МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «Програмна інженерія та інформаційні технології управління»

Звіт з індивідуального розрахункового завдання №9

З предмету «Числові методи»

Виконав

Студент групи КН-36а

Рубан Ю.Д.

Перевірив:

Гужва В.О.

Харків - 2017

Завдання: знайти власні значення та власні вектори матриці методом Левер'є.

Процес **знаходження власних значень за методом Левер'є** ділиться на два етапи: розкриття характеристичного многочлена та знаходження його коренів. Розглянемо дані етапи більш детально. Для цього, розглянемо матрицю metod_laverre2, для якої запишемо **характеристичний многочлен** у наступному вигляді:

metod_laverre14

де Метод Леверр'є корені даного многочлена. Розкладемо многочлен (1) на лінійні множники. В результаті отримаємо:

metod_laverre15

Перемноживши вирази, які містяться в правій частині (2) та звівши подібні члени, після чого прирівнявши їх з відповідними коефіцієнтами з (1), отримаємо формули, які виражають коефіцієнти **характеристичного мнгочлена** через його корені:

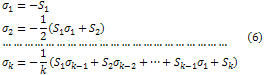
metod_laverre31

де metod_laverre17 - елементарні симетричні функції **коренів характеристичного многочлена**.

Розглянемо також наступні симетричні функції коренів metod_laverre19. З курсу вищої алгебри відомо, що metod_laverre20 і metod_laverre21 пов'язані наступним співвідношенням:

metod_laverre22

З даного співвідношення знаходимо формули для **обчислення коефіцієнтів многочлена** (1):



де metod_laverre6 — слід матриці metod_laverre7; metod_laverre9 — слід матриці metod_laverre10;...; metod_laverre24 — слід матриці metod_laverre25.

Таким чином, схема розкриття вікового визначника за **методом Левер'є** вельми проста, а саме: спочатку обчислюються metod_laverre27 — степені матриці metod_laverre28, після чого знаходяться відповідні metod_laverre24 — суми елементів головних діагоналей матриць metod_laverre291 і, нарешті, за формулами (6) визначаються шукані коефіцієнти metod_laverre30.

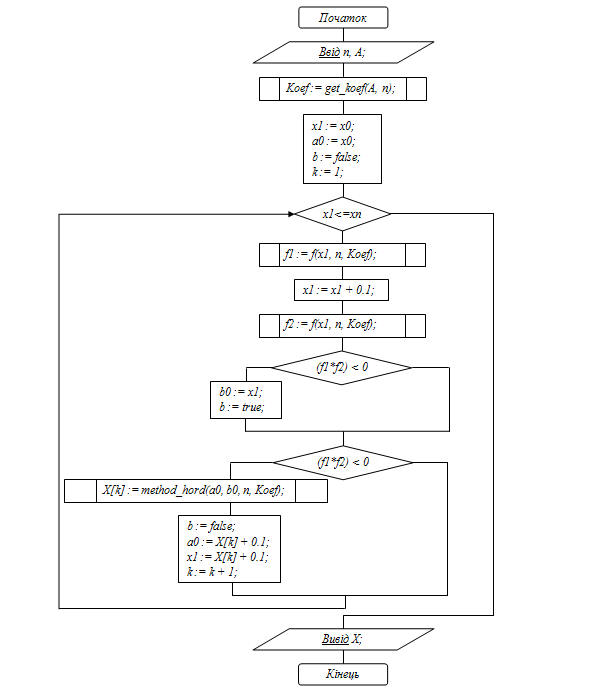


Рисунок 1 – Блок схема алгоритму пошуку власних значень і векторів методом Левер’є

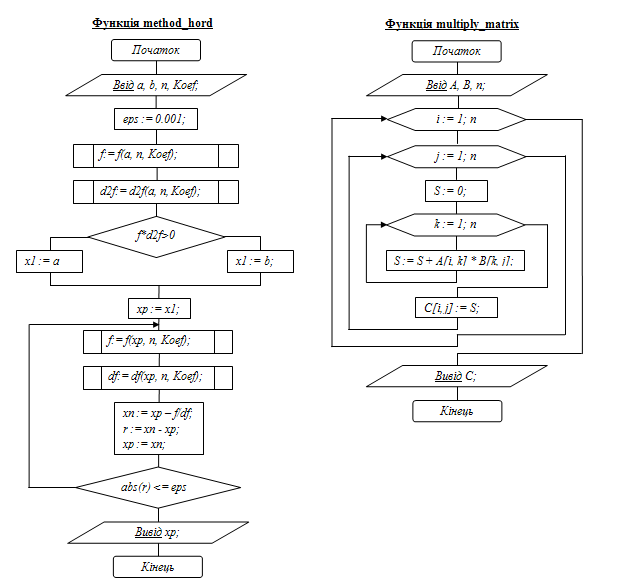


Рисунок 2 – Блок схеми алгоритмів хорд та перемноження матриць

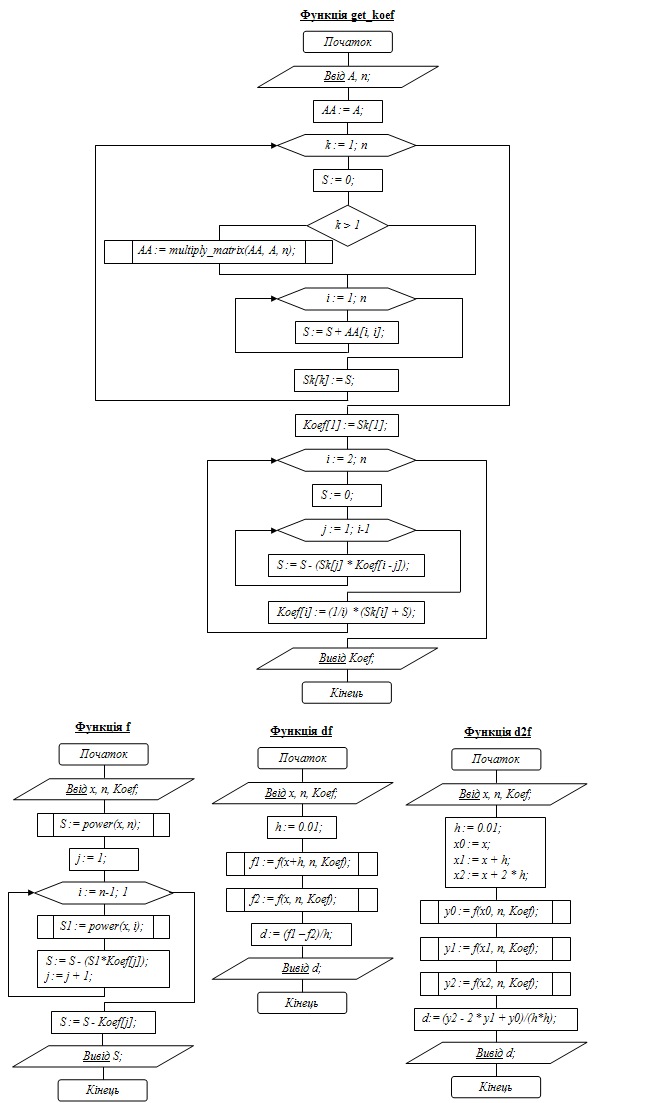


Рисунок 3 – Блок схеми додаткових функцій

Ручне рішення

2 3 2

4 -6 -4

-1 4 7

Начальный вектор

1

0

0

A^1 =

2 3 2

4 -6 -4

-1 4 7

SP1 = 3

A^2 =

14 -4 6

-12 32 4

7 1 31

SP2 = 77

A^3 =

6 90 86

100 -212 -124

-13 139 227

SP3 = 21

Характерестический многочлен

(104\*x^0)+(-34\*x^1)+(-3\*x^2)+(1\*x^3)

Собственные значения

x1 = -5.8503

x2 = 3.08162

x3 = 5.76868

Коэфициенты Горнера

1 1 1

-8.8503 0.0816206 2.76868

17.7769 -33.7485 -18.0284

y0

2

4

-1

y1

14

-12

7

y2

6

100

-13

Собственные вектора

14.0763 -19.5852 1.50896

-47.4012 -11.6735 -0.925292

15.8503 6.91838 4.23132

Фрагмент коду програми

#include"Leverye\_alg.h"

Result Leverye\_alg::do\_algorithm(vector<vector<double>>matrix, int size, vector<vector<double>>\*arg)

{

vector<double>S(size);

S[0] = Algorithms::SP(matrix);

cout << "A^1 = ";

show(matrix);

cout << "SP" << 1 << " = " << S[0] << endl << endl;

vector <vector<double>>temp = matrix;

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

{

temp = Algorithms::matrix\_multi(temp, matrix);

cout << "A^" << i+1<< " = ";

show(temp);

S[i] = Algorithms::SP(temp);

cout << "SP" << i+1 << " = " << S[i] << endl << endl;

}

vector<double>p(size);

p[0] = -S[0];

double sum = 0;

for (double i = 1; i < size; i++)

{

sum = S[i];

for (int j = 0, k = i - 1; j < i; j++, k--)

{

sum += p[j] \* S[k];

}

p[i] = -sum / (i + 1);

}

vector<double>rev = p;

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

{

rev[i] = p[size - 1 - i];

p[size - 1 - i]\*=-1;

}

rev.push\_back(1);

Polynomal P(rev);

cout << "Характерестический многочлен" << endl;

cout << P << endl;

vector<double>lambdas = \*Half\_div\_alg::solve(P.get\_function());

cout << "Собственные значения" << endl;

show\_row(lambdas);

Result res;

auto q = gorner\_coefs(lambdas, p);

show(q, "Коэфициенты Горнера");

vector<vector<double>>X(size);

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

{

X[i].resize(size);

}

vector<vector<double>>Y = \*arg;

vector<vector<double>>\*ys = new vector<vector<double>>[size];

ys[0] = matrix\_multi(matrix, Y);

Algorithms::show(ys[0], "y0");

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

{

cout << "y" << i;

ys[i] = matrix\_multi(matrix, ys[i - 1]);

Algorithms::show(ys[i]);

}

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

{

for (int i = size - 1, b = 0; i >= 0; i--, b++)

{

for (int j = 0, l = size - 1; j < size - 1; j++, l--)

{

X[k][b] += ys[l - 1][b][0] \* q[j][k];

}

}

}

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

{

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

{

X[i][j] += q[size - 1][i] \* Y[j][0];

}

}

transpose(X);

res = X;

return res;

}

vector<vector<double>>Leverye\_alg::gorner\_coefs(vector<double>lambdas, vector <double>p)

{

int n = lambdas.size();

vector<vector<double>>q(n);

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

{

q[i].resize(n);

}

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

{

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

{

if (j == 0) { q[j][i] = 1; }

else

{

q[j][i] = (lambdas[i] \* q[j - 1][i]) - p[j - 1];

}

}

}

return q;

}

Результати виконання програми

n=3

Матриця =

2 3 2

4 -6 -4

-1 4 7

Початковий вектор =

1

0

0

Власні значення =

x1 = -5.8503

x2 = 3.08162

x3 = 5.76868

Власні вектори =

14.0763 -19.5852 1.50896

-47.4012 -11.6735 -0.925292

15.8503 6.91838 4.23132

Висновок

Результати програми співпадають з результатами ручного рішення

***Список використаних джерел***

**1. Островский А.М.** [*Решение уравнений и систем уравнений*](http://pmpu.ru/vf4/references#островский). М. ИЛ, 1963, c. 137-142

2. **Уилкинсон Дж.Х.** Алгебраическая проблема собственных значений. М.Наука. 1970, с.93-94

3. **Фаддеев Д.К., Фаддеева В.Н.** Вычислительные методы линейной алгебры. М.ГИФМЛ. 1960

4. **Хорн Р.**, **Джонсон Ч.** Матричный анализ. М.Мир.1989