Національний університет «Львівська політехніка»
Інститут комп’ютерних наук та інформаційних технологій

ЗВІТ

Про виконання лабораторної роботи № 5

«Наближені методи розв’язування систем лінійних алгебраїчних рівнянь»

з дисципліни «Чисельні методи»

Лектор:
доцент кафедри ПЗ
Мельник Н.Б.

Виконав:
студ. групи ПЗ-15
Марущак А. С.

Прийняв:
асистент кафедри ПЗ
Гарматій Г.Ю
«___» ______ 2022 р.
$\sum$ = ____ ___________

Львів – 2022

Тема роботи: Наближені методи розв'язування систем лінійних алгебраїчних рівнянь

Мета роботи: ознайомлення на практиці з методами Якобі та Зейделя розв'язування систем лінійних алгебраїчних рівнянь

Теоретичні відомості

Розглянемо систему лінійних алгебраїчних рівнянь

$$A \cdot X = B$$

Наближені методи розв'язування СЛАР полягають у тому, що вектор $\bar{X}$ знаходять як границю послідовних наближень $\{X^{(n)}\}$, де n – номер ітерації. Навіть в припущенні, що обчислення виконують без заокруглень, ітераційні методи дають змогу визначити розв'язок тільки з заданою точністю

$$\|X^{(n)} - X^*\| < \varepsilon$$

Для розв'язування СЛАР наближеними методами розглянемо метод простої ітерації (Якобі) та метод Зейделя.

Метод Якобі

Розглянемо систему лінійних алгебраїчних рівнянь виду:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots + a_{2n}x_n = b_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \ldots + a_{nn}x_n = b_n. \end{cases}$$

Припустивши, що коефіцієнти $a_{ii} \neq 0$ $(i = \overline{1,n})$, розв'яжемо i -те рівняння системи відносно x_i. У результаті отримаємо таку систему:

$$\begin{cases} x_1 = \dfrac{b_1}{a_{11}} - \left(\dfrac{a_{12}}{a_{11}}x_2 + \dfrac{a_{13}}{a_{11}}x_3 + \ldots + \dfrac{a_{1n}}{a_{11}}x_n \right), \\ x_2 = \dfrac{b_2}{a_{22}} - \left(\dfrac{a_{21}}{a_{22}}x_1 + \dfrac{a_{23}}{a_{22}}x_3 + \ldots + \dfrac{a_{2n}}{a_{22}}x_n \right), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n = \dfrac{b_n}{a_{nn}} - \left(\dfrac{a_{n1}}{a_{nn}}x_1 + \dfrac{a_{n2}}{a_{nn}}x_2 + \ldots + \dfrac{a_{n,n-1}}{a_{nn}}x_{n-1} \right). \end{cases}$$

Введемо позначення

$$\beta_i = \frac{b_i}{a_{ii}}, \quad \alpha_{ij} = -\frac{a_{ij}}{a_{ii}}, \quad i = \overline{1,n}, \quad j \neq i$$

Тоді систему можна переписати у вигляді:

$$X = \beta + \alpha X$$

Нехай перше наближення системи рівне стовпцю вільних членів, тобто

$$X^{(0)} = \beta.$$

Тоді наближення розв’язку можна описати рекурентною формулою:

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

Якщо послідовність $\{X^{(n)}\}$ є збіжною, то тоді $X = \lim_{n\to\infty} X^{(n)}$ – розв'язок системи.

Алгоритм методу Якобі:

1. Повторюючи вищенаведені кроки записуємо систему в зведеному вигляді.
2. На основі отриманої СЛАР записуємо матрицю α та β.
3. Встановлюємо початкове наближення рівним вектору-стовпцю β: $X^{(0)} = \beta$.
4. Допоки не досягнемо потрібної точності, використовуючи рекурентну формулу $X^{(n)} = \beta + \alpha \cdot X^{(n-1)}$, уточнюємо розв’язок.

Метод Зейделя

Даний метод є модифікацією методу простої ітерації. Основна його ідея полягає в тому, що при обчисленні чергового k-го наближення розв’язку $x_i^{(k)}$ ($i = \overline{1,n}$) використовують вже знайдені значення k-го наближеного розв’язку $x_1^{(k)}, x_2^{(k)}, \dots, x_{i-1}^{(k)}$.

Ітераційна формула методу Зейделя має вигляд

$$x_i^{(k)} = \beta_i + \sum_{j=1}^{i-1} \alpha_{ij} x_j^{(k)} + \sum_{j=i+1}^{n} \alpha_{ij} x_j^{(k-1)}, \quad i = \overline{1,n}, \quad k = 1,2,\dots$$

Алгоритм методу Зейделя аналогічний до алгоритму методу Якобі за винятком формули, що використовується у кроці (4).

Збіжність ітераційного процесу

Збіжність ітераційного процесу залежить від величини коефіцієнтів матриці α. Тому не обов’язково за нульове наближення розв’язку вибирати стовпець вільних членів. За початкове наближення $X^{(0)}$ можна також вибирати

- вектор, усі координати якого $x_i^{(0)} = 0$ ($i = \overline{1,n}$);
- вектор, усі координати якого $x_i^{(0)} = 1$ ($i = \overline{1,n}$);
- вектор $X^{(0)}$, отриманий у результаті аналізу особливостей об’єкту дослідження та задачі, яку розв’язують.

Якщо елементи матриці α задовольняють одну з умов

$$\sum_{j=1}^{n}\left|\frac{a_{ij}}{a_{ii}}\right|<1, \quad i=\overline{1,n},$$

$$\sum_{i=1}^{n}\left|\frac{a_{ij}}{a_{ii}}\right|<1, \quad j=\overline{1,n},$$

$$\sum_{i,j=1}^{n}\left(\frac{a_{ij}}{a_{ii}}\right)^2<1,$$

то система рівнянь має єдиний розв’язок X^*, який не залежить від початкового наближення.

Критерії припинення ітераційного процесу

Якщо задана похибка ε наближеного розв’язку, то критерієм припинення ітераційного процесу вважають виконання однієї з умов:

- модуль різниці між наступним та попереднім наближенням розв’язку повинен бути меншим за ε

$$\left|X^{(k)}-X^{(k-1)}\right|=\sqrt{\sum_{i=1}^{n}\left(x_i^{(k)}-x_i^{(k-1)}\right)^2}<\varepsilon, \quad k=1,2,\ldots;$$

- максимальне значення модуля різниць між відповідними компонентами наступного та попереднього наближення розв’язку повинно бути меншим за ε

$$\max_i\left|x_i^{(k)}-x_i^{(k-1)}\right|<\varepsilon, \quad i=\overline{1,n}, \quad k=1,2,\ldots;$$

- максимальне значення модуля відносних різниць між відповідними компонентами наступного та попереднього наближення розв’язку повинно бути меншим за ε

$$\max_i\left|\frac{x_i^{(k)}-x_i^{(k-1)}}{x_i^{(k)}}\right|<\varepsilon, \quad |x_i|>>1, \quad i=\overline{1,n}, \quad k=1,2,\ldots .$$

Індивідуальне завдання

1. Ознайомитися з теоретичним матеріалом.
2. Скласти програму розв’язування системи лінійних алгебраїчних рівнянь методом Якобі та Зейделя з точністю $\varepsilon=0{,}001$. Порівняти кількість ітерацій для обох методів.

$$\begin{cases} 0,91x_1 + 0,23x_2 - 0,44x_3 - 0,05x_4 = 2,13 \\ 0,04x_1 + 0,7x_2 - 0,31x_3 + 0,15x_4 = -0,18 \\ 0,06x_1 + 0,15x_2 - 0,83x_3 - 0,17x_4 = 1,44 \\ 0,72x_1 - 0,08x_2 - 0,05x_3 + 1,08x_4 = 2,42 \end{cases}$$

Хід роботи

Проведемо деякі обчилення вручну, щоб потім мати змогу порівняти результат виконання програми з дійсним.

По-перше, поділимо кожен i-й рядок на коефіцієнт, що стоїть біля x_i. Тоді отримаємо наступне:

$$\begin{cases} x_1 + 0,253x_2 - 0,484x_3 - 0,055x_4 = 2,341 \\ 0,057x_1 + x_2 - 0,443x_3 + 0,214x_4 = -0,257 \\ -0,072x_1 - 0,181x_2 + x_3 + 0,205x_4 = -1,735 \\ 0,667x_1 - 0,074x_2 - 0,046x_3 + x_4 = 2,241 \end{cases}$$

Перенесемо з лівої частини i-го рядка всі члени, що не містять x_i, у праву сторону з протилежним знаком. Тоді наша СЛАР набуде наступного вигляду:

$$\begin{cases} x_1 = 2,341 + (-0,253x_2 + 0,484x_3 + 0,055x_4) \\ x_2 = -0,257 + (-0,057x_1 + 0,443x_3 - 0,214x_4) \\ x_3 = -1,735 + (0,072x_1 + 0,181x_2 - 0,205x_4) \\ x_4 = 2,241 + (-0,667x_1 + 0,074x_2 + 0,046x_3) \end{cases}$$

Нехай $\alpha = \begin{pmatrix} 0 & -0,253 & 0,484 & 0,055 \\ -0,057 & 0 & 0,443 & -0,214 \\ 0,072 & 0,181 & 0 & -0,205 \\ -0,667 & 0,074 & 0,046 & 0 \end{pmatrix}, \beta = \begin{pmatrix} 2,341 \\ -0,257 \\ -1,735 \\ 2,241 \end{pmatrix}, X = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix},$ тоді запишемо систему у компактному вигляді:

$$X = \beta + \alpha X$$

Обчислимо хоча б одну канонічну норму матриці α, щоб переконатись у збіжності ітераційного процесу. Нехай це буде максимум суми модулів елементів кожного рядка:

$\|\alpha\| = \max_{i\in[1,4]\cap\mathbb{Z}} \sum_{j=1}^{4} |\alpha_{ij}| = \max(\{0.792, 0.714, 0,787\}) = 0,792 < 1$,

Отже, ітераційний процес буде збіжним.

Нехай початкове наближення $X^{(0)} = \beta = \begin{pmatrix} 2{,}341 \\ -0{,}257 \\ -1{,}735 \\ 2{,}241 \end{pmatrix}$. Тоді проведемо обчислення до точності $\varepsilon = 0{,}001$ за рекурентною формулою методом Якобі та Зейделя, обчислюючи різницю між векторами-стовпцями $X^{(k)}$ та $X^{(k-1)}$, як відстань між відповідними векторами у просторі, тобто

$$\left|X^{(k)} - X^{(k-1)}\right| = \sqrt{\sum_{i=1}^{n}\left(x_i^{(k)} - x_i^{(k-1)}\right)^2}$$

Результати обчислень подамо у вигляді таблиці:

Метод Якобі

№ ітерації	x_1	x_2	x_3	x_4	$\lvert X^{(i)} - X^{(i-1)}\rvert$
0	**2,341**	**-0,257**	**1,735**	**2,241**	**-**
1	**1,690**	**-1,639**	**-2,071**	**0,581**	**2,281**
2	**1,785**	**-1,395**	**-2,028**	**0,897**	**0,413**
3	**1,762**	**-1,449**	**-2,042**	**0,853**	**0,075**
4	**1,767**	**-1,445**	**-2,044**	**0,864**	**0,013**
5	**1,765**	**-1,449**	**-2,045**	**0,861**	**0,005**
6	**1,765**	**-1,448**	**-2,046**	**0,862**	**0,00096**

Метод Зейделя

№ ітерації	x_1	x_2	x_3	x_4	$\lvert X^{(i)} - X^{(i-1)}\rvert$
0	**2,341**	**-0,257**	**1,735**	**2,241**	**-**
1	**1,690**	**-1,602**	**-2,361**	**0,886**	**2,112**
2	**1,653**	**-1,587**	**-2,084**	**0,925**	**0,283**
3	**1,785**	**-1,480**	**-2,063**	**0,846**	**0,189**
4	**1,764**	**-1,453**	**-2,043**	**0,863**	**0,043**
5	**1,767**	**-1,448**	**-2,046**	**0,861**	**0,007**
6	**1,765**	**-1,448**	**-2,045**	**0,862**	**0,003**
7	**1,765**	**-1,449**	**-2,046**	**0,862**	**0,0006**

Отож, в обох методах отримали однаковий результат:

$$X = (1{,}765 \pm 0{,}001 \quad -1{,}448 \pm 0{,}001 \quad -2{,}046 \pm 0{,}001 \quad 0{,}862 \pm 0{,}001)^T$$

Наступна мета – реалізувати подані методи програмно і переконатися в достовірності результату.

Для подальшої роботи я буду використовувати структуру даних Matrix, яка відповідає матриці заданого розміру, як зрозуміло з назви. Серед її функціональних можливостей – розклад на матриці α та β, якщо матриця представляє собою матрицю системи, а також множення і додавання матриць, перевірку на збіжність за допомогою канонічних норм. Покажемо це у коді:

```
public void Deconstruct(out Matrix alpha, out Matrix beta)
{
            if (Columns != Rows + 1)
                throw new Exception("Not acceptable matrix to decompose!");
            Matrix A = new(Rows, Rows);
            Matrix B = new(Rows, 1);

            for(int i = 0; i < Rows; i++)
            {
                A.data[i] = this.data[i].Take(rows).Select((a, j) => j == i ? 0 : -a /
this.data[i][i]).ToArray();
                B[i, 0] = this.data[i].Last() / this.data[i][i];
            }

            alpha = A;
            beta = B;
}
public static Matrix operator*(Matrix A, Matrix B)

{
            if (A.Columns != B.Rows)
                throw new ArgumentException();

            Matrix result = new(A.Rows, B.Columns);

            for(int i = 0; i < result.Rows; i++)
            {
                for(int j = 0; j < result.Columns; j++)
                {
                    double sum = 0;
                    for(int k = 0; k < A.Columns; k++)
                    {
                        sum += A[i, k] * B[k, j];
                    }
                    result[i, j] = sum;
                }
            }
            return result;
}
public static Matrix operator+(Matrix A, Matrix B)
{
            if (A.Columns != B.Columns || A.Rows != B.Rows)
                throw new ArgumentException();

            Matrix result = new(A.Rows, A.Columns);

            for (int i = 0; i < result.Rows; i++)
            {
                for (int j = 0; j < result.Columns; j++)
                {
                    result[i, j] = A[i, j] + B[i, j];
                }
```

```
            }
            return result;
}
public bool TestNorms()
{
            if (Rows != Columns) throw new Exception();

            double firstNorm = this.Select(row => row.Sum(elem => Math.Abs(elem))).Max();
            double secondNorm = this.Transposed().Select(row => row.Sum(elem =>
Math.Abs(elem))).Max();
            double thirdNorm = Math.Sqrt(this.Sum(row => row.Sum(elem => elem * elem)));

            Console.WriteLine($"Перша норма: {firstNorm:0.000}");
            Console.WriteLine($"Друга норма: {secondNorm:0.000}");
            Console.WriteLine($"Третя норма: {thirdNorm:0.000}");

            if (firstNorm < 1) return true;
            if (secondNorm < 1) return true;
            if (thirdNorm < 1) return true;
            return true;
}
```

Тоді, подані методи можемо реалізувати наступним чином.

Метод Якобі:

```
static Matrix YakobiMethod(Matrix alpha, Matrix beta, out int iterations)
{
    iterations = 0;
    //Початкове наближення - матриця бета
    Matrix X = beta.Clone();
    Matrix X_prev = new Matrix(X.Rows, X.Columns);

    do
    {
        //Зберігаємо попереднє наближення
        X_prev = X.Clone();
        //Використовуємо рекурентну формулу
        X = alpha * X + beta;
        iterations++;
    } while (X.DistanceTo(X_prev) > 1e-3); // Допоки не досягнемо потрібної точності

    //Повертаємо результат
    return X;
}
```

Метод Зейделя:

```
static Matrix ZeidelMethod(Matrix alpha, Matrix beta, out int iterations)
{
    iterations = 0;
    //Початкове наближення - матриця бета
    Matrix X = beta.Clone();
    Matrix X_prev = new Matrix(X.Rows, X.Columns);

    do
    {
        //Зберігаємо попереднє наближення
        X_prev = X.Clone();
        //Обчислюємо Xi
        for (int i = 0; i < X.Rows; i++)
        {
            double sum = 0;
            // використовуючи вже обчислені члени поточного наближення
            for (int k = 0; k < i; k++)
```

```
                {
                    sum += X[k, 0] * alpha[i, k];
                }
                //і ще не обчислені на даній ітерації члени з попереднього наближення
                for (int j = i+1; j < X.Rows; j++)
                {
                    sum += X_prev[j, 0] * alpha[i, j];
                }
                X[i, 0] = sum + beta[i,0];
            }
            iterations++;

        } while (X.DistanceTo(X_prev) > 1e-3); // Допоки не досягнемо потрібної точності

        //Повертаємо результат
        return X;
    }
```

Тоді, з умови дістанемо розширенну матрицю системи

$$\boldsymbol{A|B} = \begin{pmatrix} \mathbf{0,91} & \mathbf{0,23} & \mathbf{-0,44} & \mathbf{-0,05} & \mathbf{2,13} \\ \mathbf{0,04} & \mathbf{0,7} & \mathbf{-0,31} & \mathbf{0,15} & \mathbf{-0,18} \\ \mathbf{0,06} & \mathbf{0,15} & \mathbf{-0,83} & \mathbf{-0,17} & \mathbf{1,44} \\ \mathbf{0,72} & \mathbf{-0,08} & \mathbf{-0,05} & \mathbf{1,08} & \mathbf{2,42} \end{pmatrix}$$

Подамо цю матрицю на вхід програми і подивимось на результат.

<u>Результат</u> виконання програми:

```
Початкова матриця:

   0,910   0,230  -0,440  -0,050   2,130
   0,040   0,700  -0,310   0,150  -0,180
   0,060   0,150  -0,830  -0,170   1,440
   0,720  -0,080  -0,050   1,080   2,420

Альфа:

   0,000  -0,253   0,484   0,055
  -0,057   0,000   0,443  -0,214
   0,072   0,181   0,000  -0,205
  -0,667   0,074   0,046   0,000

Бета:

   2,341
  -0,257
  -1,735
   2,241

Перша норма: 0,791
Друга норма: 0,973
Третя норма: 1,038

Результат методу Якобі:    1,765  -1,448  -2,046   0,862
 Ітерацій: 6
Результат методу Зейделя:    1,765  -1,449  -2,046   0,862
 Ітерацій: 7
```

Рис 5.1 Результат виконання програми.

<u>Аналіз результатів</u>:

Як бачимо, результати обчислень на комп’ютері повністю співпали з результатами обчислень, здійснених вручну. Також, співпали матриці α та β. Тому, можемо судити про достовірність як і ручних обчислень, так і програми, складеної на основі вивчених алгоритмів. К-ть ітерацій обох методів при різній точності обчислень приблизно рівна.

Висновок:

Виконуючи цю лабараторну роботу ми ознайомились з 2-ма ітеративними методами розв’язування СЛАР: методом Якобі та Зейделя. За допомогою цих знань ми реалізували програму, що розв’язала наступну систему:

$$\begin{cases} 0{,}91x_1 + 0{,}23x_2 - 0{,}44x_3 - 0{,}05x_4 = 2{,}13 \\ 0{,}04x_1 + 0{,}7x_2 - 0{,}31x_3 + 0{,}15x_4 = -0{,}18 \\ 0{,}06x_1 + 0{,}15x_2 - 0{,}83x_3 - 0{,}17x_4 = 1{,}44 \\ 0{,}72x_1 - 0{,}08x_2 - 0{,}05x_3 + 1{,}08x_4 = 2{,}42 \end{cases}$$

Перед цим, ми провели деякі обчислення вручну, і проміжні результати обчислень повністю аналогічні до результату виконання програми.

Ми отримали вектор-стовпець дійсних чисел, що відображає розв' язок цієї системи:

$$X = (1{,}765 \pm 0{,}001 \quad -1{,}448 \pm 0{,}001 \quad -2{,}046 \pm 0{,}001 \quad 0{,}862 \pm 0{,}001)^T$$