МІНІСТЕРСВО ОСВІТИ І НАУКИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет комп'ютерних наук та кібернетики

ЛАБОРАТОРНА РОБОТА №2

«Розв’язання систем лінійних алгебраїчних рівнянь»

Варіант 2,5, 4x4

Виконала:

студентка 2 курсу, групи К-24

спеціальності «Комп’ютерні науки. Інформатика»

Баклан Аліса

<u>Завдання:</u>

Написати програму, яка розв’язує систему лінійних алгебраїчних рівнянь двома методами:

- методом квадратних коренів
- методом Зейделя

Знайти визначник матриці своїм прямим методом. Знайти число обумовленості.

Задача

Я обрала таку матрицю A і вектов вільних членів b:

A = 4 1 0 2 b= 14

1 3 0 0 7

0 0 2 1 10

2 0 1 4 21

Для моїх методів важливо щоб матриця була симетричною відносно головної діагоналі а також щоб числа на головній діагоналі були більше ніж модуль суми інших чисел у рядку.

Метод квадратних коренів

Використана теорія:

Метод квадратного кореня Використовується, якщо матриця A — симетрична, т.т. $A = A^T$. Матрицю A подамо у вигляді $A = S^T DS$, де

$$S = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1n} \\ 0 & s_{22} & \dots & s_{2n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & s_{nn} \end{pmatrix}; \quad D = \begin{pmatrix} \pm1 & 0 & \dots & 0 \\ 0 & \pm1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \pm1 \end{pmatrix};$$

$$d_{ii} = \operatorname{sgn} \left| a_{ii} - \sum_{p=1}^{i-1} s_{pi}^2 d_{pp} \right|; \; s_{ii} = \sqrt{\left| a_{ii} - \sum_{p=1}^{i-1} s_{pi}^2 d_{pp} \right|}, \; i = \overline{1, n};$$

$$s_{ij} = \frac{a_{ij} - \sum\limits_{p=1}^{i-1} s_{pi} d_{pp} s_{pj}}{d_{ii} s_{ii}}, \qquad i = \overline{2, n-1}, \qquad j = \overline{i+1, n}.$$

Подальше рішення зводиться до розв'язання двох систем лінійних алгебраїчних рівнянь з трикутними матрицями, з першої системи знаходять y:

$$S^T D y = b,$$

а з другої – x:

$$Sx = y.$$

Складність методу квадратного кореня: $Q(n) = \frac{1}{3}n^3 + O(n^2)$.

Зауваження. Методом квадратних коренів можна знайти визначник:

$$detA = \prod_{k=1}^{n} d_{kk} s_{kk}^2.$$

Розв'язання задачі:

$$A = \begin{pmatrix} 4 & 1 & 0 & 2 \\ 1 & 3 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 2 & 0 & 1 & 4 \end{pmatrix} \qquad b = \begin{pmatrix} 14 \\ \cancel{10}7 \\ 10 \\ 21 \end{pmatrix}$$

$$A = A^T$$

Знайдемо S та D:

$$d_{11} = sgn(a_{11}) = 1$$

$$s_{11} = \sqrt{|a_{11}|} = 2$$

$$s_{12} = \frac{a_{12}}{d_{11} s_{11}} = \frac{1}{1 \cdot 2} = 0.5$$

$$s_{13} = \frac{a_{13}}{d_{11} s_{11}} = \frac{0}{1 \cdot 2} = 0$$

$$s_{14} = \frac{a_{14}}{d_{11} s_{11}} = \frac{2}{1 \cdot 2} = 1$$

$$d_{22} = \operatorname{sgn}(a_{22} - s_{12}^2 d_{11}) = \operatorname{sgn}(3 - \tfrac{1}{4} \cdot 1) = 1$$

$$s_{22} = \sqrt{3 - \tfrac{1}{4}} = \frac{\sqrt{11}}{2} \approx 1.65831$$

$$s_{23} = \frac{a_{23} - s_{12} d_{11} s_{13}}{d_{22} s_{22}} = \frac{0 - \frac{1}{2} \cdot 1 \cdot 0}{+1\sqrt{11}/2} = 0$$

$$s_{24} = \frac{a_{24} - s_{12} d_{11} s_{14}}{d_{22} s_{22}} = \frac{0 - 1/2 \cdot 1 \cdot 1}{+1 \cdot \sqrt{11}/2} = -\frac{1}{\sqrt{11}} \approx 0,3015$$

$$d_{33} = \operatorname{sgn}(a_{33} - s_{13}^2 d_{11} - s_{23}^2 d_{22}) = \operatorname{sgn}(2 - 0 - 0) = 1$$

$$s_{33} = \sqrt{2} \approx 1,4142$$

$$s_{34} = \frac{a_{34} - s_{14} d_{11} s_{13} - s_{24} d_{22} s_{23}}{d_{33} s_{33}} = \frac{1 - 0 - 0}{1 \cdot \sqrt{2}} = \frac{1}{\sqrt{2}} \approx 0,7071$$

$$d_{44} = \operatorname{sgn}(a_{44} - s_{14}^2 d_{11} - s_{24}^2 d_{22} - s_{34}^2 d_{33}) = \operatorname{sgn}\left(4 - 1^2 \cdot 1 - \frac{1}{11} - \frac{1}{2}\right) = 1$$

$$s_{44} = \sqrt{4 - 1 - \frac{1}{11} - \frac{1}{2}} = \sqrt{\frac{53}{22}} \approx 1,5521$$

Маємо:

$$S = \begin{pmatrix} 2 & 0.5 & 0 & 1 \\ 0 & \sqrt{11}/2 & 0 & -1/\sqrt{11} \\ 0 & 0 & \sqrt{2} & 1/\sqrt{2} \\ 0 & 0 & 0 & \sqrt{53/22} \end{pmatrix} \qquad D = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$S^T D y = b$$

$$\begin{pmatrix} 2 & 0 & 0 & 0 \\ 0.5 & \sqrt{11}/2 & 0 & 0 \\ 0 & 0 & \sqrt{2} & 0 \\ 1 & -1/\sqrt{11} & 1/\sqrt{2} & \sqrt{53/22} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} = \begin{pmatrix} 14 \\ 7 \\ 10 \\ 21 \end{pmatrix}$$

$$y_1 = 7$$

$$y_2 = (7 - 3,5)\, 2/\sqrt{11} = 7/\sqrt{11} \approx 2.1105$$

$$y_3 = \frac{10}{\sqrt{2}} = 5\sqrt{2} \approx 7,071067$$

$$y_4 = \left(21 - 7 + 7/11 - 5\right) \frac{\sqrt{22}}{\sqrt{53}} = \frac{2\sqrt{1166}}{\sqrt{11}} = \frac{2\sqrt{106}}{\sqrt{11}} \approx 6,2085$$

$$Sx = y$$

$$\begin{pmatrix} 2 & 0.5 & 0 & 1 \\ 0 & \sqrt{11}/2 & 0 & -1/\sqrt{11} \\ 0 & 0 & \sqrt{2} & 1/\sqrt{2} \\ 0 & 0 & 0 & \sqrt{53}/\sqrt{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 7 \\ 7/\sqrt{11} \\ 5\sqrt{2} \\ \frac{2\sqrt{106}}{\sqrt{11}} \end{pmatrix}$$

$$x_4 = \frac{2\sqrt{106}}{\sqrt{11}} \cdot \frac{\sqrt{22}}{\sqrt{53}} = 2 \cdot 2 = 4$$

$$x_3 = (5\sqrt{2} - 4/\sqrt{2})/\sqrt{2} = 3$$

$$x_2 = 2(7/\sqrt{11} + 4/\sqrt{11})/\sqrt{11} = 2$$

$$x_1 = (7 - 4 - 0.5 \cdot 2)/2 = 1$$

$$\det(A) = 4 \cdot \frac{11}{4} \cdot 2 \cdot \frac{53}{22} = 53$$

В-дь: $(1; 2; 3; 4)$, $\det A = 53$

Отримали розв’язок (1, 2, 3,4) та визначник 53.

Реалізація в програмі:

Ініціалізуємо змінні і зчитуємо дані:

```
setlocale(LC_CTYPE, "ukr");
const int n = 4;
float A[n][n]; // матриця
float S[n][n];
float D[n][n];
float b[n], x[n], y[n];

//вводимо матрицю
cout << "введіть елементи симетричної  матриці" << endl;
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
    {
        cin >> A[i][j];
    }
```

Перевірка матриці:

```
//перевірка на симетричність
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
    {
        if (A[i][j] != A[j][i])
        {
            cout << "Матриця не симетрична" << endl;
        }
    }
```

Заповнюємо матриці S,D:

```
//заповнюємо D,S
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
    {
        S[i][j] = 0;
        D[i][j] = 0;
    }

D[0][0] = Sign(A[0][0]);
S[0][0] = sqrt(A[0][0]);

S[0][1] = A[0][1] / (D[0][0] * S[0][0]);
S[0][2] = A[0][2] / (D[0][0] * S[0][0]);
S[0][3] = A[0][3] / (D[0][0] * S[0][0]);

D[1][1] = Sign(A[1][1] - S[0][1] * S[0][1] * D[0][0]);
S[1][1] = sqrt(abs(A[1][1] - S[0][1] * S[0][1] * D[0][0]));
S[1][2] = (A[1][2] - S[0][1] * S[0][2] * D[0][0]) / (D[1][1] * S[1][1]);
S[1][3] = (A[1][3] - S[0][1] * S[0][3] * D[0][0]) / (D[1][1] * S[1][1]);

D[2][2] = Sign(A[2][2] - S[0][2] * S[0][2] * D[0][0] - S[1][2] * S[1][2] * D[1][1]);
S[2][2] = sqrt(abs(A[2][2] - S[0][2] * S[0][2] * D[0][0] - S[1][2] * S[1][2] * D[1][1]));
S[2][3] = (A[2][3] - S[0][2] * S[0][3] * D[0][0] - S[1][2] * S[1][3] * D[1][1]) / (D[2][2] * S[2][2]);

D[3][3] = Sign(A[3][3] - S[0][3] * S[0][3] * D[0][0] - S[1][3] * S[1][3] * D[1][1] - S[2][3] * S[2][3] * D[2][2]);
S[3][3] = sqrt(abs(A[3][3] - S[0][3] * S[0][3] * D[0][0] - S[1][3] * S[1][3] * D[1][1] - S[2][3] * S[2][3] * D[2][2]));
```

Знаходимо добуток матриць $S^T D$ та вирішуємо систему $S^T D\ y = b$:

```
//
float ST[4][4];
float STD[4][4];
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
    {
        ST[i][j] = S[j][i];
    }
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
    {
        if (D[j][j] == 1)
            STD[i][j] = ST[i][j];
        else STD[i][j] = (-1) * ST[i][j];
    }
//STDy=b
y[0] = b[0] / STD[0][0];
y[1] = (b[1] - y[0] * STD[1][0]) / STD[1][1];
y[2] = (b[2] - y[0] * STD[2][0] - y[1] * STD[2][1]) / STD[2][2];
y[3] = (b[3] - y[0] * STD[3][0] - y[1] * STD[3][1] - y[2] * STD[3][2]) / STD[3][3];
```

Вирішуємо Sx=y:

```
//Sx=y
x[3] = y[3] / S[3][3];
x[2] = (y[2] - S[2][3] * x[3]) / S[2][2];
x[1] = (y[1] - S[1][2] * x[2] - S[1][3] * x[3]) / S[1][1];
x[0] = (y[0] - S[0][1] * x[1] - S[0][2] * x[2] - S[0][3] * x[3]) / S[0][0];

cout << "x[1] = " << x[0] << endl;
cout << "x[2] = " << x[1] << endl;
cout << "x[3] = " << x[2] << endl;
cout << "x[4] = " << x[3] << endl;
```

Рахуємо визначник:

```
//визначник
cout << "Визначник" << endl;
float v = 1;
for (int i = 0; i<4; i++)
{
    v = v * S[i][i] * S[i][i] * D[i][i];
}
cout << v << endl;
```

Результат роботи програми: Як бачимо, збігається з розв’язком на папері.

```
S:
     2     0.5       0       1
     0 1.65831       0 -0.301511
     0       0 1.41421 0.707107
     0       0       0 1.55212
D:
     1       0       0       0
     0       1       0       0
     0       0       1       0
     0       0       0       1
x[1] = 1
x[2] = 2
x[3] = 3
x[4] = 4
Визначник: 53
Для продолжения нажмите любую клавишу . . .
```

Метод Зейделя

Використана теорія:

Метод Зейделя

Метод Зейделя є ітераційним методом для розв'язання СЛАР $Ax = b$, т.т. розв'язок знаходимо із заданою точністю ε. Початкове наближення x^0 обираємо довільним чином. Ітераційний процес має вигляд:

$$x_i^{k+1} = -\sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} x_j^{k+1} - \sum_{j=i+1}^{n} \frac{a_{ij}}{a_{ii}} x_j^k + \frac{b_i}{a_{ii}}. \qquad (22)$$

Достатня умова збіжності 1. Якщо $\forall i : i = \overline{1, n}$ виконується нерівність

$$|a_{ii}| \geqslant \sum_{j=1, j \neq i}^{n} |a_{ij}|,$$

то ітераційний процес методу Зейделя (22) збігається, при чому швидкість збіжності лінійна.

Достатня умова збіжності 2. Якщо $A = A^T > 0$, то ітераційний процес методу Зейделя (22) збігається, при чому швидкість збіжності лінійна.

Умова припинення: $||x^n - x^{n-1}|| \leqslant \varepsilon$.

Необхідні і достатні умови збіжності. Для $\forall x^0$ ітераційний процес методу Зейделя (22) збігається тоді і тільки

тоді, коли $|\lambda| < 1$, де λ – це корені нелінійного рівняння:

$$\begin{vmatrix} \lambda a_{11} & a_{12} & \dots & a_{1n} \\ \lambda a_{21} & \lambda a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ \lambda a_{n1} & \lambda a_{n2} & \dots & \lambda a_{nn} \end{vmatrix} = 0.$$

Достатня умова збіжності: Візьмемо першу умову і перевіримо на нашій задачі:

a11=4>1+2+0=3

a22=3>1+0+0=1

a33=2>0+0+1=1

a44=4>2+0+1=3

Умова виконується.

Ітераційний процес:

Формули:

$$x_1^{k+1} = \frac{1}{a_{11}}\left(-a_{12}x_2^{k} - a_{13}x_3^{k} - a_{14}x_4^{k} + b_1\right)$$

$$x_2^{k+1} = \frac{1}{a_{22}}\left(-a_{21}x_1^{k+1} - a_{23}x_3^{k} - a_{24}x_4^{k} + b_2\right)$$

$$x_3^{k+1} = \frac{1}{a_{33}}\left(-a_{31}x_1^{k+1} - a_{32}x_2^{k+1} - a_{34}x_4^{k} + b_3\right)$$

$$x_4^{k+1} = \frac{1}{a_{44}}\left(-a_{41}x_1^{k+1} - a_{42}x_2^{k+1} - a_{43}x_3^{k+1} + b_4\right)$$

У прикладі візьмемо точність e=0.01;

Реалізація в програмі:

Ініціалізуємо змінні:

```
setlocale(LC_ALL, "ukr");
// змінні
float A[4][4] = { {4,1,0,2} ,{1,3,0,0} ,{0,0,2,1} ,{2,0,1,4}, };
float b[4] = {14,7,10,21};
float x1[4];
float x2[4];
float eps=0.01;//точність
float error[4];//тут перевіряємо точність
```

Перевірка на коректність матриці:

```
for (int i = 0; i < 4; i++)
{
    if (A[i][i] < A[i][0] + A[i][1] + A[i][2] + A[i][3] - A[i][i])
        cout << "Матриця некоректна" << endl;
}
```

Початкову ітерацію беремо (0,0,0,0):

```
//початкове наближення x0(0.0.0.0)
for (int i = 0; i < 4; i++)
{
    x1[i] = 0;
}
int iter = 1;//к-сть ітерацій
```

Сам процес ітерації (з виведенням кожної):

```
do {
    x2[0] = (-A[0][1] * x1[1] - A[0][2] * x1[2] - A[0][3] * x1[3]+b[0]) / A[0][0];
    x2[1] = (-A[1][0] * x1[0] - A[1][2] * x1[2] - A[1][3] * x1[3]+b[1]) / A[1][1];
    x2[2] = (-A[2][1] * x1[1] - A[2][0] * x1[0] - A[2][3] * x1[3]+b[2]) / A[2][2];
    x2[3] = (-A[3][1] * x1[1] - A[3][2] * x1[2] - A[3][0] * x1[0]+b[3]) / A[3][3];

    error[0] = abs(x1[0] - x2[0]);
    error[1] = abs(x1[1] - x2[1]);
    error[2] = abs(x1[2] - x2[2]);
    error[3] = abs(x1[3] - x2[3]);

    if (iter == 1) cout << "Норма векторів на ітерації 1: " << setprecision(10) << MaxVector(error) << endl;
    iter++;
    x1[0] = x2[0];
    x1[1] = x2[1];
    x1[2] = x2[2];
    x1[3] = x2[3];

    cout << "Ітерація " << iter << endl;
    cout << "x[1] = " << setprecision(10) << x1[0] << endl;
    cout << "x[2] = " << setprecision(10) << x1[1] << endl;
    cout << "x[3] = " << setprecision(10) << x1[2] << endl;
    cout << "x[4] = " << setprecision(10) << x1[3] << endl;

} while (MaxVector(error) > eps);
```

Виведення результату:

```
cout << "Норма векторів на останній ітерації: " << MaxVector(error) << endl;
cout << "x[1] = " << setprecision(10) << x1[0]<<endl;
cout << "x[2] = " << setprecision(10) << x1[1] << endl;
cout << "x[3] = " << setprecision(10) << x1[2] << endl;
cout << "x[4] = " << setprecision(10) << x1[3] << endl;
```

Результат програми:

```
Норма векторів на ітерації 1: 5
Ітерація 2
x[1] = 3.5
x[2] = 1.166666627
x[3] = 5
x[4] = 2.25
Ітерація 3
x[1] = 2.083333492
x[2] = 1.638888836
x[3] = 3.875
x[4] = 3.239583254
Ітерація 4
x[1] = 1.470486164
x[2] = 1.843171239
x[3] = 3.380208492
x[4] = 3.669704914
Ітерація 5
x[1] = 1.204354763
x[2] = 1.931881785
x[3] = 3.165147543
x[4] = 3.856535673
Ітерація 6
x[1] = 1.088761806
x[2] = 1.970412731
x[3] = 3.071732044
x[4] = 3.937685966
```

```
Ітерація 7
x[1] = 1.038553715
x[2] = 1.987148762
x[3] = 3.031157017
x[4] = 3.972933769
Ітерація 8
x[1] = 1.016746044
x[2] = 1.994418025
x[3] = 3.013533115
x[4] = 3.98824358
Ітерація 9
x[1] = 1.007273674
x[2] = 1.997575402
x[3] = 3.00587821
x[4] = 3.994893551
Норма векторів на останній ітерації: 0.009472370148
x[1] = 1.007273674
x[2] = 1.997575402
x[3] = 3.00587821
x[4] = 3.994893551
```

Число обумовленості

```
double A[4][4]= { {4,1,0,2} ,{1,3,0,0} ,{0,0,2,1} ,{2,0,1,4}, };
double m1 = matnorm(A);
inversion(A, 4);
cout << "Обернена матриця:" << endl;
for (int i = 0; i < 4; i++)
{
    for (int j = 0; j < 4; j++)
        cout << A[i][j] << "  ";

    cout << std::endl;
}
double m2 = matnorm(A);
cout << "Число обумовленості:" << endl;
cout << m1 * m2 << endl;
```

```
Обернена матриця:
0.396226  -0.132075  0.113208  -0.226415
-0.132075  0.377358  -0.0377358  0.0754717
0.113208  -0.0377358  0.603774  -0.207547
-0.226415  0.0754717  -0.207547  0.415094
Число обумовленості:
6.73585
```

Висновок

У цій лабораторній роботі я вирішила систему лінійних рівнянь двома способами: методом Зейделя та методом квадратного кореня. Метод квадратного кореня легший «на папері», і, можливо, більш універсальний. Метод Зейделя легше реалізувати у програмі, він чудово працює при великих матрицях.