static final void copy(final double A[][], double B[][])

{

int ni = A.length;

int nj = A[0].length;

for(int i=0; i<ni; i++)

for(int j=0; j<nj; j++)

B[i][j] = A[i][j];

}

public static final void print(double X[])

{

int n = X.length;

for(int i=0; i<n; i++)

System.out.println("X["+i+"]="+X[i]);

}

}

class Start {

public static void main (String[] args){

double [][]A={{19, 13, 10, 10, 13, -17},

{13, 13, 10, 10, -11, 13},

{10, 10, 10, -2, 10, 10},

{10, 10, -2, 10, 10, 10},

{13, -11, 10, 10, 13, 13},

{-17, 13, 10, 10, 13, 19}};

double [][]B=new double[6][6];

double []C=new double[6];

Matrix.eigenvalues(A,B,C);

Matrix.print(C);

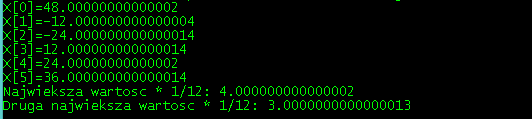
System.out.println("Najwieksza wartosc \* 1/12: "+ C[0]/12);

System.out.println("Druga najwieksza wartosc \* 1/12: "+ C[5]/12);

}

}

Wyniki:



Wektory:

x=3

v= []

x=4

v=[-