БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Лабораторная работа №5

**Методы решения проблемы собственных значений.**

**Степенной метод вычисления наибольшего по модулю собственного значения и соответствующего ему собственного вектора**

Выполнил:

Студент 2 курса 5 группы ФПМИ

Дунаев Виктор

Руководитель:

Радкевич Елена Владимировн

Минск, 2016 г.

**Оглавление**

Постановка задачи…………………………………………………………..………….……………..…..3

Нахождение наибольшего по модулю собственного значения …………….……..3

Нахождение собственного вектора матрицы…………………………………………..…….4

Листинг программы …………………………………………………………………………….………….5

Входные данные ………………………………………………………………………………….….….….8

Выходные данные ………………………………………….……………………………………………...8

**1.Постановка задачи**

Дана матрица , которая имеет следующий вид:

0.4997 -0.0658 0.0132 0.0263 0.0921

0.0684 0.7824 0.0000 -0.0526 0.0526

A = 0.0395 0.0000 0.6286 -0.1841 0.1052

-0.0789 0.1657 0.0000 0.6181 -0.0263

0.3288 0.0000 0.1184 0.0132 0.7364

1. Найти наибольшее по модулю собственное значение  матрицы  при помощи степенного метода;
2. Найти собственный вектор матрицы , соответствующий собственному значению ;
3. Найти количество итераций, необходимое для достижения заданной точности E=10^(-6);
4. Вывести на печать начальную матрицу A, наибольшее по модулю собственное значение  матрицы , собственный вектор матрицы , соответствующий собственному значению , количество итераций, необходимое для достижения заданной точности E=10^(-6).

**2.Нахождение наибольшего по модулю собственного значения**

Пусть матрица  и  является полной системой  линейно независимых векторов, то есть матрица  имеет только линейные элементарные делители. Этому требованию, например, удовлетворяют матрицы, у которых все собственные значения различны или .

Обозначим собственные значения  и соответствующие собственные векторы  Пусть для определённости . Будем строить итерационную последовательность векторов . Возьмём .

, , ..., .

Разложим  по собственным векторам:

.

Учитывая соотношение , получим

. (1)

Предположим  и . Тогда при больших  в правой части (1) первое слагаемое будет главным. Запишем (1) покоординатно.

, , , 

, , , .

Так как , , то при 

.

При достаточно больших 

. (2)

Если (2) выполняется для всех , то итерационный процесс сошёлся, и выбрано , достаточное для достижения требуемой точности. Другими словами, критерием остановки итерационного процесса является выполнение следующего неравенства:

, то есть .

**3.Нахождение собственного вектора матрицы**

При достаточно больших  в (1) все слагаемые справа, начиная со второго, будут иметь значения, меньшие принятой точности вычисления. Тогда ими можно пренебречь.

.

Так как собственный вектор определяется с точностью до постоянного множителя, можно взять .

Чтобы избежать чрезмерного роста (убывания) координат векторов , эти векторы можно нормировать. При этом вместо  получим другую систему векторов , где  - нормировочные коэффициенты, в качестве которых могут быть взяты . Тогда .

**4.Листинг программы**

**Main.cpp**

#include <iostream>

#include <fstream>

#include "cinoutFunctions.h"

#include "functions.h"

using namespace std;

int main ()

{

const int num = 5;

const double eps = 0.000001;

double \*\*A = new double\* [num],

\*y = new double [num],

eValue;

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

A[i] = new double [num];

ifstream isM ("matrix.txt");

ofstream os ("output.txt");

CinM (A, num, isM);

isM.close();

gaussTransformation (A, num);

os << "Initial matrix:" << endl << endl;

CoutM (A, num, os);

os << endl << findAbsMaxEigenvalue (A, y, eValue, num, eps) << " iterations needed." << endl << endl;

os << "Abs max eigenvalue is " << eValue << "." << endl << endl << "Eigenvector: " << endl << endl;

CoutV (y, num, os);

os.close();

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

delete [] A[i];

delete [] A;

delete [] y;

return 0;

}

**Functions.h**

#ifndef FUNCTIONS\_H

#define FUNCTIONS\_H

**//Преобразование матрицы**

template <typename T>

void gaussTransformation (T\*\* A, int num)

{

T\*\* At = new T\* [num];

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

At[i] = new T [num];

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

for (int j = i; j < num; ++j)

{

At[i][j] = 0;

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

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

}

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

for (int j = i; j < num; ++j)

A[i][j] = A[j][i] = At[i][j];

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

delete [] At[i];

delete [] At;

}

**//Нахождение максимального собственного значения**

template <typename T>

int findAbsMaxEigenvalue (T\*\* A, T\* y, T& eValue, int num, T eps)

{

int numIter = 0;

bool continueIter = true;

T norm;

T \*ytmp = new T [num],

\*ratio = new T [num];

y[0] = 1;

ratio[0] = 0;

for (int i = 1; i < num; ++i)

y[i] = ratio[i] = 0;

while (continueIter)

{

++numIter;

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

{

ytmp[i] = 0;

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

ytmp[i] += A[i][j] \* y[j];

}

norm = normVCube (ytmp, num);

continueIter = false;

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

{

eValue = ytmp[i] / y[i];

if (!continueIter && eValue - ratio[i] > eps)

continueIter = true;

ratio[i] = eValue;

y[i] = ytmp[i] / norm;

}

}

delete [] ratio;

delete [] ytmp;

return numIter;

}

**//Работа с вектором**

template <typename T>

T normVCube (T\* v, int num)

{

T norm = abs (v[0]);

for (int i = 1; i < num; ++i)

if (norm < abs (v[i]))

norm = abs (v[i]);

return norm;

}

#endif

**cinoutFunctions.h**

#ifndef CINOUTFUNCTIONS\_H

#define CINOUTFUNCTIONS\_H

#include <iostream>

using namespace std;

**//Вывод матрицы**

template <typename T>

void CoutM (T\*\* m, int num, ostream& os)

{

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

{

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

{

os.width (25);

os.precision (15);

os << m[i][j];

}

os << endl;

}

}

**//Чтение матрицы**

template <typename T>

void CinM (T\*\* m, int num, istream& is)

{

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

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

is >> m[i][j];

}

**//Вывод вектора**

template <typename T>

void CoutV (T\* v, int num, ostream& os)

{

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

{

os.precision (15);

os << v[i] << endl;

}

}

**//Вывод вектора**

template <typename T>

void CoutLV (T\* v, int num, ostream& os)

{

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

{

os.precision (15);

os << "a[" << i << "][" << i << "] (" << i << ") = " << v[i] << endl;

}

}

**//Вывод вектора**

template <typename T>

void CoutDV (T\* v, int num, ostream& os)

{

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

{

os.precision (15);

os << "dx[" << i << "] = " << v[i] << endl;

}

}

**//Чтение вектора**

template <typename T>

void CinV (T\* v, int num, istream& is)

{

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

is >> v[i];

}

#endif

**5.Входные данные**

0.4997 -0.0658 0.0132 0.0263 0.0921

0.0684 0.7824 0.0000 -0.0526 0.0526

A = 0.0395 0.0000 0.6286 -0.1841 0.1052

-0.0789 0.1657 0.0000 0.6181 -0.0263

0.3288 0.0000 0.1184 0.0132 0.7364

**6.Выходные данные**

Начальная матрица:

0.4997 -0.0658 0.0132 0.0263 0.0921

0.0684 0.7824 0.0000 -0.0526 0.0526

A = 0.0395 0.0000 0.6286 -0.1841 0.1052

-0.0789 0.1657 0.0000 0.6181 -0.0263

0.3288 0.0000 0.1184 0.0132 0.7364

Максимальное собственное значение U(max)= 0.4022.

Собственный вектор, соответствующий собственному значению U(max):

-0.842592117666229

-0.0113199683552523

-0.462176212614022

-0.177495527280946

1

Количество итераций .