МІНІСТЕРСТВО ОСВІТИ І НАУКИ
НАВЧАЛЬНО-НАУКОВИЙ КОМПЛЕКС
«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»
НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

Лабораторна робота №1

з курсу «Чисельні методи»

Тема: Власні числа та вектори

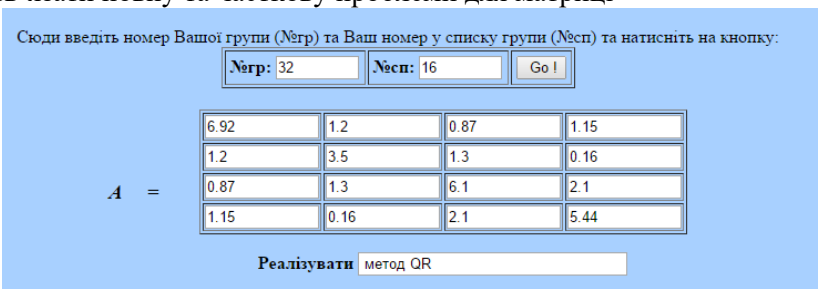
Варіант 16

Виконав: студент 3 курсу
групи КА-32
Пустовіт Д.Т.
Прийняв:
Коновалюк М.М.

Київ – 2015

Завдання:

Розв’язати повну та часткову проблеми для матриці



Математичний розв’язок задачі:

Степеневий метод застосовують для пошуку максимального за модулем власного числа. Для цього будується ітераційний процес $x^{s+1} = Ax^s$, де вектор x довільний не нульовий. При цьому вектор x збігається до власного вектора за напрямом. Критерієм закінчення є різниця відношення координат векторів x^s та x^{s+1}. при цьому власне число обчислюється за формулою:

$$|\lambda_1| \approx \frac{|x^{(s+1)}|}{|x^{(s)}|} = \sqrt{\frac{(x^{(s+1)}, x^{(s+1)})}{(x^{(s)}, x^{(s)})}}.$$

Алгоритм QR гуртується на тому факті, що для будь-якої квадратної матриці A існує розклад A=QR, де Q-ортогональна, а R верхня трикутна. Для розкладу використовують матрицю Хаусхолдера $H = E - \frac{2}{v^T v} v v^T$, де E – одинична, а v обирається з умови:

$$\begin{cases} v_i^k = 0, \text{ для } i = \overline{1, k-1} \\ v_2^k = a_{kk}^{k-1} + sign(a_{kk}^{k-1})\sqrt{\sum_{j=k}^{n}(a_{j2}^{k-1})^2} \\ v_i^k = a_{ik-1}^{k-1}, \text{ для } i = \overline{k+1, n} \end{cases}$$

, Далі отримуємо QR розклад з того, що:

$$Q = (H_{n-1}H_{n-2}...H_0)^T = H_1H_2...H_{n-1},\ R = A_{n-1}.$$

$$A^{(k)} = Q^{(k)}R^{(k)} \text{ -}$$

Тепер будуємо ітераційний процес: $A^{(k+1)} = R^{(k)}Q^{(k)}$ якщо у матриці відсутні кратні власні числа, то вона збігається до трикутної матриці. Діагональними елементами тепер є власні числа, а для отримання власних векторів необхідно перемножити ортогональні матриці $Q^{(k)}$.

Лістинг:

```
#include "stdafx.h"
#include <fstream>
#include <iostream>

#include <math.h>

#define n 4
#define eps 0.00001
#define zn 3

using namespace std;

typedef double Mat[n][n];
typedef double vec[n];

void fileMat(ofstream &);
void inMat(string, Mat*);
void outMat(char*,char*,Mat*);
void outVec(char*, char*, vec*, bool);
void QR(Mat&, Mat&);
void Matprod(Mat, Mat, Mat &);
void Hausholder(Mat &, vec);
```

```
void findEigen(Mat&, vec&, Mat&);
void correct(Mat, vec, Mat);
double step(Mat);

int _tmain(int argc, _TCHAR* argv[])
{
	Mat A,B;
	vec b;
	inMat("inMat.txt", &A);
	ofstream fout("ans.txt");
	fout << "Розвязок часткової проблеми (Max eigen): " << step(A) << endl;
	findEigen(A, b, B);
	outVec("ans.txt", "власні числа:", &b, false);
	outMat("ans.txt", "власні вектори:", &B);
	inMat("inMat.txt", &A);
	correct(A,b,B);
	system("Pause");
	return 0;
}



double step(Mat A){
	vec x, x1;
	for (int i = 0; i < n; i++) x[i] = 1;
	bool flag;
	double t;
	do
	{
		for (int i = 0; i < n; i++) x1[i] = x[i];
		for (int i = 0; i < n; i++){
			x[i] = 0;
			for (int j = 0; j < n; j++)
				x[i] += A[i][j] * x1[j];
		}
		t = x[0] / x1[0];
		flag = true;
		for (int i = 1; i < n; i++)
			if ( abs(x[i]/x1[i]-t)>eps ){
				flag = false;
				break;
			}
	} while (!flag);
	double s1=0, s2=0;
	for (int i = 0; i < n; i++){
		s1 += x[i] * x[i];
		s2 += x1[i] * x1[i];
	}
	return pow(s1 / s2, 0.5);
}
void Matprod(Mat A, Mat B, Mat &AB){
	for (int i = 0; i < n; i++)
		for (int j = 0; j < n; j++){
			AB[i][j] = 0;
			for (int k = 0; k < n; k++) AB[i][j] += A[i][k] * B[k][j];
		}
}
void Hausholder(Mat &P, vec v){
	double s=0;
	for (int i = 0; i < n; i++) s += v[i] * v[i];
	for (int i = 0; i < n; i++)
		for (int j = 0; j < n; j++)
			P[i][j] = -2 * v[i] * v[j] / s;
	for (int i = 0; i < n; i++) P[i][i] += 1;
}
void QR(Mat& A, Mat &Q){
	double s, t;
	vec v;
	Mat H,R,I;
	for (int i = 0; i < n; i++)
		for (int j = 0; j < n; j++)
			Q[i][j] = 0;
	for (int i = 0; i < n; i++) Q[i][i] = 1;
	for (int k = 0; k < n - 1; k++){
		t = 0;
		for (int i = 0; i < n; i++)
			for (int j = 0; j < n; j++){
				I[i][j] = Q[i][j];
				R[i][j] = A[i][j];
```

```
                        }
                        for (int j = k; j < n; j++) t += A[j][k]*A[j][k];
                        s = sqrtf(t);
                        if (A[k][k] < 0) s = -s;
                        for (int j = 0; j < k; j++) v[j] = 0;
                        v[k] = A[k][k] + s;
                        for (int j = k + 1; j < n; j++) v[j] = A[j][k];
                        Hausholder(H, v);
                        Matprod(I, H, Q);
                        Matprod(H, R, A);
                }
}
void findEigen(Mat& A, vec& b, Mat& B){
                Mat Q, P ,R;
                int p=0;
                double s;
                char* str = new char[20];
                for (int i = 0; i < n; i++){
                        for (int j = 0; j < n; j++)
                                if(i!=j) P[i][j] = 0;
                        P[i][i] = 1;
                }
                while (true){
                        p++;
                        for (int i = 0; i < n; i++)
                                for (int j = 0; j < n; j++)
                                        R[i][j] = A[i][j];
                        QR(R, Q);
                        Matprod(P,Q,B);
                        Matprod(R, Q, A);
                        //виведення проміжних данних
                        _itoa(p, str, 10);
                        strcat(str," iteration :");
                        outMat("debug.txt",str, &A);
                        //перевірка виходу
                        s = 0;
                        for (int k = 0; k < n - 1; k++)
                                for (int i = k + 1; i < n; i++)
                                        s += A[i][k] * A[i][k];
                        if (s < eps*eps) break;
                        for (int i = 0; i < n; i++)
                                for (int j = 0; j < n; j++)
                                        P[i][j] = B[i][j];
                }
                for (int i = 0; i < n; i++) b[i] = A[i][i];
}
void correct(Mat A, vec b, Mat B){
                vec x;
                char * c = new char[10];
                for (int i = 0; i < n; i++){
                        _itoa(i + 1, c, 10);
                        strcat(c, "-a Невязка:\n");
                        for (int j = 0; j < n; j++){
                                x[j] = 0;
                                for (int k = 0; k < n; k++)
                                        x[j] += A[j][k] * B[k][i];
                                x[j] = x[j] - b[i] * B[j][i];
                        }
                        outVec("ans.txt", c, &x, true);
                }
}
void fileMat(ofstream &fout){
                fout.width(zn + 10);
                fout.precision(zn);
                fout.setf(ios::right);
                fout.setf(ios::fixed);
}
void inMat(string s, Mat* A){
                ifstream stm(s);
                int i, j;
                for (i = 0; i < n; i++)
                        for (j = 0; j < n; j++)
                                stm >> (*A)[i][j];
                stm.close();
};
void outMat(char* s,char* msg, Mat* A){
                ofstream stm(s,ios_base::app);
                fileMat(stm);
                stm << msg << "\n";
```

```
		for (int i = 0; i < n; i++){
			for (int j = 0; j < n; j++)
				stm << (*A)[i][j] << "\t";
			stm << endl;
		}
		stm << "----------------------------\n";
};
void outVec(char* s, char* msg, vec* x, bool b){
		ofstream stm(s,ios_base::app);
		stm << msg << "\n";
		if (b) stm.setf(ios::scientific);
		int i;
		for (i = 0; i < n; i++)
			stm << (*x)[i] << endl;
		stm << endl;
}
```

Результати роботи

(файл ans.txt)

```
Розвязок часткової проблеми (Max eigen): 9.33224
власні числа:
9.33224
5.96585
4.15479
2.50712

власні вектори:
0.581	-0.779	0.079	-0.222
0.264	-0.084	-0.547	0.790
0.587	0.519	-0.436	-0.443
0.499	0.342	0.710	0.361
----------------------------
1-а Невязка:

-3.559011e-007
-3.820398e-008
2.368629e-007
1.559635e-007

2-а Невязка:

9.495280e-007
-4.591544e-006
-3.489696e-006
6.341996e-006

3-а Невязка:

-6.721007e-006
-6.880687e-007
4.448824e-006
2.956094e-006

4-а Невязка:

3.258773e-009
-2.243868e-008
-1.789355e-008
2.911521e-008
```

(файл debug.txt)

```
1 iteration :
7.865	0.973	1.242	-0.677
0.973	4.051	1.594	0.184
1.242	1.594	6.129	-1.419
-0.677	0.184	-1.419	3.915
----------------------------
2 iteration :
8.548	0.976	1.007	0.279
0.976	4.876	1.513	-0.260
1.007	1.513	5.378	0.918
0.279	-0.260	0.918	3.158
----------------------------
*
*
*
10 iteration :
9.331	0.051	0.003	0.000
0.051	5.961	0.105	-0.001
0.003	0.105	4.161	0.021
0.000	-0.001	0.021	2.507
----------------------------
11 iteration :
9.332	0.033	0.001	-0.000
0.033	5.963	0.073	0.000
0.001	0.073	4.158	-0.013
-0.000	0.000	-0.013	2.507
*
*
*
35 iteration :
9.332	0.000	0.000	-0.000
0.000	5.966	0.000	0.000
0.000	0.000	4.155	-0.000
-0.000	0.000	-0.000	2.507
----------------------------
36 iteration :
9.332	0.000	0.000	0.000
0.000	5.966	0.000	-0.000
0.000	0.000	4.155	0.000
0.000	-0.000	0.000	2.507
----------------------------
```

Висновок:

Працюючи над лабораторною роботою, я навчився розв’язувати часткову проблему власних чисел степеневим методом, а також повну проблему QR методом.