МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет “Львівська політехніка”
Інститут післядипломної освіти

ЗВІТ

Про виконання лабораторної роботи №3

«Ітераційні методи розв’язування системи лінійних рівнянь»

з дисципліни «Чисельні методи»

Виконав:
слухач групи ПЗС-11
Гринчук Тарас

Прийняла:
ст. викл. Мельник Н. Б.

«   »______________ 2014 р.

∑ ______________

ЛЬВІВ – 2014

Тема роботи: Ітераційні методи розв'язування системи лінійних рівнянь.

Мета роботи: Ознайомлення на практиці з ітераційнними методами розв'язування систем лінійних алгебраїчних рівнянь.

1. Теоретичні відомості

Метод послідовних наближень (Якобі)

Розглянемо систему лінійних алгебраїчних рівнянь виду:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ a_{31}x_1 + a_{32}x_2 + \ldots + a_{3n}x_n = b_3 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases}$$

Для розв'язання системи методами послідовних наближень необхідно виконати наступні кроки:

Кожне рівняння системи розділити на діагональний елемент a_{kk} де k=1,2...n, n – кількість рівнянь в системі, і перетворити кожне рівняння системи відносно координат вектора, індекс якого співпадає з номером рівняння:

$$\begin{cases} x_1 = \frac{b_1}{a_{11}} - (\frac{a_{12}}{a_{11}}x_2 + \frac{a_{13}}{a_{11}}x_3 + \ldots + \frac{a_{1n}}{a_{11}}x_n) \\ x_2 = \frac{b_2}{a_{22}} - (\frac{a_{21}}{a_{22}}x_1 + \frac{a_{23}}{a_{22}}x_3 + \ldots + \frac{a_{2n}}{a_{22}}x_n) \\ x_3 = \frac{b_3}{a_{33}} - (\frac{a_{31}}{a_{33}}x_1 + \frac{a_{32}}{a_{33}}x_2 + \ldots + \frac{a_{1n}}{a_{33}}x_n) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n = \frac{b_n}{a_{nn}} - (\frac{a_{n1}}{a_{nn}}x_1 + \frac{a_{n2}}{a_{nn}}x_2 + \ldots + \frac{a_{nn-1}}{a_{nn}}x_{n-1}) \end{cases}$$

Якщо ввести перепозначення

$$\frac{b_k}{a_{kk}} = \beta_k, \qquad -\frac{a_{ki}}{a_{kk}} = \alpha_{ki},$$

де k=1,2…n; i=1,2…n, то система матиме вигляд:

$$\begin{cases} x_1 = \beta_1 + \alpha_{11}x_2 + \alpha_{12}x_2 + \ldots + \alpha_{1n}x_n \\ x_2 = \beta_2 + \alpha_{11}x_1 + \alpha_{22}x_2 + \ldots + \alpha_{2n}x_n \\ x_3 = \beta_3 + \alpha_{11}x_1 + \alpha_{22}x_2 + \ldots + \alpha_{3n}x_n \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n = \beta_n + \alpha_{11}x_1 + \alpha_{22}x_2 + \ldots + \alpha_{nn}x_n \end{cases}$$

Така система називається зведеною до нормального вигляду.

$$x = \beta + \alpha \cdot x .$$

Якщо деяким чином визначити, так званий, вектор початкових значень $x^{(0)}$, який знаходиться в правій частині, то можна отримати певні значення вектора $x^{(1)}, x^{(2)}, \ldots x^{(n)}$

$$x^{(n)} = \beta + \alpha \cdot x^{(n-1)}.$$

<u>Вибір початкового наближення</u> $x^{(0)}$

В якості вектора початкових наближень $x^{(0)}$ вибирають:

- вектор, в якого всі координати хі дорівнюють 0;
- вектор, в якого всі координати хі дорівнюють 1;
- вектор, координати x_i якого дорівнюють координатам вектора вільних членів β_i ;

• координати вектору $\bar{x}$ вибирають в результаті аналізу особливостей об'єкту дослідження та задачі, яка розв'язується.

Умова закінчення ітераційного процесу

$$\left|\bar{x}^{(n)} - \bar{x}^{(n-1)}\right| \le \varepsilon .$$

Умови збіжності ітераційного процесу (теорема про збіжність)

Ітераційний процес пошуку розв'язку системи лінійних алгебраїчних рівнянь виду (1) наближеними методами збігається, якщо будь-яка канонічна норма матриці $\|\alpha\| < 1$.

Канонічні норми матриці

$$\|A\|_1 = \max_j \sum_{i=1}^{m} |a_{ij}| , \quad \|A\|_2 = \sqrt{\lambda_{\max}(AA^T)} , \quad \|A\|_\infty = \max_i \sum_{j=1}^{m} |a_{ij}| .$$

Норма матриці - додатне число, яке визначається за такими умовами:

перша канонічна норма – це максимальна з сум модулів елементів матриці коефіцієнтів α по стрічках:

$$\|\alpha\|_1 = \max_i \sum_{j=1}^{n} |\alpha_{ij}|$$

друга канонічна норма – це максимальна з сум модулів елементів матриці коефіцієнтів α по стовбцях:

$$\|\alpha\|_2 = \max_j \sum_{i=1}^{n} |\alpha_{ij}| ,$$

третя канонічна норма – це корінь квадратний з сум квадратів модулів всіх елементів матриці коефіцієнтів α:

$$\|\alpha\|_3 = \sqrt{\sum_i \sum_j |\alpha_{ij}|^2} .$$

Метод Зейделя

Метод можна розглядати як модифікацію метода Якобі.

Основна ідея - при обчисленні чергового (n+1)-го наближення до невідомого xi при i >1 використовують вже знайдені (n+1)-е наближення до x1, x2, ..., xi - 1, а не n-е наближення, як в методі Якобі.

Розрахункова формула методу:

$$x_i^{(n+1)} = b_{i1}x_1^{(n+1)} + b_{i2}x_2^{(n+1)} + \ldots + b_{i,i-1}x_{i-1}^{(n+1)} + b_{i,i+1}x_{i+1}^{(n)} + \ldots + b_{im}x_m^{(n)} + d_i ,$$

i = 1, 2, ... m..

Умова збіжності і критерій закінчення ітерацій аналогічні як в методі Якобі.

Якщо матриця А - симетрична і додатньо визначена, то при будь-якому виборі початкового наближення метод Зейделя збіжний.

2. Хід роботи

Завдання 1 (варіант 5). Розв'язати систему лінійних рівнянь методом ітерацій з точністю до 0,001, наперед оцінивши число необхідних для цього кроків.

Система рівнянь вже перетворена до вигляду:
$X = \alpha X + \beta$, зручному для ітерацій. Кількість кроків визначити із співвідношення:

$$\left\| X^* - X^{(k)} \right\| \le \frac{\|\alpha\|^{k+1}}{1 - \|\alpha\|} \|\beta\|$$

$$\begin{cases} x_1 = 0{,}18x_1 - 0{,}34x_2 - 0{,}12x_3 + 0{,}15x_4 - 1{,}33 \\ x_2 = 0{,}11x_1 + 0{,}23x_2 - 0{,}15x_3 + 0{,}32x_4 + 0{,}84 \\ x_3 = 0{,}05x_1 - 0{,}12x_2 + 0{,}14x_3 - 0{,}18x_4 - 1{,}16 \\ x_4 = 0{,}12x_1 + 0{,}08x_2 + 0{,}06x_3 + 0{,}57 \end{cases}$$

Завдання 2. Розв'язати систему лінійних рівнянь методом Зейделя з точністю до 0,001.

$$\begin{cases} x + 3.9y + z = 10.2; \\ x + 2.7y - 2z = 11; \\ 2x - 4.4y + z = 5; \end{cases}$$

Звіт до лабораторної роботи повинен містити такі структурні елементи:

1. Титульний аркуш.
2. Тема.
3. Мета.
4. Короткі теоретичні відомості.
5. Алгоритм розв'язку СЛАР.
6. Текст програми з коментарями.
7. Вигляд реалізованої програми.
8. Висновки.

3. Текст програми методу Якобі

```
#include <iostream>
#include <conio.h>
#include <clocale>
#include <math.h>
#include <cmath>

using namespace std;

int const N = 4;
const double eps = 0.001;

//якщо 0 - то ввід вхідних даних з клавіатури
int const testMode = 1;

double A[N][N], X[N], F[N];

//зчитування масиву з клавіатури
void input(double a[N][N], int n,  int m) {
	for(int i=0; i < n; i++)
		for(int j = 0; j < m; j++) {
			cout << "a[" << i << "][" << j << "]: ";
			cin >> a[i][j];
		}
}

//виведення матриці на екран
void display(double a[N][N], int n,  int m) {
	for(int i=0; i < n; i++) {
		for(int j = 0; j < m; j++)
			cout << a[i][j] << "\t";
		if(m > 1) cout<<endl;
	}
}

void inputVector(double a[N], int m) {
	for(int j = 0; j < m; j++) {
		cout << "f[" << j << "]: ";
		cin >> a[j];
	}
}

void displayVector(double X[N], int n) {
	for(int j = 0; j < n; j++)
		cout << X[j] << "\t";
}

//перевірка на евклідову норму матриці
int EvklidTest(double a[N][N]) {
	double alpha[N][N];

	for(int i=0; i < N; i++)
		for(int j = 0; j < N; j++)
			if(i!=j) alpha[i][j] = (-a[i][j])/(a[i][i]);
			else alpha[i][j] = 0;

	double Ek = 0;
	for(int i=0; i < N; i++) {
		for(int j = 0; j < N; j++)
			Ek += alpha[i][j] * alpha[i][j];
	}
	Ek = pow(Ek,0.5);
	if(Ek > 1.0) {
```

```
			cout<<"\n\n\nЕвклідову норма матриці "<<Ek<<" > 1; Програму буде завершено !\n";
			_getch();
			return 0;
		}
		return 1;
}

void main()	{
		setlocale(LC_ALL, "Ukrainian");
		if(testMode) {
			A[0][0] = -0.82;
			A[0][1] = -0.34;
			A[0][2] = -0.12;
			A[0][3] = 0.15;

			A[1][0] = 0.11;
			A[1][1] = -0.77;
			A[1][2] = -0.15;
			A[1][3] = 0.32;

			A[2][0] = 0.05;
			A[2][1] = -0.12;
			A[2][2] = -0.86;
			A[2][3] = -0.18;

			A[3][0] = 0.12;
			A[3][1] = 0.08;
			A[3][2] = 0.06;
			A[3][3] = -1.0;

			F[0] = -1.33;
			F[1] = 0.84;
			F[2] = -1.16;
			F[3] = 0.57;

		} else {
			cout<<"Введіть матрицю A:\n";
			input(A, N, N);

			cout<<"\nВведіть стовпець F:\n";
			inputVector(F, N);
		}

		//Вивід a та f
		cout<<"A:\n";
		display(A, N, N);
		cout<<"\nB:\n";
		displayVector(F, N);

		if(!EvklidTest(A)) return;

		double TempX[N];
		//норма, розрахована як найбільша різниця компонент стовпця іксів сусідніх ітерацій
		double norm;

		//обчислення розвязків
		do {
			for (int i = 0; i < N; i++) {
				TempX[i] = F[i];
				for (int g = 0; g < N; g++) {
					if (i != g)
						TempX[i] -= A[i][g] * X[g];
```

```
                }
                TempX[i] /= A[i][i];
            }
              norm = fabs(X[0] - TempX[0]);
            for (int h = 0; h < N; h++) {
                if (fabs(X[h] - TempX[h]) > norm)
                    norm = fabs(X[h] - TempX[h]);
                X[h] = TempX[h];
            }
        } while (norm > eps);


        //виведення результату
        cout<<"\n\nX:\n";
        displayVector(X, N);

        _getch();
}
```

4. Текст програми методу Зейделя

```
#include <iostream>
#include <conio.h>
#include <clocale>
#include <math.h>
#include <cmath>

using namespace std;

int const N = 3;
const double eps = 0.001;

//якщо 0 - то ввід вхідних даних з клавіатури
int const testMode = 1;

double A[N][N], X[N], F[N];

//зчитування масиву з клавіатури
void input(double a[N][N], int n,  int m) {
	for(int i=0; i < n; i++)
		for(int j = 0; j < m; j++) {
			cout << "a[" << i << "][" << j << "]: ";
			cin >> a[i][j];
		}
}

//виведення матриці на екран
void display(double a[N][N], int n,  int m) {
	for(int i=0; i < n; i++) {
		for(int j = 0; j < m; j++)
			cout << a[i][j] << "\t";
		if(m > 1) cout<<endl;
	}
}

void inputVector(double a[N], int m) {
	for(int j = 0; j < m; j++) {
		cout << "f[" << j << "]: ";
		cin >> a[j];
	}
}

void displayVector(double X[N], int n) {
	for(int j = 0; j < n; j++)
		cout << X[j] << "\t";
}

//перевірка на евклідову норму матриці
int EvklidTest(double a[N][N]) {
	double alpha[N][N];

	for(int i=0; i < N; i++)
		for(int j = 0; j < N; j++)
			if(i!=j) alpha[i][j] = (-a[i][j])/(a[i][i]);
			else alpha[i][j] = 0;

	double Ek = 0;
	for(int i=0; i < N; i++) {
		for(int j = 0; j < N; j++)
			Ek += alpha[i][j] * alpha[i][j];
	}
	Ek = pow(Ek,0.5);
	if(Ek > 1.0) {
```

```
              cout<<"\n\n\nЕвклідова норма матриці "<<Ek<<" > 1; Програму буде завершено
!\n";
              _getch();
              return 0;
       }
       return 1;
}

void main()   {
       setlocale(LC_ALL, "Ukrainian");
       if(testMode) {
              A[0][0] = 1.0;
              A[0][1] = 3.9;
              A[0][2] = 1.0;

              A[1][0] = 1.0;
              A[1][1] = 2.7;
              A[1][2] = -2.0;

              A[2][0] = 2.0;
              A[2][1] = -4.4;
              A[2][2] = 1.0;

              F[0] = -10.2;
              F[1] = -11.0;
              F[2] = -5.0;
       } else {
              cout<<"Введіть матрицю A:\n";
              input(A, N, N);

              cout<<"\nВведіть стовпець F:\n";
              inputVector(F, N);
       }

       //Вивід а та f
       cout<<"A:\n";
       display(A, N, N);
       cout<<"\nB:\n";
       displayVector(F, N);

       if(!EvklidTest(A)) return;

       double norm, v, s;
       int i;
       for (i = 0; i < N; i++) X[i] = 0;
       do {
              norm = 0;
              for (i = 0; i < N; i++) {
                     s = 0;
                     for (int j = 0; j < N; j++)
                            if (i != j) s += A[i][j] * X[j];
                     v = X[i];
                     X[i] = (F[i] - s) / A[i][i];
                     norm = fabs(X[i] - v);
              }
       } while (norm > eps);

       //виведення результату
       cout<<"\n\nX:\n";
       displayVector(X, N);

       _getch();
}
```

5. Результати виконання програм

Запустимо програму методу Якобі на виконання (рис. 5.1):

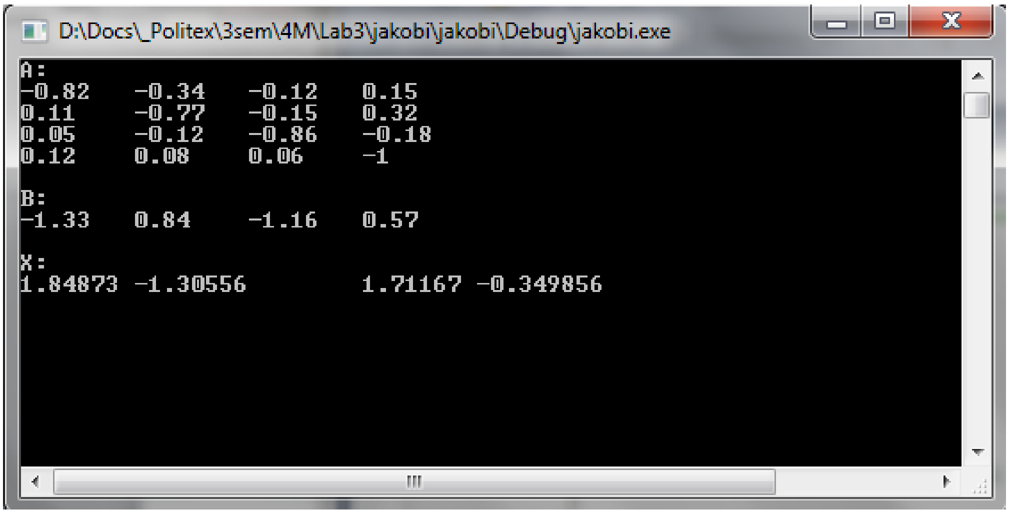

Рис. 5.1. Метод Якобі

Запустимо програму методу Зейделя на виконання (рис. 5.2):

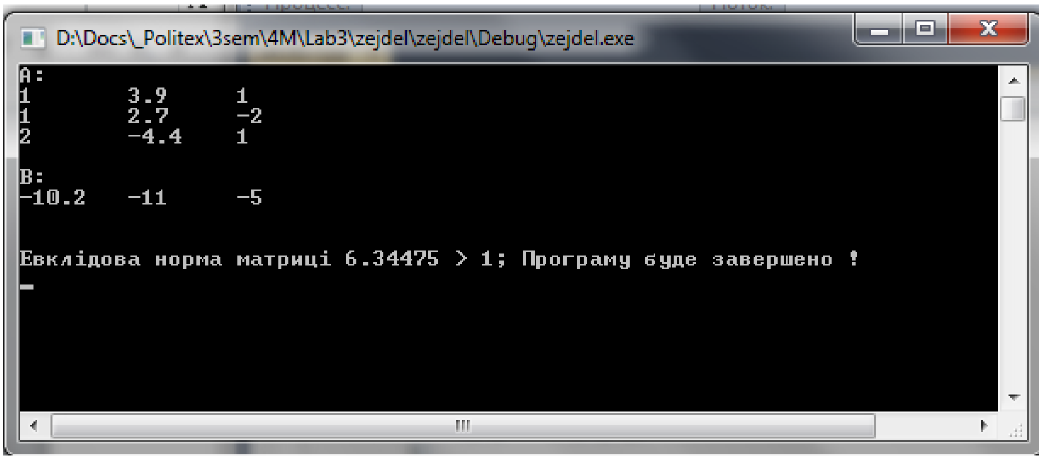

Рис. 5.2. Метод Зейделя

6. ВИСНОВКИ

В ході даної лабораторної роботи я оволодів ітераційними методами розв’язування систем лінійних алгебраїчних рівнянь, а саме:

1)методом Якобі;

2)методом Зейделя.